INTERNET TECHNOLOGIES AND APPLICATIONS

BCA - 306

Sixth Semester

Directorate of Distance Education
Maharshi Dayanand University, Rohtak

Internet Technologies & Applications

BCA-306

Directorate of Distance Education
Maharshi Dayanand University
ROHTAK – 124 001

Maharshi Dayanand University
ROHTAK – 124 001

Contents

INTERNET TECHNOLOGIES & APPLICATIONS

(BCA-306)

External Marks: 75
Internal Assessment: 25
Time: 3 hrs.

Note: **The examiner is requested to set 8 questions covering the whole syllabus, out of which the candidates will be required to attempt any 5 questions.**

Network layer functions and protocols: Switching; routing and congestion control; X.25; Internet protocol (IP).

Transport Layer functions and Protocols: Addressing flow control, connection management, multiplexing, transmission control, protocol (TCP) and user datagram protocol (UDP), socket & TLI interface.

Applications layer services and protocols: Domain name services network management protocol, electronic mail and file transfer protocol, world wide webs.

Survey of contemporary Internet Technologies: The Role, use and implementation of current tools. Basic TCP/IP, name, space, correctness, and protocols, Worldwide/HTML Techniques for text, images, links and forms.

Indexing methods: gopher, WAIS, Server side programming, CGI scripts, Security issues, Emphasis on understanding exploring and extending internet technologies using Java or perl.

Unit-1

Overview of Internet Technology

Learning Objectives

After reading this unit you should appreciate the following:

1.1 Introduction

1.2 Brief History of Internet

1.3 Present Scenario of Internet

1.4 Future of Internet

1.5 Internet Structure

1.6 Hardware and Software Requirements For Internet

1.7 Protocols Used For Internet

1.8 Internet Service Providers

1.9 Internet Accounts

1.10 Host and Terminals

1.11 ISDN

1.12 Home-Page

1.13 URL

1.14 Web-Browsers

1.15 Internet Explorer

1.16 Surfing the Net

1.16 Applications of Internet

1.17 Security threat on Web

1.18 Internet Authorities

1.19 Internet Authorities

1.1 Introduction

Internet is a network of networks that connects computers all over the world. The Internet is an outgrowth of a network established in the 1960s to meet the needs of the researchers working in the defense industry in the USA. That was called the ARPANET. From a handful of computers in 1971, the ARPANET or Internet grew to 10,000 computers by 1987 and to more than 100,000 by 1989. In 1990

ARPANET ceased to exist, but the Internet continued to grow: 1 million computers in 1992, 2 million in 1993. The Internet now offers both information access and a fast and inexpensive means of communication to the public.

1.2 Brief History of Internet

The Internet was not born in its present worldwide form of thousands of networks and connections. Many people have helped in the development of Internet. The initial phase in the development of Internet began way back in 1950s. In order to regain the space supremacy from USSR (which they stole form US by launching Sputnik in 1957, the US government created an agency called ARPA (Advanced Research Projects Agency), with J.C.R. Licklider as the head of the computer department. The history of Internet is chronicled below:

1960's

- In 1969 the Department of Defense Advanced Research Projects Agency (ARPA) created an experimental network called ARPANET. ARPANET stands for Advanced Research Projects Agency (ARPA) Network. The network was developed in 1969 by ARPA and funded by the Department of Defense. The network was chiefly experimental, and was used to research, develop and test networking technologies. The original network connected four host computers at four separate universities throughout the United States, enabling users to share resources and information.
- By 1972, there were 37 host computers connected to ARPANET. Also in this year, ARPA's name was changed to DARPA (Defense Advanced Research Projects Agency). In 1973, ARPANET went beyond the boundaries of the United States by making its first international connections to England and Norway.
- One goal of ARPANET was to devise a network that would still be operational if part of the network failed. The research in this area resulted in a set of networking rules, or protocols, called TCP/IP (Transmission Control Protocol/Internet Protocol).
- TCP/IP is a set of protocols that govern how data is transmitted across networks. It also enables different types of computer operating systems, such as DOS and UNIX, to share data across a network.
- ARPANET functioned as a "backbone" network - allowing smaller local networks to connect to the backbone. Once these smaller networks were connected to the backbone, they were in effect connected to each other.
- In 1983, DARPA decided that TCP/IP would be the standard set of protocols used by computers connecting to ARPANET. This meant that any smaller networks (for example, a university network) that wanted to connect to ARPANET also had to be using TCP/IP. TCP/IP was available for free and was increasingly used by networks. The spread of TCP/IP helped create the Internet as we know it today - the network of networks that either use the TCP/IP protocols, or can interact with TCP/IP networks.
- The ARPANET grew slowly from a handful of computers in 1971 to more than 1000 in 1984. Working with the ARPANET, researchers came to regard high-speed computer networks as an indispensable tool for academic research in all fields.

1970's

Networking tools were developed in the 1970's including:

- In 1972 the National Center for Supercomputing Applications (NCSA) developed the telnet application for remote login, making it easier to connect to a remote computer.
- In 1973 FTP (file transfer protocol) was introduced, standardizing the transfer of files between networked computers.

1980's

Several significant events occurred in 1983, including:

- The TCP/IP suite of networking protocols, or rules, became the only set of protocols used on the ARPANET. This decision set a standard for other networks, and generated the use of the term "Internet" as the network of networks, which either use the TCP/IP protocols or are able to interact with TCP/IP networks.
- To keep military and non-military network sites separate the ARPANET was split into two networks: ARPANET and MILNET.
- In 1982 and 1983, the first desktop computers began to appear. Many were equipped with an operating system called Berkeley UNIX, which included networking software. This allowed for relatively easy connection to the Internet using telnet.
- The personal computer revolution continued through the eighties, making access to computer resources and networked information increasingly available to the general public
- In 1986 the National Science Foundation (NSF) connected the nation's six supercomputing centers together. This network was called the NSFNET, or NSFNET backbone. To expand access to the Internet, the NSF supported the development of regional networks, which were then connected to the NSFNET backbone. In addition, the NSF supported institutions, such as universities, in their efforts to connect to the regional networks.
- 1987 - The NSF awards a grant to Merit Network, Inc. to operate and manage future development of the NSFNET backbone. Merit Network, Inc. collaborates with International Business Machines (IBM) Corporation and MCI Telecommunications Corporation to research and develop faster networking technologies.
- 1989 - The backbone network is upgraded to "T1" which means that it is able to transmit data at speeds of 1.5 million bits of data per second, or about 50 pages of text per second.

1990s

- 1990 - The ARPANET is dissolved.
- 1991 - Gopher is developed at the University of Minnesota. Gopher provides a hierarchical menu-based method for providing and locating information on the Internet. This tool makes using the Internet much easier.
- 1993 - The European Laboratory for Particle Physics in Switzerland (CERN) releases the World Wide Web (WWW), developed by Tim Berners-Lee. The WWW uses hypertext transfer

protocol (HTTP) and hypertext links, changing the way information can be organized, presented and accessed on the Internet.

- 1993 - The NSFNET backbone network is upgraded to "T3" which means that it is able to transmit data at speeds of 45 million bits of data per second, or about 1400 pages of text per second.
- 1993-1994 - The graphical web browsers Mosaic and Netscape Navigator are introduced and spread through the Internet community. Due to their intuitive nature and graphical interface, these browsers make the WWW and the Internet more appealing to the general public.
- 1995 - The NSFNET backbone is replaced by a new network architecture, called VBNS (very high speed backbone network system) that utilizes Network Service Providers, regional networks and Network Access Points (NAPs).

1.3 Present Scenario of Internet

Internet fever continues, growing almost unabated, as more and more organizations scramble to get their networks connected. The current Internet consists of more than 8000 networks literally spanning the globe. It extends to many countries on all seven continents.

The NSFNET in the United States currently has the fastest overall speeds, capable of transmitting 45 megabits per second.

The Internet community is expanding not only in numbers but in breadth of application. The Internet has always been, and will always be, a key part of the research and development community, but the increase in access and the network's potential for becoming the basis for worldwide communication between people in all walks of life cannot be ignored by the rest of us.

In short, the Internet gives you access to more people and more information faster than you can imagine.

Whether you have a PC or a Cray YMP supercomputer, a high-speed network or a regular telephone line, you can get connected to the Internet.

The Internet is a worldwide web of interconnected university, business, military, and science networks. It is a network of networks. The Internet is made up of little Local Area Networks (LANs), city-wide Metropolitan Area Networks (MANs), and huge Wide Area Networks (WANs) connecting computers for organizations all over the world. These networks are hooked together with everything from regular dialup phone lines to high-speed dedicated leased lines, satellites, microwave links, and fiber optic links.

Today there are more than 6.6 million host computers on the Internet connecting over 30 million users. The most popular use of the Internet is electronic mail (email). Other ways people use the Internet include accessing documents and programs using Gopher, transferring files between computers using the File Transfer Protocol (FTP), or doing research using Wais or Archie.

1.4 Future of Internet

It is hard to know what will happen in the future. As the Internet connects more people and starts to yield more applications, it will be used for more than just electronic mail and transferring files.

Internet engineering groups have played with connecting vending machines and household appliances such as toasters and stereos to the Internet, allowing them to be operated remotely.

Several recent experiments allowed network-engineering meetings in USA to be "virtually attended" by researchers in Australia and Europe and other parts of the United States by transmitting audio and video images of the conference. No doubt, other virtual reality applications incorporating multimedia-sound and graphics-will appear soon.

1.5 Internet Structure

No one person or organization completely controls Internet.

However, the U.S. government has a big influence on the federally funded parts of the Internet. The National Science Foundation (NSF), for example, provides funding to assist academic and research networks in getting started. NSF initiated the NSFNET, the nationwide backbone in the United States that connects these mid-level networks, which in turn connect universities and other organizations. For this reason, NSF sets policy for and operates a chunk of the Internet in the United States, but it does not have control over all the mid-level networks it connects.

The Internet Society is a nonprofit professional organization run by its members (both individuals and organizations in various communities, including academic, scientific, and engineering), dedicated to encouraging cooperation among computer networks to enable a global research communications infrastructure. The society sponsors several groups that determine the needs of the Internet and propose solutions to meet them.

People who have access to the Internet through an organization, such as a university or a large company, don't have to worry about how much they use the Internet. Their communication with people around the world and access to most information resources is not going to show up itemized on a long-distance bill, because the leased lines or network links are already paid for.

1.6 Hardware and Software Requirements For Internet

1.6.1 Hardware Requirements for Internet

Some of the basic hardware for Internet are:

1. Modem: A modem is a device or program that enables a computer to transmit data over telephone lines. Computer information is stored digitally, whereas information transmitted over telephone lines is transmitted in the form of analog waves. A modem converts between these two forms.
2. Connector: A device for mating and demating electrical power connections or communications media.
3. Cables: Cable is an insulated bundle of wires with connectors on the ends.(eg: serial cables, disk drive cables and Local Talk cables).
4. Adapter: An adapter is a physical device that allows one hardware or electronic interface to be adapted (accommodated without loss of function) to another hardware or electronic interface. In a

computer, an adapter is often built into a card that can be inserted into a slot on the computer's motherboard.

5. Circuits: A circuit is a discrete (specific) path between two or more points along which signals can be carried. Unless otherwise qualified, a circuit is a physical path, consisting of one or more wires and possibly intermediate switching points. A network is an arrangement of circuits.

6. Switches: A switch is a network device that selects a path or circuit for sending a unit of data to its next destination. A switch may also include the function of the router, a device or program that can determine the route and specifically what adjacent network point the data should be sent to. In general, a switch is a simpler and faster mechanism than a router, which requires knowledge about the network and how to determine the route.

7. Leased lines: A leased line is a telephone line that has been leased for private use. In some contexts, it's called a dedicated line. A leased line is usually contrasted with a switched line or dial-up line.

1.6.2 Transmission Media

1. Magnetic Media: One of the most common ways to transport data from one computer to another is to write them onto magnetic tape or floppy disks, physically transport the tape or disks to the destination machine, and read them back in again. Although the bandwidth characteristics of magnetic tape are excellent, the delay characteristics are poor. Transmission time is measured in minutes or hours, not millisecond. For many applications on on-line connection is needed

2. Twisted Wire: The oldest adds still most common. The oldest and still most common transmission medium is twisted pair. A twisted pair consists of two insulated copper wires, typically about 1 mm thick. The wires are twisted together in a helical form, just like a DNA molecule. The purpose of twisting the wires is to reduce electrical interference from similar pairs close by. The mist common application of the twisted pair is the telephone system. A twisted pair connects nearly all telephones to the telephone company office. Twisted pairs can run several kilometers without amplification, but for longer distances, repeaters are needed. When many twisted pairs run in parallel for a substantial distance, such as all the wires coming from an apartment building to the telephone company office, they are bundled together and encased in a protective sheath. Twisted pairs can be used for dither analog or digital transmission. The bandwidth depends on the thickness of the wire and the distance traveled. But several megabits/sec can be achieved for a few kilometers in many cases. Due to their adequate performance and low cost, twisted pairs are widely used and are likely to remain so for years to come.

   The oldest and still most common transmission medium is twisted pair. A twisted pair consists of two insulated copper wires, typically about 1 mm thick. The wires are twisted together in a helical form, just like a DNA molecule. The purpose of twisting the wires is to reduce electrical interference from similar pairs close by. The mist common application of the twisted pair is the telephone system. A twisted pair connects nearly all telephones to the telephone company office. Twisted pairs can run several kilometers without amplification, but for longer distances, repeaters are needed. When many twisted pairs run in parallel for a substantial distance, such as all the wires coming from an apartment building to the telephone company office, they are bundled together and encased in a protective sheath. Twisted pairs can be used for dither analog or digital transmission. The bandwidth depends on the thickness of the wire and the distance traveled. But several megabits/sec can be achieved for a few kilometers in many cases. Due to there adequate

performance and low cost, twisted pairs are widely used and are likely to remain [illegible] come.

3. Co-axial cables: Another common transmission medium is the coaxial cable. It has shielding [illegible] twisted pairs, so it can span longer distances at higher speeds. Two kinds of coaxial cable [illegible] widely used. One kind, 50-ohm cable, is commonly used for digital transmission and [illegible] of this section the other kind, 75-ohm cable, is commonly used for analog transmission and [illegible] described in the next section. A coaxial cable consists of a stiff copper wire as the core [illegible] by an insulating material. The insulator is encased by a cylinder conductor, often [illegible] woven braided mesh. The outer conductor is covered in a protective plastic sheath [illegible] construction and shielding of the coaxial cable give it a good combination of high bandwidth [illegible] excellent noise immunity. The bandwidth possible depends on the cable length. For [illegible] data rate of 1 to 2 gbps is feasible. Longer cables can also be used. But only at lower [illegible] with periodic amplifiers. Coaxial cables used to be widely used within the telephone system [illegible] have bow largely been replaced by fiber optics on long-haul routes.

4. Fiber optics: An optical transmission system has three components: the light source, [illegible] transmission medium, and the detector. Conventionally, a pulse of light indicates a bit 1 and [illegible] absence of light indicates a zero bit. The transmission medium is an ultra-thin fiber of glass. [illegible] detector generates an electrical pulse when light falls in it. By attaching a light source to one end [illegible] an optical fiber and a detector to the other, we have a unidirectional data transmission system [illegible] accepts an electrical signal, converts and transmits it by light pulses, and them reconverts the output [illegible] to an electrical signal at the receiving end. The tr4ansmission system would leak light and be [illegible] useless in practice except for an interesting principle of physics. When a ray is refracted [illegible] silica/air boundary. The amount of refraction depends on the properties of the two media. [illegible] angles of incidence above a certain critical value. The light is refracted back into the silica [illegible] it escapes into the air. Thus a light ray incident at or above the critical angle is trapped inside [illegible] fiber, and can propagate for many kilometers with virtually no loss.

1.6.3 Software requirement for Internet

The various software required for Internet are as follows:

- Application Server
- ASP
- C++
- DB2
- DELPHI
- DHTML
- JAVA
- ORACLE
- VISUAL BASIC
- UML

- WIRELESS
- Perl
- XML

All of these are described below.

Application Server: An application server is a server program in a computer in a distributed network that provides the business logic for an application program. The application server is frequently viewed as part of a three-tier application, consisting of a graphical user interface **(GUI)** server, an application (business logic) server, and a database and transaction server. More descriptively, it can be viewed as dividing an application into:

ASP: Application Server is a pure java full-featured application-server. ASP Active Server Pages. A Server side scripting environment that runs ActiveX scripts and ActiveX components on a server. Developers can combine scripts and components to create Web-based applications.

C++: C++ is a platform independent, high performance object oriented programming language.

DB2: DB2 is a Web-ready relational database management system. DB2 is a relational database management system (RDBMS) for large business computers that, according to IBM, leads in terms of database market share and performance.

Delphi: Delphi is a technique of obtaining the views of various experts located in different geographical locations. The views so obtained are compiled and the experts are accordingly informed. In case an expert wants to revise his views he could do so at this stage.This techniqye has the inbuilt advantage of obataining different views from different experts without the experts facing each another in one locations.

DHTML: Dynamic HTML is the changing of the style declarations of an HTML element by means of JavaScript. It is very important in web development

Java: Java is a platform independent, high performance, secure object oriented programming language, which serves as one of the best tools for internet programming.

Oracle: Oracle is ORDBMS(object Relational Data base Mgt. Sys) which provide facility of a relation data base mgt sys. Along with object- orientation concept as well as features(such as security& packages).

Visual Basic: Visual Basic (VB) is a programming environment from Microsoft in which a programmer uses a graphical user interface to choose and modify preselected sections of code written in the BASIC programming language.

UML: Unified Modeling Language helps you specify, visualize, and document models of software systems, including their structure and design, in a way that meets all of these requirements. (You can use UML for business modeling and modeling of other non-software systems too).

Wireless: Wireless means that your connection is not a wired one. It uses the WAP wireless application protocol. For example your mobile phones are wireless that is you can use them anywhere.

1.7 Protocols Used For Internet

Following are some important protocols used for Internet.

1. ***TCP/IP:*** If your network is the Internet, you will depend on a collection of protocols called the TCP/IP protocol suite, which manages all the information that moves across the Internet. The TCP/IP protocol suite consists of multiple protocols, each of which transfers data across the network in a different format and with different options (such as error checking). Depending on your program's requirements you may need to use a specific protocol within the TCP IP suite to transmit information across the Internet.

2. ***X.25:*** Many older public networks, especially outside the states, follow a standard called X.25 was developed during the 1970s by CCITT to provide an interface between public packet-switched networks and the customer. The physical layer protocol, called X.25, specifies the physical, electrical, and procedural interface the host and the network. The other layers are data link layer, network layer which deals with addressing, flow control, delivery confirmation, interrupts and related issues. This was much slower then frame relay but was reliable

3. ***Frame relay:*** Frame relay is a service for people who want an absolute bare-bones connection oriented way to move bits over A to B at reasonable speed and at low cost. Frame relay can be thought of as a virtual leased line. In addition to competing with leased lines, frame relay also competes with X.25 permanent virtual circuits, excepts that it operates at higher speeds, usually 1.5 Mpbs, and provider fewer feature.

4. ***ATM:*** Asynchronous transfer mode is a protocol. ATM networks are organized like the traditional WAN's with the lines and switches (routers). The speed intended for ATM networks are 155 Mbps and 622 Mbps, with the possibility of the Gigabit later. When ATM was proposed, virtually all the discussion was about video on demand to every home and replacing the telephone system as described above. The 155-Mbps speed was chosen because this is about what is needed to transmit high definition television. The 622Mbps speed was chosen so four 155 Mbps channel could be sent over it. It is basically used for ISDN connections. It is very reliable protocol

5. ***Sonnet (Synchronous Optical NETwork):*** Local telephone companies had to connect to multiple long-distance carriers, all with different optical TDM systems, need for standardization became obvious. In 1985, Bellcore, the RBOC's research arm, began working on a standard called SONET.

1.8 Internet Service Providers

Definition of ISP

For networks connected to the global Internet, an organization obtains network number from the communication company that supplies internet connections. Such companies are called internet service provider (ISPs).

To connect to the Internet by using a dial-up phone line, high-speed phone line, or leased line you first need to choose an **ISP**. If you connect via cable, your cable company serves as your **ISP**. If you use **WEBTV**, you can use **WEBTV** as your ISP or choose a different **ISP. ISP** is an organization or

business offering public access to the Internet. It is your gateway, to the Net. You have to subscribe to a provider for your Internet connection. You use your computer and modem to access the provider's system and the provider handles the rest of the details of connecting you to the Internet.

There are many types of Internet providers. You can, for instance, choose one of the big commercial on-line service providers. The primary business of an **ISP** is hooking people to the Internet by giving an Internet account to subscribers, and providing them with two different kinds of access: shell access and **SLIP/PPP** access. Most **ISP** offer both kinds of access, some offer both with a single account, and others require that you choose one or the other. Once you register, your provider will give you a user name (called a user ID), a password, and a phone number to dial. To establish the Internet connection, you have your communications program dial the number. You then log in using your particular user ID and password.

At present it is **VSNL** (Videsh Sanchar Nigam Limited) which is dominating the Internet scene in India through its GIAS (Gateway Internet Access Service). The other service providers in India are **MTNL** (Mahanagar Telephone Nigam Limited), Mantra-online, and Satyam-online.

Consider an organization that chooses to form a private TCP/IP internet which consists of four physical networks. The organization must purchase routers to interconnect the four networks, and then must assign IP addresses.

To begin, the organization chooses a unique prefix for each network.

When assigning a network prefix, a number must be chosen from class **A,B or C**; the choice depends on the size of the physical network. Usually networks are assigned class **C** address unless a **B** class is needed; class **A** is seldom justified because few networks contain more than 65536 hosts. For networks connected to the global Internet, a service providers makes the choice. For networks in private internet, local network administrator selects the class.

Class A – N.H.H.H

Class B- N.N.H.H

Class C- N.N.N.H

(N-network) (H-host)

Selecting an ISP

You will be amazed to find out how a service offered at a premium could in effect be cheaper, considering the add-on facilities that are offered along with the core service. Do not forget that apart from the Internet connection, the ISP gives you an international contact address, that is, your **e-mail** address. It is because of this **e-mail** address that you must be discerning while choosing your ISP. The **e-mail** address provided by the ISP would be all over your business, and it will not be easy for you to change your service provider if you wish to change your address. You will have to live with the ISP as well as the **e-mail** address. So, be judicious while making the choice, just as you would while choosing your life partner.

To choose an ISP, consider the following factors.

- Local phone numbers: Most ISP have many phone numbers that your computer can call to connect to the Internet.

- Price: In the U.S the standard price for unlimited usage of a dial up internet account for modem speeds up to 56KPBS is $20/month. The Indian price is in between Rs200-Rs700.
- Software: Many ISP provide CD-ROM with software that you can use to connect to and use the Internet.
- Support: You never know when you are going to have a problem, so your ISP's technical support phone number should be open 24 hours a day, 7 days a week.
- Speed: Most ISP's have local licensed numbers that work with 28.8 KBPS, 33.6 KBPS, and 56 KBPS modems.
- Accessibility: If the ISP's access numbers are frequently busy, you can waste a lot of time reading until you connect.

User-to-Telephone Ratio

That is, how many users are using or are expected to use one single telephone line. Ascertaining this however, is not easy as the number of subscribers is growing every day. Nevertheless, even the current user-to-line ratio will give you an idea about the standards the **ISP** has set for itself. This factor is very critical because it determines the ease of usage whether you would be able to connect to your **ISP** or not. Another way of finding this is to check out with some of the existing users as to how much time it normally takes to dial into a given **ISP.** If it takes more than 10 minutes to get through, that particular **ISP** should be avoided as it takes lot of time

Interface Simplicity

Very few organizations take into account the simplicity of the interface while opting for an ISP. This occurs to them only when they begin to use the Internet service across their organizations. The right kind of interface can lead to tremendous savings in cost. There are other problems too. How many users in an organization know about dial-up networking under Windows? How many can remember and use passwords correctly? To how many people would you like to give the password? Does terms like **TCP/IP** sound friendly to them? Questions like these determine the success of the Internet enabled organizations. There are some **ISP** to whom these questions do not apply. They provide an easy to use interface that once installed works by simply pressing a button.

Roaming Facility

Roaming means mobility of a person who moves from one place to other for his work. The roaming facility is particularly relevant for those who travel a lot. Though most **ISP** advertises this particular facility, there are not many who pay heed to it. Its benefits are realized only when one reaches another city and wants to access an urgent e-mail or the Internet. How does one connect to the Internet when one is not an **ISP** subscriber in that particular city? To overcome this problem, either you will have to use a facility like Hotmail to access your mail from around the world or use the roaming facility provided by your **ISP**. The roaming facility allows you to dial-in into the local node of your **ISP** or of the regional **ISP** that your service provider has a tie-up with. Then all you have to do is to plug in your computer to a telephone line, find out the numbers for dial-up access, and then using your password, access your original Internet account. A crucial point here is the number of cities that your ISP has presence in or has tie-ups for the same.

Multiple Login Facility

Multiple login facility means many users can log in to a single system at a time. Very few users know about this facility, mainly because it is hardly advertised. However, it can prove to be a lifesaver and a great help for small and medium business houses. If an organization has only one Internet connection, but more than one employees want to access the Net simultaneously then this would be possible only if the ISP offers to the organization the multiple login facility. In fact, this facility can even be availed of while being away from the organization. For instance, one user may be in New Delhi and the other user in Mumbai. But, with the e-mail ID it would be possible for the man away in Mumbai to simultaneously access the Internet. Some **ISP's** offer multiple e-mail IDs that allow you to segregate e-mail individually. But you have to pay extra for this.

1.9 Internet Accounts

To connect to the Internet, you can use one of several types of accounts: PPP and SLIP accounts, UNIX shell accounts, or online services.

Point-to-Point Protocol (PPP)

PPP is an Internet connection where phone lines abd a modem can be used to connect a computer to the Internet.

PPP, originally emerged as an encapsulation protocol for transporting IP traffic over point-to-point links. The Point-to-Point Protocol (PPP) provides a method for transmitting datagrams over serial point-to-point links.

PPP is comprised of three main components·

1. **A method for encapsulating datagrams over serial links.**
2. **A Link Control Protocol (LCP) for establishing, configuring,** and testing the data-link connection.
3. **A family of Network Control Protocols (NCPs) for establishing** and configuring different network-layer protocols.

SLIP (Serial Line Internet Protocol)

It is a **TCP/IP** protocol used for communication between two machines that are previously configured for communication with each other. For example, your Internet server provider may provide you with a SLIP connection so that the provider's server can respond to your requests, pass them on to the Internet, and forward your requested Internet responses back to you. Your dial-up connection to the server is typically on a slower serial line rather than on the **parallel** or multiplex lines such as a line of the network you are hooking up to.

SLIP account is an Internet account that uses the PPP or SLIP communications protocol, respectively. These are the most popular accounts, because the most popular software – are designed to work Explorer, Netscape Navigator, Eudora, and other programs – are designed to work with PPP and SLIP accounts. PPP is a more modern communications protocol than SLIP, so choose PPP if you have a choice when opening an account.

UNIX Shell Accounts

Before the advent of PPP and SLIP accounts, most Internet accounts were text-only *UNIX shell accounts,* and these accounts are still available from some ISPs. You run a *terminal-emulation program* on your PC to connect to an Internet host computer. Most Internets hosts run UNIX, a powerful but frequently confusing operating system, and you have to type UNIX commands to use and a UNIX shell account.

Online Services

An *online service* is a commercial service that enables you to connect to and access its proprietary information system. Most online services also provide an Internet connection, e-mail, the World Wide Web, and sometimes, other Internet services. Online services usually require special programs to connect to and use your account.

Three most popular online services are the following:

- ***America Online (AOL):*** The worlds most popular on line service, with a wide range of AOL only features. To connect to AOL, read AOL e-mail, browse the Web, and access other AOL services, you use AOL's proprietary program; the latest version is AOL 4.

- ***CompuServe (CIS):*** Online of the oldest online services, with an excellent selection of proprietary technical-and business-oriented discussion groups. CompuServe was purchased by America Online, so the two services may merge. CompuServe has access phone numbers in dozens of countries.

- ***Microsoft Network (MSN):*** Microsoft's online service. You connect to MSN by using Dial-Up Networking, send and receive e-mail by using outlook of outlook Express, and browse the Web by using Internet Explorer.

Dial-Up Internet Accounts

Most people connect to the Internet by using a modem and phone line to dial in to a PPP account on an Internet provider's computer. Most ISP's support modems at speeds 14.4Kbps, 28.2Kbps, and 56Kbps. You connect only when you want to use Internet services, and disconnect (hang up) when you are done.

To set Windows to connect automatically, follow these steps:

1. choose start|settings|control panel and run the internet program. Click the connection tab on the internet properties dialog box and make sure that the connect to the internet using a modem setting is selected.
2. click the settings button to display the dial-up settings dialog box .
3. in the first box, choose dial-up networking connection that you want to use when connecting automatically to the internet.
4. set the other options to tell windows how many times to try to make a connection and how long to wait between attempts(if your ISP's line is busy). Also, type your user name and password if you want windows to disconnect automatically after a period of inactivity, choose the disconnect if idle for _Minutes check box and type the number of minutes.

5. Click OK to dismiss the dial up settings dialog box and then click OK again to dismiss the internet properties dialog box.

Cable and DDSs Internet Accounts

Cable companies also offer Internet access over the same cable that brings you tv programs. You need a cable connection box and an account with a local cable company.

Digital satellite systems (DDSs), or direct broadcast satellite, let you get Internet information by satellite. Hughes direct PC is the only company to offer this service, which includes a 24-inch

1.10 Host and Terminals

Each computer connected to the Internet can be termed as terminal. For example, you might tell someone he can find the information he wants by connecting to a computer in Switzerland. If your computer is connected to the Internet, then it is also a **host**, even though you may not be sharing any resources with the rest of the world. If you connect to and log into a host, and then use its functions to reach out onto the Internet, you are using your computer as a **terminal** to reach another computer. Host connections are designed to use very simple text based interactions.

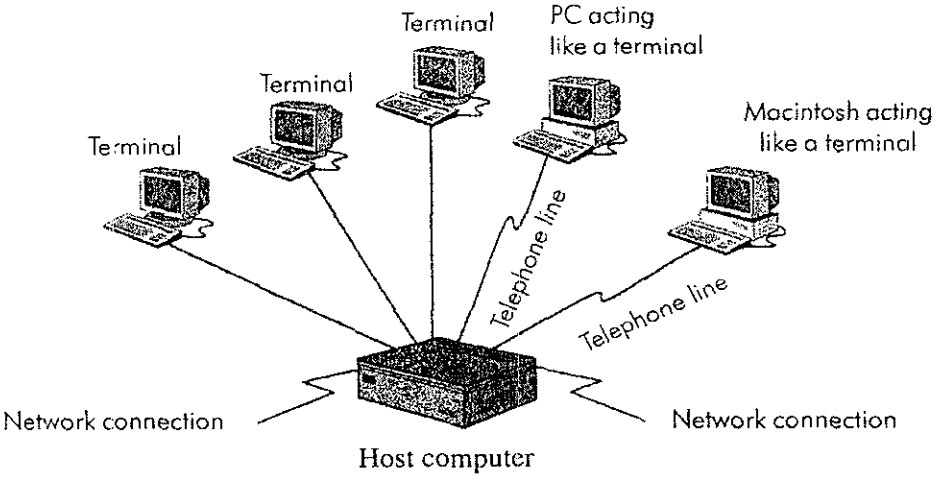


Figure : A Time sharing System

As show in the above figure the host computer is connected to different networks. The host computer is also connected to different terminals via telephone lines. They could be also connected via wireless application. In wireless application there is no wired link but they are connected to the wire less application protocol. For example **WAP** and **SMS** (Short message Services) enabled on your mollies. Being connected to the Internet means your computer system or network is actually a node on the Internet. It has an individually assigned Internet address, and client programs running on the computer system that can take full advantage of the computer's capabilities. Your workstation is a peer of every other computer on the Net. So, a node is any "addressable device" attached to a computer network.

1.11 ISDN

The first efforts to provide large-scale digital services to subscribers was launched by telephone companies under the name Integrated Services Digital Network(ISDN).

ISDN provides digitized voice and data to subscribers over conventional local loop wiring that is ISDN uses the same type of twisted pair copper wiring as the analog telephone system.

From subscriber's point of view, ISDN offers three separate digital channels, designated B B, and D(usually written 2B+D). The two B channel which each operate at a speed of 64Kbps, are intended to carry digitized voice, data, or compressed video; the D channel, which operates at 16Kbps, is intended as a control channel.

The 2B+D channels are known as the ISDN Basic Rate Interface (BRI)

Special Packages

The private **ISP's** are putting out some unique usage packages. One of them is by Mantra-on-net. It has launched a special package for night users. For those who access the Net at night, Mantra offers a dial up account, which costs almost half, compared to the regular connection. This account cannot be used during daytime. This is only the beginning as far as special packages are concerned. Soon you will find **ISP's** (especially the regional ones) coming out with packages that will fit your needs better than your cotton trousers.

Support

his is a very crucial topic and an area of service where most of the players have been found wanting. Not getting any help from the service provider and the beautifully programmed EPABX system will take you around each and every option, only to disconnect your call at the end stating: "Sorry the person handling your call is busy at the moment." In case, you happen to be using pulse-dialing equipment, you can forget using the telephone, and may as well go to their office and clear out the matter there and then.

Discounts on Renewal

Last but not the least, you must find out whether your ISP will renew your account at the same rate or whether there are any discounts to retain its old customers? This is a factor that can upset those opting for their first-buy. VSNL has been very successful in playing this card. It offers slashed rates to those subscribers who renew their accounts.

Brochure-Speak

If you can have more than a hundred different versions of the holy Ramayana, just think what the crafty marketing people can do to simple terms of the Internet. Hence, one must see through the exotic looking tariff cards of most ISP's. You must have the ability to judge beyond the gloss and the glitter. To summarize, here is what you want from an Internet Service Provider:

- Access via a local phone call
- A flat monthly fee
- An ISDN or fast (28.8 KBPS) connection
- A PPP account

- A shell account at no extra charge
- The ability to use whichever Internet clients you want
- Full Internet access to all resources
- The capability of having your own web home page
- Software support, through which you can use to connect to and use the Internet
- Technical support should be open 24 hours a day, 7 days a week
- A leased line is a telephone line that has been leased for private use. In some contexts, it's called a dedicated line. A leased line is usually contrasted with a switched line or dial-up line.

1.12 Home-Page

A home page is the front door of a Web site. Technically, the home page is just one of the pages on a Web site, but in common slang, the "home page" and the "Web site" that it introduces are used interchangeably. Each time your browser starts, it will automatically load a home page. You can turn this feature off and on, as well as specify a URL for the page to be loaded.

1.13 URL

A URL (uniform resource locator and universal resource locator are both common usage) is displayed in the Address bar, toward the top of the browser window. The URL consists of three parts: the protocol (http), the DNS. Name of the host (www.cs.vu.nl), and the file name (welcome.html), with certain punctuation's separating the pieces.

http://www.cs.vu.nl/welcome.html

The URL describes the location of the file your browser is displaying or loading. If you are viewing a page on the web, then the URL will usually begin with http://. The acronym http stands for hypertext transfer protocol. This is the format of most URL's you will be viewing. To go to a new URL, click anywhere in side the address bar, once the current URL text is highlighted, type in the URL for the site you would like to visit and hit the Enter key

URL (Uniform Resource Locator) Examples

List of schemes used within URLs:

Scheme	**Examples**
ftp	fttp://ftp.uu.net/usenet/news.answers/alt_tech/pointers.z
Gopher	Gopher://gopher.loc.gov/11/ congress
http	Hypertext resource
mail to	Mail a message to the specified address
News	Usenet newsgroup.
telnet	telnet://mantra.internet.com:1701/
WAIS	Access a WAIS database

URLs provide a standard way to specify the exact location and name of just about any Internet resource. In general, most URLs have one of two common formats:

scheme://hostname/description

scheme: description

Example 1: http://www.alan.com/alan

This example describes a particular web page on a particular computer. The URL begins with a name indicating a specific type of resource. (The name comes from the protocol used to send and receive such information: hypertext Transfer Protocol.)

Example 2: news.rec.human

This example describes a more general resource. The scheme is news, which indicates a specific discussion group.URL are used by web browsers (clients) to indicate the name of the web server from which to request information and what information to retrieve that server

1.13.1 Different types of URL

File URLs:

Suppose there is a document called "foobar.txt"; it sits on an anonymous ftp server called "ftp.yoyodyne.com" in directory "/pub/files". The URL for this file then file://ftp.yoyodyne.com/pub/files/foobar.txt

The toplevel directory of this FTP server is simply: file://ftp.yoyodyne.com/

The "pub" directory of this FTP server is then: file://ftp.yoyodyne.com/pub

Gopher URLs

Gopher URLs are a little more complicated than file URLs, since Gopher servers are a little trickier to deal with than FTP servers. To visit a particular gopher server (say, the gopher server at gopher.yoyodyne.com), use this URL: gopher://gopher.yoyodyne.com/

Some gopher servers may reside on unusual network ports on their host machines. (The default gopher port number is 70.) If you know that the gopher server on the machine "gopher.banzai.edu" is on port 1234 instead of port 70, then the corresponding URL would be:
gopher://gopher.banzai.edu:1234/

News URLs

To point to a Usenet newsgroup (say, "rec.gardening"), the URL is simply: news://rec.gardening

Currently, network clients like NCSA Mosaic don't allow you to specify a news server like you would normally expect (e.g., news://news.yoyodyne.com/rec.gardening); this may be coming down the road but in the meantime you will have to specify your local news server via some other method. The most common method is to set the environment variable NNTPSERVER to the name of your news server before you start Mosaic.

HTTP URLs

HTTP stands for HyperText Transport Protocol. HTTP servers are commonly used for serving hypertext documents, as HTTP is an extremely low-overhead protocol that capitalizes on the fact that navigation information can be embedded in such documents directly and thus the protocol itself doesn't have to support full navigation features like the FTP and Gopher protocols do.

A file called "foobar.html" on HTTP server "www.yoyodyne.com" in directory "/pub/files" corresponds to this URL: http://www.yoyodyne.com/pub/files/foobar.html The default HTTP network port is 80; if a HTTP server resides on a different network port (say, port 1234 on www.yoyodyne.com), then the URL becomes: http://www.yoyodyne.com:1234/pub/files/foobar.html

Partial URLs

Once you are viewing a document located somewhere on the network (say, the document http://www.yoyodyne.com/pub/afile.html), you can use a partial, or relative, URL to point to another file in the same directory, on the same machine, being served by the same server software. For example: If another file exists in that same directory called "anotherfile.html", then anotherfile.html is a valid partial URL at that point. This provides an easy way to build sets of hypertext documents. If a set of hypertext documents are sitting in a common directory, they can refer to one another (i.e., be hyperlinked) by just their filenames -- however a reader got to one of the documents, a jump can be made to any other document in the same directory by merely using the other document's filename as the partial URL at that point.

1.14 Web Browsers

A browser is a program that your computer runs to communicate with web servers on the Internet, which enables it to download and display the web pages that you request. Web browser understands the HTML and display text. A web browser has the ability to interpret or display many types of files.

The most popular browsers are Netscape Navigator and Microsoft Internet Explorer. Later browser is discussed in detail in succeeding sections of this unit.

1.14.1 Working of Web Browser

Web browsers consist of software that runs on your computer and displays home pages on the Web. There are clients for PC, Macintosh and UNIX computers. A Web browser displays information on your computer, by interpreting the Hypertext Markup language (HTML) that is used to build home pages on the Web. Home pages usually display graphics, sound and multimedia files as well as links to other pages, files that can be downloaded and other Internet resources.

The coding in the HTML files tells your browser how to display the text, graphics, links, and multimedia files on the home page. The HTML file that your browser loads to display the home page does not actually have the graphics, sound, multimedia files, and other resources on it. Instead, it contains HTML references to those graphics and files. Your browser uses those references to find the files on the server and then display them on the home page. The Web browser also interprets HTML tags as links to other Web sites or to other Web resources – these include graphics, multimedia files, newsgroups, or files which can be downloaded. Depending on the kind of link, it will perform different actions. For example,

if the HTML code specifies the link as another home page, the browser will retrieve the Uniform Resource Locator (URL) specified in the HTML file when the user clicks on the underlined link on the page. If the HTML code specifies a file to be downloaded, the browser will download the file to your computer.

1.14.2 Basic Features of Browsers

Following are some important features of the browsers.

1. The Web browser should be able to look at the Web pages throughout the Internet or to connect to various sites to access information, explore resources, and have fun.
2. The Web browser must enable you to follow the hyperlinks on a Web page and also to type in a URL for it to follow.
3. Another feature of browser is to have a number of other commands readily available through menus, icons, and buttons.
4. Your browser ought to include an easy way to get on-line help as well as built-in links to other resources on the Web that can give you help or answers to your questions.
5. You will definitely want a way to save links to the sites you have visited on the WWW so that you can get back to them during other sessions. Web browsers take care of those in two ways, through a history list, which keeps a record of some of the Web pages you've come across in the current session, and a bookmark list, which you use to keep a list of WWW pages you want to access any time you use your browser.
6. One of the main features of a browser is to search the information on the current page as well as search the WWW itself.
7. Browsers give you the facility to save a Web page in a file on your computer, print a Web page on your computer, and send the contents of a Web page by e-mail to others on the Internet.
8. Few Web browsers (like Netscape Communicator) are complete Internet package, means they come with components like e-mail client, newsgroup client, an HTML composer, telnet client, ftp client etc.
9. Web browser should be able to handle text, images of the World Wide Web, as well as the hyperlinks to digital video, or other types of information.
10. To take advantage of some of the most exciting things on the World Wide Web, your browser needs to properly display and handle Web pages that contain animated or interactive items. Netscape Navigator can incorporate these features through its ability to interpret programs written in Java and Java Script.
11. Web browsers interact not just with the Web, but also with your computer's operating system and with other programs, called plug-ins, that gives the browser-enhanced features.
12. Another important feature to insist on in your browser is caching. A browser that caches keep copies of the pages you visit so that it does not have to download them again if you want to return to them. Reloading a page from the cache is much quicker than downloading it again from the original source.

13. The most important feature of any browser is ease of use. While all Web browsers are fundamentally simple to use, the one you settle on should be very easy to work with; it should function as a transparent window onto the Web.
14. If you will be browsing the Web from within a secured network, you may have to configure your browser to work through a special computer on your network called a proxy server. Most popular browsers let you configure them to work with a proxy server, but some don't, so find out if you will be working through a proxy before deciding on your browser. If you are, your ISP or system administrator will tell you if you need to do anything special to use your browser.

1.15 Internet Explorer

Internet explore is a web browser which is used to open the HTML files where HTML stands for Hyper-Text Mark-up Language. We are taking IE 6 for details.

- Menu Bar: Contains menu items that open up dropdown lists for related options. Among the items are options for printing, customizing IE 6, copying and pasting text, managing Favorites, and accessing Help.
- Navigation Toolbar: Contains icons for a variety of features including navigating among Web pages, searching the Web using a selection of search tools, accessing and managing Favorites, viewing a History of visited pages, printing, and accessing email and newsgroups.

Accessing Web Resources Through Internet Explorer

1. If you have the URL (address) of a Web page

   Type the URL to go directly to the page. IE 6 gives you two ways of doing this.

   - Type the URL in the **Address** bar at the top of the screen. To accomplish this, click on the **Address** bar to highlight the current URL. Then type in the new URL and press the Enter key.
   - Click on **File/Open** at the top left of the screen. A pop-up window will appear with a text entry window. Within that window, type the URL of the file you wish to retrieve. Press the Enter key.

2. If you are on a Web page

   Click on

   - Words or images which change the shape of the mouse pointer from an arrow to a hand and display a URL on the bottom of the screen when the mouse pointer is placed over it the blue words on the display screen
   - The purple words on the display screen (the purple color indicates that the resource has been recently accessed on your terminal)

   Note: The color blue is generally the default color for text that contains a link, and purple is the default color for text representing a link that has been visited in the recent past. Nowadays, Web page creators are coloring their links in all sorts of ways. The best way to figure out which text represents a link is to position your mouse over the words and see if the pointer shape changes from an arrow to a hand. The hand represents a link.

3. If you want to use pre-installed links

   IE 6 offers a collection of Web sites in its Favorites collection. Click on **Favorites** on either the links bar or the tool bar at the top of the screen to access these resources.

Navigating Web With IE 6

IE 6 allows you to move back and forth among the Web pages that you visit during a session.

To go back to previous sites:

- Click on the small **Back** left arrow on the navigation bar near the top left corner of your screen. Each time you click on this arrow, you will return to the next previous site that you visited. If you hold your mouse over the **Back** arrow, the title of the upcoming page will briefly appear.

- To skip farther back, click on the small black triangle to the right of the word **Back**. This will bring up a list of pages you have visited. Click on any one of these choices to return to the desired page. This is the equivalent of clicking on the **Back** arrow several times.

To move forward:

- When you have returned to previous sites with the **Back** arrow, you can go forward again by clicking on the small right-pointing arrow next to the **Back** arrow. If you hold your mouse over this arrow, the title of the upcoming page will briefly appear.

- To move farther ahead, click on the small black triangle to the right of the Forward arrow on the menu bar at the top of the screen. This presents a list of several sites you have visited. Click on any of the choices to return to the desired site. This is the equivalent of clicking on the **Forward** arrow several times.

Additional Toolbar Options

Stop: The circle containing the X will stop a page while it is in the process of loading. Use this if a page is not successfully or speedily retrieving.

Refresh: The square containing the two curved arrows re-retrieves the page you are currently viewing. This is useful if the page does not load successfully or completely.

Home: The home icon takes you back to the page that was on the screen when you first started IE. You can customize your selection.

Search: The search button opens up a function that uses one or more Web search tools. You can choose the search tool(s) you want as your default.

You can also customize your search experience. After clicking on **Search**, choose the **Customize** option and make your selection. A pop-up window called "Customize Search Settings" will appear. If you choose to "Use Search Assistant" broad search topics will be displayed and the appropriate search tools will be queried. You can also opt to have IE 6 remember your last 10 searches so that you can easily repeat them.

Also notice that you can click a button called "Autosearch settings." This allows you to choose the search tool you want when you use the Address bar as a search window. You can also customize this option on the "When searching" line. You can even choose to turn off the use of the Address bar as a search window. If you do this, all words you type into the Address bar will be interpreted as URLs.

Favorites: Favorites are Web sites you have visited that you would like to store for easy access. You can add, delete and organize your Favorites.

- To *add* the current Web page as a favorite, click on **Favorites** and then **Add**. To choose the folder where you want to store this listing, click on **Create in** and choose the folder you want. At this point, you also have the option to create a new folder.
- To *delete* a Favorite, simply right click on the item and choose **Delete**. Or, you can choose **Organize Favorites** select the desired item, and click on the **Delete** button.
- To *move* a favorite to another folder, click on **Organize Favorites**, select the desired item, and click on **Move to folder**. In the pop-up window, select the folder where you would like to store this listing.

History: The history function allows you to view and select Web pages you have recently visited. You can sort your items by clicking on the black triangle to the right of the word **View**. You can sort by size, date, the number of times visited, and the order you have visited today.

Mail: You can read email from this window. Choose the email software you wish to use by going back to the Menu Bar and choosing **Tools/Internet Options/Programs**.

Print: Allows you to print the current page. This option will be explained in more detail below under Printing.

Edit: You may edit the current page in the HTML editor of your choice. Choose the editor by going back to the Menu Bar and choosing **Tools/Internet Options/Programs**.

Discuss: You may set a default Usenet newsgroup server.

Useful Options on the Menu Bar

The menu bar at the top of the screen includes some useful options. Here are a few highlights.

- File/New/Window: You can open up a second copy of IE 6 by using this feature. This allows you to visit more than one Web page at a time.
- File/Edit with...: You can edit the current Web page using the editor of your choice. Select the editor by going back to the Menu Bar and choosing **Tools/Internet Options/Programs**. You choices will be determined by software installed on your computer.
- Edit/Find (on This Page): IE 6 allows you to do a text search of the document on your screen. Choose this option and type in the word or phrase you wish to search.
- Tools/Show Related Links: IE 6 will display pages that are related in content to the current page. This is a service of Alexa, a Web content and traffic analysis company.

How to Download, Email, and Print

You can download to disk, email, or print the Web page on the IE 6 screen.

To Download

1. Click on **File/Save As** (top left of screen). A pop-up window will appear.
2. **Save in:** Choose the desired drive.
3. **Save as type:** Make sure you save the page to the file type that will be useful [illegible] save the page as a Web page, you will need a Web browser or HTML editor to [illegible] file (txt) can be viewed in a word processing program such a Word or WordPerfect
4. Click on **Save**

To Email

1. Click on **File/Send** (top left of screen).
2. You may send the current page as an email message, or you may insert the link [illegible] page within an email message. Once you make your selection, your email software [illegible] You can change the default software by going to the Menu Bar and choosing **Tools/Internet Options/Programs**. You choices will be determined by software installed on your computer.

To PRINT THE ENTIRE DOCUMENT

1. Click on the **Print** icon on the Tool Bar
2. Click on **OK**

To PRINT SELECTED PAGES

1. Click on **File/Print Preview** (top left of screen)
2. Click through the pages using the navigation arrows and make a note of which pages you want to print
3. Click on **Print** (top left of screen)
4. Click on the circle next to "Pages"
5. Type in the pages separate by commas, e.g., 1, 5-6, 7, 9
6. OR, to print the page displayed in the Print Preview window, choose **Current Page**
7. Click on **OK**

The Right Mouse Button

The right mouse button offers a number of useful features if you are using a PC [illegible] possibilities, press down on the right mouse button and hold it. Options will display in a pop up window.

The following is a selected list of right mouse button options.

1. WHEN THE MOUSE POINTER IS ON THE SCREEN (but not on a link or an image)
    - Back: Moves back to the previously visited page in your history list (same as **Back** icon)
    - Forward: Moves forward to the next page in your history list (same as **Forward** icon)
    - Select All: Selects all the text on the page for copying and pasting
    - Create Shortcut: Creates a shortcut to the current Web page on your desktop
    - Add to Favorites: Adds the current Web page to your Favorites
    - View Source: Brings up the HTML source code of the current page
    - Encoding: Allows you to choose a language
    - Print: Prints the current document
    - Refresh: Reloads the current page from the server
2. WHEN THE MOUSE POINTER IS OVER A LINK
    - Open: Opens the page
    - Open in New Window: Opens the link in a new copy of IE 6
    - Save Target As: Saves the link as a file
    - Print Target: Prints the link
    - Copy Shortcut: Copies the URL to the Clipboard for pasting into a text editor or word processing program
    - Add to Favorites: Adds the selected page to your Favorites
3. HEN THE MOUSE POINTER IS OVER AN IMAGE
    - Save Picture As: Saves the image to a disk drive of your choice
    - E-mail Picture: Opens your default email program and attaches the image to the message
    - Print Picture: Prints the image on your default printer
    - Set as Background: Uses the image as your desktop wallpaper
    - Set as Desktop Item: Sets the image as an Active Desktop item
    - Copy: Copies the image to the Clipboard for pasting into a graphics editing program
    - Add to Favorites: Adds the selected images to your Favorites

Customizing Internet Explorer

IE 6 offers a number of customization options. This section will highlight some of the more useful features available under **Tools/Internet Options** on the Menu Bar.

Tools/Internet Options is divided into six tabs. Each one is explained below.

1. General

   - Home Page: Specify the URL of the page you want to appear whenever you open IE 6 or whenever you click on the Home icon

   - Temporary Internet Files: This option allows you to view the files in your browser's cache. The cache holds viewed Web pages for subsequent quick viewing. Retrieving a file from the cache is much faster than repeated trips to the remote Web server where the file originated. You can customize the **Settings** to decide how often to check for newer pages, to specify how much disk space to reserve for your cache, and to view files in the cache.

   - History: This option customizes your access to pages you have visited with the **History** function. Here you can set the number of days to keep pages in your history.

   - Colors: Choose colors for links, visited links, and link hovers (the color appearing when your mouse is over a link). You can also set a default text and background color.

   - Fonts: Select the language script, the font displayed in Web pages, and the font displayed in plain text.

   - Languages: Select the language that will display Web pages accessed with IE 6

   - Accessibility: Choose to ignore colors, font sizes and font styles on Web pages. You can also set a style sheet as the display template for all Web pages viewed with IE 6.

2. Security

   Here you can set levels of security for individual Web pages. See IE 6's Help menus for more information.

3. Content

   - Content Advisor: You can enable ratings of objectional content to control the pages that may be viewed with this browser.

   - Certificates: This feature allows you to manage the identification certificates you may have. See the Help menus for more information.

   - Personal Information: This consists of two options. **AutoComplete** will store entered Web address, information entered into forms, and usernames and passwords needed to access sites you have visited. When you are using your browser, previous entries will come up as choices so that you don't have to retype the information. This can make your work go much faster. You can customize these options, and delete your settings. **My Profile** offers a template for entering personal information. If a Web site requests this information, you can give permission for it to be used.

3. Connections

   Here you can store the information about your Internet Service Provider, configure your LAN settings, or send your browser requests through a proxy server.

4. Programs

   Here you can set the programs you want the browser to use for HTML editing, email, Usenet news, collaboration ("Internet Call"), your calendar and contact list.

This screen offers a number of options in the categories of accessibility, browsing, HTTP settings, Microsoft VM (Virtual Machine), multimedia access, printing, searching and security. Set these options if you are comfortable with them.

1.16 Surfing the Net

Like much of the Internet, the World Wide Web operates on a client/server model. You run a Web client on your computer — called a Web browser — such as Netscape Communicator or Microsoft's Internet Explorer.

The client contacts a Web server and requests information or resources. The Web server locates and then sends the information to the Web browser, which displays the results. Web browsers are the Client software (your machine is a client to ISP's server), which have various graphics capabilities to access the information from the Internet. Modern Web browsers are capable of browsing WWW, Gopher sites, FTP sites and also provide facilities for e-mail, etc. Initially NCSA's web browser Mosaic hit the market, which actually made the browsing popular. Now Web browsers from Netscape and from Microsoft are the user's choice. You can get hold of any such browser and start browsing the Net.

Get the Latest Web Content with Channels

With channels, the latest Web content is automatically delivered to you from the world's best content providers. And if you have installed the new desktop, you can also view the channels right on your desktop or as your screen saver.

Update Your Favorite Web Sites and View them at Your Leisure

Subscribe to your favorite sites so that the content is automatically updated whenever you want—daily, weekly, or monthly. Internet Explorer can download updated Web pages or entire sites in the background while you do other work on your computer, or even while you sleep. Then you can look at them later on-line, off-line, at work, at home, or on the road.

Keep Your Most Used Web Pages Handy

Create a button on the Links toolbar just by dragging a link to it from the Address bar or a Web page. You can easily customize the Links toolbar to display buttons the way you want them. Or add a Web page to your list of favourites for easy access from the Favourites menu or Explorer bar.

Move Around the Web Faster and Easier with the Explorer Bar

Search for Web sites using the new Explorer bar. Click the Search button on the toolbar [illegible] bar appears in the left side of the browser window. Then you can click a link to view that [illegible] right side of your screen while still viewing the list of search results on the left. You [illegible] browse through your Favourites and History folders, channels, or your documents.

Let Internet Explorer Help you with Web Addresses

With AutoComplete, when you start typing a frequently used URL in the Address bar, Inter[illegible] can complete the address for you. And if a Web page [illegible] Internet Explorer can search for similar Web addresses to find a match.

1.17 Applications of Internet

Internet is an important tool for practically everybody. If you want to connect two or more [illegible] that they can communicate, you form an internetwork and Internet connects all of [illegible] applications are endless. Information required is either available on the Internet and you [illegible] the information you require.

- On the most networks, electronic mail(e-mail) is most widely used application. In fact, [illegible] TCP connections that the Internet users establish, about one-half are for sending and receiving mail messages.

- Electronic mail, which was until recently considered only an internal mechanism [illegible] becoming the most widely used application on Internet. The most common of the communication methods used by people on the Internet is the private letter, written by one individual to another (in any subject and in any language), and sent between any two connected Internet sites or through an Internet E-Mail gateway to or from a service which provides an Internet gateway.

- Internet can exchange visual information in readable and reusable formats – such as charts, figures, tables, images, databases, software code — opens up possibilities for collaboration at the global as well as local levels. With the trend this ability is not only to communicate but also to actually work with colleagues in the same field scattered all over the world makes long distance collaboration feasible.

- The resources for on-line research are multiplying at an astounding rate. Searchable databases, library holdings, alerting services, pre-prints, and other information systems are all changing the way research is done And it is not only the research community that is responsible for this. Library shelves are overflowing with journals and proceeding, and with acquisitions budgets receiving deep cuts, a likely scenario for the future is one in which libraries archive electronically, share resources and become information clearing houses instead of closets.

- Multimedia is another very important application of Internet. Live music concerts, radio broadcasts, live or recorded television shows, interactive audio and web phone, and video conferencing are [illegible] more a dream on Internet, even for a desktop PC user.

- Internet provides a variety of information to everybody ranging from entertainment to [illegible] business application to areas of daily life such as:

1. Magazines and newspapers.
2. Household shopping items.
3. Ordering novelties from anywhere in the world.
4. Radio and TV broadcast schedules and sometimes the broadcast itself.
5. Tour and travel plan guides and bookings, etc.
6. Health consultation.
7. Tips for doing various things.
8. Talking to friends and relatives in any part of the globe.
9. Games of various kinds.
10. Language interpreter.
11. On-line education course material, examination conduction, advertising on popular information sites, making payments on the net and getting an item, net banking.

1.18 Security Threat on Web

The ability to connect any computer, anywhere, to ant other computer, anywhere, is a mixed blessing. For individuals at home, wandering around the Internet is lots of fun. For corporate security managers, it is a nightmare. Most companies have large amounts of confidential information on-line, trade secrets, product development plans, marketing strategies, financial analyses, etc. Disclosure of this information to a competitor could have dire consequences.

In addition to the danger of the information leaking out, there is also a danger of information leaking in. In particular, viruses, worms, and other digital pests can breach the security, destroy the valuable data, and waste lots of time of data base administrator to clean up the mess.

Commercial and government enterprises are reluctant to use the Internet because of security concerns. During the past several years, attacks on routers have become frequent, with attackers realizing that they can create many more problems by targeting the routing infrastructure than attacking any single system. The Internet currently uses BGP (border gateway protocol) for inter domain routing. Also, because BGP session's use TCP to transmit data between routers, the recent increase in TCP based attacks is an additional threat to BGP security.

In the past, the Internet community used SNMP (simple network management protocol) to monitor the health of the network, and to debug operational problems. But now it is pretty easy to eavesdrop on SNMP traffic and discover the appropriate community name, and then access the SNMP database on the management network device. This is especially true when the management network device uses no authentication. There are also programs that guess passwords, trying millions of combinations of letters to find a match. Once an attacker gains access to the SNMP database, he can attack in many ways. If security relevant information, such as a password, is stored in the database, the attacker can learn the password.

The need to augment security is being alarmingly realized with the emergence of E-commerce. Presently, E- commerce operations are marked by fear of loss of money and privacy. One recent survey,

undertaken by Equinox and Harris Associates, determined that over two-thirds of Internet consumers considered privacy concerns to be very important.

E-mail Threats

E-mail when sent across the Internet is more like a post card. It can be intercepted at any stage and read by anybody who can lay his hands on it. To ensure the secrecy of the message, the sender as well as the receiver should agree on a secret key. There starts the problem. If your intended recipient is in a far away country, then you have to distribute the key first to him before you can send him the message. Now this presents a logistical problem. Public key cryptography was designed to overcome this problem through what is known as public key private key pair. Another way of ensuring the secrecy of e-mail messages is through the use of a technique called signing a message.

Firewall

Firewall is just a modern adaptation of that old medieval security standby: digging a deep moat around your castle. This design forced everyone entering or leaving the castle to pass over a single drawbridge where the input output police could inspect them. With the networks, the same trick is possible. A company can have many LANs connected in arbitrary ways, but all traffic to or from the company is forced through an electronic draw bridge that is firewall.

Are you planning to connect your organization to the Internet? Do your employees dial into your computers from remote places? Do you have multiple branches connected to each other? If your answer is yes, then Firewall is what you need to protect your Intranet. Firewall is typically defined as a system or group of systems that enforces an access control policy between two networks. It may also be defined as a mechanism used to protect a trusted network from an untrusted network. Firewall is a collection of components or a system placed between two networks.

Firewall acts as a gatekeeper between a company's internal network and the outside world. It acts as a electronic barrier to stop unauthorized entry.

A firewall basically performs two important functions: Gatekeeping and monitoring.

Gatekeeping by Firewall:	Firewall acts as a gatekeeper between the company's internal network and the outside network. It examines the location from which the data enters your system and then decides, based on your instructions, whether or not to allow that information.
Monitoring by Firewall:	In addition to Gatekeeping, the firewall also monitors information. Monitoring is one of the most important aspects of firewall design. Monitoring functions include logging of all system activities and generation of reports for system administration. Monitoring can be active or passive. In active monitoring, a firewall notifies a manager whenever an incident occurs. The firewall product, Smart wall, alerts the administrator via e-mail or a pager about suspicious on-line activity. In passive monitoring, a firewall logs a record of each incident in a file or a disk. Then a manager can analyze the log periodically to determine whether attempts to access the organization have increased or decreased over time.

1.19 Internet Authorities

The Internet has no single owner, yet everyone owns (a portion of) the Internet. The Internet has no central operator, yet everyone operates (a portion of) the Internet. The Internet has been compared to anarchy, but some claim that it is not nearly that well organized!

Some central authority is required for the Internet, however, to manage those things that can only be managed centrally, such as addressing, naming, protocol development, standardization, etc. Among the significant Internet authorities are:

Internet Society (ISOC)

- It was chartered in 1992 and is a non-governmental international organization providing coordination for the Internet, and its internetworking technologies and applications. ISOC also provides oversight and communications for the Internet Activities Board.

Internet Activities Board (IAB)

- IAB governs administrative and technical activities on the Internet.

Internet Engineering Task Force (IETF)

- IETF is one of the two primary bodies of the IAB. The IETF's working groups have primary responsibility for the technical activities of the Internet, including writing specifications and protocols. The impact of these specifications is significant enough that ISO accredited the IETF as an international standards body at the end of 1994. RFCs 2028 and 2031 describe the organizations involved in the IETF standards process and the relationship between the IETF and ISOC, respectively.

Internet Engineering Steering Group (IESG)

- IESG is the other body of the IAB. The IESG provides direction to the IETF.

Internet Research Task Force (IRTF)

- IRTF comprises a number of long-term reassert groups, promoting research of importance to the evolution of the future Internet.

Internet Engineering Planning Group (IEPG)

- IEPG coordinates worldwide Internet operations. This group also assists Internet Service Providers (ISPs) to interoperate within the global Internet.

Forum of Incident Response and Security Teams

- This forum is the coordinator of a number of Computer Emergency Response Teams (CERTs) representing many countries, governmental agencies, and ISPs throughout the world. Internet network security is greatly enhanced and facilitated by the FIRST member organizations.

Student Activity

1. What are the hardware and software requirements for establishing an Internet?
2. What is ISDN? Discuss its relevance.
3. What is Internet Explorer? Outline its main features.
4. What do you mean by Internet Accounts?
5. What is URL? Discuss its usefulness.
6. What do you understand by net surfing?
7. Who owns Internet?
8. What are various security threats on the web?
9. Differentiate between hosts and terminals.
10. What home-page? Outline the parameters for a good home-page.

Unit-2

OSI Reference Model and TCP/IP

Learning Objectives

After reading this unit you should appreciate the following:

2.1 ISO OSI Reference Model

2.2 Working of OSI Layers

2.3 TCP/IP Reference Model

2.4 OSI Versus TCP/IP Reference Model

2.5 Organizations For Standards

2.1 ISO OSI Reference Model

Present day modern computer networks are designed in a highly structured way. In order to reduce their design complexity, most networks are organized as a series of layers, each one built upon its predecessor.

The OSI Reference Model is based on a proposal developed by the International Organization for Standardization (ISO) in 1984. The model is called ISO OSI (Open Systems Interconnection) Reference Model because it deals with connecting open systems - that is, systems that are open for communication with other systems. The OSI reference model is a conceptual model composed of seven layers, each specifying particular network functions. This model is now considered the primary architectural model for inter-computer communications.

The OSI reference model has following seven layers:

1. Layer 7—Application
2. Layer 6—Presentation
3. Layer 5—Session
4. Layer 4—Transport
5. Layer 3—Network
6. Layer 2—Data link
7. Layer 1—Physical

An easier way to remember the seven layers is the sentence *"All people seem to need data processing."* The beginning letter of each word corresponds to a layer.

- All—Application layer
- People—Presentation layer

- Seem—Session layer
- To—Transport layer
- Need—Network layer
- Data—Data link layer
- Processing—Physical layer

2.1.1 Layering Principles

The principles that were applied to arrive at the seven layers are as follows:

- A layer should be created where a different level of abstraction is needed.
- Each layer should perform a well-defined function.
- The function of each layer should be chosen with an eye toward defining internationally standardized protocols.
- The layer boundaries should be chosen to minimize the information flow across the interfaces.
- The number of layers should be large enough that distinct functions need not be thrown together in the same layer out of necessity, and small enough that the architecture does not become

2.1.2 Classification of OSI Layers

The seven layers of the OSI reference model can be divided into two categories:

- Upper layers and
- Lower layers.

Upper Layers

The *upper layers* of the OSI model deal with application issues and generally are implemented only in software.

The highest layer, the application layer, is closest to the end user. Both users and application layer processes interact with software applications that contain a communications component.

The term upper layer is sometimes used to refer to any layer above another layer in the OSI model.

Lower Layers

The *lower layers* of the OSI model handle data transport issues. The physical layer and the data link layer are implemented in hardware and software. The lowest layer, the physical layer, is closest to the physical network medium (the network cabling, for example) and is responsible for actually placing information on the medium.

Figure 2.1 illustrates the division between the upper and lower OSI layers.

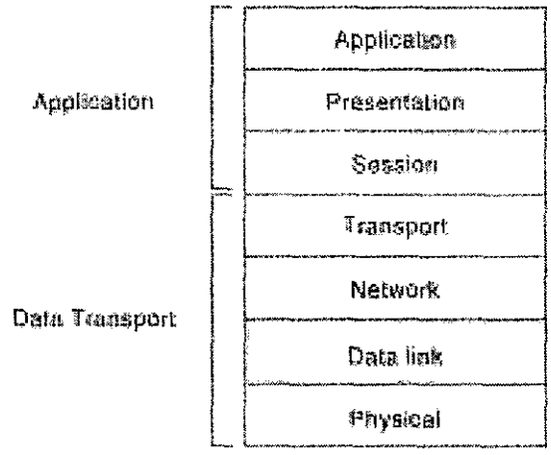


Figure 2.1: Classification of OSI Layers

2.2 Working of OSI Layers

ISO also developed a comprehensive set of standards for the various layers of the OSI model. The standards making up the OSI architecture were not widely implemented in commercial products for computer networking. However, the OSI model is still important.

The concepts and terminology associated with the OSI model have become widely accepted as a basis for discussing and describing network architectures. The OSI model is often used in categorizing the various communications protocols that are in common use today and in comparing one network architecture to another.

The OSI model defines the seven functional layers provides in Fig. 2.2. Each layer performs a different set of functions, and the intent was to make each layer as independent as possible from all others.

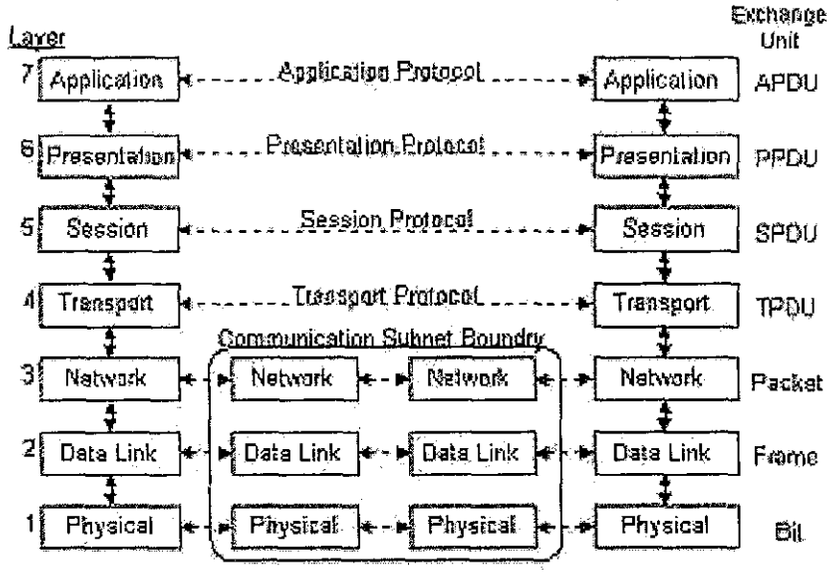


Figure 2.2: OSI Layers

Brief description of each of the seven layers of the OSI model is provided below, working from the bottom up.

2.2.1 Physical Layer

The physical layer defines the physical characteristics of the interface, such as:

- Mechanical components and connectors,
- Electrical aspects such as voltage levels representing binary values, and
- Functional aspects such as setting up, maintaining, and taking down the physical link.

Various physical media can be used for the actual transmission. Some of them are

- Magnetic media
- Twisted pair
- Baseband coaxial cable
- Fiber optics

Well-known physical layer interfaces for local area networks (LANs) include Ethernet, Token-ring and Fiber Distributed Data Interface (FDDI).

2.2.2 Data-Link Layer

The data link layer defines the rules for sending and receiving information across the physical connection between two systems. This layer encodes and frames data for transmission, in addition to providing error detection and control. Because the data link layer can provide error control, higher layers may not need to handle such services. However, when reliable media is used, there is a performance advantage by not handling error control in this layer, but in higher layers. Bridges operate at this layer in the protocol stack.

The data link layer provides reliable transit of data across a physical network link. Different data link layer specifications define different network and protocol characteristics, including the following:

- Physical addressing -- Physical addressing (as opposed to network addressing) defines how devices are addressed at the data link layer.
- Network topology -- Data link layer specifications often define how devices are to be physically connected (such as in a bus or a ring topology).
- Error notification -- Error notification involves alerting upper layer protocols that a transmission error has occurred.
- Sequencing of frames -- Sequencing of data frames involves the reordering of frames that are transmitted out of sequence.
- Flow control -- Flow control involves moderating the transmission of data so that the receiving device is not overwhelmed with more traffic than it can handle at one time.

The Institute of Electrical and Electronics Engineers (IEEE) has subdivided the data link layer into two sub-layers:

- Logical Link Control (LLC) and
- Media Access Control (MAC).

2.2.3 Network Layer

The network layer defines protocols for opening and maintaining a path on the network between systems. It is concerned with data transmission and switching procedures, and hides such procedures from upper layers.

Routers operate at the network layer. The network layer can look at packet addresses to determine routing methods. If a packet is addressed to a workstation on the local network, it is sent directly there. If it is addressed to a network on another segment, the packet is sent to a routing device, which forwards it on the network.

2.2.4 Transport Layer

The transport layer provides a high level of control for moving information between systems, including more sophisticated error handling, prioritization, and security features.

The transport layer provides quality service and accurate delivery by providing connection oriented services between two end systems. It controls the sequence of packets, regulates traffic flow, and recognizes duplicate packets.

The transport layer assigns packetized information a traffic number that is checked at the destination. If data is missing from the packet, the transport layer protocol at the receiving end arranges with the transport layer of the sending system to have packets re-transmitted.
This layer ensures that all data is received and in the proper order.

2.2.5 Session Layer

The session layer coordinates the exchange of information between systems by using conversational techniques, or dialogues.

Dialogues are not always required, but some applications may require a way of knowing where to restart the transmission of data if a connection is temporarily lost, or may require a periodic dialog to indicate the end of one data set and the start of a new one.

The session layer establishes, manages, and terminates communication sessions between presentation layer entities. Communication sessions consist of service requests and service responses that occur between applications located in different network devices. These requests and responses are coordinated by protocols implemented at the session layer.

This layer provides application management. It allows remote users to establish multiple connections to remote devices. Token management occurs here and sockets are also used here.

2.2.6 Presentation Layer

The presentation layer presents data to the application layer. The OSI has protocol standards that define how standard data should be formatted. Task like data compression, decompression, encryption, and decryption are associated with this layer.

The presentation layer provides a variety of coding and conversion functions that are applied to application layer data. These functions ensure that information sent from the application layer of one system will be readable by the application layer of another system.

Protocols at the presentation layer are part of the operating system and application the user runs in workstation. Information is formatted for display or printing in this layer. Codes within the data, such as tabs or special graphics sequences, are interpreted.

Some examples of presentation layer coding and conversion schemes follow:

- Common data representation formats -- The use of standard image, sound, and video format allow the interchange of application data between different types of computer systems
- Conversion of character representation formats -- Conversion schemes are used to exchange information with systems using different text and data representations (such as EBCDIC and ASCII).
- Common data compression schemes -- The use of standard data compression schemes allows data that is compressed at the source device to be properly decompressed at the destination.
- Common data encryption schemes -- The use of standard data encryption schemes allows data encrypted at the source device to be properly unencrypted at the destination.

Presentation layer implementations are not typically associated with a particular protocol stack. Some well-known standards follow:

- Data: ASCII, EBCDIC, Encryption
- Visual Imaging: PICT, TIFF, GIF, JPEG
- Video: MIDI, MPEG, QuickTime

2.2.7 Application Layer

Applications access the underlying network services using defined procedures in this layer.

The application layer is used to define a range of applications that handle file transfers, terminal sessions, network management, and message exchange.

Supports the components that deal with communicating aspects of an application. It is responsible for identifying and establishing the availability of the intended communication partner

The application layer interacts with software applications that implement a communicating component.

Application layer functions typically include the following:

- Identifying communication partners -- The application layer identifies and determines the availability of communication partners for an application with data to transmit.
- Determining resource availability -- The application layer must determine whether sufficient network resources for the requested communication are available.
- Synchronizing communication -- Communication between applications requires cooperation that is managed by the application layer.

The application layer is the OSI layer closest to the end user. That is, both the OSI application layer and the user interact directly with the software application.

2.3 TCP/IP Reference Model

The historical and technical standard of the Internet is the TCP/IP model. The U.S. Department of Defense (DoD) created the TCP/IP reference model, because it wanted to design a network that could survive any conditions, including a nuclear war.

In a world connected by different types of communication media such as copper wires, microwaves, optical fibers and satellite links, the DoD wanted transmission of packets every time and under any conditions. This very difficult design problem brought about the creation of the TCP/IP model.

Unlike the proprietary networking technologies mentioned earlier, TCP/IP was developed as an open standard. This meant that anyone was free to use TCP/IP. This helped speed up the development of TCP/IP as a standard.

The TCP/IP model has the following four layers (see Figure 2.3):

- Application layer
- Transport layer
- Internet layer
- Network access layer

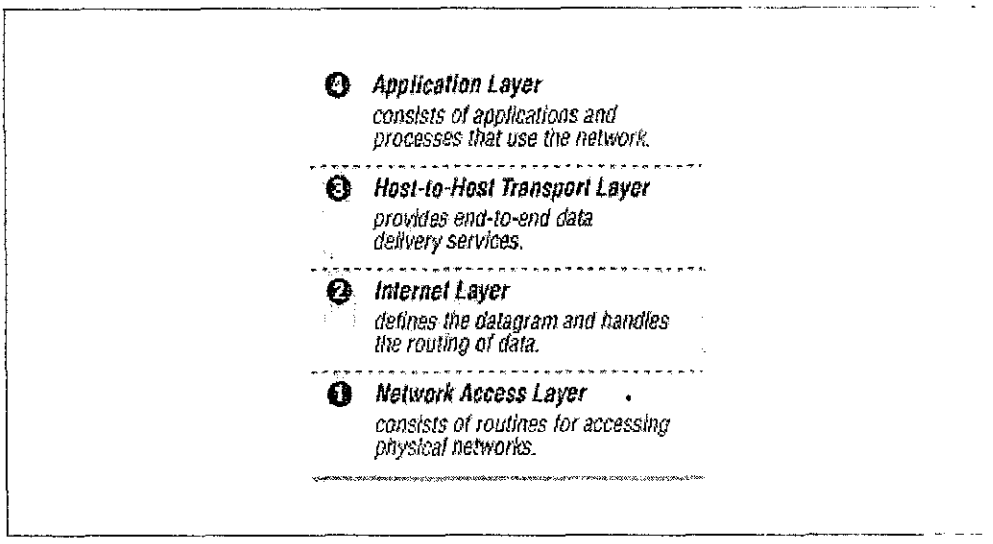


Figure 2.3: TCP/IP Layers

Although some of the layers in the TCP/IP model have the same name as layers in the OSI model, the layers of the two models do not correspond exactly. Most notably, the application layer has different functions in each model.

The designers of TCP/IP felt that the application layer should include the OSI session and presentation layer details. They created an application layer that handles issues of representation, encoding, and dialog control.

The transport layer deals with the quality of service issues of reliability, flow control and error correction. One of its protocols, the transmission control protocol (TCP), provides excellent and flexible ways to create reliable, well-flowing, low-error network communications.

TCP is a connection-oriented protocol. It maintains a dialogue between source and destination while packaging application layer information into units called segments. Connection-oriented does not mean that a circuit exists between the communicating computers. It does mean that Layer 4 segments travel back and forth between two hosts to acknowledge the connection exists logically for some period.

The purpose of the Internet layer is to divide TCP segments into packets and send them from any network. The packets arrive at the destination network independent of the path they took to get there. The specific protocol that governs this layer is called the Internet Protocol (IP). Best path determination and packet switching occur at this layer.

The relationship between IP and TCP is an important one. IP can be thought to point the way for the packets, while TCP provides a reliable transport.

The name of the network access layer is very broad and somewhat confusing. It is also called the host-to-network layer. This layer is concerned with all of the components, both physical and logical, that are required to make a physical link. It includes the networking technology details, including all of the details in the OSI physical and data link layers.

There are some of the common protocols specified by the TCP/IP reference model layers. The most commonly used application layer protocols include the following:

- File Transfer Protocol (FTP)

- Hypertext Transfer Protocol (HTTP)
- Simple Mail Transfer Protocol (SMTP)
- Domain Name System (DNS)
- Trivial File Transfer Protocol (TFTP)

The common transport layer protocols include:

- Transport Control Protocol (TCP)
- User Datagram Protocol (UDP)

The primary protocol of the Internet layer is:

- Internet Protocol (IP)

The network access layer refers to any particular technology used on a specific network.

Regardless of which network application services are provided and which transport protocol is used, there is only one Internet protocol, IP. This is a deliberate design decision. IP serves as a universal protocol that allows any computer anywhere to communicate at any time.

2.4 OSI Versus TCP/IP Reference Model

A comparison of the OSI model and the TCP/IP models will point out some similarities and differences.

Similarities

OSI and TCP/IP reference model have the following major similarities:

- Both have layers.
- Both have application layers, though they include very different services.
- Both have comparable transport and network layers.
- Both models need to be known by networking professionals.
- Both assume packets are switched. This means that individual packets may take different paths to reach the same destination. This is contrasted with circuit-switched networks where all the packets take the same path.

Differences

OSI and TCP/IP reference model have the following main differences:

- TCP/IP combines the presentation and session layer issues into its application layer.
- TCP/IP combines the OSI data link and physical layers into the network access layer.
- TCP/IP appears simpler because it has fewer layers.

- TCP/IP protocols are the standards around which the Internet developed, so the TCP/IP model gains credibility just because of its protocols. In contrast, networks are not usually built on the OSI protocol, even though the OSI model is used as a guide.

2.5 Organizations For Standards

A wide variety of organizations contribute to internetworking standards by providing forums for discussion, turning informal discussion into formal specifications, and proliferating specifications after they are standardized.

Most standards organizations create formal standards by using specific processes:

- Organizing ideas,
- Discussing the approach,
- Developing draft standards,
- Voting on all or certain aspects of the standards, and then
- Formally releasing the completed standard to the public.

Some of the best-known standards organizations that contribute to internetworking standards include these:

International Organization for Standardization (ISO)

ISO is an international standards organization responsible for a wide range of standards, including many that are relevant to networking. Its best-known contribution is the development of the OSI reference model and the OSI protocol suite.

American National Standards Institute (ANSI)

ANSI, which is also a member of the ISO, is the coordinating body for voluntary standards groups within the United States. ANSI developed the Fiber Distributed Data Interface (FDDI) and other communications standards.

Electronic Industries Association (EIA)

EIA specifies electrical transmission standards, including those used in networking. The EIA developed the widely used EIA/TIA-232 standard (formerly known as RS-232).

Institute of Electrical and Electronic Engineers (IEEE)

IEEE is a professional organization that defines networking and other standards. The IEEE developed the widely used LAN standards IEEE 802.3 and IEEE 802.5.

International Telecommunication Union Telecommunication Standardization (ITU-T)

Formerly called the Committee for International Telegraph and Telephone (CCITT), ITU-T is now an international organization that develops communication standards. The ITU-T developed X.25 and other communications standards.

Internet Activities Board (IAB)

IAB is a group of internetwork researchers who discuss issues pertinent to the Internet and set Internet policies through decisions and task forces. The IAB designates some Request For Comments (RFC) documents as Internet standards, including Transmission Control Protocol/Internet Protocol (TCP/IP) and the Simple Network Management Protocol (SNMP).

Student Activity

1. What is ISO OSI Reference model?
2. Why layering is needed in OSI model?
3. Outline the functions of OSI layers.
4. What is TCP/IP model?
5. Differentiate between TCP/IP and OSO Reference Models.
6. Outline the commonly used application layer protocols.
7. What are transport layer protocols?
8. Outline the various protocols in TCP/IP.
9. Which layers are common in both OSI and TCP/IP models?
10. What are various standardization bodies?

Unit 3

Network and Transport Layers: Functions and Protocols

Learning Objectives

After reading this unit you should appreciate the following:

3.1 Network Layer Functions

The role of the network layer is to transport packets from a sending host to a receiving host. In order to do so, three important network layer functions can be identified:

- Path determination
- Switching
- Call Setup

These are discussed below:

Path Determination

- The network layer must determine the route or path taken by packets as they flow from a sender to a receiver. The algorithms that calculate these paths are referred to as *routing algorithms.*
- A routing algorithm would determine the path.

Switching

- When a packet arrives at the input to a router, the router must move it to the appropriate output link.

Call Setup

A "3-way handshake" is required before data actually flows from sender to receiver. This allows the sender and receiver to setup the needed state information (e.g., sequence number and initial flow control window size).

In an analogous manner, some network layer architectures (e.g., ATM) requires that the routers along the chosen path from source to destination handshake with each other in order to setup state before data actually begins to flow.

In the network layer, this process is referred to as **call setup**. The network layer of the Internet architecture does not perform any such call setup.

Before delving into the details of the theory and implementation of the network layer, however, let us first take the broader view and consider what different types of *service* might be offered by the network layer.

3.2 Network Services

The network service model defines the characteristics of end-to-end transport of data between one "edge" of the network and the other, i.e., between sending and receiving end systems.

When the transport layer at a sending host transmits a packet into the network (i.e., passes it down to the network layer at the sending host),

- Can the transport layer count on the network layer to deliver the packet to the destination?
- When multiple packets are sent, will they be delivered to the transport layer in the receiving host in the order in which they were sent?
- Will the amount of time between the sending of two sequential packet transmissions be the same as the amount of time between their receptions?
- Will the network provide any feedback about congestion in the network?
- What is the abstract view (properties) of the channel connecting the transport layer in the two hosts?

The answers to these questions and others are determined by the *service model* provided by the network layer.

3.3 Working of Network Layer

The network layer is concerned with controlling the operation of the subnet. A key design issue is determining how packets are routed from source to destination. Routes could be based on static tables that are "wired into" the network and rarely changed. They could also be determined at the start of each conversation, for example a terminal session. Finally, they could be highly dynamic, being determined anew for each packet, to reflect the current network load.

If too many packets are present in the subnet at the same time, they will get in each other's way, forming bottlenecks. The control of such congestion also belongs to the network layer.

Since the operators of the subnet may well expect remuneration for their efforts, there is often some accounting function built into the network layer. At the very least, the software must count how much packets or characters or bits are sent by each customer, to produce billing information. When a packet crosses a national border, with different rates on each side, the accounting can become complicated.

When a packet has to travel from one network to another to get to its destination, many problems can arise. The addressing used by the second network may be different from the first one. The second one may not accept the packet at all because it is too large. The protocols may differ, and so on. It is up to the network layer to overcome all these problems to allow heterogeneous networks to be interconnected.

In broadcast networks, the routing problem is simple, so the network layer is often thin or even nonexistent.

Example: X.25 Connection Establishment

X.25 layer 3 manages connections between a pair of DTEs. Two forms of connection are provided: virtual calls and permanent virtual circuits. A virtual call is like an ordinary telephone call: a connection is established, data is transferred, and then the connection is released. In contrast, a permanent virtual circuit is like a leased line. It is always present, and the DTE at either end can just send data whenever it wants to, without any setup. Permanent virtual circuits are normally used in situations with a high volume of data.

Connections (virtual calls) are made as follows. When a DTE wants to communicate with another DTE, it must first set up a connection. To do this, the DTE builds a CALL REQUEST packet and passes it to its DCE. The subnet then delivers the packet to the destination DCE, which then gives it to the destination DTE. If the destination DTE wishes to accept the call, it sends a CALL ACCEPTED packet back. When the originating DTE receives the CALL ACCEPTED packet, the virtual circuit is established.

At this point both DTEs may use the full-duplex connection to exchange data packets. When either side has had enough, it sends a CLEAR REQUEST packet to the other side, which then sends a CLEAR CONFIRMATION packet back as an acknowledgment.

3.4 Transport Layer Services and Functions

The transport layer offers the following connection-oriented services:

1. Transport-connection establishment
2. Data transfer
3. Transport-connection release

Transport layer functions may include:

- Mapping transport-addresses onto network-addresses.
- Multiplexing transport-connections onto network-connections.
- Establishment and release of transport-connections.
- End-to-end sequence control on individual connections.
- End-to-end error detection and any necessary monitoring of quality of service.
- End-to-end error recovery
- End-to-end segmenting, blocking and concatenation.
- End-to-end flow control on individual connections.
- Supervisory functions.
- Expedited transport-service-data-unit transfer.

3.5 TCP/IP Protocols Classification

TCP/IP includes a wide range of protocols, which are used for a variety of purposes on the network. The set of protocols that are a part of TCP/IP is called the TCP/IP protocol stack or the TCP/IP suite of protocols.

Considering the many protocols, message types, levels, and services that TCP/IP networking supports, it would be very helpful to categorize the various protocols that support TCP/IP networking and define their respective contribution to the operation of networking.

As mentioned previously, there are four TCP/IP layers. They are link, network, transport, and application. The link layer is the hardware layer that provides ability to send messages between multiple locations. The criteria for this classification includes essential, critical, important, advanced, useful.

1. Essential - Without this all other categories are irrelevant.
2. Critical - The network, as designed, is useless without this ability.
3. Important - The network could function, but would be difficult to use and manage.
4. Advanced - Includes enhancements that make the network easier to use and manage.
5. Useful - Functionality that you would like to be able to use as a network user. Applications or some functionality is supported here. Without this, why build a network?

The categorisation is shown in Table 3.1.

Table 3.1: TCP/IP Suite of Protocols

Name (layer)	Importance	Names of protocols	What it does
Hardware(link)	Essential	Ethernet, SLIP, PPP, Token Ring, ARCnet	Allows messages to be packaged and sent between physical locations.
Package management(network)	Essential	IP, ICMP	Manages movement of messages and reports errors. It uses message protocols and software to manage this process. (includes routing)
Inter layer communication	Essential	ARP	Communicates between layers to allow one layer to get information to support another layer. This includes broadcasting
Service control(transport)	Critical	TCP, UDP	Controls the management of service between computers. Based on values in TCP and UDP messages a server knows what service is being requested
Application and user support	Important	DNS, RPC	DNS provides address to name translation for locations and network cards. RPC allows remote computer to perform functions on other computers
Network Management	Advanced	RARP, BOOTP, DHCP, IGMP, SNMP,RIP, OSPF, BGP, CIDR	Enhances network management and increases functionality
Utility(Application)	Useful	FTP, TFTP, SMTP, Telnet, NFS, ping, Rlogin	Provides direct services to the user

There are exceptions to this categorizations that don't fit into the normal layering scheme, such as ICMP is normally part of the link layer, but it has been tried to list these categorizations according to network functions and their relative importance to the operation of the network. Also note that Ethernet, which is not really a protocol, but an IEEE standard along with PPP, SLIP, TokenRing, and ArcNet are not TCP/IP protocols but may support TCP/IP at the hardware or link layer, depending on the network topology.

The list below gives a brief description of each protocol:

- Ethernet - Provides for transport of information between physical locations on Ethernet cable. Data is passed in Ethernet packets
- SLIP - Serial line IP (SLIP), a form of data encapsulation for serial lines.
- PPP - Point to point protocol (PPP). A form of serial line data encapsulation that is an improvement over SLIP.

- IP - Internet Protocol (IP). Except for ARP and RARP all protocols' data packets will be packaged into an IP data packet. Provides the mechanism to use software to address and manage data packets being sent to computers.
- ICMP - Internet control message protocol (ICMP) provides management and error reporting to help manage the process of sending data between computers.
- ARP - Address resolution protocol (ARP) enables the packaging of IP data into Ethernet packages. It is the system and messaging protocol that is used to find the Ethernet (hardware) address from a specific IP number. Without this protocol, the Ethernet package could not be generated from the IP package, because the Ethernet address could not be determined.
- TCP - A reliable connection oriented protocol used to control the management of application level services between computers.
- UDP - An unreliable connection less protocol used to control the management of application level services between computers.
- DNS - Domain Name Service, allows the network to determine IP addresses from names and vice versa.
- RARP - Reverse address resolution protocol (RARP) is used to allow a computer without a local permanent data storage media to determine its IP address from its ethernet address.
- BOOTP - Bootstrap protocol is used to assign an IP address to diskless computers and tell it what server and file to load which will provide it with an operating system.
- DHCP - Dynamic host configuration protocol (DHCP) is a method of assigning and controlling the IP addresses of computers on a given network. It is a server based service that automatically assigns IP numbers when a computer boots. This way the IP address of a computer does not need to be assigned manually. This makes changing networks easier to manage. DHCP can perform all the functions of BOOTP.
- IGMP - Internet Group Management Protocol used to support multicasting.
- SNMP - Simple Network Management Protocol (SNMP). Used to manage all types of network elements based on various data sent and received.
- RIP - Routing Information Protocol (RIP), used to dynamically update router tables on WANs or the internet.
- OSPF - Open Shortest Path First (OSPF) dynamic routing protocol.
- BGP - Border Gateway Protocol (BGP). A dynamic router protocol to communicate between routers on different systems.
- CIDR - Classless Interdomain Routing (CIDR).
- FTP - File Transfer Protocol (FTP). Allows file transfer between two computers with login required.
- TFTP - Trivial File Transfer Protocol (TFTP). Allows file transfer between two computers with no login required. It is limited, and is intended for diskless stations.
- SMTP - Simple Mail Transfer Protocol (SMTP).

- NFS - Network File System (NFS). A protocol that allows UNIX and Linux systems to remotely mount each other's file systems.
- Telnet - A method of opening a user session on a remote host.
- Ping - A program that uses ICMP to send diagnostic messages to other computers to verify that they are reachable over the network.
- Rlogin - Remote login between UNIX hosts. This is outdated and is replaced by Telnet

Each protocol ultimately has its data packets wrapped in an Ethernet, SLIP, or PPP packet (at the link level) in order to be sent over the Ethernet cable. Some protocol data packets are wrapped sequentially multiple times before being sent. For example FTP data is wrapped in a TCP packet, which is wrapped in a IP packet which is wrapped in a link packet (normally Ethernet). Figure 3.1 shows the relationship between the protocols' sequential wrapping of data packets.

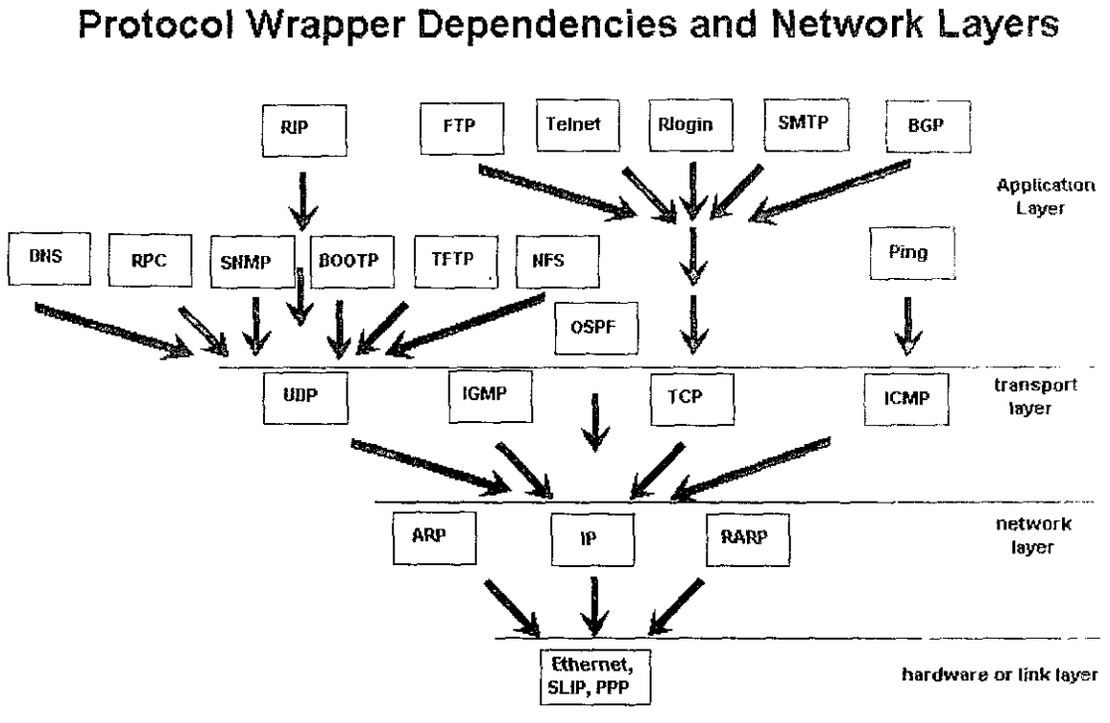


Figure 3.1: Relationship Between Protocols

3.6 X.25 Network

X.25 is an International Telecommunication Union-Telecommunication Standardization Sector (ITU-T) protocol standard for WAN communications that defines how connections between user devices and network devices are established and maintained.

X.25 is designed to operate effectively regardless of the type of systems connected to the network. It is typically used in the packet-switched networks of common carriers, such as the telephone companies. Subscribers are charged based on their use of the network.

The development of the X.25 standard was initiated by the common carriers in the 1970s. At that time, there was a need for WAN protocols capable of providing connectivity across public data networks. X.25 is now administered as an international standard by the ITU-T.

3.6.1 Types of X.25 Network Devices and Protocol Operation

X.25 network devices fall into three general categories:

- Data terminal equipment (DTE),
- Data circuit-terminating equipment (DCE), and
- Packet-switching exchange (PSE).

DTE Devices

- Data terminal equipment (DTE) devices are end systems that communicate across the X.25 network.
- They are usually terminals, personal computers, or network hosts, and are located on the premises of individual subscribers.

DCE Devices

- DCE devices are communications devices, such as modems and packet switches, which provide the interface between DTE devices and a PSE, and are generally located in the carrier's facilities.

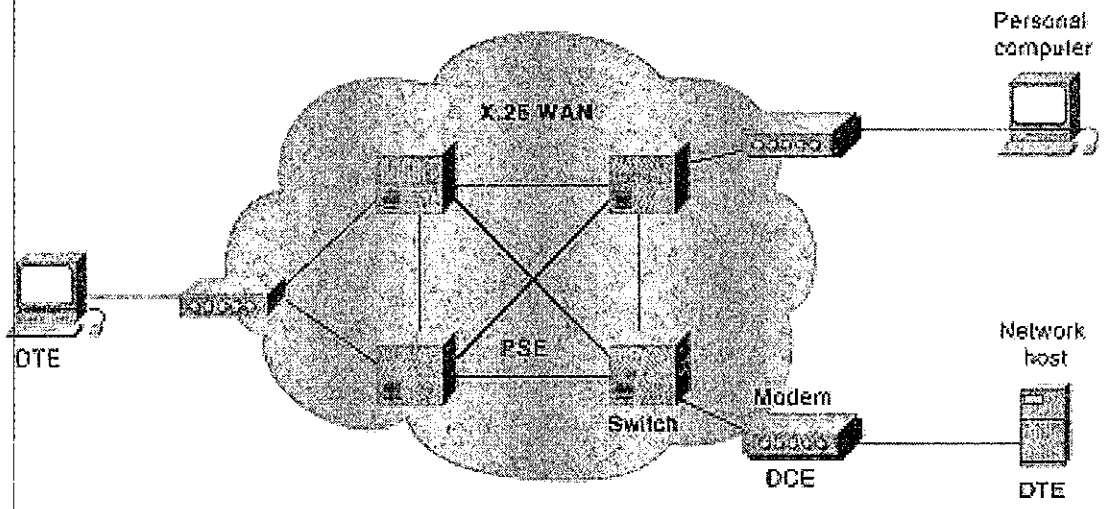


Figure 3.2: X.25 Network

PSE Devices

- PSEs are switches that compose the bulk of the carrier's network.
- They transfer data from one DTE device to another through the X.25 PSN.

Figure 3.2 illustrates the relationships among the three types of X.25 network devices.

3.6.2 Packet Assembler/Disassembler (PAD)

PAD is a device commonly found in X.25 networks. PADs are used when a DTE device, such as a character-mode terminal, is too simple to implement the full X.25 functionality. The PAD is located between a DTE device and a DCE device, and it performs three primary functions:

- Buffering (storing data until a device is ready to process it),
- Packet assembly, and
- Packet disassembly.

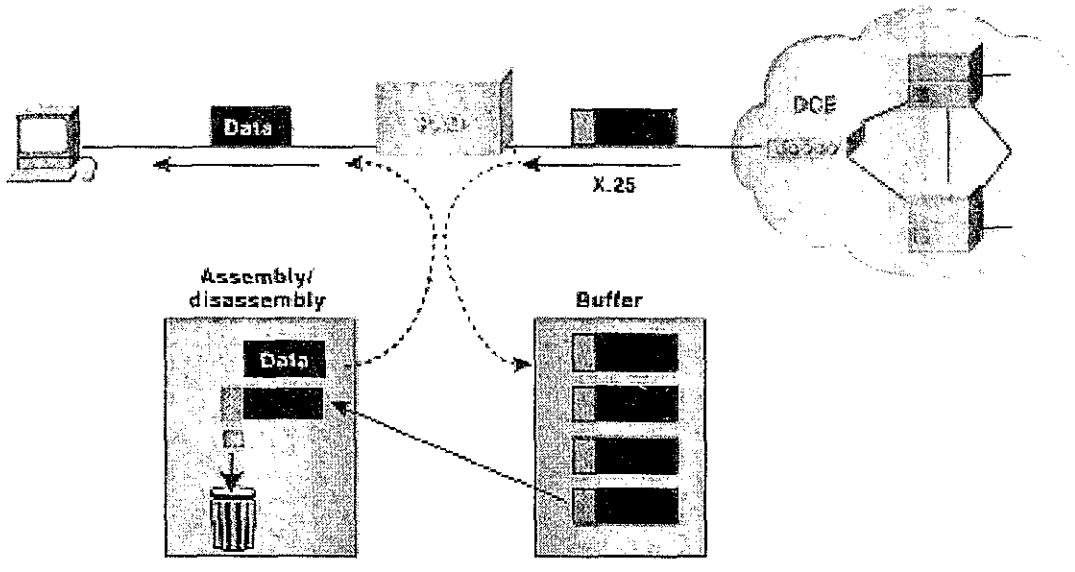


Figure 3.3: Functions of PAD

Figure 3.3 illustrates the basic operation of the PAD when receiving packets from the X.25 WAN.

The PAD buffers data sent to or from the DTE device. It also assembles outgoing data into packets and forwards them to the DCE device. This includes adding an X.25 header.

Finally, the PAD disassembles incoming packets before forwarding the data to the DTE. This includes removing the X.25 header.

3.6.3 X.25 Session Establishment

X.25 sessions are established when one DTE device contacts another to request a communication session. The DTE device that receives the request can either accept or refuse the connection.

If the request is accepted, the two systems begin full-duplex information transfer. Either DTE device can terminate the connection. After the session is terminated, any further communication requires the establishment of a new session.

3.6.4 X.25 Virtual Circuits and Its Types

A *virtual circuit* is a logical connection created to ensure reliable communication between two network devices. A virtual circuit denotes the existence of a logical, bidirectional path from one DTE device to another across an X.25 network. Physically, the connection can pass through any number of intermediate nodes, such as DCE devices and PSEs.

Multiple virtual circuits can be multiplexed onto a single physical circuit (a physical connection). Virtual circuits are demultiplexed at the remote end, and data is sent to the appropriate destinations. Figure 3.4 illustrates four separate virtual circuits being multiplexed onto a single physical circuit.

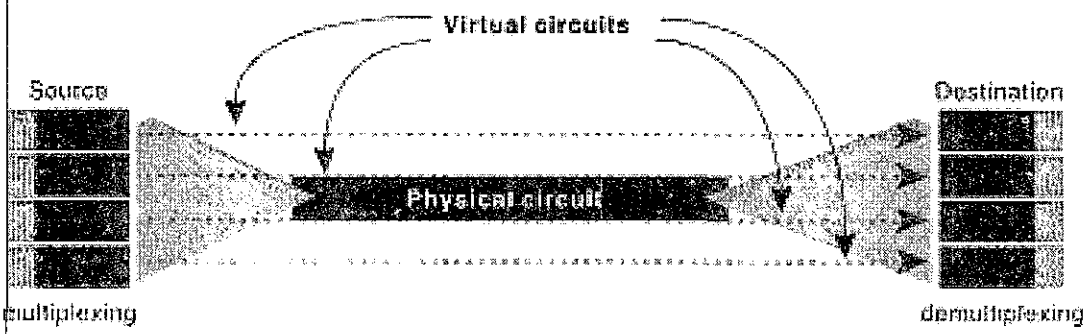


Figure 3.4: Virtual Circuits Can Be Multiplexed onto a Single Physical Circuit

Two types of X.25 virtual circuits exist:

- Switched virtual circuits and
- Permanent virtual circuits.

Switched virtual circuits(SVCs)

SVCs are temporary connections used for sporadic data transfers. They require that two DTE devices establish, maintain, and terminate a session each time the devices need to communicate.

Permanent virtual circuits (PVCs)

PVCs are permanently established connections used for frequent and consistent data transfers. PVCs do not require that sessions be established and terminated. Therefore, DTEs can begin transferring data whenever necessary because the session is always active.

The basic operation of an X.25 virtual circuit begins when the source DTE device specifies the virtual circuit to be used (in the packet headers) and then sends the packets to a locally connected DCE device. At this point, the local DCE device examines the packet headers to determine which virtual circuit to use and then sends the packets to the closest PSE in the path of that virtual circuit. PSEs (switches) pass the traffic to the next intermediate node in the path, which may be another switch or the remote DCE device.

When the traffic arrives at the remote DCE device, the packet headers are examined and the destination address is determined. The packets are then sent to the destination DTE device. If communication occurs over an SVC and neither device has additional data to transfer, the virtual circuit is terminated.

3.7 X.25 Protocol Suite

The X.25 protocol suite maps to the lowest three layers of the OSI reference model. The following protocols are typically used in X.25 implementations:

- Packet-Layer Protocol (PLP),
- Link Access Procedure, Balanced (LAPB), and
- Those among other physical-layer serial interfaces (such as EIA/TIA-232, EIA/TIA-449, EIA-530, and G.703).

Figure 3.5 maps the key X.25 protocols to the layers of the OSI reference model

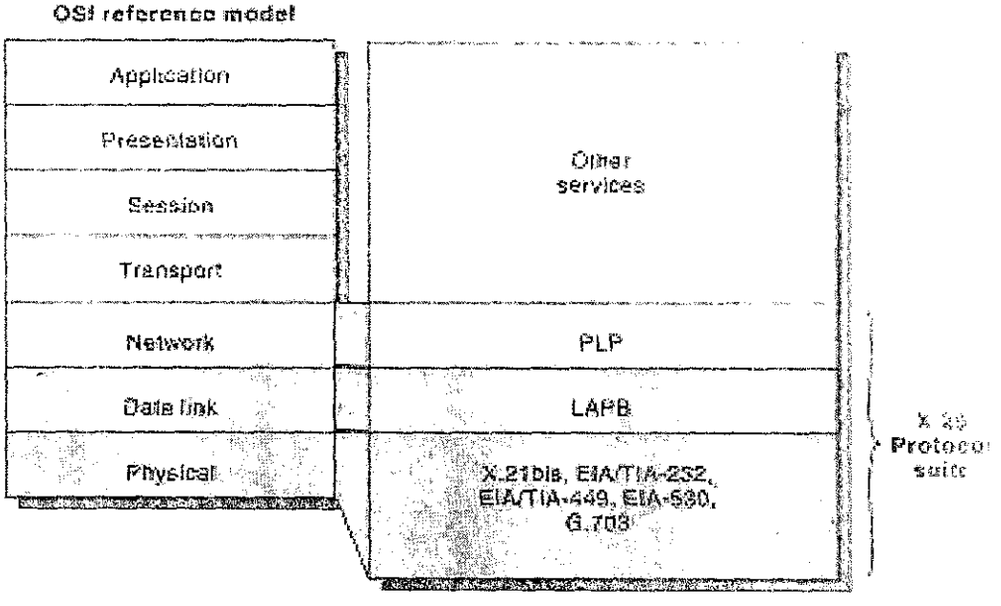


Figure 3.5: Mapping of key X.25 Protocols to the 3 Lower Layers of the OSI Model

3.7.1 Packet-Layer Protocol (PLP)

PLP is the X.25 network layer protocol. PLP manages packet exchanges between DTE devices across virtual circuits. PLPs also can run over Logical Link Control 2 (LLC2) implementations on LANs and over Integrated Services Digital Network (ISDN) interfaces running Link Access Procedure on the D channel (LAPD).

The PLP operates in five distinct modes:

- call setup,
- data transfer,
- idle,
- call clearing, and
- restarting.

Call setup mode

Call setup mode is used to establish SVCs between DTE devices. A PLP uses the X.121 addressing scheme to set up the virtual circuit.

The call setup mode is executed on a per-virtual-circuit basis, which means that one virtual circuit can be in call setup mode while another is in data transfer mode.

This mode is used only with SVCs, not with PVCs.

Data transfer mode

Data transfer mode is used for transferring data between two DTE devices across a virtual circuit. In this mode, PLP handles segmentation and reassembly, bit padding, and error and flow control.

This mode is executed on a per-virtual-circuit basis and is used with both PVCs and SVCs.

Idle mode

Idle mode is used when a virtual circuit is established but data transfer is not occurring. It is executed on a per-virtual-circuit basis and is used only with SVCs.

Call clearing mode

Call clearing mode is used to end communication sessions between DTE devices and to terminate SVCs. This mode is executed on a per-virtual-circuit basis and is used only with SVCs.

Restarting mode

Restarting mode is used to synchronize transmission between a DTE device and a locally connected DCE device.

This mode is not executed on a per-virtual-circuit basis. It affects all the DTE device's established virtual circuits.

Four types of PLP packet fields exist:

- **General Format Identifier (GFI)**
  - Identifies packet parameters, such as whether the packet carries user data or control information, what kind of windowing is being used, and whether delivery confirmation is required.
- **Logical Channel Identifier (LCI)**
  - Identifies the virtual circuit across the local DTE/DCE interface.
- **Packet Type Identifier (PTI)**
  - Identifies the packet as one of 17 different PLP packet types.
- **User Data**
  - Contains encapsulated upper-layer information. This field is present only in data packets. Otherwise, additional fields containing control information are added.

3.7.2 Link Access Procedure, Balanced (LAPB) Protocol

LAPB is a data link layer protocol that manages communication and packet framing between DTE and DCE devices. LAPB is a bit-oriented protocol that ensures that frames are correctly ordered and error free.

Three types of LAPB frames exist:

- Information,
- Supervisory, and
- Unnumbered.

The information frame (I-frame) carries upper-layer information and some control information. I-frame functions include sequencing, flow control, and error detection and recovery. I-frames carry send- and receive-sequence numbers.

The supervisory frame (S-frame) carries control information. S-frame functions include requesting and suspending transmissions, reporting on status, and acknowledging the receipt of I-frames. S-frames carry only receive-sequence numbers.

The unnumbered frame (U frame) carries control information. U-frame functions include link setup and disconnection, as well as error reporting. U frames carry no sequence numbers.

3.7.3 X.21bis Protocol

X.21bis is a physical layer protocol used in X.25 that defines the electrical and mechanical procedures for using the physical medium.

X.21bis handles the activation and deactivation of the physical medium connecting DTE and DCE devices. It supports point-to-point connections, speeds up to 19.2 kbps, and synchronous, full-duplex transmission over four-wire media.

Figure 3.6 shows the format of the PLP packet and its relationship to the LAPB frame and the X.21bis frame.

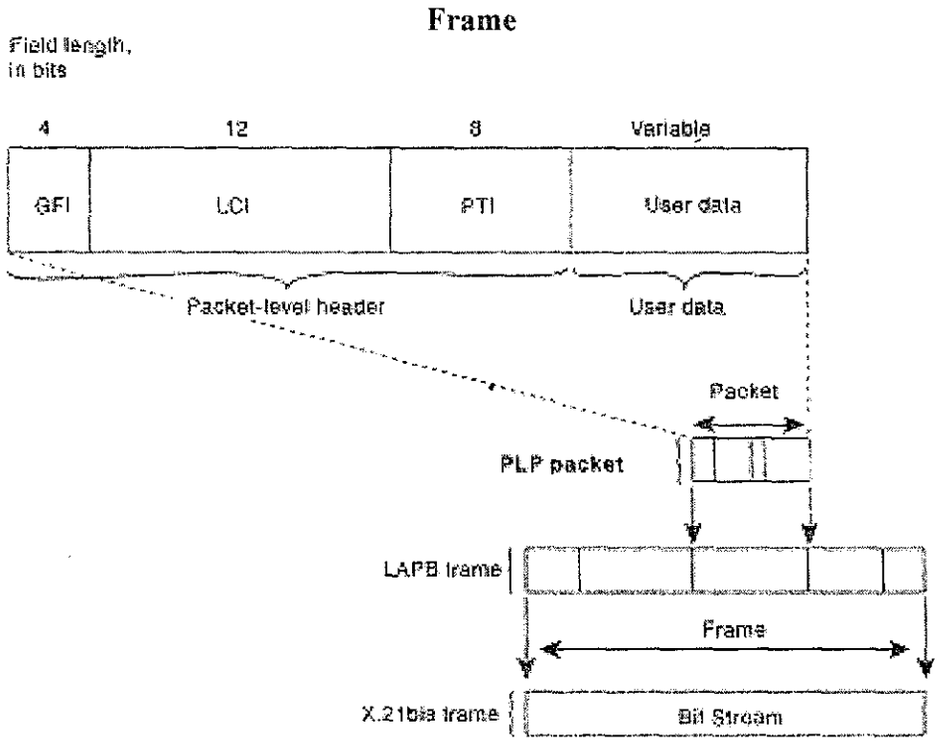


Figure 3.6: The PLP Packet Is Encapsulated Within the LAPB Frame and the X.21bis

LAPB Frame Format

LAPB frames include a header, encapsulated data, and a trailer. Figure 3.7 illustrates the format of the LAPB frame and its relationship to the PLP packet and the X.21bis frame.

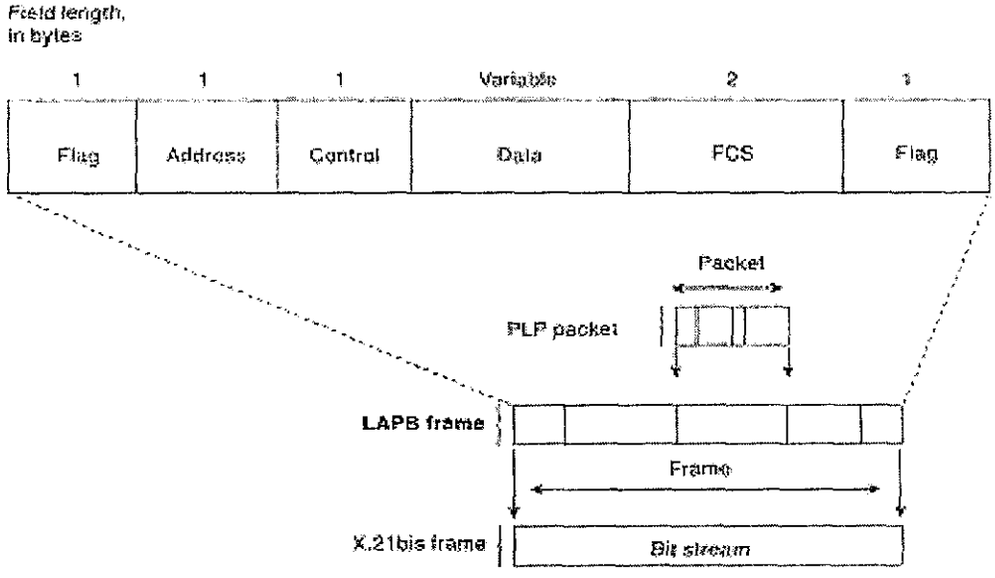


Figure 3.7: LAPB Frame Format

The following descriptions summarize the fields illustrated in Figure 2.6:

- **Flag**—Delimits the beginning and end of the LAPB frame. Bit stuffing is used to ensure that the flag pattern does not occur within the body of the frame.
- **Address**—Indicates whether the frame carries a command or a response.
- **Control**—Qualifies command and response frames and indicates whether the frame is an I-frame, an S-frame, or a U-frame. In addition, this field contains the frame's sequence number and its function (for example, whether receiver-ready or disconnect). Control frames vary in length depending on the frame type.
- **Data**—Contains upper-layer data in the form of an encapsulated PLP packet.
- **FCS**—Handles error checking and ensures the integrity of the transmitted data.

X.121 Address Format

X.121 addresses are used by the X.25 PLP in call setup mode to establish SVCs. Figure 3.8 illustrates the format of an X.121 address.

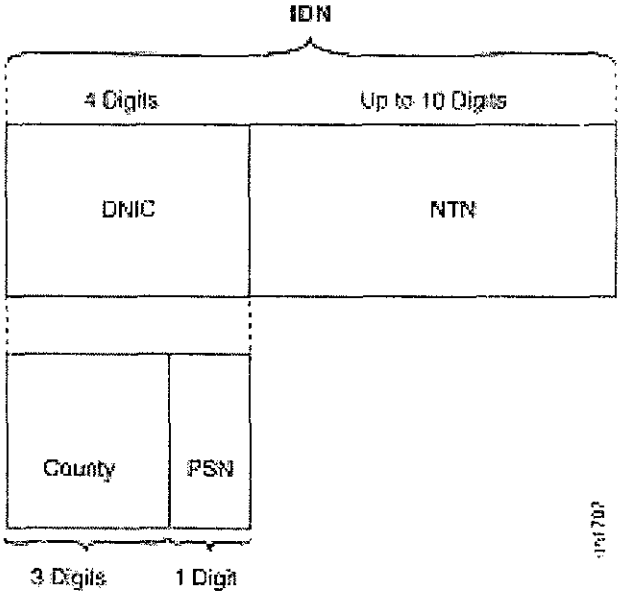


Figure 3.8: X.121 Address Format

The X.121 Address field includes the International Data Number (IDN), which consists of two fields:

- Data Network Identification Code (DNIC) and
- National Terminal Number (NTN).

DNIC is an optional field that identifies the exact PSN in which the destination DTE device is located. This field is sometimes omitted in calls within the same PSN. The DNIC has two subfields:

- Country and
- PSN.

The Country subfield specifies the country in which the destination PSN is located. The PSN field specifies the exact PSN in which the destination DTE device is located.
The NTN identifies the exact DTE device in the PSN for which a packet is destined. This field varies in length.

3.8 Switching Networks

A *switching network* is one in which there are too many computers to have a shared transmission medium or even full point-to-point connections, because of severe contention, complexity or expense. In this type of network consists of a (large) network of *nodes*, connected together in some particular topology. Some nodes attach to endpoints (aka *stations* (computers, telephones, terminals, etc.), while some are only connected to other nodes. Each node acts like a switch that accepts data on one of several inputs and switches it through to one of several outputs.
Although not all nodes are directly connected, there are one or more *paths* through which any two endpoints can communicate. *Routing* is the method of finding an appropriate path to connect two endpoints. Depending on the type of nodes and connections, the network may only be capable of supporting a subset of the possible connections at one time. In other words, not all endpoints may be able to communicate simultaneously. Under typical loads this is not a problem.
There are two main types of switching network:

- *Circuit-switched* and
- *Packet-switched.*

3.9 Circuit-switched Network

In a circuit switched network, a *circuit* is established between two endpoints in the network and the data is streamed over the connection, usually in full-duplex mode. The circuit is held open until one end disconnects. Often there is a certain data rate (or bandwidth) that is devoted by the network to the circuit regardless of whether it is actually being used at any given time or not.

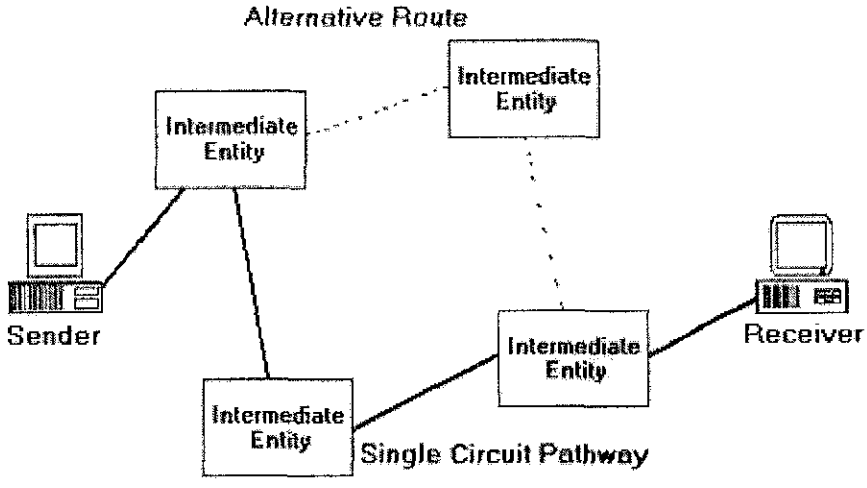


Figure 3.9: Circuit-switched Network

Example
The best example of a circuit-switched network is the public telephone system. A phone call establishes a connection and it is held open with a full-duplex 4 KHz link until one of the parties disconnects. With

the bursty nature of data (non-voice) communications, a circuit switched connection may often be idle, wasting network capacity.

3.10 Packet-switched Network

A packet-switched network breaks a data communications stream between two endpoints up into a sequence of *packets*, which may be fixed or variable-sized. The switched network absorbs packets and *routes* them through the network to the appropriate destination, where the packets can be unpacked and the data stream reproduced. The packets are buffered (copied) from node to node as they propagate through the network.

Packet switched networks generally utilize the network much more effectively than circuit-switched ones, since a packet is only sent if there is data to send and transferring a packet from node to node in the network only ties that connection up for a fraction of a second. This means the network can be "carrying" many more simultaneous communications.

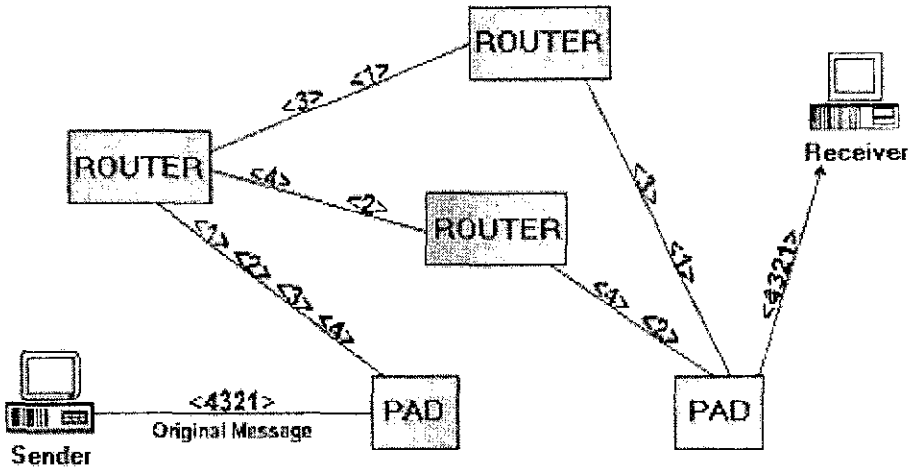


Figure 3.10: Packet-switched Network

Example(s)

Two good examples of packet switched networks are:

- ATM (Asynchronous Transfer Mode) and
- Internet.

3.11 Packet Switching Considerations

The following are major issues relating to packet switching:

- Datagrams vs. Virtual Circuits
- Quality-of-Service (QoS)
- Packet Size
- Congestion Control

All of these issues are discussed below.

Datagrams vs. Virtual Circuits

While the packet switching approach makes better use of network resources, there are some issues that arise when breaking a data communications stream into packets. First, since there are usually multiple

possible routes, some packets may arrive out of sequence and the receiver must be able to rearrange the packets into the proper order so that the data does not get scrambled. Also, a packet could get lost or corrupted due to a network glitch.

Rather than requiring each application to implement its own sequencing, flow control and error control, some packet switching networks have an option to handle these details for the application by providing a *virtual circuit.*

A virtual circuit looks to an application like a circuit in a circuit-switched network: it provides a connection-oriented data stream between two endpoints.

The network VC code takes care of breaking the data stream into packets at the sender and reassembling it at the receiver; handles out-of-order packets (packet sequencing), lost and corrupted packets, etc. Unlike true circuit-switched systems, an endpoint may have multiple virtual circuits in operation at once.

While most network applications would prefer to use a virtual circuit, some applications [e.g. *Ping*, etc.] may actually prefer to deal with a more low-level packet interface.

For this reason, most packet switched networks also offer *datagram* service: a raw packet service with little or no error detection.

It is more efficient and flexible than a VC (no connection setup/teardown, no flow or error control, etc.), but it requires the application to take over some of the work of packetizing, error control, etc.

Quality of Service (QoS)

While a virtual circuit simulates a circuit-switched connection, it may not provide the dedicated bandwidth between endpoints that the latter does. Instead the bandwidth available to an application may depend on external factors like how busy the network is with other packets. This means that the data rate of a virtual circuit may fluctuate greatly over the course of a connection.

While this is tolerable for applications like e-mail, real-time networking applications like video conferencing, airline reservations, electronic banking, etc. require a certain data rate to be responsive.

To remedy this, some packet switching networks (e.g. ATM) offer the capability to prioritize certain kinds of traffic or even request and reserve a certain data rate (quality of service) for a VC connection.

Packet Size

In a packet-switched network, there is a fundamental trade-off to be made in selecting the packet size. In nutshell:

- Larger packets: Less overhead, slower switching.
- Smaller packets: More overhead, faster switching.

With larger packets, less packets need to be sent. Since each packet contains some control information, less overall control information is needed. Also, the amount of control information relative to the amount of data is smaller. This all translates into greater efficiency. However, larger packets also take longer to copy from node to node and thus increase the delay for packets waiting in a nodes queues.

With smaller packets there is more overhead due to having more packets (as described above), but the transmission time per packet is small, and more packets can be switched faster; the data will arrive sooner at the destination.

However, there is another factor to consider: there is a minimum amount of control data that must be sent with each packet. If the amount of data per packet becomes too small then the total amount of control data eventually becomes inordinately large and network congestion and delays can result.

Usually a packet size is chosen that provides reasonably fast switching while avoiding [illegible] overhead.

Congestion Control

Congestion control is similar to flow control in point-to-point networks. There must be [illegible] the number of packets entering the network, otherwise the system can become overwhelmed packets, and throughput drops dramatically.

Queuing theory can provide insights in how to handle congestion control. Think of each [illegible] two queues (input and output) for each connection. A node examines each packet in each input [illegible] and places it in the appropriate output queue. The packets are removed from the output queue [illegible] as they can be sent to the next node connected to that queue.

If packets arrive on the input queues at a faster rate than they can be transmitted from the output [illegible] then the queues are going to grow, and there is a longer delay for each packet entering the node. If this continues indefinitely, the queues will grow to fill all available memory in the node; at that point the node cannot accept any more packets and will either have to drop incoming packets or tell the node upstream that it cannot send any more. Network traffic comes to a standstill.

If a node uses some kind of flow-control with the node upstream then it causes that node to hold its packets, exacerbating *its* situation. In effect, traffic can back up all the way to the sender and throughout the entire network. Thus, flow control as used on a point-to-point network is not sufficient and we need something that handles the "big picture".

The basic idea is to throttle the stations that are actually generating the packet traffic in the first place. If this is done properly we can keep the queues constrained so that queuing delay does not become excessive. On a per-node basis, the rate of packet acceptance should be about 80% of the maximum possible throughput of the node.

The main methods of throttle a sending station far upstream from a congested node include the following:

1. Use control-flow with the sender and or/other transmitting endpoints. Each packet sent contains the senders as well as the receivers' address (i.e. source and destination). A control packet can be to the source, telling it to slow down or cease sending packets.
2. Have the sender base the rate of packet submissions on routing information. If the sender has information about how fast or slow the links are comprising the route of the packets, it can adjust its transmission rate accordingly.
3. Have the sender monitor the throughput of the system and adjust the packet rate accordingly. The source can measure throughput on the network with a "ping"-like packet; if there is excessive delay the source can reduce the number or rate of packets it is sending.
4. Allow nodes to add congestion or throughput information to data packets that are traversing the network. The sender or receiver can note this and can adjust the rate accordingly

All of these methods require the cooperation of the sender and have pros and cons. However, the goal of congestion control is to avoid a network throughput collapse. In other words, if packets are being emitted into a network faster than the network can handle, the delay will increase indefinitely; however, with proper congestion control the network will at least keep the packets flowing as fast as possible, providing graceful degradation.

3.12 Circuit Switching Techniques

Communication in which a dedicated path is established between any two devices through one or more intermediate switching nodes is called circuit switching. Thus in a path consisting of a sequence of links between nodes, a channel is dedicated to a connection. The most common example of this is the telephone network.

There are three phases in communication via circuit switching:

1. Circuit establishment: End to end circuit is first established from a station wishing to communicate, by requesting the first connected node and then establishing a path through nodes which allocate free channels in node-node multiplexed links, until the desired destination node is reached.
2. Information transfer: The information which could be data or voice (analog or digital) are exchanged between source and destination. The ISDN standard envisages full digital exchange of data.
3. Circuit disconnection: Once information transfer is completed the connection is terminated by either party and the allocated channels are freed.

The best-known examples of circuit switching are the public telephone network, private branch exchange (PBX) in offices and in private wide area networks.

A public telephone network consists of subscribers e.g. telephones, local loop or subscriber loop which still uses twisted pairs of length a few kilometers, end office or exchanges that connect to the subscriber and trunks that connect between exchanges making use of multiplexed circuits.

Function(s) of Circuit Switching

The function of a circuit-switching node is to provide a transparent signal path between any pair of attached devices. Typically the connection must provide full duplex capability.

Blocking versus non blocking networks

A blocking network is one in which two stations may not be able to connect because all possible paths between them are already in use. This is typically the case in telephone networks. A non blocking network permits all stations to be connected in pairs at once and grants all possible connection requests as long as the calling party is free. This is usually the case in data communication networks.

Space Division versus Time Division Switching

A circuit switching technique in which each connection through the switch takes a physically [illegible] and dedicated path is ***called space division switching***. Thus the signal paths do not use any shared [illegible] that is they are physically separated. A typical example is the cross matrix switch.

Time Division Switching is a technique in which time slots are manipulated to pass [illegible] an input to an output. This is used in digital transmission and is achieved by using [illegible] Division multiplexing. For example, in TDM Bus switching all devices are connected on the bus. [illegible] slots are assigned to two devices that wish to communicate to each other. Thus during the assigned time slot only the two enabled devices are permitted to have data on the bus. For this scheme to work the internal bus must have a data rate significantly higher than the data rate on each I/O line

3.13 Routing and Control Signalling

The operation of a circuit switching network involves the establishment of a circuit between [illegible] systems that may be attached to different nodes., Two essential functions are involved in establishing such circuits:

- Routing and
- Control signalling.

There are discussed below.

Routing

In a large circuit switching network may of the circuit connections will require a path through [illegible] one switch. When a call is placed, the network must devise a route through the network from calling subscriber to called subscriber that passes through several switches and trunks. Two approaches

- Static hierarchical approach and
- Dynamic routing approach.

In the static approach the switches are organized in a tree and a path is established from the calling node to a common node and then to the called node. But this is not an efficient approach when the traffic is high.

A dynamic routing decision is one in which routing decisions are influenced by current traffic conditions. Thus all nodes are capable of performing the same functions. Dynamic routing is more complex but flexible. An example of dynamic type of routing is alternate routing.

Alternate Routing

The possible routes between two end offices are predefined. It is the responsibility of the originating switch to select the appropriate route for each call. Each switch is given a set of preplanned routes for each destination, in order of preference. If a direct trunk connection exists between two switches this

usually the preferred choice. Else the second choice is tried and so on. The routing priorities are based on an analysis of traffic patterns.

3.14 Packet Switching

In typical data communication most of the time the line is idle. Thus by providing a fixed connection capacity is wasted. Another problem in circuit switching is that due to the nature of the circuit connection the data rates of the transmitting station and the receiving station should be the same. These shortcomings are overcome in packet switching.

Packet switching is a method of transmitting messages through a communications network, in which messages are broken up into short packets. Each packet contains a portion of the user's **data** plus some **control** information. The control information includes the information required that the network requires to be able to route the packet through the network and deliver it to the intended destination. *At each node enroute the packet is received, stored briefly and passed on to the next node.*

A transmitting computer or other device sends a message as a sequence of packets. Each packet includes control information indicating the destination station (computer or other device). The packets are initially sent to the node to which the sending station attaches. As each packet arrives at this node, it is stored briefly, and the node determines the next leg of the route, and queues the packet onto an outward link. When the link is available, each packet is transmitted to the next node. All the packets eventually work their way through the network and are delivered to the intended destination.

3.15 Packet Switching Techniques

There are two approaches to packet switching, for linking two stations.

- Datagram technique
- Virtual Circuit technique

Datagram Technique

In this approach, each packet is treated separately, with no reference to the packets that have gone before. Each node chooses the next node on a packet's path, taking into account information from neighbouring nodes about the traffic on the network, line failures etc.

Thus the packets for a single message need not follow the same path, and they may arrive out of sequence at the destination. Either the destination station or the exit node is responsible for restoring the order of the packets. Each such packet treated separately is called a datagram.

Benefits of Datagrams Technique

Advantages of the datagram approach include the following:

1. Call setup phase is avoided. Thus if there are only a few packets to be transmitted datagram delivery will be quicker.

2. The datagram approach is more primitive and so more flexible. If congestion develops in one part of the network, incoming datagrams can be routed away from the congestion [illegible] possible when a virtual circuit has been established.
3. Datagram approach is more reliable since even if a node fails, alternate routes can be established. In virtual circuits if a node in a virtual circuit fails then all virtual circuits through that node fail.

Virtual Circuit Technique

In the **virtual circuit** approach, a preplanned route is established before any packets are sent. Once a route is established, all the packets between a pair of communicating parties follow this route through the network. Thus the route is fixed for the entire duration of a logical connection.

Since it is similar to the circuit switching technique in this respect, it is called a virtual circuit. Each packet contains a virtual circuit identifier as well as the data. Note that there is no dedicated path as in circuit switching; rather there is a dedicated path only for each packet duration.

Routing decisions are made dynamically (**adaptive routing**) depending upon failure of nodes and congestion. For adaptive routing, information about the state of the network must be exchanged between the nodes. This however can also result in overheads and increased traffic.

Benefits of Virtual Circuit Technique

Advantages of the Virtual circuit approach include the following:

1. If two stations want to exchange data over a long period of time then the virtual circuit approach is better since the network provides the sequencing and error control.
2. Since no routing decisions are to be made at each node the processing time is less. Hence virtual circuits are inherently faster than datagrams.

3.16 Congestion Control

Congestion control is required in-order to maintain the number of packets within the network or region of a network below a controllable level. A queue of packets is maintained at each switching node for each outgoing link. When the rate at which packets arrive at a node is greater than the rate at which packets leave a node, a queue starts building up. The objective of congestion control is to limit the queue lengths at the nodes. This however requires exchange of status information between nodes and control information between end nodes and attached stations limiting the flow of data. The X.25 protocol specifies the interface between a host station and the packet switching network. This standard specifies the physical interfaces between the host station and the switching node, the link level control specifications similar to the HDLC protocol we briefly reviewed earlier and packet level specifications for a virtual circuit service.

3.17 Comparison of Circuit Switching and Packet Switching

Advantages of packet switching

1. Line efficiency is greater since a line is shared by many packets and thus idle time is minimized.
2. Data rate conversion is possible using packet switching. Thus devices transmitting and receiving at differing data rates can talk to each other. This is made possible by the buffering at nodes.
3. When traffic increases in circuit switching, calls get blocked. Packet switching networks are non-blocking although when traffic increases the delays can also increase.
4. Priorities can be easily implemented in packet switching thus permitting higher priority packets to experience less delay.

Disadvantages of packet switching

1. In situations like voice communication or high throughput data communication, the delays on nodes for queuing and processing may call for circuit switching.
2. Since packets may take differing routes and varying delays a jitter is experienced in packet switching. This may affect real time applications.
3. Overhead information has to be added to each packet thus reducing the effective capacity.
4. Processing overheads are involved at each node.

Student Activity

1. What are network layer functions? Discuss.
2. Outline the working of Network Layer.
3. Discuss the classification of TCP/IP protocols.
4. What do you mean by X.25 protocol suite? Discuss.
5. What are switching circuits?
6. Differentiate between circuit -and packet-switched networks.
7. What are various circuit-switching techniques?
8. What is congestion control?
9. What are various packet switching techniques?
10. What is routing?

Unit 4

Internet Protocols, Routing Algorithms and Multiplexing

Learning Objectives

After reading this unit you should appreciate the following:

4.1 Introduction

The Internet protocols are the world's most popular open-system protocol suite because they can be used to communicate across any set of interconnected networks and are equally well suited for LAN and WAN communications.

The Internet protocols consist of a suite of communication protocols, of which the two best known are:

- Transmission Control Protocol (TCP) and
- Internet Protocol (IP).

The Internet protocol suite not only includes lower-layer protocols (such as TCP and IP), but it also specifies common applications such as electronic mail, terminal emulation, and file transfer.

Internet protocols were first developed in the mid-1970s, when the Defense Advanced Research Projects Agency (DARPA) became interested in establishing a packet-switched network that would facilitate communication between dissimilar computer systems at research institutions. With the goal of heterogeneous connectivity in mind, DARPA funded research by Stanford University and Bolt, Beranek, and Newman (BBN). The result of this development effort was the Internet protocol suite, completed in the late 1970s.

TCP/IP later was included with Berkeley Software Distribution (BSD) UNIX and has since become the foundation on which the Internet and the World Wide Web (WWW) are based.

Documentation of the Internet protocols (including new or revised protocols) and policies are specified in technical reports called Request For Comments (RFCs), which are published and then reviewed and analyzed by the Internet community. Protocol refinements are published in the new RFCs.

To illustrate the scope of the Internet protocols, Figure 4.1 maps many of the protocols of the Internet protocol suite and their corresponding OSI layers.

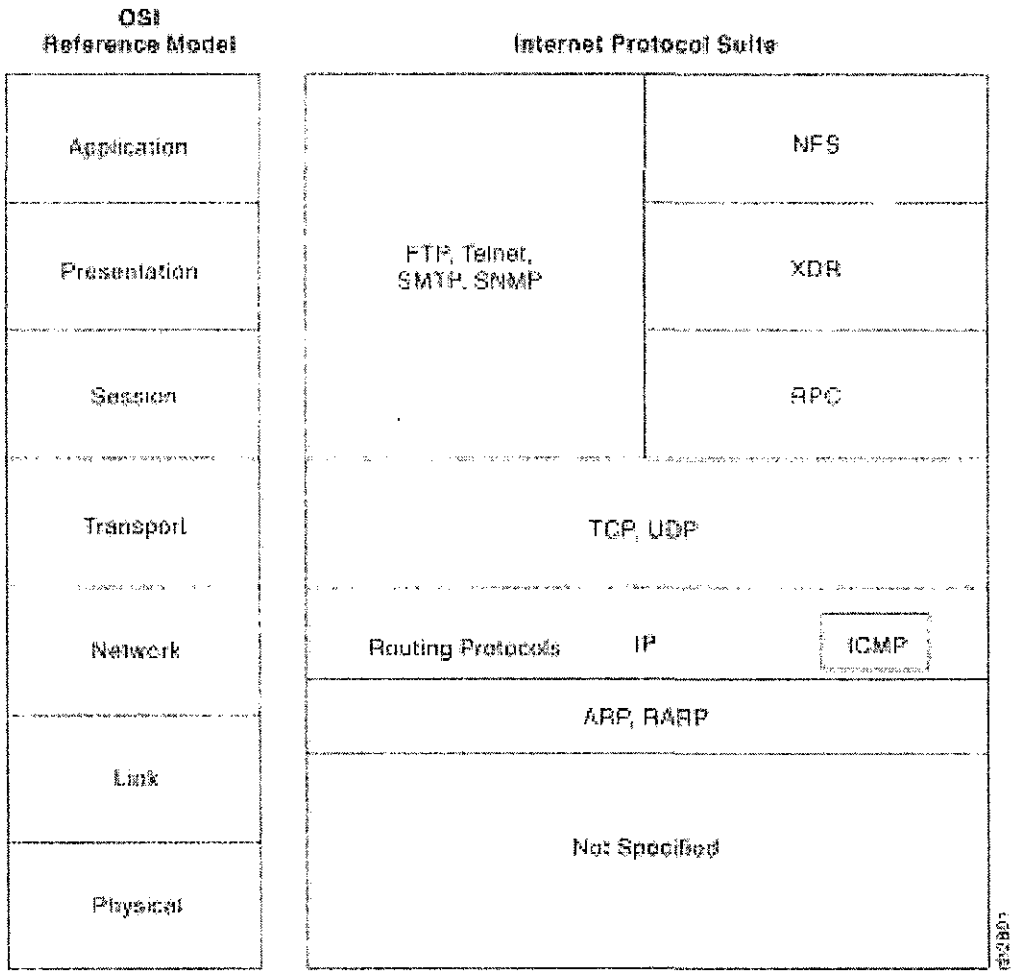


Figure 4.1: Internet protocols and OSI Layers.

4.2 Internet Protocol (IP)

The Internet Protocol (IP) is a network-layer (Layer 3) protocol that contains addressing information and some control information that enables packets to be routed. IP is documented in RFC 791 and is the primary network-layer protocol in the Internet protocol suite.

Along with the Transmission Control Protocol (TCP), IP represents the heart of the Internet protocols. IP has two primary responsibilities:

- Providing connectionless, best-effort delivery of datagrams through an internetwork; and
- Providing fragmentation and reassembly of datagrams to support data links with different maximum-transmission unit (MTU) sizes.

4.2.1 IP Packet Format

An IP packet contains several types of information, as illustrated in Figure 4.2.

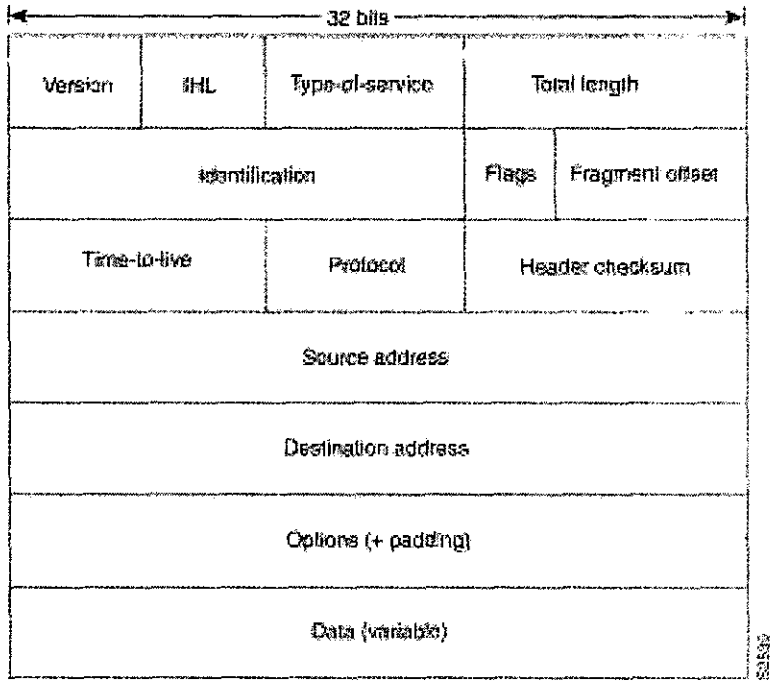


Figure 4.2: IP packet format

There are 14 fields in the IP packet, which are described below:

- *Version:* Indicates the version of IP currently used.
- *IP Header Length (*IHL): Indicates the datagram header length in 32-bit words.
- *Type-of-Service:* Specifies how an upper-layer protocol would like a current datagram to be handled, and assigns datagrams various levels of importance.
- *Total Length:* Specifies the length, in bytes, of the entire IP packet, including the data and header.

- *Identification:* Contains an integer that identifies the current datagram. This field is used to help piece together datagram fragments.
- *Flags:* Consists of a 3-bit field of which the two low-order (least-significant) bits control fragmentation. The low-order bit specifies whether the packet can be fragmented. The middle bit specifies whether the packet is the last fragment in a series of fragmented packets. The third or high-order bit is not used.
- *Fragment Offset:* Indicates the position of the fragment's data relative to the beginning of the data in the original datagram, which allows the destination IP process to properly reconstruct the original datagram.
- *Time-to-Live:* Maintains a counter that gradually decrements down to zero, at which point the datagram is discarded. This keeps packets from looping endlessly.
- *Protocol:* Indicates which upper-layer protocol receives incoming packets after IP processing is complete.
- *Header Checksum:* Helps ensure IP header integrity.
- *Source Address:* Specifies the sending node.
- *Destination Address:* Specifies the receiving node.
- *Options:* Allows IP to support various options, such as security.
- *Data:* Contains upper-layer information.

4.3 IP Addressing

As with any other network-layer protocol, the IP addressing scheme is integral to the process of routing IP datagrams through an internetwork. Each IP address has specific components and follows a basic format. These IP addresses can be subdivided and used to create addresses for subnetworks, as discussed in more detail later in this chapter.

Each host on a TCP/IP network is assigned a unique 32-bit logical address that is divided into two main parts:

- Network number and
- Host number.

The network number identifies a network and must be assigned by the Internet Network Information Center (InterNIC) if the network is to be part of the Internet. An Internet Service Provider (ISP) can obtain blocks of network addresses from the InterNIC and can itself assign address space as necessary.

The host number identifies a host on a network and is assigned by the local network administrator.

4.3.1 · IP Address Format

The 32-bit IP address is grouped eight bits at a time, separated by dots, and represented in decimal format (known as *dotted decimal notation*). Each bit in the octet has a binary weight (128, 64, 32, 16, 8, 4, 2, 1). The minimum value for an octet is 0, and the maximum value for an octet is 255.

Figure 4.3 illustrates the basic format of an IP address.

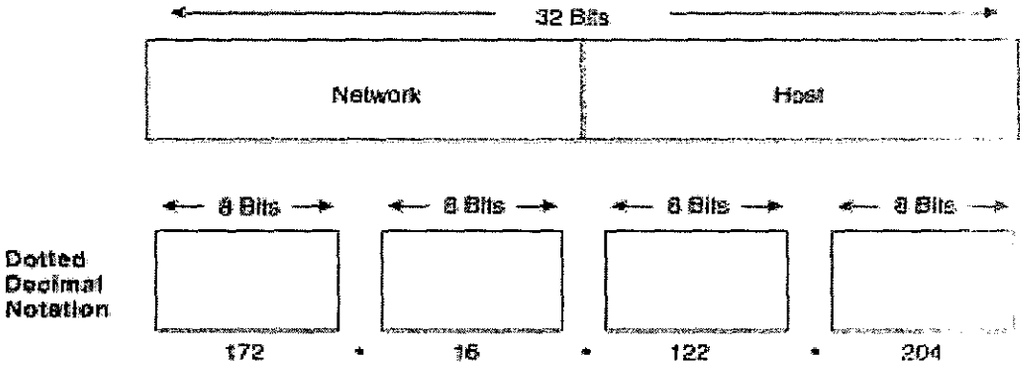


Figure 4.3: IP address format

4.3.2 IP Address Classes

IP addressing supports five different address classes: A, B,C, D, and E. Only classes A, B, and C are available for commercial use. The left-most (high-order) bits indicate the network class.

Table 4.1 provides reference information about the five IP address classes.

Table 4.1: Reference Information About the Five IP Address Classes

IP Address Class	Format	Purpose	High-Order Bit(s)	Address Range	No. Bits Network/Host	Max. Hosts
A	N.H.H.H[1]	Few large organizations	0	1.0.0.0 to 126.0.0.0	7/24	16,777,214[2] (2^{24} - 2)
B	N.N.H.H	Medium-size organizations	1, 0	128.1.0.0 to 191.254.0.0	14/16	65, 534 (2^{16} - 2
C	N.N.N.H	Relatively small organizations	1, 1, 0	192.0.1.0 to 223.255.254.0	22/8	245 (2^{8} - 2)
D	N/A	Multicast groups (RFC 1112)	1, 1, 1, 0	224.0.0.0 to 239.255.255.255	N/A (not for commercial use)	N/A
E	N/A	Experimental	1, 1, 1, 1	240.0.0.0 to 254.255.255.255	N/A	N/A

[1]N = Network number, H = Host number.

[2]One address is reserved for the broadcast address, and one address is reserved for the network.

Figure 4.4 illustrates the format of the commercial IP address classes.

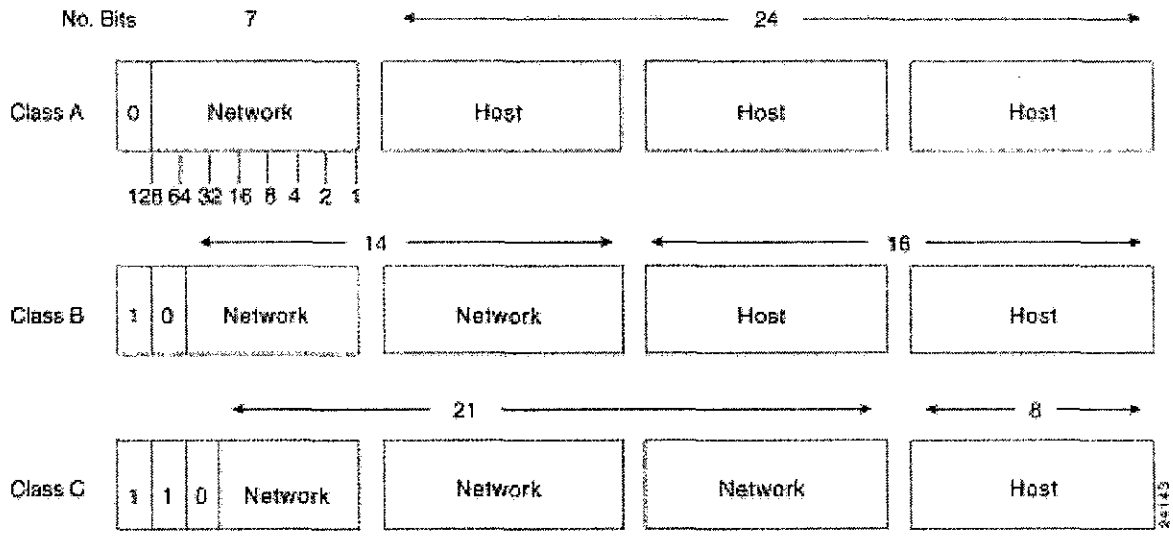


Figure 4.4: IP address formats A, B, and C

The class of address can be determined easily by examining the first octet of the address and mapping that value to a class range in the following table.

In an IP address of 172.31.1.2, for example, the first octet is 172. Because 172 falls between 128 and 191, 172.31.1.2 is a Class B address.

Table 4.2 summarizes the range of possible values for the first octet of each address class.

Table 4.2: Range of values exists for the first octet of each address class.

Address Class	First Octet in Decimal	High-Order Bits
Class A	1 Ð 126	0
Class B	128 Ð 191	10
Class C	192 Ð 223	110
Class D	224 Ð 239	1110
Class E	240 Ð 254	1111

24144

4.4 IP Subnet Addressing

IP networks can be divided into smaller networks called sub-networks (or subnets). Subnetting provides the network administrator with several benefits, including extra flexibility, more efficient use of network addresses, and the capability to contain broadcast traffic (a broadcast will not cross a router).

Subnets are under local administration. As such, the outside world sees an organization as a single network and has no detailed knowledge of the organization's internal structure.

A given network address can be broken up into many subnetworks. For example, 172.16.1.0, 172.16.2.0, 172.16.3.0, and 172.16.4.0 are all subnets within network 171.16.0.0. (All 0s in the host portion of an address specifies the entire network.)

4.4.1 IP Subnet Mask

A subnet address is created by "borrowing" bits from the host field and designating them as the subnet field. The number of borrowed bits varies and is specified by the subnet mask. Figure 4.5 shows how bits are borrowed from the host address field to create the subnet address field.

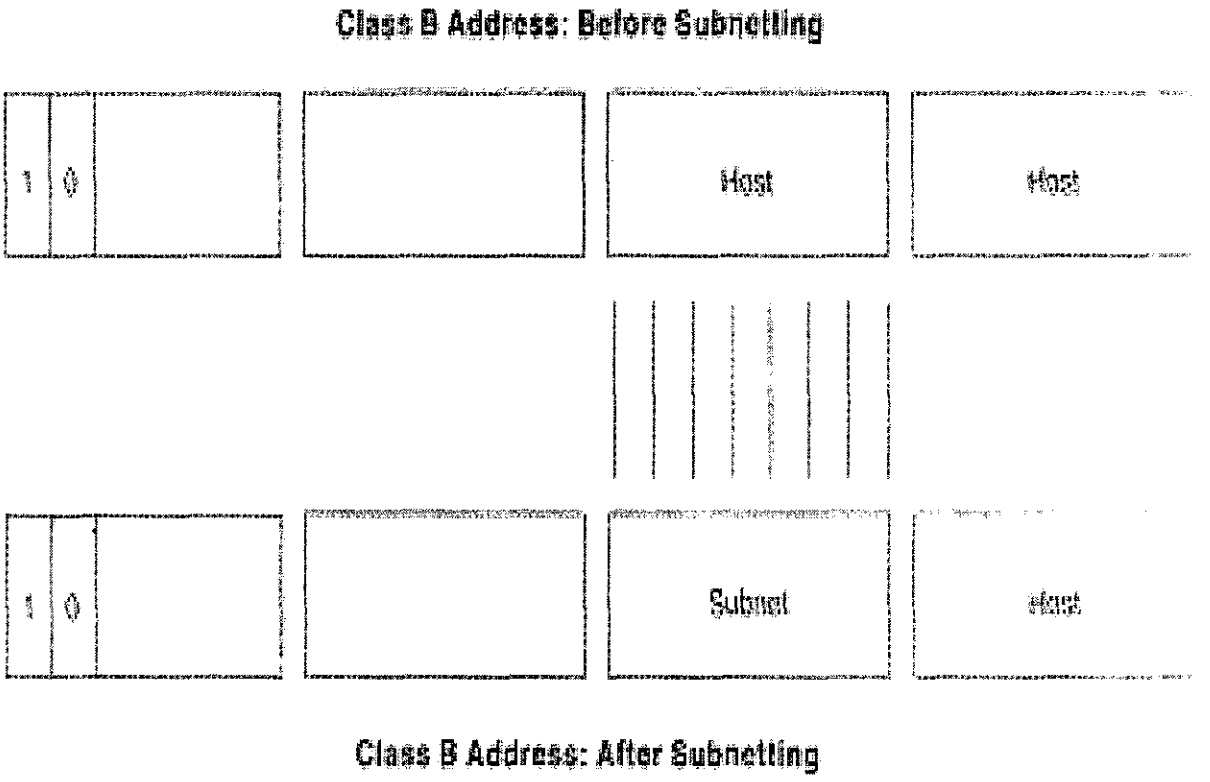


Figure 4.5: Creation of subnet address field

Subnet masks use the same format and representation technique as IP addresses. The subnet mask, however, has binary 1s in all bits specifying the network and subnetwork fields, and binary 0s in all bits specifying the host field.

Figure 4.6 illustrates a sample subnet mask.

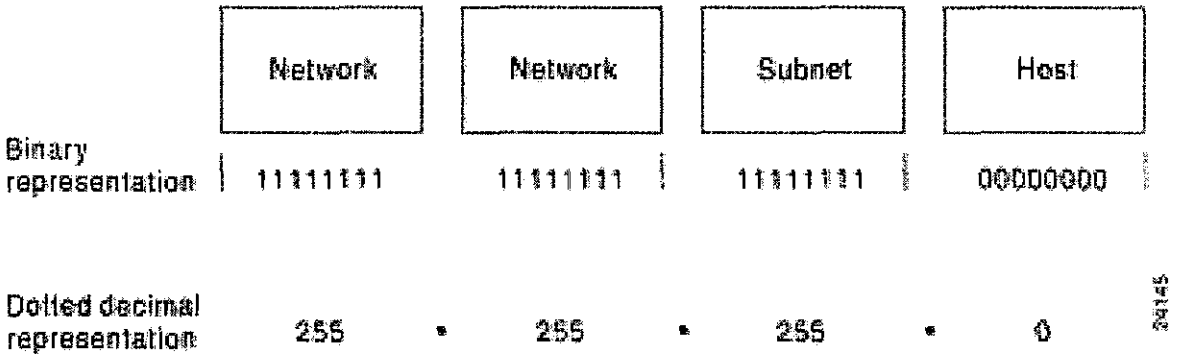


Figure 4.6: A sample subnet mask consists of all binary 1s and 0s.

Subnet mask bits should come from the high-order (left-most) bits of the host field, as Figure 4.7 illustrates. Details of Class B and C subnet mask types follow. Class A addresses are not discussed in this chapter because they generally are subnetted on an 8-bit boundary.

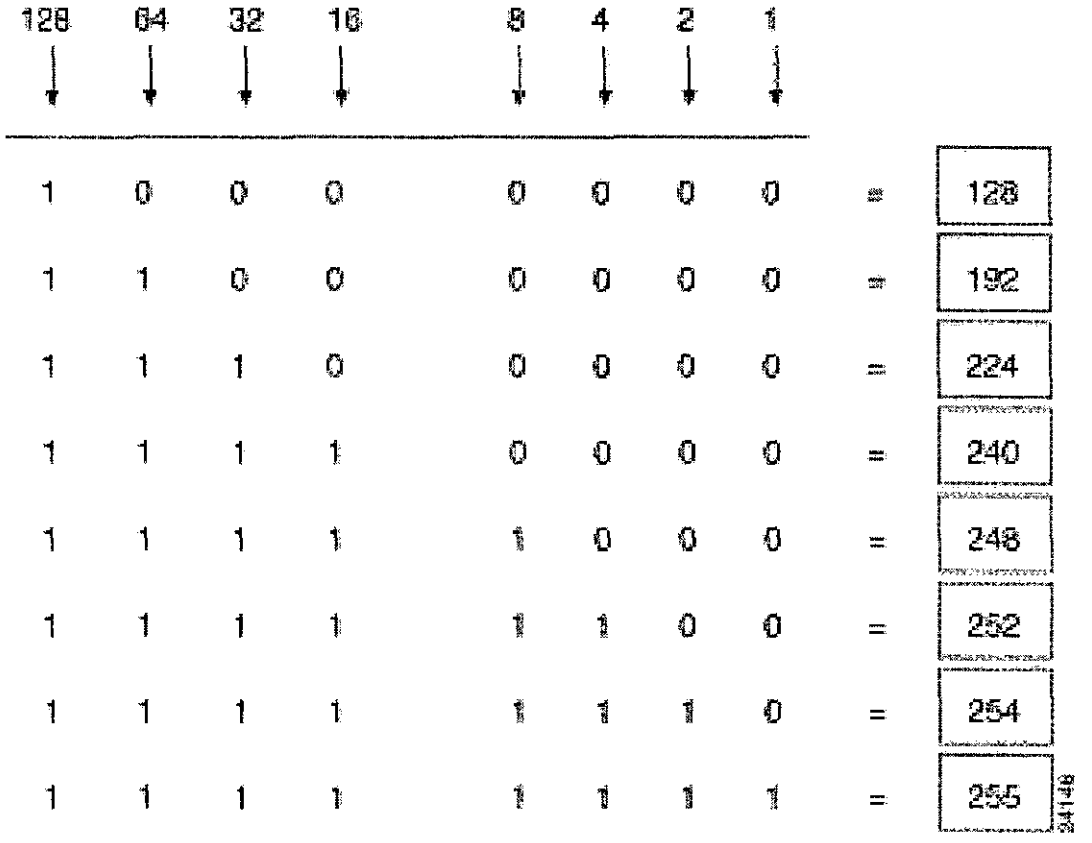


Figure 4.7: Subnet mask bits come from the high-order bits of the host field.

Various types of subnet masks exist for Class B and C subnets.

The default subnet mask for a Class B address that has no subnetting is 255.255.0.0, while the subnet mask for a Class B address 171.16.0.0 that specifies eight bits of subnetting is 255.255.255.0. The

reason for this is that eight bits of subnetting or 2^8 - 2 (1 for the network address and 1 for the broadcast address) = 254 subnets possible, with 2^8 - 2 = 254 hosts per subnet.

The subnet mask for a Class C address 192.168.2.0 that specifies five bits of subnetting is 255.255.255.248.With five bits available for subnetting, 2^5 - 2 = 30 subnets possible, with 2^3 - 2 = 6 hosts per subnet.

The reference charts shown in Table 4.3 and Table 4.4 can be used when planning Class B and C networks to determine the required number of subnets and hosts, and the appropriate subnet mask.

Table 4.3: Class B Subnetting Reference Chart

Number of Bits	Subnet Mask	Number of Subnets	Number of Hosts
2	255.255.192.0	2	16382
3	255.255.224.0	6	8190
4	255.255.240.0	14	4094
5	255.255.248.0	30	2046
6	255.255.252.0	62	1022
7	255.255.254.0	126	510
8	255.255.255.0	254	254
9	255.255.255.128	510	126
10	255.255.255.192	1022	62
11	255.255.255.224	2046	30
12	255.255.255.240	4094	14
13	255.255.255.248	8190	6
14	255.255.255.252	16382	2

Table 4.4: Class C Subnetting Reference Chart

Number of Bits	Subnet Mask	Number of Subnets	Number of Hosts
2	255.255.255.192	2	62
3	255.255.255.224	6	30
4	255.255.255.240	14	14
5	255.255.255.248	30	6
6	255.255.255.252	62	2

4.4.2 Using Subnet Masks to Determine Network Number

The router performs a set process to determine the network (or more specifically, the subnetwork) address. First, the router extracts the IP destination address from the incoming packet and retrieves the internal subnet mask. It then performs a *logical AND* operation to obtain the network number. This causes the host portion of the IP destination address to be removed, while the destination network number remains. The router then looks up the destination network number and matches it with an outgoing interface. Finally, it forwards the frame to the destination IP address. Specifics regarding the logical AND operation are discussed in the following section.

Logical AND Operation

Three basic rules govern logically "ANDing" two binary numbers. First, 1 "ANDed" with 1 yields 1. Second, 1 "ANDed" with 0 yields 0. Finally, 0 "ANDed" with 0 yields 0. The truth table provided in Table 4.5 illustrates the rules for logical AND operations.

Table 4.5: Rules for Logical AND Operations

Input	Input	Output
1	1	1
1	0	0
0	1	0

Two simple guidelines exist for remembering logical AND operations: Logically "ANDing" a 1 with a number yields the original value, and logically "ANDing" a 0 with any number yields 0.
Figure 4.8 illustrates that when a logical AND of the destination IP address and the subnet mask is performed, the subnetwork number remains, which the router uses to forward the packet.

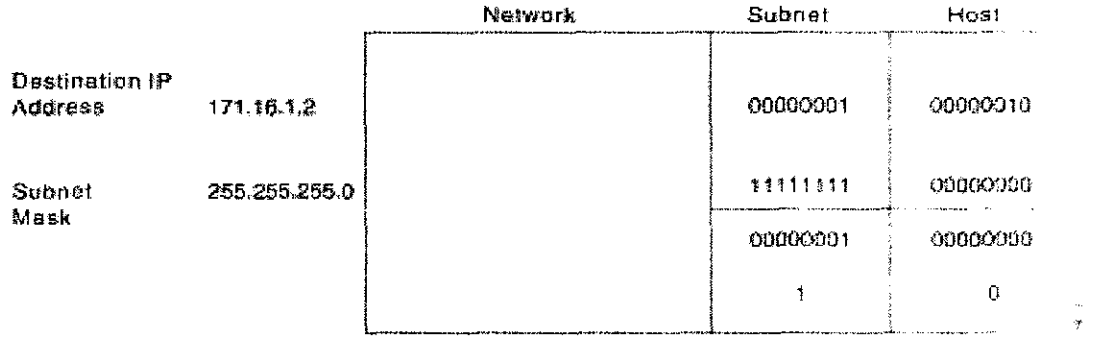


Figure 4.8: Producing subnetwork number

4.5 Address Resolution Protocol (ARP)

For two machines on a given network to communicate, they must know the other machine's physical (or MAC) addresses. By broadcasting Address Resolution Protocols (ARPs), a host can dynamically discover the MAC-layer address corresponding to a particular IP network-layer address.

After receiving a MAC-layer address, IP devices create an ARP cache to store the recently acquired IP-to-MAC address mapping, thus avoiding having to broadcast ARPS when they want to recontact a device. If the device does not respond within a specified time frame, the cache entry is flushed

In addition to the Reverse Address Resolution Protocol (RARP) is used to map MAC-layer addresses to IP addresses. RARP, which is the logical inverse of ARP, might be used by diskless workstations that do not know their IP addresses when they boot. RARP relies on the presence of a RARP server with table entries of MAC-layer-to-IP address mappings.

4.6 Internet Routing

Internet routing devices traditionally have been called gateways. In today's terminology, however, the term gateway refers specifically to a device that performs application-layer protocol translation between devices. Interior gateways refer to devices that perform these protocol functions between machines or networks under the same administrative control or authority, such as a corporation's internal network. These are known as autonomous systems. Exterior gateways perform protocol functions between independent networks.

Routers within the Internet are organized hierarchically. Routers used for information exchange within autonomous systems are called interior routers, which use a variety of Interior Gateway Protocols (IGPs) to accomplish this purpose. The Routing Information Protocol (RIP) is an example of an IGP.

Routers that move information between autonomous systems are called exterior routers. These routers use an exterior gateway protocol to exchange information between autonomous systems. The Border Gateway Protocol (BGP) is an example of an exterior gateway protocol.

Using the hierarchical core router model to distribute routing information has a major weakness: *every route must be processed by the core.*

This places a tremendous processing burden on the core, and as the Internet grew larger the burden increased. In network-speak, we say that this routing model does not "scale well." For this reason, a new model emerged.

Even in the days of a single Internet, core groups of independent networks called *autonomous systems* (AS) existed outside of the core. The term "autonomous system" has a formal meaning in TCP/IP routing.

An autonomous system is not merely an independent network. It is a collection of networks and gateways with its own internal mechanism for collecting routing information and passing it to other independent network systems.

The routing information passed to the other network systems is called *reachability information.* Reachability information simply says which networks can be reached through that autonomous system.

The *Exterior Gateway Protocol* (EGP) was the protocol used to pass reachability information between autonomous systems and into the core (see Figure 4.9).

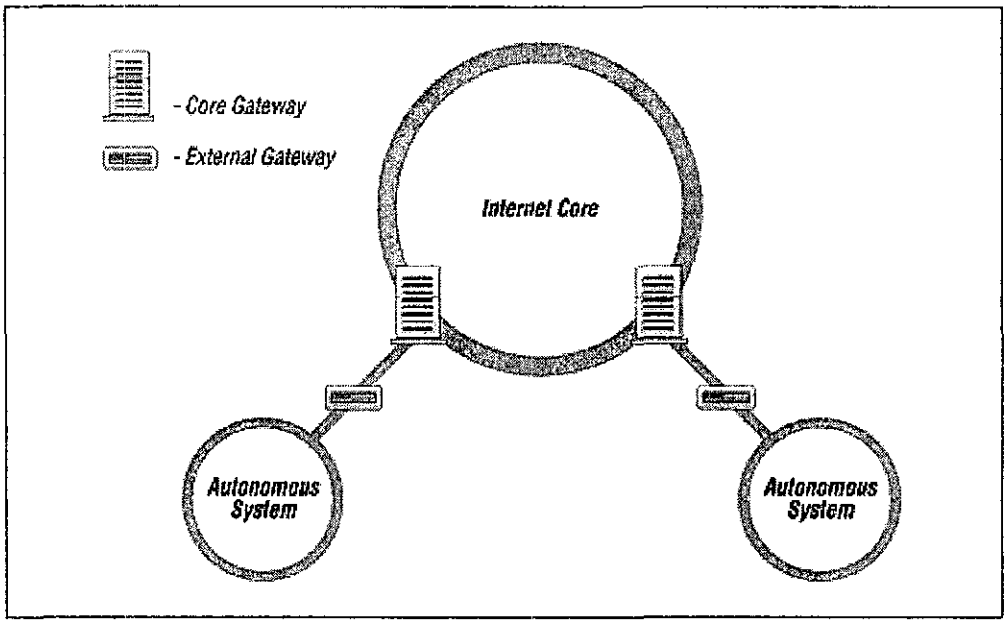


Figure 4.9: Gateway hierarchy

The new routing model is based on co-equal collections of autonomous systems, called *routing domains*. Routing domains exchange routing information with other domains using *Border Gateway Protocol* (BGP).

Each routing domain processes the information it receives from other domains. Unlike the hierarchical model, this model does not depend on a single core system to choose the "best" routes. Each routing domain does this processing for itself; therefore, this model is more expandable.

Figure 4.10 represents this model with three intersecting circles. Each circle is a routing domain. The overlapping areas are border areas, where routing information is shared. The domains share information but do not rely on any one system to provide all routing information.

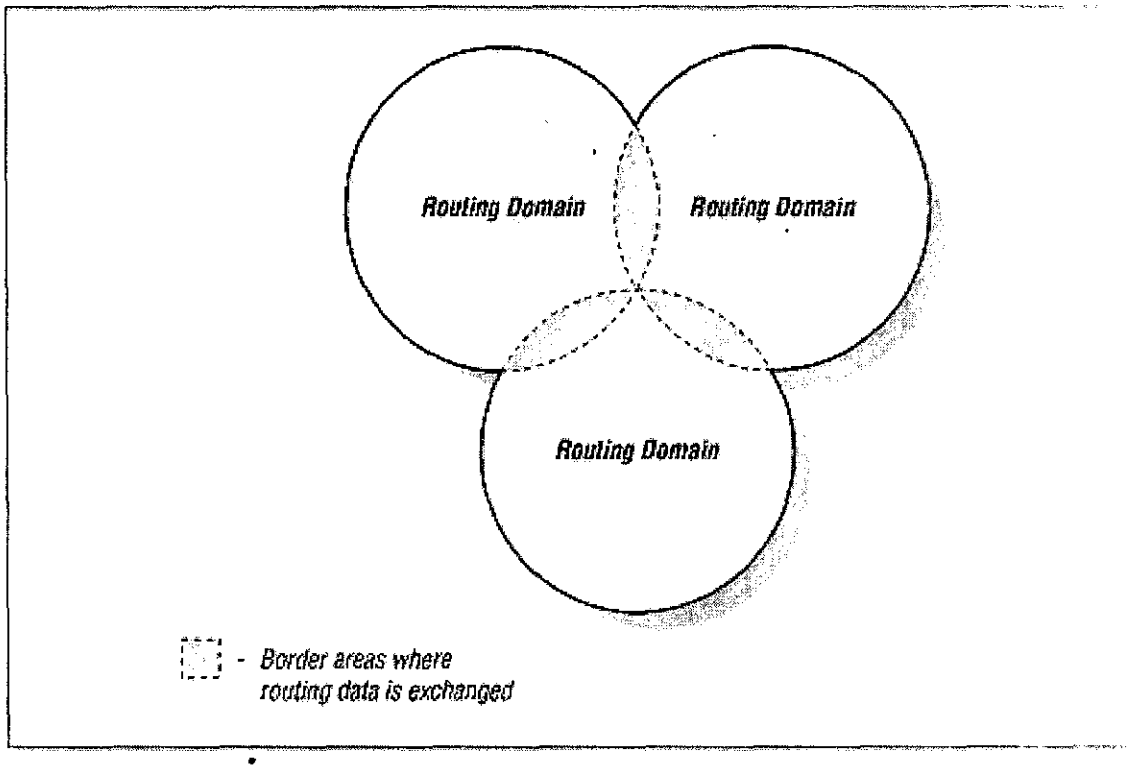


Figure 4.10: Routing domains

The problem with this model is: how are "best" routes determined in a global network if there is no central routing authority, like the core, that is trusted to determine the "best" routes? In the days of the NSFNET, the *policy routing database* (PRDB) was used to determine whether the reachability information advertised by an autonomous system was valid. But now, even the NSFNET does not play a central role.

To fill this void, NSF created the *Routing Arbiter* (RA) servers when it created the *Network Access Points* (NAPs) that replaced the role of the NSFNET. A route arbiter is located at each NAP. The server provides access to the *Routing Arbiter Database* (RADB), which replaced the PRDB. Internet Service Providers can query servers to validate the reachability information advertised by an autonomous system.

Many ISPs do not use the route servers. Instead they depend on formal and informal bilateral agreements. In essence, two ISPs get together and decide what reachability information each will accept from the other. They create, in effect, local routing policies. This is a slow manual process that probably will not be flexible enough for a rapidly growing Internet.

The RADB is only part of the *Internet Routing Registry* (IRR). As befits a distributed routing architecture, there are multiple organizations that validate and register routing information. Europeans were the pioneers in this. The Reseaux IP Europeens (RIPE) Network Control Center (NCC) provides the routing registry for European IP networks. Big network carriers, like MCI and ANS, provide registries for their customers. All of the registries share a common format based on the RIPE-181 standard.

Creating an effective routing architecture continues to be a major challenge for the Internet that will certainly evolve over time. No matter how it is derived, eventually the routing information winds up in your local gateway, where it is used by IP to make routing decisions.

4.7 IP Routing

IP routing protocols are dynamic. Dynamic routing calls for routes to be calculated automatically at regular intervals by software in routing devices. This contrasts with static routing, where routers are established by the network administrator and do not change until the network administrator changes them.

An IP routing table, which consists of destination address/next hop pairs, is used to enable dynamic routing. An entry in this table, for example, would be interpreted as follows:

- To get to network 172.31.0.0,
- Send the packet out Ethernet interface 0 (E0).

IP routing specifies that IP datagrams travel through internetworks one hop at a time. The entire route is not known at the onset of the journey, however. Instead, at each stop, the next destination is calculated by matching the destination address within the datagram with an entry in the current node's routing table.

Each node's involvement in the routing process is limited to forwarding packets based on internal information. The nodes do not monitor whether the packets get to their final destination, nor does IP provide for error reporting back to the source when routing anomalies occur. This task is left to another Internet protocol, the Internet Control-Message Protocol (ICMP), which is discussed in the following section.

4.8 Internet Control Message Protocol (ICMP)

The *Internet Control Message Protocol (ICMP)* is a network-layer Internet protocol that provides message packets to report errors and other information regarding IP packet processing back to the source. ICMP is documented in RFC 792.

4.8.1 ICMP Messages

ICMPs generate several kinds of useful messages, including the following:

- Destination Unreachable,
- Echo Request and Reply message,
- Redirect message,
- Time Exceeded message, and
- Router Advertisement and Router Solicitation message.

If an ICMP message cannot be delivered, no second one is generated. This is to avoid an endless flood of ICMP messages.

Destination Unreachable Message

When an ICMP destination-unreachable message is sent by a router, it means that the router is unable to send the package to its final destination. The router then discards the original packet. Two reasons exist for why a destination might be unreachable. Most commonly, the source host has specified a nonexistent address. Less frequently, the router does not have a route to the destination.

Destination-unreachable messages include four basic types:

- Network unreachable,
- Host unreachable,
- Protocol unreachable, and
- Port unreachable.

Network-unreachable messages usually mean that a failure has occurred in the routing or addressing of a packet.

Host-unreachable messages usually indicates delivery failure, such as a wrong subnet mask.

Protocol-unreachable messages generally mean that the destination does not support the upper layer protocol specified in the packet.

Port-unreachable messages imply that the TCP socket or port is not available.

Echo Request and Reply message

An ICMP echo-request message, which is generated by the ping command, is sent by any host to test node reachability across an internetwork. The ICMP echo-reply message indicates that the node can be successfully reached.

Redirect message

An ICMP Redirect message is sent by the router to the source host to stimulate more efficient routing. The router still forwards the original packet to the destination. ICMP redirects allow host routing tables to remain small because it is necessary to know the address of only one router, even if that router does not provide the best path. Even after receiving an ICMP Redirect message, some devices might continue using the less-efficient route.

Time-exceed message

An ICMP Time-exceeded message is sent by the router if an IP packet's Time-to-Live field (expressed in hops or seconds) reaches zero. The Time-to-Live field prevents packets from continuously circulating the internetwork if the internetwork contains a routing loop. The router then discards the original packet.

4.9 ICMP Router-Discovery Protocol (IDRP)

IDRP uses Router-Advertisement and Router-Solicitation messages to discover the addresses of routers on directly attached subnets. Each router periodically multicasts Router-Advertisement messages from each of its interfaces. Hosts then discover addresses of routers on directly attached subnets by listening for these messages. Hosts can use Router-Solicitation messages to request immediate advertisements rather than waiting for unsolicited messages.

IRDP offers several advantages over other methods of discovering addresses of neighboring routers. Primarily, it does not require hosts to recognize routing protocols, nor does it require manual configuration by an administrator.

Router-Advertisement messages enable hosts to discover the existence of neighboring routers, but not which router is best to reach a particular destination. If a host uses a poor first-hop router to reach a particular destination, it receives a Redirect message identifying a better choice.

4.10 Transmission Control Protocol (TCP)

The TCP provides reliable transmission of data in an IP environment. TCP corresponds to the transport layer (Layer 4) of the OSI reference model. TCP provides the following services:

- Stream data transfer,
- Reliability,
- Efficient flow control,
- Full-duplex operation, and
- Multiplexing.

Stream data transfer

With stream data transfer, TCP delivers an unstructured stream of bytes identified by sequence numbers. This service benefits applications because they do not have to chop data into blocks before handing it off to TCP. Instead, TCP groups bytes into segments and passes them to IP for delivery.

Reliability

TCP offers reliability by providing connection-oriented, end-to-end reliable packet delivery through an internetwork. It does this by sequencing bytes with a forwarding acknowledgment number that indicates to the destination the next byte the source expects to receive. Bytes not acknowledged within a specified time period are retransmitted. The reliability mechanism of TCP allows devices to deal with lost, delayed, duplicate, or misread packets. A time-out mechanism allows devices to detect lost packets and request retransmission.

Efficient flow control

TCP offers efficient flow control, which means that, when sending acknowledgments back to the source, the receiving TCP process indicates the highest sequence number it can receive without overflowing its internal buffers.

Full-duplex operation

Full-duplex operation means that TCP processes can both send and receive at the same time.

Multiplexing

Finally, TCP's multiplexing means that numerous simultaneous upper-layer conversations can be multiplexed over a single connection.

4.10.1 TCP Connection Establishment

To use reliable transport services, TCP hosts must establish a connection-oriented session with one another. Connection establishment is performed by using a "3-way handshake" mechanism, shown in Figure 4.11.

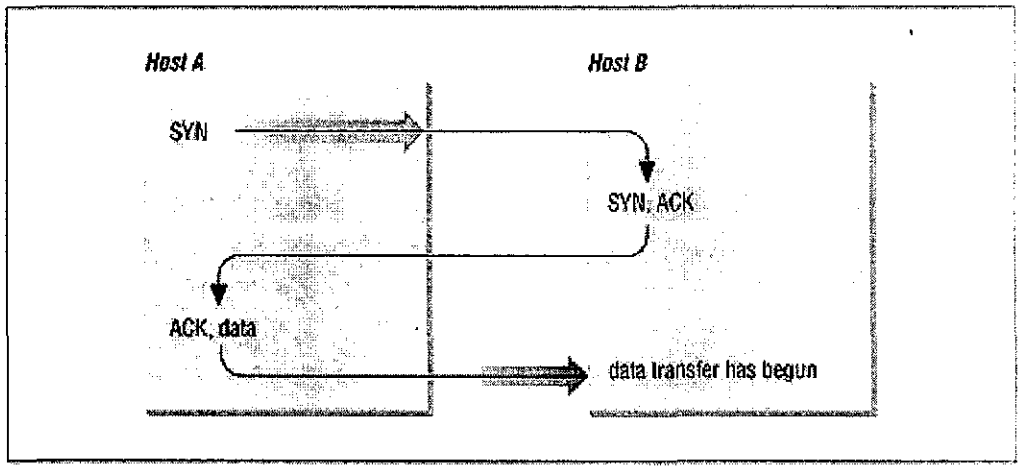


Figure 4.11: 3-way handshake

A 3-way handshake synchronizes both ends of a connection by allowing both sides to agree upon initial sequence numbers. This mechanism also guarantees that both sides are ready to transmit data and know

that the other side is ready to transmit as well. This is necessary so that packets are not transmitted or retransmitted during session establishment or after session termination.

Each host randomly chooses a sequence number used to track bytes within the stream it is sending and receiving. Then, the three-way handshake proceeds in the following manner:

- The first host (Host A) initiates a connection by sending a packet with the initial sequence number (X) and SYN bit set to indicate a connection request.
- The second host (Host B) receives the SYN, records the sequence number X, and replies by acknowledging the SYN (with an ACK = X + 1).
- Host B includes its own initial sequence number (SEQ = Y).
- An ACK = 20 means the host has received bytes 0 through 19 and expects byte 20 next. This technique is called *forward acknowledgment.*
- Host A then acknowledges all bytes Host B sent with a forward acknowledgment indicating the next byte Host A expects to receive (ACK = Y + 1).
- Data transfer then can begin.

4.10.2 Positive Acknowledgment and Retransmission (PAR)

A simple transport protocol might implement a reliability-and-flow-control technique where the source sends one packet, starts a timer, and waits for an acknowledgment before sending a new packet. If the acknowledgment is not received before the timer expires, the source retransmits the packet. Such a technique is called *positive acknowledgment and retransmission* (PAR).

By assigning each packet a sequence number, PAR enables hosts to track lost or duplicate packets caused by network delays that result in premature retransmission. The sequence numbers are sent back in the acknowledgments so that the acknowledgments can be tracked.

PAR is an inefficient use of bandwidth, however, because a host must wait for an acknowledgment before sending a new packet, and only one packet can be sent at a time.

4.10.3 TCP Sliding Window

A *TCP sliding window* provides more efficient use of network bandwidth than PAR because it enables hosts to send multiple bytes or packets before waiting for an acknowledgment.

In TCP, the receiver specifies the current window size in every packet. Because TCP provides a byte-stream connection, window sizes are expressed in bytes. This means that a window is the number of data bytes that the sender is allowed to send before waiting for an acknowledgment. Initial window sizes are indicated at connection setup, but might vary throughout the data transfer to provide flow control. A window size of zero, for instance, means "Send no data."

In a TCP sliding-window operation, for example, the sender might have a sequence of bytes to send (numbered 1 to 10) to a receiver who has a window size of five. The sender then would place a window around the first five bytes and transmit them together. It would then wait for an acknowledgment.

The receiver would respond with an ACK = 6, indicating that it has received bytes [illegible] and is expecting byte 6 next. In the same packet, the receiver would indicate that its window size is 5. The sender then would move the sliding window five bytes to the right and transmit bytes 6 to 10. The receiver would respond with an ACK = 11, indicating that it is expecting sequenced byte 11 next. In this packet, the receiver might indicate that its window size is 0 (because, for example, its internal buffers are full). At this point, the sender cannot send any more bytes until the receiver sends another packet with a window size greater than 0.

4.10.4 TCP Packet Format

Figure 4.13 illustrates the fields and overall format of a TCP packet.

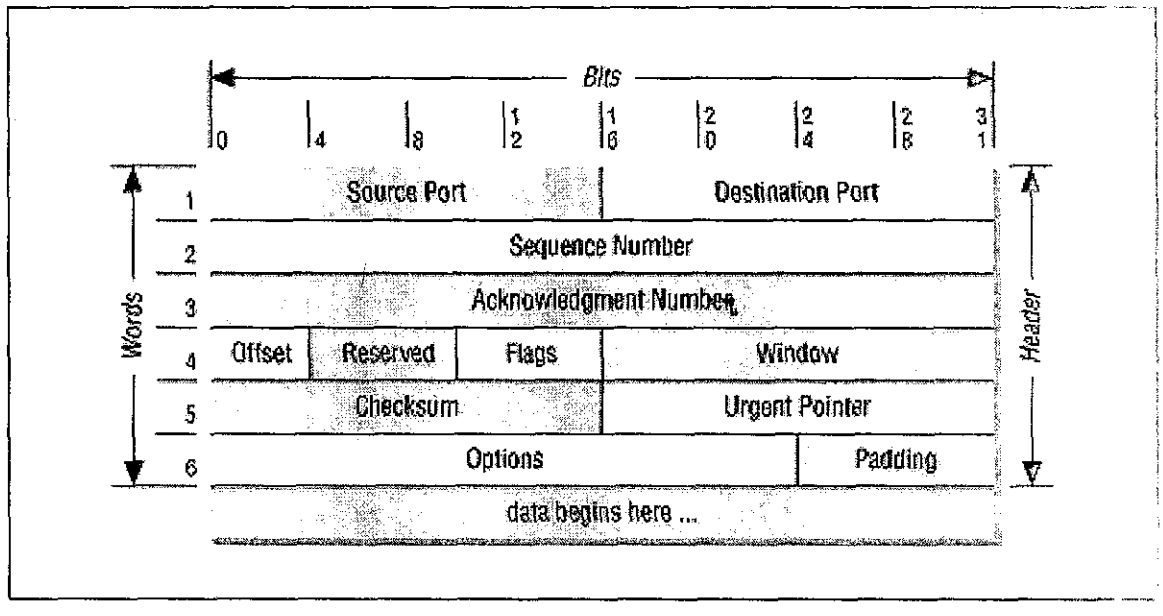


Figure 4.13: Format of a TCP packet

The description of the TCP packet fields is provided below:

- *Source Port* and *Destination Port:* Identifies points at which upper-layer source and destination processes receive TCP services.
- *Sequence Number:* Usually specifies the number assigned to the first byte of data in the current message. In the connection-establishment phase, this field also can be used to identify an initial sequence number to be used in an upcoming transmission.
- *Acknowledgment Number:* Contains the sequence number of the next byte of data the sender of the packet expects to receive.
- *Data Offset:* Indicates the number of 32-bit words in the TCP header.
- *Reserved:* Remains reserved for future use.
- *Flags:* Carries a variety of control information, including the SYN and ACK bits used for connection establishment, and the FIN bit used for connection termination.

- *Window:* Specifies the size of the sender's receive window (that is, the buffer space available for incoming data).
- *Checksum:* Indicates whether the header was damaged in transit.
- *Urgent Pointer:* Points to the first urgent data byte in the packet.
- *Options:* Specifies various TCP options.
- *Data:* Contains upper-layer information.

4.11 User Datagram Protocol (UDP)

The User Datagram Protocol (UDP) is a connectionless transport-layer protocol (Layer 4) that belongs to the Internet protocol family. UDP is basically an interface between IP and upper-layer processes. UDP protocol ports distinguish multiple applications running on a single device from one another.

Unlike the TCP, UDP adds no reliability, flow-control, or error-recovery functions to IP. Because of UDP's simplicity, UDP headers contain fewer bytes and consume less network overhead than TCP.

UDP is useful in situations where the reliability mechanisms of TCP are not necessary, such as in cases where a higher-layer protocol might provide error and flow control.

UDP is the transport protocol for several well-known application-layer protocols, including Network File System (NFS), Simple Network Management Protocol (SNMP), Domain Name System (DNS), and Trivial File Transfer Protocol (TFTP).

The UDP packet format contains four fields, as shown in Figure 4.14. These include source and destination ports, length, and checksum fields.

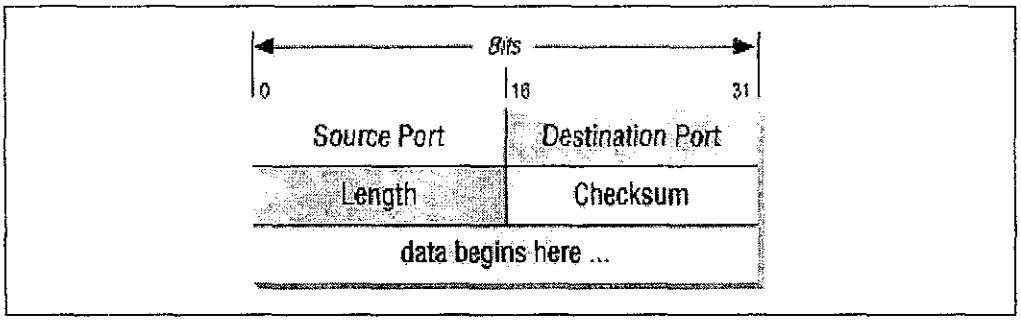


Figure 4.14: UDP packet format

Source and destination ports contain the 16-bit UDP protocol port numbers used to demultiplex datagrams for receiving application-layer processes. A length field specifies the length of the UDP header and data. Checksum provides an (optional) integrity check on the UDP header and data.

The purpose of each field in UDP datagram is provided in Table 4.6.

Table 4.6: Purpose of UDP Datagram fields

Field	Size in Bits	Purpose
Source port	16	The number of the calling port
Destination port	16	The number of the called port
Length	16	The length of the datagram
Checksum	16	Sum of header and data fields (error correction feature)
Data	-	Data from upper layers

4.12 Protocols and Ports

Once data is routed through the network and delivered to a specific host, it must be delivered to the correct user or process. As the data moves up or down the TCP/IP layers, a mechanism is needed to deliver it to the correct protocols in each layer. The system must be able to combine data from many applications into a few transport protocols, and from the transport protocols into the Internet Protocol. Combining many sources of data into a single data stream is called *multiplexing*.

Data arriving from the network must be *demultiplexed*: divided for delivery to multiple processes. To accomplish this task, IP uses *protocol numbers* to identify transport protocols, and the transport protocols use *port numbers* to identify applications.

Some protocol and port numbers are reserved to identify *well-known services*. Well-known services are standard network protocols, such as FTP and telnet, which are commonly used throughout the network. The protocol numbers and port numbers allocated to well-known services are documented in the *Assigned Numbers* RFC. UNIX systems define protocol and port numbers in two simple text files.

4.12.1 Protocol Numbers

The protocol number is a single byte in the third word of the datagram header. The value identifies the protocol in the layer above IP to which the data should be passed.

On a UNIX system, the protocol numbers are defined in */etc/protocols*. This file is a simple table containing the protocol name and the protocol number associated with that name. The format of the table is a single entry per line, consisting of the official protocol name, separated by whitespace from the protocol number. The protocol number is separated by whitespace from the "alias" for the protocol name. Comments in the table begin with #. An */etc/protocols* file is shown below Table 4.7:

Table 4.7: A Sample *protocols* file

```
% cat /etc/protocols
#ident "@(#)protocols 1.2    90/02/03 SMI"   /* SVr4.0 1.1   */

#
```

```
# Internet (IP) protocols
#
ip      0    IP     # internet protocol, pseudo protocol number
icmp    1    ICMP   # internet control message protocol
ggp     3    GGP    # gateway-gateway protocol
tcp     6    TCP    # transmission control protocol
egp     8    EGP    # exterior gateway protocol
pup     12   PUP    # PARC universal packet protocol
udp     17   UDP    # user datagram protocol
hmp     20   HMP    # host monitoring protocol
xns-idp 22   XNS-IDP # Xerox NS IDP
rdp     27   RDP    # "reliable datagram" protocol
```

The listing shown above is the contents of the */etc/protocols* file from a Solaris 2.5.1 workstation. This list of numbers is by no means complete. If you refer to the Protocol Numbers section of the *Assigned Numbers* RFC, you'll see many more protocol numbers. However, a system needs to include only the numbers of the protocols that it actually uses. Even the list shown above is more than this specific workstation needed, but the additional entries do no harm.

What exactly does this table mean? When a datagram arrives and its destination address matches the local IP address, the IP layer knows that the datagram has to be delivered to one of the transport protocols above it. To decide which protocol should receive the datagram, IP looks at the datagram's protocol number. Using this table you can see that, if the datagram's protocol number is 6, IP delivers the datagram to TCP. If the protocol number is 17, IP delivers the datagram to UDP. TCP and UDP are the two transport layer services we are concerned with, but all of the protocols listed in the table use IP datagram delivery service directly. Some, such as ICMP, EGP, and GGP, have already been mentioned. You don't need to be concerned with the minor protocols.

4.12.2 Port Numbers

After IP passes incoming data to the transport protocol, the transport protocol passes the data to the correct application process. Application processes (also called *network services*) are identified by port numbers, which are 16-bit values. The source port number, which identifies the process that sent the data, and the destination port number, which identifies the process that is to receive the data, are contained in the first header word of each TCP segment and UDP packet.

On UNIX systems, port numbers are defined in the */etc/services* file. There are many more network applications than there are transport layer protocols, as the size of the table shows. Port numbers below 256 are reserved for well-known services (like FTP and telnet) and are defined in the *Assigned Numbers* RFC. Ports numbered from 256 to 1024 are used for UNIX-specific services, services like **rlogin** that were originally developed for UNIX systems. However, most of them are no longer UNIX-specific.

Port numbers are not unique between transport layer protocols; the numbers are only unique within a specific transport protocol. In other words, TCP and UDP can, and do, both assign the same port numbers. It is the combination of protocol and port numbers that uniquely identifies the specific process to which the data should be delivered.

A partial */etc/services* file from a Solaris 2.5.1 workstation is shown below. The format of this file is very similar to the */etc/protocols* file. Each single-line entry starts with the official name of the service, separated by whitespace from the port number/protocol pairing associated with that service. The port numbers are paired with transport protocol names, because different transport protocols may use the same port number. An optional list of aliases for the official service name may be provided after the port number/protocol pair.

Table 4.8: A Sample *services* file

```
peanut% cat head -20 /etc/services
#ident "@(#)services   1.13   95/07/28 SMI"   /* SVr4.0 1.8   */

#
# Network services, Internet style
#
tcpmux      1/tcp
echo        7/tcp
echo        7/udp
discard     9/tcp       sink null
discard     9/udp       sink null
systat      11/tcp      users
daytime     13/tcp
daytime     13/udp
netstat     15/tcp
chargen     19/tcp      ttytst source
chargen     19/udp      ttytst source
ftp-data    20/tcp
ftp         21/tcp
telnet      23/tcp
smtp        25/tcp      mail
```

This Table 4.8, combined with the */etc/protocols* table, provides all of the information necessary to deliver data to the correct application. A datagram arrives at its destination based on the destination address in the fifth word of the datagram header. Using the protocol number in the third word of the datagram header, IP delivers the data from the datagram to the proper transport layer protocol. The first word of the data delivered to the transport protocol contains the destination port number that tells the transport protocol to pass the data up to a specific application.

Figure 4.15 shows this delivery process.

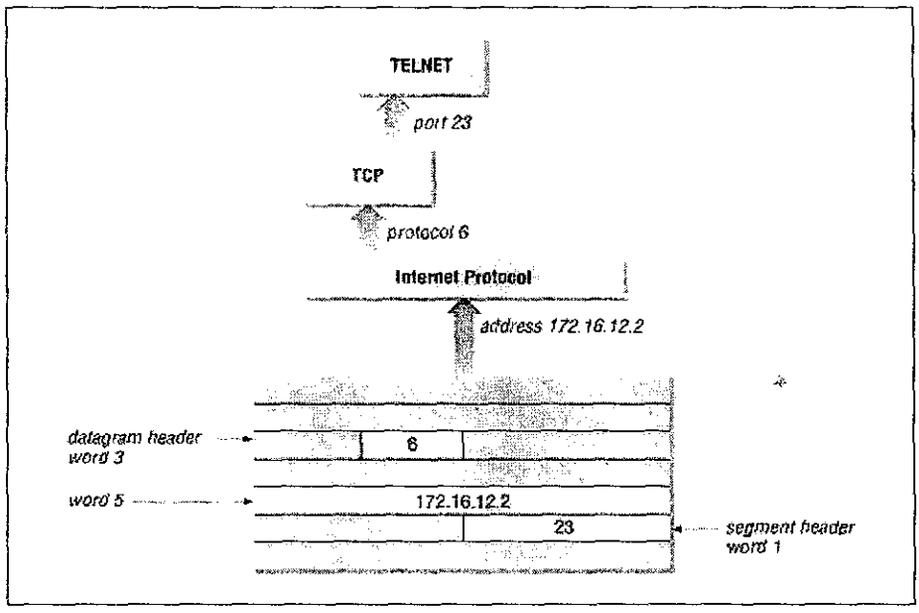


Figure 4.15: Protocol and port numbers

Despite its size, the */etc/protocols* file does not contain the port number of every well-known application. You won't find the port number of every *Remote Procedure Call* (RPC) service in the *services* file. Sun developed a different technique for reserving ports for RPC services that doesn't involve registering well-known port numbers. When an RPC service starts, it picks any unused port number and registers that number with the **portmapper**. The **portmapper** is a program that keeps track of the port numbers being used by RPC services. When a client wants to use an RPC service, it queries the **portmapper** running on the server to discover the port assigned to the service. The client can find **portmapper** because it is assigned well-known port 111. **portmapper** makes it possible to install well-known services without formally obtaining a well-known port.

4.13 Sockets

Well-known ports are standardized port numbers that enable remote computers to know which port to connect to for a particular network service. This simplifies the connection process because both the sender and receiver know in advance that data bound for a specific process will use a specific port. For example, all systems that offer telnet do so on port 23.

There is a second type of port number called a *dynamically allocated port*. As the name implies, dynamically allocated ports are not pre-assigned. They are assigned to processes when needed. The system ensures that it does not assign the same port number to two processes, and that the numbers assigned are above the range of standard port numbers.

Dynamically allocated ports provide the flexibility needed to support multiple users. If a telnet user is assigned port number 23 for both the source and destination ports, what port numbers are assigned to the second concurrent telnet user? To uniquely identify every connection, the source port is assigned a dynamically allocated port number, and the well-known port number is used for the destination port.

In the telnet example, the first user is given a random source port number and a destination port number of 23 (telnet). The second user is given a different random source port number and the same destination port. It is the pair of port numbers, source and destination, that uniquely identifies each network connection. The destination host knows the source port, because it is provided in both the TCP segment header and the UDP packet header. Both hosts know the destination port because it is a well-known port.

Figure 4.15 shows the exchange of port numbers during the TCP handshake. The source host randomly generates a source port, in this example 3044. It sends out a segment with a source port of 3044 and a destination port of 23. The destination host receives the segment, and responds back using 23 as its source port and 3044 as its destination port.

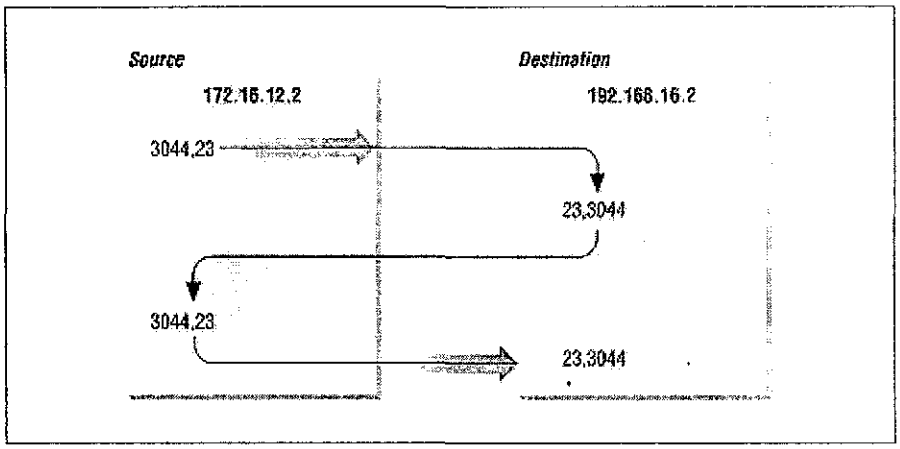


Figure 4.15: Passing port numbers

The combination of an IP address and a port number is called a *socket*.

A socket uniquely identifies a single network process within the entire Internet. Sometimes the terms "socket" and "port number" are used interchangeably.

In fact, well-known services are frequently referred to as "well-known sockets." In the context, a "socket" is the combination of an IP address and a port number. A pair of sockets, one socket for the receiving host and one for the sending host, define the connection for connection-oriented protocols such as TCP.

Let's build on the example of dynamically assigned ports and well-known ports. Assume a user on host 172.16.12.2 uses telnet to connect to host 192.168.16.2. Host 172.16.12.2 is the source host. The user is dynamically assigned a unique port number - 3382. The connection is made to the telnet service on the remote host which is, according to the standard, assigned well-known port 23. The socket for the source side of the connection is 172.16.12.2.3382 (IP address 172.16.12.2 plus port number 3382). For the destination side of the connection, the socket is 192.168.16.2.23 (address 192.168.16.2 plus port 23).

The port of the destination socket is known by both systems because it is a well-known port. The port of the source socket is known, because the source host informed the destination host of the source socket when the connection request was made.

The socket pair is therefore known by both the source and destination computers. The combination of the two sockets uniquely identifies this connection; no other connection in the Internet has this socket pair.

4.14 Transport Level Interface (TLI)

SCO OpenServer, SCO UnixWare 2.1.X, and UnixWare 7 all support the X/Open Transport Interface (XTI) and the Transport Level Interface (TLI). Both XTI and TLI are APIs that allow user processes to access transport providers in a mostly transport-independent fashion.

The Transport Level Interface was modeled after the industry standard *ISO Transport Service Definition* (ISO 8072). The resulting interface was called the Transport Level Interface (TLI) library, first introduced with AT&T UNIX System V Release 3.0 in 1986.

XTI is the X/Open-sponsored successor to TLI, and is defined in the *Networking Services, Issue 5* document.

From the standpoint of syntax and semantics, the two libraries are nearly identical, but XTI is the preferred interface for new applications, since it is the industry standard. The TLI library routines are maintained for compatibility with previous releases only.

The UnixWare 7 transport level functions are defined in **libnsl** and support both the older TLI semantics and the newer XTI semantics. The library entry points have their traditional names for the TLI functions, such as **t_open**. For the XTI functions, however, the entry points have new names, such as **_xti_open**. When applications are compiled on UnixWare 7, the TLI function names are translated to the XTI entry points by macros by including the file *xti.h* in the source code.

In this way, UnixWare 7 is able to support older applications that use the TLI interface, while still providing the new XTI interface.

To compile and link a program using XTI semantics, do the following:

1. Include the xti.h header file at the beginning of your source files. The syntax of the include preprocessor directive to use is:
2. #include <xti.h>
3. Specify the libnsl library as one of the libraries to be searched on the cc command line:
4. cc option file -lnsl

To compile a program using the TLI interfaces exclusively, include the TLI header file (tiuser.h) instead of the XTI header file. The syntax of the include preprocessor directive to use is

#include <tiuser.h>

4.15 Routing Algorithms

The purpose of the routing algorithm is to be the decision-maker for the router regarding the best paths for data transfer. It takes data(metrics) supplied by a particular routing protocol to determine the most suitable path for data to travel from the source to the destination via interconnected networks.

Routing algorithms are inherent in routing protocols. In general, a routing algorithm will take in values laid out in a routing table produced by the routing protocol, then proceed to determine the quickest route for data transfer. Different protocols does not mean that they have different algorithms, but it may suggest that different metrics are used by the algorithm for it's calculations.

The choice of routing algorithm is the single most important consideration for the overall performance of a network. However, you cannot directly choose the algorithm that your router uses, it is the the choice of routing protocol that determines which algorithm will be used.

Since the algorithms have such a major impact on the performance of the network, one needs to have a solid understanding in how different types of routing algorithms work before implementing a particular protocol that comprises that type of algorithm.

To illustrate this, consider a simple analogy where an internet search engine is the routing algorithm and the user is the routing protocol. Suppose the user(routing protocol) wants to search for information related to 'toshiba laptops', and she just submits 'laptops' to the search engine(routing algorithm), she would know that the results would not be as appropriate to her as those returned from submitting 'toshiba laptops' as the search criteria. So this suggests that having a good understanding in the way the search engine(routing algorithm) reacts to her search criteria, will help in determining the most appropriate routing algorithm to use and consequently the correct routing protocol to use.

4.15.1 Characteristics of Routing Algorithms

Routing algorithms will ideally have one or several of the following characteristics.

- Optimality
- Simplicity and low overhead
- Robustness and stability
- Rapid convergence
- Flexibility

These characteristics are described below.

Optimality

This is the characteristic that refers to the algorithm's capability to produce the optimal route from a set of metrics. However, the routing protocol must also do their part in defining strict metric calculation algorithms that produce reliable metrics for the routing algorithm to use.

Simplicity and low overhead

This is the characteristic that refers to the algorithm's efficiency. Efficiency in this sense refers to getting the work done with the minimal use of system resources, in other words, minimal overhead. That is, the utilisation of software and hardware resources are minimal.

Robustness and stability

This is the characteristic that refers to the algorithm's defensiveness to unexpected and stressful situations. Hardware failures, high data load conditions and incorrect implementations on the network all exemplify such situations. The algorithm's defense against such situations means that it will give appropriate alerts and perform other procedures to get around the problem.

Rapid convergence

This is the characteristic that refers to the algorithm's ability to quickly produce the optimal route so that the router it resides in can quickly distribute routing update messages over the networks. This is so all routers will have the latest routing information that stimulate recalculation of the next set of optimal routes. Routing algorithms that converge slowly can cause time-consuming routing loops or network outages.

Flexibility

This is the characteristic that refers to the algorithm's scalability in the face of different network circumstances. It can be programmed to adapt to changes in network bandwidth, router queue size, and network delay, among other variables. Suppose a portion of a network is dysfunctional, the algorithm will quickly determine the next best route to overcome the problem.

4.15.2 Criteria for Routing Algorithms

Metrics are a unit of measure for each link connecting devices on a network. They are used by the routing algorithms for determining the most 'suitable' route for data to travel. The simplest routing algorithms use one form of metric while more sophisticated routing algorithms can base route selection on multiple metrics, combining them into one 'hybrid' metric. The following are certain metrics used by routing algorithms:

- Path Length
- Reliability
- Delay
- Bandwidth
- Load
- Communication costs

These are described below.

Path Length

Path length could be used to refer to the hop count, which is the number of times a data packet crosses a router or some other networking device in the internetwork to go from the source to the destination machine. Alternatively, path length could refer to the total cost of network links travelled if each network link was to be assigned a cost value. The most preferred route would be the one with the lowest hop count or total cost.

Reliability

This metric concerns the reliability rating of each network link. Factors such as the bit-error rate of each link and any similarly related factors are taken into account to give the link a reliability rating. The route defined to be most reliable is the chosen one.

Delay

This metric concerns the time-length for a data packet to travel from source to destination. Several factors that add up to give the delay include the bandwidth of intermediate network links, the port queues at each router along the way, network congestion on all intermediate network links, and the physical distance to be travelled. This metric takes into account many network variables that is why it is very useful. Obviously, the route with the lowest delay is preferred.

Bandwidth

This metric refers to the available traffic capacity of a link. Generally, a link with greater bandwidth is the preferred link for data transfer. However, suppose data travels over a large capacity link of 10Mbps that is extremely busy, the data could actually travel slower than a link of smaller capacity but with a light load on it.

Load

This metric refers to the degree to which a router is busy, or any other network device in fact. Load can be measured in varying ways, such as in terms of CPU utilization or the number of data packets processed per second. The preferred route would naturally be the one requiring the least load on the network.

Communication Costs

This metric refers to the monetary costs of transmission mediums over which data is traversed. There are times when network performance are a lower priority than network expenditure and this is when the route with the lower costs associated with it is preferred.

4.16 Types of Routing Algorithms

Following are some important routing algorithms:

1. Link-state versus distance vector Algorithms
2. High Performance Algorithms
3. General Categories of Algorithms
    a. Static vs. Dynamic
    b. Single path vs. multi-path
    c. Flat vs. hierarchical
    d. Host-intelligent vs. router-intelligent
    e. Intra-domain vs. inter-domain

All of these algorithms are discussed in the following sections.

4.16.1 Link-state (LS) vs. Distance Vector (DV) Algorithms

Distance Vector Algorithm

A distance vector algorithm uses metrics (or parameters) known as costs (includes delay, reliability, etc) to help determine the best path to a *destination.* The path with the lowest total cost is chosen as the best path.

Essentially, distance vector algorithms calculates the *end to end* paths for a data packet then determines the best (usually lowest cost) path from these, which it then broadcasts to its neighbours only.

An example of a popular distance vector type algorithm today is BGP (Border Gate Protocol).

Link-State Algorithm

Once a router receives a data packet, it creates an *LSA (Link State Advertisement),* which is then *flooded* to all the direct interfaces of that router. In doing so, all routers will have complete knowledge of the topology of the area in which it belongs so that the algorithm can calculate all the paths from one node to another node that is directly connected, then determines which of these is the *intermediate* 'best path' to where the packet wants to *eventually end up* at.

- Node refers to a network or router.
- *Intermediate* is referring to node-to-node links (e.g. router to router).
- 'Best path' usually means 'least cost' where cost can refer to metrics such as hop count, bandwidth, delay and so forth.

Any one router on a network will only have direct knowledge of the routers and networks that are directly connected to it (or in other words, the *state* of its own links). So unlike the distance vector algorithm, the link-state algorithm usually computes much less complex paths thereby converging quicker and saving system resources such as processor use.

Internet currently uses the link-state type routing. This is also referred to as Dijkstra's algorithm

4.16.2 High Performance Algorithms

Some currently implemented algorithms employ techniques that hinder performance greatly in certain situations. A prime example is the growing demand in performance for very large distributed systems such as Internet file sharing system Napster.

The problem with these systems is that they do not facilitate scalability well and thus as traffic increases, performance drops dramatically. Recent development and research in this area have come up with new algorithms that facilitate increasing traffic flow and a prime example of these is referred to as DHTs (Distributed Hash Tables).

In these DHTs:

- Files are associated with a key (produced for instance by hashing the file name) and each node in the system is responsible for storing a certain range of keys.
- There is one basic operation in these DHT systems, *lookup (key)*, which returns the identity (e.g. the IP address) of the node storing the object with that key.
- This operation allows nodes to *put* and *get* files based on their key, thereby supporting the hash table like interface.

Due to the intense developments in the area of distributed application development, DHTs should pave the way for future developments that are robust and scalable.

4.16.3 General Categories of Algorithms

Static vs. Dynamic Algorithms

Static implies 'unchanging' and when applied to algorithms suggest unchanging routes. These routes are predefined by the network administrator prior to the start of routing, and stay that way unless she changes them. This type of algorithm is suited to small network environments that don't change frequently.

Dynamic implies 'always changing' and in terms of algorithms, it suggests that they analyse incoming routing information about the network environment and calculates the best routes every time. In this way, network changes are catered for and data can be successfully routed to appropriate nodes.

Dynamic routing algorithms can complement static routes where appropriate. For example, a router of last resort (a router to which all unroutable packets are sent), can be designated to act as a repository for all unroutable packets, ensuring that all messages are at least handled in some way.

Single path vs. Multipath Algorithms

Multipath implies multiplexing over multiple lines. In other words, mulitple channels of data can traverse on the same path over multiple paths. These types of algorithms are generally inherent in more

sophisticated routing protocols that pushes the limits of throughput as well as providing network reliability.

Flat vs. Hierarchical Algorithms

In a hierarchical routing system, there exists a 'backbone' of routers that acts like an interface for routers that are not directly connected to each other. Packets from a source are sent to backbone routers which sends it to the other appropriate non-backbone routers until the destination is reached. Whilst for a flat routing system, all routers are peers to each other and there are no sets of backbone routers to which other routers must interface with.

Commonly, routers and network devices are gathered into groups. These groups are referred to as domains, areas or AS (autonomous systems). For hierarchical routing systems, these domains contain backbone routers that interfaces with other domain's backbone routers. Often only these backbone routers can communicate with routers outside a particular domain, non-backbone routers are limited to communication within the domain. This is usually for the purpose of reflecting the business' architecture that consists of many departments. In structuring the network this way, the business' network traffic loads can be supported and managed. Such a situation is when the majority of the traffic is within the domain networks. Routing algorithms can deal with less complex routes thereby effectively improving the networks performance.

Host-intelligent vs. router-intelligent

In host-intelligent algorithms, the data packet's route from source to destination is assumed to have been determined by the hosts. In effect, the routers are just intermediate repositories that forwards data until the data reaches it's destination without considering the effects of network changes, thus it can be very data unreliable.

On the other hand, router-intelligent algorithms assume that the hosts are ignorant or the routes. It is the routers themselves that determine the paths for data to be transmitted on. They calculate the best routes after having received appropriate updates on the status of the network and are thus more robust and less sensitive to network changes.

Intra-domain vs. inter-domain

Intra-domain algorithms are designed to work within domains where as inter-domain algorithms are designed to work between domains. Both types have their design traits that make them efficient for the particular situation (intra-domain or inter-domain) and are thus not interchangeable.

4.17 Routing Strategies

In most packet-switched networks there are multiple paths a packet could take from source to destination. A *routing algorithm* is used to select a path through the network.

The routing algorithm is critical to the smooth operation of the network, because it is the primary means by which network traffic is controlled. If the routing algorithm is not properly designed it can contribute

to *network congestion*–excessive packet "traffic jams" at certain parts of the network, which can negatively impact any other packet traffic that must pass through that area.

There are two approaches to routing:

- *Adaptive routing and*
- *Non-adaptive routing*

The latter type make routing choices in isolation (without taking into account the current state of the network); the former type try to work around and minimize congestion by exchanging information with neighboring nodes about network conditions.

Another factor for routing is whether a route is calculated statically or dynamically for individual packets. In the case of virtual circuits, some networks set up the route when the VC is established and not for individual packets in the VC; others can dynamically reroute any packet.

4.17.1 Non-Adaptive Routing Strategies

Fixed Routing

Fixed routing tables are computed by finding the minimum *hop-count* (number of nodes traversed from source to destination) or *cost* (a generalized measure of the "cost" of a link, which can reflect throughput, expense, utilization, etc.). There are algorithms for determining the least-cost path through a network based on these criteria.

These algorithms produce a routing matrix that specify the preferred next hop given a current node (the node holding the packet) and a destination node. Each node only needs to store one column of the matrix (the column using itself as the "current node"), this column can be stored as a table or database

When a packet enters a node, the node extracts the packet's destination address from the packet header. This address is an index into the node's routing table or database; the record at the index contains the preferred node to which the packet should be forwarded.

Generally these tables are updated only when the network topology changes (i.e. a node is added, or at most at periodic intervals, like twice a day).

Pros: The algorithm is simple, fast, correct (provided the information is accurate), and fair.

Cons: Bad robustness; this scheme doesn't adapt to changing conditions such as heavy congestion or node failures. One option is to supply additional (backup) routes in the table to help robustness

Flooding Routing

Another routing possibility is to have each node forward packets on to all possible outputs (except the link that it came in on), a process known as *flooding*. The basic idea is that the packet is bound to find a valid route to the destination. While this is true, it also leads to an exponential explosion in duplicated packets in the network.

Nevertheless, the technique is sometimes used because of its interesting ramifications. Because all routes are tried, there will be one with a minimal hop count that gets through. If the route can be recorded, a single "set-up" packet can be used to establish a static route for a virtual circuit (VC). Also, flooding can be used to disseminate important information throughout the network, such as emergency messages, routing information, or broadcasts. This is possible because under flooding a packet will traverse every known node connected with the source node either directly or indirectly.

There are two main problems with flooding.

- First,
  - The receiver must know how to discard multiple copies of the same packet, several of which are likely to arrive at the destination.
  - This can be handled with an appropriate sequence or unique identification number in the packet.
- Second,
  - There must be a way to keep circularities in the network graph from letting the packet cloning explosion from going on unchecked and quickly saturating the network.
  - The simplest solution to this problem is to use a *hop count*. A hop count is a numerical field in the packet that gets decremented whenever the packet is forwarded.
  - The hop count is examined before forwarding, and if the hop count has reached zero the packet is discarded.
  - The hop count is initialized to some reasonable value (such as the maximum of all minimum-hop routes in the network).

Pros: The algorithm is fair, simple, and robust.

Cons: Very high traffic, overhead, inefficiency

Random Routing

A third simple routing scheme is to have each node randomly choose a link on which to forward the packet (except the source of the packet, as with flooding). This is similar in effect to flooding, without the attendant explosion in packets.

Pros: Fair, good (random) traffic distribution, simple, somewhat robust.

Cons: The route taken is not optimal, and so the network must bear a greater load of traffic; it also does not try to avoid congested areas or downed links, although the random behavior will eventually allow packets to find a way to their destination.

4.17.2 Adaptive Routing Strategies

Adaptive routing describes routing strategies that adapt to changing network conditions. We would like each node's route algorithm to avoid choosing paths that lead to downed links and network congestion points.

In order to be able to do this, a node must have information about the network, and be able to update this dynamically. To be reasonably responsive, it must update this info frequently, otherwise it is not much better than a fixed routing scheme. This means that nodes must measure local network conditions and communicate this information, which adds additional traffic to the network and overhead and complexity to the routing algorithm.

Two general communication strategies exist:

- Distributed routing, in which each node just communicates with its neighbors and relies on info propagation to "spread the word", and
- Centralized routing, in which each node communicates with a central node, which recalculates and distributes the routing information.

The distributed algorithm is more common, as it avoids the obvious bottleneck of a central information server.

What kind of information can be exchanged to determine network conditions?

- Packet delay is the most useful single metric for determining both link failures and congestion. A downed link will have infinite packet delay and a congested link will have a very high packet delay.

What is the easiest way to measure packet delay?

- It can be approximated, for example, by measuring the outgoing queue length for a link, or it can be measured directly with suitable timing facilities. It is the sum of the time that a packet waited in a queue on a node plus the time spent transmitting it to the next node.

As an example of routing strategies, lets examine the Internet routing algorithm. It has been updated at least three times to improve congestion avoidance.

First-generation Internet Routing

The first Internet routing algorithm was based on the routing tables as described under fixed routing above, except that the tables could be altered dynamically. Besides showing the next link to take for each possible destination address, the table also recorded an approximation of the packet delay to that destination.

Periodically, neighbor nodes would exchange tables, which allowed a node to update it's own tables and recalculate the delay to each destination based on the neighbor data plus any knowledge of the delay to the neighbors. If it found a shorter delay to any particular destination through an alternate neighbor link it would change the routing table entry to point to the new route.

The major shortcoming of the first-generation protocol was that a link's delay was approximated by measuring the outgoing queue length for that link. This was not a necessarily accurate view of the delay because it didn't take into account link speed, etc. The algorithm as a whole seemed to respond rather slowly to congestion situations.

Second-generation Internet Routing

After 10 years of experience using the old routing scheme, a new one was implemented to address the problems outlined above. The major changes were twofold:

1. Packet delay was measured directly, using knowledge of link propagation time and data rate.
2. Every 10 seconds, the average packet delay for the link was calculated. If the difference from the last average was significant, *the data was sent to all nodes using flooding*. Each node maintained a complete routing table with delay estimates, and could recompute the optimal route to a destination from it's own address.

This technique provided quicker response to network congestion since every node got notice of significant changes anywhere in the network.

Third-generation Internet Routing

The second generation routing system was overhauled for a third time 8 years later as the Internet begin to grow significantly. The major problem identified was that the second-generation algorithm was working *too* well; all nodes were adapting to the global network knowledge by trying to find the optimal route.

This behavior leads to network congestion oscillations. In essence, once part of the network gets congested all of the nodes in that area begin to reroute traffic to a lighter, less congested, area. Then the lighter area gets congested and all of the nodes reroute traffic back into the originally congested area, which has since become lighter. The scenario then repeats.

Rather than change the data propagation algorithm, the designers decided to change the measure of delay to a measure of *utilization* and to smooth out the values as well by averaging over greater periods. This had the effect of dampening the network oscillations.

4.18 Congestion Control Algorithms

Congestion is nothing but the degradation of performance because of the presence of too many packets in the subnet. When there are many packets arriving on some input lines & they all need same output line, there will be congestion and the packets will be lost. To overcome this memory can be allocated but this helps until some point after that adding memory worsens the congestion. Slow processors can cause congestion.

Congestion prevention policies applicable to different layers are outlined below:

Transport Layer

- Retransmission policy
- Out of catching policy
- Acknowledgement policy
- Flow control policy
- Timeout determination

Network Layer

- Packet requiring & service policy
- Packet discard policy
- Routing algorithm
- Packet lifetime management
- Virtual circuits v/s datagram inside the subnet

Data Link Layer

- Retransmission policy
- Out of order catching policy
- Acknowledgement policy
- Flow control policy

One of the causes of congestion is bursty traffic. If the hosts transmit at uniform rate, congestion would be less common. Regulating the average rate of data transmission is called Traffic shaping

The two algorithms related to traffic policing are:

- Leaky Bucket algorithm and
- Token Bucket algorithm.

These are discussed below:

Leaky bucket algorithm

Leaky bucket algorithm allows the constant outgoing flow irrespective of the rate of inflow the input queue is not filled completely. When the input queue is full this algorithm discards the input packet. every clock tick one packet is transmitted.

Token bucket algorithm

Token bucket algorithm is used because we want output flow to speed up when the large bursts arrive which is not possible in leaky bucket as it allows only constant rate of outflow. This algorithm throws away tokens when buckets fills up but never discards packets.

A packet can only be transmitted if enough tokens are available to cover its length in bytes. Fractional tokens are kept for future use. Each token represents the right to send not one packet, but in k bytes.

4.19 Congestion Control Method For Virtual Circuits And Datagram

For Virtual circuit following methods are used.

a. Admission control: In this method once the congestion has been signaled, no more virtual circuits are set up until the problem is solved.

b. Otherwise we can negotiate an agreement between the host & subnet when a virtual circuit is set up. This agreement covers the volume and shape of traffic, quality of service required.

Choke Packets

Whenever the level of traffic moves above the threshold, th eoutput line enters a warning state . each newly arriving packet is checked if its output line is in warning state. If that is the case then the router sends a choke packet to the source.This is the implicit feedback which helps control the congestion in the network.

Weighted Fair Queuing

When a chock packet is sent back to the source host, it is quite possible that only that source reduces its rate of transmitting data but other routers keep blasting the data in the network. Hence we need to implement this technique. This method scans the queues repeatedly, until it finds the tick on which each packet will be finished. The packets are then sorted in order of their finishing and sent in that order. In the absence of new arrivals the packets will be sent in the order listed.

Hop by Hop Choke packets

When there are number of hops involved between source & destination, the time taken by the choke packet to reach the source host is considerably large & does not serve the purpose of congestion control. Hence choke packet is sent to the router encountered lately & request it to send the data at lower rate. This router sends the choke packet to its nearest router on the path. This phenomenon continues until choke packet is sent to source.

Load Shedding

In this method the router discards the additional packets those come after the congestion takes place. Now which packet to reject depends on the type of service being served? In multimedia new packets are more important compared to old packets whereas in file transfer the old packets are more important. Hence the approach in keeping new packets is called Milk Method whereas the one in keeping the old packets is called Wine method.

Jitter Control

In audio & video applications, the transit time of the packets should be constant otherwise there is jitter. There are methods available for jitter control if the packet is lagging behind the router gives it a passage to go quickly or if the packet is leading it is made to wait so that old packets catch up with this packet.

4.20 Multiplexing

Efficiency is affected by under utilization of the link. Sliding-window protocols allowed better utilization, *provided that there are more frames to send.* It is often the case that a networking application does not require the full bandwidth or data rate provided by the link.

Multiplexing is a way to allow a number of separate channels (links) to share the bandwidth provided by the transmission medium. This is most often used for long-distance, high-capacity transmission media like fiber, coaxial or microwave. For example, voice communications can be carried perfectly adequately in 4 KHz of bandwidth. Long-distance telephone companies routinely carry thousands of calls on a fiber optic cable (which has immense bandwidth) through the use of multiplexing techniques.

4.21 Types of Multiplexing

There are several types of multiplexing techniques:

- Frequency division multiplexing
- Time division multiplexing

These are discussed in the following sections.

4.21.1 Frequency Division Multiplexing

Frequency division multiplexing is used with analog signalling techniques. Each input signal is fed into a multiplexer (mux) which modulates it onto a different carrier frequency from all the others; the result is that each modulated signal occupies a separate, non-overlapping spectrum.

It is important that each modulated signal (i.e. *channel*) have a different spectrum; if there is any frequency overlap the signals will interfere with each other and the data could be scrambled this is called *crosstalk* noise. For this reason the carriers are usually spaced far enough apart (in frequency) so that there is some unused spectrum in between channels.

The output signal is a composite analog signal containing all of the modulated signals. It has a bandwidth that is at least the sum of the bandwidths of the modulated signals, plus any unused spectrum in between channels.

The demultiplexer (mux) uses bandpass filters to filter out (separate) each modulated signal and then each one can be demodulated into the original data signal, which may be digital or analog.

A well-known example of this technique is FM radio.

FDM has been used to multiplex analog voice signals onto long-distance telephone lines. It is expensive to lay cable or set up microwave links and so the long distance carriers look for ways to share a single line for many users.

FDM is still used in many instances, but many telecommunications and computer networking standards are shifting to a multiplexing technology that is used to produce digital output signals–TDM.

4.21.2 Time Division Multiplexing

Time division multiplexing (TDM) multiplexes several inputs onto a single output by dividing the transmission medium into time blocks rather than frequency blocks. There are two main types of TDM:

1. Synchronous TDM
2. Asynchronous TDM

Synchronous TDM

The inputs to a TDM mux are usually buffered digital signals. A *scan* operation takes a bit or character from each input and merges them into the output signal. One set of bits or characters from each source is called (unfortunately) a *frame*. The scan must run fast enough to produce an output rate that is N times the individual data rates for N inputs (i.e. the sum of the input rates). There is usually a little extra overhead allotted for the scan so the multiplexed rate will be slightly higher than the sum of the individual data rates.

Each bit or character in the frame is a *slot*. If there is nothing to send for a particular input the corresponding slot is empty. This is the origin of the "synchronous" label–each channel is tied to a slot. This is obviously inefficient if some data sources are intermittent (almost all network activity is bursty in nature).

Data link control is handled on a per-channel basis.

Each device connected to a channel appears to have an independent link; the mux has no knowledge of the contents of a channel and a channel has no knowledge that it is operating over a mux. Having the data link protocol operate on individual channels rather than on the TDM link itself means that flow and error control happening on one channel do not affect any of the other channels. It also means different link protocols can be used on different channels if desired.

A full-duplex link is usually used between the TDM mux and demux, providing full-duplex channels.

Asynchronous (Statistical) TDM

Network data transfers between computers often happen in bursts, rather than continuously. If there is nothing to send on a particular input stream, a synchronous TDM will send an "empty" slot, since each slot in a frame is reserved for a particular channel. Now if the majority of the bit traffic on the input streams is bursty, then statistically there are going to be a lot of empty slots on the output stream, which means the link is being underutilized, and data link capacity is being wasted.

If we could use some of the empty slots on a TDM link to carry extra bits from busy channels, it would increase the data rate on those channels (at least temporarily) and we might be able to fully exploit the capacity of the link. Alternatively, we could carry more channels on the link, since it is unlikely that all of them will be transmitting at once.

This is what statistical TDM does. It dynamically allocates slots to input channels to "balance" the load across the inputs, trying to minimize the number of empty slots. This requires more sophisticated logic

at the mux and demux ends, because frames no longer have a simple fixed format: a particular slot can "belong" to many different channels, depending on the state of the inputs to the mux. This also adds overhead to the TDM link protocols to keep track of the slot allocations.

Student Activity

1. What do you mean by Internet Protocols?
2. What is the significance of IP addressing?
3. What is Internet routing?
4. What is the purpose of ICMP?
5. Discuss the significance of IDRP protocol.
6. What is the purpose of ARP?
7. What are sockets?
8. What are various routing algorithms?
9. Outline different routing strategies.
10. What are various types of multiplexing? Discuss.

Unit 5

Application Layer Services and Protocols

Learning Objectives

After reading this unit you should appreciate the following:

5.1 Introduction

5.2 Application-Layer Internet Protocols

5.3 Telnet

5.4 File Transfer Protocol

5.5 Types of FTP Servers

5.6 Working With FTP server

5.7 Simple Mail Transfer Protocol

5.8 Simple Network Management Protocol

5.9 Domain Name Service

5.10 NFS and RPC Protocols

5.11 XDR Protocol

5.12 X Windows Protocol

5.1 Introduction

The application layer interacts with software applications that implement a communicating component.

Application layer functions typically include the following:

- Identifying communication partners -- The application layer identifies and determines the availability of communication partners for an application with data to transmit.
- Determining resource availability -- The application layer must determine whether sufficient network resources for the requested communication are available.
- Synchronizing communication -- Communication between applications requires cooperation that is managed by the application layer.

The application layer is the OSI layer closest to the end user. That is, both the OSI application layer and the user interact directly with the software application.

5.2 Application-Layer Internet Protocols

The Internet protocol suite includes many application-layer protocols that represent a wide variety of applications, including the following:

- *File Transfer Protocol (FTP)*—Moves files between devices
- *Simple Network-Management Protocol (SNMP)*—Primarily reports anomalous network conditions and sets network threshold values
- *Telnet*—Serves as a terminal emulation protocol
- *X Windows*—Serves as a distributed windowing and graphics system used for communication between X terminals and UNIX workstations
- *Network File System (NFS), External Data Representation (XDR), and Remote Procedure Call (RPC)*—Work together to enable transparent access to remote network resources
- *Simple Mail Transfer Protocol (SMTP)*—Provides electronic mail services
- *Domain Name System (DNS)*—Translates the names of network nodes into network addresses

Table 5.1 lists these higher-layer protocols and the applications that they support.

Table 5.1: Higher-Layer Protocols and Their Applications

Application	Protocols
File transfer	FTP
Terminal emulation	Telnet
Electronic mail	SMTP
Network management	SNMP
Distributed file services	NFS, XDR, RPC, X Windows

5.3 Telnet

Telnet is a protocol that enables one computer to connect to another computer. This process is referred to as remote login. The user's computer, which initiates the connection, is referred to as the local computer, and the machine being connected to, which accepts the connection, is referred to as the remote, or host computer. The remote computer can be physically located in the next room or on the other side of the world. Parts of the world.

To use a TELNET client, you tell the client the name (address) of the remote host to which you want to connect. The client in turn will connect by establishing a TCP connection to port 23(the well-known port

for TELNET). Then the remote host will present you with a login prompt at which you can type your username. Once connected, the user's computer emulates the remote computer.

When the user types in commands, they are executed on the remote computer. The user's monitor displays what is taking place on the remote computer during the telnet session. The procedure for connecting to a remote computer will depend on how your Internet access is set up. Once a connection to remote computer is made, instructions or menus may appear.

The Telnet protocol is used to establish an online connection to a remote machine. It is used in the same way as the Telnet program, which is not surprising, since most browsers just call the Telnet program as a helper application. We can use TELNET client programs to log into a remote host as though your computer is a terminal attached directly to the remote host's network.

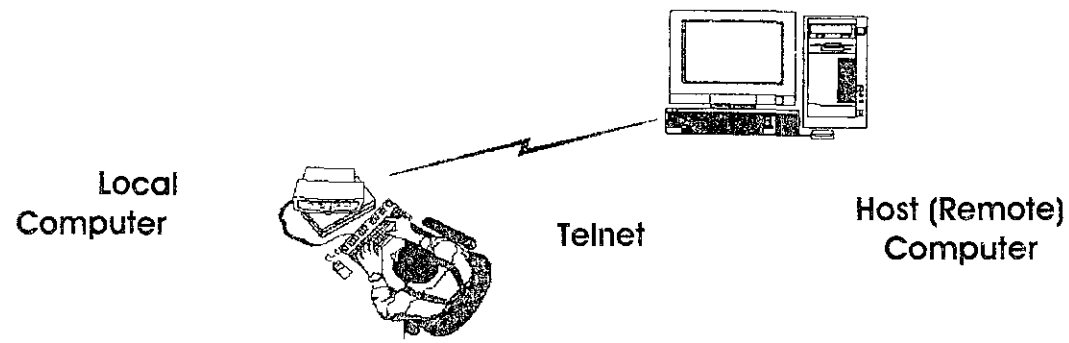


Figure 5.1: Remote Login

Some remote machines may require a user to have an account on the machine, and may prompt users for a username and password. Telnet also operates on the client/server principle. The local computer uses a telnet client program to establish the connection and displays data on the local computer's monitor. The remote or host computer uses a telnet server program to accept the connection and send responses to requests for information back to the local computer.

A variety of resources are available through telnet. For example, library catalogs, databases, other Internet tools such as FTP, Gopher and the World Wide Web, etc. To summarize, the Internet standard for remote login service is found in a protocol known as TELNET; its specification is part of the TCP/IP documentation. The TELNET protocol specifies exactly how a remote login client and a remote login server interact. The standard specifies, for example, how the client contacts the server and how the client encodes keystrokes for transmission to the server. As both the TELNET client and server programs adhere to the same specification, they agree on communication details.

Telnet allows you to connect to and log into a remote computer. You can access any of the public services or tools at the remote site. Telnet can be used for access to libraries, databases, and other Internet services.

5.4 File Transfer Protocol

The FTP protocol is used to access files by FTP, the Internet file transfer protocol. FTP has been around for more than two decades and is well entrenched. Numerous FTP servers all over the world allow people anywhere on the Internet to login and download whatever files have been placed on the server. It is possible to access a local file as a web page, either by using the file protocol or simply, by just meaning it.

For example the news protocol allows a web user to call up a news article as though it were a web page. This means that a web browser is simultaneously is a newsreader. In fact, many browsers add extra menu items to make reading. USENET news even easier than using standard newsreaders. Two formats are supported for the news protocol. The first format specifies a newsgroup and the second format allows the identifier of a specific news article to be given. The browser than fetches the given article from a pre configured new site using the NNTP protocol.

FTP controls most file transfer operation across the Internet. As you will learn, FTP uses two TCP connections to perform file transfer operations. FTP uses the first TCP connection to transfer commands and reply similar to those SMTP (Simple mail transfer protocol) and POP3 protocol. However, FTP use the second TCP connection for data transfer operations

5.4.1 File Transfer Protocols

Trivial file transfer protocol (TFTP) intentionally omits most of FTP's capabilities and simply focuses on performing two files transfer operation, reading a file and writing a file from server operations. TFTP uses the UDP (user data gram protocol).

Like all Internet services, FTP uses a client/server system. You run a client program on your computer which connects, to a server program on a remote computer. When you copy a file from a remote computer to your own computer, we say that you are downloading the file. When you copy a file from your computer to the remote one, you are uploading the file. In FTP terminology your computer is called the Local Host and the other computer is called the Remote Host. Thus, we can say that FTP is used to upload and download files to and from a remote host.

Like FTP, Web clients request files from Web servers. Using Hypertext Transfer Protocol (HTTP), Web clients (browsers) request files from Web browsers (Web Sites). The big difference is that FTP traffic goes both ways: "up" to the server, as well as "down" to the client. Note that FTP communication is always between clients and servers, never between clients. Just because two people have FTP software does not mean they can exchange files with each other. One person might send a file to the server and the other person might then download that files, but a server must still be in the middle.

5.5 Types of FTP Servers

As with Web servers, the Internet contains thousands of FTP servers. Many organizations that run Web servers also run FTP servers, which they use to handle the distribution of various files for example product documentation, or data files. Often, when visitors to a Web site click a link to download something, the link actually redirects the visitor's browser to an FTP server. FTP servers are usually independent of the Web, and supply other files besides those listed on Web pages. In general, most files on FTP servers support the needs of the most technical users of the Internet, such as network and software developers. Files on FTP servers can be accessed through:

- Private FTP Service
- Public FTP Service

5.5.1 Private FTP Service

Some files on FTP servers are accessible only by private users such as customers or members of the organization that run the server. It requires that you sign on to the remote host with your own account or ID and password, which is assigned by the server to you, and gives you various degrees of privileged access to private folders (directories).

5.5.2 Public ("Anonymous") FTP Service

Some files on FTP servers may be accessible to the general public. On many FTP servers of educational and non-profit organizations, public access is a service only due to the generosity of that organization. If the server supports public access, the public logs on using the login name anonymous, with either no password required, or the user's E-Mail address required as the password. For that reason, public servers are usually called public FTP servers.

Anonymous FTP

Anonymous FTP lets you get information off a host without having to have an account on that host. In other words, Anonymous FTP allows you to retrieve files that are publicly available from another computer system on the Internet. Naturally the other computer has to be set up to allow for this.

Systems that allow anonymous FTP sessions are called anonymous FTP sites and the collection of files they make available are called FTP archives. As you might imagine, allowing people all over the Internet to access your computer is not something you do frivolously. There are a number of security considerations which must be addressed. For example, the system manager can control exactly which files on the computer are available for anonymous FTP access.

The rest of the files are off limits and cannot be accessed except by people with valid user accounts. As an extra security measure, you can copy all the files you want from the remote host to your computer, but you cannot copy files from your computer to the host.

Advantages of Anonymous FTP

1. It allows you to download virtually any type of information. Since, it is an ever-growing library that never closes, it covers every conceivable topic and, best of all being, it is free.
2. It is a principal way of distributing software on the Internet.
3. It is used to archive and disseminate the technical information that defines the Internet itself.
4. As a programmer, you should find the File Transfer Protocol interesting for several reasons.

First, its sheer popularity and widespread use on the internet should give you incentive to understand it. An accomplish Internet programmer should know how to use FTP and how to develop programs that perform similar operations.

Second, FTP is similar to SMTP and POP3_the e-mail protocols) in that it uses NVT ASCII [illegible] strings and reply codes. Like SMTP and POP3, the FTP uses a lock-step protocol approach (send command and wait for reply before sending the next command) for file-transfer operations.

Whatever FTP client you use, you begin by connecting to the server. Then transfer files and disconnect.

5.6 Working With FTP server

To connect to an FTP server, you give the host name of the server (for example, rtfm.mit.edu) to your FTP client software and then you log in. There are two ways, you can log in to a server using FTP:

- As a regular user of that system: If you have an account on the remote system (FTP server), you can transfer files to or from your own home directory, just log in using, your regular user ID and password. You can access all the files that your user name gives you permission to use.
- As an anonymous user: If you do not have an account of the FTP server, and you want to get files that are kept in the server's public FTP archive, you need to log in as an anonymous user. For anonymous FTP, you supply anonymous as your user name and your E-Mail address as your password.

Enter the URL of the FTP server as you would enter the URL of a Web server. Type the URL in the Location box near the top of the Netscape Navigator window, or the Address box of Internet Explorer, for instance.

For anonymous FTP	:	URL starts with ftp: //, as in ftp: //rtfm.mit.edu.
For private FTP	:	In URL, you have to give a user ID and password and has to be in the form: ftp: // user ID: password@servername, e.g., ftp: // reader: readit2u@ftp.whatever.edu.

Once you are connected, your browser displays directory names and file names as a list of links.

Downloading Files	:	To download a file, click it. The browser prompts you for a name and location on your computer. The browser will download and display the file, rather than prompt you for a filename and location.
Uploading Files	:	Netscape Navigator and certain other browsers can also upload files.

Once you have logged in to an FTP server, the server may display welcoming and instructional text about using the server. FTP servers transmit messages to let you (i.e. your client program) know what's going on. For example, when you have transferred a file, you might see the message 226 transfers complete, or you may see a dialog box saying the same thing in different words.

5.6.1 Transferring Files

FTP servers typically contain many different directories (folders). Once you are connected to an FTP server, you select a particular folder, called the current working directory, from which you will download, or to which you will upload files. If you have permission to do so, you may be able to create additional folders, rename folders, or delete them. You transfer a file — by either uploading or downloading —by using one of the following two modes:

- ASCII mode: Use ASCII mode when you want to transfer text files (including HTML files). Different computer system use different characters to indicate the end of lines. In ASCII mode, the FTP software automatically adjusts line endings for the system to which the file is transferred.
- Binary (or Image): Unformatted text files, can be transferred using Binary mode. In Binary mode, the FTP Software does not make any changes to the contents of the file during transfer. Use Binary mode when transferring graphical files, audio files, video files, programs, or any other kind of file other than plain text.

5.6.2 Disconnecting from the Server

When you have finished using an FTP site, you disconnect from that site (or your client program disconnects when you exit the program). Some FTP client programs allow you to connect to several FTP sites at once; disconnecting from one site does not affect your connection to other sites.

5.6.3 Common FTP Commands

FTP clients and servers use a variety of commands. The Table 5.1 given below summarizes commonly found commands on FTP clients and servers.

Table 5.1

Function	Command	Description
Start FTP	FTP hostname	Starts FTP and connects to the FTP server named hostname.
Server Connection	Open hostname	Establishes a connection to a specified computer hostname.
	Close	Closes the connection to remote host, say in ftp.
	User name	Sets user name.
	password bye	Disconnects from FTP server and exits the FTP client software.
Transfer mode (type)	ascii	(Default) sets file type to ASCII text file, i.e., transfer files in ASCII mode.
	binary	Transfers files in binary mode.
Directories	cd dir	Remote: changes to specified directory.
	cdup	Remote: changes to parent directory.
	dir directory	Remote: displays a long directory listing.
	lcd directory	Local: changes directory.
	ls directory	Remote: displays a short directory listing.
	pwd	Displays the current directory on the FTP server.
File transfer	get old new	Downloads the file old to your computer and name it new. Omit new to use the same name.
	mget my	Downloads the files to your computer that match the wildcard pattern my.

	put old new	Uploads the file old to the FTP server and name[illegible]
	mput my	Uploads the files to the FTP server that match the wild[illegible] pattern my.
Basic	?	Displays a list of all the FTP commands
	help	Displays a list of all commands that the FTP client [illegible]
	help command	Displays one-line summary of the specified com[illegible]
File management	Delete name	Deletes the file name on the FTP server
	mdelete my	Deletes the files that match the wildcard patter[illegible] server.
	rename old new	Renames the file named old on the FTP server, [illegible] new.

5.7 Simple Mail Transfer Protocol

Simple Mail Transfer Protocol (SMTP) is used to send mail across the Internet. There are [illegible] programs used in the process of sending and receiving mail. They are:

- MUA - Mail users agent.
  - This is the program a user will use to type e-mail. It usually incorporates [illegible] support. The user types the mail and it is passed to the sending MTA.
- MTA - Message Transfer Agent
  - It is used to pass mail from the sending machine to the receiving machine. [illegible] MTA program running on both the sending and receiving machine. Sendmail is [illegible]
- LDA - Local Delivery Agent
  - This agent on the receiving machine receives the mail from its MTA. This program usually procmail.
- Mail notifier
  - This program notifies the recipient that they have mail. Normally this requires programs, biff and comsat. Biff allows the administrator or user to turn [illegible] service.

The MTA on both machines use the network SMTP (simple mail transfer protocol) to pass mail between them. Other components of mail service include:

- Directory services - A list of users on a system. Microsoft provides a Global Address [illegible] Personal Address Book.
- Post Office - This is where the messages are stored.

5.7.1 Mail Protocols

- SMTP - Simple Mail Transport Protocol is used on the internet, it is not a transport layer protocol but is an application layer protocol.
- POP3 - Post Office Protocol version 3 is used by clients to access an internet mail server to get mail. It is not a transport layer protocol.
- IMAP4 - Internet Mail Access Protocol version 4 is the replacement for POP3.
- MIME - Multipurpose Internet Mail Extension is the protocol that defines the way files are attached to SMTP messages.
- X.400 - International Telecommunication Union standard defines transfer protocols for sending mail between mail servers.

- MHS - Message Handling Service by Novell is used for mail on Netware networks.

5.7.2 Directory Services

- Lightweight Directory Access Protocol (LDAP)
- X.500 - This is a recommendation outlining how an organization can share objects and names on a large network. It is hierarchical similar to DNS, defining domains consisting of organizations, divisions, departments, and workgroups. The domains provide information about the users and available resources on that domain. This X.500 system is like a directory. Its recommendation comes from the International Telegraph and Telephone Consultative Committee (CCITT)

5.7.3 Mail API

Mail application programming interfaces (APIs) allow e-mail support to be integrated into application programs.

- MAPI - Microsoft's Messaging API which is incorporated throughout Microsoft's office products supports mail at the application level
- VIM - Vendor-Independent Messaging protocol from Lotus is supported by many vendors exclusive of Microsoft.

Three parts of a mail message:

1. Envelope - Includes recipient and sender addresses using the MAIL and RCPT commands.
2. Headers - Each header has a name followed by a colon and its value. Some headers are From, Date, Reply To, Received, Message ID, To, and Subject.
3. Body - The contents of the message sent in 7 bit ASCII code.

5.7.4 SMTP Commands:

- HELO - Sent by client with domain name such as mymachine.mycompany.com.
- MAIL - From <myself@mymachine.mycompany.com>
- RCPT - To <myfriend@theirmachine.theirorg.org>
- DATA - Sends the contents of the message. The headers are sent, then a blank line, then the message body is sent. A line with "." and no other characters indicates the end of the message.
- QUIT

5.8 Simple Network Management Protocol

Simple Network Management Protocol (SNMP) is used as the transport protocol for network management. Network management consists of network management stations communicating with network elements such as hosts, routers, servers, or printers. The **agent** is the software on the network element (host, router, printer) that runs the network management software. Therefore when the word agent is used it is referring to the network element. The agent will store information in a **management information base (MIB)**.

Management software will poll the various network devices and get the information stored in them. RFC 1155, 1157, and 1213 define SNMP with RFC 1157 defining the protocol itself. The manager uses UDP port 61 to send requests to the agent and the agent uses UDP port 62 to send replies or messages to the manager. The manager can ask for data from the agent or set variable values in the agent. Agents can reply and report events.

There are three supporting pieces to TCP/IP network management:

1. Management Information BASE (MIB) specifies variables the network elements maintain.
2. A set of common structures and a way to reference the variables in the database.
3. The protocol used to communicate between the manager and the network element agent which is SNMP.

SNMP collects information two ways:

1. The devices on the network are polled by management stations.
2. Devices send alerts to SNMP management stations. The public community may be used for the alert list so all management stations will receive the alert.

SNMP must be installed on the devices to do this. SNMP terms:

- Baseline - A report outlining the state of the network.
- Trap - An alert that is sent to a management station by agents.
- Agent - A program at devices that can be set to watch for some event and send a trap message to a management station if the event occurs.

The network manager can set the threshold of the monitored event that will trigger the sending of a trap message. SNMP enables counters for monitoring the performance of the network and in conjunction with Performance Monitor.

5.8.1 SNMP Communities

An SNMP community is the group that devices and management stations running SNMP belong to. It helps define where information is sent. The community name is used to identify the group. A SNMP device or agent may belong to more than one SNMP community. It will not respond to requests from management stations that do not belong to one of its communities. SNMP default communities are:

- Write = private
- Read = public

5.8.2 SNMP Security

SNMP should be protected from the internet with a firewall. Beyond the SNMP community structure, there is one trap that adds some security to SNMP.

- Send Authentication Trap - When a device receives an authentication that fails, a trap is sent to a management station.

Other configuration parameters that affect security are:

- Accepted Community Names - Only requests from computers in the list of community names will be accepted.
- Accept SNMP Packets from Any Host - This is checked by default. Setting specific hosts will increase security.
- Only Accept SNMP Packets from These Hosts - Only requests from hosts on the list of IP addresses are accepted. Use IP, or IPX address or host name to identify the host.

5.8.3 SNMP Message Types

There are five types of messages exchanged in SNMP. They are referred to by Protocol Data Unit (PDU) type.

PDU Type	Name	Description
0	get-request	Get one or more variables .(manager to element)
1	get-next-request	Get next variable after one or more specified variables. (manager to element)
2	set-request	Set one or more variables. (manager to element)
3	get-response	Return value of one or More variables. (element to manager)
4	Trap	Notify manager of an event. (element to manager)

The SNMP message with PDU type 0-3 consists of:

1. Version of SNMP
2. Community - A clear text password character string
3. PDU type
4. Request ID - Used to associate the request with the response. For PDU 0-2, it is set by the manager.
5. error status - An integer sent by the agent to identify an error condition

Error	Name	Description
0	no error	OK
1	too big	Reply does not fit into one message
2	no such name	The variable specified does not exist
3	bad value	Invalid value specified in a set request.
4	read only	The variable to be changed is read only.
5	general error	General error

6. error index - Specifies which variable was in error when an error occurred. It is an integer that
7. name - The name of the variable (being set or read).
8. value - The value of the variable (being set or read)
9. any other names and values to get/set

The SNMP message with PDU type 4 (trap) consists of:

1. PDU type
2. Enterprise - The agents OBJECT IDENTIFIER or system objects ID. Falls under the node in the MIB tree.
3. agent addr - The IP address of the agent.
4. Trap type - Identifies the type of event being reported.

Trap Type	Name	Description
0	cold start	Agent is booting
1	warm start	Agent is rebooting
2	link down	An interface has gone down
3	link up	An interface has come up
4	authentification failure	An invalid community (password) was received
5	egp neighbor loss	An EGP peer has gone down.
6	enterprise specific	Look in the enterprise code for information on the trap

5. Specific code - Must be 0.
6. Time stamp - The time in 1/100ths of seconds since the agent initialized.
7. name
8. Value
9. Any other names and values

Types of data used:

- INTEGER - Some have minimum and maximum values.
- OCTET STRING - The number of bytes in the string is before the string.
- DISPLAY STRING - Each byte must be an ASCII value
- OBJECT IDENTIFIER - Specifies a data type allocated by an organization with responsibility for a group of identifiers. A sequence of integers separated by decimals which follows a structure.
- NULL - Used as the value of all variables in a get request.
- IpAddress - A 4 byte long OCTET STRING. One byte for each byte of the IP address.
- PhysAddress - A 6 byte octet string specifying an ethernet or hardware address.
- Counter - A 32 bit unsigned integer
- GaugeAn unsigned 32 bit integer with a value that can increase or decrease but will not fall below a minimum or exceed a maximum.

- TimeTicks - Time counter. Counts in 1/100 of seconds.
- SEQUENCE - Similar to a programming structure with entries of type IPAddress called udpLocalAddress and type INTEGER called udpLocalPort.
- SEQUENCE OF - An array with elements with one type.

5.8.4 The MIB data structure RFC 1213

In the above list the data type "OBJECT IDENTIFIER" is listed as a part of the management information database. These object identifiers are referenced very similar to a DNS tree with a directory at the top called root. Each node in the tree is given a text name and is also referenced numerically similar to IP addresses. There are multiple levels in the tree with the bottom level being variables, and the next one up is called group. The packets sent in SNMP use numeric identifiers rather than text. All identifiers begin with iso(1).Org(3). dod(6).internet(1).mgmt(2).mib(1). Numerically, that is 1.3.6.1.2.1. In text it is "iso.org.dod.internet.mgmt.mib". Under mib are the following groups. The information in these groups is not complete and you should refer to the RFC for full information.

1. system
    1. sysDesc (DisplayString) - Description of entity
    2. sysObjectID (ObjectID) - Vendors ID in the subtree (1.3.6.1.4.1.
    3. sysUPTime (Timer) - Time the system has been up
    4. sysContact (DisplayString) - Name of contact person
    5. sysName (DisplayString) - Domain name of the element such as mymachine.mycompany.com
    6. sysLocation (DisplayString) - Physical location of the element.
    7. sysServices 0x1-physical, 0x02-datalink, 0x04-internet, 0x08 end to end, 0x40-application. If the bit is set the service is provided
2. interfaces
    1. ifNumber (INTEGER) - Number of network interfaces
    2. ifTable (table)
        1. ifIndex
        2. ifDescr - Description of interface
        3. ifType - 6=ethernet, 7=802.3 ethernet, 9=802.5 token ring, 23 = PPP, 28=SLIP
        4. ifMtu
        5. ifSpeed - Bits/second
        6. ifPhysAddress
        7. ifAdminStatus - Desired state of interface 1=up, 2=down, 3=testing
        8. ifOperStatus - Current state of interface 1=up, 2=down, 3=testing
        9. ifLastchange
        10. ifInOctets - Total bytes received
        11. ifInUcastPkts
        12. ifInNUcastPkts
        13. ifInDiscards
        14. ifInErrors
        15. ifInUnknownProtos
        16. ifOutOctets

17. ifOutUcastPkts
18. ifOutNUcastPkts
19. ifOutDiscards
20. ifOutErrors
21. ifOutQLen
22. ifSpecific

3. at - Address translation group
    1. atIfIndex (INTEGER) - Interface number
    2. atPhysAddress (PhyAddress)
    3. atNetAddress (NetworkAddress) - IP address
4. ip
    1. ipForwarding
    2. ipDefaultTTL (INTEGER)
    3. ipInReceives (counter)
    4. ipInHdrErrors (counter)
    5. ipInAddrErrors (counter)
    6. ipForwDatagrams (counter)
    7. ipInUnknownProtos (counter)
    8. ipInDiscards (counter)
    9. ipInDelivers (counter)
    10. ipOutRequests (counter)
    11. ipOutDiscards (counter)
    12. ipOutNoRoutes (INTEGER)
    13. ipReasmTimeout (counter)
    14. ipReasmReqds (counter) - Number of IP fragments received that need to be reassembled
    15. ipReasmOKs (counter)
    16. ipReasmFails (counter)
    17. ipFragOKs (counter)
    18. ipFragFails (counter)
    19. ipFragCreates (counter)
    20. ipRoutingDiscards (counter)
    21. ipAddrTable (table)
        1. ipAddrEntry (index)
            1. ipAdEntAddr
            2. ipAdEntIfIndex
            3. ipAdEntNetMask
            4. ipAdEntBcastAddr
            5. ipAdEntReasmMaxSize
5. icmp
6. tcp
7. udp
    1. udpInDatagrams (counter) - UDP datagrams delivered to user processes.
    2. udpNoPorts (counter) - UDP datagrams which were not received at the port since there was no application to receive it.
    3. udpInErrors (counter) - Number of UDP datagrams not delivered for reasons other than no applications available to receive them.

4. udpOutDatagrams (counter) - Number of UDP datagrams sent.
5. udpTable (table)
    1. udpEntry - Specifies the table entry number
        1. udpLocalAddress
        2. udpLocalPort

The ordering of data in the MIB is numeric. When the getnext function is used it gets the next data based on the numeric ordering.

5.9 Domain Name Service

Domain Name Service (DNS) is the service used to convert human readable names of hosts to IP addresses. Host names are not case sensitive and can contain alphabetic or numeric letters or the hyphen. Avoid the underscore.

5.9.1 Host Name

A fully qualified domain name (FQDN) consists of the host name plus domain name as in the following example:

computername.domain.com

The part of the system sending the queries is called the resolver and is the client side of the configuration. The nameserver answers the queries. Read RFCs 1034 and 1035. These contain the bulk of the DNS information and are superceded by RFCs 1535-1537. Naming is in RFC 1591. The main function of DNS is the mapping of IP addresses to human readable names. Three main components of DNS

1. resolver
2. name server
3. database of resource records(RRs)

5.9.2 Domain Name System

The Domain Name System (DNS) is basically a large database which resides on various computers and it contains the names and IP addresses of various hosts on the internet and various domains. The Domain Name System is used to provide information to the Domain Name Service to use when queries are made. The service is the act of querying the database, and the system is the data structure and data itself.

The Domain Name System is similar to a file system in Unix or DOS starting with a root. Branches attach to the root to create a huge set of paths. Each branch in the DNS is called a label. Each label can be 63 characters long, but most are less. Each text word between the dots can be 63 characters in length, with the total domain name (all the labels) limited to 255 bytes in overall length.
The domain name system database is divided into sections called **zones**. The name servers in their respective zones are responsible for answering queries for their zones. A zone is a subtree of DNS and is administered separately. There are multiple name servers for a zone. There is usually one primary nameserver and one or more secondary name servers. A name server may be authoritative for more than one zone.

DNS names are assigned through the Internet Registries by the Internet Assigned Number Authority (IANA). The domain name is a name assigned to an internet domain. For example,

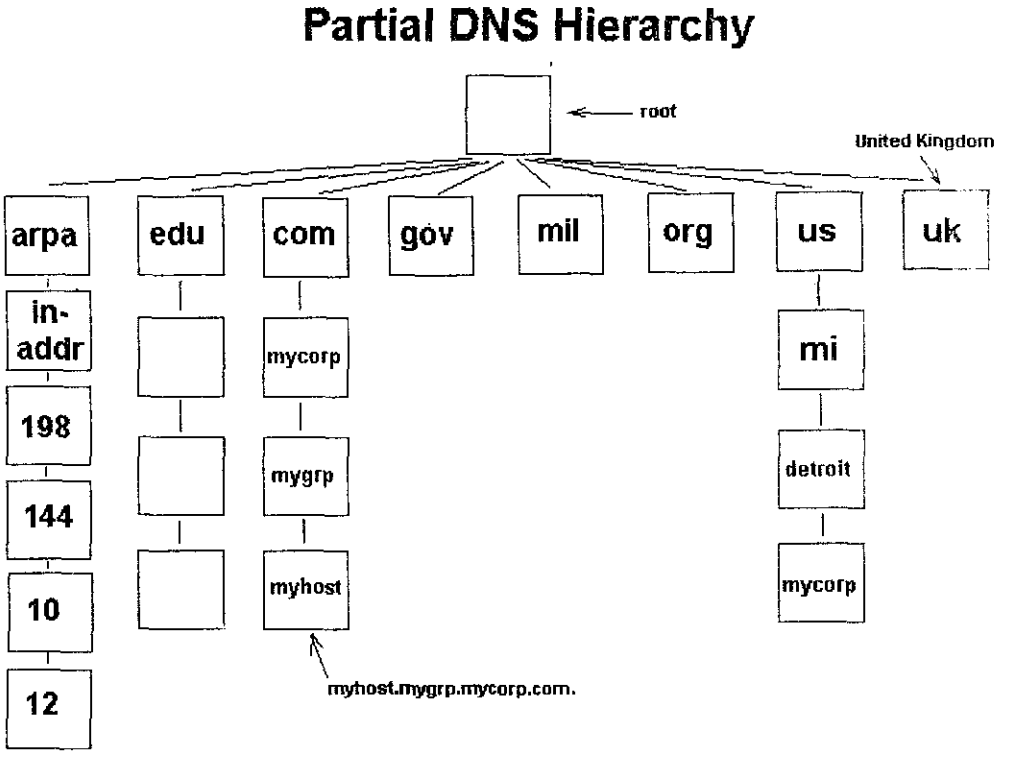
- mycollege.edu represents the domain name of an educational institution.
- microsoft.com and 3Com.com represent the domain names at those commercial companies.

Naming hosts within the domain is up to individuals administer their domain. Access to the Domain name database is through a resolver which may be a program or part of an operating system that resides on users workstations. In Unix the resolver is accessed by using the library functions "gethostbyname" and "gethostbyaddr". The resolver will send requests to the name servers to return information requested by the user. The requesting computer tries to connect to the name server using its IP address rather than the name.

5.9.3 DNS Structure

The drawing below shows a partial DNS hierarchy. At the top is what is called the root and it is the start of all other branches in the DNS tree. It is designated with a period. Each branch moves down from level to level. When referring to DNS addresses, they are referred to from the bottom up with the root designator (period) at the far right.

Example: "myhost.mycompany.com.".



DNS is hierarchical in structure. A domain is a subtree of the domain name space. From the root, the assigned top-level domains in the U.S. are:

- GOV - Government body.
- EDU - Educational body.
- INT - International organization
- NET - Networks
- COM - Commercial entity.
- MIL - U. S. Military.
- ORG - Any other organization not previously listed.

Outside this list are top level domains for various countries.

Each node on the domain name system is separated by a ".".

Example: "mymachine.mycompany.com.". Note that any name ending in a "." is an absolute domain name since it goes back to root.

5.9.4 DNS Message Format

Bits	Name	Description
0-15	Identification	Used to match responses to requests. Set by client and returned by server.
16-31	Flags	Tells if query or response, type of query, if authoritative answer, if truncated, if recursion desired, and if recursion is available.
32-47	Number of questions	
48-63	Number of answer RRs	
64-79	Number of authority RRs	
80-95	Number of additional RRs	
96-??	Questions - variable lengths	There can be variable numbers of questions sent.
??-??	Answers - variable lengths	Answers are variable numbers of resource records.
??-??	Authority - variable lengths	
??-??	Additional Information variable lengths	

Question format includes query name, query type and query class. The query name is the name being looked up. The query class is normally 1 for internet address. The query types are listed in the table below. They include NS, CNAME, A, etc.

The answers, authority and additional information are in resource record (RR) format, which contains the following.

1. Domain name
2. Type - One of the RR codes listed below.
3. Class - Normally indicates internet data which is a 1.
4. Time to live field - The number of seconds the RR is saved by the client.
5. Resource data length specifies the amount of data. The data is dependent on its type such as CNAME, A, NS or others as shown in the table below. If the type is "A" the data is a 4 byte IP address.

The table below shows resource record types:

Type	RR value	Description
A	1	Host's IP address
NS	2	Host's or domain's name server(s)
CNAME	5	Host's canonical name, host identified by an alias domain name
PTR	12	Host's domain name, host identified by its IP address
HINFO	13	Host information
MX	15	Host's or domain's mail exchanger
AXFR	252	Request for zone transfer
ANY	255	Request for all records

Usage and file formats

If a domain name is not found when a query is made, the server may search for the name elsewhere and return the information to the requesting workstation, or return the address of a name server that the workstation can query to get more information. There are special servers on the Internet that provide guidance to all name servers. These are known as root name servers. They do not contain all information about every host on the Internet, but they do provide direction as to where domains are located (the IP address of the name server for the uppermost domain a server is requesting). The root name server is the starting point to find any domain on the Internet.

Name Server Types

There are three types of name servers:

1. The primary master builds its database from files that were preconfigured on its host. Called zone or database files. The name server reads these files and builds a database for the zone it authoritative for.
2. Secondary masters can provide information to resolvers just like the primary masters, but they get their information from the primary. Any updates to the database are provided by the primary
3. Caching name server - It gets all its answers to queries from other name servers and saves (caches) the answers. It is a non-authoritative server.

The caching only name server generates no zone transfer traffic. A DNS Server that can communicate outside of the private network to resolve a DNS name query is referred to as **forwarder**

DNS Query Types

There are two types of queries issued:

1. **Recursive** queries received by a server forces that server to find the information requested post a message back to the querier that the information cannot be found.

2. **Iterative** queries allow the server to search for the information and pass back the best information it knows about. This is the type that is used between servers. Clients used the recursive query.
3. **Reverse** - The client provides the IP address and asks for the name. In other queries the name is provided, and the IP address is returned to the client. Reverse lookup entries for a network 192.168.100.0 is "100.168.192.in-addr arpa".

Generally (but not always), a server-to-server query is iterative and a client-resolver-to-server query is recursive. You should also note that a server can be queried or it can be the person placing a query. Therefore, a server contains both the server and client functions. A server can transmit either type of query. If it is handed a recursive query from a remote source, it must transmit other queries to find the specified name, or send a message back to the originator of the query that the name could not be found.

DNS Transport protocol

DNS resolvers first attempt to use UDP for transport, then use TCP if UDP fails.

The DNS Database

A database is made up of records and the DNS is a database. Therefore, common resource record types in the DNS database are:

- A - Host's IP address. Address record allowing a computer name to be translated into an IP address. Each computer must have this record for its IP address to be located. These names are not assigned for clients that have dynamically assigned IP addresses, but are a must for locating servers with static IP addresses.
- PTR - Host's domain name, host identified by its IP address
- CNAME - Host's canonical name allows additional names or aliases to be used to locate a computer.
- MX - Host's or domain's mail exchanger.
- NS - Host's or domain's name server(s).
- SOA - Indicates authority for the domain
- TXT - Generic text record
- SRV - Service location record
- RP - Responsible person
- HINFO - Host information record with CPU type and operating system.

When a resolver requests information from the server, the DNS query message indicates one of the preceding types.

DNS Files

- CACHE.DNS - The DNS Cache file. **This file is used to resolve internet DNS queries**. On Windows systems, it is located in the WINNTROOT\system32\DNS directory and is used to configure a DNS server to use a DNS server on the internet to resolve names not in the local domain.

5.10 NFS and RPC Protocols

NFS, defined by RFC 1094, is a method for client systems to use a filesystem on a remote host computer. NFS uses the UDP protocol and is supported by RPC.

RPC, defined by RFC 1057, is a set of function calls used by a client program to call functions in a remote server program. The port mapper program is the program used to keep track of which ports programs supporting RPC functions use. The port mappers port is 111. In Redhat Linux the portmapper daemon is started in the /etc/rc.d/init.d/portmap and the daemon program is called "portmap '.

The rpcinfo command

The command "rpcinfo -p" will show the port numbers that are assigned to the RPC services

```
program vers proto   port
 100000    2   tcp    111  portmapper
 100000    2   udp    111  portmapper
 100011    1   udp    747  rquotad
 100011    2   udp    747  rquotad
 100005    1   udp    757  mountd
 100005    1   tcp    759  mountd
 100005    2   udp    762  mountd
 100005    2   tcp    764  mountd
 100003    2   udp   2049  nfs
```

Services that may be listed include:

- rquotad - Enforces the set quotas for remote mounted NFS systems.
- mountd - Performs the requested mounts.
- nfs - Handles the user interface to the kernel module that performs NFS.

NFS related services in Linux include:

- amd - Runs the automount daemon for automatic remote filesystem mounting such as nfs. It especially worthwhile for working with removeable media such as floppies or CD ROM disks.
- autofs - This is the startup, stop, and status script for the automount program used to configure mount points for automatic mounting of file systems.
- nfs - Provides Network File System server services.
- netfs - Mounts and unmounts Network Fils System (NFS), Windows (SMB), and Netware (NCP) file systems. The mount command is used to perform this operation and no daemon is run in background.

The /etc/exports file is used to configure exported filesystems.

5.11 XDR Protocol

XDR protocol is a standard for describing and encoding data. The XDR standard assumes that bytes (or octets) are portable, where a byte is defined as 8 bits of data. It also assumes that hardware that encodes bytes onto various media will preserve the bytes' meanings across hardware boundaries. (For example, the Ethernet standard suggests that bytes be encoded in "little-endian" style, or least significant bit first.)

Once XDR data is shared among machines, it shouldn't matter that the data produced on an IRIS is consumed by a VAX (or vice versa). Similarly, the choice of operating systems should have no influence on how the data is represented externally. For programming languages, data produced by a C program should be readable by a FORTRAN or Pascal program.

5.12 X Windows Protocol

X Windows is the predominate windowing system on UNIX computers, developed by the X Consortium, lead by M.I.T.

An X *server* manages the display on the workstation. *Clients* can connect to server via TCP/IP and perform graphics operations. This makes X Windows much more network capable than Microsoft Windows, for example, which can only be accessed via a local API.

X Windows operates over TCP, typically using server port numbers starting with 6000. The X server for a system's first display listens on port 6000; if the system has a second display, its server listens on port 6001; a third display would listen on 6002; etc. The protocol used over this reliable stream connection is essentially request/reply, and it's reputation is as a fat protocol that consumes a lot of bandwidth. Lightweight X (LWX), introduced in X11R6, attempts to reduces X's bandwidth needs to the point where it can be run over dialup modem connections.

The X Protocol, documented in a postscript file, defines dozens of messages that can be exchanged between a client and a server. They can generally be classified into four categories:

- Requests,
- Replies,
- Events, and
- Errors.

Typical requests include Draw PolyLine, Draw Text, Create Window, Fill. Replies are matched to particular Requests. Events are asynchronous occurances such as keystrokes and mouse clicks. Errors are matched to particular Requests.

If a window is partially or fully obscured by another, overlapping window, the server has two options available to it. The server can allocate additional memory, called *backing store*, to record to contents of the obscured window. This is purely optional, however. The server can simply ignore the obscured part of the window. Later, when that part of the window becomes visible again, the server sends an Expose event to the client, which must then redraw the affected area. The client, therefore, must be prepared to redraw any part of its windows at any time.

Applications do not need to access the X Windows protocol directly. X Windows supports several APIs. The most basic of these is Xlib, which interfaces fairly directly to the underlying network protocol. Most X client applications are linked against Xlib, which allows them to operate on either a local or remote X server, simply by adjusting either an environment variable or a command-line argument.

Widgets layer on top of Xlib and provide X Windows with an object-oriented programming model. A widget is an X window
capable of handling most of its own protocol interaction. The most popular widget sets are Athena Widgets (aw) and Motif.

Student Activity

1. What is the purpose of application layer?
2. What are various application layer protocols?
3. What is Telnet?
4. What is file transfer protocol? How does it work?
5. What SMTP protocol meant for?
6. What is the significance of DNS?
7. What is X Windows protocol meant for?
8. What SNMP? Discuss its relevance.
9. What are NFS and RPC protocols meant for?
10. What are XDR protocols?

Unit 6

World Wide Web and Internet Tools

Learning Objectives

After reading this unit you should appreciate the following:

6.1 World Wide Web (WWW)
6.2 Web Browsers
6.3 Web Pages in Other Languages
6.4 Browsing the Web
6.5 Downloading Information Using Internet
6.6 Web Search Engines
6.7 Search Engine (ALTA VISTA)
6.8 Gopher
6.9 Veronica
6.10 MOSAIC
6.11 WAIS
6.12 Internet Relay Chat (IRC)
6.13 Web-Chat
6.14 E-Mails
6.15 E-mail Packages
6.16 Pine
6.17 Eudora
6.18 Outlook
6.19 Mailing Lists
6.20 Usenet Newsgroup

6.1 World Wide Web (WWW)

The World Wide Web (also called WWW, or W3, or simply the Web) is a tool that helps you to find and retrieve information, using links to the other WWW pages. Web links are stored within the page itself and when you wish to "jump" to the page that is linked, you select the "hotspot" or "anchor". This technique is some times called hypermedia or hypertext. If you have used a Windows or Macintosh based help system in which you click on an underlined phrase to jump to that topic, you are already

familiar with hypermedia. On the Web, a link can point anywhere else on the Internet. Thus, it is possible to follow one link after another, jumping from one computer to another, all around the Net.

The WWW provides a network of interactive documents and the software to access them. To navigate the WWW, you surf from one page to another by pointing and clicking on the hyperlinks in text or graphics. The World Wide Web and HTTP:

- Allow you to create links from one piece of information to another.
- Can incorporate references to text, graphics, sound, and movies (or multimedias).
- Understand other Internet protocols, such as FTP, gopher, and telnet.
- The World Wide Web is non-linear with no top, or no bottom. The meaning of non-linear is that you do not have to follow a hierarchical path to information resources. So you can:
- Jump from one link (resource) to another.

Go directly to a resource if you know the Uniform Resource Locator (URL), i.e., its address.

Even jump to specific parts of a document.

Since the Web is not hierarchical and can handle graphics, it offers a great deal of flexibility in the way information resources can be organized, presented, and described.

6.2 Web Browsers

Like much of the Internet, the World Wide Web operates on a client/server model. You run a Web client on your computer — called a Web browser — such as Netscape Communicator or Microsoft's Internet Explorer. That client contacts a Web server and requests information or resources. The Web server locates and then sends the information to the Web browser, which displays the results. Web browsers are the Client software (your machine is a client to ISP's server) which have various graphics capabilities to access the information from the Internet. Modern Web browsers are capable of browsing WWW Gopher sites FTP sites and also provides facilities for e-mail, etc. Initially NCSA's web browser Mosaic hit the market, which actually made the browsing popular. Now Web browsers from Netscape and from Microsoft are the user's choice. You can get hold of any such browser and start browsing the Net

When Web browsers contact servers, they are asking to view pages built with Hypertext Markup Language (HTML). They interpret those pages and display them on your computer. They display application programs, animations, and similar material created with programming languages such as Java and ActiveX, and scripting languages such as JavaScript. Sometimes, home pages contain links to files that the Web browser cannot play or display, such as sound or animation files. In that case, you will need a plug-in or a helper application. You configure your Web browser to use the helper application or plug-in whenever it encounters a sound or animation file that the browser cannot run or play

Over the years, Web browsers have become increasingly sophisticated. They have become full-blown software suites that can do everything from video-conferencing to letting you create and publish HTML pages. They have also begun to blur the line between your local computer and the Internet-in essence they can make your computer and the Internet function as a single system. To facilitate the usage of computers over Internet, Microsoft has integrated web browsing and the Internet directly into the

operating system. For example, with Internet Explorer 4.0 and above, and with Windows 98, the Windows desktop can be HTML based. This means Web links can be directly embedded into the desktop.

- It allows you to enter the address of the site you want to jump to (called a URL-or uniform resource locator), or to jump there by clicking hotspots, high lighted words, buttons, pictures, or icons called hyperlinks on your screen.
- It formats Web documents for display on your screen.
- It allows you to back up and go forward through pages you have already visited.
- It allows you to copy text from the screen and paste it into a word processing program.
- It allows you to print the document you see on the screen.
- It makes it possible to transfer files-text, graphics, movies, animations, sounds, and programs – from other computers to your computer (called downloading).
- It allows you to send and receive e-mail and other Internet services such as ftp (file transfer protocol), gopher, and Usenet news groups.

Now Web browsers from Netscape and from Microsoft are the user's choice. You can get hold of any such browser and start browsing the Net.

Like much of the internet, the World Wide Web operates on a client/server model. You run a Web client on your computer — called a Web browser — such as Netscape Communicator or Microsoft's Internet Explorer. That client contacts a Web server and requests information or resources. The Web server locates and then sends the information to the Web browser, which displays the results. When Web browsers contact servers, they are asking to view pages built with Hypertext Markup Language (HTML). They interpret those pages and display them on your computer. They display application programs, animations, and similar material created with programming languages such as Java and ActiveX, and scripting languages such as JavaScript. Sometimes, home pages contain links to files that the Web browser cannot play or display, such as sound or animation files. In that case, you will need a plug-in or a helper application. You configure your Web browser to use the helper application or plug-in whenever it encounters a sound or animation file that the browser cannot run or play.

Over the years, Web browsers have become increasingly sophisticated. They have become full-blown software suites that can do everything from video-conferencing to letting you create and publish HTML pages. They have also begun to blur the line between your local computer and the Internet-in essence; they can make your computer and the Internet function as a single system. To facilitate the usage of computers over Internet, Microsoft has integrated web browsing and the Internet directly into the operating system. For example, with Internet Explorer 4.0 and above, and with Windows 98, the Windows desktop can be HTML based. This means Web links can be directly embedded into the desktop.

So, you can have links to your favorite Web pages right on the desktop. And even applications such as word processors now have Web capabilities bust into them — such as being able to browse the Web, or build home pages. Even more significantly, using technology that Microsoft calls Active Desktop, internet based desktop components can live on the desktop. These components can be things such as stock tickers, which deliver live Web content directly to the desktop. You don't need to fire up your

Web browser to get the information; it's delivered straight to your Windows desktop without you having to do anything. Both Microsoft and Netscape have also built entire suites of software around their browsers. Netscape for example, calls its suite Netscape Communicator. Communicator includes modules for reading newsgroups; for reading, sending and managing internet mail; for audio conferencing; for collaborative work on whiteboard applications in which people can view and mark up the same documents simultaneously; and more. These enhancements will help users in all areas of collaborative computing and communication.

It allows you to enter the address of the site you want to jump to (called a URL-or uniform resource locator), or to jump there by clicking hotspots, highlighted words, buttons, pictures, or icons called hyperlinks on your screen.

- It formats Web documents for display on your screen.
- It allows you to back up and go forward through pages you have already visited.
- It allows you to copy text from the screen and paste it into a word processing program.
- It allows you to print the document you see on the screen.
- It makes it possible to transfer files-text, graphics, movies, animations, sounds, and programs from other computers to your computer (called downloading).
- It allows you to send and receive e-mail and other Internet services such as ftp (file transfer protocol), gopher, and Usenet news groups.

6.2.1 Working of a Web Browser

Web browsers consist of software that runs on your computer and displays home pages on the Web. There are clients for PC, Macintosh and UNIX computers. A Web browser displays information on your computer, by interpreting the Hypertext Markup language (HTML) that is used to build home pages on the Web. Home pages usually display graphics, sound and multimedia files as well as links to other pages, files that can be downloaded and other Internet resources.

The coding in the HTML files tells your browser how to display the text, graphics, links and multimedia files on the home page. The HTML file that your browser loads to display the home page does not actually have the graphics, sound, multimedia files, and other resources on it. Instead, it contains HTML references to those graphics and files. Your browser uses those references to find the files on the server and then display them on the home page.

The Web browser also interprets HTML tags as links to other Web sites or to other Web resources – these include graphics, multimedia files, newsgroups, or files which can be downloaded. Depending on the kind of link, it will perform different actions. For example, if the HTML code specifies the link as another home page the browser will retrieve the Uniform Resource Locator (URL) specified in the HTML file when the user clicks on the underlined link on the page. If the HTML code specifies a file to be downloaded, the browser will download the file to your computer.

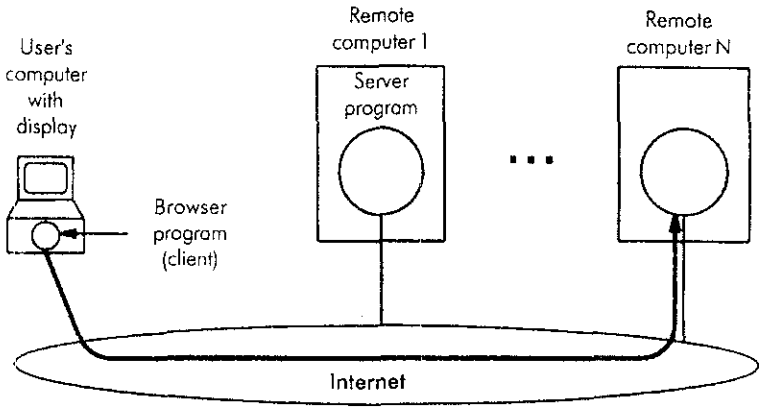


Figure 6.1: WWW Browser

6.2.2 Basic Features of Browsers

Before we get involved in all the details, let us discuss some important browser features.

1. The Web browser should be able to look at the Web pages throughout the Internet or to connect to various sites to access information, explore resources, and have fun.
2. The Web browser must enable you to follow the hyperlinks on a Web page and also to type in a URL for it to follow.
3. Another feature of browser is to have a number of other commands readily available through menus, icons, and buttons.
4. Your browser ought to include an easy way to get on-line help as well as built-in links to other resources on the Web that can give you help or answers to your questions.
5. You will definitely want a way to save links to the sites you have visited on the WWW so that you can get back to them during other sessions. Web browsers take care of those in two ways, through a history list, which keeps a record of some of the Web pages you've come across in the current session, and a bookmark list, which you use to keep a list of WWW pages you want to access any time you use your browser. The name of the site and its URL are kept in these lists. The bookmark list is particularly important and the browser will contain tools to manage and arrange it.
6. One of the main features of a browser is to search the information on the current page as well as search the WWW itself.
7. Browsers give you the facility to save a Web page in a file on your computer, print a Web page on your computer, and send the contents of a Web page by e-mail to others on the Internet.
8. Few Web browsers (like Netscape Communicator) are complete Internet package, means they come with components like e-mail client, newsgroup client, an HTML composer, telnet client, ftp client, etc.
9. Web browser should be able to handle text, images of the World Wide Web, as well as the hyperlinks to digital video, or other types of information.
10. To take advantage of some of the most exciting things on the World Wide Web, your browser needs to properly display and handle Web pages that contain animated or interactive items. Netscape Navigator can incorporate these features through its ability to interpret programs written in Java and Java Script.

11. Web browsers interact not just with the Web, but also with your computer's operating system and with other programs, called plug-ins, that gives the browser enhanced features.
12. Another important feature to insist on in your browser is caching. A browser that caches keeps copies of the pages you visit so that it does not have to download them again if you want to return to them. Reloading a page from the cache is much quicker than downloading it again from the original source.
13. The most important feature of any browser is ease of use. While all Web browsers are fundamentally simple to use, the one you settle on should be very easy to work with. It should function as a transparent window onto the Web.

If you will be browsing the Web from within a secured network, you may have to configure your browser to work through a special computer on your network called a proxy server. Most popular browsers let you configure them to work with a proxy server, but some don't, so find out if you will be working through a proxy before deciding on your browser. If you are, your ISP or system administrator will tell you if you need to do anything special to use your browser.

6.3 Web Pages in Other Languages

Does your Web browsing take you around the world to Web pages written in languages other than the one you use on your computer?

You can add the character sets for those languages, so that the pages are displayed correctly.

6.3.1 Elements of the Internet Explorer Window

The Internet Explorer Window shown in Figure 6.2 describes all the elements.

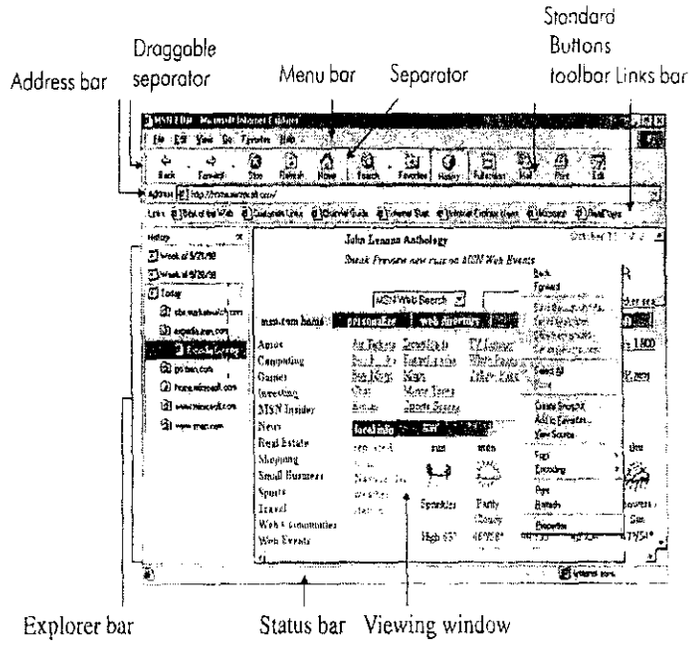


Figure 6.2: Elements of Internet Explorer Window

A) The Main Browser Window

When Internet Explorer is first opened up on your computer, the main screen of the program will appear. This main window has many parts to it, these parts are described in detail below.

- The "Title Bar" at the very top of the window tells you what the title of the page you are viewing is. The "Title Bar" will also tell you what Internet Explorer application is currently active.
- Directly under the "Title Bar" is the "Main Menu Bar". This bar has many different sub-menus which control all options, functions, and commands for the entire Internet Explorer program. Some of the browsing controls can also be found in these sub-menus.
- Beneath this menu is the "Internet Explorer Toolbar". This toolbar contains all of the most frequently used commands and all of the browsing functions.
- Under the toolbar is the "Address Bar". This will tell you the exact HTTP/URL location of the page you are currently viewing. You can also type a Web address directly into this bar and then press enter to go to that site.
- Below the "Address Bar" is the "Link Bar". These buttons will take you to pages at Microsoft's Main home site where they have applications and information specifically designed for your easy use.
- Underneath the "Link Bar" is the "Main Browser Window". This window will display all of the information that is located at the Web site you are currently located at. Any text, images, movies, animation, links, or any other application files will be shown in this window. The scroll bars located on the right side and on the bottom of this window allow you to continue viewing the page you are located at even when the page is too large to fit in your screen.
- The very bottom of the page is the "Status Bar". This bar tells you what the progress of the browser is while it downloads files to the page, where links go to when you move over them, whether or not a document is secure, and any other information that the program feels is necessary for you to know.

B) The Main Explorer Toolbar

Figure 6.3: Explorer Toolbar

The main toolbar is composed of eleven different buttons. Each of these buttons has a different function and purpose in Internet Explorer. The individual buttons will each be discussed in the following sections.

1. The Back Button: This button will take you back to whatever document you were previously viewing. Pressing it immediately takes you back one document. If you have browsed many pages, or are well into a multi-page document, pressing it repeatedly will continue to back you

up one page at a time. Once you reach your starting location, it will be greyed-out and unavailable.

2. The Forward Button: This button will take you forward to the next document if you have previously browsed multiple documents and had then backed-up to the page you are currently viewing. (If you have not backed up at all, the forward button will be greyed-out) Pressing it repeatedly will continue to move you forward one page at a time. You can move forward until you reach the last page that you had browsed, at which time the forward button will be greyed-out.

3. The Stop Button: The stop button stops ANY current operations by Internet Explorer. It will stop any type of file from loading. It can also be used to stop animations from continuing once a page is loaded. If you press it before a page has finished loading, the page will display everything it had finished loading before the stop button was pressed. If a document is completely loaded and there are no animations, movies, or other files still running, the stop button will have no immediate function.

4. The Refresh Button: This button will reload the current document that you are viewing. It is useful if the page updates very frequently so that you can view these changes as soon as they are available. If you are loading a document and the transfer was interrupted, you can reload the full document again by clicking here.

5. The Home Button: This button will return you to the page you have selected as the default start-up page for Internet Explorer. It will not take you back to the beginning of your web browsing, it will just return you to your home location from where you are. If you press back after reaching your home page, you will go back to the page you left after you hit the Home button.

6. The Search Button: This button will take you to the page you have selected as the default web search page for Internet Explorer. If you have not selected a page it will take you to Microsoft's default search page.

7. The Favorites Button: This button will open up the Favorites menu. You can choose a favorite that you wish to go to from the list, add a favorite to the list, or organize your favorites from this menu.

8. The Print Button: The print button will bring up a Print dialog box. In the box you can decide if you would like to print the contents of the page you are viewing, how many pages you will print, and also how many copies you will print. Keep in mind that if you try to print a page that is graphics intensive, you will need a printer that is capable of printing graphics well, the more graphics and pages a Web site has, the longer it will take to print.

9. The Font Button: Pressing this button causes Internet Explorer to cycle through the available font sizes. This button is useful if the text is too small to read, or too large to fit comfortably in the window.

10. The Mail Button: This button will open into a drop down menu from which you can select to read or send E-Mail. You can also open up your newsgroups from this menu.

11. The Edit Button: This button will ONLY be on your toolbar if you have a Windows system Web editor (such as Microsoft Frontpage or Microsoft Word) installed on your computer. If

you press this button, it will launch that editor and open the document you are currently viewing in it.

C) Beginning Basic Browsing

Your first time that you browse the web, you may have some difficulty. Efficiently browsing the Web is just like any other complex task in life, it takes practice to be good at it. Internet Explorer has some built-in features which will help to make it easier for you to browse the web. The fastest way to get to a place that you can search from, is to click on the "Search" button on the Internet Explorer main toolbar. This button will take you to a document within Microsoft's home site. On this document you will find a choice of categories to look through and a list Search Engines to use. A Search Engine is a application that will attempt to find any documents that contain the subject or phrase that you enter into the search parameters. You can also browse through the categories of Web sites that the search engines have already organized for you.

Another important thing to remember when you first begin browsing the web is that if you know the Web address of a site, you don't need use a search engine to find the page you wish to visit. Go up to the "Address Bar" near the top of the page, and click on it. Now you can type in the Web address of the site you want, and then press enter. Internet Explorer will go to this site directly from whatever document you were currently viewing. This is much faster than going to a search engine and trying to locate the site you want in their directories, or searching for it with a query (Address Bar shown below)

6.3.2 Address Bar

An Address bar is a space to type and display the address for a Web page. You do not even need to type the entire Web site address to go to that page. Just start typing, and the AutoComplete feature suggests a match based on Web sites you visited previously. With the Address bar, you can also search for Web sites just by typing find, go, or ? followed by the word that you are searching for.

Figure 6.4: Address Bar

6.3.3 Link Bar

The Link bar is a row of icons linked to Web pages. It is the most convenient, easiest-to-access place to put links to Web sites that you visit regularly. The Links bar comes loaded with Web sites that Microsoft wants you to visit, but need not stay that way. If you display the Links bar at all, you can claim it for yourself: remove the original links and insert new links to the pages that you visit most often.

6.3.4 Explorer Bar

The Explorer bar is a way to browse through a list of links, such as your History or channels, while displaying the pages those links open in the right side of the browser window. For example, if you click the Search button on the toolbar, the Explorer bar opens and you can use it to search for the Web site you want.

You can display your Favorites list, History list, channels, or Search by clicking the toolbar. In addition, you can gain access to these items by clicking the View menu, and then pointing to Explorer bar.

6.3.5 Viewing Window

This is the window where Web page will be displayed. The only way to hide this window is to minimize the entire IE window, since the main point of running a browser is to view pages.

6.3.6 Status Bar

This bar is placed at the bottom of the Internet Explorer window. It displays a variety of useful information. For example, when the cursor passes over a link in the viewing window, the URL of the link appears in the status bar. The progress of downloading a Web page is also displayed on this bar. If you want to hide the status bar, choose View >> Status Bar, uncheck its option.

6.4 Browsing the Web

The purpose of a Web browser is to show you Web pages. But once a page is visible, you may want to search it for a word or phrase, edit it as an HTML composing program, print it, save it, or even view its HTML source code. Web browsers contact servers, they are asking to view pages built with Hypertext Markup Language (HTML). They interpret those pages and display them on your computer. They display application programs, animations, and similar material created with programming languages such as Java and ActiveX, and scripting languages such as JavaScript. Sometimes, home pages contain links to files that the Web browser cannot play or display, such as sound or animation files. In that case, you will need a plug-in or a helper application. You configure your Web browser to use the helper application or plug-in whenever it encounters a sound or animation file that the browser cannot run or play.

Over the years, Web browsers have become increasingly sophisticated. They have become full-blown software suites that can do everything from video-conferencing to letting you create and publish HTML pages. They have also begun to blur the line between your local computer and the Internet in essence they can make your computer and the Internet function as a single system. To facilitate the usage of computers over Internet, Microsoft has integrated web browsing and the Internet directly into the operating system. For example, with Internet Explorer 4.0 and above, and with Windows 98 the Windows desktop can be HTML based. This means Web links can be directly embedded into the desktop.

So, you can have links to your favorite Web pages right on the desktop. And even applications such as word processors now have Web capabilities bust into them — such as being able to browse the Web, or build home pages. Even more significantly, using technology that Microsoft calls Active Desktop, Internet based desktop components can live on the desktop. These components can be things such as stock tickers, which deliver live Web content directly to the desktop. You don't need to fire up your Web browser to get the information; it's delivered straight to your Windows desktop without you having to do anything. Both Microsoft and Netscape have also built entire suites of software around their browsers. Netscape for example, calls its suite Netscape Communicator. Communicator includes modules for reading newsgroups; for reading, sending and managing internet mail for video conferencing; for collaborative work on whiteboard applications in which people can view and mark up the same documents simultaneously; and more. These enhancements will help user in what is called collaborative computing and communication.

Web browsers consist of software that runs on your computer and displays home pages on the Web. There are clients for PC, Macintosh and UNIX computers. A Web browser displays information on your computer, by interpreting the Hypertext Markup language (HTML) that is used to build home pages on the Web. Home pages usually display graphics, sound and multimedia files as well as links to other pages, files that can be downloaded and other Internet resources.

The coding in the HTML files tells your browser how to display the text, graphics, links, and multimedia files on the home page. The HTML file that your browser loads to display the home page does not actually have the graphics, sound, multimedia files, and other resources on it. Instead, it contains HTML references to those graphics and files. Your browser uses those references to find the files on the server and then display them on the home page. The Web browser also interprets HTML tags as links to other Web sites or to other Web resources – these include graphics, multimedia files, newsgroups, or files which can be downloaded. Depending on the kind of link, it will perform different actions. For example, if the HTML code specifies the link as another home page, the browser will retrieve the Uniform Resource Locator (URL) specified in the HTML file when the user clicks on the underlined link on the page. If the HTML code specifies a file to be downloaded, the browser will download the file to your computer.

It allows you to enter the address of the site you want to jump to (called a URL-or uniform resource locator), or to jump there by clicking hotspots, high lighted words, buttons, pictures, or icons called hyperlinks on your screen.

It formats Web documents for display on your screen.

It allows you to back up and go forward through pages you have already visited.

It allows you to copy text from the screen and paste it into a word processing program.

It allows you to print the document you see on the screen.

It makes it possible to transfer files-text, graphics, movies, animations, sounds, and programs – from other computers to your computer (called downloading).

It allows you to send and receive e-mail and other Internet services such as ftp (file transfer protocol), gopher, and Usenet news groups.

6.4.2 Basic Features of Browsers

Before we get involved in all the details, let us discuss some important browser features.

1. The Web browser should be able to look at the Web pages throughout the Internet or to connect to various sites to access information, explore resources, and have fun.
2. The Web browser must enable you to follow the hyperlinks on a Web page and also to type in a URL for it to follow.
3. Another feature of browser is to have a number of other commands readily available through menus, icons, and buttons.
4. Your browser ought to include an easy way to get on-line help as well as built-in links to other resources on the Web that can give you help or answers to your questions.

5. You will definitely want a way to save links to the sites you have visited on the WWW so that you can get back to them during other sessions. Web browsers take care of those in two ways through a history list, which keeps a record of some of the Web pages you've come across in the current session, and a bookmark list, which you use to keep a list of WWW pages you want to access every time you use your browser.
6. One of the main features of a browser is to search the information on the current page as well as search the WWW itself.
10. Browsers give you the facility to save a Web page in a file on your computer, print a Web page on your computer, and send the contents of a Web page by e-mail to others on the Internet
11. Few Web browsers (like Netscape Communicator) are complete Internet package, means they come with components like e-mail client, newsgroup client, an HTML composer, telnet client, ftp client etc.
12. Web browser should be able to handle text, images of the World Wide Web, as well as the hyperlinks to digital video, or other types of information.
13. To take advantage of some of the most exciting things on the World Wide Web, your browser needs to properly display and handle Web pages that contain animated or interactive items.
14. Netscape Navigator can incorporate these features through its ability to interpret programs written in Java and Java Script.
15. Web browsers interact not just with the Web, but also with your computer's operating system and with other programs, called plug-ins, that gives the browser-enhanced features.
16. Another important feature to insist on in your browser is caching. A browser that caches keeps copies of the pages you visit so that it does not have to download them again if you want to return to them. Reloading a page from the cache is much quicker than downloading it again from the original source.
17. The most important feature of any browser is ease of use. While all Web browsers are fundamentally simple to use, the one you settle on should be very easy to work with. It should function as a transparent window onto the Web.
18. If you will be browsing the Web from within a secured network, you may have to configure your browser to work through a special computer on your network called a proxy server. Most popular browsers let you configure them to work with a proxy server, but some don't, so find out if you will be working through a proxy before deciding on your browser. If you are, your ISP or system administrator will tell you if you need to do anything special to use your browser.

6.5 Downloading Information Using Internet

These are tools that keep track of many web sites around the world and let you search for particular items whenever you want. The result of a search is a custom list of links, pointing to whatever the search engine found that met your criteria.

You can find information on the Web in a variety of ways. When you click the Search button on the toolbar, the Explorer bar appears at the left of the window. It provides access to a number of search services that offer different kinds of searching capabilities. Try out the different search services to see

what kinds of information they provide. If you want to find information faster, you can use the Auto Search feature by typing *to, find,* or? Followed by a word or phrase, right in the Address bar. Internet Explorer immediately starts a search using its predetermined search service.

Then, after you go to a Web page, you can search for specific text on that page.

6.6 Web Search Engines

A program that searches documents for specified keywords and returns a list of the documents where the keywords were found. Although *search engine* is really a general class of programs, the term is often used to specifically describe systems like Alta Vista and Excite that enable users to search for documents on the World Wide Web and USENET newsgroups.

Typically, a search engine works by sending out a *spider* to fetch as many documents as possible. Another program, called an *indexer,* then reads these documents and creates an index based on the words contained in each document. Each search engine uses a proprietary algorithm to create its indices such that, ideally, only meaningful results are returned for each *query.*

These are tools that keep track of many web sites around the world and let you search for particular items whenever you want. The result of a search is a custom list of links, pointing to whatever items the search engine found that met your criteria.

You can find information on the Web in a variety of ways. When you click the Search button on the toolbar, the Explorer bar appears at the left of the window. It provides access to a number of search services that offer different kinds of searching capabilities. Try out the different search services to see what kinds of information they provide. If you want to find information faster, you can use the AutoSearch feature by typing *to, find,* or ? followed by a word or phrase, right in the Address bar. Internet Explorer immediately starts a search using its predetermined search service. Then, after you go to a Web page, you can search for specific text on that page.

6.6.1 Examples of Search Engine

Various search engines are listed below.

1. www.google.com
2. www.lycos.com
3. www.altavista.com
4. www.hobot.com
5. www.excite.com
6. www.freefind.com
7. www.dogpile.com
8. www.infoseek.go.com
9. www.yahoo.com
10. www.search.com

Where to Search First?

The last time we looked, the Open Directory Project listed 370 search engines available for Internet users. There are about ten major search engines, each with its own anchor Web site (although some have an arrangement to use another site's search engine or license their own search engine for use by other Web sites). Some sites, such as Yahoo, search not only using their search engine but also give you the results from simultaneous searches of other search indexes. Sites that let you search multiple indexes simultaneously include:

- Yahoo (http://www.yahoo.com)
- search.com (http://search.com)
- EasySearcher (http://www.easysearcher.com)

Yahoo first searches its own hierarchically structured subject directory and gives you those entries. Then, it provides a few entries from the AltaVista search engine. It also launches a concurrent search for entries matching your search argument with six or seven other major search engines. You can link to each of them from Yahoo (at the bottom of the search result page) to see what the results were from each of these search engines.

A significant advantage of a Yahoo search is that if you locate an entry in Yahoo, it's likely to lead to a Web site or entire categories of sites related to your search argument.

A search.com search primarily searches the Infoseek index first but also lets you search the other major search engines as well.

EasySearcher lets you choose from either the popular search engines or a very comprehensive collection of specialized search engine/databases in a number of fields.

6.6.2 Working of Search Engine

Search engines use software robots to survey the Web and build their databases. Web documents are retrieved and indexed. When you enter a query at a search engine website, your input is checked against the search engine's keyword indices. The best matches are then returned to you as hits.

By "How to Search," we mean a general approach to searching: what to try first, how many search engines to try, whether to search USENET newsgroups, when to quit.

1. If you know of a specialized search engine such as SearchNetworking that matches your subject (for example, Networking), you'll save time by using that search engine. You'll find such specialized databases accessible from Easy Searcher 2.
2. If there isn't a specialized search engine, try Yahoo. Sometimes you'll find a matching subject category or two and that's all you'll need.
3. If Yahoo doesn't turn up anything, try AltaVista, Google, Hotbot, Lycos, and perhaps other search engines for their results. Depending on how important the search is, you usually don't need to look below the first 20 entries on each.
4. For efficiency, consider using a ferret that will use a number of search engines simultaneously for you.

5. At this point, if you haven't found what you need, consider using the subject directory approach to searching. Look at Yahoo or someone else's structured organization of subject categories and see if you can narrow down a category your term or phrase is likely to be in. If nothing else, this may give you ideas for new search phrases.

6. If you feel it's necessary, also search the Usenet newsgroups as well as the Web.

7. As you continue to search, keep rethinking your search arguments. What new approaches could you use? What are some related subjects to search for that might lead you to the one you really want?

8. Finally, consider whether your subject is so new that not much is available on it yet. If so, you may want to go out and check the very latest computer and Internet magazines or locate companies that you think may be involved in research or development related to the subject.

There are two primary methods of text searching--keyword and concept. Keyword searching is far more common.

6.6.3 Keyword Searching

This is the typical form of text search on the Web. Most search engines do their text query and retrieval using keywords.

Unless the author of the Web document specifies the keywords for her document (this is possible by using meta tags), it's up to the search engine to determine them. Essentially, this means that search engines pull out and index words that are believed to be significant. Words that are mentioned towards the top of a document and words that are repeated several times throughout the document are more likely to be deemed important.

Most search engines now index every word on every page. Others index only part of the document, such as the title, headings, subheadings, hyperlinks to other sites, and the first 20 lines of text.

Full-text indexing system generally pick up every word in the text except commonly occurring stop words such as "a," "an," "the," "is," "and," "or," and "www." AltaVista claims to index all words, even the articles, "a," "an," and "the." Some of the search engines discriminate upper case from lower case; others store all words without reference to capitalization.

6.6.4 Concept-based Searching

Unlike keyword search systems, concept-based search systems try to determine what you mean, not just what you say. In the best circumstances, a concept-based search returns hits on documents that are "about" the subject/theme you're exploring, even if the words in the document don't precisely match the words you enter into the query.

Excite is currently the best-known general-purpose search engine site on the Web that relies on concept-based searching.

This is also known as clustering -- which essentially means that words are examined in relation to other words found nearby.

How does it work? There are various methods of building clustering systems, some of which are highly complex, relying on sophisticated linguistic and artificial intelligence theory that we won't even attempt to go into here. Excite sticks to a numerical approach. Excite's software determines meaning by

calculating the frequency with which certain important words appear. When several words or phrases that are tagged to signal a particular concept appear close to each other in a text, the search engine concludes, by statistical analysis, that the piece is "about" a certain subject.

For example, the word heart, when used in the medical/health context, would be likely to appear with such words as coronary, artery, lung, stroke, cholesterol, pump, blood, attack, and arteriosclerosis. If the word heart appears in a document with others words such as flowers, candy, love, passion, and valentine, a very different context is established, and the search engine returns hits on the subject of romance.

6.6.5 Refining Your Search

Most sites offer two different types of searches--"basic" and "advanced." In a "basic" search, you just enter a keyword without sifting through any pulldown menus of additional options. Depending on the engine, though, "basic" searches can be quite complex.

Advanced search refining options differ from one search engine to another, but some of the possibilities include the ability to search on more than one word, to give more weight to one search term than you give to another, and to exclude words that might be likely to muddy the results. You might also be able to search on proper names, on phrases, and on words that are found within a certain proximity to the search terms.

Many search engines now automatically recognize company names and can direct a search to the corporate website when such a name is entered as a query. Phrase recognition is also becoming more common; i.e., you might expect to get relevant hits for the term Cold War if you enter it without the quotation marks that typically denote a phrase. (In the past, you simply would have received all documents with the words "cold" and "war" in them.

6.6.6 Meta Tags

Some search engines also allow you to specify what form you'd like your results to appear in, as well as whether you wish to restrict your search to certain fields on the internet (i.e., Usenet or the Web) or to specific parts of Web documents (i.e., the title or URL).

Many, but not all search engines allow you to use so-called Boolean operators to refine your search. These are the logical terms AND, OR, NOT, and the so-called proximal locators, NEAR and FOLLOWED BY.

Boolean AND means that all the terms you specify must appear in the documents, i.e., "heart" AND "attack." You might use this if you wanted to exclude common hits that would be irrelevant to your query.

Boolean OR means that at least one of the terms you specify must appear in the documents, i.e., bronchitis, acute OR chronic. You might use this if you didn't want to rule out too much

Boolean NOT means that at least one of the terms you specify must not appear in the documents. You might use this if you anticipated results that would be totally off-base, i.e., nirvana AND Buddhism NOT Cobain.

Not quite Boolean + and - Some search engines use the characters + and - instead of Boolean operators to include and exclude terms.

NEAR means that the terms you enter should be within a certain number of words of each other.

FOLLOWED BY means that one term must directly follow the other.

ADJ, for adjacent, serves the same function. A search engine that will allow you to search on phrases uses, essentially, the same method (i.e., determining adjacency of keywords).

Phrases: The ability to query on phrases is very important in a search engine. Those that allow it usually require that you enclose the phrase in quotation marks, i.e., "space the final frontier."

Capitalization: This is essential for searching on proper names of people, companies or products. Unfortunately, many words in English are used both as proper and common nouns--Bill, bill, Gates, gates, Oracle, oracle, Lotus, lotus, Digital, digital--the list is endless.

All the search engines have different methods of refining queries. The best way to learn them is to read the help files on the search engine sites and practice!

Some search engines are now indexing Web documents by the meta tags in the documents' HTML (at the beginning of the document in the so-called "head" tag). What this means is that the Web page author can have some influence over which keywords are used to index the document, and what description of the document that appears when it comes up as a search engine hit.

This is important if you are trying to draw people to your website based on how your site ranks in search engines hit lists.

There is no perfect way to ensure that you'll receive a high ranking. Even if you do get a great ranking, there's no assurance that you'll keep it for long. For example, in April 1999 one of our Spider's Apprentice pages was the number one ranked hit on AltaVista for the phrase "how search engines work." A few months later, however, it had dropped way down in the listings.

There is a lot of conflicting information out there on meta-tagging. If you're confused about this subject, it may be because different search engines look at meta tags in different ways. Some rely heavily on meta tags, others don't use them at all. As of this update (2001), the general opinion seems to be that meta tags are being used less than they were a couple of years ago, because of the high rate of spamming (web authors using false and misleading keywords in the meta tags).

Of all the possible meta tags (and there are many), the "keywords" and the "description" meta tags are important to the search engines that index meta tags.

Some search engines will use the "description" meta tag as their short summary of your URL, so make sure your description is one that will entice surfers to your site.

As for the "keywords" meta tag, list a few keywords, synonyms for keywords, or foreign translation of keywords (if you anticipate traffic from foreign surfers). Make sure the keywords refer to or are directly related to the subject or material on the page. Do NOT use false or misleading keywords in an attempt to get a higher ranking.

As noted above, the "keyword" meta tag has been abused by some webmasters. For example, one popular ploy was to put the words "Pamela Anderson" into keyword meta tags, in hopes of luring searchers to one's website by using the keywords for one of the most popular searches on the Web.

The search engines are aware of such deceptive tactics, and have devised various methods to circumvent them. Use keywords that are appropriate to your subject.

One thing you should never do is use some other company's trademarks in your meta tags. Some website owners have been sued for trademark violations because they've used other company names in the meta tags.

6.7 Search Engine (ALTA VISTA)

As the second largest search engine on the net Alta Vista is not one to be ignored. Alta Vista offers a comprehensive suite of services, and is one of the few search engines that clearly state their policies concerning tricks to inflate the performance of a web site.

Alta Vista also offers some unique features, such as Alta Vista's translation services and their image finder. They have consistently strived to provide a quality index, and a level of service beyond the scope of the normal search engine.

AltaVista search engine was introduced in 1995. Alta Vista quickly became a major player in the search engine fields. Alta Vista at a Glance is shown in Figure 6.5 and 6.6.

Size	125 Million Urls (approx)
Spider Class	Deep
Meta Tag support	Yes
Frame support	Yes
Image Map support	Yes
Alt Text support	Yes
HTML Comments	No
Url Searching	Yes
Embedded Directory	Yes
Submission URL	Submit

Figure 6.5

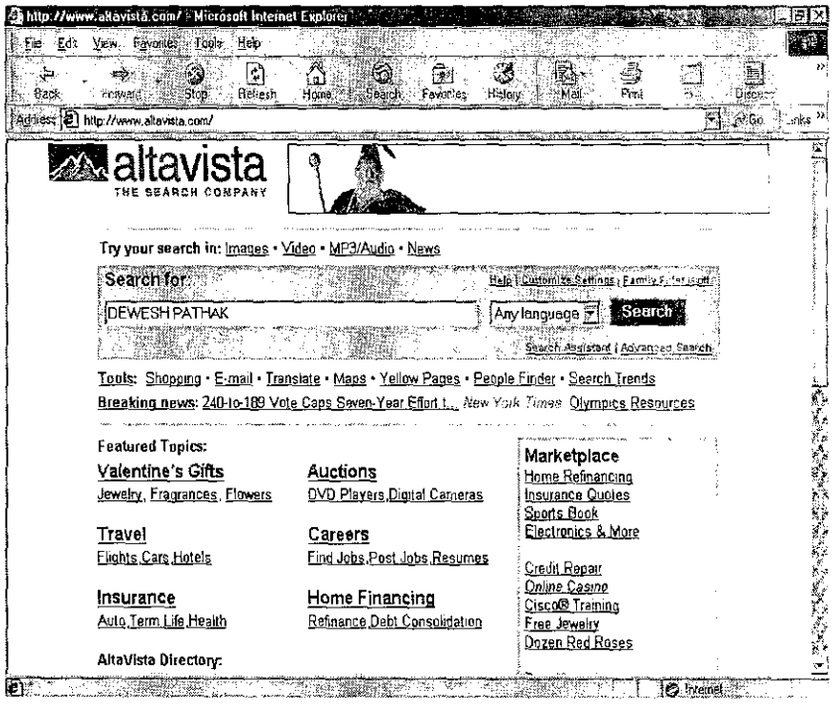


Figure 6.6

Submission Policy

"Sometimes sites submit a large number of pages to AltaVista hoping to have them show up often on our result pages. They submit pages with numerous keywords, or with keywords that are unrelated to the content of the pages.

Attempts to fill AltaVista's index with misleading or promotional pages lower the value of the index for everyone and render Web indices and your search experience worthless. Alta vista do not allow URL submissions from customers who Spam the index and will exclude all such pages from the index."

Alta vista further stated that they frown upon:

1. pages with text that is not easily read, either because it is too small or is obscured by the background of the page, pages with off-topic or excessive keywords,

2. duplication of content, either by excessive submission of the same page, submitting the same pages from multiple domains, or submitting the same content from multiple hosts,

3. machine-generated pages with minimal or no content, whose sole purpose is to get a user to click to another page,

4. pages that contain only links to other pages, or pages whose primary intent is to redirect users to another page.

Alta Vista is committed to keeping their index as "clean" as possible.

Alta Vista has changed their submission policy to halt the submission of sites from automated submission systems. As a result they now display a computer generated series of alphanumeric symbols which you must hand re-type in order to successfully submit your site to them. This makes it impossible for a submission to Alta vista from any sort of program. You must either hand submit your site to Alta Vista or you must insure that the submission service you use, hand submits for you.

6.8 Gopher

Gopher protocol is used by the gopher system that was designed at the university of Minnesota and named after the schools Athletic teams, the golden Gophers (meaning Go for i.e. go fetch). It is an information retrieval scheme, conceptually similar to the web itself, but supporting text and no images. When the user logs into a gopher server, he is presented with a menu of files and directories, any of which can be linked to another gopher menu anywhere in the world.

Gopher's big advantage over the web is that it works very well with 25x80 ASCII terminals, of which there are still quite a few around, and because it is a text based, it is very fast. Using the gopher protocol, web users can access gopher and have each gopher menu presented as a click able web page

Gopher is a menu based document delivery system. Individuals use Gopher to access various types of information such as files, documents, address books, and images. Gopher also allows access to FTP, Telnet, and searchable databases. Selecting items do all of this from menus. Gopher is a menu based document delivery system.

6.9 Veronica

Veronica is a tool that can be used to perform a search and look for all the menu items in gopher space containing certain keywords. A related tool, Jug head, does the same thing for a specific group of gopher menus. After veronica or jug head finishes searching, a menu appears on the screen containing the names of whatever items were found.

6.10 MOSAIC

This browser is similar in function to Netscape and Internet Explorer, though it differs in specific features. It is also said as an alternative to the granddaddy of all web browsers that made us love web that is Netscape, even though it is small and well maintained, and free. Look for the well-maintained part to fall by the wayside, since the NCSA recently suspended support for the package. Other features include the Auto surf crawler, which automatically retrieves specified pages for offline viewing, and an option to reload text only while pulling images straight from the cache. Don't look for Java applets JavaScript support, or frames here, however download it.

6.11 WAIS

WAIS (**W**ide **A**rea **I**nformation **S**erver) is a combination of indexing and querying software that allows a WAIS administrator to create a database of indexed documents. The WAIS database contains a list of all of the words (except common words such as 'the') in all of the indexed documents. A WAIS user can then search this database for a keyword or combination of keywords and get back a list of documents which match their search criteria. The information returned by WAIS in response to a query is quite complex, but is usually processed by a search program (such as ours) to display only the most useful information

6.11.1 WAIS Indexing Features

- Rapid Indexing
    - The WAIS indexing program is quite fast, indexing hundreds of documents in seconds
- Automatic "Headlines"
    - The WAIS indexing program automatically generates a "headline" for each document indexed. How the headline is generated depends on the "WAIS type" of the document, as specified by the WAIS administrator.
- Multiple Types
    - Many different types of data files (HTML, text, graphics, postscripts, etc.) can be combined into a single WAIS database.

6.11.2 WAIS Query Features

Boolean Searching

The WAIS query software allows "Boolean" searching. This means that you can use multiple keywords in your search, as well as the Boolean operators AND, OR, and NOT. This allows a user to tailor their query to return precisely the documents required.

Automatic Weighting

The WAIS query software automatically "weights" the results of a query based on factors such as how many times the keyword(s) occur, whether they occur in the headline, how close together they occur, etc. The closer the weight (score) is to 1000, the closer the match. Results are returned in weight order, highest first.

RadixNet's WAIS Interface

RadixNet has it's own WAIS Interface, designed to make using WAIS as easy and as productive as possible for both the WAIS end-user and the WAIS administrator. Unique features include:

Simple, Web-Based Interface

RadixNet's WAIS interface is based almost entirely on Web pages, forms, and scripts. End-user and administrator alike can simply fill out and submit forms through their favorite web browser. In case of questions, additional information is only a mouse-click away.

Easy-to-Read Formatting of Results

The results of a WAIS indexing or query are specially formatted to make them useful and easy to read. Instead of simply *listing* the results in plain text, RadixNet's WAIS Interface turns the results into a *fully formatted web page*. Search results are displayed with headlines shown in **bold** and automatic links to the actual documents.

Automatic Linking to Actual Documents

RadixNet's Search Interface displays the names of documents in search results as actual *links*, not just long, confusing file names. To view a listed document, simply *click* on the document's name.

WAIS stands for Wide Area Information System. The newer tools and programs provide a common method to access almost all the others. Having a single way to access all the Internet services means having more than one super-duper program or exceptional tool. The idea behind the World Wide Web (WWW) is a single means of access to virtually everything available through the Internet: services, resources, tools, and information.

6.12 Internet Relay Chat (IRC)

IRC is an Internet wide talk facility developed in 1988 by Jarkko Oikarinen in Finland and is used in over 60 countries around the world. The Internet Relay Chat (IRC) service provides a way for many users to communicate about a given topic. Each communication occurs on a separate "channel". A user who creates a channel chooses a topic and specifies whether the channel is open to anyone or restricted

to the set of people specified by the channel creator. A user can request a list of the IRC channels currently in progress, and can choose to join one of the channels. When a user joins an IRC channel, the user enters a nickname that IRC uses to identify the user to others (e.g., dancer or big shot). Users often choose nicknames that disguise their identity. For example, a woman might choose computer man. A can be used by anyone on the Net at any time.

A participant receives each line of text that another participant enters along with the participant nickname. Thus, a participant who enters a line of text knows that all other participants will receive a copy of the text on their screens.

6.12.1 Internet's Chat Protocol

Computer and software that work like IRC switchboards, letting users connect to them by using an IRC client programs are known as IRC servers. IRC works because a series of IRC servers band together in a network to share channels of communication, like communicating with everyone on a single radio frequency. If you connect to one server in such a network, you have access to all the channels and all the users connected to any of the servers on that network.

- Most IRC servers are at universities or ISPs. There are three major networks of IRC servers EFnet, UnderNet, and DALnet.

- EFnet: It is short for Eris Free net, the first, traditional, and largest net. If you are on IRC but not sure which net you are using, then chances are more of you being on EFnet. More than almost 40,000 people all over the world are always connected to EFnet, at any time – day or night.

- UnderNet: In 1992 this net was evolved. UnderNet, is smaller than ERnet, and more community oriented alternative network. The name UnderNet was originally chosen to refer to as "underworld net", hidden from the main IRC network. It was formed when a group of IRC users grew disconnected with what they perceived as privacy breaches and slowdowns, and decided to form a new net. For more about this net visit at: http://www.undernet.org.

- DALnet: One of the oldest nets, it keeps pace with UnderNet. It started as a role-playing-game alternative network. See http://www.dal.net for more information.

To connect to an IRC net, you connect to a server on that net. Each net has its own group of servers means that if you want to access the UnderNet. Each server has its own host name, usually consisting of the location of the server. For example, every UnderNet server has a host name ending with undernet.org. The first time you run an IRC client, you might have to select a server to connect to (or it might automatically select one for you). After that, you will connect whenever you start the program.

The next step is to list what channels (conversation "rooms") are available out there and choose one to join. Then you join a channel. You will be able to see who else is on same chat channel, and you will also see what everyone types (except for their private communications). You can type messages to the channel or send private message to individuals. Quit the channel when you are done and then the program. The following Table 6.1 lists a few servers for each of the above three-major IRC nets. Add them, if not in your IRC server list. These servers should give you a basis to start exploring IRC.

Table 6.1

EFnet Servers	UnderNet Servers	DALnet Servers
ir.magic.ca	ann-arbor.mi.us.undernet.org	irc.dat.net
irc.colorado.edu	austin.tx.us.undernet.org	irc.service.dal.net
irc.c-com.net	blacksburg.va.us.undernet.org	glass.oh.us.dal.net
irc.blackened.com	chicago.il.us.undernet.org	groucho.ca.us.dal.net
irc.stanford.edu	davis.ca.us.undernet.org	datashopper.dal.net
newbrunswick.nj.us.undernet.org		

6.12.2 IRC Clients

To use IRC you need an IRC client program to connect to a server. Once you are connected, your client acts as your interface into IRC. If you use a shell account, you will run a UNIX client on the remote host. If you use the Net via PPP or a direct network connection, you will run a client program on your own computer. The original IRC programs were text-based UNIX programs named **irc** and **ircii.** There are number of IRC clients available on the Net. For example, mIRC, Microsoft Chat, V-Chat for Windows, and Ircle for Macintosh.

6.12.3 Channels

Users meet in channels, the IRC equivalent of a chat room. When you first connect to IRC, you choose a channel to join. An unlimited number of users can be in a channel, and it takes only one user to open or create a channel. Each channel name starts with a sign (#). When you feel like switching, you can join another one. You can join as many channels as you want at the same time. At one particular time, you will find many different channels. For instance, EFnet has about 7,000 channels during the day and more than 8,000 in the evening hours.

Some channels are for discussing specific topics. Others are just for talking about whatever comes up. There are public, private, and secret "invisible" channels. How do you create a channel? It is very simple. Whenever you enter the command to join a channel, IRC checks to see if that channel already exists. If so, you can join it. Otherwise, IRC will create a new channel using whatever name you specified. When the last person leaves a channel, IRC removes it automatically.

6.12.4 Nicknames

Each IRC user has a nickname, a unique word he has chosen. Each time you connect to an IRC server, you specify a nickname, which you can also change at any time.

The nickname should be nine characters or less. Some chatters have an at-sign(@) at the beginning of their nickname. An @ signifies that a person is a channel operator or chanop – the person who manages the channel.

Nicknames are important as they allow people to participate freely and openly while still retaining their anonymity.

6.13 Web-Chat

A Web site is often more than simply something to look at or a resource to use. Many Web site operators want to provide their users with a community that encourages them to be part of the

experience. Web based chat systems fill this goal, building an online network of people who interact not just with the Web page, but with other users as well. Web based chat is accessed through your Web browser. Chat on a Web site usually falls into either of two broad types — interactive i.e., immediate chatting which is much like IRC, and that are similar to Usenet newsgroups or mailing lists. Fortunately, chatting on the Web is very simple — easier than using an IRC program or newsreader— so learning the basics will enable you to adapt to any system that Web site operator might provide.

6.13.1 Interactive Web Chat

Web sites that enable you to converse directly with other Web users in a manner similar to IRC are providing interactive Web chat. Usually, these sites are referred to as chat rooms, a concept popularized by America Online. However, America Online chat rooms can be used only by AOL users whereas anyone on the Internet with the right software can access a Web based chatting system.

You can find two main categories of interactive chat systems. Some are basically textual in nature you type something, and what you type appears on the screens of people around the world who are connected to the same site. Others use graphics and even 3-D animation to provide you with a visual avatar — your online representation in the chat room, which interacts with other users' avatars as you send and receive messages. Most chat rooms are written in Java, a computer language that enables programs to be distributed and run over the Internet.

Text based chat rooms are very much like IRC. You use your Web browser to connect to the Web site choose a handle, and see the messages in the chat room; when you type messages, anyone using the same chat room can read what you type. For example, Figure 6.7 shows the Yohoo! Chat site at http://chat.yahoo.com.

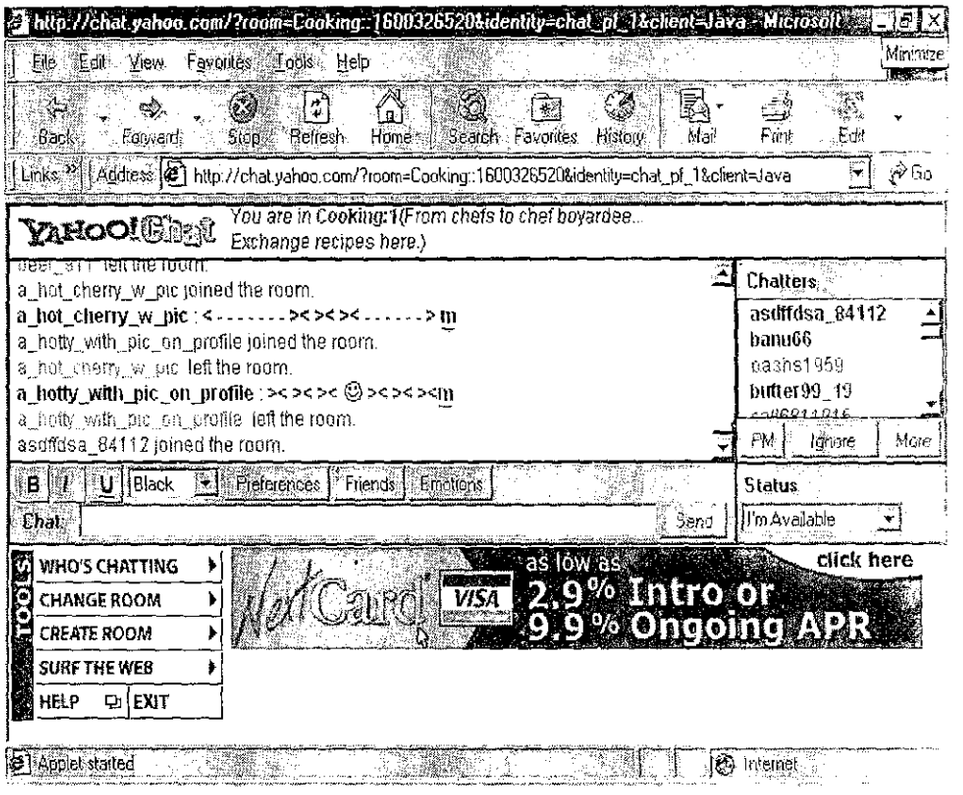


Figure 6.7: Chatting on the Yahoo! Chat Web site

6.14 E-Mails

On most networks, electronic mail (e-mail) is widely used application. In fact of all TCP connections that Internet users establish, about one-half are sending and receiving e-mail messages. Even though you may never need to write your own Internet application, studying e-mail concepts is not waste of your time. Internet Email concepts provide you with an excellent example of how to implement a simple yet effective Internet service. Each e-mail messages has a sender and receivers, both of which have a user – interface into the networks electronic –mail system. The network e-mail systems consist of about going queue (collection of messages), a client process, a server process, and mailboxes for incoming mail. Although the user-interface (the e-mail application program) is often an integral part of an e-mail system, it doesn't have to be. In other words, the user-interface can be separate client program that uses a client/server model to interact with the e-mail system. A **mailbox** can refer to a user address (user identification and host name- username @name. domain) or **container file** that stores e-mail data.

Electronic mail, or E-Mail, lets you communicate with other people on the Internet. E-Mail is one of the basic Internet services, and by far, the most popular. It can be used for conversation purpose, to keep in touch with friends, get information, start relationships, or express your opinion. It is called E-Mail because:

Electronic mail can be defined as the transmission of letters and memos from one computer to another. When E-Mail originated in the 1970s, it was just the sending of messages. The capability to send various items has rapidly become true of E-Mail: users now can attach spreadsheets, business forms, lengthy documents, scanned images, faxed images, computer graphics, meeting schedules, sound, and video to their messages (and the list continues to expand).

Basic structure of your e-mail is given in the Figure 6.8 below where basic elements in the network are explained.

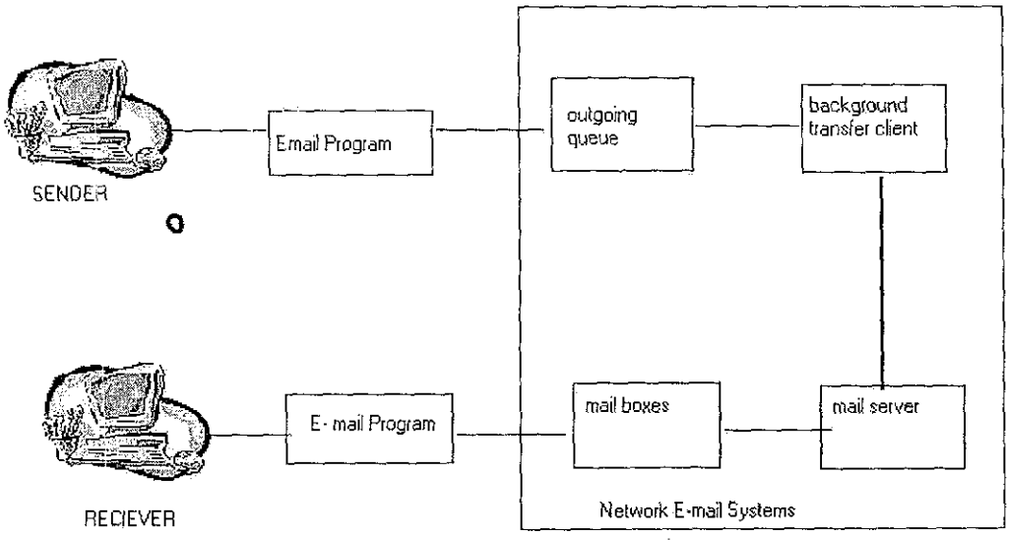


Figure 6.8: Basic Structure of E-Mail

You put it into an electronic envelope and address it.

You post it or hand the message to someone else (i.e., the network) to be delivered.

You may not know when the E-Mail is read.

You get E-Mail back in you mailbox, if you addressed it incorrectly.

If the recipient leaves a forwarding address, the E-Mail system will keep trying to route it to him/her until it runs out of forwarding locations.

If the network is unable to deliver your E-Mail, it will return the mail (this is called bounced mail)

As E-Mail does travel over computers and computer networks instead of in trucks and airplanes, it's important to remember some differences between E-Mail and paper mail:

Your E-Mail may reach people you did not intend to send it, because it is very easy for other people to pass around or forward on.

The transmission of E-Mail message through the Internet relies on the SMTP, which stands for Simple Mail Transfer Protocol. SMTP is part of the TCP/IP family of protocols. The SMTP protocol is used to transport messages between computer systems in the Internet. SMTP uses TCP, Transmission Control Protocol, which provides a reliable means of communication. Throughout the Internet, there are millions of computers using SMTP to send and receive mail.

SMTP protocol is used to send and receive mail behind the scenes. The question now arises, how does the mail get from the transfer agent to you. The computer that provides the Internet connection also acts as the mail host. Typically, this computer runs a transport agent program, which is connected to the Internet 24 hours a day. This means, whenever your mail arrives, the transport agent available accepts it and saves it in a file called a MAILBOX. Each person who has an account on the host computer is given his own mailbox file. In this way the host computer always keeps everyone's mail in an organized manner and at the same time it assures you that no one can read your (or anyone else's) messages

How you access your mailbox depends on what type of Internet account you have. If you have a shell account, you run all your Internet programs on the host computer. To run the mail system, you run a Unix mail program, called a USER AGENT, on the host computer. The user agent acts as the interface between you and the Internet mail system. The two most popular UNIX mail programs are called Pine and Elm. If you have a PPP or SLIP account, you use the mail system by running a MAIL CLIENT on your own computer. Whenever your client needs to send or receive mail, it connects to a MAIL SERVER which runs on the host computer. Together, the mail client and the mail server provide the functionality of a user agent. The mail client and mail server communicate with one another by using a system called POST OFFICE PROTOCOL or POP. For this reason, a mail server of this type is sometimes referred to as a POP SERVER.

6.14.1 Components of E-Mail

The headers are pieces of information that tell you and the E-Mail system a number of things about a particular piece of E-Mail. Each of these headers has a specific name and a specific purpose. You will see some, but not necessarily all of the headers each time you read a piece of E-Mail. You have to fill in all of them, they are all generated and put in the proper form by the E-Mail program you use. Here we have covered the most important header lines. However, you should understand there is considerable variation in how the lines may be constructed.

Table 6.2

Header Type	Description
From	This line shows who sent the message. In our previous example, the message was sent by user ID GeethaSham@in.Oracle.com.
Return Path	This tells your mail program where to send a reply if you choose to respond to the message.
Date	This shows the date and time that the message was sent, according to the sender's computer.
To	This shows the user ID (E-Mail) of the person to whom the message was addressed. This means that if the message is for you, this line will contain your address. If the message was also sent to other people, their addresses will appear on this line as well.
Cc	This line shows any user IDs who are to receive copies of the message. That is additional recipient(s) of the message. (Cc is an abbreviation for "carbon copy", a term that is outdated but still in use.)
Subject	This line shows the subject of the message. This is a short description that is typed by the person who composes the message. If the subject starts with the letters Re:, it indicates the message is a response to a previous message. For example, say that you send a friend a message with the following subject:
	Subject: Congratulations! Party next Saturday?
	If your friend uses his mail program to send a reply to you, the mail program will automatically insert the characters Re: at the beginning of the subject: line. Now the subject line will look like this:
	Subject: Re: Congratulations! Party next Saturday?
	Think of Re: as meaning "This is a reply to the previous message with the subject ... "
Reply-To or Return-Path	Your E-Mail application automatically uses this address when you reply to the message.
Received	Contains information from each host service that relayed the message.
Message-ID	The unique ID that identifies this message (generally not useful).
X-Sender	Adds a layer of authentication to the message by identifying the sender.
Mime-Version	The version of MIME used (the Multipurpose Internet Mail Extension is used for attachments and for HTML-formatted messages).
Content-Type	The MIME data format used. Frequently, the data format is text/plain with some further information to identify the text type.
Lines	Number of lines of text in the message.
X-UIDL	A unique identifier added by some POP E-Mail applications to identify messages that have been downloaded.

6.14.2 Message Body

The second part of the message is the actual content of the E-Mail; what you send and what you receive. When you are sending E-Mail to a computer system where your message will be interpreted by a computer program, you will be given instructions to use specific words or phrases in the message body.

6.14.3 Signature

The signature is not a signed name but a sequence of lines usually giving some information about the person who sent the E-Mail. It's optional and it is made up of anything the user wants to include. Usually a signature has the full name of the sender and some information about how to contact the person by E-Mail, phone, or fax. Many people also include a postal address, description of their organization, job title, and even favourite quotation or some graphics created by characters typed from the keyboard.

You do not have to type in the signature each time. E-Mail programs will automatically append the contents of a specified file to each outgoing message. The name of the file depends on the program you're using for E-Mail. Some common names are signature, .sig, or .signature.

The ability to send attachments was a great stride forward in the development of E-Mail. By attaching files to E-Mail, you can exchange documents for revision, pass on spreadsheets for data entry, or send a presentation for review. Of course, you can also attach electronic pictures, sounds, movies or whatever can be put in file form. As E-Mail was originally designed to convert only text, your E-Mail program must convert other types of files to a text like format that can pass through the Internet mail system. The receiving E-Mail program converts the message back to its original format. The following are the three most common formats for E-Mail attachments:

- IME: MIME is the newest and best standard method for sending attachments.
- Uuencoding: Uuencoding is the old standard and is the only method supported by some older E-Mail applications, especially UNIX E-Mail programs.
- BinHex: BinHex, the least common format, is used primarily by Mac E-Mail programs.

6.15 E-mail Packages

There are many ways to access your E-Mail:

- You may use a mail client, such as Eudora, Outlook, or any one of the other popular packages that download your incoming messages from the POP server to your computer and upload your outgoing messages to the SMTP server. This may occur through a local area network (LAN) or through a dial-up connection.
- You may use a Web based E-Mail service. These Web sites are described in the section "Web Based E-Mail," later in this chapter.
- You may use a commercial provider, such as CompuServe or America On-line, which have their own E-Mail programs.

- You may get your E-Mail through a LAN, a common system at large organizations. If your organization has some sort of Internet connection, E-Mail arrives in the company's POP server. You then read your E-Mail either on the server, using an E-Mail application, or on your own computer, by downloading your E-Mail from the server through the LAN by using an E-Mail application. Your company may use a POP server or some kind of proprietary protocol (for instance, Lotus cc: Mail, which is not a POP mail client).

- You may have a UNIX shell account and use a UNIX E-Mail program (such as Pine, Elm, or Mutt) that reads your POP mailbox directly.

- Here are a number of E-Mail software available these days. They are generally of two kinds:

- E-Mail services that are available on a host you access either by Telnet or a direct link (for example, Pine); and

- E-Mail software that runs on your personal workstation (for example, Eudora).

With Pine, you explicitly run the Pine interface and deal with your mail. If you have multiple windows, you can leave a Pine Window running, but otherwise you go into Pine, handle your mail, and go back to your other tasks.

Eudora runs like any other application on your machine. You can leave it running in the background to deliver your mail and look for new mail, or shut it down to save on phone expenses.

For sending or composing a message in Pine, perform the following steps:

1. Enter C from the Pine main screen to select message composition. A message composition window will be displayed.

2. Specify the Headers (To, Cc, Subject. etc.).

3. Type the message text.

4. Send the message through CTRL-X command.

Sending a file as attachment:

1. Press CTRL-J. Pine will ask you the file name and a comment.

2. You can also specify the file by moving to the Attachment: line and type the file name.

3. You can send more than one file as attachment.

Reading Your E-Mail

When you invoke Pine, your mail is automatically incorporated into your inbox, as illustrated in Figure 6.9. Then, you can call up a menu of available mail to choose which ones you want to read. When you highlight and choose a message, it appears in a new window.

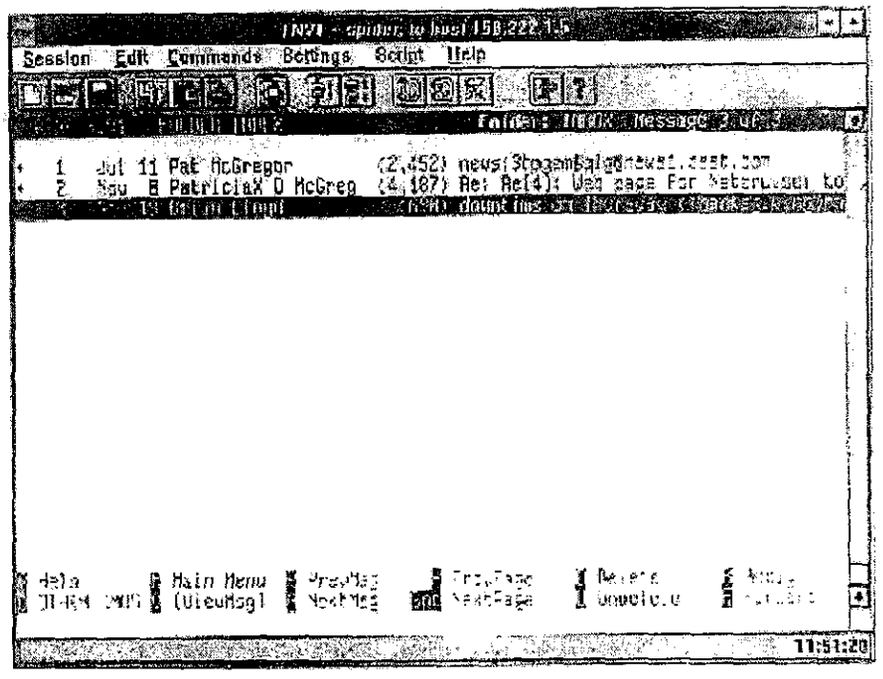


Figure 6.9: The Pine Inbox

With each message you will see following information:

- Date
- Name of the sender
- Size of the message in bytes
- Subject of the message
- Certain codes in the extreme left

+ (plus)	:	This means the message was sent "directly to you". In other words, the message is not a copy of a message sent to someone else; nor was it sent to you as part of a mailing list.
N	:	Means the message is new and you have not yet looked at it.
D	:	This means the message has been marked for deletion.
A	:	Means you have already answered the message.

To read the message, highlight it using the arrow keys. Press Return, the message will appear on the screen.

While you are reading a message, there are several commands to use: to page through the message, search for patterns, and control what you see. For all these operations and more, the Table 6.3 below summarizes the commands used to read a message.

Table 6.3

Command Key	Description
DOWN or N	Move to next message
UP or P	Move to previous message
ENTER or V	View the message
SPACE or + or ^v	Move down one screen
^Y or -	Move up one screen
JnumberRETURN	Jump to message number
WpatternRETURN	Jump to next occurrence of pattern
WRETURN	Jump to previous occurrence of pattern
V	View attachment
M	Return to main menu
C	Compose a new message

Deleting a Message

While you are reading a message, there are number of things you can do with it. For example, you can delete the message, reply it, send a copy to someone else, and so on. The most important command is D (delete). If you have read a piece of mail and you are positive that you do not want to save it, you should delete it so it does not clutter up your Inbox (and waste precious hard disk storage space). To delete a message, highlight it and press D.

If you change your mind and decide that you really did not want to delete a message, choose either of the following methods:

- Use P (previous) command, to move back to the message, then press U to undelete the message.
- Press I to return to the index, and then move to the message and press U.

6.16 Pine

Pine is designed for ease-of-use with the novice computer user in mind. It is based on Internet mail protocols and currently runs on a variety of UNIX platforms over any type of network connection. Originally based on the Elm program, Pine has evolved much since the original version was written. The guiding principles for achieving ease-of-use in Pine are: careful limitation of features, one-character commands, accessible command menus, immediate feedback to the user, and high tolerance for user mistakes. Pine can be learned by exploration rather than by reading manuals. This document should be

read at your computer while you use Pine. The text that follows does not tell you everything about Pine It only introduces Pine's main options and summarizes basic email guidelines. The best way to learn to use Pine is to explore it on your own. Pine was designed to make it easy for you to learn to use email there is information on each screen and in the online help that shows you what to do or answers your questions. Try the different options and have fun experimenting with Pine.

Before you can use Pine to correspond by email, you need to have a userid (an account) on a computer and log in.

6.16.1 Features of Pine

- There is an address book for saving addresses and personal distribution lists using nicknames Pine's message composition editor, Pico, is a very simple and easy-to- use text editor with text justification and a spelling checker. It also assists with entering and formatting addresses and provides direct access to the address book.
- Pine has a Mail Index showing a message summary, which includes the status, sender, size date and subject of messages.
- You can view and process mail with commands such as forward, reply, save, export, print, delete, capture address, and search.
- There are multiple folders and a folder management screen for filing messages.
- On-line help is available specific to each screen and context.
- Pine supports access to remote mail repositories. Pine also supports multi-part mail conforming to the proposed Multipurpose Internet Mail Extensions standard (MIME). In future, this will allow attachments to mail messages such as graphics (GIF, TIFF, etc.), sounds, spreadsheets and other binary files.

6.16.2 Starting Pine

There are two ways to start Pine. Either choose the Pine menu item under the Basic Services UNIX menu, or type the following at the GPU UNIX command line and press Enter or Return:

gpu{userID}:**pine**

The first Pine screen you see is referred to as the Main Menu. This lists the various things that you can do, such as compose and send a message, read a message, or quit the program. There are help lines at the bottom of the screen indicating the keys to press to begin these functions. The cursor is at the top of the screen waiting for you to press a key corresponding to one of the menu items.

6.16.3 Quitting Pine

To quit the Pine program, type the letter q when you are in the Main Menu. Respond with y (yes) when you are asked if you really wish to quit.

6.16.4 Sending a Message

- To send a message:
- Press c and then Return to compose a message. You are shown the Compose Message screen

- At the To: prompt, type in the e-mail address of the person that you wish to send your message to and press Return. For example: helpdesk@gpu.srv.ualberta.ca
- At the Cc: prompt, type the e-mail addresses of any other individuals to whom you want to send copies and press Return. If you do not wish to send copies to anyone, just press the Return key.
- At the Attachment: prompt, type the name of the text-only file you want to send with your message and press Return. If you do not wish to send a file along with your message, just press Return.
- At the Subject: prompt, type the topic of your message; press Return.
- Type the text of your message. Pine uses an editor for composing your message. The default editor for Pine on the Logon Server is Pico, an easy-to-use editor with a command menu on the bottom.

  Type your message using the Pico editor. If you make a mistake, use the backspace or delete key to erase it, then retype the text. You can use the cursor keys to move around in the text as well.

  Other key combinations for Pico are given at the bottom of the screen. Note that the symbol that appears at the bottom of the screen (^) is shorthand for the key labelled Control or CTRL on your keyboard. So, for example, ^x means hold down the Control key and type x.
- Then you have finished typing your message in Pico, hold down Control and type **x** to exit.
- To send the message, type y at the Send Message? [y]: prompt.

6.16.5 A Quicker Way to Send a Message

Once you are logged into your Logon Server account, you can send a message to someone without first having to go into Pine's Main Menu as described above. To do this, type the following at the GPU UNIX command line and press Enter or Return:

pine name@address

The name@address is the e-mail address of the individual to whom you want to send a message. For example:

pine smith@gpu.srv.ualberta.ca

- Pine will place you in the Pico editor. Compose and send the message. Enclosing a File in a Pine Message To enclose a text-only file as part of your mail message:
- Compose a message as explained in steps 1 to 6 of "Sending a Message."
- Hold down the Control key and then type **r** to select the Read File options.
- At the Insert file: prompt, type the name of the file you want to enclose and press Return. The contents of the file are inserted into your message and you can add additional text as you wish.
- Press ^x (hold down the Ctrl key and press x) to send the message.

6.16.6 Hints for Writing a Message

To:

In this field, type the email addresses of your recipients. Separate the addresses with commas. When you are finished, press **<Return>**. Note that if you type in only the userid (login name) of your recipient, the Pine program assumes that the "right-hand part" of your recipient's address is the same as yours (e.g., *art.somewhere.edu)*, unless you explicitly enter a different one. Always check the addresses in both the To: and the Cc: fields for accuracy and completeness before you send a message.

Finding and Formatting Addresses. The best way to get a person's email address is to ask him or her for it. For more information on finding and formatting email addresses on local and remote computers, press **<Control>G** (Get Help) in the To: field.

Using the Address Book. In both the To: and the Cc: fields, you can enter a person's email address manually, or you can use an entry from your Pine Address Book. See "Using the Address Book"

Cc:

In this field, type the email addresses of the persons to whom you want to send copies. Separate their addresses with commas. When you are finished, or if you do not want to send any copies, press **<Return>**.

Attachment:

This is an advanced Pine feature that allows you to attach files, including word processing documents, spreadsheets, or images that exist on the same computer where you are running Pine. If you do not want to attach a file to your message, press **<Return>**. For more information, with your cursor in the Attachment: field, press **<Control>G** (Get Help).

Subject:

In this field, enter a one-line description of your message. A short, pertinent description is appreciated by the recipients, since this is what they see when they scan their email index. When finished, press **<Return>**.

Message Text:

Type your message. To move around, use the **arrow keys**. To delete characters, press **<Backspace>** or **<Delete>**. To delete a line, press **<Control>K**. To justify your text, press **<Control>J**. To check your spelling, press **<Control>T**. To see other editing commands, press **<Control>G** (Get Help).

Inserting a Plain Text File

If you want to send a "plain text" file with your message, you can insert the file in the body of your message using the **<Control>R** (Read in a File) command. Plain text files are files created by text

editors such as Pico, the editor you use when you compose a Pine message. For information about inserting files, with your cursor in the Message Text: field, press **<Control>G** (Get Help).

6.16.7 Hints for Sending a Message

Sending a Message.

After your message is composed, press **<Control>X**, and then type **y** or press **<Return>**. Your message is sent and a copy is saved to the sent-mail folder. If a message cannot be delivered, it eventually is returned to you. If you want to re-send a message, you can use the **F** (Forward) command.

Changing Your Mind.

If you change your mind after typing **<Control>X**), type **n** instead of **y** to continue to work on your message. While you are writing your message, you can press **<Control>O** (Postpone) to hold your message so you can work on it later, or you can press **<Control>C** (Cancel) to delete your message entirely. You are asked to confirm whether or not you want to cancel a message.

Reading a Message

From the Main Menu:

- To see a list of messages you've received, type i for the Mail Index.
- Use the n and p keys to move to the next or previous message in the list of messages. The highlighted message is the current message.
- Press Return to view the message currently highlighted. The message will be displayed in full.
- Once you've read the message, press i to go back to the Mail Index.
- Select another message to read by repeating steps 2 and 3, or return to the mail menu by pressing **m**.

Saving a Message in a Folder

You can save messages you receive or send within a Pine folder.

On the Index Menu, highlight the message you want to save and type s; or, while reading the message you want to save, type **s**.

Pine will invite you to save the message in a folder named after the sender of the message. For example, if you save a message from jdoe, Pine will respond with: SAVE to folder [jdoe]:

Press Return and the message is saved in the jdoe folder in the file called saved-messages (in the Mail directory of your GPU Logon Server account).

Replying to a Message

- After reading a message you have received, type **r** to reply to it.
- At the Include original message in Reply? (y/n) [n]: prompt, type y if you want the text of the original message included in your reply. Otherwise, type n.
- Type your reply. (Follow steps 6 to 8 under "Sending a Message" to compose your reply.)

- If you responded y (yes) to the Include original message? prompt, the text of the original message will be included in your reply. You will be put into the Pico editor and the original message will be denoted by a vertical row of greater-than signs (>) to its left. You can edit the original message as well as compose your reply to it.

6.17 Eudora

Eudora is a comprehensive electronic mail (email) software program for Windows and Macintosh that accesses your Internet Service Provider (ISP) or network to receive and send your email messages

In Eudora, you can:

- write messages and send them with custom stationary and signatures
- send files, both text and graphic, created in other programs and "attach" them to your email messages
- forward messages, redirect them, or reply to all recipients of a message
- set up mailboxes and folders for your mail
- build "filters" to sort messages, alert you, and even send an automatic reply
- store your favorite addresses in your personal address book, or let Eudora search for addresses for you
- if you use more than one ISP, Eudora allows you to create "multiple personalities" to send and receive mail to multiple accounts

6.17.1 Configuring Eudora

For instructions on how to configure Eudora, click on the operating system you use below.

- Windows 95/98/NT/ME/2000
- Mac

6.17.2 Changing your Password

UVA's Central Mail Service (CMS) does not support the use of Eudora's Change Password feature, so in order to change your password you'll need to either navigate to the CMS Mail Configuration webpage or secure telnet to config.mail.virginia.edu, login, and select the change password option from the menu

6.17.3 Using a Vacation Message

Eudora has a vacation message feature, but it only works if your computer is up and you are logged into Eudora. Alternatively, Central Mail Service offers a vacation message service regardless of your computer or Eudora status. To set up a vacation message navigate to the CMS Mail Configuration webpage or telnet to config.mail.virginia.edu, login, and select the vacation message option from the menu.

6.17.4 Working Offline

You can use Eudora without actually connecting to the network and it will hold any messages generated while working offline in queue in the Out mailbox, and will send them as soon as it is online and checks for mail. Other than the fact that the messages aren't actually sent and you can't check for new mail, there's no difference between working online and offline.

6.17.5 Start Using Eudora

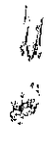
At this point you are ready to start using Eudora. If the instructions for the Mac differ from those for the PC, they are listed separately.

Continue reading the on-line tutorial, beginning with Working with Messages, to learn how to:

- Receive messages
- Read a message
- Reply to a message
- Forward a message
- Redirect a message
- Compose a message
- Print a message
- Save a message
- Delete a message

6.17.6 Filtering System

1. Create a mailbox for the listserv. (I put all of my listserv mailboxes into a listserv folder.)
2. Open a message from the listserv & copy the FROM: information (if you pretend to reply to it, the email address will appear in the TO: field of the new message, then CTRL+C, and cancel the message)
3. Go to Tools, Filters, New
4. Make sure "incoming" is checked
5. Change "Header" selection to "From"
6. Click in box after "Contains" and paste in the email address you copied
7. Under "Action," change "none" to "Transfer to:"
8. Navigate to the folder and mailbox
9. Close the window, it will ask you if you want to save changes



TIPS:

- If you can't get to a "close" box (it's a bug) you can pretend to compose a message, then save and close the Filter later.
- You can test a filter by temporarily setting it to "manual" first.
- Some listservs have different header info, so you may need to create more than one. Less is more on the info you put in the "contains" box.

6.18 Outlook

Outlook 97, Outlook 98, and Outlook Express are the Microsoft's major entries into the E-Mail market. Outlook 98 is more than an E-Mail program: it includes an address book, calendar, To Do list, and other features.

The program is free for registered users of Microsoft Office. Check Microsoft's Web site for information about the latest version. Outlook 97 came with early versions of Office 97.

If you prefer an E-Mail program that has fewer features and uses fewer computer resources, you may want to consider Outlook Express, which comes with Windows 98 and Internet Explorer 4. This section describes Outlook 98, but most of the instructions also work for Outlook Express, although often you will find that you can skip a step or two.

6.18.1 Outlook 98 and Outlook Express Highlights

Here are the basic Outlook commands (first select a message in a folder, or display it by double clicking):

- Print: Click the Print button on the toolbar, choose File >> Print, or press CTRL+P.
- Delete: Click the Delete button on the toolbar or press the DELETE key.
- Forward: Click the Forward button. You can also either choose Action >> Forward from the menu (Compose >> Forward in Outlook Express) or press CTRL+F.
- Reply: Click Reply or Reply to All button on the toolbar, or choose Action >> Reply or Action >> Reply to All (Compose >> Reply to Author and Compose >> Reply to All in Outlook Express). The shortcut keys are CTRL+R and CTRL+SHIFT+R, respectively.
- Check Spelling: Press F7. Choose Tools >> Spelling. Outlook automatically checks the spelling of each message when you click the Send button, if you choose Tools >> Options from the main menu, click the Spelling tab, and then select the Always Check Spelling Before Sending option.

6.18.2 Sending Messages

Create a new E-Mail message by clicking the Create New Message button on the toolbar. You can also press CTRL+N to create a new E-Mail message. Fill the To and Subject boxes, and write the message. Send the message by clicking the Send button, which appears to the right of the header information. You can use Microsoft Word as your E-Mail editor when you use Outlook 98 (not Outlook Express). To set

Word enables you to take advantage of Word's automatic another E-Mail program (that reads rich text E-Mail messages to see the formatting). If you have Word and you are not already using it as your editor, choose Tools >> Options from the menu, click the E-Mail tab, and then click the Use Microsoft Word as the E-Mail Editor option so that a check mark appears in the check box.

6.18.3 Filling Messages

Outlook allows you to file messages in hierarchical folders. All folders in Outlook can contain both messages and other folders. You may want to create all mail folders under the Inbox folder. To create a new mailbox or folder, follow these steps:

1. Display folder list by choosing View | Folder List from the menu.
2. Select the folder in which you want the new folder to be stored. For instance, if you want the new folder to appear indented under the Inbox, select Inbox in the folder list. If you want the new folder to be a top level folder, select the folder that appears at the top of the folder list.
3. Select File | New | Folder from the menu or press CTRL+SHIFT+E. The Create New Folder dialog box is displayed.
4. Name the folder or mailbox in the space provided. You can change the location of the folder at this point if you need to.
5. Click OK or press ENTER to create the mailbox or folder.

To move mail to a folder, either drag the message information to the folder, or right-click the message, choose Move to Folder, and then select the folder from the folder list (You can open folders in this dialog box by clicking the + next to a folder name).

6.19 Mailing Lists

Another benefit of E-Mail is that it gives you the opportunity to participate in mailing lists. Electronic mailing lists enable people to send E-Mail to large groups (it is sometimes called, "broadcasting the E-Mail"), and these lists usually function as discussion forums for people who are interested in the same topics. A group of hobbyists can exchange tips and answer each other's questions, or a college professor and her class can conduct an ongoing discussion of the assigned readings for the class. The mailing lists (or list for short) allows anyone to send mail to a single address on the list and have it forwarded to everyone else on the list. These lists function as discussion groups on any subject you can imagine, with subscribers all over the globe.

Anyone with an Internet mailing address can participate in lists of this sort. And every subscriber can be a contributor as well. Sending a message to a mailing list is called posting. Depending on how a mailing list is configured, it can be an on-line newsletter, a public forum for open discussion, a moderated discussion, or a private meeting room with a complete transcript of the proceedings.

6.19.1 Moderated and Unmoderated Mailing Lists

Some mailing lists are moderated. In a moderated mailing list, each message is first read by a volunteer or a moderator who decides if the message is appropriate. for the group. If

the message meets the guidelines for the group, it is sent to every person on the list. In an unmoderated mailing list, all messages are automatically sent to everyone in the group. Some moderators organize messages into a collection called a Digest. A digest is like an issue of an electronic magazine: a whole set of messages and articles in one easy to read package

6.19.2 Mailing Lists Administration

The most important part of administrating a mailing list is keeping track of the people on the list. When your request is to be put on a mailing list, we say that you subscribe to that list. When you ask to be taken off the list, you Unsubscribe (For subscribing mailing list service, remember, there is no charge). There are two basic ways in which mailing lists are administered:

6.19.3 Manually Maintained Lists

A manually maintained mailing list is managed by an administrator who keeps track of every member. The E-Mail address for a manually maintained list typically contains the word "request" (example: hang-gliding-request @ lists.utah.edu).

- Subscribe: To join manually maintained mailing list, you simply send an E-Mail message to the administrative address. Within a few days you will start receiving messages from the mailing list
- Unsubscribe: To leave a manually maintained mailing list, you simply send an E-Mail message to the administrative address.

6.19.4 Automated Lists

An automated mailing list is managed by a program that keeps track of every member. The E-Mail address for an automated list typically starts with the name of the program used to manage the list (example: listproc @ chaos.taylored.com). There are three major programs – listproc, listserv and majordamo. The mode of subscribing and unsubscribing is similar to that of manually maintained list

6.19.5 Mailing List Management Programs

A computer program manages almost all mailing lists. Although every list has a person who acts as an administrator, he or she does not handle the housekeeping details by hand. The program does it all. When a computer program that helps with list management and the list administrator, that program is called mailing list management program. The mailing list management program responds to all messages addressed to both the list address and the list's administrative address. Messages to the list address are distributed to the subscribers. The program processes messages to the administrative address: for example, a message with the command subscribes might add the sender to the subscriber list. Subscribers communicate with the MLM entirely by sending commands to the administrative address for the list.

Popular Mailing List Management Programs (MLMs)

There are three common mailing list management programs:

- LISTSERV: ListProc, and Majordomo. Unfortunately, each MLM uses a different set of commands
- LISTSERV: The name "LISTSERV" stands for "List Server". It is a specific program developed by Eric Thomas. LISTSERV allows a user to send electronic mail to addresses like these

- listserv@kentvm.kent.edu
- listserv@cunyvm.cuny.edu
- listserv@is.internic.net

The LISTSERV program was originally developed to run on IBM mainframe computers. When the Internet became popular, it did have Usenet. Although Internet users could participate in Bitnet discussion groups via the Bitnet/Internet gateways, many people wanted to start their own lists, and the LISTSERV software worked only on IBM mainframe computer. To fill this gap, a version of LISTSERV program were developed which run under Unix called LISTSERV.

LisiProc: Another very popular Listserv like the Corporation developed program for Research and Educational Networking (CREN), called "Listproc", which stands for "List Processor". ListProc has added a Web interface to make subscribing and signing off easier. For more detail, see its home page at http://www.cren.net/listproc.

Finally, a completely different list system was developed, called "majordomo". This program acts like an automatic butler handling all the requests from the users of the mailing list.

6.19.5 Mailing List Addresses

To subscribe to a mailing list, all you have to do is mail a simple message to the computer program that administers the list. For each mailing list there are two different addresses, so you must take care to use the right one.

- Administrative address: This is used to communicate with the mailing list program. This address uses the name of the program and the name of the computer. Also this is the addresses to which you would send a message to subscribe, unsubscribe, or perform any other administrative function.
- List address: List address is the one which you use to send messages to the list itself, or it would send messages to the list, uses the name of the list and the name of the computer.

Here is a simple example to illustrate this for example, say you are reading my book Internet and Web Page Design, and you see the following mailing list:

- Buckminister Fuller : Listserv mailing list
- List Name : geodesic
- Subscription Address : listserv@upvm.cc.buffalo.edu

Here, listserv@upvm.cc.buffalo.edu. is administrative address and geodesic@upvm.cc. buffalo.edu is list address.

- Listserv: Subscribing and unsubscribing : To understand this, here is an example. Your name is Alok, and you want to subscribe to the Listserv list named geodesic on the computer named ubvm.cc.buffalo.edu. Use the following message:

To : Listserv@ubvm.cc.buffalo.edu

Subject: Subscribe geodesic Alok

This message asks the listserv program at ubvm.cc.buffalo.edu to add the person named Alok to the geodesic mailing list. To unsubscribe from the list in the previous example, you would use:

6.20 Usenet Newsgroup

The name usenet is a contraction of "User's Network". It is a system of discussion groups in which individual articles are distributed throughout the world. To participate in Usenet, one can use a client program called a newsreader.

Usenet is a large distributed system of message (also called articles or postings), involving millions of people from all over the world. As so many messages are sent every day, they are divided into newsgroup, with each newsgroup concentrating on one topic. Jokes, recipes, mathematics, philosophy, computers, biology, science, fiction, movies, music — just about any subject you can think of has its own group. You can read the newsgroups articles, post replies to articles, or post new articles.

Like E-Mail mailing lists, you can read articles whenever you want rather than reading it online simultaneously (at the same time that people are writing them). Any one can read articles, without subscribing to the newsgroup (as we do with mailing lists).

Now one question that must arise in your mind is: What does it cost to use Usenet? The answer is Usenet is absolutely free!

You may have to pay something for Internet access, but there is no charge for Usenet. In fact if you have access to the Internet at no cost, then everything, including Usenet, is free.

Unlike the commercial services (such as America Online, and Compuserve), there is no central authority that controls Usenet. Thus, whenever the users decide that there should be a new discussion group, they form one. Two significant results can be drawn from this system: First, new groups can be created in a timely manner whenever the need arises. Second, there are a great many groups devoted to esoteric subjects. That is, no matter what your interests, there will be Usenet discussion groups for you.

Usenet itself is often referred to as the News or Netnews. Similarly, the Usenet discussion groups are usually referred to as Newsgroups or, more simply, GROUPS. The individual contributions, within each newsgroup, are called Articles or Postings. Submission of an article to a newsgroup is called Posting of the article.

The people who use it run Usenet. A person called the News Administrator runs each Usenet site. In some places, the news administrator is the same person who manages the system. But in large organizations such as a university, the news administrator may be a staff member, or even an undergraduate volunteer, who reports to the system manager. The job of each news administrator is to manage his or her own site.

The news administrator contacts with all others. In this way, nobody can tell anybody else what to do. This lack of central authority is what gives Usenet its charm, and is what distinguishes it from other discussion group systems where (such as CompuServe) there are Rules and People-In-Charge.

This does not mean that Usenet is total anarchy. There are certain conventions and from a responsible Usenet member, it is expected to learn and follow these conventions. Still, when people misbehave, more right-thinking people usually have to confine themselves to public criticism or to sending letters of complaint to the person's electronic mailbox.

6.20.1 Relationship Between Netnews and E-Mail

The relationship between network news and electronic mail is more than accidental. Developers of network news decided to use electronic mail messages for news articles because it provided an easy way to integrate the two services. A person can use electronic mail to send an article to a newsgroup. When a user saves a copy of an article from a newsgroup, the saved copy has the same format as an electronic mail message. Thus, one can use the same software to manipulate either a copy of a news article or a copy of an electronic mail message.

Student Activity

1. What is WWW?
2. What are web browsers?
3. How may we plan web pages in other languages?
4. What are web search engines?
5. What are commonly used e-mail packages?
6. Outline the features of Outlook?
7. What is Usenet Newsgroup meant for?
8. What is web chatting?
9. What for Gopher and Veronica are used?
10. What are mailing lists?

Unit 7

Web Authoring and HTML Techniques

Learning Objectives

After reading this unit you should appreciate the following:

7.1 Creating a Web Page

To start making a web page, user has to go through the following procedure:

- Planning: an act of formulating a program for a definite course of action; "the planning was more fun than the trip itself" 2: the act or process of drawing up plans or layouts for some project or enterprise 3: the cognitive process of thinking about what you will do in the event of something happening "his planning for retirement was hindered by several uncertainties

- Organizing: To put together into an orderly, functional, structured whole. The four basic steps in organizing your page is to divide it into logical units, establish a hierarchy of importance and generality, use the hierarchy to structure relationships among chunks, then analyze the functional and aesthetic success of your system.
- Creating: Steps for creating:
  - Gather Information
  - Determine the Intended Audience
  - Create a Storyboard
  - Plan Your Navigational Tools
  - Create an Aesthetically Appealing Webpage
- Testing: To create an intuitive library information gateway that will help user find, and access appropriate resources the web interface should be clear, easy to manure and provide built in redundancy to accommodate different testing strategies. By testing we can evaluate the effectiveness of online library instructions and can understand how to create a more useable interface.

The components required for Web pages are described below:

- Text: text: A text consisting of an HTML file and any related files for scripts and graphics, and often hyperlinked to other documents on the Web.
- Pictures: Pictures can be still images or video images which we can put up in a web page. A picture is a visual representation or image painted, drawn, photographed, or otherwise rendered on a flat surface
- Animated Graphics: The term "animated graphics" means text or images that do not remain static but that may move when viewed in a browser. Animated graphics: A simulation of movement created by displaying a series of pictures, or frames. Cartoons on television is one example of animation
- Audio files: A sound that has been digitized and stored on a computer. All performance characteristics and sounds or voices are included in the file; and sounds will sound the same on any computer, except for differences in speaker quality. Audio files are very large, and larger files equal better sound quality. One minute of CD quality stereo takes about 10 MB.

We need the following tools for creating a web page

- A text editor, or a Web page editor that adds the HTML codes for you. A drawing program, if you want to create your own graphics.
- A supply of clip art, if you do not want to create your own graphics.
- Sound video equipment, if you plan to make audio or video files to include on your site.

7.2 Document Organization Types

There are three types of organizations at your disposal:

- Hierarchical
- Linear
- Webbed

7.2.1 Hierarchical Organization

Using the hierarchical organization is much like the technique used when creating an organization chart for a company. The hierarchy starts with top officials, then shows the managers who work for them, the employees who work for those managers, and so on. With a Web site, it's the same way. Organizing information in a hierarchical structure, you present a first group of equally important topics, followed by another group of equally important topics, and so on.

If you choose hierarchical organization, remember to keep it simple. When you organize information in a hierarchical structure, you present a first group of equally important topics, followed by another group of equally important topics, and so on. The hierarchy starts with top officials, then shows the managers who work for them, then the employees who work for those managers, and so on.

The same will be true for a Web site. You can provide several main points, and under each of them you can include sub-points. For example, the UPTEC Web site uses hierarchical organization in its main page according to the major topics, as shown in Figure 7.1.

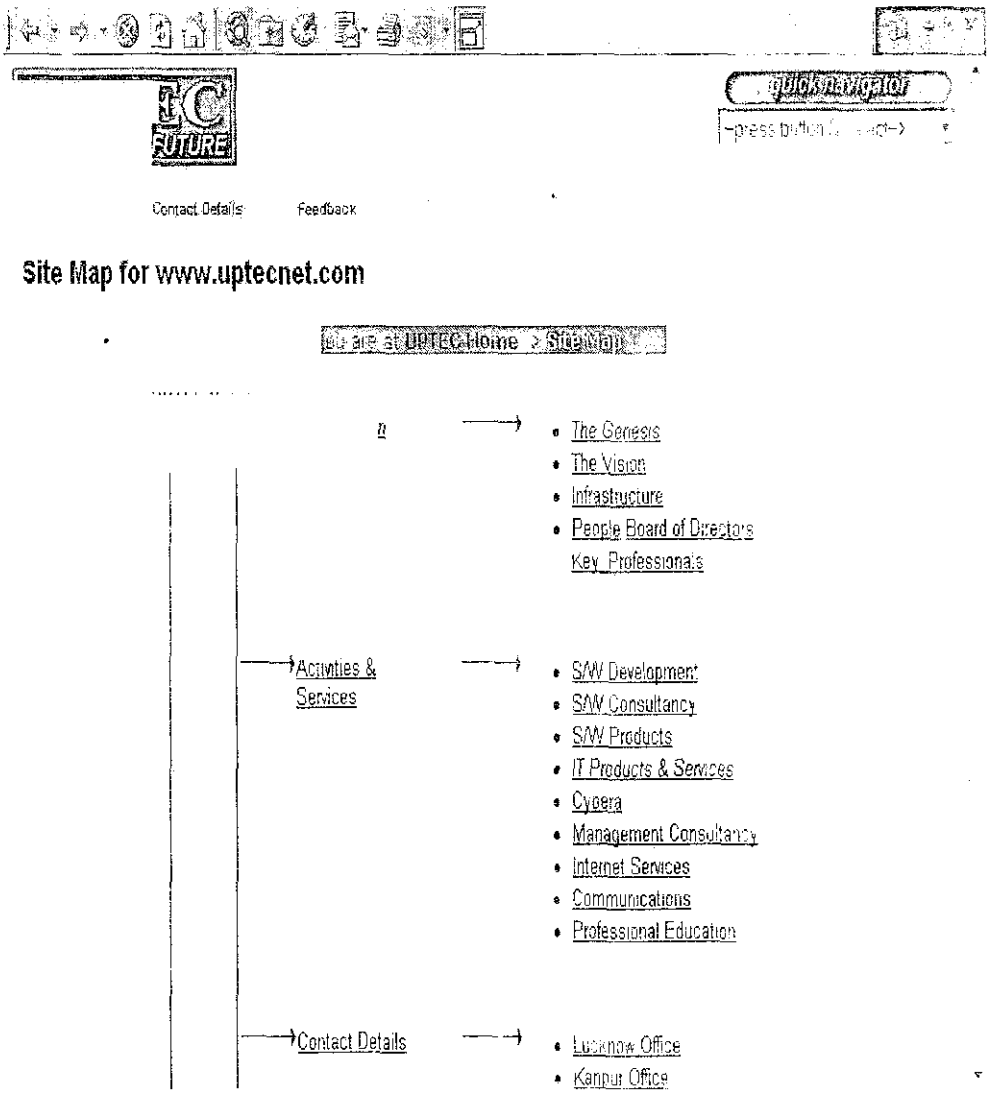


Figure 7.1: Hierarchical Organization

7.2.2 Linear Organization

A Web site that uses linear organization allows a visitor to move forward and backward within a particular set of pages but cannot jump to other pages. If you have surfed the Web a good amount I'm sure you've come across a page that is set up using linear organization, commonly noticed by the "next, back, and home" buttons. Because this can occasionally frustrate a visitor who wants to get to other pages, you need to use linear organization only when it's necessary. When using linear organization try to follow these two guidelines:

- Remember that the visitor cannot roam to other pages, therefore, be sure the linear process is essential to the task at hand.
- Keep the linear sequence as short as possible so that visitors focus on the process and complete it successfully.

In a linear organization, you organize information in a particular order. Instructions and procedures are examples of this type of organization. Microsoft Wizard is another example of a linear organization, in which you start the Wizard, and then proceed in a linear fashion from one screen to the next until you click Finish. You can go back up to a step or two if necessary, but if you do not complete all the steps, you terminate the procedure.

At a Web site that uses linear organization, a visitor can move forward and backward within a sequence of pages but cannot jump to other pages. Because this can occasionally frustrate a visitor who wants to get to other pages, you need to use linear organization only when it's necessary.

7.2.3 Webbed Organization

This organization is a fairly new type of organization that has evolved with on-line technology and provides visitors with multiple, unorganized paths to resources at a site. A visitor can link from one Web page to many other pages at the same Web site or at another Web site. You often hear stories about Web surfers becoming disoriented or lost — they do not know where they are or where they have been. Webbed organization is the culprit. When using webbed organization try to follow these two guidelines:

- Provide information on each page that allows visitors to orient themselves. For example, include a running footer or logo on each page.
- Provide a link to your home page on all pages. If you do so, visitors can easily return to a familiar page.

7.3 Creating HTML Documents

After you have adequately planned your HTML documents, deciding which information to include and how to organize it, you are ready to start creating HTML documents. The Web page formatting commands are called HTML page, the commands that are included in the formatting language that Web browsers understand when displaying Web pages. Web page files usually have the filename extension html or .htm.

Here are the general steps you follow to create and maintain a Web site:

1. Plan the structure of the site, so that you have an idea as to what information will be on the home page and other key pages. Be sure that you have thought about the audience for the site, your main purpose is, and how often you plan to update the site.

2. Using a text editor or Web page editor, create the pages for your site (or some of them) and save them as HTML files. Use a graphics editor to create or view graphics for the pages.

3. Using your own browser, view the HTML files that you created (In Netscape Navigator or Internet Explorer, press CTRL+O to open a page that is stored on your own computer). Check that the text is spelled correctly, the graphics look good, and the links among your pages work. Repeat Steps 2 and 3 until your site looks good enough to publish.

4. Publish your Web site by putting all of its files (HTML files and graphics files) on a Web server (described in the next section).

5. Using your browser, view the Web pages as stored on the Web server. If you expect a large audience for your Web site, view the page by using the two most recent versions of the most popular Web browsers (for example, Netscape Navigator 3 and 4.5, and Internet Explorer 3 and 4), because different browsers format pages slightly differently. Also, view the pages from a computer other than the one on which you created the pages, so that you can spot accidental references to files on your own hard disk.

Publicize your site (as described in the section "Publicizing Your Site". later in this chapter), get feedback, get new ideas, and repeat the steps.

7.4 Linking Web Pages

The most important capability of html is its ability to create hyperlink to documents elsewhere on the server and on different server and thereby make possible a worldwide network of linked documents and information. In html both text and images can act as anchors to link to other pages on the web.

 yahoo:<a href= http://www.yahoo.com>

http://www.yahoo.com

links are inserted using the a(anchor) element. The anchor element is unlike the elements that we've seen, certain attributes inside it's opening tag in order to activate the hyperlink. the most important attribute is a location to which you would like the anchoring object to e link.

This location can be any accessible page, file or URL. To specify the address you would like to link to, insert the here attribute into the attribute tags follows:

<a href=address> In the above case the address we are linking ton is http://www.yahoo.com

Anchors can also link to email address when someone clicks on this type of anchored link the default email program initiates an email message to the linked address.

Email links use syntax almost identical to that for links to other web pages.

<p>my email address is ......

it is important that this whole attribute, including the mail to :,be placed in quotation marks

7.5 Publishing HTML Documents on Web

Publishing means putting HTML documents on a Web server. The exact process you will use for publishing documents depends on your situation. Some large organizations have well-defined publishing procedures; in such cases, you might simply fill out HTML forms and save your files in a specified folder. If your organization does not have procedures or if you are publishing on the Internet, you will need to upload your files, i.e., to copy files from your computer to a server.

Deciding where to store your pages is one of the final steps in creating your Web site plan. To make your site available to everyone on the Web, you need to publish the site on a Web server. You can either set up your own server or post your files to some one else's Web server.

Web pages are coded using Hyper Text Markup Language (HTML) tags to specify the way text and graphics are displayed. When you create your own Web pages, you can either type the tags manually in a text editor or use a Web page editor to do the coding for you. These editors are also called Web publishing tools. There are number of Web page editors available for making Web pages. Web page editors are available as stand-alone applications or bundled into the Internet software package. Some popular editors are Netscape Composer (which is part of the Netscape Communicator suite), FrontPage and FrontPage Express (from Microsoft), PageMill (from Adobe), and HotDog Professional (from Sausage Software). All run under Windows 98 and some (including Netscape Composer) are also available for the Mac. This section describes FrontPage, in some detail, as an example of how a Web page editor works. The other editors work in a similar way.

7.6 Publishing Website

After your site is published on the Web and you have joined the global on-line community, how are you going to publicize the site? The first step is to register your site with some search engines, so that people doing on-line searches can find the site. Apart from advertising your site by the word of mouth, here are some other ways to let other people know that your site is on the web. Use the method that is appropriate for your site:

- Get the URL of your site printed on your business cards, stationary, and in yellow pages.
- Add your URL to your signature block in e-mail and newsgroup messages.
- Include the URL in your return address whenever you send greeting cards or holiday cards.
- Have some bumper stickers printed showing your group or club's URL.
- Print the URL in your church bulletin each week.
- Post the URL on all school bulletin boards.

7.7 HTML

HTML stands for Hypertext Markup Language. HTML represents a way to take ordinary [illegible] into hypertext by just adding special elements – called markup tags – that tell Web browsers display a Web page's contents.

HTML is one member of a family of markup languages called SGML. SGML was developed by the International Organization for Standards in 1986 to define markup languages designed for [illegible] different purposes. Every language in the SGML family conforms to certain requirements, the best of which being that all of the symbols are strictly defined using a DTD, or Document Type Definition. The DTD for HTML defines what tags are available and how they can be used. HTML shares [illegible] characteristics with its SGML parent:

- A character based method for describing and expressing content.
- A desire to deliver that content equally to multiple platforms.
- A method for linking document components together to compose compounds documents.

HTML The first version of HTML was just called HTML. The basic tags have not changed [illegible] HTML has tried to stay backward – compatible so that new versions of HTML do not [illegible] versions.

HTML Dave Raggett worked on a successor to HTML called HTML + in 1993. Although [illegible] an official specification, many of its ideas were incorporated into HTML 2.0.

Table 7.1

HTML 2.0	A specification for HTML 2.0 was created in July 1994, and after editing by Dan Cannolly, HTML 2.0 became a standard approved by IETF in November 1994. HTML 2.0 was the first popular version of HTML and the massive explosion of Web popularity took place from late 1994 to 1995 using Web pages written in HTML 2.0. HTML 2.0 was criticized for not containing more formatting commands.
HTML 3.2	HTML 3.2, originally code-named "Wilbur", became finalized in January 1997. HTML 3.2 was immediately popular, mostly because it supported existing practices in a logical way and was more compatible with HTML 2.0. It added support for tables. IE3 and Netscape supported HTML 3.2 almost completely.
HTML 4.0	Code-named "Cougar", HTML 4.0 is the latest version of HTML. The largest difference between HTML 4.0 and previous versions of HTML is the character set. HTML 4.0 uses a character set called "Unicode" that allows thousands of different characters. HTML 4.0's specification strongly encourages the use of style sheets and discourages tags and attributes that are solely for formatting, visual appearance, and layout. HTML 4.0 also introduces the <OBJECT> tag, which is used to present multimedia.

7.8 Structure of HTML Documents

HTML documents are essentially plain text files. They contains no images no sounds, no video, and no animations; however, they can include "pointers", or links, to these other file types, which is how Web pages end up looking as if they contain non-text elements.

HTML itself is a system of codes made up of tags and attributes that serve to identify parts and characteristics of HTML documents. Some tags provide document structure; others reference other files. Attributes provide additional information within tags. HTML tags identify logical document parts — that is, the major structural components are part of the HTML documents such as headings, lists, and paragraphs. For example, if you want to include a heading, a paragraph, and a list in your document, you type the text and apply the appropriate tags to the text.

HTML is made up of tags and attributes, which work together to identify document parts and tell browsers how to display them. Tags identify document parts by specifying, for example, that a chunk of information be displayed as a heading. Attributes are optional parts of tags and modify or specify information in tags, such as colors, alignment, height, or width.

7.8.1 Adding Tags

Tags are easy to read and use once you become familiar with their components. First, all tags are composed of elements that are contained within angle brackets (< >). The angle brackets simply tell browsers that the text between them is a HTML command. A tag with its angle brackets looks like this:

<TAG>

Second, most tags are paired, with an opening tag (<TAG>) and a closing tag (</TAG>). Both tags look alike, except the closing tag also includes a forward slash (/). To apply tags to information in your document, place the opening tag before the information, and place the closing tag after the information, like this:

<TAG> information that the tag apply to </TAG>)

Including Attributes

Attributes provide extra information about a tag. For example, if you apply a heading tag (<H1>) to a line of text, like this:

<H1> A heading goes here </H1>

You can further specify that using an attribute, like this, should center it:

<H1 Align = "Centre"> A centered heading goes here </H2>

As a general rule, most attributes — those that include only letters, digits, hyphens, or periods — work fine without quotes. For example, you can type ALIGN = Centre or Align = "Centre"; all browsers should display these in the same way.

Attributes that have other characters, such as spaces, or # signs, however, do require, quotes. For example, if you use the Width = attribute to indicate percentage of the document window, type Width = "75%".

7.8.2 Applying Structure Tags

Structure tags provide browsers with information about document characteristics, such as the version of HTML used, introductory information about the document, and the title. Most structure tags, although part of the HTML document, do not appear in the browser window. Instead, structure tags work "behind the scenes" and essentially tell the browser which elements to include and how to display them

All HTML documents should include five structure tags, nested and ordered.

7.8.3 Elements of HTML Syntax

A tag takes a generic form that looks something like this:

<Tag Name {Attribute {="Value"} ...} > Text {</Tag Name>}

(Text within tags is case insensitive)

Table 7.2

<TAG NAME> All HTML tags have names. For example, H1 is a level-one header; OL is an ordered list. Tags are surrounded by angle brackets to mark their contents for special attention from the parser.	{ATTRIBUTE {="VALUE}...} Some HTML tags require or permit named attributes to be associated with them. Some attributes require that value be associated with them.
For example, in an <IMG> tag (used to point to a graphics file), a required attribute is SRC (Source), to provide a pointer to the file where the graphic resides, as in IMG SRC = "../gifs/redball.gif."> (...) indicates that some HTML tags may include multiple attributes, each of which may or may not take a value.	Text This is content that is modified by a tag. For example, if the tag were a document title, the HTML string <TITLE> UPTEC Computer Consultancy I td </TITLE> would display the text in the title bar at the top of a graphical browser's window.
{</TAG NAME>} A closing tag name is denoted by a left angle bracket (<), followed by a forward slash (/), then the tag name, and finally a closing right angle bracket (>); curly braces indicate that this element does not always occur.	

The ampersand (&) is another special HTML control character. It is used to denote a special character for HTML content that may not belong to a 7-bit ASCII character set. Such tagged items are called character entities. Some HTML elements have no content. For example, the horizontal rule element (which uses the <HR> start tag) has no content; its only role is to create a line. Elements with no content are called "empty elements", and they never have end tags.

7.9 HTML Example

To create this HTML example, just start your text editor and type in the following HTML code and save it as S11.htm.

<HTML>

<HEAD>

<TITLE> A Hello World Example in HTML </TITLE>

</HEAD>

<BODY>

Hello, World !

</BODY>

</HTML

Netscape Navigator will display this code as a simple web page.

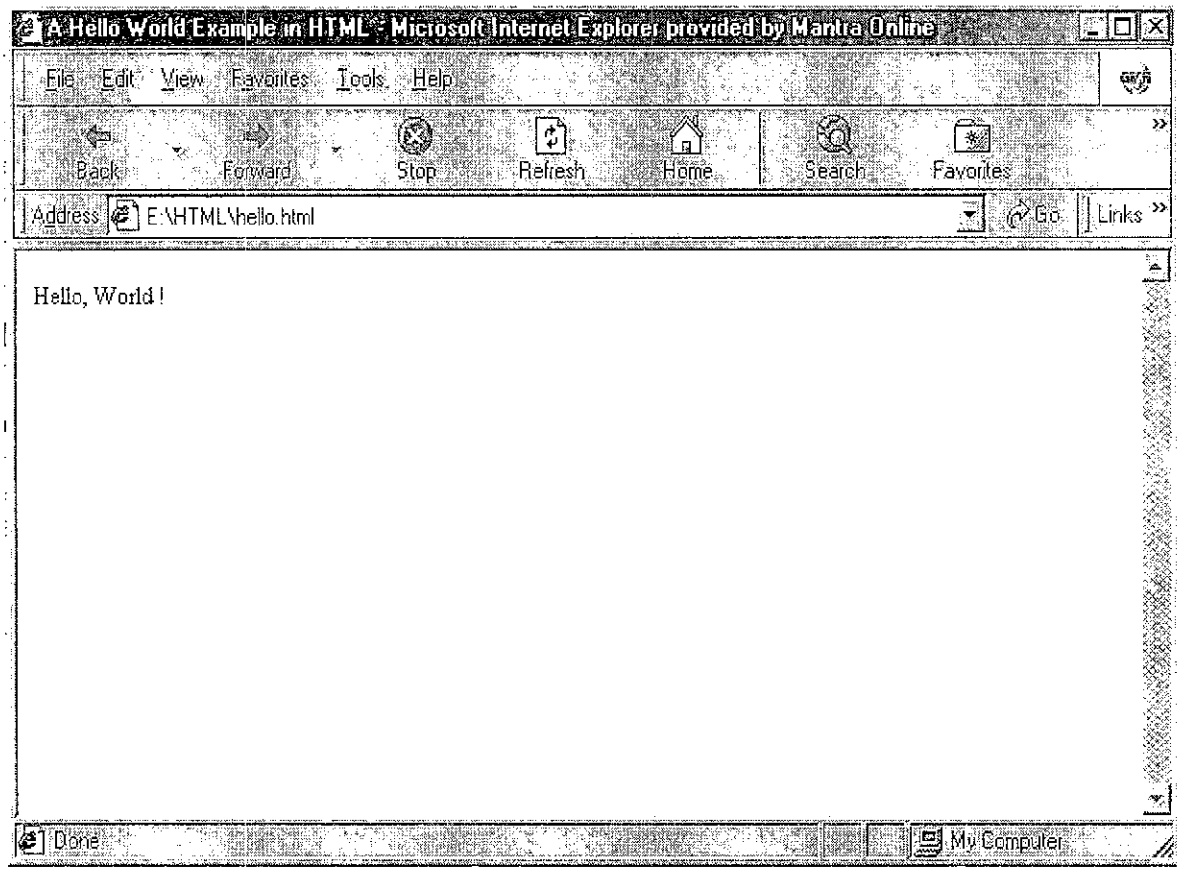


Figure 7.2

7.10 HTML Layout Techniques

HTML was designed to allow the end user, via their browser, to control the presentation of Web pages. The following are some important techniques for deciding HTML layout.

- Formatting Text
- Images
- Links
- Lists
- Tables
- Forms
- Frames

All of these techniques are discussed in this unit later.

7.11 Basic Structure of HTML Document

Well-structured HTML documents come in these three parts:

1. A head that identifies a document as HTML and establishes its title.
2. A body that contains the content for a Web page. This part holds all displayed text on a page as well as most links to graphics, multimedia, locations inside the same file, and to other Web documents.
3. A footer that labels a page by identifying its author, date of creation, and version number.

Describing Documents with the Head Element

The head element is used to mark the position of the head section. The head section contains elements that define certain information about an HTML document, such as what its title is, who the author is, and reference information about the document. To create a head element, start with <HEAD> tag, then include all of the elements you want in your head section, then end the head element with a </HEAD> tag.

Naming Documents with the Title Element

Titles are displayed by browsers on the top of the page, usually in the title bar. Every HTML document must have a title contained in a <TITLE> start tag and a </TITLE> end tag. For example,

<HEAD>

<TITLE> A Hello World Example in HTML </TITLE>

</HEAD>

Wrapping Your Content with the Body Element

The real content for any HTML document occurs in the body section, which is enclosed between <BODY> and </BODY> tags.

Two Categories of Body Elements

There are two basic categories of HTML elements used in the body section:

- Block-Level Elements
- Text-Level Elements

Block-Level Elements

Block-level elements are used to define groups of text for a specific role. They include tags that position text on the page, begin new paragraphs, set heading levels and create lists. Some commonly used block-level elements and their tags are:

Paragraph: <P> and </P>

Heading, level one: <H1> and </H1>

Heading, level two: <H2> and </H2>

Horizontal rule: <HR>

Centreing: <CENTER>

Text-Level Elements

Text-level elements are for mark up bits of text, including creating links, inserting things like images or sounds, and changing the appearance of text. Some commonly used text-level elements are:

Bold: <B> and </B>

Italic: <I> and </I>

Line-break:

Link anchor: <A HREF = "URL"> and </A>

Image: <IMG SRC = "URL">

7.12 Footer

Technically speaking, HTML does not include a separate tag to denote a page footer. But it is recommended because a good footer helps to identify a document's vintage and contents and let interested readers contact the author if they spot errors or want to provide feedback.

e.g.: Type the following HTML code, and save it as S12.htm in your web folder.

<HTML>

```
<HEAD>
<TITLE> Rupert's Fabulous T-shirt Company </TITLE>
</HEAD>
<BODY>
<HI> Welcome to Rupert's Fabulous T-shirts! </H1>
<H2> Fabulous T-shirts Since 1752 </H2>
Our Company, <B> Rupert's Fabulous T-shirt Company </B>, is your <B> <I> second-best choice </I>
</B> for T-shirts. (The best choice is <A HREF = "http://WWW. inkyfingers.com/"> Inky Fingers
</A>.)
Write us at: <BR>
555 Garment Way <BR>
Alameda, (A 94412
<P>
(all us at (510) 555-9912. <I> <B> We're here to help: </B> </I>
<IMG SRC = "http://www.emf.net/~estephen/images/turle shirts.jpg")
<HR>
<CENTER> Why not visit <A
HREF = "http: //www.yahoo.com/"> Yahoo</A>?
</CENTER>
</BODY>
</HTML>
```

7.13 Text Formatting & Alignment

Text Tags

Text tags provide a logical structure for content. Using block-level elements to structure the document. Block-level elements contain blocks of text and can organize text into paragraphs. Block-level elements, according to the W3C standard for HTML 4.0, should have a line-break or paragraph break before and after the element. Some common block-level elements are:

> headings (<H1> and </H1>, and <H2> and </H2>), paragraphs (<P> and </P>), horizontal rules (<HR>), and centered text (<CENTER> and </CENTER>).

Using text-level elements

- Text-level elements mark up bits of text, in order to change the appearance or function of that text. Text-level elements do not start new paragraphs; instead, text-level elements are usually used within a paragraph. So, text-level elements:
    - can define character appearance and function.

- must be nested in the proper order.
- do not generally cause paragraph breaks.
- can contain other text-level elements but not block-level elements.

Tag: <ABBR> ... </ABBR> or <ACRONYM> ... </ACRONYM>

Tag Name: Abbreviation

Explanation

It is a text-level element that indicates that the enclosed text is an acronym or abbreviation.

Attributes: TITLE = "text"

Provides the expanded version of the abbreviation and appears in a pop-up box when the mouse is over the acronym.

e.g.: I spy for the <ACRONYM TITLE = "Federal Bureau of Investigation"> FBI </ACRONYM>.

Tag: <BLOCKQUOTE> ... </BLOCKQUOTE>

Tag Name: Block quote

Explanation

This tag or block-level element is used for quoting one or more paragraphs from another source. Navigator and IE indent the entire block of quoted text.

Attributes: CITE = "text"

Provides information about the source of the quote.

e.g.: <P> From the Bridges of the New York City, Queensbury Ballads by Levi Asher (http: //www.levity.com/brooklyn/index.htm): <BLOCKQUOTE>

It is not just that everybody hates the city, the more time I spend with these people the more I understand that they hate everything. Or at least they seem to, because it is the culture of Wall Street to never show joy.

</BLOCKQUOTE>

Tag:

Tag Name: Line Break

Explanation

It is a text-level element (an empty element, consisting of the
 tag) which forces a line break in HTML text.

Attributes: CLEAR = (LEFT ¦ ALL ¦ RIGHT ¦ NONE)

LEFT inserts space that aligns the following text with the left margin directly below a left-aligned floating image.

ALL places the following text past all floating images.

RIGHT inserts space that aligns the following text with the right margin directly below a right-aligned floating image.

NONE is the default, which does nothing.

e.g.: Hello

World

Line-breaks are useful for addresses and short items.

Tag: <CITE> ... </CITE>

Tag Name: Short Citation

Explanation

It is a text-level element used to indicate that enclosed text is a citation (titles of excerpt , references) from another source.

Text within <CITE> and </CITE> is usually rendered in italics.

e.g.: <P> I have read and reread <CITE> Moby Dick

</CITE> but I still cannot make heads nor tails of it. </P>

Tag: <CODE> ... </CODE>

Tag Name: Code

Explanation

It is a text-level element used to enclose programs or samples of code to offset them from normal text and is usually displayed in a mono spaced font.

e.g.: <P> To use the automatic date feature in Excel, just enter <CODE> = Date ( )</CODE> into a cell </P>.

Tag: <DEL> ... </DEL>

Tag Name: Delete Section

Explanation

It marks text that has been deleted from a previous version of the Web document.

Attributes: CITE = "URL"

Points to another document that describes why the text was deleted.

DATETIME = YYYY-MM-DDThh: mm: ssTZD

Marks the time when you changed the document. The attribute's value confirms to the ISO8601 time/date specification. The abbreviations shown above refer to the following date and time information

YYYY	=	The year.
MM	=	The two-digit month – for example, "03" for March.
DD	=	The day.
T	=	The beginning of the time section.
hh	=	The hour, in military time (0-23 hours), without pm specifications.
mm	=	The minute.
ss	=	The second.
TZD	=	The time zone.
Z	=	The Coordinated Universal Time (CUT).
+hh:mm	=	The local time that is hours (hh) and minutes (mm) ahead of the CUT.
-hh:mm	=	The Local time that is hours (hh) and minutes (mm) behind the CUT.

Tag: <INS> ... </INS>

Tag Name: Inserted Section

Explanation

Identifies sections of a Web page that have been inserted in revision.

Attributes:

Same as that of the <DEL> ... </DEL> tag.

<INS CITE = "URL"> or <DEL CITE = "URL">

<INS DATETIME = "DATE & TIME"> or <DEL DATETIME = "DATE & TIME">

<DEL>... </DEL> and its companion tag <INS> ... </INS> were created to show revisions to web pages.

Both these elements are special hybrid elements. They are unique in this respect - they are the only two elements used in the body of an HTML document that are not text-level or block-level elements.

Example

<H2> Latest News </H2>

<INS DATETIME = "1998-04-22TII:38:00 07:00"

CITE = "http://www.tori.com/updatelog.htm">

<P> We've just received some new information.

<P> Apparently there will be <STRONG> two shows </STRONG> on Sunday.

</INS>

<P> The show will Start at 7:00 P.M.

This code marks the middle two paragraphs as new. They were changed on April 22, 1998 at 11:38 a.m., information about the change can be found at the URL listed in the CITE attribute.

Tag: <DFN> ... </DFN>

Tag Name: Defined Term

Explanation

It is a text-level element used to mark terms that appear for the first time in the Web document. These definitions are often in italics so the user can identify the first occurrence of the term.

e.g.: <P> It is not strange that <DFN>SGML</DFN> (Standard Generalized Markup Language) is so similar to HTML.</P>.

By marking your definitions, this way, special software programs can define an index or glossary for your document.

Tag: <EM> ... </EM>

Tag Name: Emphasis

Explanation

A text-level element that adds emphasis to enclosed text.

Example

<P> I simply <EM> must </EM> get your recipe for chili, karen Doason ! </P>

Tag: <KBD> ... </KBD>

Tag Name: Keyboard Text

Explanation

It is a text-level element that marks text to be entered by the user at the keyboard.

It changes the type style for all the text it contains, typically into a monospaced font like those used in character mode computer terminal displays.

e.g.: <P> To start the program, hit the <KBD> S </KBD> key and press the <KBD> Carriage Return </KBD>, then hold onto your hat!!</P>

Tag: <P> ... </P>

Tag Name: Paragraph

Explanation

The paragraph element's <P> tag marks the beginning of a paragraph. The end of a paragraph can be marked with </P>. It is a block-level element.

Attributes: ALIGN = (LEFT|CENTER|RIGHT|JUSTIFY)

Tag: <PRE> ... </PRE>

Tag Name: Preformatted Text

Explanation

This block-level element forces the browser to display the exact formatting, indentation, and white space that the original text contains. This is valuable in reproducing formatted tables or other text, such as code listings.

Attributes: WIDTH = number

This specifies the maximum number of characters for a line and allows the browser to select an appropriate font and indentation setting.

Example

<P> morning wind

my hair moves

with the clouds and trees

<PRE> morning wind

my hair moves

with the clouds and trees </PRE>

morning wind

my hair moves with the clouds and trees.

Tag: <Q> ... </Q>

Tag Name: Short Quotation

Explanation

This element is used to highlight short quotation from outside resources. The quote element is very similar to the block quote element; the main difference is that since the quote element is not block-level, it does not start a new paragraph. Instead, it is used within a paragraph to mark a quotation.

Attributes: CITE = "text"

Provides information about the source of the quote.

Example

<P> Churchill said, <Q> "We have chosen shame and will get war," </Q> but he was [illegible] in 1066.</P>

Tag: <SAMP> ... </SAMP>

Tag Name: Sample Text

Explanation

This text-level element uses the <SAMP> and </SAMP> tags to indicate sample output from a computer program. Like the keyboard element, the sample element's text is often rendered in a monospaced font; and multiple spaces are collapsed. The keyboard element is used for text that you must enter, whereas the sample element is used for text that a computer generates in response to your action.

e.g.: <P> Instead of giving me the expected results, the computer kept printing <SAMP> [illegible] and no play makes Jack a dull boy </SAMP> over and over again. I am not sure what it means.

Tag: <STRONG> ... </STRONG>

Tag Name: Strong Emphasis

Explanation

It is a text-level element that provides strong emphasis for key words or phrases within normal body text, lists, and other block-level elements. Text within a strong element is usually rendered in bold or given a strident pronunciation by a text-to-speech reader.

e.g.: <P> I swear, if they do not give me that raise <STRONG> tomorrow </STRONG>, [illegible] </P>

Tag: _{...}

Tag Name: Subscript

Explanation

This text-level element specifies that the enclosed text should be rendered in subscript, slightly lower than the surrounding text.

e.g.: This line of HTML Code contains the chemical formula for water:

We all need H₂O

Tag: ^{...}

Tag Name: Superscript

Explanation

This element renders the enclosed text in superscript (a bit higher than regular text).

This element is also useful for mathematical formulas.

Example

Here's Einstein's most famous equation:

E = MC ²

Another good use of the ^{TM} today

Tag: <VAR> ... </VAR>

Tag Name: Variable text

Explanation

This text-level element marks up the variables used in computer programs, or the parts of a computer command chosen by the user.

The text is usually rendered as mono spaced, and like the keyboard element, multiple spaces are collapsed.

e.g.: <P> The formula for the <VAR> distance traveled </VAR> (in miles) is <VAR> Speed </VAR> (in miles per hour) multiplied by <VAR> time </VAR> (in hours). </P>.

7.14 Font Control

Text Styles

Using Font-style Elements

Font-Style elements change the appearance of text. These font-style elements are known as physical markup. All font-style elements are a subcategory of text-level elements, and they all require both start and end tags. They can all be nested according to the normal rules of nesting text-level elements. The seven font-style elements are: bold (<B>), italic (<I>), underline (<U>), strikeout (<STRIKE> or <S>), big (<BIG>), small (<SMALL>), and teletype (<TT>).

Tag: <B> ... </B>

Tag Name: Boldface

Explanation

It indicates that the enclosed text is in bold face type. The bold element does not indicate strong emphasis when read by some text-only or text-to-speech browsers.

Tag: <I> ... </I>

Tag Name: Italic

Explanation

This element marks up text in italics (text slanted diagonally upward to the right).

It.is appropriate to use the italics element to indicate text in a foreign language.

e.g.: <I> corpse diem </I>

Remember that its effectiveness fades quickly with overuse. If you are including a citation or bibliographic resource in your document, use <CITE> instead of <I>.

Tag: <U> ... </U>

Tag Name: Underlined text

Explanation

The Underline element underlines text.

e.g.: <U> Hello, World ! </U>

Tag: <STRIKE> ... </STRIKE>

Tag Name: Strikethrough

Explanation

This element indicates that the enclosed text should have a line drawn through the middle of the text.

Strikethrough text is difficult to read onscreen, so use this tag sparingly. Use <STRIKE> to show text removed from earlier versions of a document, the old text will appear with a line through it.

e.g.: <STRIKE> I'm struck ! </STRIKE>

Tag: <BIG> ... </BIG> Tag Name: Big text

Explanation

It makes text font one size larger than the <BASEFONT> size.

Nesting <BIG> tags can produce text in a larger font than using only one <BIG> tag.

e.g.: <BIG> The Big and Tall Company </BIG>

<BIG> <BIG> The Very Big and Tall Company </BIG> </BIG>

Tag: <SMALL> ... </SMALL>

Tag Name: Small text

Explanation

This element makes text one size smaller than the base font size.

Nesting <SMALL> tags can produce text in a smaller font than using only one <SMALL> tag

e.g.: <SMALL> The Small and Short Company </SMALL>

Tag: <TT> ... </TT>

Tag Name: Teletype text

Explanation

The Teletype element renders the enclosed text in Teletype font. This means that the text will be mono-spaced to look like a typewriter font (browsers will often use Courier font by default).

e.g.: <P> All the vowels on my typewriter are broken. I keep typing in a standard phrase and it comes out like this: <TT> Th qck brwn FX jmps vr th lz dg </TT>. I think I need a typewriter repairman. <P>

7.15 Arranging Text in Lists

Lists

Lists provide methods to lay out item or element sequences in document content. There are three main types of lists:

- Unordered lists
- Ordered lists
- Definition lists

Ordered lists are numbered in some fashion, while unordered lists are bulleted. Definition lists consist of a term followed by its definition. Both ordered and unordered lists require start and end tags as well as the use of a special element to indicate where each list item begins (the /LI tag). In addition to these three types of lists, HTML also allows two other types of lists that are not very commonly used:

- Directory lists
- Menu lists

Tag: <LI>

Tag Name: List Item

Explanation

<LI> is a singleton tag.

It is a child element that is used to create a list item in an ordered list, unordered list, menu list, or dir list.

Attributes: TYPE = (DISC ¦ SQUARE ¦ CIRCLE) or (1¦a¦A¦i¦I)

When an ordered list <OL> is used, the <LI> element will be rendered with a number. One can control that number's appearance with the <TYPE> attribute.

Similarly, inside an unordered list <UL>, one can control the type of bullet displayed with <TYPE>. VALUE = number changes the count of ordered lists as they progress.

Tag: <UL> ... </UL>

Tag Name: Unordered List (Block-level element)

Explanation

It creates an unordered list with bullets preceding each list item. Unordered lists can be preceded by any one of several bullet styles: a closed circle, an open circle, or a square.

Attributes:

COMPACT: Renders the list as compactly as possible by reducing line leading and spacing.

TYPE: (DISC ¦ SQUARE ¦ CIRCLE)

The type of bullet can be suggested with the <TYPE> attribute. The CIRCLE attribute value creates a hollow bullet, the DISC type creates a solid bullet, and the SQUARE attribute value creates a solid block.

The default appearance for a list is with a disc.

You can use an optional </LI> end tag at the end of each list item.

The following example generates two separate lists:

```
<TITLE> Two Shopping Lists </TITLE>
<BODY>
<UL>
<LI> Eggs
<LI> Milk
<LI> Apples
<LI> Razor Blades
</UL>
<UL TYPE = "SQUARE">
<LI> Hammer
<LI> Screwdriver
<LI TYPE = "DISC"> Screws
<LI TYPE = "CIRCLE"> Chainsaw
</UL>
```

Some browsers do not recognize the TYPE attribute at all, and most browsers do not recognize that the TYPE attribute can be used with the <LI> tag. In fact, even IE 4 does not recognize it.

One important aspect of lists is that you can nest one list inside another to create a sub list. The default appearance of the sub lists will vary from the main list, with the first sub list using circle bullets, and the next nested list-using squares.

For example:

```
<UL>
<LI> Body
 <UL>
<LI> Head
```

<LI> Hand

<UL>

<LI> Finger

<LI> Thumb

</UL>

</UL>

<LI> Mind

<UL>

<LI> Brain

<UL>

<LI> Neuron

</UL>

</UL>

<LI> Spirit

<UL>

<LI> Soul

<UL>

<LI> Light body

</UL>

</UL>

</UL>

Tag: <OL> ... </OL>

Tag Name: Ordered List

Explanation

These tags are used to create ordered lists. Ordered lists are identical in behavior to unordered lists except they use numbers instead of bullets, and you can use an attribute to start numbering at a number other than one.

Attributes: TYPE = (1¦ a ¦ A ¦ i ¦ I)

Changes the style of the list

TYPE = "1" (Arabic numbers)

TYPE = "a" (Lowercase alphanumeric)

TYPE = "A" (Uppercase alphanumeric)

TYPE = "i" (Lowercase Roman numbers)

TYPE = "I" (Uppercase Roman numbers)

[Arabic Numbers are the default]

COMPACT: Renders the list as compactly as possible by reducing line leading and spacing.

START = "Value"

Indicates where the list numbering or lettering should begin.

In ordered lists, the <LI> tag can use the VALUE attribute to force to make a particular list item to have a certain number.

e.g.: <OL>

<LI> Milk

<LI> Bread

<LI> Turkey Bacon

<LI VALUE = "10"> Dark Chocolate

<LI> Avocados

</OL>

Tag: <DT>

Tag Name: Definition Term

Explanation

The <DT> tag is a singleton tag. It is a child element and can only be used in a definition list element. It creates a term that can be defined in a definition list.

Tag: <DD>

Tag Name: Definition

Explanation

It is a singleton tag which marks the definition/description for a term in a glossary list. It is used for glossaries or other lists in which a single term or line needs to be associated with a block of indented text.

Tag: <DL> ... </DL>

Tag Name: Definition List

Explanation

Definition lists require a start tag (<DL>) and end tag (</DL>) and two special elements: one for definition terms (the <DT> tag) and one for definition (the <DD> tag). The list is rendered without bullets.

Attributes

Compact: Renders the list as compactly as possible by reducing line leading and spacing.

e.g.: <DL>

<DT> Term A

<DD> Definition of Term A

<DT> Term B

<DD> Definition Term B

</DL>

Tag: <DIR> ... </DIR>

Tag Name: Directory List

Explanation

This block-level element marks un bulleted list of short elements, such as filenames.

e.g.: <DIR>

<LI> Item 1

<LI> Item 2

<LI> Item 3

</DIR>

Tag: <MENU> ... </MENU>

Tag Name: Menu List

Explanation

It encloses a menu list in which each element is typically a word or short phrase that fits on a single line.

This list is rendered more compactly than most other list types.

Attributes: COMPACT

Renders the list as compactly as possible by reducing line leading and spacing.

e.g.: <MENU>

<LI> Sourdough

<LI> Butter milk

<LI> Rolls

</MENU>

For both directory and menu lists, the only item that should be contained is a list item element (the LI tag).

7.16 Images in Web Pages

You can add the following attributes to the <IMG> tag to adjust the picture and control how text flows around the picture:

Height and width control the size (in pixels) at which the graphic appears on the Web page. These attributes are options; use them only if you do not like the default size and need to resize the picture. The Web browser that displays the Web page adjusts the height and width of the graph to the sizes that you specify. When you use the HEIGHT and WIDTH attributes, make sure that you keep the same proportions as the original graphic; if you do not, the picture looks like you s-t-r-e-t-c-h-e-d it either horizontally or vertically. Resizing a graphic to be larger than the original is rare. The larger the number of pixels, the bigger the picture. For example, the following <IMG> tag displays the picture in the file picture if as 30 by 50 pixels, regardless of the size of the stored picture:

<IMG SRT="picture.gif" HEIGHT="30" WIDTH="50">

Using small (but legible) graphics on Web pages is best, because they land faster than large graphics. To make a graphic small, be sure to use a graphics program to reduce the size of the file. Do not just change HEIGHT and width attributes, which do not change the file size (or speed up downloading time), only the way it is displayed by the browser.

Align controls how text flows around the graphic. Align has five possible values:

Top places one line of text even with the top of the image.

Middle places one line of text at the middle of the image.

Bottom places one line of text even with the bottom of image. Top, Middle, and Bottom are useful when you have a single line of text that you want placed next to a graphic.

Left places the graphic on the left side of the page, with your text paragraph wrapped around the left side of the graphic. LEFT and RIGHT are useful when you have paragraphs of text that you want to wrap around a graphic. For example, the following tag displays a picture on the left side of the Web page with the surrounding text wrapped around its right side:

<IMG SRC="picture.fig" ALIGN="LEFT">

HSPACE and VSPACE control the amount of white space around the image. Both values are indicated in pixels. HSPACE sets the amount of space at the left and right of the image; you use this attribute to control the distance between the text that is wrapped around your graphic and the graphic itself. VSPACE sets the amount of space above and below the graphic. The following tag inserts a picture with 25 blank pixels to either side, and 10 blank pixels above and below the picture:

<IMG SRC="picture.gif" HSPACE="25" VSPACE="10">

BORDER indicates that a border should be placed around the image, and controls the width of the border. The width is measured in pixels. The next tag inserts a picture with a border that is three pixels wide (a heavy line):

<IMG SRC="picture.gif" BORDER="3">

ALT contains the text that appears while the picture is loading, or if a user has a browser that does not display graphics, or if the user has opted not to load graphics when viewing Web pages. Always include ALT attributes in all image tags, to make your Web site accessible to vision impaired users, who use special software to read the text on the screen. For example,

<IMG SRC="picture.gif" ALT="Book cover of this week's selection">

Note that you can use as many or as few of the attributes as you need. Only the SRC attribute, which specifies the filename, is necessary. All attributes are placed in the same <IMG> tag, separated by spaces. For example, if you decided to use all the attributes listed here, your <IMG> tag might look like this:

<IMG SRC="picture.gif" JEOGJT="30" Width="50" ALIGN="LEFT" HSPACE="25" VSPACE="10" BORDER="3" ALT="Book cover of this week's selection">

When readers click the "list of book club members" link, they jump to the Book Club Members location. You can add an anchor name to jump to an existing anchor in any Web page, like this:

<A HREF=*http://www.sample.com/index.html #members">list of book club members</A>

7.17 Tables

A table is a rectangular collection of *cells*, each of which is a mini-window containing HTML. These cells are the <td> elements. Cells are collected together in rows, the <tr> elements. Rows are components of the <table> element as illustrated below in Table.

Table 7.3

```
<table>
  <tr>
    <td>
      ...
    </td>
    <td>
      ...
    </td>
        ...
  </tr>
  <tr>
    <td>
      ...
    </td>
    <td>
      ...
    </td>
        ...
  </tr>
  ...
</table>
```

The <table> Element

When a <table> element is encountered the browser breaks the current line, inserts the table beginning on a new line, then restarts the text on a new line below the table.

Attributes

align=left|center|right
: This attribute indicates whether the table is flush against the left or right margins.

bgcolor="#rrggbb"|*colour*
: This attribute specifies a different background colour from the document's background.

border=*number*
: This attribute tells the browser to draw lines around the table and the cells within it. The value of this attribute is the pixel width of the border (default value is 1).

cellpadding=*number*
: This attribute specifies the amount of space between the cell's border and its contents in pixels (default value is 1).

cellspacing=*number*
: This attribute specifies the amount of space between the cells in pixels (default value is 2).

height=*number|percentage*
: The height of the table, either in pixels or as a percentage of the browser window height

hspace=*number*
: The number of pixels to the left and right of an aligned table.

vspace=*number*
: The number of pixels above and below an aligned table.

width=*number|percentage*
: This attribute specifies the width of the table, either in pixels or as a percentage of the screen width.

The <caption> Element

The <caption> element is used to provide a brief summary of the table's content or purpose. It must appear immediately after the <table> tag and precede all other tags.

Attributes

align=top|bottom|left|center|right
: This attribute positions the caption relative to the table. Borders, captions, padding, and spacing are brought together in Figure 7.3.

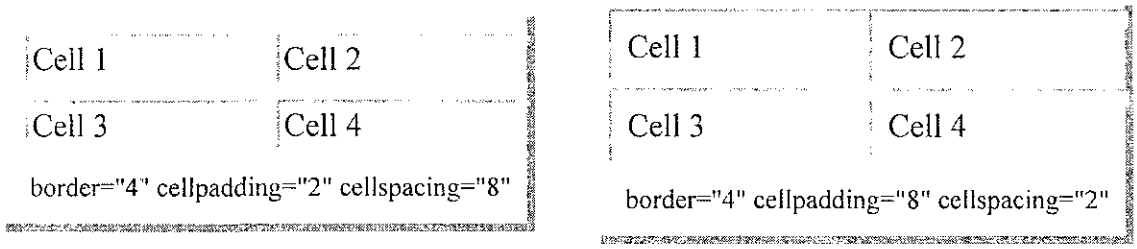


Figure 7.3: Borders, Padding, and Spacing

Note that the default border style is *outset*. This can only be changed with the use of style sheets.

The <tr> Element

Every row in a table is contained in a <tr> element. The end tag can be omitted. Every row in a table has the same number of cells as the longest row.

Attributes

align=left|center|right
This attribute changes the default flush-left horizontal alignment of all the cells in a row.

bgcolor=*"#rrggbb"|colour*
This attribute sets the background colour of the entire row.

nowrap
This attribute stops normal word wrapping in all the cells of the row. Tags such as
 and <p> are honoured.

valign=top|baseline|middle|bottom
This attribute changes the default central vertical alignment of all the cells.

The <td> Element

The <td> element creates a cell within the row. The end tag can be omitted.

Attributes

align=left|center|right
This attribute controls the horizontal alignment of the cell contents as illustrated below.

Alignment
Left
Center
Right

bgcolor="#rrggbb"|*colour*

This attribute sets the background colour of the cell.

colspan=*number*

This attribute allows a cell to span several columns (see Figure 4 below).

height=*number*|*percentage*

This attribute specifies the height of the cell, either in pixels or as a percentage of the table height.

nowrap

This attribute stops normal word wrapping in the cell. Tags such as
 and elements such as <p> are honoured.

rowspan=*number*

This attribute allows a cell to span several rows as illustrated below.

Cell 1	Cell 2 and Cell 3	
Cell 4	Cell 5	Cell 6
Cell 7	Cell 8	Cell 9

valign=top|baseline|middle|bottom

This attribute controls the vertical alignment of the cell, as illustrated below.

Alignment	**Top**	**Baseline**	**Middle**	**Bottom**
Baseline...	ABCxyz	ABCxyz		
Another line			ABCxyz	
				ABCxyz

Figure 7.4: Vertical Alignment

width=*number*

This attribute specifies the width of the cell, either in pixels or as a percentage of the table width.

The <th> Element

The <th> element is the same as the <td> element, but is used for headings, with a default central horizontal alignment and text in bold. See above for an example. This element uses the same attributes as the <td> element, see above.

Tables vs Style Sheets

The HTML 4.0 Specification favours Cascading Style Sheets instead of tables for page layout. Until style sheets are universally and consistently supported, tables are the most reliable tool for constructing complex page layouts.

7.18 Background Images & Colors

7.18.1 Displaying Wallpaper in the Background

The background of your page can be either an image or a specific colors. To set an image as the background of your page, use the BACKGROUND attribute and place the name of your image file within quotes:

<Body Background="image.gif">

Like tags, attribute names can appear in upper or lowercase. For the filename, be sure to use the same capitalization that is used in the actual filename. When a browser displays the page, the image in the file that you specify is tiled to fill the background of the Web page, that is, it is repeated across and down the page.

7.18.2 Choosing a Background Color

Instead of using wallpaper, you can specify a solid background colors for your Web page. Use the BGCOLOR attribute in the <BODY> tag. For the colors, you can enter either a hexadecimal value that represents the colors, or one of 16 standard colors names. The colors names are the following: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow. To indicate a standard colors, use this tag:

<BODY BGCOLOR="BLUE">

Using a hexadecimal value is safer, if you want your colures to be read by a wide variety of browsers. The hexadecimal value is six characters that represent the amount of red, green, and blue in the colors. The first two characters indicate the amount of red, the next two characters the amount of green, and the final two characters the amount of blue. For example, this code specifies light aqua:

<BODY BGCOLOR="#99FFFF">

7.19 Forms

This section describes the important elements of forms.

The <form> Element

Forms are supported by most browsers and allow us to interact with servers. The <form> element encloses a form containing a number of elements that the user completes before submitting the form to the server.

Attributes

action=*url*

This attribute defines the URL of the application that is going to process the form.

method=GET|POST

This attribute defines how the browser sends the form's data to the server. For example, the form associated with this document is given below.

<form action="/cgi-bin/form-demo.cgi" method="POST">.

7.19.1 Form Elements

There are a variety of elements for gathering information from a form. Every form element, except <submit> and <reset>, must be given a name so that the form-processing application can sort the information. The <input> element defines a form element and takes a number of attributes.

Attributes

maxlength=*number*
The maximum number of characters accepted for this element.

name=*text*
The name of this input element.

size=*number*
The number of characters displayed for this element.

type=checkbox|hidden|password|radio|reset|submit|text
The type of this input element.

value=*text*
The default value of this element.

Text Fields

The type=text attribute allow us to type arbitrary characters into a field as illustrated below.

Figure 7.5: What is Your Name?

The type=password attribute is like a text element, except that the typed characters are not displayed as Illustrated below.

Figure 7.6: What is Your Password?

Checkboxes

The type=checkbox attribute allows us to select one or more items as illustrated below.

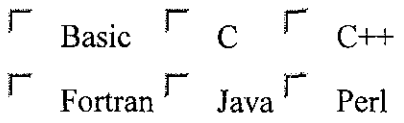


Figure 7.7: What Programming Languages do You Know?

Radio Buttons

The type=radio attribute allows us to select only one item, as illustrated below.

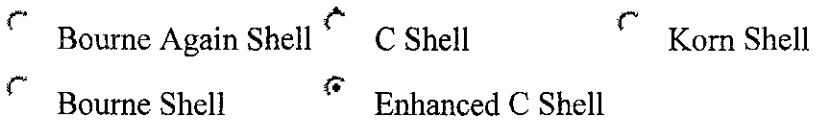


Figure 7.8: What Shell do You Use?

Action Elements

The type=submit attribute submits the form to the server. The type=reset attribute resets all the elements of the form to their default values. These two elements are often seen together as illustrated below.

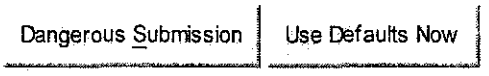


Figure 7.9: Submit or Reset?

Hidden Elements

The type=hidden attribute enables us to embed information in the form that is ignored by the browser. This is useful for passing information to CGI scripts.

7.19.2 Text Areas

The <textarea> element encloses a multi-line area for user input. The attributes rows and cols define the size of the text area, as illustrated below.

Figure 7.10: How Would You Describe Yourself?

7.19.3 Multiple Choice Elements

The <select> element encloses a number of options for selection. These are more compact than using checkbox or radio buttons.

The <option> element defines an item within the <select> element, as illustrated below.

Figure 7.11: Where Are You in the UNIX Hierarchy?

By default, the first <option> in the list displays when the form is loaded, but you can pre-select another by adding the selected attribute to the appropriate <option>.

The multiple attribute of the <select> element allows us to select multiple items, as illustrated below. Use the Shift key to select a range of items, or the Ctrl key to select individual items.

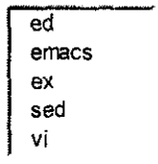


Figure 7.12: Which UNIX Editors do You Use Frequently?

The size attribute of the <select> element allows us to generate a scrolling list, as illustrated below

Figure 7.13: Which UNIX Tools do You Use Frequently?

7.19.4 Forms Processing

Our form is processed by an application running on the server when we select the submit button in Figure.

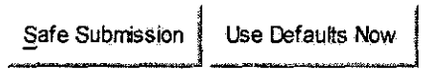


Figure 7.14: Is It Safe?

7.20 Frames

Frames allow multiple HTML documents to be presented as independent windows within one main browser window. This allows the user to present two or more documents at once.

Frameset documents have a different structure than normal HTML documents. A regular HTML document has a head element and a body element, but a frameset document has a head element and a frameset element.

Tag: <FRAMESET> ... </FRAMESET>

Tag Name: Frame Group Definition

Explanation

- This special frame structure element creates a frameset that defines how multiple documents can appear in different frames of a browser's window.
- It hosts the <FRAME>, <FRAMESET>, and <NOFRAMES> elements.
- <FRAMESET> is used instead of the <BODY> tag.

Attributes:

COLS = "col - width ¦%¦*"

Creates a frame document with columns. One can specify the column dimensions by percentage (%), pixels, or relative size (*). If you use percentages or relative sizes, then enclose the associated values in double quotation marks.

ROWS = "row - height¦%¦*"

Creates a frame document with rows. Specify the row dimensions by percentage (%), pixels, or relative size (*). The asterisk assigns the remaining unused portion of the display area. The user can:

a. Nest frames within other frames.

b. Set both ROWS and COLS simultaneously to create a grid.

c. Use nested (FRAMESET) pairs to construct complicated layouts.

Examples

i. FRAMESET COLS = "25%, 75%">

The element specifies two columns. The first column takes 25 per cent of the browsers window space. The second column takes 75 per cent of the browser's window space.

ii. <Frameset cols = "1*, 3*">

This element also specifies two columns. The first column will get one-quarter of the windows width, while the second column will get three quarters (75 per cent).

iii. <FRAMESET COLS = "100, 25%, 2*, 3*>

If you have a window that is 500 pixels wide, this element will assign 100 pixels of space to the first column. The second column will get 25 per cent of the total space, or 125 pixels. That leaves 275 pixels for the last two columns.

Two-fifths of this remaining space will go to the third column (110 pixels). The last column will receive three-fifths of the remaining space (165 pixels).

iv. <FRAMESET COLS = "250, *" ROWS = "50%, 25%, 25%">

This would create two columns and three rows, for a total of six frames. The first column is 250 pixels wide, while the second column gets the remaining space. The first row is half of the window's height, while the second and third row are each a quarter of the window's height.

Tag: <FRAME>

Tag Name: Frame Definition

Explanation

It defines a single frame in a <FRAMESET>. This is not a container, so a matching end tag is not used

Attributes:

Frame border = (1 ¦ 0)

Renders a 3-D border around the frame. The default (1) inserts a border, 0 displays no border

Margin height = (number ¦ " %")

Controls the margin height (in pixels) for the frame.

Margin Width = (number ¦ "%")

Controls the margin width (in pixels) for the frame.

Name = "text"

Provides a target name for the frame.

no resize

Prevents the user from resizing the frame.

Scrolling = (Yes ¦ no ¦ auto)

Creates a scrolling frame.

SRC = "URL"

Displays the source file for the frame.

Examples

i. The HTML code for one frame with borders on and one frame with no borders:

```
<FRAMESET ROWS = "25%, 75%">
  <FRAME FRAMEBORDER = "0" SRC = "topnovbar.htm">
  <FRAMESET COLS = "50%, 50%">
  <FRAME FRAMEBORDER = "0" SRC = "left.htm">
  <FRAME FRAMEBORDER = "1" SRC = "right.htm">
  </FRAMESET>
</FRAMESET>
</HTML>
```

ii. To create a frame with a yellow border.

```
<FRAME SRC = "document.htm "BORDERCOLOR = "YELLOW">
```

iii. `<FRAME SRC = "document.htm" SCROLLING = "YES">`

iv. `<FRAME SRC = "document2.htm" MARGINHEIGHT = "100" MARGINWIDTH = "200">`

This frame element would create a frame filled by the document2.htm file, which would be displayed with a top and bottom margin of 100 pixels, and a left and right margin of 200 pixels.

v. Suppose, we create a frameset document with two frames. We can name the second frame so that the links in the first frame will target it.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset // EN">
    <HTML LANG = "EN">
    <HEAD>
        <TITLE> Named Frames </TITLE>
    </HEAD>
    <FRAMESET COLS = "1*, 3*">
        <FRAME SRC = "navbar.htm">
        <FRAME SRC = "main.htm" NAME = "main">
    </FRAMESET>
```

We can create links in the left frame's document (navbar.htm) that target the right frame. A sample link in navbar.htm might look like this:

Read `<A HREF = "page2.htm" TARGET = "main"> page two </A>` for more information.

Tag: `<NOFRAMES> ... </NOFRAMES>`

Tag Name: Frame Alternatives

Explanation

Anyone using a non-frames-enabled browser will see a completely blank page if they view a frameset document that does not include alternate content. Therefore, it is imperative to use the no frames element to allow people to access the content of your Web site.

e.g.:
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset //EN">
<HTML LANG = "EN">
<HEAD>
<TITLE> NOFRAMES Alternative Text Example
</TITLE>
</HEAD>
<FRAMESET COLS = "50%, 50%">
<FRAME SRC = "document1.htm">
<FRAME SRC = "document2.htm">
<NOFRAMES>
<BODY>
If you see this text, your browser does not support frames or is not configured to display frames. Please go to the <A HREF = "noframes.htm"> alternate version </A> of this web page.
</BODY>
</NOFRAMES>
</FRAMESET>
</HTML>
```

Tag: <IFRAME> ... </IFRAME>

Tag Name: Inline Frame

Explanation

It defines an inline or floating frame.

Attributes:

ALIGN = (LEFT|CENTRE|RIGHT|TOP|MIDDLE|BOTTOM)

Sets the alignment of the frame or the surrounding text.

FRAMEBORDER = (1 | 0)

Renders a 3-D border around the frame. The default (1) inserts a border. To display no border, use 0

HEIGHT = (number ¦ "%")

Controls the height (in pixels) of the Inline frame.

MARGINHEIGHT = (number ¦ "%")

Controls the margin height (in pixels) for the frame.

MARGINWIDTH = (number ¦ "%")

Controls the margin width (in pixels) for the frame.

NAME = "text"

Provides a target name for the frame.

Scrolling = (yes no)

Creates a scrolling frame.

SRC = "URL"

Displays the source file for the frame.

WIDTH = (number ¦ "%")

Controls the width (in pixels) of the inline frame.

e.g.: The following HTML code inserts three inline frames centered within the document after the words "Primary Colors".

```
<!Doctype html public "-//W3C//DTD html 4.0//EN">

<html lang = "en">

<head>

<title> first inline frames example </title>

 </head>

<body bgcolour = "#000000" text = "# ffffff">

<P>

Primary Colors

<Center>

<Iframe src = "red.htm "marginheight = "0" frameborder = "10" width = "300" height = "25">

<P> The <A hRef = "red.htm"> red </A> file is available.

</Iframe>
```

<Iframe src = " yellow.htm" MARGINHEIGHT = "0" FRAMEBORDER = "10" WIDTH "300" HEIGHT = "25">

<P> The <A HREF = "yellow.htm"> yellow </A> file is available.

</IFRAME>

<IFRAME SRC = "blue.htm" MARGINHEIGHT = "0" FRAMEBORDER = "10" WIDTH "300" HEIGHT = "25">

<P> The <A HREF = "blue.htm"> blue </A> file is available.

</IFRAME>

</CENTER>

</BODY>

</HTML>

Each inline frame defaults to automatic scrolling, just like regular frames.

Student Activity

What does HTML stand for?

1. List the basic tags of HTML.
2. Write an HTML code using the Basic tags.
3. Write an HTML code to differentiate between <Block quote> and <P>?
4. How is a Subscript and Superscript displayed in HTML?
5. What tag is used to display the code of any programming language?
6. Differentiate between Strong and Bold tag.
7. How is the <Dir> tag different from <Menu> tag?
8. With an example explain the difference between ordered and unordered list?
9. Write the significance of using <DL>, <DT> and <DD> tags?
10. What is the role of <LI> tag in creating a list?
11. How are images displayed in the Web Page?
12. Write the attributes of <img> tag?
13. List the various format types of images, which can be inserted in a Web Page?
14. Can we use the name of the colors in the BGCOLOR property of the <Body>?

Unit 8

Server-side Programming, CGI and Perl Programming

Learning Objectives

After reading this unit you should appreciate the following:

8.1 Server-side Programming

8.2 Applications of Server-side Programming

8.3 Types of Server-side Programming

8.4 Client-Server Models

8.5 Common Gateway Interface (CGI)

8.6 CGI Programming Languages

8.7 Structure of a CGI Script

8.8 Environment Variables

8.9 Creating CGI Applications

8.10 Making CGI Applications Accessible

8.11 Example: Program in CGI and PERL

8.12 CGI Security Issues

8.13 CGI Script Command Line

8.14 Data Input to the CGI Script

8.15 Protocol-Specific Metavariables

8.16 Data Output from CGI Script

8.17 Client-side Programming

8.18 Other Scripting Languages

8.19 Brief Overview of Perl

8.20 Running Perl

8.21 Perl Command-Line Arguments

8.22 Perl Script

8.1 Server-side Programming

Server-side programming represents a set of techniques and technologies for extending the capabilities of web servers.

Opposed to client-side programming, which concentrates on the end user interface and data presentation at the client side, server side programming allows for enriching the functionality at the server side.

Using server-side programming technology, an http server can be enriched with a range of functionalities, from dynamic and custom web presentation to a complex framework for multi-tier distributed processing.

8.1.1 Need of Server-side Programming

A need for server side programming came in the early age of the WWW. Being based on the http protocol, web servers are much restricted in their functionality, especially when used for more complex web-based applications. Such a restriction made the http protocol a conceptual bottleneck of the web-based processing.

To preserve its original simplicity and to allow further advances, several server side techniques were introduced. Common Gateway Interface (CGI), Personal Homepage Processor (PHP), Active Server Pages (ASP) Server-Side Includes (SSI), Servlets and Java Server Pages (JSP) are among the most popular techniques.

8.1.2 Uses of Server-side Techniques

The use of server-side techniques to extend HTTP servers brings multiple advantageous allowing for a complex web-based processing (normally not supported by the sole use of web servers).

The following list presents some of the added value functionalities:

- Session management
- Support for other protocols (like ftp, smtp, pop, etc.)
- Support for other distributed services like rlogin , finger etc.
- Ability to program arbitrary functionality and services at the server
- Support for multi-tier platforms i.e. Middleware
- Support for the use of distributed databases
- Thin client programming
- Transparent functionalities at the client side

Most of the complex web-based systems are developed using server side programming techniques. The technology provides for server side extensions, while at the same side, does not affect the size and complexity of the client side processing.

8.1.3 Relationship of Server-side Techniques with Others

Server-side techniques are closely related to Web servers (e.g. Apache) script-based programming languages (in simpler applications using PHP, PEARL etc)) and multi-tier distributed systems architectures in advanced web based systems (using Java programming language and Enterprise Java Beans or .NET system framework).

8.1.4 Limitations of Server-side Programming

Conceptual limitation of the server side programming, in the Web based systems is in the overhead that "extra processing" of the server-side scripts. Execution of the server-side extensions, done in the context of the server machine operating systems or in the context of the web servers requires extra time. Stand-alone solution would have been a better approach.

8.2 Applications of Server-side Programming

All present web applications use some kind of server side programming. Since http protocol is fairly simple, even minimal functional requirements have to be programmed at the server side. Nowadays, there are numerous web-based applications, with the clear tendency to make most of the computing systems web enabled.

Some examples of the applications are:

- On-line banking
- Flight reservation
- Library systems
- E-learning
- On-line trading
- On-line games etc
- Interface to legacy systems ....

The typical tasks, build in to the most of the above mentioned systems are programmed by the server-side programming. Some of the typical tasks are:

- Security provision
- Session management
- DB connectivity
- Transaction management
- Support for SOAP and/or other protocols
- Support for web services

8.3 Types of Server-side Programming

The following are two types of server-side programming:

- Server-side scripts: Programs that build dynamic web pages
- Server pages: Web pages that have scripts embedded to created dynamic content

8.3.1 Server-side Scripting

Server-side scripts are small programs that reside on the web server to interact with the web environment. When browsers execute scripts that link to the code file, the script will do its designed functions and return a dynamically generated page.

Typical use of server-side programming:

- Personalized page content to the user's needs
- Enable the web browser to act as a database interface
- e-Commerce shopping carts

Types of Server-side Scripting

- CGI (Common Gateway Interface)
  - Simple scripting mechanisms supported by practically all web servers. Most commonly written in Perl, C/C++, or Python.
- Java Servlets
  - Ability to execute Java objects on the server. Newer, less common technology but has dramatic performance advantages over CGI.

Scripting tips

- When browsing the web you can typically tell that a hyperlink is calling a script by the file extension. Instead of the standard .html/.htm you will see .cgi, .pl or other extension.
- Many inexpensive, even free, scripts are available to enhance your web site.
- With most scripts the source code can be changed allowing custom applications at a fraction of the costs.

8.3.2 Server pages

Server Pages describes a technology for embedding scripts into web pages that can interact with the web environment. Server pages provide virtually the same dynamic abilities as server-side scripting However, because they are mostly standard web pages it is usually easier for web designers to alter for non-dynamic content. Server pages typically have a performance advantage over CGI scripts (not Servlets).

Typical use of server-side programming:

- Personalized page content to the user's needs
- Enable the web browser to act as a database interface
- Alter page content based on browser/OS platform

Types of Server page

Following are some common Server page types:

- ASP (Active Server Pages)
  - Microsoft's version of a scaleable engine. Built into Microsoft IIS
- JSP (Java Server Pages)
  - Newer, very powerful platform. Built into some web servers can be purchased for others.
- ColdFusion
  - Popular scaleable web applications can be purchased for most UNIX and WinNT servers

8.4 Client-Server Models

Java and Perl are probably the two most popular languages for server-side CGI programming. Perl is probably the best of the two because of its superior text processing facilities.

8.4.1 Two-tier Client-server Model

In the traditional two-tier client/server model, the server is handling the complete workload, which can be excessive on a heavily used e-commerce site, such as Amazon or Yahoo. Consider Figure 8.1.

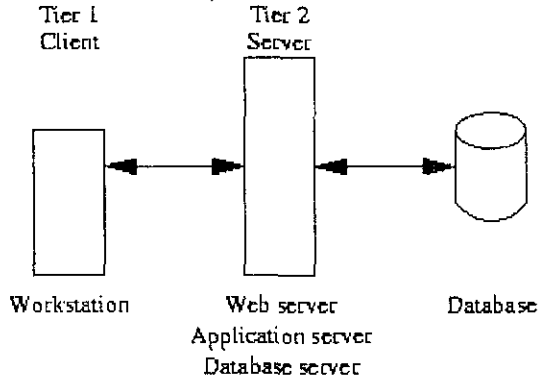


Figure 8.1: Two-tier Client-Server

In this case the server is not just serving web pages, it is also running applications (typically CGI scripts) and interacting with the database(s). However, this is perfectly acceptable for a lightly-used site, such as Mick's Apache server, where his Electronic Card Server maintains information in a local MySQL database.

8.4.2 Three-tier Client-Server Model

The three-tier model migrates the database transaction processing over to a back-end database server, thus relieving the web server from some of its workload. Consider Figure 8.2.

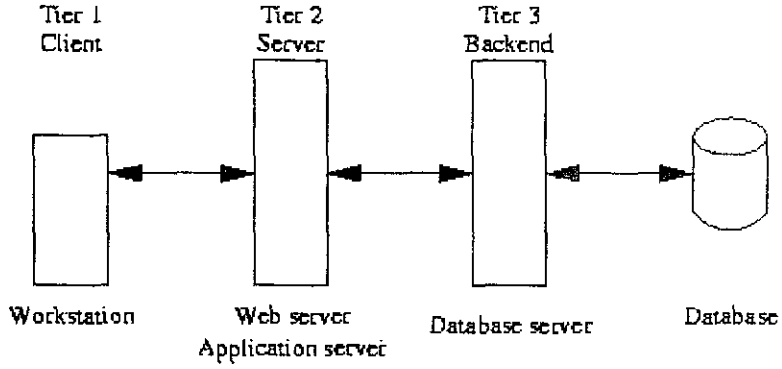


Figure 8.2: Three-tier Client-Sserver

Using two computers instead of one can result in a huge increase in the number of simultaneous clients that can be supported. The client request is formulated into an HTTP message and sent to the web server. The web server sends a request to the database server for information. The information returned from the database server is formatted by the web server and returned to the client.

8.4.3 N-tier Client-Server Model

A full separation of function into n tiers means that each server can be optimised for one particular job. Consider Figure 8.3.

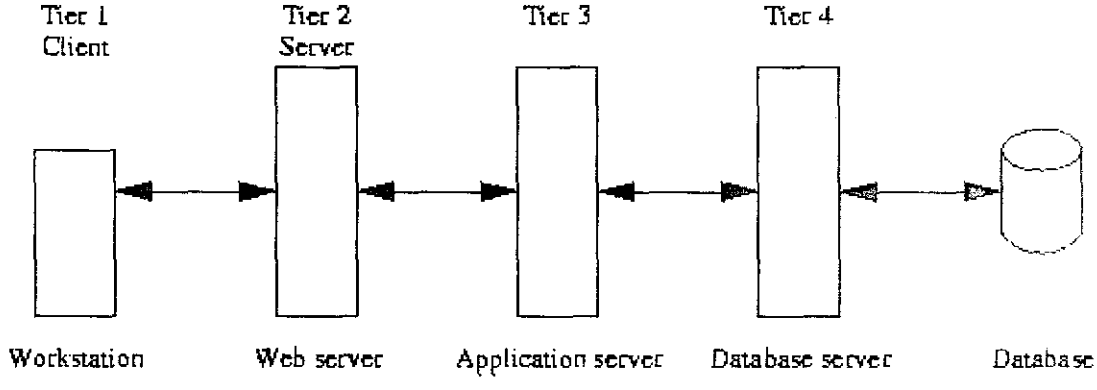


Figure 8.3: N-tier Client-Server

In this four-tier model the web server can be optimised for delivering web pages, the application server can be optimised for specific applications, and the database server can be optimised for specific database transactions.

By including some load balancing and redundancy, Figure 8.4 could be arrived.

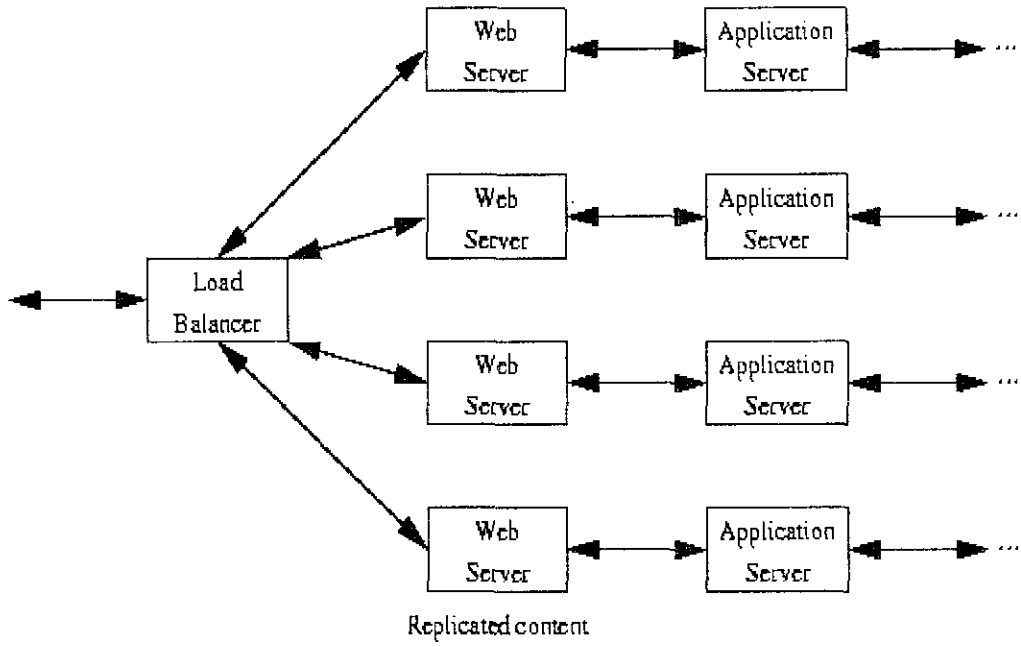


Figure 8.4: Server Cluster

A busy and important site can now balance the load across the server cluster (sometimes called a *server farm*) and, if one service fails, spread the load across the remaining units. This situation also scales reasonably well.

8.5 Common Gateway Interface (CGI)

The Common Gateway Interface (CGI) is a standard for interfacing external applications with information servers, such as HTTP or Web servers. A plain HTML document that the Web daemon **retrieves** is **static**, which means it exists in a constant state: a text file that doesn't change. A CGI program, on the other hand, is **executed** in real-time, so that it can output **dynamic** information.

The Common Gateway Interface (CGI) emerged as the first way to present dynamic content on the WWW. CGI allows the computer to generate web pages instantly at the user's request rather than just returning static content. It is also amazingly simple. Anyone with a small amount of programming experience can write rudimentary scripts that work.

CGI isn't magic; it is just programming with special types of input and some strict rules on output. Any language that can interrogate its environment variables, read its standard input, and write its standard output can be used. This simple model is illustrated in Figure 8.4.

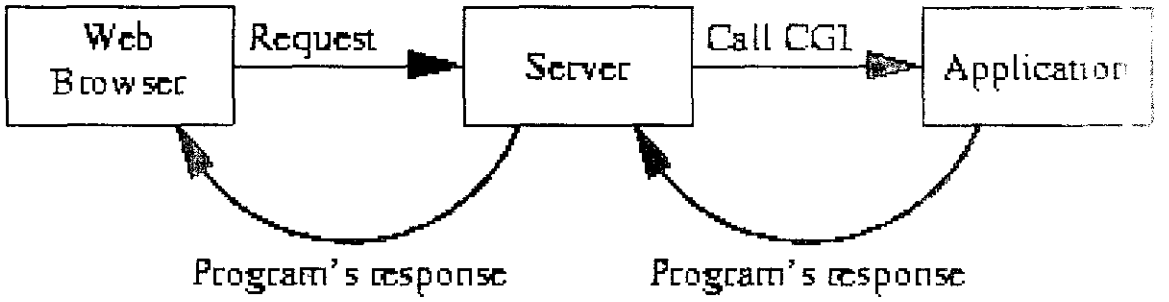


Figure 8.5: CGI in Action

The (client) web browser sends a form to the web server, which then invokes the appropriate CGI program with the form details as input. The output from this program is collected by the server and returned to the browser. Note that it is the responsibility of the CGI program to generate a correctly formed HTML document, including some of its header information.

The client typically requests a service and does very little work, other than rendering the HTML pages. The server receives these requests and responds appropriately. This may involve the server in some hefty processing, depending on the type of request.

A typical request message from a client to a server consists of three parts.

- A request line
- Optional request headers
- An optional entity body

The *request line* consists of a command, a target resource, and a protocol name and version. The *request headers* consist of name/value pairs providing additional information about the client and the request. The *entity body* provides yet more information for the server, e.g. cookies.

8.6 CGI Programming Languages

A CGI program can be written in any language that allows it to be executed on the system, such as

- C/C++
- Fortran
- PERL
- TCL
- Any Unix shell
- Visual Basic
- AppleScript

It just depends what you have available on your system. If you use a *programming language* like C or Fortran, you know that you must compile the program before it will run. If you look in the cgi-bin

directory that came with the server distribution, you will find the source code for some of the CGI programs in the /cgi-bin directory. If, however, you use one of the *scripting languages* instead, such as PERL, TCL, or a Unix shell, the script itself only needs to reside in the /cgi-bin directory, since there is no associated source code. Many people prefer to write CGI scripts instead of programs, since they are easier to debug, modify, and maintain than a typical compiled program.

8.7 Structure of a CGI Script

CGI is not a language but it is a simple protocol that can be used to communicate between Web forms and your program. A CGI script can be written in any language that can read STDIN, write to STDOUT, and read environment variables, i.e. virtually any programming language, including C, Perl, or even shell scripting.
Here's the typical sequence of steps for a CGI script:

1. Read the user's form input.
2. Do what you want with the data.
3. Write the HTML response to STDOUT.

The first and last steps are described below.

Reading the User's Form Input

When the user submits the form, your script receives the form data as a set of name-value pairs. The names are what you defined in the INPUT tags (or SELECT or TEXTAREA tags), and the values are whatever the user typed in or selected. (Users can also submit files with forms, but this primer doesn't cover that.)

This set of name-value pairs is given to you as one long string, which you need to parse. It's not very complicated, and there are plenty of existing routines to do it for you.

So just split on the ampersands or semicolons, then on the equal signs. Then, do two more things to each name and value:

1. Convert all "+" characters to spaces, and
2. Convert all "%xx" sequences to the single character whose ascii value is "xx", in hex. For example, convert "%3d" to "=".

This is needed because the original long string is **URL-encoded**, to allow for equal signs, ampersands, and so forth in the user's input.

So where do you get the long string? That depends on the HTTP method the form was submitted with:

- For GET submissions, it's in the environment variable **QUERY_STRING**.
- For POST submissions, read it from STDIN. The exact number of bytes to read is in the environment variable **CONTENT_LENGTH**.

Sending the Response Back to the User

First, write the line

Content-type: text/html

plus another blank line, to STDOUT. After that, write your HTML response page to STDOUT and it will be sent to the user when your script is done. That's all there is to it.

8.7.1 CGI Scripting Process Example

Figure 2 illustrates a request for Mick's cookie program (from a Windows NT system running the Netscape 6 browser).

```
GET /cgi-bin/cookie.cgi HTTP/1.0
Host: penguin.dcs.bbk.ac.uk
User-Agent: Mozilla/5.0 (Windows, U; Windows NT 5.0; en-US; m18) Gecko/20001108 Netscape6/6.0
Accept: */*
Accept-Language: en
Accept-Encoding: gzip,deflate,compress,identity
Keep-Alive: 300
Connection: keep-alive
```

Figure 8.6: Requesting the Cookie Program

Note that the browser will accept any form of content, preferably in English, and requesting that the connection remain open in case of further transfers.

Figure 8.7 illustrates the response from Mick's server (a Red Hat Linux system running the Apache web server.

```
HTTP/1.0 200 OK
Date: Wed, 24 Jan 2001 18:41:51 GMT
Server: Apache/1.3.12 (Unix) (Red Hat/Linux) PHP/3.0.15 mod_perl/1.21
Keep-Alive: timeout=15, max=99
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html
Content-Length: 1884

<HTML>
...
</HTML>
```

Figure 8.7: Response Message

The response is an HTML page delivered in (TCP) segments. Although the response is HTTP version 1.0, the server also wants the connection to remain open. The CGI program (in this case a Perl script, see below) only provides the Content-type information, leaving the server to provide the remainder of the details the HTTP protocol to transfer the data to the client. Figure 8.8 is a listing of the Perl script.

```
#!/usr/bin/perl -Tw
# Simple cookie demonstrator
#
# 0.0.0     Original implementation (mick, 17 Nov 1997)
# 0.1.0     Use server-side includes (mick, 24 Apr 2000)
# 0.2.0     Use global stylesheet (mick, 24 Jul 2000)
# 0.2.1 Modified stylesheet (mick, 4 Sep 2000)
#
use CGI qw(:standard);
use strict;

my $colour = param('colour');
my $cookie_name = 'ColourChoice';
my $cookie_value = cookie($cookie_name);
my $default;
my @rainbow = qw(Red Orange Yellow Green Blue Indigo Violet);
my $text;
if ($colour) {
        $default = $colour;
        $text = "(previous choice was $colour)";
} elsif ($cookie_value) {
        $default = $cookie_value;
        $text = "(cookie was $cookie_value)";
} else {
        $default = $rainbow[rand @rainbow];
        $text = "(don't be shy)";
}
my $cookie = cookie(-name => $cookie_name,
                    -value => $default);
print header(-cookie => $cookie);
print start_html(-dtd => '-//W3C//DTD HTML 4.0 Transitional//EN',
                 -head => Link({-href => '/penguin.css',
                                -rel => 'stylesheet',
                                -type => 'text/css'}),
                 -title => 'Colour Choice');
if (open HEADER, "$ENV{'DOCUMENT_ROOT'}/header.ssi") {
        print <HEADER>;
        close HEADER;
}
print h1({-align => 'center'},
         'Colour Choice!'),
        start_form(-action => $ENV{'SCRIPT_NAME'}),
        p({-class => 'clear'},
          "Choose a colour $text"),
        radio_group(-name => 'colour',
                    -values => \@rainbow,
                    -default => $default),
        br(),
        submit('Submit'),
        end_form();
if (open FOOTER, "$ENV{'DOCUMENT_ROOT'}/footer.ssi") {
        print <FOOTER>;
        close FOOTER;
}
print end_html();

BEGIN {
        die "Invalid user $<\n"
                unless $< == getpwnam('mick') or $< == getpwnam('nobody');
}
```

Figure 8.8: Cookie Program

The function call header(...) generates the Content-type and cookie strings. The function call start_html(...) generates the beginning of the document.

```
<HTML>
<HEAD>
...
</HEAD>
<BODY>
```

The function call end_html() generates the terminating string </BODY></HTML>. This script can be executed in one of three ways, considering the conditional statement following the variable declarations.

1. From the script itself, in which case the parameter colour is defined.
2. From a previous visit, in which case the cookie value is defined (assuming that it hasn't expired).
3. This is the first visit, or a previous cookie had expired.

This script uses the CGI module to generate correctly formed HTML. This module provides a function for each HTML element, so h1(...) generates <H1>...</H1>, p(...) generates <P>...</P>, etc. Function calls can even be nested. Element attributes are passed as an anonymous hash, e.g. {-align => 'center'}.

8.8 Environment Variables

Environment variables are a series of hidden values that the web server sends to every CGI you run. Your CGI can parse them, and use the data they send. Environment variables are stored in a hash called %ENV.

Table 8.1: Environment Variables

Variable Name	Value
DOCUMENT_ROOT	The root directory of your server
HTTP_COOKIE	The visitor's cookie, if one is set
HTTP_HOST	The hostname of your server
HTTP_REFERER	The URL of the page that called your script
HTTP_USER_AGENT	The browser type of the visitor
HTTPS	"on" if the script is being called through a secure server
PATH	The system path your server is running under
QUERY_STRING	The query string (see GET, below)
REMOTE_ADDR	The IP address of the visitor

REMOTE_HOST	The hostname of the visitor (if your server has reverse-name-lookups on; otherwise this is the IP address again)
REMOTE_PORT	The port the visitor is connected to on the web server
REMOTE_USER	The visitor's username (for .htaccess-protected pages)
REQUEST_METHOD	GET or POST
REQUEST_URI	The interpreted pathname of the requested document or CGI (relative to the document root)
SCRIPT_FILENAME	The full pathname of the current CGI
SCRIPT_NAME	The interpreted pathname of the current CGI (relative to the document root)
SERVER_ADMIN	The email address for your server's webmaster
SERVER_NAME	Your server's fully-qualified domain name (e.g. www.cgi101.com)
SERVER_PORT	The port number your server is listening on
SERVER_SOFTWARE	The server software you're using (such as Apache 1.3)

Some servers set other environment variables as well. Notice that some environment variables give information about your server, and will never change from CGI to CGI (such as SERVER_NAME and SERVER_ADMIN), while others give information about the visitor, and will be different every time someone accesses the script.

It can be seen from the above example that information needed by CGI applications are made available via (UNIX) environment variables. In Perl, these are in the %ENV hash (associative array).

8.9 Creating CGI Applications

There are many things that one must consider when designing CGI applications (CGI-apps). This section covers the basic requirements of such applications, including the input and output needed to make a CGI-app work, and how side effects can be used to an advantage.

The output from a CGI application contains two sections: the header and the body.

The header is always the first output that a CGI-app generates. A blank line immediately follows the header information, and the body follows this blank line. Generally, the header includes information about the data contained in the body.

A. Header Output

Usually, a CGI application doesn't need to produce much header information. When a WWW server returns a (static or dynamic) object to a client, it includes information about the object in the header. This information could be the time it was last modified, etc.

Such information is provided to the client by the server whether or not the page is created by a CGI-app if it is, then this information is merged with the header information that the CGI-app produces. There are three main pieces of information that a CGI application can include in its header:

- Content type. This describes the type of data contained in the body that the CGI application generates. This must always be included in the header. If this information isn't included, then the WWW client that receives the CGI-app output will not know what it is, and will not know how to handle it.

  The content type of a document should be a valid MIME type/subtype combination, and it should correspond with the type of data within that document. For example, if a CGI application produces an HTML document, it should have the MIME type/subtype combination of text/html. Here are some examples of document types and their associated MIME type/subtype combinations (Figure 8.9):

Document Type	MIME type/subtype
HTML	text/html
ASCII text	text/plain
GIF image	image/gif
QuickTime movie	video/quicktime

Figure 8.9: Some MIME Types

  The content type should appear in the CGI-app header output on a single line as follows:

  Content-type: *type/subtype*

  where *type/subtype* is the MIME type/subtype combination for the data contained in the CGI output body.

- Location. This gives an alternate location for the client to access; how this location is used is up to the client. This location should appear in the CGI header output on a single line as follows:

  Location: *URL*

  where *URL* is the alternate location.

- Status. This returns a status code to the WWW client. This should appear in the CGI header output on a single line as follows:

  Status: *XXX Message*

  where *XXX* is a three-digit status code and *Message* is a message string.

a) Body. The body of the CGI-app output follows the blank line which separates it from the header. The body contains all of the data which is to be displayed by the WWW client. If the content type

specified by the header is text/html, then the body should contain the HTML code that the CGI application generates. If a CGI-app generates a GIF image, then the body of the output should contain the bytes that make up a valid GIF image.

b) Handling Input. When a client asks a server to launch a CGI application, it can give the server some input to provide to the CGI-app. There are three important ways to give input from a client to a CGI application:

1. Through a query string
2. Through the command line
3. Through standard input

1. Query String. A WWW client can send what's called a query to a CGI application by appending a '?' followed by a query string to the URL for the CGI application. This query string is composed of name-value pairs in the form of *name=value*, where *name* is the name of a variable and *value* is the value assigned to it. Each name-value pair in the query string is separated by a '&'. The query string is sent to the CGI application through the environment variable QUERY_STRING.

   The most common way to send a query string to a CGI application is by setting up an HTML form that uses the HTTP GET method. Each item in the form that can take on a value is given a unique name. When the form is filled out by a user, the user's WWW client sends back the names of the items and the values the user gave those items in the form of a query string. The CGI application to which the client sends this information can then dissect the query string to find the information provided by the client. Since the query is appended to the URL for the CGI application, it is "URL encoded," and must be decoded. There are some characters that have special purposes in a URL, such as the colon (':') and forward slash ('/'). Some characters are not allowed to appear in URLs at all, such as spaces. For this reason, URLs are encoded in a special way. All spaces are replaced by a plus sign ('+'), and all special characters are replaced by *%xx*, where *xx* is the hexadecimal representation of the ASCII value for a character. All URL encoded strings must be decoded before they can be used properly.

2. Command Line Parameters. Passing parameters to a program on the command line is an easy way to provide information to an application. This method, however, is rarely used for CGI applications. It is useful, though, when an author of a CGI application wants to pass a single parameter to the application without having to parse a query string. This method was originally designed for use with the ISINDEX tag, but it may be used in other ways.

   Let's say that there is a server called "www.nowhere.com", and on that server in a directory called "cgi-bin" exists a CGI application called "foo.cgi". This application may be accessed through the following URL:

   http://www.nowhere.com/cgi-bin/foo.cgi

   If we were to send a query to this CGI application with no equals sign ('=') in the query string, then the query string would be passed to the CGI application through the command line. If an equals sign is present in the query string after the question mark ('?'), then the entire query string is provided through the QUERY_STRING environment variable. So, if the following URL is accessed:

http://www.nowhere.com/cgi-bin/foo.cgi?chicken

then the string "chicken" would be the first parameter passed to foo.cgi when the server started its execution. If the following URL is accessed:

http://www.nowhere.com/cgi-bin/foo.cgi?yummy=chicken

then the string "yummy=chicken" would be placed in the QUERY_STRING environment variable foo.cgi should ignore anything in the list of command line parameters passed to it upon execution.

3. Standard Input. Information is only given to a CGI application through standard input when the CGI is accessed with either the POST or PUT methods. The body of the POST or PUT request (sent by the client to the server) is used as the standard input to the CGI application. The information in the body transferred by the POST method is URL encoded.

   For example, if a user fills out a World Wide Web form, the WWW client can send the user's form data to a CGI-app using the POST method. The CGI application which receives this data will have the form information provided to it through standard input. In fact, the format of the information will be the same as if it were provided in the QUERY_STRING environment variable via the GET method.

8.10 Making CGI Applications Accessible

When a CGI application is ready to be made available to users, it must be stored in a location that can be accessed by the World Wide Web server running on the host machine. This way, when a user tries to access the URL that points to that CGI-app, the server will have the ability to execute it. The server must, however, know that the file to which the user's URL points is a CGI application and not a static document, such as a text or HTML file. There are usually two ways of accomplishing this:

Store the CGI application in a place that is used solely for CGI use. Usually, WWW servers have certain directories into which only CGI applications are placed. Depending upon how the server is configured, it may be necessary to have administrative access to place CGI-apps into these directories.

- Add a certain extension to the CGI application file name. If the file name for a CGI application ends with an extension that lets the server know that the file is a CGI-app, then the server will execute it instead of trying to return it as a static document.

8.11 Example: Program in CGI and PERL

Let's write a simple first program. Enter the following lines into a new file, and name it "first.pl"

```
#!/usr/bin/perl

Print "Hello, world!\n";
```

Save the file. Now, in the Unix shell, you'll need to type:

```
chmod 755 first.pl
```

This changes the file permissions to allow you to run the program. You will have to do this every time you create a new script; however, if you're editing an existing script, the permissions will remain the same and won't need to be changed again.

Now, in the Unix shell, you'll need to type this to run the script:

./first.pl

If all goes well, you should see it print Hello, world! to your screen.

A CGI program is still a Perl script. But one important difference is that a CGI usually generates a web page (for example: a form-processing CGI, such as a guestbook, usually returns a "thank you for writing" page.) If you are writing a CGI that's going to generate a HTML page, you must include this statement somewhere in the script, before you print out anything else:

print "Content-type:text/html\n\n";

This is a content header that tells the receiving web browser what sort of data it is about to receive - in this case, an HTML document. If you forget to include it, or if you print something else before printing this header, you'll get an "Internal Server Error" when you try to access the CGI. A good rule of thumb is to put the Content-type line at the top of your script (just below the #!/usr/bin/perl/ line).

Now let's take our original first.pl script, and make it into a CGI script that displays a web page. If you are running this on a Unix server that lets you run CGIs in your public_html directory, you will probably need to rename the file to first.cgi, so that it ends in the .cgi extension. Here is what it should look like:

#!/usr/bin/perl

print "Content-type:text/html\n\n";

print "<html><head><title>Test Page</title></head>\n";

print "<body>\n";

print "<h2>Hello, world!</h2>\n";

print "</body></html>\n";

to run this Just move it into your public_html or CGI-bin directory, and type the direct URL for the CGI

the output would be

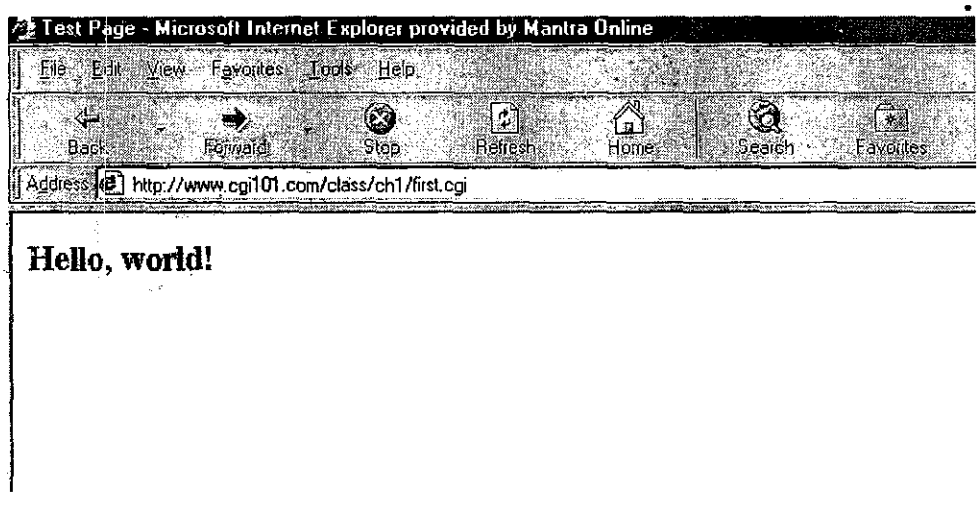


Figure 8.10

8.12 CGI Security Issues

CGI programming offers you something amazing: as soon as your script is online, it is immediately available to the entire world. Anyone from almost anywhere can run the application you created on your web server. This may make you excited, but it should also make you scared.

Not everyone using the Internet has honest intentions. Crackers may attempt to vandalize your web pages in order to show off to friends. Competitors or investors may try to access internal information about your organization and its products.

Not all security issues involve malevolent users. The worldwide availability of your CGI script means that someone may run your script under circumstances you never imagined and certainly never tested. Your web script should not wipe out files because someone happened to enter an apostrophe in a form field, but this is possible, and issues like these also represent security concerns.

1. **Malicious Attacks.** Whenever a server is operating on a computer, that server can provide potential security leaks. If a server is providing read-only access to a set of documents, there is some possibility that documents may be read by people for whom the documents were not intended. If a server provides write access, then the computer is opened up to many more security threats. It is when a server provides the ability to run programs on the host computer that the worst consequences can arise, unless the server is adequately protected.

   Read access. The main purpose of a World Wide Web server is to provide many people with the ability to read certain documents. A server must be set up properly to avoid allowing others to read files on the host computer that were not meant to be accessible, such as system configuration or password files. If we assume that this is the case, then there are really only two ways to limit access to the public documents -- through server controls, and through CGI application-based authorization.

   Most WWW servers are designed to allow server administrators to restrict access to the documents that the system provides. For example, a server may grant access only to client requests coming from within a certain internet domain. Or, a server may only grant access to users who authenticate themselves. CGI authorization relies on some means of authentication.

   Write access. Some WWW servers allow clients to write directly to the document space that the server makes available. This allows a great deal of flexibility, yet provides the potential for some sticky security problems. Write access is provided by servers in the form of the PUT and DELETE methods of HTTP/1.1, and the multipart/form-data encoding type, which allows file uploading to take place within a WWW .It is up to the server to ensure that no one is allowed to maliciously alter the document space using the PUT and DELETE methods. But it is also up to the server administrator to ensure that any files uploaded to the host machine are appropriate and that they don't contain any viruses. The use of forms for file uploading requires the use of a CGI application to process the completed form that is POSTed to the

server. This CGI-app can take the responsibility of placing any uploaded documents in a safe place until they can be reviewed, or can automatically perform virus checking on the documents. In this way, CGI-apps can provide some safeguards against attacks while still providing flexibility to users.

Execute access. The greatest potential for security problems occurs when a server allows users to execute programs on the host computer. Consider this: a person can't get a virus from reading an email message (reading a message only involves viewing text characters), but if a person executes a program attached to an email message, and that program has a virus, then the person's system will probably become infected by the virus. The virus can only spread when part of it is being executed by the computer.

When a CGI application is used as a gateway to another application, the CGI-app will often make system calls -- that is, it will execute other programs on the host system. Frequently, the CGI-app will pass command-line parameters in these system calls, and the information provided in these parameters will come from a query string sent by a WWW client. If a user designs a clever query string, and the CGI-app doesn't sense what the user is doing, it is possible for a user to gain the ability to issue commands on the host system as if they were a user logged onto the system. Sometimes, servers will execute CGI applications with administrative access privileges -- which may give the intrusive user full control over the system!

A common illustration of this potential security hole is a "finger" gateway. The "finger" utility is a way of finding out a little bit of information about users of a system -- information that the users provide about themselves to the public. For example, if one were to issue the command "finger vanmetre@csgrad.cs.vt.edu" on a Unix or similar system, one could find out some more about the author of this section. Common "finger gateways" are CGI applications that accept a user name as a query string from a WWW client, perform a system call to the finger utility, and return the results to the client. So, if a client were to access a CGI application called finger.cgi with the URL

http://www.nowhere.com/cgi-bin/finger.cgi?vanmetre

the CGI-app would issue the system command

finger vanmetre

and return the results to the WWW client. Now suppose a user submitted the URL

http://www.nowhere.com/cgi-bin/finger.cgi?vanmetre;rm+-rf+%2F

After the CGI decodes this, it will issue the system command

finger vanmetre;rm -rf /

If the CGI-app was executed with full system privileges, this will not only return the finger information for user vanmetre, but it will also erase the contents of the hard drives on the system! Even if the CGI-app was not executed with full privileges, a user could create a query string that could mail off a password file, open a remote telnet connection, or do something else that would aid a break-in to the system.

2. **Authentication.** There are two important steps to maintaining a certain level of security within a World Wide Web site -- authentication and authorization. Authentication is how a server identifies users and makes sure that they really are who they say they are+.

   Authorization is the process of determining to which documents an authenticated user has access.

   Often, in the context of the World Wide Web, authentication is provided by a simple name/password combination. When a user enters a protected site, the server asks the user's WWW client for authentication. The WWW client then asks the user for a name/password combination which it then sends back to the server. If the combination is a valid one, the user is authenticated and can then be authorized to access the server's documents. Each time the authenticated user makes a request for a protected document, the name/password combination is resubmitted by the client to the server.

   Servers can also provide authorization protection for specific documents or for sets of documents within the server document space. For example, all of the contents of a directory on a server can be assigned a list of authorized users, who must properly be authenticated before they can access the directory; all users who fail authentication will be denied access.

   To provide a bit more flexibility with document control, authorization can be left up to a CGI application. When an authenticated user accesses a CGI application, an environment variable is set which provides the name of the user to the CGI-app; the CGI-app can decide whether or not to provide access to the user. In addition, a CGI application that generates complex WWW pages can present different pages to different users, as long as they are authenticated. A WWW site can keep a list of preferences or configuration details for each user and apply these preferences to the page generation methods of the site.

3. **Language.** There are many points to consider when choosing a language to use when developing CGI applications. Here is a list of some of them:

   Interpreted vs. Compiled. There are many interpreted languages, such as PERL, that have become popular among CGI-app developers, and there are many compiled languages including C, that are popular, too. There are advantages and disadvantages to both; these must be considered before developing CGI applications.

   Programs written in compiled languages generally run faster than equivalent programs written in interpreted languages. However, interpreted languages can be more flexible, and can make prototyping much easier. If any changes need to be made to a CGI application, they can be made to an interpreted version much more quickly that to a compiled version, because the compiled version must be recompiled before the changes can take effect. Once a CGI application is in place, though, a user may not see much of a difference between an interpreted CGI-app and a compiled equivalent.

   Portability/Modifiability. When developers choose a language to use for a CGI application they must always think of the long term. They must consider how the site will operate in several years to make sure that the CGI application will not only meet the needs of the site today, but also in the future. The World Wide Web is changing extremely rapidly, so it is often difficult to keep the future in perspective.

The needs of a WWW site are always changing. As server technology advances, it is likely that a site will undergo some hardware and/or software modifications, while the contents of the site must stay the same. To make such transitions as smooth as possible, one must consider the portability and modifiability of CGI applications. If a language is chosen for a site that uses proprietary hardware or software, then the CGI-apps developed with that language may not be usable should that hardware or software change. One should choose a stable, flexible language (such as C) that will persist while needs and resources may fluctuate and evolve.

Development Time. Quite often, a WWW site needs to be constructed very quickly. If CGI applications are needed at that site, then it may be necessary to choose a language that allows rapid development, yet may be weaker in other areas. CGI creation often takes the greatest proportion of development time, so any steps to minimize that development time may weigh heavily when a language is to be chosen. For a site that will not exist for very long (for example, a site that provides up-to-date Olympics results), development time is much more important than preparing for the future of the site. Interpreted languages such as PERL are often used for rapid development.

Systems Interfacing. When a CGI-app is providing an interface to a second application, the dominant concern for language choice is often dictated by that second application. If a developer is creating a CGI-based Web interface to a proprietary database, then the CGI applications will have to access the database's programming interface, which may be very limiting. Many CGI applications are forced to make system calls (run programs from the command line) to interface with other applications; in this case, the secondary application may not have as much influence on language choice.

4. **Performance.** For World Wide Web sites that serve mainly static document requests, the performance bottleneck is often the server's network connection. It doesn't take much time or computational power to read a file from a hard disk, but it does take some time to send that file over a slow network connection. As servers gain faster and faster network connections, the source of congestion may not be the network. This is especially true on WWW servers that provide many dynamically produced documents. In this case, to serve one client request, it takes much more time and effort than it does to simply read a file and return it. When running a CGI application, the system needs to start a new process, execute the code (which could be complex), collect the results, and then return them. If the CGI-app is a gateway to a second application, say, a database, then there may be much time spent by the second application to provide any data needed by the CGI-app. While CGIs increase the flexibility of a server, they can significantly decrease the server's performance.

8.12.1 Need of CGI Security

Many CGI developers do not take security as seriously as they should. Following are certain aspects that make web security very important.

- *On the Internet, your web site represents your public image.*

  If your web pages are unavailable or have been vandalized, that affects others' impressions of your organization, even if the focus of your organization has nothing to do with web technology.

- *You may have valuable information on your web server.*

  You may have sensitive or valuable information available in a restricted area that you may wish to keep unauthorized people from accessing.

  For example, you may have content or services available to paying members, which you would not want non-paying customers or non-members to access. Even files that are not part of your web server's document tree and are thus not available online to anyone (e.g., credit card numbers) could be compromised.

- *Someone who has cracked your web server has easier access to the rest of your network.*

  If you have no valuable information on your web server, you probably cannot say that about your entire network. If someone breaks into your web server, it becomes much easier for them to break into another system on your network, especially if your web server is inside your organization's firewall.

- *You sacrifice potential income when your system is down.*

  If your organization generates revenue directly from your web site, you certainly lose income when your system is unavailable. However, even if you do not fall into this group, you likely offer marketing literature or contact information online.

  Potential customers who are unable to access this information may look elsewhere when making their decision.

- *You waste time and resources fixing problems.*

  You must perform many tasks when your systems are compromised. First, you must determine the extent of the damage. Then you probably need to restore from backups. You must also determine what went wrong.

  If a cracker gained access to your web server, then you must determine how the cracker managed this in order to prevent future break-ins. If a CGI script damaged files, then you must locate and fix the bug to prevent future problems.

- *You expose yourself to liability.*

  If you develop CGI scripts for other companies, and one of those CGI scripts is responsible for a large security problem, then you may understandably be liable. However, even if it is your company for whom you're developing CGI scripts, you may be liable to other parties.

  For example, if someone cracks your web server, they could use it as a base to stage attacks on other companies. Likewise, if your company stores information that others consider sensitive (e.g., your customers' credit card numbers), you may be liable to them if that information is leaked.

These are only some of the many reasons why web security is so important. You may be able to come up with other reasons yourself. So now that you recognize the importance of creating secure CGI scripts, you may be wondering what makes a CGI script secure. It can be summed up in one simple maxim: *never trust any data coming from the user.* This sounds quite simple, but in practice it's not.

8.12.2 Example: CGI Script Security

A CGI script is a program that anyone in the world can run *on your machine*. Accordingly, look out for security holes as you write your script.

Mostly, **don't trust the user input.** In particular, don't put user data in a shell command without verifying the data carefully, lest a hacker drive a virtual truck through this security hole.

Let's say you have a CGI program that lets users run "finger" on your host. Such a Perl script might have a line like

```
system "finger $username" ;
```

But if a malicious user enters "james; rm -rf /" as the username, your program runs

```
system "finger james; rm -rf /" ;
```

which erases as many of your files as possible, probably not what you intended. So verify that the username is valid, with something like

```
$username!~ /[^\w.-]/  || die "Whoa!  Nice try, buddy." ;
```

or use a different form of the system command:

```
system("finger", $username) ;
```

or come up with another way to solve the problem.

It's easy for a hacker to send *any* form variables to your script, with any values (even non-printable characters). Your security shouldn't rely on fields having certain values, or even existing or not existing.

8.13 CGI Script Command Line

Some systems support a method for supplying an array of strings to the CGI script. This is only used in the case of an 'indexed' query. This is identified by a "GET" or "HEAD" HTTP request with a URL query string not containing any uuencoded "=" characters. For such a request, servers SHOULD parse the search string into words, using the following rules:

```
search-string = search-word *( "+" search-word )

search-word   = 1*schar
```

schar = xunreserved | escaped | xreserved

xunreserved = alpha | digit | xsafe | extra

xsafe = "$" | "-" | "_" | "."

xreserved = ";" | "/" | "?" | ":" | "@" | "&"

After parsing, each word is URL-decoded, optionally encoded in a system defined manner, and then the argument list is set to the list of words.

If the server cannot create any part of the argument list, then the server SHOULD NOT generate any command line information. For example, the number of arguments may be greater than operating system or server limitations permit, or one of the words may not be representable as an argument.

Scripts SHOULD check to see if the QUERY_STRING value contains an unencoded "=" character, and SHOULD NOT use the command line arguments if it does.

8.14 Data Input to the CGI Script

Information about a request comes from two different sources: the request header, and any associated message-body. Servers MUST make portions of this information available to scripts.

Request Metadata (Metavariables)

Each CGI server implementation MUST define a mechanism to pass data about the request from the server to the script. The metavariables containing these data are accessed by the script in a system defined manner. The representation of the characters in the metavariables is system defined.

This specification does not distinguish between the representation of null values and missing ones. Whether null or missing values (such as a query component of "?" or "", respectively) are represented by undefined metavariables or by metavariables with values of "" is implementation-defined.

Case is not significant in the metavariable names, in that there cannot be two different variables whose names differ in case only. Here they are shown using a canonical representation of capitals plus underscore ("_"). The actual representation of the names is system defined; for a particular system the representation MAY be defined differently than this.

Metavariable values MUST be considered case-sensitive except as noted otherwise.

The canonical metavariables defined by this specification are:

AUTH_TYPE

CONTENT_LENGTH

CONTENT_TYPE

GATEWAY_INTERFACE

PATH_INFO

PATH_TRANSLATED

QUERY_STRING

REMOTE_ADDR

REMOTE_HOST

REMOTE_IDENT

REMOTE_USER

REQUEST_METHOD

SCRIPT_NAME

SERVER_NAME

SERVER_PORT

SERVER_PROTOCOL

SERVER_SOFTWARE

Metavariables with names beginning with the protocol name (*e.g.*, "HTTP_ACCEPT") are also canonical in their description of request header fields. The number and meaning of these fields may change independently of this specification.

AUTH_TYPE

This variable is specific to requests made *via* the "http" scheme.

If the Script-URI required access authentication for external access, then the server MUST set the value of this variable from the 'auth-scheme' token in the request's "Authorization" header field. Otherwise it is set to NULL.

AUTH_TYPE = "" | auth-scheme

auth-scheme = "Basic" | "Digest" | token

Servers MUST provide this metavariable to scripts if the request header included an "Authorization" field that was authenticated.

CONTENT_LENGTH

This metavariable is set to the size of the message-body entity attached to the request, if any, in decimal number of octets. If no data are attached, then this metavariable is either NULL or not defined.

CONTENT_LENGTH = "" | 1*digit

Servers MUST provide this metavariable to scripts if the request was accompanied by a message-body entity.

CONTENT_TYPE

If the request includes a message-body, CONTENT_TYPE is set to the Internet Media Type of the attached entity if the type was provided *via* a "Content-type" field in the request header, or if the server can determine it in the absence of a supplied "Content-type" field. The syntax is the same as for the HTTP "Content-Type" header field.

CONTENT_TYPE	=	"" | media-type
media-type	=	type "/" subtype *(";" parameter)
type	=	token
subtype	=	token
parameter	=	attribute "=" value
attribute	=	token
value	=	token | quoted-string

The type, subtype, and parameter attribute names are not case-sensitive. Parameter values MAY be case sensitive..

Example:

application/x-www-form-url encoded

There is no default value for this variable. If and only if it is unset, then the script MAY attempt to determine the media type from the data received. If the type remains unknown, then the script MAY choose to either assume a content-type of application/octet-stream or reject the request with a 415 ("Unsupported Media Type") error.

Servers MUST provide this metavariable to scripts if a "Content-Type" field was present in the original request header. If the server receives a request with an attached entity but no "Content-Type" header field, it MAY attempt to determine the correct datatype, or it MAY omit this metavariable when communicating the request information to the script.

GATEWAY_INTERFACE

This metavariable is set to the dialect of CGI being used by the server to communicate with the script Syntax:

GATEWAY_INTERFACE = "CGI" "/" major "." minor

major = 1*digit

minor = 1*digit

Note that the major and minor numbers are treated as separate integers and hence each may be more than a single digit. Thus CGI/2.4 is a lower version than CGI/2.13 which in turn is lower than CGI/12.3. Leading zeros in either the major or the minor number MUST be ignored by scripts and SHOULD NOT be generated by servers.

8.15 Protocol-Specific Metavariables

These metavariables are specific to the protocol *via* which the request is made. Interpretation of these variables depends on the value of the SERVER_PROTOCOL metavariable

Metavariables with names beginning with "HTTP_" contain values from the request header, if the scheme used was HTTP. Each HTTP header field name is converted to upper case, has all occurrences of "-" replaced with "_", and has "HTTP_" prepended to form the metavariable name. Similar transformations are applied for other protocols. The header data MAY be presented as sent by the client, or MAY be rewritten in ways which do not change its semantics. If multiple header fields with the same field-name are received then the server MUST rewrite them as though they had been received as a single header field having the same semantics before being represented in a metavariable. Similarly, a header field that is received on more than one line MUST be merged into a single line. The server MUST, if necessary, change the representation of the data (for example, the character set) to be appropriate for a CGI metavariable.

Servers are not required to create metavariables for all the request header fields that they receive. In particular, they MAY decline to make available any header fields carrying authentication information, such as "Authorization", or which are available to the script *via* other metavariables, such as "Content-Length" and "Content-Type".

PATH_INFO

The PATH_INFO metavariable specifies a path to be interpreted by the CGI script. It identifies the resource or sub-resource to be returned by the CGI script, and it is derived from the portion of the URI path following the script name but preceding any query data. The syntax and semantics are similar to a decoded HTTP URL 'path', with the exception that a PATH_INFO of "/" represents a single void path segment.

PATH_INFO = "" | ("/" path)

path = segment *("/" segment)

segment = *pchar

pchar = <any CHAR except "/">

The PATH_INFO string is the trailing part of the <path> component of the Script-URI that follows the SCRIPT_NAME portion of the path.

Servers MAY impose their own restrictions and limitations on what values they will accept for PATH_INFO, and MAY reject or edit any values they consider objectionable before passing them to the script.

Servers MUST make this URI component available to CGI scripts. The PATH_INFO value is case-sensitive, and the server MUST preserve the case of the PATH_INFO element of the URI when making it available to scripts.

PATH_TRANSLATED

PATH_TRANSLATED is derived by taking any path-info component of the request URI, decoding it , parsing it as a URI in its own right, and performing any virtual-to-physical translation appropriate to

map it onto the server's document repository structure. If the request URI includes no path-info component, the PATH_TRANSLATED metavariable SHOULD NOT be defined.

PATH_TRANSLATED = *CHAR

For a request such as the following:

http://somehost.com/cgi-bin/somescript/this%2eis%2epath%2einfo

the PATH_INFO component would be decoded, and the result parsed as though it were a request for the following:

http://somehost.com/this.is.the.path.info

This would then be translated to a location in the server's document repository, perhaps a filesystem path something like this:

/usr/local/www/htdocs/this.is.the.path.info

The result of the translation is the value of PATH_TRANSLATED.

The value of PATH_TRANSLATED may or may not map to a valid repository location. Servers MUST preserve the case of the path-info segment if and only if the underlying repository supports case sensitive names. If the repository is only case-aware, case-preserving, or case-blind with regard to document names, servers are not required to preserve the case of the original segment through the translation.

The translation algorithm the server uses to derive PATH_TRANSLATED is implementation defined. CGI scripts which use this variable may suffer limited portability.

Servers SHOULD provide this metavariable to scripts if and only if the request URI includes a path-info component.

QUERY_STRING

A URL-encoded string; the <query> part of the Script-URI.

QUERY_STRING = query-string

query-string = *uric

Servers MUST supply this value to scripts. The QUERY_STRING value is case-sensitive. If the Script URI does not include a query component, the QUERY_STRING metavariable MUST be defined as an empty string ("").

REMOTE_ADDR

The IP address of the client sending the request to the server. This is not necessarily that of the user agent (such as if the request came through a proxy).

REMOTE_ADDR = hostnumber

hostnumber = ipv4-address | ipv6-address

Servers MUST supply this value to scripts.

REMOTE_HOST

The fully qualified domain name of the client sending the request to the server, if available, otherwise NULL.. Domain names are not case sensitive. Servers SHOULD provide this information to scripts.

REMOTE_IDENT

The identity information reported about the connection ,request to the remote agent, if available. Servers MAY choose not to support this feature, or not to request the data for efficiency reasons.

REMOTE_IDENT = *CHAR

The data returned may be used for authentication purposes, but the level of trust reposed in them should be minimal.

Servers MAY supply this information to scripts.

REMOTE_USER

If the request required authentication using the "Basic" mechanism (*i.e.*, the AUTH_TYPE metavariable is set to "Basic"), then the value of the REMOTE_USER metavariable is set to the user-ID supplied. In all other cases the value of this metavariable is undefined.

REMOTE_USER = *OCTET

This variable is specific to requests made *via* the HTTP protocol.

Servers SHOULD provide this metavariable to scripts.

REQUEST_METHOD

The REQUEST_METHOD metavariable is set to the method with which the request was made.

REQUEST_METHOD = http-method

http-method = "GET" | "HEAD" | "POST" | "PUT" | "DELETE"

| "OPTIONS" | "TRACE" | extension-method

extension-method = token

The method is case sensitive. CGI/1.1 servers MAY choose to process some methods directly rather than passing them to scripts.

This variable is specific to requests made with HTTP.

Servers MUST provide this metavariable to scripts.

SCRIPT_NAME

The SCRIPT_NAME metavariable is set to a URL path that could identify the CGI script (rather than the script's output). The syntax and semantics are identical to a decoded HTTP URL 'path' token.

SCRIPT_NAME = "" | ("/" [path])

The SCRIPT_NAME string is some leading part of the <path> component of the Script-URI derived in some implementation defined manner. No PATH_INFO or QUERY_STRING segments are included in the SCRIPT_NAME value.

Servers MUST provide this metavariable to scripts.

SERVER_NAME

The SERVER_NAME metavariable is set to the name of the server, as derived from the <host> part of the Script-URI. SERVER_NAME = hostname | hostnumber

Servers MUST provide this metavariable to scripts.

SERVER_PORT

The SERVER_PORT metavariable is set to the port on which the request was received, as used in the <port> part of the Script-URI.

SERVER_PORT = 1*digit

If the <port> portion of the script-URI is blank, the actual port number upon which the request was received MUST be supplied.

Servers MUST provide this metavariable to scripts.

SERVER_PROTOCOL

The SERVER_PROTOCOL metavariable is set to the name and revision of the information protocol with which the request arrived. This is not necessarily the same as the protocol version used by the server in its response to the client.

SERVER_PROTOCOL = HTTP-Version | extension-version

| extension-token

HTTP-Version = "HTTP" "/" 1*digit "." 1*digit

extension-version = protocol "/" 1*digit "." 1*digit

protocol = 1*(alpha | digit | "+" | "-" | ".")

extension-token = token

'protocol' is a version of the <scheme> part of the Script-URI, but is not identical to it. For example, the scheme of a request may be "https" while the protocol remains "http". The protocol is not case sensitive, but by convention, 'protocol' is in upper case.

A well-known extension token value is "INCLUDED", which signals that the current document is being included as part of a composite document, rather than being the direct target of the client request

Servers MUST provide this metavariable to scripts.

SERVER_SOFTWARE

The SERVER_SOFTWARE metavariable is set to the name and version of the information server software answering the request (and running the gateway).

SERVER_SOFTWARE = 1*product

product = token ["/" product-version]

product-version = token

Servers MUST provide this metavariable to scripts.

8.16 Data Output from CGI Script

There MUST be a system defined method for the script to send data back to the server or client; a script MUST always return some data. Unless defined otherwise, this will be *via* the 'standard output' file descriptor.

There are two forms of output that scripts can supply to servers:

- Non-parsed header (NPH) output, and
- Parsed header output.

Servers MUST support parsed header output and MAY support NPH output. The method of distinguishing between the two types of output (or scripts) is implementation defined.

Servers MAY implement a timeout period within which data must be received from scripts. If a server implementation defines such a timeout and receives no data from a script within the timeout period, the server MAY terminate the script process and SHOULD abort the client request with either a '504 Gateway Timed Out' or a '500 Internal Server Error' response.

8.16.1 Non-Parsed Header Output

Scripts using the NPH output form MUST return a complete HTTP response message. NPH scripts MUST use the SERVER_PROTOCOL variable to determine the appropriate format for a response.

Servers SHOULD attempt to ensure that the script output is sent directly to the client, with minimal internal and no transport-visible buffering.

8.16.2 Parsed Header Output

Scripts using the parsed header output form MUST supply a CGI response message to the server as follows:

CGI-Response = *optional-field CGI-Field *optional-field NL [Message-Body]

optional-field = (CGI-Field | HTTP-Field)

CGI-Field = Content-type

| Location

| Status

| extension-header

The response comprises a header and a body, separated by a blank line. The body may be NULL. The header fields are either CGI header fields to be interpreted by the server, or HTTP header fields to be included in the response returned to the client if the request method is HTTP. At least one CGI-Field MUST be supplied, but no CGI field name may be used more than once in a response. If a body is supplied, then a "Content-type" header field MUST be supplied by the script, otherwise the script MUST send a "Location" or "Status" header field. If a Location CGI-Field is returned, then the script MUST NOT supply any HTTP-Fields.

Each header field in a CGI-Response MUST be specified on a single line; CGI/1.1 does not support continuation lines.

CGI header fields

The CGI header fields have the generic syntax:

generic-field = field-name ":" [field-value] NL

field-name = token

field-value = *(field-content | LWSP)

field-content = *(token | tspecial | quoted-string)

The field-name is not case sensitive; a NULL field value is equivalent to the header field not being sent

1. Content-Type. The Internet Media Type of the entity body, which is to be sent unmodified to the client.

   Content-Type = "Content-Type" ":" media-type NL

   This is actually an HTTP-Field rather than a CGI-Field, but it is listed here because of its importance in the CGI dialogue as a member of the "one of these is required" set of header fields.

2. Location. This is used to specify to the server that the script is returning a reference to a document rather than an actual document.

Location	=	"Location" ":" (fragment-URI | rel-URL-abs-path) NL
fragment-URI	=	URI [# fragmented]
URI	=	scheme ":" *qchar
fragmented	=	*qchar
rel-URL-abs-path	=	"/" [hpath] ["?" query-string]
hpath	=	fpsegment *("/" psegment)
fpsegment	=	1*hchar
psegment	=	*hchar
hchar	=	alpha | digit | safe | extra | ":" | "@" | "&" | "="

The Location value is either an absolute URI with optional fragment, or an absolute path within the server's URI space (*i.e.*, omitting the scheme and network-related fields) and optional query-string. If an absolute URI is returned by the script, then the server MUST generate a '302 redirect' HTTP response message unless the script has supplied an explicit Status response header field. Scripts returning an absolute URI MAY choose to provide a message-body. Servers MUST make any appropriate modifications to the script's output to ensure the response to the user-agent complies with the response protocol version. If the Location value is a path, then the server MUST generate the response that it would have produced in response to a request containing the URL

scheme "://" SERVER_NAME ":" SERVER_PORT rel-URL-abs-path

Note: If the request was accompanied by a message-body (such as for a POST request), and the script redirects the request with a Location field, the message-body may not be available to the resource that is the target of the redirect.

3. Status. The "Status" header field is used to indicate to the server what status code the server MUST use in the response message.

   Status = "Status" ":" digit digit digit SP reason-phrase NL

   reason-phrase = *<CHAR, excluding CTLs, NL>

   If the SERVER_PROTOCOL is "HTTP/1.1", then the status codes defined in the HTTP/1.1 If the script does not return a "Status" header field, then "200 OK" SHOULD be assumed by the server.

   If a script is being used to handle a particular error or condition encountered by the server, such as a '404 Not Found' error, the script SHOULD use the "Status" CGI header field to propagate the error condition back to the client. *E.g.*, in the example mentioned it SHOULD include a "Status: 404 Not Found" in the header data returned to the server.

4. Extension header fields. Scripts MAY include in their CGI response header additional fields not defined in this or the HTTP specification. These are called "extension" fields, and have the syntax of a generic-field. The name of an extension field MUST NOT conflict with a field name defined in this or any other specification; extension field names SHOULD begin with "X-CGI-" to ensure uniqueness

8.17 Client-side Programming

8.17.1 JavaScript

JavaScript is a powerful scripting language that can be embedded directly in HTML. It allows you to create dynamic, interactive Web-based applications that run completely within a Web browser; you don't have to do any server-side programming, like writing CGI scripts. It is currently being standardised under the name *ECMA Script*.

JavaScript is a simpler language than Java. It can be embedded directly in web pages without compilation, so it is more flexible and easier to use for simple tasks like animation. However, although

you can write reasonably robust and complete Web applications using JavaScript alone, JavaScript is not a substitute for Java.

In fact, JavaScript is a good client-side complement to Java; using the two together allows you to create more complex applications than are possible with JavaScript alone.

Figure 8.11 is a listing of Mick's Today program.

```
<HTML>
<HEAD>
<TITLE>Today (Javascript)</TITLE>
<META name="author"
    content="Mick Farmer">
<META name="keywords"
    content="javascript">
<LINK href="/~mick/mick.css"
    rel="stylesheet"
    type="text/css">
<LINK href="../networks.css"
    rel="stylesheet"
    type="text/css">
</HEAD>
<BODY class="chapter">
<H1>Today's Date</H1>
<P>
<SCRIPT language="javascript">
 var day = new Array(7);
 day[0] = "Sunday";
 day[1] = "Monday";
 day[2] = "Tuesday";
 day[3] = "Wednesday";
 day[4] = "Thursday";
 day[5] = "Friday";
 day[6] = "Saturday";
 var month = new Array(12);
 month[0] = "January";
 month[1] = "February";
 month[2] = "March";
 month[3] = "April";
 month[4] = "May";
 month[5] = "June";
 month[6] = "July";
 month[7] = "August";
 month[8] = "September";
 month[9] = "October";
 month[10] = "November";
 month[11] = "December";
 var now = new Date();
 document.write("Today is " + day[now.getDay()]
          + ", " + now.getDate()
          + " " + month[now.getMonth()]
          + " " + now.getYear()
          + ".");
</SCRIPT>
</P>
</BODY>
</HTML>
```

Figure 8.11: Today or Whatever

The <SCRIPT> element is replaced by whatever was generated by the document.write(...) function call. Interestingly, Netscape Navigator (on UNIX) returned tomorrow's day and day of the month, and 101 for the year!

8.17.2 Java Applets

These are Java programs embedded in an HTML document. They identify themselves through code signing, i.e. using a digital signature.

What Applets Can't Do

- Reference specified area of memory
- Access local file system
- Execute other programs
- Load local libraries
- Create or manage threads in others
- Connect to other hosts except originator
- Open windows

Example

Figure 8.12 is a listing of the Java source file of Mick's Today Program applet.

```
import java.applet.Applet;
import java.awt.*;
import java.util.Date;
import java.text.SimpleDateFormat;

public class today extends Applet {
        SimpleDateFormat sdf;

        public void init(){
                sdf = new SimpleDateFormat("EEEE, d MMMM yyyy");
        }

        public void paint(Graphics g){
                g.drawString("Today is " + sdf.format(new Date()), 20, 40);
        }
}
```

Figure 8.12: Java Source for Today Applet

Most of the work is performed in the formatting of the date. This Java source file today.java is then compiled to produce the class file today.class containing the applet.

Figure 8.13 lists the HTML page containing the embedded applet.

```
<HTML>
<HEAD>
<TITLE>Today (Java Applet)</TITLE>
</HEAD>
<BODY>
<H1>Today's Date</H1>
```

```
<APPLET CODE="today.class"
    WIDTH=300
    HEIGHT=100
    CODEBASE=.>
</APPLET>
</BODY>
</HTML>
```

Figure 8.13: Today Applet

The applet runs with a *canvas* 300 pixels wide by 100 pixels high. The applet is found in the same directory as the HTML page. The text produced by running the applet is displayed in the canvas by the browser.

8.18 Other Scripting Languages

Perl

Although commonly used for server-side scripting, Perl is available for client-side work through a subset called Perlscript. Brief overview of Perl is given in next section of this unit

Python

Python is becoming popular. A web browser,has been written in Python, which is capable of running Python applets.

Tcl

Tcl is a popular choice for systems programming. A Tcl plug-in and some demonstration programs are available on the Internet.

VBScript

VBScript is a widely used language but unfortunately platform specific. It is only available under the Microsoft Windows operating system and the applications will only run inside Internet Explorer.

8.19 Brief Overview of Perl

Perl *(practical extraction and report language)* has captured the hearts and minds of computer users everywhere. So much, so, in fact, that some people go so far as to write poetry in Perl, taking advantage of Perl's confusing syntax.

Perl is a very flexible scripting language. Originally designed for UNIX system administration tasks, Perl has branched out to tens of thousands of users and millions of uses.

8.19.1 Uses of Perl

Some of the main uses include:

- CGI, or Common Gateway Interface, scripts for dynamic Web pages.
- Generating reports on system resource usage.
- Interfacing with databases.
- Cross-platform software development, such as Perl scripts to automate C and C++ build procedures.
- Client-server applications.
- System administration on UNIX and Windows.

There's more than one way to get things done in Perl, as the infamous motto states, which forms one of Perl's great strengths and also one of its greatest weaknesses.

Perl is strong in text-handling and calling system functions.

Perl is definitely weak in its cryptic syntax and use of just about every punctuation mark as syntax, especially the global variables like $_ and @_.

Perl is free and available over the Internet.

One of the best parts about Perl is that it runs on just about every version of UNIX as well as Windows 95 and NT. System administrators make great use of Perl to automate their daily tasks.

8.19.2 Perl and CGI

Perl is the language most commonly used for CGI. Perl stands for Practical Extraction and Reporting Language. It is called such for its built-in utilities for scanning files and processing their content.

CGI stands for Common Gateway Interface and is a method for creating web based documents rather than a specific programming language. Other languages may be used to create CGI like **TCL or Tk**.

Perl is the easiest to learn, the most popular and the most widely supported. Standalone Perl scripts end in the file extension **.pl**, web-based CGI scripts end in the file extension **.cgi**. They are both written in Perl, but what they do and where they are placed differ.

HTML is wonderful, but it is *stateless*, meaning that it cannot accept input or maintain input and does not have normal programming functions like looping. HTML was designed for the purpose of opening and viewing files and little else.

Java and **JavaScript** have added much power to HTML, but neither have the ease and functionality of Perl/CGI. CGI makes on-line shopping possible.

Cookies and **Shopping Carts** are Perl/CGI scripts; they drive web commerce and make information transfer from platform to platform simpler.

8.20 Running Perl

Perl is a scripting language, so you don't have to compile and link. Instead, you execute the Perl interpreter, **perl**, with your script file. Perk scripts are stored in text files, and you can use your editor of choice to create your Perl scripts, be it **emacs**, **vi**, **Write** or **Word**.

To execute a Perl script, pass the script file name to the **perl** interpreter, for example:

perl hello.pl

Invoking Perl on UNIX

UNIX systems have another way to invoke an interpreter on a script file. Place a line like

```
#!/usr/local/bin/perl
```

at the start of the Perl file. This tells UNIX that the rest of this script file is to be interpreted by /usr/local/bin/perl. Then make the script itself executable:

```
chmod +x sample.pl
```

You can then "execute" the script file directly and let the script file tell the operating system what interpreter to use while running it.

Invoking Perl on Windows NT

Windows NT, on the other hand, is quite different. You can use File Manager (Explorer under Windows NT 4 or Windows 95) to create an association between the file extension .PL and the Perl executable. Whenever a file ending in .PL is invoked, Windows will know that Perl should be used to interpret it

8.21 Perl Command-Line Arguments

Perl takes a number of optional command-line arguments for various purposes. These are listed in Table 8.2. Most are rarely used but are given here for reference purposes.

Table 8.2: Perl 5 Command-Line Switches

Option	*Arguments*	*Purpose*	*Notes*
-0	Octal character code	Specify record separator	Default is newline (\n)
-a	None	Automatically spli recordst	Used with -n or or -p
-c	None	Check syntax only	Do not execute
-d	None	Run script using Perl debugger	If Perl debugging option was included when Perl was installed
-D	flags	Specify debugging behavior	See table 2
-e	command	Pass a command to Perl from the command line	Useful for quick operations
-F	regular expression	If -a used	Expression to split by default is white space
-i	extension	Replace original file with results	Useful for modifying contents of files
-I	directory	Specify location of include files	
-l	octal character code	Drop newlines when used	With -n and -p and use designated character as line-termination character
-n	none	Process the script using each specified file as an argument	Used for performing the same set of actions on a set of files
-p	none	Same as -n but each line is printed	
-P	none	Run the script through the C preprocessor before Perl compiles it	
-s	none	Enable passing of arbitrary switches to Perl	Use -s -what -ever to have the Perl variables $what and $ever defined within your script
-S	none	Tell Perl to look along the path for the script	
-T	none	Use taint checking; don't evaluate expressions supplied on the command line	
-u	none	Make Perl dump core after compiling your script; intended to allow for generation of Perl executables	Very messy; wait for the Perl compiler
-U	none	Unsafe mode; overrides Perl's natural caution	Don't use this!
-v	none	Print Perl version number	
-w	none	Print warnings about script syntax	Extremely useful, especially during development

You can supply Perl command-line arguments on the interpreter invocation line in UNIX scripts. The following line is a good start to any Perl script:

```
#!/usr/local/bin/perl -w -t
```

Table 2 shows the debug flags, which can be specified with the -D command-line option. If you specify a number, you can simply add all the numbers of each flag together so that 6 is 4 and 2. If you use the letter as a flag then simply list all the options required. The following two calls are equivalent:

```
#perl -d -D6 test.pl
#perl -d -Dls test.pl
```

Table 8.3: Perl Debugging Flags

Flag Number	Flag Letter	Meaning of Flag
1	P	Tokenizing and parsing
2	S	Stack snapshots
4	L	Label stack processing
8	T	Trace execution
16	O	Operator node construction
32	C	String/numeric conversions
64	P	Print preprocessor command for -P
128	M	Memory allocation
256	F	Format processing
512	R	Regular expression parsing
1024	X	Syntax tree dump
2048	U	Tainting checks
4096	L	Memory leaks (not supported anymore)
8192	H	Hash dump; usurps values()
6384	X	Scratchpad allocation (Perl 5 only)
32768	D	Cleaning up (Perl 5 only)

8.22 Perl Script

A Perl program consists of an ordinary text file containing a series of Perl commands. Commands are written in what looks like a silly amalgam of C, shell script, and English. In fact, that's pretty much what it is.

Perl code can be quite free-flowing. The broad syntactic rules governing where a statement starts and ends are:

- Leading white space is ignored. You can start a Perl statement anywhere you want: at the beginning of the line, indented for clarity (recommended), or even right-justified (definitely frowned on) if you like.
- Commands are terminated with a semicolon.
- White space outside of string literals is irrelevant; one space is as good as a hundred. That means you can split statements over several lines for clarity.
- Anything after a pound sign (#) is ignored. Use this to pepper your code with useful comments.

8.22.1 Data Types

Perl has a small number of data types. If you're used to working with C, where even characters can be either signed or unsigned, this makes a pleasant change. In essence, there are only two data types:

- *Scalars* and
- *Arrays*.

There is also a very special kind of array called an *associative array* that merits a section all to itself.

Scalars

All numbers and strings are scalars. Scalar variable names start with a dollar sign. All Perl variable names, including scalars, are case sensitive.

$Name and $name, for example, are two completely different quantities.

8.22.2 Arrays

A collection of scalars is an array. An array variable name starts with an @ sign, while an explicit array of scalars is written as a comma-separated list within parentheses:

```
@trees = ("Larch", "Hazel", "Oak");
```

Array subscripts are denoted using square brackets: $trees[0] is the first element of the @trees array. Notice that it's @trees but $trees[0]; individual array elements are scalars, so they start with a $.

Mixing scalar types in an array is not a problem. For example,

```
@items = (15, 45.67, "case");
print "Take $items[0] $items[2]s at \$$items[1] each.\n";
```

results in

```
Take 15 cases at $45.67 each.
```

All arrays in Perl are dynamic. Memory allocation and management is managed by Perl itself. Combine that with the fact that arrays can contain arrays as sub-arrays, and you're free to say things like the following:

```
@A = (1, 2, 3);
@B = (4, 5, 6);
@C = (7, 8, 9);
@D = (@A, @B, @C);
```

which results in the array @D containing numbers 1 through 9.

8.22.3 Associative Arrays

There is a certain elegance to associative arrays that makes experienced Perl programmers a little snobbish about their language of choice. Associative arrays give Perl a degree of database functionality at a very low yet useful level. Many tasks that would otherwise involve complex programming can be reduced to a handful of Perl statements using associative arrays.

Arrays of the type we've already seen are *lists of values indexed by subscripts*. In other words, to get an individual element of an array, you supply a subscript as a reference:

```
@fruit = ("Apple", "Orange", "Banana");
print $fruit[2];
```

This example yields Banana because subscripts start at 0, so 2 is the subscript for the third element of the @fruit array. A reference to $fruit[7] here returns the null value, as no array element with that subscript has been defined.

Now, here's the point of all this: associative arrays are lists of values indexed by strings. Their implementation is more complex, obviously, as all of the strings need to be stored in addition to the values to which they refer.

When you want to refer to an element of an associative array, you supply a string (also called the key) instead of an integer (also called the subscript). Perl returns the corresponding value. Consider the following example:

```
%fruit = ("Green", "Apple", "Orange", "Orange", "Yellow", "Banana");
print $fruit{"Yellow"};
```

This prints Banana as before. The first line defines the associative array in much the same way we have already defined ordinary arrays; the difference is that instead of listing values, we list *key-value pairs*. The first value is Apple and its key is Green; the second value is Orange, which happens to have the same string for both value and key; and the final value is Banana and its key is Yellow.

8.22.4 File Handle

Really this is not a data type but a special kind of literal string. A file handle behaves in many ways like a variable, however, so this is a good time to cover them. Besides, you won't get very far in Perl without them.

A file handle can be regarded as a pointer to a file from which Perl is to read or to which it will write. Most programmers will be familiar with the concept. The basic idea is that you associate a file handle with a file or device and then refer to the handle in the code whenever you need to perform a read or write operation.

File handles are generally written in all uppercase. Perl has some useful predefined file handles, which are listed in Table 8.4.

Table 8.4: Perl's Predefined File Handles

File Handle	Points To
STDIN	Standard input, normally the keyboard.
STDOUT	Standard output, normally the console.
STDERR	Device where error messages should be written, normally the console.

The print statement can take a file handle as its first argument:

```
print STDERR "Oops, something broke.\n";
```

Note that there is no comma after the file handle, which helps Perl to figure out that the STDERR is not something to be printed. If you're uneasy with this implicit list syntax, you can put parentheses around all of the print arguments:

```
print (STDERR "Oops, something broke.\n");
```

The open function may be used to associate a new file handle with a file:

```
open (INDATA, "/etc/stuff/Friday.dat");

open (LOGFILE, ">/etc/logs/reclaim.log");

print LOGFILE "Log of reclaim procedure\n";
```

By default, open opens files for reading only. If you want to override this default behavior, add one of the special direction symbols from Table 8.5 to the file name. That's what the > at the start of the file name in the second output statement is for; it tells Perl that we intend to write to the named file.

Table 8.5: Perl File Access Symbols

Symbol	Meaning
<	Opens the file for reading. This is the default action.
>	Opens the file for writing.
>>	Opens the file for appending.
+<	Opens the file for both reading and writing.
+>	Opens the file for both reading and writing.
| (before file name)	Treat file as command into which Perl is to pipe text.
| (after file name)	Treat file as command from which input is to be piped to Perl.

8.22.5 Flow Control

Perl has all of the flow control mechanisms you'd expect to find in a high-level language, and this section takes you through the basics of each.

Logical Operators

The two most important logical operators are: the || (or) and && (and) operators. They take two operands and return either True or False depending on the operands:

Conditional Expressions

The basic kind of flow control is a simple branch: A statement is either executed or not depending on whether a logical expression is True or False. This can be done by following the statement with a modifier and a logical expression:

```
open ( INFILE, "./missing.txt") if $missing;
```

The execution of the statement is contingent upon both the evaluation of the expression and the sense of the operator.

The expression evaluates as either True or False and can contain any of the relational operators listed in Table 8.6, although it doesn't have to. Examples of valid expressions are

```
$full
$a == $b
<STDIN>
```

Table 8.6: Perl's Relational Operators

Operator	Numeric Context	String Context
Equality	==	eq
Inequality	!=	ne
Inequality with signedresult	<=>	cmp
Greater than	>	gt
Greater than or equal to	>=	ge
Less than	<	lt
Less than or equal to	<=	le

There are four modifiers, each of which behaves the way you might expect from the corresponding English word:

- if

  The statement executes if the logical expression is True and does not execute otherwise. Examples:

  ```
  $max = 100 if $min < 100;
  print "Empty!\n" if !$full;
  ```

- unless

  The statement does not execute if the logical expression is True and executes otherwise. Examples:

  ```
  open (ERRLOG, "test.log") unless $NoLog;
  print "Success" unless $error>2;
  ```

- while

  The statement executes repeatedly until the logical expression is False. Examples:

  ```
  $total -= $decrement while $total > $decrement;
  $n=1000;  "print $n\n" while $n- > 0;
  ```

- until

  The statement executes repeatedly until the logical expression is True. Examples:

  ```
  $total += $value[$count++] until $total > $limit;
  print RESULTS "Next value: $value[$n++]" until  $value[$n] = -1;
  ```

Note that the logical expression is evaluated once only in the case of if and unless but multiple times in the case of while and until. In other words, the first two are simple conditionals, while the last two are loop constructs.

8.22.6 Compound Statements

The syntax changes when we want to make the execution of multiple statements contingent on the evaluation of a logical expression. The modifier comes at the start of a line, followed by the logical expression in parentheses, followed by the conditional statements contained in braces. Note that the parentheses around the logical expression are required, unlike with the single statement branching described in the previous section.

For example,

```
if ( ( $total += $value ) > $limit ) {
  print LOGFILE "Maximum limit $limit exceeded.",
" Offending value was $value.\n";
close (LOGFILE);
 die "Too many! Check the log file for details.\n";
  }
```

This is somewhat similar to C's if syntax, except that the braces around the conditional statement blocks are required rather than optional.

The if statement is capable of a little more complexity, with else and elseif operators:

```
if ( !open( LOGFILE, "install.log") )  {
  close ( INFILE );
  die "Unable to open log file!\n";
  }
elseif ( !open( CFGFILE, ">system.cfg") ) {
  print LOGFILE "Error during install:",
" Unable to open config file for writing.\n";
close ( LOGFILE );
  die "Unable to open config file for writing!\n";
  }
else {
  print CFGFILE "Your settings go here!\n";
  }
```

8.22.7 Loops

The loop modifiers (while, until, for, and foreach) are used with compound statements in much the same way:

```
until ( $total >= 50 ) {
  print "Enter a value: ";
  $value = scalar (<STDIN>);
  $total += $value;
  print "Current total is $total\n";
  }
print "Enough!\n";
```

The while and until statements were described in the earlier "Conditional Expressions" section.

The for statement resembles the one in C: It is followed by an initial value, a termination condition, and an iteration expression, all enclosed in parentheses and separated by semicolons:

```
for ( $count = 0; $count < 100; $count++ ) {
  print "Something";
  }
```

The foreach operator is special. It iterates over the contents of an array and executes the statements in a statement block for each element of the array.

A simple example is the following:

```
@numbers = ("one", "two", "three", "four");
foreach $num ( @numbers ) {
  print "Number $num...\n";
  }
```

The variable $num first takes on the value one, then two, and so on. That example looks fairly trivial, but the real power of this operator lies in the fact that it can operate on any array:

```
foreach $arg ( @ARGV ) {
  print "Argument: \"$arg\".\n";
  }
foreach $namekey ( sort keys %surnames ) {
  print REPORT "Surname: $value{$namekey}.\n",
        "Address: $address{$namekey}.\n";
  }
```

8.22.8 Labels

Labels may be used with the next, last, and redo statements to provide more control over program flow through loops. A label consists of any word, usually in uppercase, followed by a colon.

The label appears just before the loop operator (while, for, or foreach) and can be used as an anchor for jumping to from within the block:

```
RECORD: while ( <INFILE> ) {
  $even = !$even;
  next RECORD if $even;
  print;
  }
```

That code snippet prints all the odd-numbered records in INFILE.

The three label control statements are

- next
  - Jumps to the next iteration of the loop marked by the label or to the innermost enclosing loop if no label is specified.
- last

- Immediately breaks out of the loop marked by the label or out of the innermost enclosing loop if no label is specified.

- redo
  - Jumps back to the loop marked by the specified label or to the innermost enclosing loop if no label is specified. This causes the loop to execute again with the same iterator value.

8.22.9 Subroutines

The basic subunit of code in Perl is a subroutine. This is similar to a function in C and a procedure or a function in Pascal. A subroutine may be called with various parameters and returns a value. Effectively, the subroutine groups together a sequence of statements so that they can be re-used.

The Simplest Form of Subroutine

Subroutines can be declared anywhere in a program. If more than one subroutine with the same name is declared each new version replaces the older ones so that only the last one is effective. It is possible to declare subroutines within an eval() expression, these will not actually be declared until the runtime execution reaches the eval() statement.

Subroutines are declared using the following syntax:

```
sub subroutine-name {
        statements
}
```

The simplest form of subroutine is one that does not return any value and does not access any external values. The subroutine is called by prefixing the name with the & character. (There are other ways of calling subroutines, which are explained in more detail later.) An example of a program using the simplest form of subroutine illustrates this:

```
#!/usr/bin/perl -w
# Example of subroutine which does not use
# external values and does not return a value
&egsub1; # Call the subroutine once
&egsub1; # Call the subroutine a second time
sub egsub1 {
   print "This subroutine simply prints this line.\n";
}
```

Returning Values from Subroutines

Subroutines can also return values, thus acting as functions. The return value is the value of the last statement executed. This can be a scalar or an array value.

It is possible to test whether the calling context requires an array or a scalar value using the wantarra construct, thus returning different values depending on the required context.

For example,

```
wantarray ? (a, b, c) : 0;
```

as the last line of a subroutine returns the array (a, b, c) in an array context, and the scalar value 0 in a scalar context.

```
#!/usr/bin/perl -w
# Example of subroutine which does not use
# external values but does return a value
# Call the subroutine once, returning a scalar #value
$scalar-return = &egsub2;
print "Scalar return value: $scalar-return.\n";
# Call the subroutine a second time, returning an #array value
@array-return = &egsub2;
print "Array return value:", @array-return, ".\n";
sub egsub2 {
   print "This subroutine prints this line and returns a value.\n";
   wantarray ? (a, b, c) : 0;
}
```

Passing Values to Subroutines

The next important aspect of subroutines, is that the call can pass values to the subroutine. The call simply lists the variables to be passed, and these are passed in the list @_ to the subroutine. These are known as the parameters or the arguments.

It is customary to assign each value a name at the start of the subroutine so that it is clear what is going on. Manipulation of these copies of the arguments is equivalent to passing arguments by value (that is, their values may be altered but this does not alter the value of the variable in the calling program).

```
#!/usr/bin/perl -w
# Example of subroutine is passed external values by #value
$returnval = &egsub5(45,3); # Call the subroutine once
print "The (45+1) * (3+1) is $returnval.\n";
$x = 45;
$y = 3;
$returnval = &egsub5($x,$y);
print "The ($x+1) * ($y+1) is $returnval.\n";
print "Note that \$x still is $x, and \$y still is $y.\n";
sub egsub5 { # Access $x and $y by value
local($x, $y) = @_;
return ($x++ * $y++);
}
```

To pass scalar values by reference, rather than by value, the elements in @_ can be accessed directly. This will change their values in the calling program. In such a case, the argument must be a variable rather than a literal value, as literal values cannot be altered.

```
#!/usr/bin/perl -w
# Example of subroutine is passed external values by #reference
$x = 45;
$y = 3;
print "The ($x+1) * ($y+1) ";
$returnval = &egsub6($x,$y);
print "is $returnval.\n";
print "Note that \$x now is $x, and \$y now is $y.\n";
sub egsub6 { # Access $x and $y by reference
   return ($_[0]++ * $_[0]++);
}
```

Subroutine Recursion

One the most powerful features of subroutines is their ability to call themselves. There are many problems that can be solved by repeated application of the same procedure. However, care must be taken to set up a termination condition where the recursion stops and the execution can unravel itself. Typical examples of this approach are found when processing lists: Process the head item and then process the tail; if the tail is empty do not recurse. Another neat example is the calculation of a factorial value

```
#!/usr/bin/perl -w
#
# Example factorial using recursion

for ($x=1; $x<100; $x++) {
      print "Factorial $x is ",&factorial($x), "\n";
}

sub factorial {
      local($x) = @_;
      if ($x == 1) {
            return 1;
      }
      else {
            return ($x*($x-1) + &factorial($x-1));
      }
}
```

Scope with my() and local()

Issues of scope are very important with relation to subroutines. In particular all variables inside subroutines should be made lexical local variables (using my()) or dynamic local variables (using local()). In Perl 4, the only choice is local() because my() was only introduced in Perl 5.

Variables declared using the my() construct are considered to be lexical local variables. They are not entered in the symbol table for the current package. Therefore, they are totally hidden from all contexts other than the local block within which they are declared. Even subroutines called from the current block cannot access lexical local variables in that block. Lexical local variables must begin with an alphanumeric character or an underscore.

Variables declared using the local() construct are considered to be dynamic local variables. The value is local to the current block and any calls from that block. It is possible to localize special variables as dynamic local variables, but these cannot be made into lexical local variables. The following two differences from lexical local variables show the two cases in Perl 5 where it is still advisable to use local() rather than my():

- Use local() if you want the value of the local variables to be visible to subroutines
- Use local() if you are localizing special variables

Student Activity

1. What do you understand by server-side programming?
2. What do you understand by CGI?
3. What is the structure of CGI script?
4. Outline the capabilities of Perl.
5. What are the steps used in CGI script?
6. What are the header and the body section in CGI application?.
7. What are three important ways to give input from a client to a CGI application?
8. How CGI applications can be created?
9. What are the Security issues in CGI?
10. Write a CGI and PERL program to print Welcome to the world of CGI in web page is the user inputs all the fields in the form.