UNIT 4 INTERNETWORKING: CONCEPT, ARCHITECTURE AND PROTOCOLS

Structure **Page Nos.**

4.0 INTRODUCTION

An internetwork is a collection of packet-switching and broadcast networks, connected by bridges, switches, or routers which are intermediate networking devices, that functions as a single large network. So all users and devices can communicate, regardless of the network segment to which they are attached. *Figure 1* illustrates some different kinds of network technologies that can be interconnected by routers and other networking devices to create an internetwork.

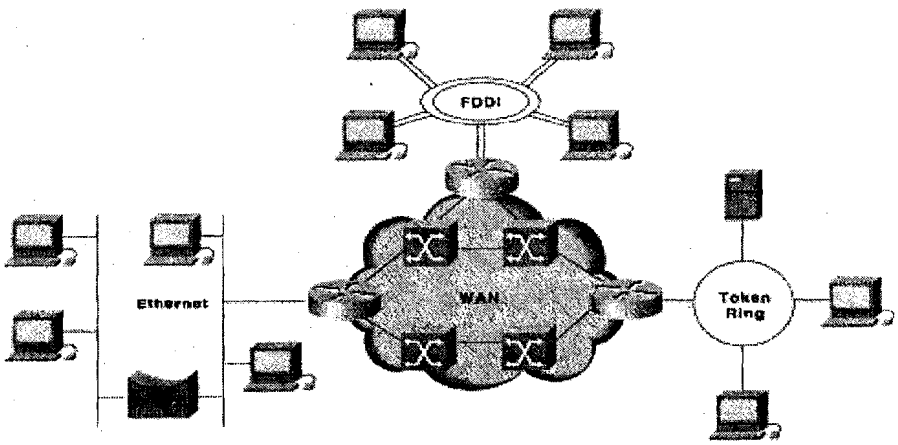

Figure 1: Internetworking

In this unit we will explain the concepts, architecture and protocols of internetworking.

4.1 OBJECTIVES

After going through this unit you should be able to:

- define packet switching concept;
- differentiate between virtual circuit & datagram, and
- understand the functioning of a large number of protocols.

4.2 HISTORY OF INTERNETWORKING

Networking allows computers to share information, applications and even hardware devices. For any two computers to communicate with each other , they must follow a set of rules called protocol. In early years of networking, every vendor was concerned only with his own product. Each vendor defined its own standard for communication between its computers. This resulted in many different types of standards for network hardware and software that did not share a common protocol. This meant that different systems could not interact with each other .Two computers using network products of different vendors could not be connected together.

In the late 1970s, the networking community came together in an effort to replace these closed systems with open systems. They wanted that all networking products should be compatible with other vendor products. The International Standards Organisation (ISO) developed Open Systems Interconnection (OSI) reference model for networking the model gives guidelines about how the parts of a network communication system should work together.

The Open System Interconnection (OSI) reference model describes how information from a software application in one computer moves through a network medium to a software application in another computer. The OSI reference model is a conceptual model composed of seven layers, each specifying particular network functions. The model was developed by the International Standards Organisation (ISO) in 1984, and it is now considered the primary architectural model for inter computer communications. The OSI model divides the tasks involved with moving information between networked computers into seven smaller, more manageable task groups. A task or group of tasks is then assigned to each of the seven OSI layers. Each layer is reasonably self-contained so that the tasks assigned to each layer can be implemented independently. This enables the solutions offered by one layer to be updated without adversely affecting the other layers. The following list details the seven layers of the Open System Interconnection (OSI) reference model:

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

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

Figure 2 : The OSI Reference Model Contains Seven Independent Layers

4.3 PACKET SWITCHING

Data communication takes place between two devices that are directly connected through some form of transmission medium. It is impractical for two devices to be connected directly. Instead, a network of switching nodes provides a transfer path between two devices. Packet switching involves the breaking up of messages into smaller components called packets. Packets often range in size from about 128 bytes to over 4096 bytes depending on the system involved. Each packet contains source and destination information, and is treated as an individual message. These mini-messages are received and routed through optimal routes by various nodes on a wide area network. For example a file to be transmitted between two machines may be broken into many packets that are sent across the network one at a time. The network H/W delivers the packet to the end where the network software reassembles them into a single file again. There are two major types of packets to be switched. They are **datagram** and **virtual circuit.**

In the **datagram approach,** each packet is treated independently and may follow a different path through the network. Packets may be re-ordered, dropped or delivered in wrong sequence. The communication protocols will have to provide error recovery and sequencing of packets at the destination.

In the **virtual circuit** approach, a fixed logical path through the network from sender to destination is established before any packets are sent. This path remains unchanged for the duration of the connection or session. Although no resources are reserved along the path, packets are buffered at intermediate nodes awaiting transmission.

Thus, a virtual circuit only defines a path for packets to follow without actually reserving dedicated channels along the route as is the case with circuit switching. Virtual circuit may provide a number of services including sequencing, error control and **flow control.**

In comparing datagram and virtual circuit switching with other switching technologies, there are several factors to be considered. First of all, packet switching is faster because messages are not stored in their entirety for later retrieval. Each packet is small enough to be stored in a router's machine memory until it can be routed an instant later. Secondly, packet switching allows the avoidance of route failure due to excessive traffic loads. This is accomplished by routing packets along routes that are the most free and clear. Thirdly, packet switching spreads the load of communication across several paths.

The motivation for adopting packet switching are cost and performance. Because multiple machines can share network fewer interconnections are required and cost is kept low. Packet switched networks that span large geographical distances are fundamentally different from those that span short distances. To help characterize the differences in capacity and intended use, packet switched technologies are often divided into three broad categories: Wide Area Network (WAN), Metropolitan Area Network (MAN) and Local Area Network. There are structural protocols for each category. In the next section we will examine one such protocol.

4.4 INTERNETWORKING CONCEPTS

We have seen how machines connect to individual networks. The question arises, "How are networks interconnected to form an internetwork?" The answer has two parts. Physically, two networks can only be connected by a computer that attaches to both of them. A physical attachment does not provide the interconnection we have in mind, however, because such a connection does not guarantee that the computer will cooperate with other machines that wish to communicate. To have a viable internet, we need computers that are willing to shuffle packets from one network to another. Computers that interconnect two networks and pass packets from one to the other are called internet gateways+ or internet routers.

Consider an example consisting of two physical networks shown in *Figure 3*. In the figure, machine G connects to both network 1 and network 2. For G to act as a gateway, it must capture packets on network 1 that are bound for machines on network 2, and packets on network 2 that are destined for machines on network 1 and transfer them.

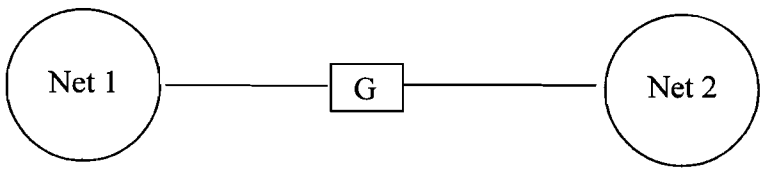

Figure 3: Two networks interconnected by G, a gateway (router)

Interconnection through IP Gateways or Routers

When internet connections become more complex , gateways need to know about the topology of the internet beyond the networks to which they connect. For example, Figure 4 shows three networks interconnected by two gateways.

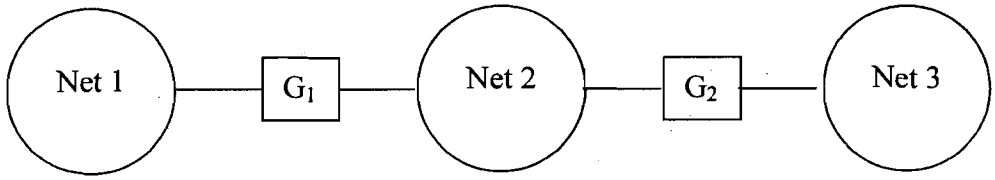

Figure 4: Three networks interconnected by two gateways

In this example, gateway G must move from network 1 to network 2 all packets destined for machines on either network 2 or network 3. As the size of the internet expands, the gateway's task of making decisions about where to send packets becomes more complex.

The idea of a gateway seems simple, but it is important because it provides a way to interconnect networks, not just machines. In fact, we have already discovered the principle of interconnection used throughout an internet:

In a TCP/IP internet, computers called gateways provide all interconnections among physical networks.

You might suspect that gateways, which must know how to route packets to their destination, are large machines with enough primary or secondary memory to hold information about every machine in the internet to which they attach. However, gateways used with TCP/IP internets are usually minicomputers; they often have little or no disk storage and limited main memories. The trick to building a small internet gateway lies in the following concept:

Gateways route packets based on destination network, not on destination host.

If routing is based on networks, the amount of information that a gateway needs to keep is proportional to the number of networks in the internet, not the number of machines.

4.5 INTERNET ADDRESSES

In the previous section we defined a TCP/IP internet as a virtual network built by interconnecting several physical networks through gateways which are usually controlled by the internet service providers think of internet as a large network like any other network. The difference of course is that the internet is a virtual structure imagined by its designers and implemented entirely in TCP/IP s/w. Each machine (host) on an internet is assigned a unique 32 bit internet address that is used in all communication with that machine. Conceptually each address is a paid (netid, hosted) where **netid** identifies a network and hosted identifies a host on that network. Because IP addresses comprise both a network as a host on that network they do not specifiy an individual machine but a connection to a network. For example a getway connecting networks has distinct IP address, one for each network connection.

Unicasting, Broadcasting, and Multicasting

When an IP datagram is sent to an individual IP address, it is called a unicast IP datagram. The process of sending the datagram is called unicasting. Unicasting is used when two IP nodes are communicating with each other.

When an IP datagram is sent to all nodes on a specific network, it is called broadcasting.

There is a third mode of sending an IP datagram called multicasting. In multicasting the IP datagram is delivered to a group of systems identified by a class D address. The systems that have the same multicast address are said to belong to a multicast group. Members of the multicast group, while being assigned a class D address, must also be assigned an IP address (from the class A, B, or C address group). The multicast group can receive an IP datagram in two ways:

- IP datagrams sent directly to their individual IP address (class A, B, C).
- IP datagrams sent to their multicast address (class D).

4.6 CONFIGURING IP ADDRESSES

Hosts on a TCP/IP network need to be configured using proper IP addresses. The actual configuration procedure depends on the operating system. The configuration procedure can be classified into the following categories.

- Command line based
- Menu interface
- Graphical User Interface based
- Obtained dynamically when host boots from a central server

Table 1: Configuring IP Address and other Parameters for Hosts

Method	Operating System/ Protocol
Command line	Unix, VMS, Router, devices, and MS-DOS
Menu interface	Router devices, Unix, NetWare servers, and MS-DOS
Graphical User Interface	Microsoft Windows products.
Dynamically assigned	DHCP and BOOTP protocols. Available on almost all major operating system platforms.

Many systems offer a command that can be executed to modify the IP address. In these systems, the command line is often placed in the startup script for the operating system. The menu interface is built using extended line drawing character sets. You are prompted to enter the IP address and other IP parameters. The Graphical User Interface is for systems — such as X-windows and Microsoft's Windows operating systems—that offer a pixel-based graphical view. The dynamically assigned IP address is used in conjunction with BOOTP or DHP protocol. When starting up, a device requests its IP address and other parameters from a central server that can deliver this information using either the BOOTP or the DHCP protocol.

The IP address information for the host is recorded in a number of places. When the IP address information is entered, it is cached in memory and is available for use by the TCP/ IP software. Alternatively, this information can be recorded on a system file in the operating system. IP addresses of other systems can be discovered by consulting a special file, usually called the "hosts" file, or by using the DNS protocol. The DNS service is typically used to determine a host's IP address given its symbolic DNS name. In some systems, proprietary protocols can be used to discover an IP address on a network. An example of this is the WINS service used in Microsoft's operating systems.

You must consult your operating system manuals for actual details on configuring IP addresses. The following sidebar provides examples of configuring IP addresses for most Unix implementations, and for the Windows NT operating System.

Imagine that you are configuration network interface to IP address 144.19.74.102, subnet mask 255.255.0.0, and directed broadcast address 144.19.255.255.

For a Unix Configuration

1) Log on as root user

2) Run the following command:

Ifconfig eth 0 144.19.74.102 netmask 255.255.0.0. broadcast 144.19.255.255.

Replace eth0 with the Unix logical name of the network interface.

For Windows NT

1) Log on as Administrator user.

2) Select the following:

- Start
- Settings
- Control Panel
- Network
- Select the TCP/IP protocol
- Select Properties

3) You will see a dialog box in which you can enter the IP address and subnet mask.

Windows NT, by default, uses the appropriate directed broadcast address of 144.19.255.255 based on the subnet mask that you specify.

4.7 TCP/IP

Perhaps no other protocols designed to work above the Data Link and Physical OSI layers are as popular as TCP/IP. That's primarily because this global protocol suite has been used by and continually promulgated by thousands of government and educational institutions world-wide.

The Transmission Control Protocol (TCP) and the Internet Protocol (IP) are commonly referred to collectively as TCP/IP. TCP represents a transport layer protocol that provides end to end reliable transmission. To do so, TCP includes such functions as flow control, error control, and exchange of status information. In comparison, IP represents a connectionless-mode network layer protocol designed to route message between networks. In addition to TCP, the Internet suite specifies an optional connection less –mode layer 4 transport protocol known as the User Datagram Protocol(UDP).UDP is used for transaction – based applications, such as transmission of network management information when transmission efficiency is more important than reliability.

Figure 5 illustrates the layering structure of the TCP/IP protocol suite to include a few of the application services included in the suite.

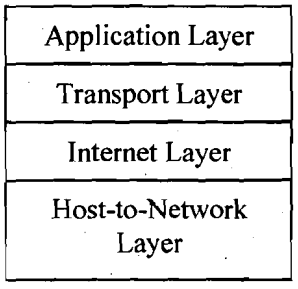

Figure 5: TCP/IP reference model

The advantage of TCP/IP for a network operating system is simple: Interconnectivity is possible for any type of operating system and hardware platform that you might want to add.

TCP/IP is not a single protocol but a set of more than a dozen protocols. Each protocol within the TCP/IP family is dedicated to a different task. All the protocols that make up TCP/IP use the primary components of TCP/IP to send packets of data.

Transmission Control Protocol and Internet Protocol are two of the primary protocols in the TCP/IP family. The different protocols and services that make up the TCP/IP family can be grouped according to their purposes. The groups and their protocols are the following:

Transport: These protocols control the movement of data between two machines.

TCP (Transmission Control Protocol): A connection-based service, meaning that the sending and receiving machines communicate with each other through a stream of messages. TCP has message delivery assurance routines incorporated into it.

☞ **Check Your Progress 1**

1) What is a packet?

.....

.....

.....

2) Which layer of the OSI model has been divided into two sublayers, and what are they?

.....

.....

.....

3) Although UDP is connection less, what is the benefit of UDP?

.....

.....

.....

4) Describe how a TCP message gets from one place to another in step sequence?

.....

.....

.....

4.8 ADDITIONAL TCP/IP-RELATED PROTOCOLS

There are several additional protocols designed to assist TCP and IP. Since routing is so important on a packet-switched network like the Internet, specialized protocols have been designed to assist in this function. Special protocols for determining

addressing on the Internet have also been devised. Additionally, some additional protocols may be involved in error-checking and flow control, just to name a few. Let's explore some of these additional protocols that are included in the TCP/IP suite of protocols.

- **FTP File Transfer Protocol** allows the transfer of copies of files between one node and another. FTP is not hardware-dependent so its services can function just about anywhere. Using this utility to copy data is typically referred to as "FTPing" a file.

- **NFS Network Filing System** was developed by Sun Microsystems Inc. It provides shared access to files in a very transparent and integrated way. This protocol is discussed in more detail a little later.

- **TELNET Remote Terminal Emulation** allows users to communicate with diverse hosts. The TELNET application provides terminal-type access to PCs.

- **UDP User Datagram Protocol** is a bare-bones rapid transmission protocol that uses IP packets to deliver data with no reliability features like connections and ACKs. The forte of UDP is speed, not reliability. It is used in NFS.

- **SMTP Simple Mail Transfer Protocol** is the middleman that uses UDP to move data around from one internetwork host to another. Applications run on both hosts that make use of SMTP.

- **ICMP Internet Control Message Protocol** offers flow control and error-detection to the unreliable delivery method of IP. It provides a facility for routers and gateways on the net to communicate with a source if there is a problem. It also provides a mechanism for determining if a destination cannot be reached.

- **RIP Routing Information Protocol** provides information for routing devices about pathways and number of hops to achieve them. RIP was popularized by its use in a Berkeley UNIX application called "Routed". RIP is ideal for smaller networks, but considered impractical for larger internetworks.

- **ARP & RARP Address Resolution Protocol & Reverse Address Resolution Protocol** are special protocols to allow TCP/IP to interact in environments such as Ethernet. ARP maps TCP/IP addresses to Ethernet Data Link layer addresses. RARP maps the Ethernet Data Link layer address to the TCP/IP address.

That's an overview of some of the better-known additional protocols.

4.9 APPLICATION LAYER PROTOCOLS

The first generation of Internet applications from the late 1960s and early 1970s addressed the need to transfer files between computers, to have remote access to computing resources, and to support communication between users at different hosts. The applications, which met these needs: FTP, Telnet, HTTP, SNMP and DNS, were the predominant applications in the first decades of the Internet. The emergence of the World Wide Web in the early 1990s quickly made web browsing the most popular Internet application. Recently, applications for streaming audio and video, telephony over the Internet, and peer-to-peer file sharing have again changed the landscape of Internet applications.

This section discusses knowledge about how applications interact with the Internet and how applications take advantage of the Internet to provide network services to end-users.

4.9.1 File Transfer Protocol

The **File Transfer Protocol (FTP)** for copying files between computer systems is one of the oldest application- layer protocols and was created before the TCP/IP protocol suite. FTP is a client-server protocol where an FTP client accesses an FTP server. The FTP client authenticates itself with a user name and a password to establish an FTP session. A successful authentication results in the establishment of an FTP session, where the FTP client can download ("get") and upload ("put") files and file lists. When transferring files, FTP respects the ownership and access privileges of files.

Most hosts have a utility program, generally also called FTP, which provides an interactive command line interface to run an FTP session. . FTP clients can also be integrated in other applications, such as web browsers. Anonymous FTP is a special form of file transfer service that allows public access to files at an FTP server. An FTP client can establish an anonymous FTP session by supplying the user name "anonymous" and an arbitrary password (The FTP server usually requests to supply an email address as password).

FTP employs TCP as its transport protocol, thereby ensuring a reliable transfer of transmitted data. Two TCP connections are established for each FTP session, called control connection and data connection. The control connection is used for commands from the client and messages from the server. The data connection is used for the transport of files. FTP server uses the well-known TCP port 21 for the control connection, and the well-known TCP port 20 for the data connection, and an FTP client selects available ephemeral port numbers. The control connection is established at the beginning of the FTP session and stays up for the entire lifetime of the session. The control connection is used by the FTP client for authentication, for setting various session parameters, and for commands to download or upload files. The data connection is opened and closed for each transfer of a file or file list. As soon as a file or a file list has been transferred, the data connection is closed. If there is another file transfer, the data connection is re-opened. By default, the data connection is established upon request by the FTP server. The FTP client starts a TCP server that waits for a connection on an ephemeral port, and sends the port number of this port on the control connection to the FTP server. After the FTP server receives the port, it can request a data connection to the FTP client. A security concern with FTP is that the user name and the password submitted by the FTP client on the control connection at the beginning of an FTP session are not encrypted. Therefore, anyone with the ability to capture traffic from an FTP client can obtain the user name and password used by an FTP client.

The FTP client sends commands to the FTP server, and the FTP server responds with a three-digit response code and an explaining text message. The commands from the FTP client as well as the responses from the FTP server are transmitted as ASCII

Characters.13 The end of a client command and the end of a server response is represented by an end-of-line sequence, which consists of the ASCII special characters Carriage Return (ASCII 10) followed by Line Feed (ASCII 13). When the TCP connection to TCP port 21 of the FTP server is established, the FTP server sends a message that it is ready to interact on a new FTP session. Then, the user supplies a user name and a password. If the authentication is successful, the user sends the IP address and port number of its ephemeral port for the data connection. The IP address and port number is sent in dotted-decimal notation, where the first four numbers

specify the IP address and the last two numbers specify the port number. By default, files are transmitted as text files using ASCII characters. However, the FTP client can change this default so that files are transmitted bit-by-bit without any interpretation. When the file transfer has been completed, the FTP server sends a message to the FTP client. At this 13 ASCII (American Standard Code for Information Interchange) is an encoding format for textual data, which represents an alphanumeric character or special character by seven bits. Application-layer protocols transmit each ASCII character in a byte (octet) with the most significant bit set to zero. The FTP client can download or upload another files, or end the FTP sessions by issuing the command "Quit".

4.9.2 Trivial File Transfer Protocol (TFTP)

The **Trivial File Transfer Protocol (TFTP)** is a minimal protocol for transferring files without authentication and no separation of control information and data as in FTP. Therefore TFTP must not be used on computer where sensitive /confidential information is stored. TFTP is frequently used by devices without permanent storage for copying an initial memory image (bootstrap) from a remote server when the devices are powered on. Due to the lack of any security features, the use of TFTP is generally restricted.

TFTP uses the unreliable transport protocol UDP for data transport, whereas FTP uses TCP. Each TFTP message is carried in a separate UDP datagram. The first two bytes of a TFTP message specify the type of message, which can be a request to download a file, request to upload a file, a data message, or an acknowledgement or error message. A TFTP session is initiated when a TFTP client sends a request to upload or download a file from an ephemeral UDP port to the (well-known) UDP port 69 of a TFTP server. When the request is received the TFTP server picks an ephemeral UDP port of its own and uses this port to communicate with the TFTP client. Thus, both client and server communicate using ephemeral ports.

Since UDP does not recover lost or corrupted data, TFTP is responsible for maintaining the integrity of the data exchange. TFTP transfers data in blocks of 512 bytes. Each block is assigned a 2-byte long sequence number and is transmitted in a separate UDP datagram. A block must be acknowledged before the next block can be sent. When an acknowledgment is not received before a timer expires, the block is retransmitted.

4.9.3 TELNET

Telnet is a remote login protocol for executing commands on a remote host. The Telnet protocol runs in a client-server mode and uses the TCP protocol for data transmission. A client initiates a Telnet session by contacting a Telnet server at a remote host. Recently, the use of Telnet in public networks has been discouraged since Telnet does not offer good protection against third parties that can observe ("snoop") traffic between a Telnet client and a Telnet server. At the Telnet client, a character that is typed on the keyboard is not displayed on the monitor, but, instead, is encoded as an ASCII character and transmitted to a remote Telnet server. At the server, the ASCII character is interpreted as if a user had typed the character on the keyboard of the remote machine. If the keystroke results in any output, this output is encoded as (ASCII) text and sent to the Telnet client, which displays it on its monitor. The output can be just the (echo of the) typed character or it can be the output of a command that was executed at the remote Telnet server.

Telnet uses a single TCP connection for communications. The Telnet server uses the well known TCP port 23 and the Telnet client uses an ephemeral TCP port. After establishing the TCP connection, the Telnet client and server negotiate a set

of parameters for the Telnet session, including terminal type, line speed, if typed characters should be echoed to the client or not, and so on. Unless a Telnet session is explicitly configured not to do so, Telnet client sends one TCP segment for each typed character. Telnet provides some independence from the differences of hardware and software at hosts by mapping input and output to a virtual device, called Network Virtual Terminal (NVT) or pseudo terminal. The Telnet client and Telnet server map the input from a keyboard and the output to a monitor to the ASCII character set. Thus, before a character is sent over the network it is encoded as ASCII character. Also, when an ASCII character is received on the TCP connection, the receiving host interprets the character and translates it into its local character format.

4.9.4 Remote Login

Rlogin is an alternative remote login application program for hosts that run the Unix operating system. Rlogin takes advantage of the fact that both the client and server run a similar operating system, and, for this reason, is simpler than Telnet. Rsh is an application program for the execution of a single command on a remote Unix host. There is also an application program for file transfers between Unix hosts, called rcp. However, this group of applications has poor security features, and is, therefore, often disabled.

The **Secure Shell** suite of protocols provides application layer services for remote login and file transfer services, similar to FTP, Telnet, rlogin, rsh, and rcp, but ensures secure encrypted communications between untrusted hosts. All components of Secure Shell provide authentication, confidentiality, and integrity of data, using a variety of encryption and authentication algorithms, and protect against common attacks on the security or integrity of communications between hosts.

4.9.5 Electronic Mail(Email)

Electronic Mail (email) is the primary Internet service for exchanging messages between users at different hosts(computer) on the Internet. The exchange of email messages is asynchronous, meaning that the transmission and retrieval of an email message can occur at different times.

A user on a host prepares an email on a mail preparation program, called a user agent. Examples of user agents on Unix operating systems are mail, xmail, elm, or pine 14. Email messages are written in plain text and users can add non-text files to an email message. The user agent passes the email to a mail server, where the message is queued for transmission., the user agent and the mail server are on the same host, but it is also possible that they are on different hosts. The mail server uses the application-layer protocol SMTP (Simple Mail Transfer Protocol) to transmit the email message to the mail server of the receiver of the email. The sending mail server uses DNS, the domain name service, to locate the correct remote mail server. In addition to translating host names into IP addresses, DNS also provides the IP addresses of mail servers. For the email ,the mail server at Host A queries DNS for the IP address of the mail server that is responsible for the domain of the email receiver. Once the IP address of the remote mail server is obtained, the sending mail server starts an SMTP client and initiates a TCP connection to the SMTP server of the remote mail server at the well-known TCP port 25. As soon as the TCP connection is established, the SMTP client and server exchange SMTP commands and transfer the email message. These are user agents for hosts with a UNIX operating system. Commands and the email message are transmitted as plain text using ASCII characters. If an email message contains parts that are not text files, these parts are converted to ASCII characters before transmission.. As in FTP, all messages are transmitted as ASCII characters,

using the end-of-line sequence to indicate the end of a command. The SMTP client issues commands to the server, and the server responds to each command with a three-digit response code and explaining text. After the TCP connection is established, the remote mail server sends a brief message.

An email message may traverse multiple SMTP servers before reaching its destination. For example, on most networks a single host is dedicated to handle all outgoing email messages for all hosts of the network. In this case, all hosts forward their outgoing email message to the dedicated mail server, which relays the emails to the proper destinations. Also, in many networks, the receiving mail server does not have access to the receiver's mailbox, and relays incoming messages to the mail server on a host where the receiver's mailbox resides. When a mail server relays an email message, it adds lines to the header of an email message. These lines can be used by the receiver of an email message to trace the path of an incoming email. A mail server adds incoming emails to the mailbox of a user, and assumes that the user has access to the mailbox. Often, however, a user is not permanently connected to its mail server. In such situations, the user can employ mail access protocols to retrieve mail messages from their mailboxes at a remote host. Currently, two popular mail access protocols are in wide use: Post Office Protocol Version 3 (POP3) and Internet Mail Access Protocol (IMAP). Mail access protocols are generally integrated as a component of the user agent.

Exchange of POP3 messages between a POP3 client and a POP3 server. An exchange of POP3 commands between a POP3 client and a POP3 server. All commands and messages are in plain ASCII text, and lines are separated by an end-ofline sequence (CR and LF). The POP3 server acknowledges each command from the client.

For outgoing messages, the user agent at Host X still uses SMTP. , The SMTP client of the user agent at Host X connects to the SMTP server of Host A. Here, Host A acts as a relay and forwards the message to the mail server of the email receiver. Alternatively, the SMTP client at Host X can directly connect to the mail server of the receiver.

Browser-based Emails

Browser based emails allow user to access emails through web browser. The user agent (the web browser) is used HTTP (not POP/IMAP) for the interaction between the browser and email server. When a user wants to send or view the received mail, the browser sends all such commands in the form of HTTP

Example of browser based emails are rediffmail, Gmail, etc.

Multipurpose Internet Mail extensions (MIME)

Traditional email systems are text based. *Multipurpose Internet Mail extensions (MIME)* system extends the basic email system by permitting users to send binary file, e.g., multimedia file, any other arbitrary format

4.10 WORLD WIDE WEB (WWW OR WEB)

The World Wide Web (WWW or Web) emerged in the early 1990s as a new application for access to content stored on Internet hosts. Within a few years, the Web became the most popular Internet application, and traffic on the Internet was dominated by Web applications. The Web is a distributed hypertext system, which is implemented as a client-server application.

A Web client program, called a Web browser, retrieves and displays documents from a Web server. The documents, often referred to as web pages, are formatted using the Hypertext Markup Language (HTML). HTML documents are text files that contain HTML tags, which describe how text should be displayed in the user interface of a Web browser. A hyperlink is special type of tag. It is a reference to another document, which may be located at a different Web server. When displayed in a browser, hyperlinks can be activated with a mouse click.

When activated, the browser retrieves the document that is referenced in the hyperlink. A Web browser makes it convenient to access a variety of documents at different web servers, which refer to each other by hyperlinks, thus, providing users with a feeling of navigating a global database of documents. On the Web, the location of a document is expressed in terms of a Uniform Resource Locator (URL). A URL specifies a unique location for a document on the web. It can reference an HTML document, but also any other file that can be accessed with a Web browser. An example of a URL is http://www.ignou.ac.in/index.html, which specifies that an HTML document with name *index.html* can be accessed via the protocol *HTTP* from a host with the name www.ignou.ac.in

There is no notion of sessions, as, for example, in Telnet and FTP.

In older versions of HTTP, which are still in use today, an HTTP client initiates one TCP connection for each request to the HTTP server. When a single client issues multiple requests to the HTTP server, the number of TCP connections between the HTTP client and HTTP server can grow large. To reduce the number of TCP connections, HTTP/1.1, the current version of HTTP, permits multiple HTTP requests and responses on the same TCP connection, leaves the TCP connection open after a request has been served. The HTTP client does not need to wait until a request is completed before issuing new requests. As in many other application layer protocols of the Internet, HTTP messages are transmitted as ASCII text, using the end-of-line character sequence to indicate the end of a message. The most common HTTP messages sent by a client are requests for HTML or other documents. The server responds to such a request either with a message that contains the requested document or, if the request cannot be satisfied, with an error code. Let us now discuss a request and a response between an HTTP client and an HTTP. Suppose a user has typed the URL is http://www.ignou.ac.in/index.html in a web browser.

When the URL is typed, the web browser starts an HTTP client, which establishes a TCP connection to port 80 of the HTTP server of host www.ignou.ac.in

4.11 DOMAIN NAME SYSTEM

The Internet Protocol address is a 32- bit integer. If somebody wants to send a message it is necessary to include the destination address, but people prefer to assign machines pronounceable, easily remembered names (host names). For this reason the Domain Name System is used. These logical names also allow independence from knowing the physical location of a host. A host may be moved to a different network, while the users continue to use the same logical name. The idea behind domain names is simple: Rather than forcing people to memorize IP numbers, why not give them cryptic names to remember instead?

Domain Name System maps a name to an IP address and conversely an address to a name. Initially when the size of the Internet was small all machines used to maintain a host.txt file, which was passed on, incase of Updates. This host.txt file was centrally managed but looking at the present Internet scenario this does not seem to be a feasible option due to the following reasons:

- Size of the file will be very large, scalability also becomes an issue
- This is centrally managed but since the Internet has distributed management, the management of name space should also be of distributed nature.
- There can be inconsistent results for queries.
- Because of the centralized management as the frequency of lookups increases the time for reply can be very large.

The Domain Name System (DNS) is a distributed database used by TCP/IP applications to map between hostnames and IP addresses, and to provide electronic mail routing information. Each site (university department, campus, company, or department within a company, for example) maintains its own database of information and runs a server program that other systems across the Internet can query. The DNS provides the protocol, which allows clients and servers to communicate with each other.

The system accesses the DNS through a resolver. The resolver gets the hostname and returns the IP address or gets an IP address (*Figure 6*) and looks up a hostname. As we can see in *Figure 6,* the resolver returns the IP address before asking the TCP to open a connection or sending a datagram using UDP.

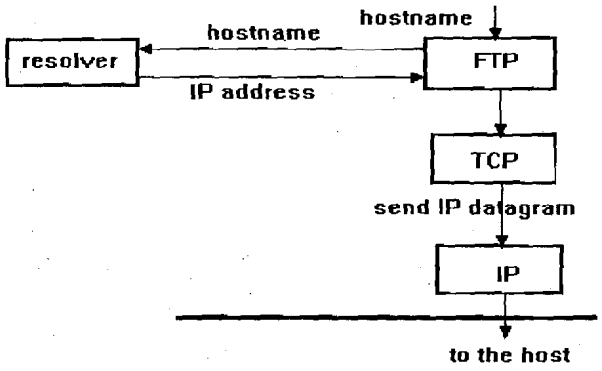

Figure 6: DNS working scheme

DNS Design Goals

The primary goal is a consistent name space, which will be used for referring to resources. Names should not be required to contain network identifiers, addresses, routes, or similar information as part of the name. Name space should be maintained in a distributed manner, with local caching to improve performance. Mechanisms for creating and deleting names; these should also be distributed.

The costs of implementing such a facility dictate that it is generally useful, and not restricted to a single application. We should be able to use names to retrieve host addresses, mailbox data, and other as yet undetermined information. All data associated with a name is tagged with a type, and queries can be limited to a single type.

The name space should be useful in dissimilar networks and applications.

In short design goal of DNS:

1) Distributed ownership: Since the Internet has distributed ownership; the ownership of name space should also be of distributed nature.

2) Have no obvious size limits for names, name components, data associated with a name, etc.

3) DNS protocol should be independent of the network topology.

4) OS/Architecture independent

Design Principles

Hierarchy: The name space as well as management space should be hierarchical. The name space can be represented as a tree with the root label as a null string. The domain name system uses a hierarchical naming scheme known as domain names, which is similar to the Unix file system tree. The root of the DNS tree is a special node with a null label. The name of each node (except root) has to be up to 63 characters. The domain name of any node in the tree is the list of labels, starting at that node, working up to the root, using a period ("dot") to separate the labels (individual sections of a name might represent sites or a group, but the domain system simply calls each section a label). Thus, the domain name "ignou.ac.in" contains three labels: "ignou", "ac", and "in". Any suffix of a label in a domain name is also called a domain. In the above example the lowest level domain is "ignou.ac.in" (the domain name for the IGNOU in India), the second level domain is "ac.in" (the domain name for Academic organizations of India), and the top-level domain (for this name) is "in" (the domain name for India). The node in is the second level node (after root) *(Figure 7)*.

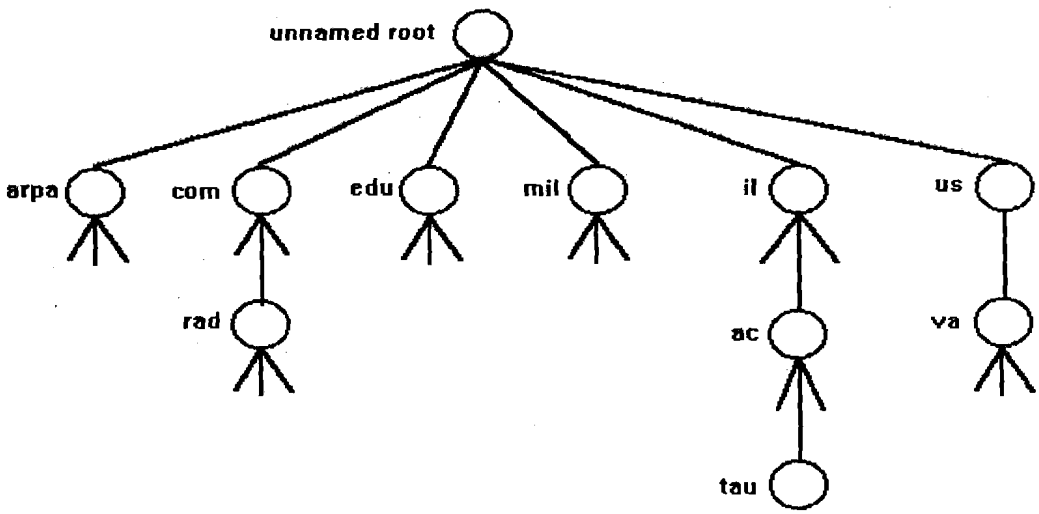

Figure 7: Hierarchical organisation of the DNS

Caching: A fundamental property of the DNS is caching. That is, when a name server receives information about a mapping, it caches that information. Thus a later query for the same mapping can use the cached result. The DNS uses the caching for optimizing search cost. Caching is required as otherwise there will be:

- long time for lookup
- congestion at the root server

Every server has a cache for recently used names as well as a record of where the mapping information for that name was obtained. When a client asks the server to resolve certain name the server does as follows:

1) Check if it has authority for the name. If yes, the server doesn't need caching information.

2) If not, the server checks its cache whether the name has been resolved recently. If yes, the server reports the caching information to its clients.

We can examine the cache when the server cashed the information once, but didn't change it. Since information about a particular name can change, the server may have incorrect information in its caching table. The Time to Live (TTL) value is used to decide when to age information. Whenever an authority responds to a request, it includes a TTL value in the response, which specifies how long it guarantees the binding to remain.

DNS Architecture

NAME SERVERS are server programs, which hold information about the domain tree's structure and set information. A name server may cache structure or set information about any part of the domain tree, but in general a particular name server has complete information about a subset of the domain space, and pointers to other name servers that can be used to lead to information from any part of the domain tree.

REVOLVERS are programs that extract information from name servers in response to client requests. Revolvers must be able to access at least one name server and use that name server's information to answer a query directly.

Data in the DNS consists of Resource Records. There exists a data type for each record. It is of the form (A, MX) where A is the 32-bit IP address, MX is a 16-bit value along with a host name which acts as the mail exchange for the domain. DNS can be used for both forward lookup (host name to IP address) and reverse lookup (IP address to host name). Name space has an entire subtree for reverse mapping.e.g. INADDR.ARPA for reverse lookup

DNS Zones

The zone *(Figure 8)* is a subtree of the DNS that is administered separately. Whenever a new system is installed in a zone, the DNS administrator for the zone allocates a name and an IP address for the new system and enters these into the name server's database. Within a zone DNS service for subsidiary zones may be delegated along with a subsidiary domain. A name server can support multiple zones.

Zones are contiguous regions of the name space, where each can be forked into sub zones. Each of these sub zones can have its independent management.

For example, the IGNOU zone has 2-sub zones cse and lib, which have their own management.

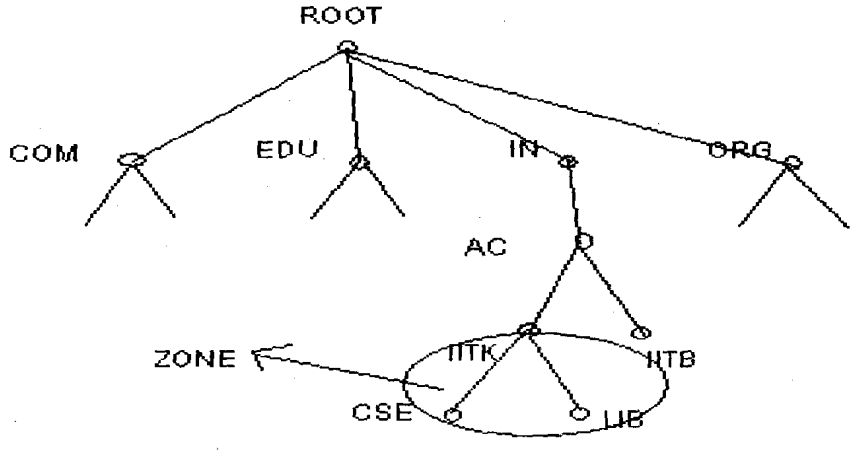

Figure 8: DNS Zones

A name server can support multiple zones; several sub zones can use the same name server. Name servers have pointers among each other.

Remarks on DNS

- DNS uses datagram based access, although a DNS query requires reliability, TCP is not used, as it is a query response mechanism.
- Root server is replicated for improved reliability.
- Address resolution is done recursively. Any DNS server will pass requests it cannot handle to a higher-level server and so on until either the request can be handled or until the root of the DNS name space is reached.

DNS for System break-in

DNS is highly vulnerable to attacks and spoofing. An intruder can intercept virtually all requests to translate names to IP addresses, and supply the address of a subverted machine instead; this would allow the intruder to spy on all traffic, and build a nice collection of passwords if desired.

IP spoofing attacks can be prevented to an extent. Ssh provides an improved type of authentication. The server has a list of host keys stored in /etc/ssh_known_host, and additionally each user has host keys in $HOME/.ssh/known_hosts. Ssh uses the name servers to obtain the canonical name of the client host, looks for its public key in its known host files, and requires the client to prove that it knows the private host key. This prevents IP and routing spoofing attacks.

rlogin and rsh permit ordinary users to extend trust to remote host/user combinations. In that case, individual users, rather than an entire system, may be targeted by source routing attacks. The information required for this attack are the target hostname, trusted hostname and the user name, which are obtained by the "finger" command.

Attack is done as below:

In spoofing a host or application to mimic the actions of another. The attacker pretends to be an innocent host by following IP addresses in network packets. rlogin service can use this method to mimic a TCP connection from another host by guessing TCP sequence numbers.

These attacks can be prevented by:

- Prevent datagram routing with invalid source addresses.
- Introduce unpredictability into connection control mechanisms, such as TCP sequence numbers and the allocation of dynamic port addresses.
- Letting rsh/rlogin to do forward loop along with the reverse lookup.

Allowing to do forward lookup creates a problem called " poisoning the cache " where the attacker sends an unsolicited record along with the PTR record(PTR-a pointer to another part of the domain name space).

This attack can be subverted by rejecting with the record, which arrives along with the PTR record.

4.12 SNMP AND UDP

Simple Network Management Protocol (SNMP)

SNMP *(Figure 9)* is the simple network management protocol. It is used by network management frameworks to manage and monitor network devices, such as hubs and routers. Some computer systems also respond to SNMP queries

SNMP is not actually a protocol: it's a client server application that runs on the UDP (User Datagram Protocol) service of the TCP/IP protocol suite to manage the

network. Network management means to ensure that network is up and running, taking corrective measures and performing maintenance activities. It was developed to be an efficient means of sending network management information over UDP, using Ports 161(SNMP) and 162 (SNMP TRAP).

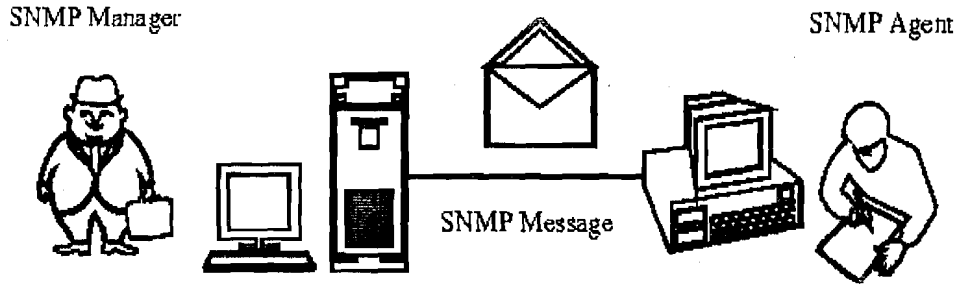

Figure 9: SNMP

Network management system contains two primary elements: a manager and agents. The Manager is the console through which the network administrator performs network management functions. Agents are the entities that interface to the actual device being managed. Bridges, Hubs, Routers or network servers are examples of managed devices that contain managed objects. These managed objects might be hardware, configuration parameters, performance statistics, and so on, that directly relate to the current operation of the device in question. These objects are arranged in what is known as a virtual information database, called a management information base, also called MIB. SNMP allows managers and agents to communicate for the purpose of accessing these objects.

The model of network management architecture looks like *(Figure 10)* this:

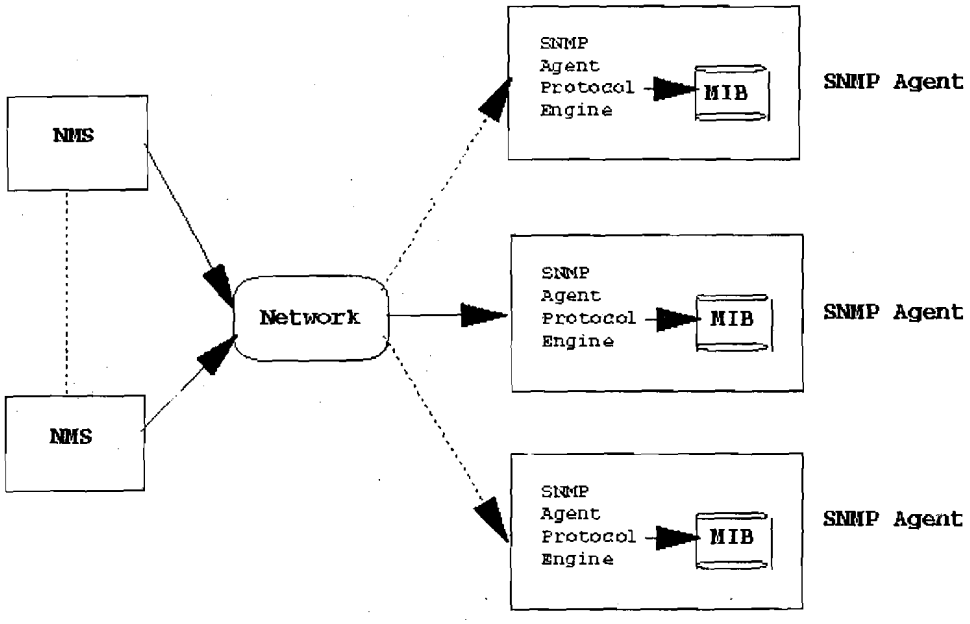

Figure 10: SNMP Architecture

A typical agent usually:

- Implements full SNMP protocol.
- Stores and retrieves management data as defined by the Management Information Base.
- Can asynchronously signal an event to the manager.
- Can be a proxy for some non-SNMP manageable network node.

A typical manager usually:

- Implemented as a Network Management Station (the NMS)
- Implements full SNMP Protocol
- Able to
  - Query agents
  - Get responses from agents
  - Set variables in agents
  - Acknowledge asynchronous events from agents

SNMP uses the *User Datagram Protocol* (UDP) as the transport protocol for passing data between managers and agents. UDP was chosen over the *Transmission Control Protocol* (TCP) because it is connectionless; that is, no end-to-end connection is made between the agent and the NMS when *datagrams* (packets) are sent back and forth. This aspect of UDP makes it unreliable, since there is no acknowledgment of lost datagrams at the protocol level. It's up to the SNMP application to determine if datagrams are lost and retransmit them if it so desires. This is typically accomplished with a simple timeout. The NMS sends a UDP request to an agent and waits for a response. The length of time the NMS waits depends on how it's configured. If the timeout is reached and the NMS has not heard back from the agent, it assumes the packet was lost and retransmits the request. The number of times the NMS retransmits packets is also configurable.

At least as far as regular information requests are concerned, the unreliable nature of UDP isn't a real problem. At worst, the management station issues a request and never receives a response. For traps, the situation is somewhat different. If an agent sends a trap and the trap never arrives, the NMS has no way of knowing that it was ever sent. The agent doesn't even know that it needs to resend the trap, because the NMS is not required to send a response back to the agent acknowledging receipt of the trap.

The upside to the unreliable nature of UDP is that it requires low overhead, so the impact on your network's performance is reduced. SNMP has been implemented over TCP, but this is more for special-case situations in which someone is developing an agent for a proprietary piece of equipment. In a heavily congested and managed network, SNMP over TCP is a bad idea. It's also worth realizing that TCP isn't magic, and that SNMP is designed for working with networks that are in trouble—if your network never failed, you wouldn't need to monitor it. When a network is failing, a protocol that tries to get the data through but gives up if it can't is almost certainly a better design choice than a protocol that will flood the network with retransmissions in its attempt to achieve reliability.

SNMP uses the UDP port 161 for sending and receiving requests, and port 162 for receiving traps from managed devices. Every device that implements SNMP must use these port numbers as the defaults, but some vendors allow you to change the default ports in the agent's configuration. If these defaults are changed, the NMS must be made aware of the changes so it can query the device on the correct ports.

Figure 11 shows the TCP/IP protocol suite, which is the basis for all TCP/IP communication. Today, any device that wishes to communicate on the Internet (e.g., Windows NT systems, Unix servers, Cisco routers, etc.) must use this protocol suite. This model is often referred to as a protocol stack, since each layer uses the information from the layer directly below it and provides a service to the layer directly above it.

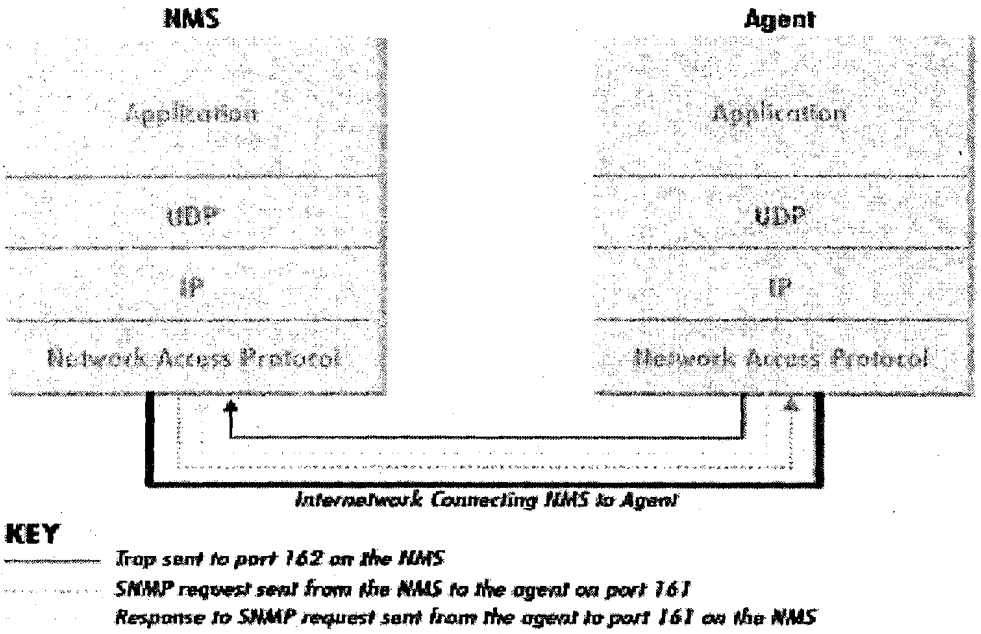

Figure 11: TCP/IP communication model and SNMP

When either an NMS or an agent wishes to perform an SNMP function (e.g., a request or trap), the following events occur in the protocol stack:

Application

First, the actual SNMP application (NMS or agent) decides what it is going to do. For example, it can send an SNMP request to an agent, send a response to an SNMP request (this would be sent from the agent), or send a trap to an NMS. The application layer provides services to an end user, such as operator requesting status information for a port on an Ethernet switch.

UDP

The next layer, UDP, allows two hosts to communicate with one another. The UDP header contains, among other things, the destination port of the device to which it is sending the request or trap. The destination port will either be 161 (query) or 162 (trap).

IP

The IP layer tries to deliver the SNMP packet to its intended destination, as specified by its IP address.

Medium Access Control (MAC)

The final event that must occur for an SNMP packet to reach its destination is for it to be handed off to the physical network, where it can be routed to its final destination. The MAC layer is comprised of the actual hardware and device drivers that put your data onto a physical piece of wire, such as an Ethernet card. The MAC layer also is responsible for receiving packets from the physical network and sending them back up the protocol stack so they can be processed by the application layer (SNMP, in this case).

☞ **Check Your Progress 2**

1) What are the various categories of TCP/IP Well-known Services?

.....

.....

2) What is standard HTTP port?

.....

.....

.....

3) Can a server will work as an FTP server and a webserver?

.....

.....

.....

4) Multiple choice

i) TCP/IP is the main protocol used by computers on the Internet

a) TRUE

b) FALSE

ii) If I wanted to login to a host computer via the Internet, I would use

a) telnet

b) traceroute

c) snmp

d) ftp

iii) To determine if another computer was reachable (alive), the utility I could use is

a) ping

b) nslookup

c) telnet

d) ftp

iv) My computer is very slow. I wish to take advantage of a much faster computer on the network and have my program run there. Which of the following would I use?

a) ftp

b) snmp

c) rsh

d) ping

5) I need to print a document on a printer attached to a UNIX server. Which command would I use?

telnet

b) ping

c) rsh

d) lpr

I wish to gain some statistics from a network server about the number of data packets being sent and received. Which of the following should I use?

a) snmp

b) telnet

c) rsh

d) ftp

4.13 SUMMARY

This unit introduced the building blocks on which internetworks are built. Internetworks are complex systems that, when viewed as a whole, are too much to understand. Only by breaking the network down into the conceptual pieces can it be easily understood. As you read and experience internetworks, try to think of them in terms of OSI layers and conceptual pieces.

Understanding the interaction between various layers and protocols makes designing, configuring, and diagnosing internetworks possible. Without understanding of the building blocks, you cannot understand the interaction between them.

Assigning Internet address to the nodes on the network is a very common task you will perform when building a TCP/IP network. Two types of IP addresses exist: those for IP version 4 and those for IP version 6. IP addresses must be unique on a network. in special cases, it is possible to introduce non-unique addresses, called private addresses.

IP addresses posses a certain structure; they are divided into a network number (netid) and a host number (hostid) portion. This chapter examines the strengths and weaknesses of this scheme. Not all IP addresses can be assigned as a unique identification for network connections. Only class A, B, and C addresses are assignable to individual network connections. Class D is used for IP multicasting. In addition, there are several special addresses used for broadcasting, loop back addresses, and special circumstances. Besides IP & TCP, we looked at several other protocols.

4.14 SOLUTIONS/ANSWERS

Check Your Progress 1

1) A packet is a unit of data that is sent from an originating host to a destination host on a network.

2) Datalink; logical Link Control (LLC) and Media Access Control(MAC)

3) UDP can transfer data faster than TCP

4) (1) First, TCP receives a request from a higher-application that a connection to a remote machine is needed.

   (2) TCP sends a request message to the destination machine. The request message contains a unique number called the socket number that identifies the sending machine's IP address and a port number.

   (3) The message is passed down to IP

   (4) IP assembles a datagram and send it to the destination

(5) The destination's IP, when received the request, sends back its socket number. This process called "exchange of socket number".

(6) TCP sends back up message to the application.

(7) The application then sends data to TCP as a stream of data.

(8) TCP assembles the data in to packets, so called "TCP segments".

(9) TCP then adds a header (Protocol Data Unit) to the data.

(10) TCP send the packaged segment to IP.

(11) IP encapsulates it and sends it over the network, so called "datagram".If there is several segments, a sequence number will created and added to the datagram.

(12) If a segment is missing, TCP sends message back to the sending machine with the faulty sequence number.

(13) After properly received, TCP generated either a positive acknowledgment (ACK), or a request to resend a segment.

Check Your Progress 2

1) The port numbers are divided into three ranges: the Well Known Ports, the Registered Ports, and the Dynamic and/or Private Ports.

   - The well known ports are those from 0 through 1023.
   - The Registered Ports are those from 1024 through 49151
   - The Dynamic and/or Private Ports are those from 49152 through 65535

2) standard port number :80

3) Yes , a server can be running several services and listening at different ports for request

4) 1 (a)

   2 (a)

   3 (a)

   4 (c)

   5 (d)

   6 (a)

4.15 FURTHER READINGS

- *"Internetworking with TCP/IP,* Douglas Comer, Volume I, Fourth Edition", Prentice Hall, 2000.
- *Computer Networking,* Kurose and Ross, Second Edition, Addison-Wesley, 2003 (optional, in RBR).
- *"Computer Networks: A Systems Approach",* Peterson & Davies, Morgan Kaufmann, Second Edition, 2000 W.

- *"TCP/IP Illustrated"*, Volume 1, The Protocols, Richard Stevens, Addison-Wesley, 1994.
- *"Internetworking with TCP/IP"*, Douglas Comer, Volume 3, BSD Socket Version, Prentice Hall, 1993.
- *"Computer Networks"*, Tanenbaum, Third Edition, Prentice-Hall 1996.
- *Data Networks,* Dimitri Bertsekas and Robert Gallager, Second Edition, Prentice-Hall 1992.
- *"OSI A Model for Computer Communication Standards"*, Uyless Black, Prentice Hall, 1991.
- *"Data and Computer Communications"*, William Stallings, Fourth Edition, MacMillan, 1994.

NOTES