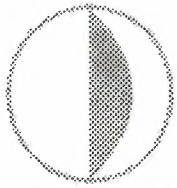


NEAR EAST UNIVERSITY

Faculty of Engineering

Depertmant of Computer Engineering

TCP/IP BASED DATA COMMUNICATIONS

Graduation Project
COM- 400

Ertan Sönmez

Asst.Prof Dr Doğan İbrahim Akay

Nicosia - 2004

ACKNOWLEDGMENTS

"First, I would like to thank my family for their constant encourgement and support during the preparation of this project...

Second, I would like to thank Dr.Doğan Ibrahim for his invaluable advice and belief in my work and myself over the course of this Graduation Project.

Finally, I would also like to thank my friends for their advice and support."

ABSTRACT

FTP-File Transfer Protocol.IP-Internet Protocol.SNMP-Simple Network Management Protocol.SMTP-Simple Mail Transfer Protocol.OSI-Organization for Standardization.
TCP-Transmission Control Protocol.

INTRODUCTION

TCP/IP (Transmission Control Protocol/Internet Protocol) is the basic communication language or protocol of the Internet. It can also be used as a communications protocol in a private network (either an intranet or an extranet). When you are set up with direct access to the Internet, your computer is provided with a copy of the TCP/IP program just as every other computer that you may send messages to or get information from also has a copy of TCP/IP. TCP/IP is a two-layer program. The higher layer, Transmission Control Protocol, manages the assembling of a message or file into smaller packets that are transmitted over the Internet and received by a TCP layer that reassembles the packets into the original message. The lower layer, Internet Protocol, handles the address part of each packet so that it gets to the right destination. Each gateway computer on the network checks this address to see where to forward the message. Even though some packets from the same message are routed differently than others, they'll be reassembled at the destination.

TCP/IP uses the client/server model of communication in which a computer user (a client) requests and is provided a service (such as sending a Web page) by another computer (a server) in the network. TCP/IP communication is primarily point-to-point, meaning each communication is from one point (or host computer) in the network to another point or host computer. TCP/IP and the higher-level applications that use it are collectively said to be "stateless" because each client request is considered a new request unrelated to any previous one (unlike ordinary phone conversations that require a dedicated connection for the call duration). Being stateless frees network paths so that everyone can use them continuously. (Note that the TCP layer itself is not stateless as far as any one message is concerned. Its connection remains in place until all packets in a message have been received.)

Many Internet users are familiar with the even higher layer application protocols that use TCP/IP to get to the Internet. These include the World Wide Web's Hypertext Transfer Protocol (HTTP), the File Transfer Protocol (FTP), Telnet (Telnet) which lets you logon to remote computers, and the Simple Mail Transfer Protocol (SMTP). These and other protocols are often packaged together with TCP/IP as a "suite."

TABLE OF CONTENTS

Chapter 1:Indroduction to Data Communications

1.1 Layers, Protocols and Interfaces

The communication system is responsible for providing communication between nodes in a distributed system. This system allows any network node to transmit information any other node connected to the communication network. Computer network architectures facilitate interconnectivity among homogeneous and (especially in an open sys arena) heterogeneous systems. Because communication systems are complex, it is common to divide them into lave. Some layer structures represent formal models.

1.1.1 A Layer Model of Communication

To illustrate the concepts of layered communication models, It is possible to develop three-layered model that generally describes communication. Consider the following three layers in terms of standard, person-to-person communication:

1. The Cognitive layer includes concepts, such as understanding, knowledge, and existence of shared, mutually agreed upon symbols. This is the level at which information becomes available for human use. Computer user interfaces work at this level.

2. The Language layer is used to put concepts and ideas into words. Examples for humans are words or mathematics. Computers might use ASCII or EBCDIC characters.

3. The Physical transmission layer provides the medium for the actual communication. This layer may be exemplified in several forms, such as sound vibration in the air, written words on paper, or visual signs. Data communication may use electrical, radio, or light signals on a variety of media.

This example illustrates the nature of layered models. The three layers are independent of each other. The "upper" layers require the support of the "lower" ones. To communicate ideas, the language and conveyance are required, but the opposite may not be true.The goals of a layered architecture mirror the goals of structured programming: to define functional modules with clearly defined

interfaces. Modules are conceptually simplified and easier to maintain. Provided mat the interfaces for a module remain stable, the internals of the module may be freely modified.

1.1.2 A Layer Model of Client/Server Computing

All major network architectures share the same high-level objectives.

1. Connecitity permits various hardware and sowtware to be interconnected into a uniform, single system image, networking system.
2. Modularity allows building of divirse networking systems from a relatively small number of geneeral-purpose components.
3. Reliability supports error-free communication via error detection and correction availability.
4. Ease of implementation,use, and modification provides generalized,widely acceptable solutions for network installation,modification and managment,and by supplying end users with network-transparent communication facilities.

To achieve these high-level objectives,network architecture support modular design.Each module's functions are organized into functional,hierarchical, architected layers

A layered approach is especially useful when analyzing client/server computing. Communication between distributed processors takes place on numerous levels, from signals on wires to applications that exchange control information or data. At each level, the nature of the communication is somewhat different. Network hardware works in terms of pulses electrical voltages and currents. Applications communicate through a number of mechanisms with names such as; Named Pipes or Advanced Peer-to-Peer Communication; between are numerous other mechanisms.

Client/server uses features at all of these layers to facilitate a tigh integration of processes running on different computers. Some understanding of the layers involved is necessary in order to understand how client/server computing works.

In data communication models, the layers are composed of entities, which can be hardware components and software processes. Entities of the same layer but in different network nodes are called peer entities. Layers at the same level in different nodes are peer layers.

The typical distributed system architecture consists of the following functional layers:

1. **Application Layer.** This is the topmost layer of the architecture. Typically it performs management of application processes, distribution of data; inter process communication, and decomposition of application functions into distributable processes. .Application layer functionality is supported by lower-level layers.

2. **Distributed Operating System Layer.** This layer provides the system-wide distributed services required by the application layer. It supports global naming, directory, addressing, sharing of local resources, protection and synchronization, intercommunication and recovery. The distributed operating system unifies the distributed functions into a single logical entity and is responsible for creating the single system Image (SSI).

3. **Local Management and Kernel Layer.** This layer supports the distributed operating system in the individual nodes. It supports local inter process communications, memory and I/O access, protection, and multitasking. This layer supports the higher-level layers by providing these services and by communicating with its peer layer in other nodes.

4. **Communication System Layer.** This layer supports communication required by the application, distributed operating system, and local management layers.

The layered architecture provides several important benefits:

1. **Layer Independence.** Each layer is only aware of the services provided by the layer immediately below it.

2. **Flexibility.** An implementation change in one layer does not affect the layers above and below it.

3. **Simplified Implementation and Maintenance.** The support of a modular-layered design and architected decomposition of overall system functionality into simpler, smaller units.

4. **Standardization.** Encapsulation of layer functionality, services, and interface into a carefully architected entities permits standards to be developed more easily.

Communication over different communications links is a complex task. Aside from its use in computer technology, the term *protocol* is *possibly* most familiar in diplomatic settings in which a protocol is an agreement between parties that specifies precise rules of behavior.

Society requires adherence to protocols every day in the use of language. The grammar of any language defines a set of rules that governs interpersonal communication. If two people converse in the same language and dialect they will probably exchange information smoothly and without error. If a German and a Japanese diplomat endeavor to engage in an error-free discussion, they employ translators who understand the language specific protocols and can perform the necessary translations.

Communication between layers is governed by protocols. Protocols include, but are not limited to, formats and order of the information exchange, and any actions to be taken on the information transmission and receipt; the rules and formats for the information exchange across the boundary between two adjacent layers comprise an interface between layers.

Data communications can easily be compared to the United Nations. Translation between communication protocols is a necessary and exacting process. In both cases, many protocols must be comprehended and carefully translated for information to be exchanged readily and without error.

To make the task of conceptualizing and organizing network communication protocols more manageable, the early designers of networking systems standards divided the process into several discrete parts.

1.1.3 The Seven Layerrs of the OSI Communications Model

The *Open System Interconnection* (OSI) model breaks up the job of moving data from one point to another into seven different tasks. The tasks are arranged hierarchically. Each layer contributes to the assembly/disassembly of a packet.
Data moves through the communications network in discrete bundles of bits known as *packets.* Each packet in turn, is divided into four distinct parts:

1. The starting characters alert receiving boards that a packet is on the way.
2. A packet header explains where the packet is going where it came from, and what kind of packet it is (either a data or a network controlling packet)
3. The data the packet is carrying
4. The final error-checking bits and the end-of-packet characters

The OSI model is concerned primarily with the contents of the header section of the packet, which is the part the packet which tells it where to go. In the header section, the layers built up on outbound packets and conversely stripped off on inbound packets. The layers are arranged in a hierarchical fashion. Each layer sends information only to layers immediately above and below it. Figure 1.1 shows the building of a packet.

Each layer sends packets to the layers above it and below it, but each layer only understands and works with information that comes from the same layer on another stack. The network layer (layer three), for example, sends an inbound packet to layer four only after it strips off any layer three information. This same layer three sends an outbound packet to layer two only after it adds layer three information to the packet. On inbound packets, it examines the layer three information to see if it needs to take any action. If the layer three information says that this packet is bound for address 04 and the receiving board is address 90, it discards the packet and does not pass it on to layer seven.

Table 1

Protocol	Headers	Data	Layer
Start B îs		Upper Data	
	Applica		Application
	Pres		Presentation
	Session		Session
	Tran		Transport
	Net		Network
Link			Data Link
Physical Pulses			Physical
Communications Medium			

Figure 1.1 Building an OSI packet.

On outbound packets, layer three adds the .source and destination addresses to packets and passes this enlarged packet to layer two for further processing. Layer three on the receiving board responds only to what layer three on the sending board adds to packet. Each layer on the stack communicates with the same layer on another stack and is unconcerned with what other layers do.

Physical Layer

The physical layer generates the physical pulses, electrical currents, and optical pulses involved in moving data from the *Network Interface Card* (NIC) to the communicate system. RS-232 is an example of physical-layer standard. The units managed at the physical layer are bits.

This layer does not include the communications system, but it does include the connection to it. It handles rise times arid pulse durations. The physical layer does not manage the details of connectors and cabling, which are sometimes unofficially nickname "level 0'' layer- of the model.

Data-Link Layer

The data-link layer is the first level that collects bits and handles data as packets. This does the final assembly on departing packets and performs first inspection on arriving packets. It adds error correction to leaving packets and performs the checksum on arriving packets. Incomplete and defective packets are discarded. If the link layer can determine where the defective packet came from, it returns an error packet- SDLC and HDLC are examples of protocols operating at this level.

Network Layer

When local area networks (LANs) exceed a certain size or geographic area, they must be divided into smaller logical LANs. Devices named routers, bridges, and gateways are used to divide the IAN and create the smaller sub-networks. The network layer routes packets through multiple devices to ensure that a packet arrives at the correct device on the correct sub-LAN.

This level maintains routing tables and determines which route is the fastest available and when alternative routes should be used. This is the first layer at which a device begins filtering out packets that are not going from one network to another so that overall network traffic is reduced. Internet protocol, the IP of TCP/IP, operates at this level, as does NetWare's *Internet work Packet Exchange* (IPX). This network layer is the level at which "connectionless" or "datagram" services operate.

Transport Layer

Transmission Control Protocol, (the TCP of TCP/IP) operates at the transport layer. This is a transition level (that is, the last of the levels that manages routing packets and error recovery). It accommodates any deficiencies that cannot be covered at the network level. If packets are received reliably at the network level, this level is a very simple one. If the communications system cannot provide reliable packet transmission, this level compensates by becoming more complex.

Session Layer

In many network settings İt is desirable to establish a formal connection between communicating entities. This connection assures that messages are sent and received with a high level of reliability. Such precautions are often necessary when the reliability of a network is in question, as is almost always the case when telecommunications are employed. Therefore, session orientation is the norm in most mainframe communications. LANs are generally regarded as highly reliable, and session control is less common in LAN communication.

The session layer is the level that maintains "connection-oriented" transmissions. The process of making and breaking a connection at this level is one of "binding" and "unbinding" sessions. At this level, packets are presumed to be reliable. Error checking is not part of this level's function. The TCP component of TCP/IP, IBM's NetBIOS and Netware's SPX operate at this level.

Presentation Layer

This level is not yet fully defined or widely used. Processing at this level performs any conversion that may be required to render the data usable by the application layer. Data compression/decompression and data encryption/decryption processes might be implemented at the presentation level. Data encryption and compression, however, can be performed by user applications that run above the OSI application layer.

Translations of data formats also can be performed at this layer. An example translation between the ASCII and EBCDIC encoding schemes. This function however, is most frequently performed by the end-user's application.

The presentation layer is frequently misunderstood as presenting data to the user .The presentation layer is a network communication layer.It does not interface directly with end-user display devices. Production of screen displays is the responsibility of the application program executing on the user's workstation.

Application Layer

The application layer also is in the process of being defined. This layer deals with security issues and the availability of resources. The application layer is likely to deal with file transfer, job transfer, and virtual terminal protocols.

Extra Layers

The OSI model was developed when hierarchy was the norm and before the common LAN protocols (ARC net, Ethernet, and Token Ring), were widely used. Since it is inception, its layer definitions have evolved to make it fit in a world filled with LANS as well as with minicomputers and mainframes. One of the adaptations is the informal addition of new layers and sub layers to the original seven layers. An example is the informal addition of level 0 to cover hardware details such as cable connectors and fiber optics.

1.2 Client/Server Connectivity Components

In client/server architecture, client and server systems are constructed from of a number of interconnected components. Each component provides an interface through which it communicates with other components. Communication between clients and server can be viewed as the communication between relevant components. The two distinct classes of components are process components and resource components. Process components software components that actively perform some functions. Resource components the services requested by process components.

From the point of view of client/server interactions, an active resource component act as a server and the users of the resource component and process components act as clients.

Process and resource components, in addition to clients and servers, enter into an association with each other for the purpose of communication. This association connection between a sender of information and a receiver of that information. The client/server connections can be static or dynamic. Static connections are up set at compile, load, or at system initialization time and cannot be changed. Dynamic can be changed "in-flight" at runtime.

1.2.1 Communication and Synchronization

In a client/server environment, coordination and cooperation between components is provided by the communication system's communication and synchronizaction actions. Communication functions involve the exchange of information. They are supported by flow control, error control, naming and addressing, and blocking and segmenting. Synchronization functions involve the coordination of actions between two or more components.

Communication and synchronization are closely related. When communication is performed in shared memory, closely-coupled systems, such as Symmetric Multiprocessing systems and software components, such as semaphores or monitors, are used for synchronization. In more traditional loosely coupled systems that are interconnected by communication networks, mechanisms such as message passing must be used for communication and synchronization.

In either case, the communication service provided by a communication system can be connectionless or connection-oriented. Connectionless services are those where each message transaction is independent of previous or subsequent ones. These are low service overhead services that are relatively simple to implement. An example is called *datagram* service, in which the user is not provided with any form of response to a transaction. Typically, datagram services are used for broadcast or multi destination message transmission.

A connection-oriented service provides a relationship between the sequences of units of information transmitted by a particular communication layer. This is similar to the process of establishing a connection between two telephones on the

public telephone system. Switches at the telephone central offices establish a route through the telephone system that connects the two telephones. This route will probably be different each time. After the route is established, a *circuit* established, and the telephones can communicate as if a continuous piece of wire connected them, in the earl days of telephone communication, this circuit was literally a continuous set of wires, and it was possible to physically trace the connection between the conversing telephones. Such a connection is a *physical circuit.*

Modern communications systems rarely establish physical circuits. Instead they take advantage of new, high capacity communication channels, such as optical fiber, microwaves, and satellites that can carry hundreds or thousands of distinct messages. Devices seeking to establish a connection will be routed through an intermediate path. Individual parts of a message may even be disassembled and transmitted through different paths to be reassembled at the receiving end. Such flexible techniques ensure that messages will be handled efficiently, taking maximum advantage of available capacity.

Switching is invisible to the end devices, however, and after a data connection is established between two computer devices, the connection appears to be a continuous wire run between the computers. A connection that appears physical but in fact uses multiple routes is an *irtual circuit.*

The activity to maintain a virtual circuit between two devices is more complex than connectionless communication since. Most, connection-oriented services have three phases of operation:

1. Establishment. An establishment phase can be used to negotiate connection or session options and quality of service.
2. Data.
3. Termination.

A terminal session to a remote computer or X.25 packet-switched network proto_ cols accepted is the CCITT are examples of connection-oriented services.

A connection-oriented system requires a lot of overhead to control session establishment and termination. This effort maybe is needed to ensure reliable communication over questionable, media. How ever, when the media can be assumed to be reliable, as most LANs are the virtual connection can be dispensed with. Messages may just be sent confidence that they will be received by the intended device. The overhead associated with connectionless communication is low, and performance is consequently higher.Therefore, most LANs use connectionless (also called datagram) communication modes.

Connection -oriented communication is often termed reliable, not because every message is guaranteed to arrive, but because devices are guaranteed to be informed if a message is not delivered as required Connectionless communication is often termed unreliable because the network does not detect the failure to deliver a message. This makes it the responsibility of upper-lever software, perhaps the application itself, to determine that an expected reply has not been received in a reasonable time. This approach works well on LANs.

With respect to connections in general, the flow of information can be uni- or bidirectional. The latter involves a return message and synchronization in response to initial request.

Bidirectional communications are an essential form of communication for the clement/server architecture. The client component requests some service from its possibly remote server. It then waits until the results of its request are returned. Bidirectional client/server interactions can be provided by a message-oriented communication implemented in such request-reply protocols as IBM's Logical Unit 6.2 (LU6.2) or by a procedure-orient communication such as remote procedure calls (RPC).

1.2.2Procedure-OrientedCommunication

Procedure-oriented communication allows applications in a distributed computing environment, such as Open Software Foundation's Distributed Computing Environment (DCE.), to run over a heterogeneous network. The basic technology that enables this functionality is the remote procedure call (RPC).

The RPC model is based cm the need to run individual process components of an application on a system elsewhere in a network. Rocs use a traditional programming construct, the procedure call, which is, in this case, extended from a single system to a network of systems. In the context of a communication system role in a client/server environment, an RPC requesting a particular service from a resource server is issued by a process component (client). The location of the resource component is hidden from the client. Rocs are highly suitable for client/server applications. They usually provide developers with a number of powerful tools that are necessary to build such applications. These tools include two major components: a language and a compiler and a run-time facility. A language and a compiler simplify the development of distributed client/server applications by producing portable source code. A run-time facility enables distributed applications to run over multiple, heterogeneous nodes. They make the system architectures and the underlying network protocols transparent to the application procedures.

The DCE RPC standard appears to be one of the strongest candidates for RPC implementation and deserves closeexamination.

To develop a distributed, DCE-compliant, client/server application, a developer creates an interface definition using the Interface Definition Language (IDL). IDL syntax is similar to ANSI C language with the addition of several language constructs appropriate for a network environment. After the definitions are created, the IDL compiler translates them into stubs that are bound with the client and the server (see fig. 1.2). The stub on a client system acts as a substitute for the required server procedure. Similarly, the server stub substitutes for a client. The stubs are

needed to automate otherwise manual operations-copying arguments to and from RPC headers, converting data as necessary, and calling the RPC runtime.

RPC runtimes should have the following features:

1. Transparency and independence from the underlying networking protocols.
2. Support for reliable transmission, error detection, and recovery from network failures.
3. Support for a common method of network naming, addressing, and directory services, while at the same time being independent of network directory services.
4. Multithreading support for parallel and concurrent processing and capability to handle multiple requests simultaneously thus reduces the time required to complete an application.
5. Portability and interoperability with various system environments.
6. Support for resource integrity and application security.

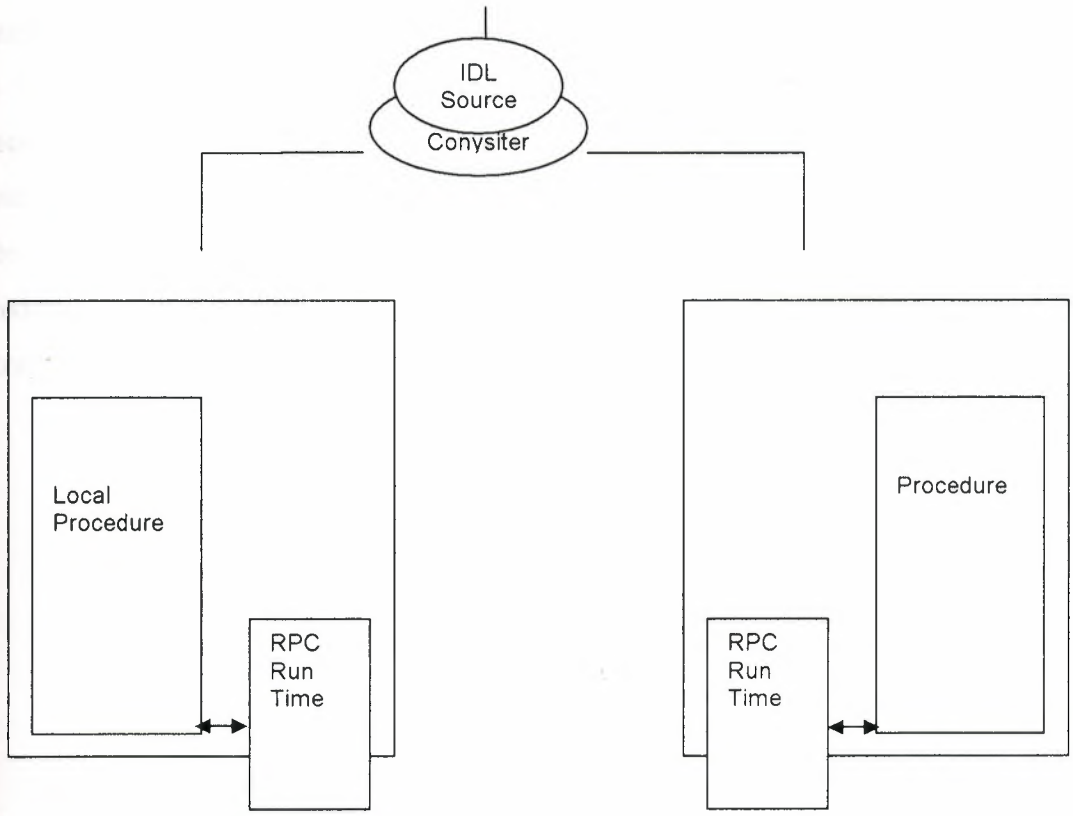


Figure 1.2 An RPC implementation

The DCE implementation of the remote procedure calls includes specific semantics both network transport independence and transparency. DCE RPC includes a pipe facility to eliminate such resource limitations as inadequate main memory. The ISO X.500 standard is used to provide global directory services. DCE RPC utilizes Kerberos authentication and authorization to support security service and asynchronous threads support concurrent and parallel processing.

1.3 Transmission and Access Control Methods

The physical transmission of information over a local area network can be described by two main categories: the actual medium used for the transmission and way this medium is used.

1.3.1 Transmission Media

Current LANs are generally implemented with one of three media types: coaxial cable, and fiber-optic links. Although they will not be discussed further radio and microwave links also are occasionally employed.

Twisted pairs of two individually and brained stands of copper wire. Usually, such pairs form a cable by grouping pairs together and enclosing them in a common protective jacket. The typical intrabuilding telephone wiring is an example of such a cable. Relatively low cost and high availability of such cabling system *have* resulted in the popularity of the twisted-pair wire for LAN implementation. Unshielded twisted-pair wire (UTP) can support transmission speeds of up to 10 MB.Figure 1.5 illustrates UTP cable along with the RJ-45 connector usually employed.

Figure 1.3 Unshielded twisted-pair (UTP) cable.

To eliminate possible electrical interference, a twisted pair cable can be enclosed in a special, high-quality protective sheath. Such cable is know -Asshielded twisted pair cable (STP). .Although it is slightly more expensive, it can be used where higher reliability higher transmission rates over longer distances are required. The IBM Cabling Type 1 and 2 cables are examples of twisted pair cables.

Coaxial cable is familiar to television viewers, especially those with cable TV. Coaxial cable contains a central conducting core (generally copper), that is surrounded by the insulating material, another conductor (braided wire mesh or a solid sleeve), and yet another insulated protective and flexible jacket. Although more expensive, coaxial cable is better isolated from electrical interference than a twisted pair. Coaxial cable can support transmission rates of up to 100 Mbps. DEC connect Communication System's Tin and Standard Ethernet cables are examples of the coaxial cable used in LAN implementations. Figure 1.4 shows an example of coaxial cable along with a typical connect cable with a typical connector.

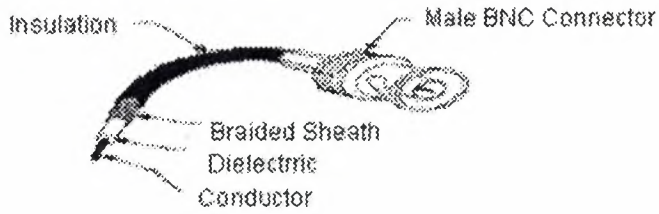


Figure 1.4 Coaxial cable with a typical connector.

Fiber-optic links are the newest transmission medium available for commercial LAN implementations. Optical fiber contains a core of an extremely thin glass thread surrounded by a concentric layer of light insulator, called cladding. Optical signals take form of modified light beams that traverse the length of the fiber-optic link. The cladding is made of a material whose refractive index is lower than that of the core.

The light signals traveling along the core are reflected from the cladding back into the core.

A number of these optical fibers are bound together into one fiber-optic cable surrounded by a protective sheath. Fiber-optic cables are characterized by lighter weight than coaxial cables and significantly higher costs. The light signals transmitted over a fiber optic cable are not subject to electrical interference. Optical transmission medium can support extremely high transmission rates. Rates of up to 565 Mbps can be found in commercially available systems, and experiments have demonstrated data rates of up to 200,00 IBM Cabling System's Type 5 cable is an example of fiber-optic cables used for computer network. Figure 1.4 shows how fiber-optic cable and connectors are constructed.

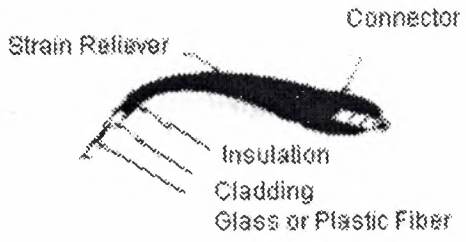


Figure 1.5 Fiber-optic cable with a typical connector

Various implementations of these physical links can be found in a large number of commercially available cabling systems offered by general-purpose communication vendors as well as vendors of various computer networks.

1.3.2 Transmission Techniques

Whichever transmission medium is used in a given LAN environment, the LAN designer must select a technique that the LAN will use to transmit signals over a physical communication link. In general, there are two available transmission

techniques for transmission over a physical communication channel: base band and broadband.

Base band transmission uses discrete signals (pulses of electricity or light that represent a binary 0 and 1) to carry information over a physical transmission medium. This signaling technique is called digital signaling. Base band transmission uses the entire communication channel capacity to transmit a single data signal. LANs typically employ digital, base band signaling.

By dividing the available transmission time into time slots, multiple stations attached to a network can share a common communication channel. This technique is known as time-division multiplexing (TDM). TDM frequently is used to combine several LAN base band signals for transmission through a high-speed network such as a fiber-optic backbone, which can be used to interconnect various buildings or departments in large installations.

Broadband transmission typically employs analog (continuously varying) signals. Digital information is encoded into analog waves by using amplitude, frequency, or phase modulation of the base (carrier) signal. This technique is comparable to the practice of combining multiple television signals so that they may be transmitted over a single cable in a CATV system. A tuner can select the frequency for any individual signal and isolate it from the many signals that are being carried on the cable. Because the signals are each identified with a different frequency on the cable, this technique is known as frequency-division multiplexing (F DM).

In general, the higher the frequencies of the carrier signal, the higher the volume of information that can be carried by this signal. The difference between the highest and lowest frequencies that are carried over the channel reflects that channel's information carrying capacity and is referred to as the channel bandwidth. The bandwidth is directly related to another measurement of channel capacity the number of bits per second that can be carried over the channel, known as the data rate.

Typical base band LANs operate at data rates of 10-20 Mbps. When networks grow to "campus" size, 10-20 Mbps of performance may be inadequate to connect large numbers of workstations in multiple departments or buildings. A common strategy is

to interconnect local base band networks via broadband networks, (frequently designated as backbones), which simultaneously transport multiple signals.

1.3.3 Transmission Control

Technologies covered in tins section are frequently referred to as "media access control" methods in that they are used to ensure that nodes on the network access and share the network media in an organized fashion. In general, various transmission control methods can be classified as follows:

1. **Emissions to transmit**.
2. **Random Control Centralized control**. One station controls the entire network and gives stations p**rol.** Enables any station to transmit without being given specific permission.
3. **Distributed control.** Gives the right to transmit to only one station at a time. The right to transmit is passed from one station to the all stations cooperate to control access to the network.

Each of these transmission control methods offers its own advantages and disadvantages and has access control methods specifically designed to work best with that particular transmission control.

1.3.3.1Centralized Access-Control

Centralized transmission control provides for easier network coordination and management and requires simple station-to-network interfaces. At the same time, centralized transmission Control, by definition, provides a single point of failure and a potential network bottleneck. Centralized control may employ the following access control methods:

1. **Polling.** A master station sends a notification to all other (secondary) stations *indicating that a given station is allowed to transmit. Media access conflicts are eliminated since only the polled station may transmit.*

2. **Circuit Switching.** This can be used successfully in a centralized control LAN implemented by using a star topology. Here, a central station receives requests for a transmission from a secondary station and establishes a connection between the sender and its intended receiver. Circuit switching is widely used in telephony, especially in private branch exchanges (PBX).

3. **Time-Division Multiple Access (TDMA).**This provides a specific time slot for each station on a network. The station is allowed to transmit only during this time slot. The time cycle is started and synchronized by a master station. TDMA can be successfully used on a bus topology.

1.3.3.2 Random Access Control

One of the best-known access control techniques for random transmission control is Carrier Sense Multiple Access with Collision Detection(CSMA/CD).Using this method, stations listen politely to determine whether the network is in use. If not, a station will attempt to transmit.

Carrier Sense Multiple Access with Collision Detection (CSMA/CD) ,used in Apple's Local Talk network, is very similar to the CSMA/CD access methods. However, stations utilize a variety of timing strategies to reduce the likelihood that collisions will occur.

The performance of both CSMA schemes is random since it cannot be predicted when stations will attempt to transmit. CSMA is an extremely simple access control mechanism, and there is little work associated with managing network traffic. However, the random element can cause problems when network traffic demands are high, and CSMA network performance can deteriorate rapidly as demand approaches opacity limits.

1.1.3.3 Distributed Access Control

Token ring passing is the most widely used method of distributed access control and is most frequently used in ring topology networks (for example, IBM's Token Ring).A token is small message that is constantly circulating around the ring. Token passing can be used in a bus, star or tree topology. Token bus methods are similar to a token ring, emulating the token ring method on a logical topology level. IBM Token Ring is the most commonly cited example of the use of token passing to control media access.

Token passing is a more complicated mechanism that the CSMA schemes just discussed. It, is therefore, more expensive implement and the network must have more controls in place to ensure proper operation. When network traffic is high, however, token passing becomes an effective means of ensuring that all network stations are allowed equal network access.

When designing local are networks, many interdependent factors should be taken into consideration: the transmission medium, transmission control and access methods, network topology, band width, and data rates. All these factors affect network performance and cost. Decisions regarding network topology, transmission control, and access control, and access control methods should be made based on the processing and cost requirements of a particular LAN.

Chapter 2: Introduction to TCP/IP

2.1 Brief History of TCP/IP

Perhaps no organization has more complex networking requirements than United States Department of Defense (DoD), Simply enabling communication among of wide variety of computers found in the various services is not enough DoD computers often need to communicate with contractors and organizations that do defense-related research, including most universities. Defense-related network components must be able to withstand considerable damage so that the nation's defenses remain operable during a disaster.

The DoD initiated research into networking protocols (investigating technology now known as packet switching), therefore, is not surprising. In fact, research on the protocols that eventually became the TCP/IP protocol suite began in 1969.Among the goals for this research were the following:

1. **Common protocols.** The DoD required a common set of protocols that could he specified for all networks. Common protocols would greatly simplify the procurement process.
2. **Interoperability,** If equipment from various vendors could interoperate development efficiency could be improved and competition among vendors would be promoted.
3. **Robust communication.** A particularly dependable network standard was required to meet the nation's defense needs. These protocols needed to provide reliable, high-performance networking with the relatively primitive wide-area network technologies then available.
4. **Ease of reconfiguration.** Because the DoD would depend on the network reconfiguring the network and adding and removing computers without disrupting communication needed to possible.

In 1968, the DoD Advanced Research Project Agency (then called ARPA bui redoubled DARPA) initiated research into networks using the technology now called packet switching. The first experimental network connected four sites: the University of California at Los Angeles (UCLA), the University of California at Santa Barbara (UCSB), the University of Utah, and SRI International. Early tests were encouraging, and additional sites were connected to the network. The ARPA net, as it came to be called, incorporated 20 hosts by 1972,

Originally intended to facilitate communication among the DoD, commercial research firms, and universities, the ARPA net gradually became a medium for non-DoD communication as well. By the mid-1980s, the ARPA net had become the backbone of an inter network that connected large numbers of educational institutions and defense contractors, as well as the military network called MIL net. This extended network, which used the ARPA net as its backbone, becomes known as Internet.

In 1986, groundwork was laid for the commercialization of the ARPA net work to isolate military networks from the ARPA net commenced backbone. The ARPA net backbone was dismantled, replaced by a network funded by the National Science *Foundation. NSF net now functions as the Internet backbone, managed by Advanced* Network Services (ANS).

The evolutionary approach of the ARPA net produced a community of users that became involved in debating and setting standards for the network protocols are developed under the control of the companies that develop them. Debate on the Internet protocols, however, has long taken place on a public forum. Consequently, these protocols are not "owned" by any particular company. Responsibility for setting Internet standards rests with an Internet Activity Board.

The initial set of TCP/IP protocols was developed in the early 1980s and become the standard protocols for the ARPA net in 1983. The protocols gain popularity in the user community when TCP/IP was incorporated into version 4.2 of the BSD (Berkley Standard Distribution) UNIX. This version of UNIX is used widely in educational and research institutions and was used as the foundation of several commercial UNIX

implementations, including Sun's SunOS and Digital's Ultrix. Because UNIX established a relationship between TCP/IP and the UNIX operating system the vast majority of UNIX implementations now incorporate TCP/IP.

Evolution of the TCP/IP protocol suite continues in response to the evolution of the Internet. In recent years, access to the Internet has extended beyond original community and is available to virtually anyone who has a computer .This dramatic growth has stressed the Internet and has pushed the design limitations of several protocols. On the Internet, nothing is permanent except change.

2.2 Introduction to TCP/IP

TCP/IP is a family of protocols used for computer communications. The letters stand for Transmission control protocol/ Internet Protocol, but oilier than in the press, the full name rarely is used. TCP and IP are both individual protocols that can be discus separately, but they are not the only two protocols used in the family. Often a TCP/IP user does not use the TCP protocol itself, but some other protocol from the family. To talk about using TCP/IP in this situation is still proper, however, because the name applies generically to the use of any protocol in the TCP/IP family. Because TCP/IP developed by the Department of Defense, the protocol family is sometimes called The DOD Suite, but does not have a classical "marketing" name as does Apple's AppleTalk suite protocols.

Protocols usually are grouped into "families" (sometimes called suites or stacks). Which protocols are grouped together is usually determined by the protocols' implementers. Many protocol families are developed by commercial organizations; for example, AppleTalk is a family of protocols developed by Apple Computers. Each protocol in family supports a particular network capability. No protocol is of much use on its own and requires the use of other protocols in its family. In some ways, protocol families are like a set of golf clubs; each club is used for a particular purpose, and no one club can be used to play an entire game. Usually a golfer purchases all the clubs in a set from the same vendor. Just as each vendor might offer a slightly different set of clubs, network protocol families try to solve the same network problems with a slightly different set of protocols, but many are similar from family to family.

The TCP/IP protocol family includes protocols such as Internet Protocol (IP), Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP), Transport Control Protocol (TCP), Routing Information Protocol (RIP), Telnet, Simple Mail Transfer Protocol (SMTP), Domain Name System (DNS), and numerous others. Keeping all the acronyms straight can be difficult, especially because some are reused by other protocols (for example, the Novell, or IPX, family has a RIP protocol different from the TCP/IP family RIP protocol). An understanding of all the protocols in a particular family is not a prerequisite to knowing how a network basically works. This chapter concentrates on the IP and ARP protocols (mentioning the RIP and ICMP protocols briefly). This focus, coupled with a minimal discussion of a particular link protocol (Ethernet is used for the examples in this chapter), ilhısıraîes how a TCP/IP network causes data to flow smoothly across an internet.

2.3 Understanding Basic Network Concepts

Before answering the preceding questions (or possibly even before understanding ı ıhey are asking), yon must know the meanings of some terms and concepts discuss this chapter.

2.3.1 Addressing

The central concept of networking is addressing. In networking, the address of a device is itsunique identification. Network addresses are usually numerical and have a standart, welldefined format (each defined in its specification document). All devices on a *network need to be given a unique identifier that conforms to a standard format. This* identifier is the device's address. In routed networks, the address has at least two pieces: a network (or area) piece and a node (or host) piece.

In this chapter, network refers to a set of machines connected to the same physical wire (or set of wires connected only by bridges and repeaters). Internet means one or more networks connected by routers. The word internet (lowercase i) is not to be confused with the Internet (uppercase I). The Internet is a specific internet that connects millions computers worldwide and is becoming predominant in the press and elsewhere.

If two devices on an internet have addresses with the same network number, they are located on the same network and thus on the same wire. Devices on the same wire can communicate directly with each other by using their data link layer protocol (that is Ethernet). The examples in this chapter use Ethernet as the medium connecting the devices. Although some particulars might differ, the concepts are the same if the networks are built on Token Ring, Fiber Distributed Data Interface (FDDI), or many other common physical media.

Correct addressing of devices on a network requires that every device connected to same network (wire) be configured with the same network number. Also, every device with the same network number must have a different node (or host) number from every other device with the same network number. Finally, every network in an internet must have a unique network number .To rephrase, every network on an internet must have unique network number, and every node on a network must have a unique node within that network. This rule ensures that no two devices on an internet ever have same network and node number and therefore have a unique address within the internet. In addition to a unique address for every device on an internet, special addresses often are used to address multiple nodes simultaneously. These addresses are called broadcast or multicast addresses.

The following discussion references two different types of addresses: addresses and Media Access control (MAc) layer addresses. These two address types completely independent, of each other. The network layer addresses are IP addresses. These addresses are used to communicate between nodes across an IP internet work. The MAC addresses are used to communicate from node to node on the same wire and often are built right into the communications card (for example, the Ethernet card).MAC addresses are the lowest level addresses and are the means by which all information is ultimately transferred from device to device.

2.3.2 Packets

On most networks, such as TCP/IP networks, the information sent is broken down into pieces called packets (or datagrams) for two main reasons: resource sharing error detection and correction. On networks that have more than two computers (for example Ethernet or Token Ring), the medium connecting the devices is shared. If any two devices are communicating, no other devices can communicate at the same

type. A network works like a party line in that respect. If two devices want to share a very large amount of information. it is unfair for them to become the sole user of network for a long period of time; other devices might have urgent information to transfer to other parties. If the large block of information is broken into many small blocks, each of these can be sent individually, enabling other devices to interweave their own messages between the packets of the extended conversation. As long as each piece is individually addressed to the; intended destination and contains enough information for the receiver to piece it back together, the fact that it is broken into pieces does not matter.

The other main use of packets is for error detection and correction .Networks are ultimately made up of wires (or radio waves or light beams) that are prone to interference which can corrupt a signal sent across them. Dealing with corrupted messages is a big part of networking; in fact, most of the complexity in networking involves dealing with the what-if-something-gets-corrupted scenarios. Many error detection and correction techniques are based on checksums: when a sender transmits information (as bytes of data) running total adding up all the bytes sent is kept and then transmitted at the end of the data transmission. The receiver computes the total of the data received and compares it to the total transmitted. If a difference exists between the total bytes received and total bytes computed, then the data or the total is corrupted. The sender is asked to retransmit the data. This version is much simpler than what really happens, but it is sufficient to illustrate the concept.

If the medium on which the transmission takes place has an average error rate of one bit in one million (that is, for even one million bits sent, one is corrupted on average), then there is a practical upper limit to the amount of data that can be sent in one transmission. Imagine that ten million bits are sent. Normally a transmission of this size contains ten errors, the checksum is wrong, and the sender is asked to retransmit. The retransmission size is the same as the original transmission, so it contains on average ten errors again. The only way to break out of this loop is to break the data into smaller pieces, each with its own checksum that can be retransmitted individually.

2.3.3 Protocols

Each of these packets is a stream of bytes. For true communication to occur, these must have meaning associated with them. This meaning is provided by the protocol specification. A protocol is a set of rules that defines two things—the format of the packets and the semantics of their use.

Most packets have a format that includes a header and a body. The header often includes information such as a source and destination address, the length of the packet, and some type indicator so that the receiver knows how to decode the body. The body can be raw data (for example, a piece of a file or an e-mail message), or it can contain another packet that has its own format defined by its own specification. Packet formats usually are depicted in a specification by a rectangular picture that gives the order, size, and names of the pieces of information that make up the packet. Figure 2.1 is an example of Ethernet frame.

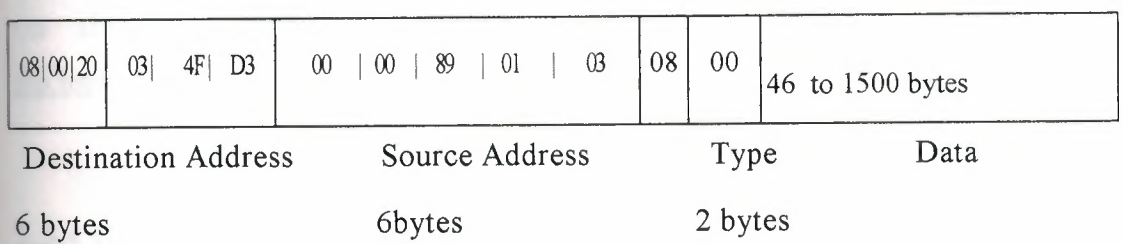


Figure 2.1 An Ethernet frame

You must know more than the formal of a packet to understand the protocol. You also must know when to send which packets and what to do when they are received. Many protocols have very simple formats, but their use is very complicated. Imagine teaching a non-English speaker to behave as an English-speaking receptionist. The "packet formats" might be as follows:

"Hello, this is company X."

"Now May I direct your call?"

"Please hold."

"Good bye."

"That line is busy, may I take a message?"

"That person does not work here."

The "protocol" for answering the phone needs to include when to say each of these the phrases, how to look up an extension in the company phone book, what to say if the being called is not in, how to take a message, what to do with the message after talking it, what to do with a wrong number, and so on.

A protocol specification specifies the format of the information exchange (the packets) and the correct sequencing of that information as well as the additional actions (logging, mail delivery, table updates, and so on) that might be required. Just as the receptions described earlier was only trained to direct incoming calls (and not answer tech support questions), each protocol has a specific set of functions with which it is design to deal.

In the TCP/IP world, most protocol specifications are available online as Request For Comment (RFGs). These specifications tend to be very technical in nature and are directed at engineers who intend to implement these protocols. One site (of many) on the internet that makes the RFCs available for anonymous ftp is ftp.internic.net.An index is available at that site in the file /rfc/rfc-index.txt.

2.3.4 Routes and End Nodes

Routed networks have two classes of devices: end nodes and routers (see fig.2.2).End nodes are the devices with which users interact—workstations and PCs, printers, file servers, and so on. Routers are devices that connect networks. Routers have the responsibility to know how the whole network is connected and how to move information from one part of the network to another. They shield end nodes from needing to know much about the network so that the end nodes can spend their time doing user task. Routers are connected to two or more networks. Every device on a particular network must have the same network number as every other device on that network, and every network must have a different network number. Thus routers must have a separate address for every network to which they are connected. Routers are very much the "post office" of the network. End nodes send information

they don't know how to deliver to the local router, and the router takes care of getting it to its final destination. Sometimes a device such as a file server also is a router, for example, when that end node is connected to more than one network and is running software that enables it to route information between those networks. Routing is often a CPU-intensive chore and can significantly impact the performance of a machine doing tasks other than routing. For this reason, most routers are dedicated machines.

Routers are introduced to networks for several reasons. Routers enable more devices to ultimately be interconnected because they extend the address space available by having multiple network numbers. Routers help overcome physical limitations of the medium by connecting multiple cables.

The most common reason for using a router is to maintain political isolation.Routers enable two groups of machines to communicate with each other while remaining physically isolated, which is especially important when the two groups are controlled by different organizations. Many routers have filtering functions that enable the network administrator to strictly control who uses and what is used on the network. Problems that occur on one network do not necessarily disrupt other networks.

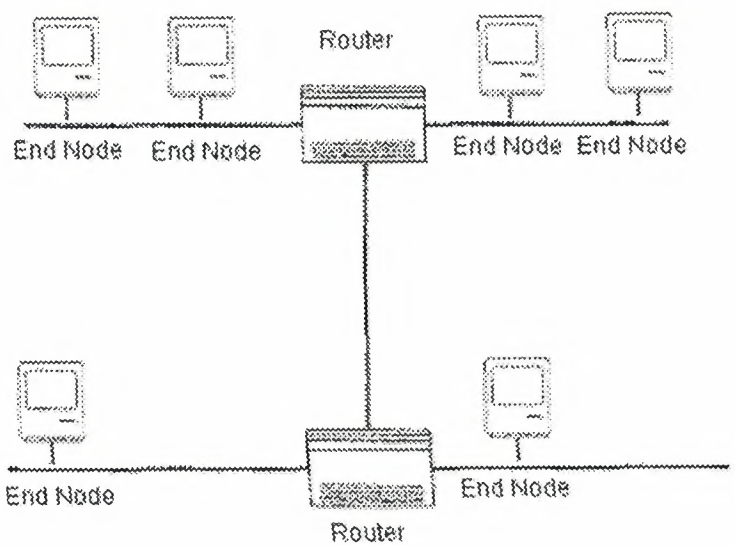
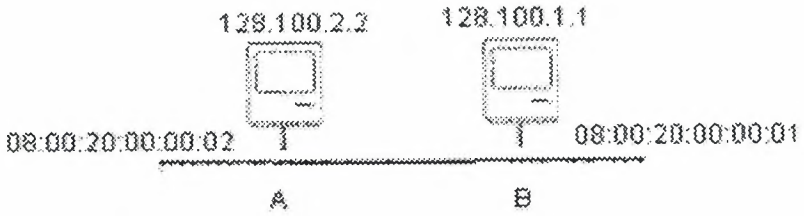


Figure2.2 Routers and end nodes in a network.

2.3.5 End Node Send and Receive Behavior

When a node on a TCP/IP network has an IP packet to send to another node, it follows a simple algorithm to decide how to proceed. The sending node compares the network portion of the destination address with the network portion of its own address. If the two networks are the same, it implies that the two nodes are on the same wire—either directly connected to the same cable or on cables separated only by repeaters or bridges (see fig2.3). In this case, the two nodes can communicate directly using the data link layer (for example, Ethernet). The sending node uses ARP to discover the destination node's MAC layer address and encapsulate the IP packet in a data link layer frame to be delivered directly to the destination node.



Ether Dst	Ether Src	Ether Type	IP Dst	IP Src	
08:00:20:00:00:01	08:00:20:00:00:02	IP	128.100.1.1	128.100.1.2	Data

Figure 2.3 Two end nodes communicating on the same network.

If the network portions are different, the two nodes are separated by at least one router which implies that the sending node cannot deliver the packet without using a router as intermediary. The packet is encapsulated in a data link layer frame addressed to the MAC address of a router on the same wire (if no router is on the wire, then that particular network is isolated and cannot send IP packets to other networks). The router delivers the IP packet to the remote network.

When an end node receives an IP packet, it compares the destination address in the IP packet to its own address and to the IP broadcast address with which it is configured. If the destination address matches either of these addresses, the end node accepts the packet and processes it further. The way it is processed depends on which sub protocol if IP it is. If the destination address does not match, the packet is dropped (ignored), as shown in the following end-node algorithm:

```
Receive
If ((dst addr == my addr) or (dst addr = = broadcast) {   process packet    }

else
if (     drop    (    ignore    )  packet     }
Send
If ( dst net = my net )
 {    deliver    (    may    need    to    "ARP    "    )                    }
else
 {          send to router        }
```

2.3.6 Router Send and Receive Behavior

When a node is functioning as a router and it receives an IP packet, it examines the destination IP address in the packet and compares it to its own IP address. If the addresses are the same or the destination IP address is the IP broadcast address, the

packet is processed as it would be for an end node. Unlike an end node, a router does not auto-matiadly drop packets that are received but not addressed to it. These are packets that end nodes on the network are sending to the router to be forwarded to other networks (see fig. 2.4). All routers maintain routing tables that indicate how to reach other networks. The router compares the network portion of the destination address with each network in its routing table. If the router cannot find the destination network in its routing table, it checks for a default route (typically listed as a route to 0.0.0.0). If it does not find a default route, the packet is dropped (and an ICMP destination unreachable message is sent to the source IP address in the dropped packet).

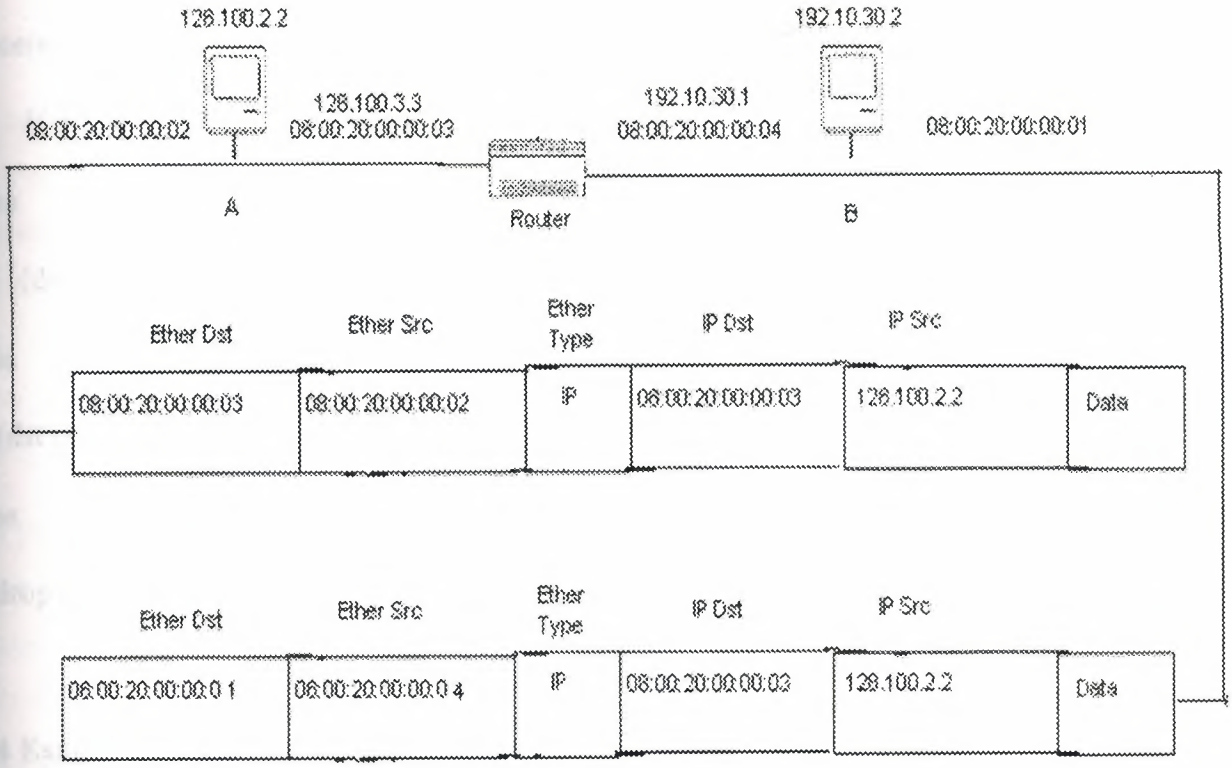


Figure 2.2 Two end nodes communicating on different networks.

When a matching route to a network is found (or a default route exists), the router checks the distance to the remote network. If the distance is listed as 0, the network is directly connected to the router. In this case, the router sends an ARP request for destination IP address and encapsulates the IP packet in a data link layer frame addressed to the MAC address of the destination returned in the ARP response. If the distance is greater than 0, the packet must travel through at least one more router. In this case the router uses the next router field from this route and sends an ARP request for that router encapsulating the IP packet in a data link layer frame addressed to the MAC address of the next router. This way, an IP packet travels across an internet, maintaining the same source and destination IP addresses the entire time, hut having the source and destination MAC addresses change for each hop. The algorithm a router uses when receiving a packet is as follows:

Receive

If ((dst addr = = my addr) or (dst = = broadcast) { process packet }

Else

If (dst net is directly connected) { deliver (may need to " ARP ") }

else

if (dst net in table) { deliver to next router }

else

{ drop (ignore) packet }

2.4 Examine the Format of an IP Address

For any routable protocol to be efficiently routable, the address must have two parts. TCP/IP addresses have two components—a network component and a host(or node) component. Addresses used with TCP/IP are four-byte (32-bit) quantities called simply IP addresses (not TCP/IP addresses) (see fig. 2.5). These addresses are written in standard detonation, which means that each byte is written as a decimal

number separated by dot notation, which means that each byte is written as a decimal number separated by dots. (The period character)—for example, 192.37.54.23 (pronounced "192 dot 37 dot 54 dot 23"). Because each piece of the IP address is 1 byte, its value must be between 0 and 255 inclusive—for example, the IP address 125.300,47.89 could not be a legal because IP address because 300 is greater than 255 and would not fit in a single byte.

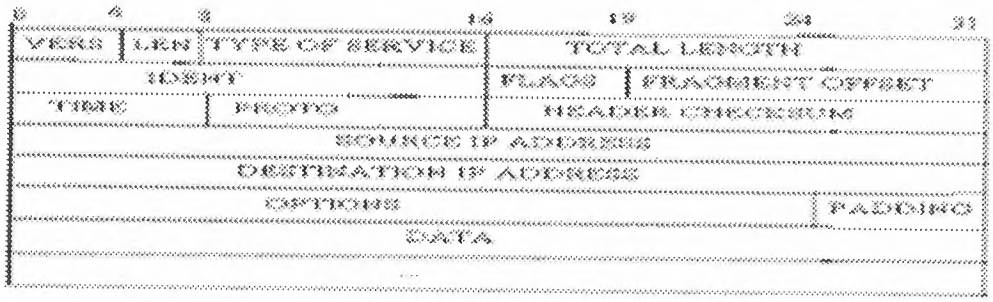


Figure 2.5 Format of an IP address.

IP addresses are composed of a network portion and a host portion. The split is not as simple as the first two bytes being the network portion and the last being the host portion. The designers of the TCP/IP protocols were concerned that they not limit the size of potential networks too severely, so they opted for a graduated method and host division. If the split was to be two bytes for each, no network could have more than 2^{16} hosts on it. Also, smaller networks would waste much of the address space by using only a fraction of the available nodes on any given network.

To provide for efficient address use, IP addresses are divided into classes. The three most important classes of networks are A, B, and C. IP addresses are split into these classes according to the first few bits of the address (or the value of the first byte,if you don't like working in binanry), as in figure 2.6.

An IP network is customarily referred to as an IP address whose host portion consists of all zeroes—for example, 10.0.0.0 or 128.37.0.0 or 200.23.45.0. For example,

137.103.210.2 is a class B address that has a network portion of 137.103 and a host portion of 210.2. This network, the 137.103.0.0 network, can have up to two bytes worth (2^{16}) of which must share the exact same first two bytes 137.103 and must have unique host portions.

0	1	8	16	24	31
0	Netid		hostid		

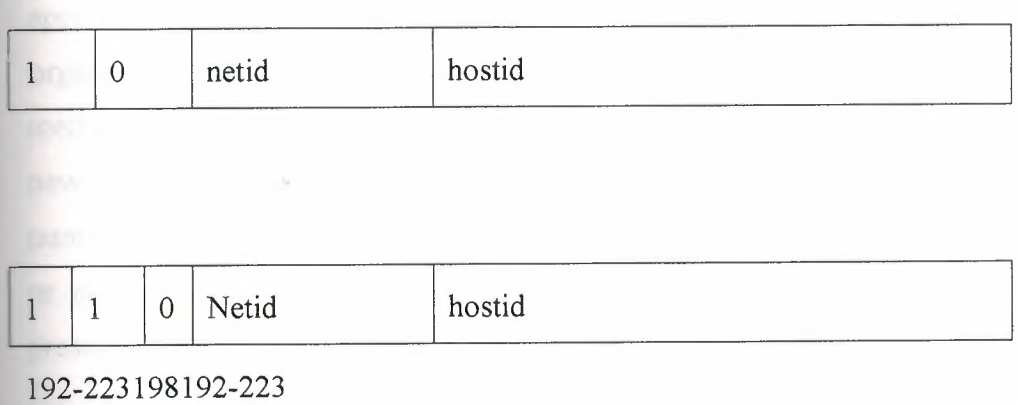


Figure 2.6 The three classes of IP addresses

2.5 Assigning IP Addresses to TCP/IP Devices

IP addresses can be assigned in a number of ways. If an organization wants to build a TCP/IP inter network that never will be connected to any other TCP/IP network outside the organization, then it is acceptable to pick any class A, B, or C network number that allows an appropriate number of hosts on it. This method is rather short-sighted,as much of the benefit of having a TCP/IP network is the capability to connect to the outside world and share resources beyond those in the organization—for example, connecting to the internet. A better strategy is to contact the InierNIC's registration services at network Solutions, Inc. and request an officially assigned network number. The InterNIC ensures that the network number assigned to each applicant is globally unique. All the host ids on that network are free to be assigned as the assignee sees fit.

Sometimes when an organization connects to the Internet through another organization (for example, a commercial service provider or a university), that second organization provides the network number. In addition, many larger organizations have internal network administrators in charge of assigning IP addresses to individual users with company.

When an IP network number has been acquired from an internal network administrator, service provider, or the Inter NIC, it is possible to start assigning specific host IP addresses from that network to individual devices. Usually an organization keeps records of which IP addresses are assigned already and has some method of distributing the unused IP addresses to individuals who need to configure new IP devices. IP addresses must be configured into devices with the same network number as all other devices on the same wire but with a unique host portion. If two or more devices have the same IP address, they will not work reliably and will present a very difficult situation to debug.

Most IP devices require manual configuration. The person installing the device must obtain a unique and correct IP address and type it in to some configuration program or console, usually along with other information such as IP broadcast address, suhnet mask and default gateway address.

Some sites support dynamic configuration of IP devices. Protocols such as Boot Protocol (BOOTP) and Dynamic Host Configuration Protocol (DHCP) enable the use of centralized servers to hand out unique IP addresses and other configuration information on an as needed basis. At the time of this writing, this sort of configuration is not a mature enough technology to find widespread use.

2.6 Mapping IP Addresses to MAC Addresses

Ultimately, all computer communication takes place by moving data from node to node over some form of link such as Ethernet, Token Ring, FDDÎ, and the Point-to-Point (PPP). Many links support attaching more than two nodes and therefore require that all data sent over them be addressed to a specific destination to be delivered correctly. These addresses have nothing to do with IP addresses; they are completely separate and in addition to IP addresses. These addresses are MAC addresses—sometimes called physical, hardware, or link addresses. Unlike IP addresses that are assigned, most MAC layer addresses are built into the hardware by the manufacturer of the device or network interface card, (NIC).

On an Ethernet network, every device on the network has a built-in Ethernet address. This address is a six-byte quantity usually written using hexadecimal numbers with a colon separating the bytes—for example, 08:00:20:OA:8C:6D. Ethernet address are assigned by the Institute of Electrical and Electronics Engineers (IEEE) and are unique among all Ethernet devices. No two devices should ever have the same Ethernet address(manufacturing errors do occur on occasion). The Ethernet address is divided into two parts; the first three bytes constitute the vendor code. Each vendor of Ethernet equipment obtains a unique vendor code from the IEEE. Every piece of equipment supporting Ethernet made by that vendor is programmed with an Ethernet address that begins t with that vender code. In the preceding example, the vendor code 08:00:20 corresponds to Sun Microsystems; every Ethernet device manufactured by Sun begins with those three bytes (see fig. 2.7). The vendor is responsible for making sure that every Ethernet device it manufactures has the same first three bytes (vendor code) and a different remaining three bytes to guarantee that every Ethernet device in the world has built in.

If every Ethernet device already has a unique address, why are IP addresses necessary?. First of all, not every device has Ethernet support; IP addresses enable device that connect to fiber and Token Ring and serial lines to use IP without having to get an Ethernet address. Secondly, Ethernet addresses are organized by equipment vendor rather than by owner organization. To come up with an efficient routing scheme based on who made the equipment rather than on where it is located would be impossible. IP addresses are assigned based on a network topology, not on who manufactures the device. Finally, and most important, is that devices can be more easily move or repaired when an extra level of addressing exists. If an Ethernet card breaks, it can be replaced without getting a new IP address. If an IP node is moved from one network to another, it can be given a new IP address without getting a new Ethernet card.

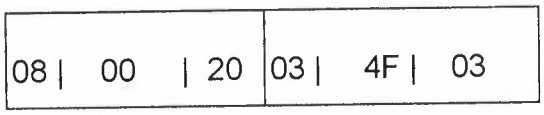


Figure 2.7 A typical Ethernet address including a vendor code.

Network hardware communicates only with other network hardware (for example, two Ethernet cards on two network devices). This network hardware often uses an addressing system that is not friendly to humans but is convenient for the hardware itself. Users and services on networks communicate with other users and services. These services are easier to access if they are addressed in a way that makes sense to people. Addressing that is easy for humans to understand, however, is not always easy for hardware to manage. To solve this problem, a method of mapping user-level addresses to hardware is needed.

Ethernet addresses are long and cryptic and not meant to be regularly dealt with by users. To provide a mechanism for nodes to determine each other's hardware addresses without intervention from the user is possible. For TCP/IP, this mechanism is ARP. When an IP node wants to communicate with another node with the same network number, it assumes that having the same network number implies that the destination is on the same wire. On an Ethernet, for example, the source Ethernet card can directly communicate with the destination Ethernet card if it knows the Ethernet address. To determine the Ethernet address of a node on she same wire, the sending device sends an ARP request to the Ethernet broadcast address (see fig. 2.8). This address is a special address that all Ethernet cards are

configured to listen to (it consists of six bytes of all ones, written in hex as FF:FF:FF:FF:FF:FF). Setting the destination Ethernet address to this value and sending an Ethernet packet causes every device on the Ethernet to accept the packet as if it were addressed specifically to it. It is the Ethernet equivalent of the U.S. postal address "'Occupant."

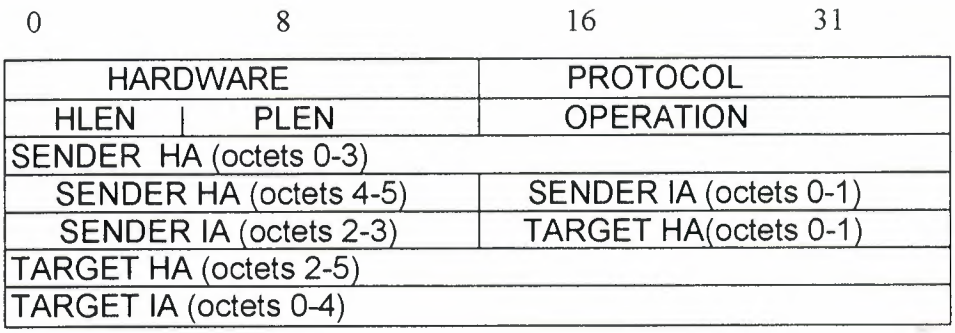


Figure 2.8 The format of an ARP packet

An ARP request asks every node on the wire what is the Ethernet address for a particular IP address. The ARP request contains (among other things) the sender's source IP address and Ethernet address as well as the IP address with which sender wants to the communicate. Every Ethernet device on the network accepts this packet and, if the receiving device supports IP, recognizes that it is an ARP request. The receiving device then compares its configured IP address to the IP address being looked for. If an exact match occurs, the receiving device sends an ARP response back to the sender(through the Ethernet address in the ARP request, not as a broadcast) containing its Ethernet address. The sender can then encapsulate the IP packet it wants to send in an Ethernet packet with a destination Ethernet address as specified in the ARP response.

Why doesn't the sending node simply broadcast every packet it sends? On a large or busy network, this would require every node to be interrupted to process every packet on the network to determine whether the packet was destined for it. This interruption would be very inefficient and would slow the network down considerably. To make sure that broadcasts are minimized, nodes on broadcast networks requiring the use of ARP maintain a list of IP addresses and the Ethernet

addresses that correspond to them. (Determined by previous ARP requests). This list is called the ARP cache and is updated whenever an ARP response is received. A node needing to send many IP packets to the same destination sends an ARP request the first time it tries to contact node and records the Ethernet address it receives in the ARP response. Subsequent IP packets use the Ethernet address in the cache instead of sending another ARP request. Each entry in the cache is kept for some amount of time decided by the implementer of TCP/IP software in use. This timeout might be as little as 30 seconds or as much as several hours or even be configurable. The shorter the time, the more broadcast ARP request there are. But if the time is too long, then a node that changes its Ethernet address(because the the Ethernet card was replaced, for example) cannot be contacted until entry is updated.

2.7 Examining How End Nodes Find a Router

To send a packet to a node on another network, an end node requires the aid of a router. If two nodes on the same network want to communicate, they can do so directly by encapsulating their IP datagrams in link level frames (for example, Ethernet frames) and sending them to each other. This procedure works because nodes on the same network are attached to the same wire (separated only by cable, repeaters, and bridges).When a destination is on another network, it is on another wire and can't be reach directly.The end node encapsulates the IP datagram in a link level frame destined for a router.The router then determines where to send the packet next. Because a router is needed to contact a node on another network, it is necessary for the router to be on the same network as the source node (otherwise the source node would need a router to reach the router!). Routers behave much like a post office for U.S. mail. If you want to deliver messages to someone very close (for example, next door). You would most likely deliver the message yourself. But if the destination is unfamiliar or is far away, you would deliver the message to the nearest post office. The post office would deliver the message for you if the message is for a local address serviced by that post office; otherwise it looks up which post office should deal with it next. The letter might pass through a number of post offices before being delivered.

To deliver a packet to a node on a different network, a source node sends the unmodified IP packet to the local router by encapsulating it in a link level packet addressed to the router's MAC address. If the link level is Ethernet, the source node needs to know the Ethernet address of the local router. For reasons given previously, nodes should not deal with Ethernet addresses directly; therefore, TCP/IP end nodes need to know how to obtain the Ethernet address of a router. By using the ARP protocol, an end node that knows the IP address of a router can obtain the Ethernet address. TCP/ IP end nodes need to be manually configured with the address of at least one router (usually called a *default gateway).* Some TCP/IP implementations enable the router's address to be obtained dynamically by "eavesdropping" on the routers' conversations. In this case the node is configured to "listen" to a particular routing protocol such as RIP.

2.8 How Routers Learn the Network Topology

For routers to fulfill their role as "post office" of the network, they need to know which networks are reachable and how to get to them. To accomplish this, routers store information about the topology of the network. This topology is usually stored as a *routing table* that lists each known network, tells how "far" away the network is, and indicates which router is the next one to send a packet to to reach a network not directly connected (see table 2.1 and fig. 2.9)

Table 2.1

A Routing Table for a Three-Router Network

Network	Distance	Next Router.
	Router 1	
1	0	—
2	0	—
3	1	222.222.222.2

Router 1			
	4	1	
222.222.222.2			
	5		2
222.222.222.2			
	6	0	—

Router 2			
	1		1
222.222.222.1			
	2	1	
222.222.222.1			
	3	0	—
	4	0	—
	5	1	200.15.22.3
	6	0	—

Router 3		
1	2	200.15.22.1
2	2	200.15.22.1
3	1	200.15.22.1
4	0	—
5	0	—
6	0	200.15.22.1

The *cost* of a network can be declared in many ways (depending on whose network you look at), but is most often simply a count of how many routers a packet must go

through to reach a network. The cost to a network is often called the distance or number of hops to a network.

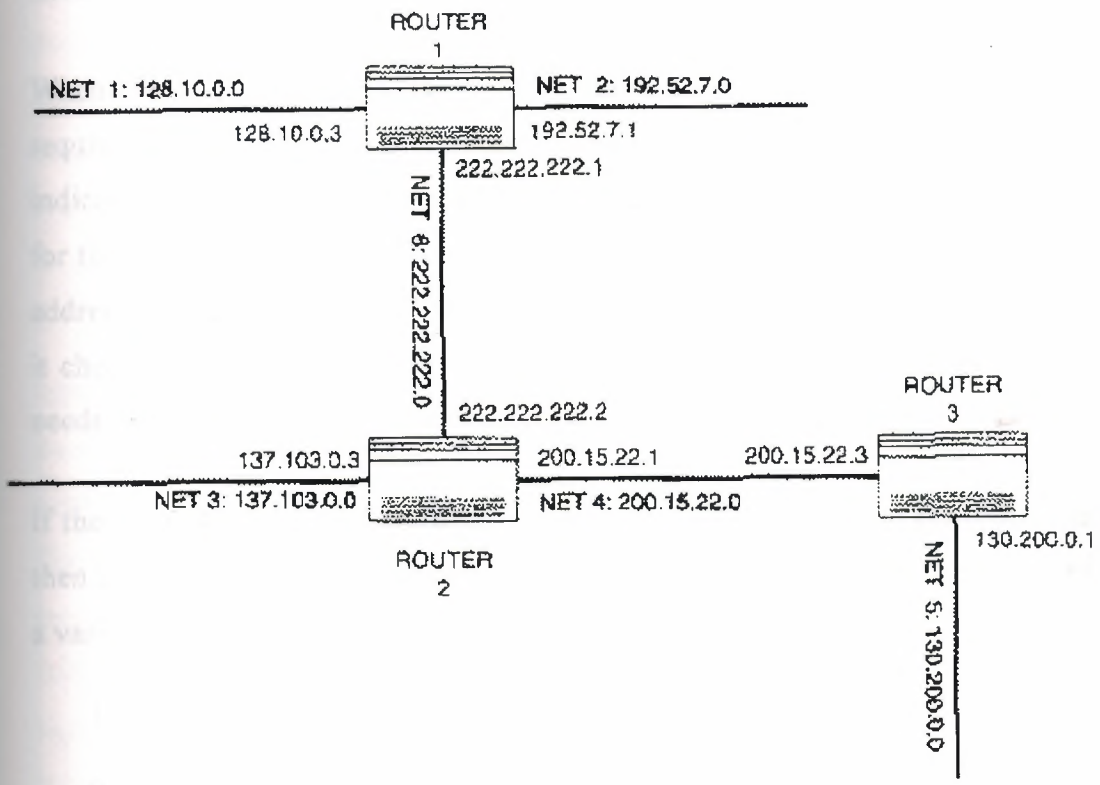


Figure 2.9 A network with three routers.

A cost of zero means that the specified network is directly connected to the router. Packets destined for such a network can be delivered from the router to the final destination by encapsulating the datagram in a data link layer frame (for example, Ethernet) by sending an ARP request for the node. For this reason, the next router field for directly connected networks is meaningless.

When the cost is non-zero, the network in question is not directly connected and requires routing through at least one more router. In this case, the routing table indicates which router is the next one to send to. The router sends an ARP request for this "next hop" router and encapsulates the datagram in a data link layer packet addressed to MAC address of the next router. When this router receives the packet, it checks its routing table and determines if the packet ran be delivered locally or needs to be routed to yet another router.

If the destination network (or default network) does not appear in the routing table, then the packet cannot be delivered and is dropped (ignored). This could happen for a variety of reasons, including the following:

1. The sending node was mistaken or misconfigured
2. The router was misconfigured and does not know about the network
3. All routes to that network are no longer operating (a router farther along the path to the network went down)

Usually when a packet is dropped due to the lack of a router sends an ICMP u Destination

Unreachable message to the source, which should cause the node to log a message informing me user that data is not getting through.

Routing tables are set up in routers by two means: manual configuration and dynamic acquisition. (Sometimes a combination of both methods is used.) Manual configuration is the most straight forward method, but the least robust in the face of a changing network and also can be impossible to maintain in a very large network. When manual configuration is used, the person installing the router is responsible for typing in the various fields for the routing table—telling the router which networks are reachable, how far away they are, and which routers should be used to reach them.

Dynamic acquisition of routing tables is achieved by means of one or more routing protocols. In TCP/IP, the most commonly used routing protocol is RIP (not to be confused for the IPX routing protocol of the same name). RIP is a simple protocol that enables router to tell each other what networks they know about and how far away they are .With this information, each router can assemble a table of every network on an internet, enabling packets to be sent from any network to any other network, which could mean excessively large routing tables if the network were attached to a worldwide network such as the Internet. Therefore, provisions are made to "clump" many networks together in a default route represented by the IP address 0.0.0.0. Routers advertising connectivity to network 0.0.0.0 are saying, "If you don't see the network number anywhere else send the packet to me."

RIP updates are broadcast by every router on every network every 30 seconds.Because these updates can impact network performance considerably on very large or slow networks, more efficient (in bandwidth, at least) protocols are being developed.Open Shortest Path First (OSPF) is a routing protocol becoming popular. OSPF provides a number of benefits to large networks, such as less traffic and faster "flooding" of information regarding changes to the network, but at the expense of a more complex algorithm(implying the need for more memory) to implement. Other protocols are used by routers to learn dynamically the topology of the network and advertise changes in that topology. The mechanics of each one is different, but the general purpose of binding the network together is the same.

2.9 Finding and Using Services

In the end, the purpose of all this encapsulating and routing is to provide users access to services. Users are interested in terminal emulation, printing, file sharing and e-mail; they are not concerned with how these services are created.

Services require support to make them easy to find and use. Most people are not very good at dealing with numbers even if they have sensible structures like IP addresses(never mind Ethernet addresses), for example. People like to deal with names that are like words (if not words themselves). The command Telnet server is far easier remember than telnet 192.34.50.3, for example.

Most services in the TCP/IP world are found through well-known names. Such services are found published in books, in company documents, or by word-of-mouth from the system administrator to users. You can access services if you know the name of the device that provides the service and what program to use to access it. FTP the file printers.txt from server company .com might be the directions to access the file that describes the names of all the printers you can use. A user would type ftp server.company.com; log in; and type get printers.txt to access the file.

IP packets cannot be addressed to a name; they require a four-byte IP address. Much like ARP is used to map an IP address to a hardware address, a service name can be mapped to an IP address in a number of ways. The simplest way is to maintain files that contain name and IP addresses of devices of interest. This file is often called the hosts file because in Unix it is found in the file /etc/hosts, and many IP implementations for other platforms have maintained the convention of calling the file hosts. This solution is simple but not efficient on large networks. To maintain an up-to-date file on every single IP device can be difficult.

Usually a network administrator configures one or more servers to maintain a network accessible database of name-to-IP mappings. Two commonly used methods are the Domain Name System (DNS) and Sun's Network Information System (NTS). Maintaining such a database requires that one or more machines be designated as "keepers of the database" and all other machines send requests to these servers to have a name converted into an IP address or vice versa. A network-accessible database[1] of name-to IP mappings is easy to maintain on a large network and requires little per-device configuration (only needing to know which machine to go to for lookups).

Given the amount of information currently available on the Internet and the rate at which that amount is increasing, obviously, to have just the files and lists of where to find information is entirely inadequate. To ease the burden of locating services and information on a network of the scale of the Internet, many different applications are being built. Applications such as Mosaic, gopher, archie, and World Wide Web (WWW) are being worked on by many people, companies, and universities. These applications try to make wading through the vast amount of information available a little easier (or at least more fun). These applications usually have a graphical user

interface or a simple text-based interface that makes sorting through some subset of everything out there much EASIER.

2.10 TCP and UDP

Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) travel encapsulated in IP packets to provide access to particular programs (services) running on remote network devices.

SOURCE PORT	DESTINATION PORT
LENGTH	UDP CHECKSUM
DATA..... .	

Throughout this chapter the discussion of TCP/IP has revolved entirely around IP. IP addresses enable data to be addressed to a particular node on an internet. After the data arrives, some mechanism is needed to enable the proper service within the device to receive the data. The data might be e-mail or a file or part of a print job. To direct the data to the appropriate program, another level of addressing is needed. Each service available on a node is accessed through a unique address called a port (sometimes also referred to as a socket). Ports are identified by a simple decimal number. For example, port 25 is the SMTP address. These numbers are contained in the TCP and UDP headers of TCP and UDP packets, which are encapsulated within IP packets (see figs. 2.10 and 2.11).

Figure 2.10 Format of a UDP packet

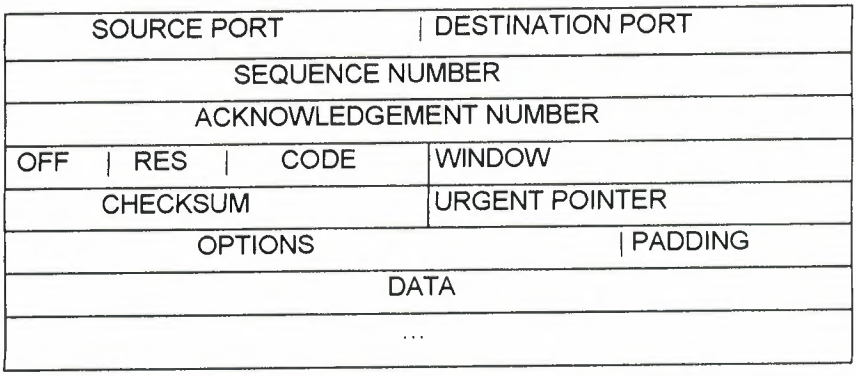


Figure 2.11 Format of a TCP packet

To understand the difference between UDP and TCP, you must know what is meant by datagram versus stream-oriented protocols, and what is meant by reliable versus unreliable protocols.

A datagram-based protocol is one that imposes a maximum size on the amount of data that can be sent in a single transmission. Ethernet, IP, and UDP are all datagram-based protocols. An upper limit to how much data can be sent in a single transmission exists. This type of protocol is analogous to sending a normal letter through the U.S. Postal Service. A single stamp limits the amount of "data" you can send at one time.

TCP is a stream-oriented protocol a user of a TCP-based protocol does not need to worry about the maximum size of a transmission. TCP breaks the transmission into smaller sizes, retransmitting lost pieces, reordering data delivered out of order, and filtering out any extras that might occur due to faulty retransmissions. This type of transmission is analogous to a commercial freight carrier that can deliver as much "data" as the customer wants. The overhead necessary to support TCP is proportionally higher than that of UDP. An application that uses TCP requires more memory and more bandwidth to ensure that the transmission is completed properly.

The other factor that differentiates UDP from TCP is reliability. UDP is an unreliable or best-effort protocol. This definition does not mean that reliable data transfer cannot happen if based on UDP, but that the UDP protocol itself does not handle

reliable data transfer .An application using UDP is responsible for implementing retransmissions; duplicate filtering, and so on, itself. If a UDP packet is lost or corrupted in transmission, it must be noticed by the application sending the data, which is again analogous to the U.S. Postal Service for normal mail. If the post office loses a letter, the letter is gone .They do not store a copy of it and "retransmit" it. Ethernet and IP also are best-effort protocols.

By pushing the overhead needed for reliability into an application, it is possible to make a reliable protocol or application that uses UDP. Sun Microsystems has implemented an entire file system—NFS—on top of UDP. NFS uses a less efficient set of algorithm than TCP to implement reliability, but the overhead is far less. UDP is appropriate for networks in which an application like NFS is used because the level of loss and corruption on LAN is usually very low.

TCP is a reliable protocol .This definition does not mean that TCP guarantees delivery of the data it sends, but that TCP delivers the data if at all possible and reports back to the application if the data cannot be delivered (for example, if the destination node crashed).This reliability requires a great deal of overhead compared to UDP. Overhead is incurred to provide this service efficiently. TCP fragments and reassembles the data stream (so that the data can fit in datagram-based IP packets), retransmits lost packets, filter's out duplicates caused by hasty retransmissions, handles flow control between computers of different speeds, and maintains windows (packets sent ahead that don't wait for an acknowledgment). If the network connectivity is preserved during the transmission, the data arrives in order and uncorrupted. If the connectivity is lost (the receiving program or machine crashed or an intermediate router went down), that fact is to the application using TCP.

Applications that invoke sessions usually use TCP to transfer data. These applications usually require the user to log in or connect before data can be moved. Applications that claim to be *stateless* are usually built on UDP, such as NFS.

Applications and protocols that use UDP include the following:

1. NFS
2. RIP
3. Trivial File Transfer Protocol (TFFP)
4. Simple Network Management Protocol (SNMP)

Applications and protocols that use TCP are as follows:

1. FTP
2. Telnet
3. SMTP

Chapter 3:TCP/IP ARCHITECTURE

The architecture of a computer system refers to overall system design. Microsoft designs its networking products usinjng fairly elaborate networking architecture that enables *the* products to support several popular networking protocols: Net!BEUI, Novell's IPX/SPX, and TCP/IP.. This chapter focuses on the characteristics of network protocol architectures in general and on the specific characteristics of the TCP/IP protocol architecture.

The name TCP/IP describes much more than the TCP and IP protocols. TCP/IP is a suite of protocols in which each protocol performs a subset of overall network communication task.Network implementers select among these protocols to achieve a desired network functionality. The TCP/IP protocol suite architecture defines the way the various TCP/IP protocols fit together.

Microsoft networks can involve multiple protocol suites, so establishing a model that clarifies the relations between the various protocols is essential. The conventional model for comparing protocol suite is the OSI reference model, which is the first topic this chapter covers. The discussion uses the OSI reference model to illustrate common characteristics all network protocol suites share.

After discussing the OSI reference model, this chapter examines TCP/IP architecture.

3.1 Understanding the Communication Process

The communication process is surprisingly complicated. People tend to take communication for granted because they are immersed in many forms of it every day of their lives. Even in the simplest of situations, however, communication can be quite involved. Data communication is a highly technical topic, one in which becoming bogged down in the details can be hard to avoid. Before things get too formal, it is illuminating to look at some examples of communication in which humans engage.

3.1.1 Dialogs

Humans and computers both exchange information in orderly conversations called dialogs. Consider the everyday event of meeting a friend on the street and holding a conversation, illustrated in figure 3.1. Even this simple situation has rules:

1. One person initiates communication by greeting the other person. They might need to negotiate .some details, such as what language to use.

2. The other person acknowledges the greeting. This exchange initiates a conversation, which in data communication terminology is called a session.

3. The individuals undertake an orderly exchange of information. Generally, one person talks while another listens, then the parties exchange roles. A complex set of rules helps the speakers exchange roles without interfering with each other's utterances. Without these rules, orderly conversation breaks down and information exchange grinds to a halt. An example of failed communication is a shouting match when both parties are talking at once and receive little information.

4. Rules are observed for ending the conversation. Polite exchanges take place when both parties feel they have completed their messages. If the conversation breaks down mid-message, communication errors can occur. Data communication often requires a session termination procedure.

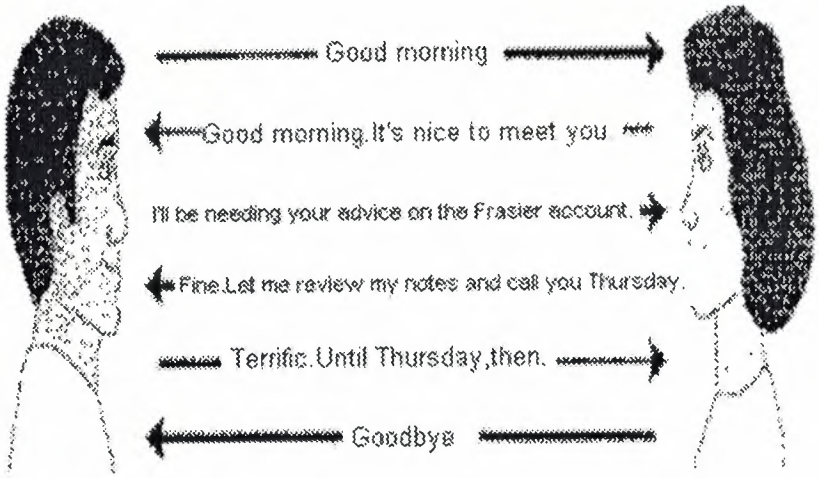


Figure 3.1 An example of a dialog.

Human communication is not always so formal. If you need to warn someone about a falling brick, you do not need to establish a conversation. A single, shouted warning might be the only communication that takes place. In such instances, however, the two parties must be in reasonable agreement about the rules of communication. A warning shout works if both parties speak the same language but might be useless if the warning is in Japanese and the potential victim speaks only German.

Computers establish communication in much the same way as humans. They exchange greetings and negotiate the rules under which communication takes place. The result is a connection between the devices that ensures an orderly dialog. After the need for the connection ceases, another orderly procedure closes it. In that way, neither device is left hanging with incomplete information.

3.1.2CommunicationProtocols

Data communication is surprisingly similar to human conversation. People and computers both utilize formal communication for complex data exchanges, and informal processes for special purposes, such as warnings. Both follow protocols, rules that enable the subjects to exchange information in an orderly, error-free manner. Protocols are obeyed to establish and end communication, so that neither

party is left hanging in an undesirable state. Just as the rude interruption of a conversation can offend a person, so too can the interruption of data communication without an orderly termination process confuse a computer.

The first characteristic to notice about the communication process, therefore, is that error-free communication can be achieved only by following communication protocols. The inability of two entities to communicate directly complicates communication. Consider what happens when one person mails another person a letter:

1. Sender writes or types the letter on paper.
2. Sender inserts the paper in an envelope and labels that envelope with his address and the intended receiver's address.
3. Sender places the envelope at a specific location, where the letter carrier picks it up and places it in a mail bag for transport.
4. The letter carrier takes the mail bag to the local post office branch, where yet another person or persons remove the letter from the bag, sort it, and place it in a mail bag destined for the city specified by the receiver's address.
5. Someone else transports the mail bag to the receiver's city.
6. After the mail bag arrives in the receiver's city, a person sorts the letters in the bag so that the letters can be given to the appropriate letter carriers for each region in the city.
7. The letter carrier transports the letter to the receiver's address.
8. Finally, the receiver removes the letter from its envelope to recover the original message from the sender.

His scenario illustrates several important characteristics of communication.

The diagram of this scenario as shown in figure 3.2 closely resembles the architecture models that this chapter examines later.

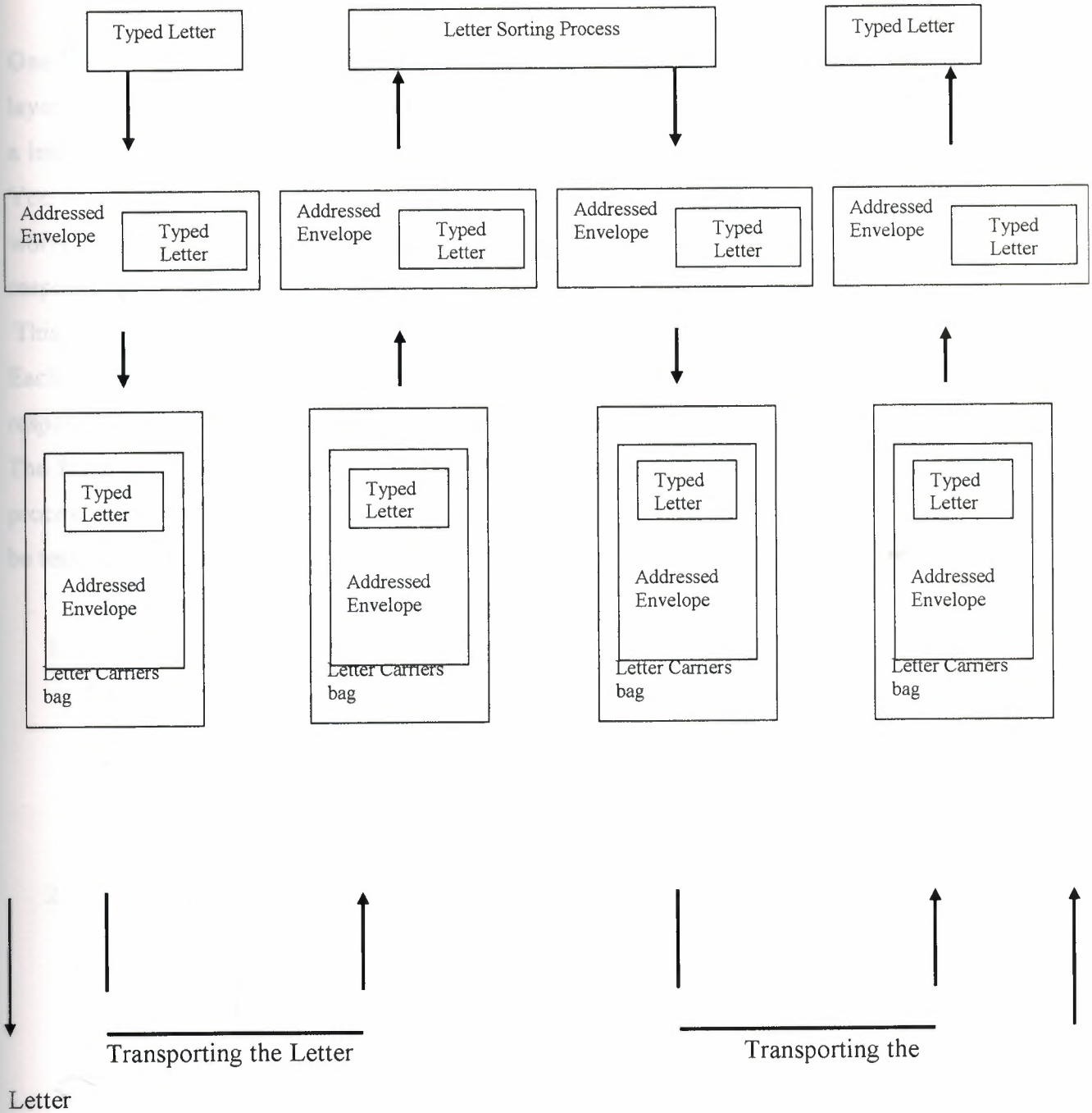


Figure 3.2 A model for communicating by letter.

One feature of the model shown in figure 3.2 is that communication takes place in layers,each of which has a specifically defined area of responsibility. When you want a letter delivered, you need only place it in an envelope and address the envelope. You need not be concerned with the other layers of the process. You don't need to worry about whether the letter is transported by truck or by plane—another layer is responsible for that decision.

This approach to designing a communication system is known as a layered architecture. Each layer has specific responsibilities and specific rules for out those responsibilities , and knows nothing about the procedures the other layers follow. The layer carries out its tasks and delivers the message to the next layer in the process— and that is enough. Quite a few characteristics of layered architectures can be teased out of the preceding example:

1. Layer breaks the communication process into manageable chunks. Designing a small part of a process is much easier than designing the entire process, and it simplifies engineering. A postal customer must establish local rnail room procedures but does not need to devise a complete mechanism for delivering mail to remote sites.

2. A change at one layer does not affect the other layers. The process of addressing an envelope does not change, whether the letier is being shipped by truck, train, or carrier pigeon- New delivery technologies can be introduced without affecting other layers. (This is the chief reason, in fact, for implementing protocols in layers.)

3. When a layer receives a message from an upper layer, the lower layer frequently encloses the message in a distinct package. Letters are placed in mail bags when they are sent across the country, for example. In data communication terms, lower-layer protocols frequently treat upper-layer messages in a similar way, enclosing them in a "data envelope" .The technical term for this process is encapsulation.

4. The protocol at the various layers have the appearance of a stack, and a complete model of a communication architecture is often called a protocol stack.

5. I,ayers can be mixed and matched to achieve different requirements. By adding an overnight label to the envelope, the sender of a letter can have the letter routed through an overnight delivery service rather than standard first-class delivery.

6. Layers follow specific procedures for communicating with adjacent layers. The interfaces between layers must be clearly defined.

7. An address mechanism is the common element that allows letters to be routed through the various layers until it reaches its destination. Sometimes, layers add their own address information. The postal service often adds a bar code to letters, which enables letters to be routed more efficiently within the postal system. Postal addresses have two parts: a city and zip code that enable the postal service to deliver the message to the correct letter carrier, and a street address that enables the letter carrier to deliver the message to the correct recipient.

8. Essentially, each layer at the sender's end communicates with the corresponding layer at the receiver's end. The process of putting the letter in an envelope, for example, has a matching process of opening the envelope in the destination city. The contents of the envelope are examined only by the sender and the receiver, and the contents have no significance at other layers.

9. Errors can occur at any of the layers. For critical messages, error-detecting mechanisms should be in place to either correct errors or notify the sender when they occur.

10. Each of these characteristics has its counterpart in data communication, and the time has come to examine some real data communication models. The first model this chapter considers is the OSI reference model, which is generic enough that it can serve to illustrate general characteristics of data communication models.

3.2 The OSI Reference Model

The International Organization for Standardization (ISO) developed the OSI reference model as a guide for defining a set of open protocols. Although interest in the OSI protocols has waned, the OSI reference model remains the most common standard for describing and comparing protocol suites.

Figure 3.2 shows the seven-layer OSI reference model. Each laver provides a specific type of network service. This chapter describes each of these layers.

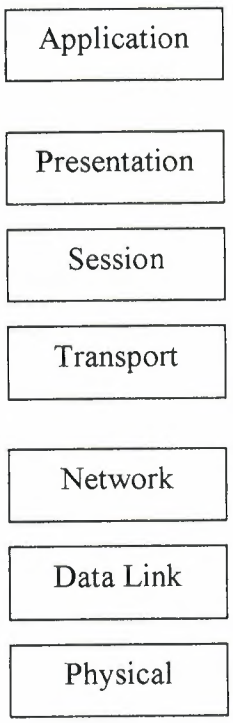


Figure 3.3 The layers of the OSI reference model.

Figure 3.3 illustrates why groups of related protocols are frequently called protocol stacks. To an extent, network designers can build a protocol stack that meets their specific requirements by choosing an appropriate protocol for each layer and stacking them like digital Lego blocks. Layering makes changing the physical network easy, for

example, from Ethernet to token ring without the need for changes at other layers.

3.2.1 The Physical Layer

The physical layer communicates directly with the communication medium and has two responsibilities; sending bits and receiving bits. A binary digit, or bit, is the basic unit of information in data communication. A bit can have only two values, 1 or 0, represented by different states on the communication medium.

Other communication layers are responsible for collecting these bits into groups that represent message data.

Bits are represented by changes in signals on the network medium. Some wire media represent 1s and 0s with different voltages, some use distinct audio tones, and yet others use more sophisticated methods, such as state transitions. (changes from high-to-low or from low-to-high voltages).

A wide variety of media are used for data communication, including electric cable, fiber optics, light waves, radio, and microwave. The medium used can vary—a different medium simply necessitates substituting a different set of physical layer protocols. Thus, the upper layers are completely independent from the particular process used to deliver bits through the network medium.

An important distinction is that the OSI physical layer does not, strictly speaking, describe the media themselves. Physical layer specifications describe how data are encoded into media signals and the characteristics of the media attachment interface, but the specifications do not describe the medium itself. In actual practice, however, many physical layer standards cover characteristics of the OSI physical layer as well as characteristics of the medium.

3.2.2 The Data Link Layer

Devices that can communicate on a network frequently are called nodes.(Other names include station and device.) The data link layer is responsible for providing node-to-node communication on a single, local network. To provide this service, the data link layer must perform two functions. It must provide an address mechanism that enables messages to be delivered to the correct nodes. Also, it must translate messages from upper layers into bits that the physical layer can transmit.

When the data link layer receives a message to transmit it formats the message into a data frame. (You also hear data frames referred to as packets.) Figure 2.4 illustrates the format of a typical frame context of. The sections of a frame are called fields. The fields in the given example are as follows:

1. **Start Indicator.** A specific bit pattern indicates the start of a data frame.
2. **Source Address.** The address of the sending node is also included so that replies to message can be addressed properly.

3. **Destination Address.** Each node is identified by an address. The data link layer of the sender adds the destination address to the frame. The data link layer of the receiver looks at the destination address to identify messages that it should receive.

4. **Control.** In many cases, additional control information must be included. The specific information is determined by each protocol.

5. **Data.** This field contains all data that were forwarded to the data link layer from upper protocol layers.

6. **Error Control.** This field contains information that enables the receiving node to determine whether an error occurred during transmission. A

common approach is cyclic redundancy checksum (CRC), which is a calculated value that summarizes all of the data in the frame. The sending node calculates a checksum and stores it in the frame. The receiver recalculates the checksum. If the receiver's calculated CRC matches the CRC value in the frame, it can safely be assumed that the frame was transmitted without error.

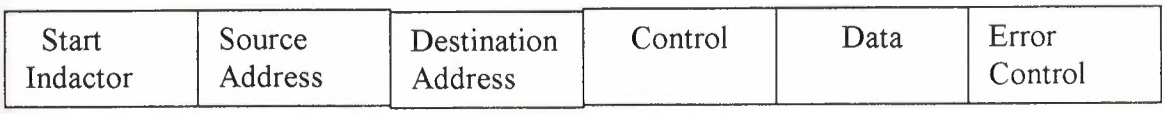


Figure 3.4 An example of a data frame.

Frame delivery on a local network is extremely simple. A sending node simply transmits the frame. Each node on the network sees every frame and examines the destination address. When the destination address of a frame matches the node's address, the data link layer at the node receives the frame and sends it up the protocol stack..

3.2.3 The Network Layer

Only the smallest networks consist of a single, local network. The majority of networks must be subdivided, as shown in figure 3.5, A network that consists of several network segments is frequently called an inter network, or an internet (not to be confused with the Internet).

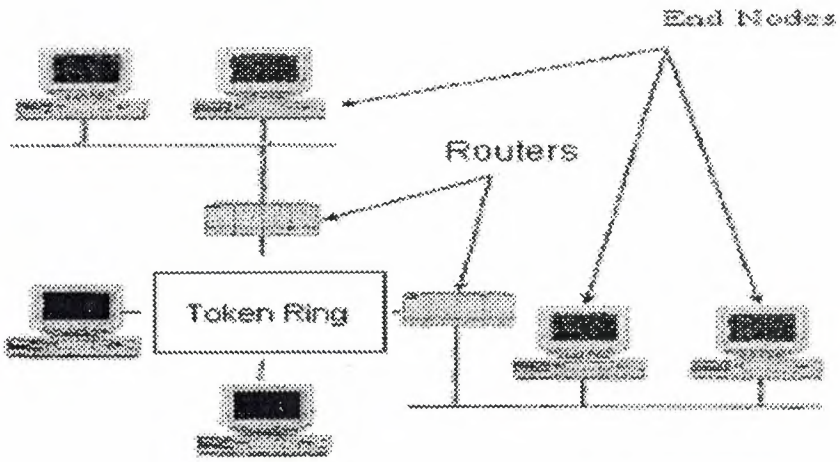


Figure 3.5 An inter network consisting of several networks

These subdivisions can be planed to reduce traffic on the network segment or to isolate remote networks connected by slower communication media, when networks are subdivided, it can no longer be assumed that messages will be delivered on the local network. A mechanism must be put in place to route messages from one network to another.

To deliver messages on an inter network, each network must be uniquely identified by a network address. When it receives a message from upper layers, the network layer adds a header to the message that includes the source and destination network address. This combination of data plus the network layer is called a packet. The network address information is used to deliver a message to the correct network. After the message arrives on the correct network, the data link layer can use the node address to deliver the message to a specific node.

Forwarding packets to the correct network is called routing, and devices that route packets are called routers. As you can see in figure 2.5, an inter network has two types of nodes:

1. End nodes provide user services. End nodes do use a network layer to add network address information to packets, but they do not perform routing. End nodes are sometimes called end systems (the OSI term) or host's (the TCP/IP term).
2. Routers incorporate special mechanisms that perform routing. Because routing is a complex task, routers usually are dedicated devices that do not provide services to end users. Routers sometimes are called intermediate systems (the OSI term) or gateways (the historic TCP/IP term).

The network layer operates independently of the physical medium, which is a concern of the physical layer. Because routers are network layer devices, they can be used to forward packets between physically different networks. A router can join an Ethernet to a token ring network, for example. Routers also are often used to connect a local area network, such as Ethernet, to a wide-area network, such as ATM.

3.2.4 The Transport Layer

All network technologies set a maximum size for frames that can be sent on the network. Ethernet, for example, limits the size of the data field to 1,500 bytes. This limit is necessary for the following two reasons:

1. Small frames improve network efficiency when many devices must share the network. If devices could transmit frames of unlimited size, they might monopolize the network for an excessive period of time. With small frames, devices take turns at shorter intervals, and devices are more likely to have ready access to the network.

2. With small frames, less data must be retransmitted to correct an error. If a message that consists of 100 KB encounters an error of a single byte, the entire 100 KB message must be retransmitted. If the message is divided into 100 frames, each limited in size to 1 KB, a one-byte error requires the retransmission merely of a single 1 KB frame.

One responsibility of the transport layer is to divide messages into fragments that fit within the size limitations establish by the network. At the receiving end, the transport layer reassembles the fragments to recover the original message.

When messages are divided into multiple fragments, the possibility that segments might not be received in the order sent increases. Figure 2.6 illustrates how the network can route packets differently as routers attempt to send each paket by the most efficient available route. When the packets arc received, the transport layer must reassemble the message fragments in the correct order. To enable packets to be reassembled in their original order, the transport layer includes a message sequence number in its header.

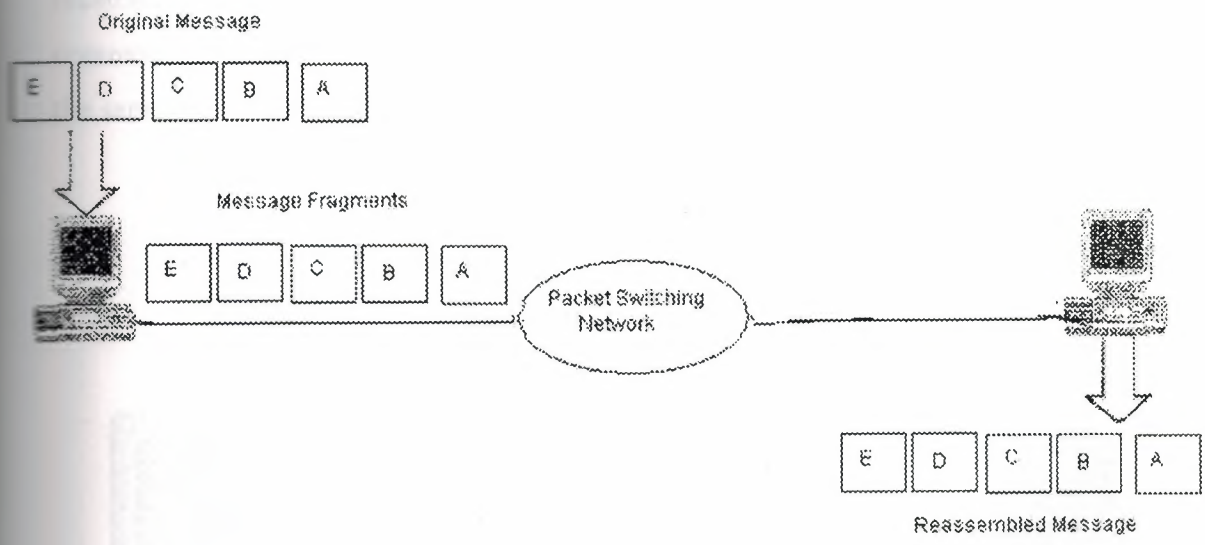


Figure 3.6 The fragmentation and reassembly of a message on a packet switching network.

Most computers are multitasking, running several programs at once. A user's workstation might, for example, be simultaneously running processes to transfer files to another computer, retrieve e-mail, and access a network database. The transport layer is responsible for delivering messages from a specific process on one computer to the corresponding process on the destination computer.

Under the OSI model, the transport layer assigns a service access point (SAP) ID to each packet. (The TCP/IP term for a service access point is port.) The SAP ID is an address that identifies the process that originated the message. The SAP ID enables the transport layer of the receiving node to route the message to the appropriate process. Identifying messages from several processes so that the messages can be transmitted through the same network medium is called multiplexing. The procedure of recovering messages and directing them to the correct process is called demultiplexing. Figure 3.7 illustrates how message multiplexing and demultiplexing work. Multiplexing is a common occurrence on networks, which are designed to enable many dialogs to share the same network medium.

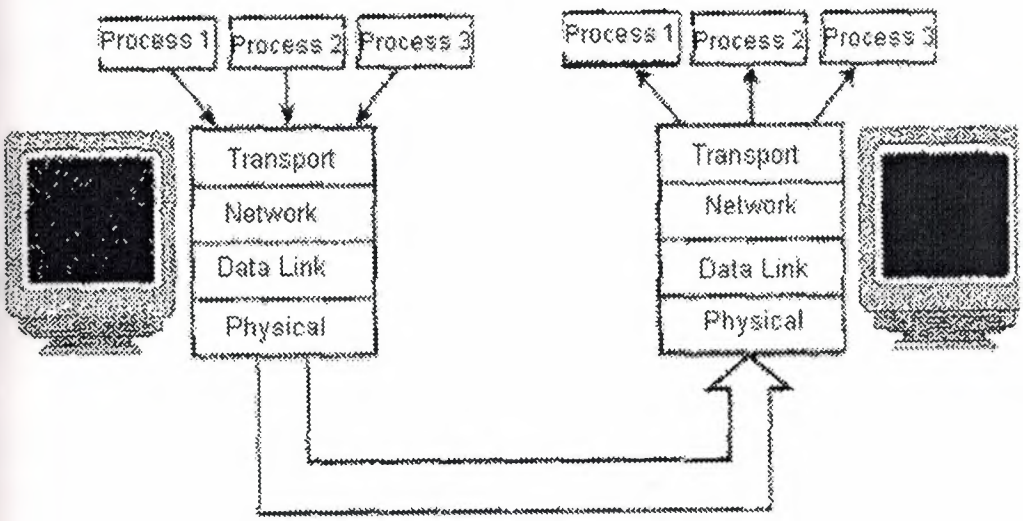


Figure 3.7 Message multiplexing and Demultiplexing.

Multiple protocols can be supported for any given layer, multiplexing and demultiplexing can occur at many layers. Some examples of multiplexing are presented tn the following list:

1. Transport of different Ethernet frame types over the same medium data (link layer).
2. Simultaneous support for NWLink and TCP/IP on Windows NT computers (data link layer).
3. Messages for multiple transport protocols, such as TCP and UDP on TCP/IP systems (transport layer).
4. Messages for multiple application protocols (such as Telnet, FTP, and SMTP) on a UNIX host (session and higher layers).

5. As subsequent chapters discuss layers of the TCP/IP protocol suite, the methods of multiplexing messages at each level are examined. One more responsibility of the transport layer must be examined. Although the data link and network layers can be assigned responsibility for detecting errors in transmitting data, that, responsibility generally is dedicated to the transport layer. Two general categories of error detection can be performed by the transport layer:

6. **Reliable delivery.** Reliable delivery does not mean that errors cannot occur; only that errors are detected if they do occur. Recovery from a detected error can take the form of simply notifying upper-layer processes that the error occurred. Often, however, the transport layer can request the retransmission of a packet for which an error was detected.
7. **Unreliable delivery.** Unreliable delivery does not mean that errors are likely to occur, but rather, indicates that the transport layer does not check for errors. Because error checking takes time and reduces network performance, unreliable delivery often is preferred when a network is known to be highly reliable, which is the case with the majority of local area networks. Unreliable

delivery generally is used when each packet contains a complete message, whereas reliable delivery is preferred when messages consist of large numbers of packets. Unreliable delivery is often called datagram delivery, and independent packets transmitted in this way frequently are called datagrams.

Assuming that reliable delivery is always preferable is a common mistake among new students of network protocols. Unreliable delivery actually is preferable in at least two cases: when the network is fairly reliable and performance must be optimized, and when entire messages are contained in individual packets and loss of a packet is not a critical problem.

3.2.5 The Session Layer

The session layer is responsible for dialog control between nodes. A dialog is a formal conversation in which two nodes agree to exchange data.

Communication can take place in three dialog modes; illustrated in figure 3.8:

1. **Simplex.** One node transmits exclusively, while another exclusively receives.
2. **Half-duplex.** Only one node can send at a given time, and nodes take turns transmitting.
3. **Full-duplex.** Node can transmit and receive simultaneously. Full-duplex communication typically requires some from of flow control to ensure that neither device sends faster than the other device can receive.

Sessions enable nodes to communicate in an organized manner. Each session has three phases:

1. Connection establishment. The nodes establish contact. They negotiate the rules of communication, including the protocols to be used and communication parameters.

2. **Data transfer.** The nodes engage in a dialog to exchange data.

3. **Connection release.** When the nodes no longer need to communicate, they engage in an orderly release of the session.

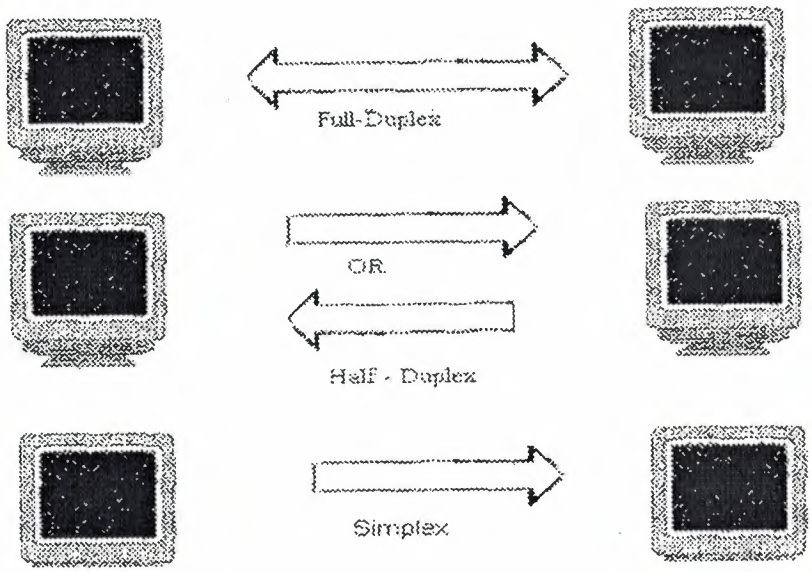


Figure 3.8 Communication dialog nodes.

Steps 1 and 3 represent extra overhead for the communication process. This extra overhead might be undesirable for brief communication. Consider, for example, the communication associated with network management. When devices are managed on a network, they periodically send out brief status reports that generally consist of single frame messages. If all such messages were sent as part of a formal session, the connection establishment and release phases would transfer far more data than the message itself.

In such situations, communicating using a connectionless approach is common. The sending node simply transmits its data and assumes availability of the desired receiver.

A connection-oriented session approach is desirable for complex communication. Consider transmitting a large amount of data to another node.

Without formal controls, a single error anytime during the transfer would require resending of the entire file. Alter establishing a session, the sending and receiving nodes cart agree on a checkpoint procedure. If an error occurs, the sending node must retransmit only the data sent since the previous checkpoint. The process of managing a complex activity is called activity management.

3.2.6 The Presentation Layer

The presentation layer is responsible for presenting data to the application layer. In some cases, the presentation layer directly translates data from one format to another. IBM mainframe computers use a character encoding scheme called EBCDIC, whereas virtually all other computers use the ASCII encoding scheme. If data are being transmitted from an EBCDIC computer to an ASCII computer, for example, the presentation layer might be responsible for translating between the different character sets. Numeric data is also represented quite differently on different computer architectures and must be converted when transferred between different machine times.

A common technique used to improve data transfer is to convert all data to a standard formal before transmitting the data. This standard format probably is not the native data format of any computer. All computers can be configured to retrieve standard format data, however, and convert it into their native data formats. The OS1 protocol standards define Abstract Syntax Representation, Revision 1 (ASN. 1) as a standard data syntax for use at the presentation layer. Although the TCP/IP protocol suite does not formally define a presentation layer, a

protocol that serves a similar function is External Data Representation (XDR), which is used with the Network File System (NFS).

Other functions that may correspond to the presentation layer are data encnptioıı/ decryption and compression./decompression.

The presentation layer is the least frequently implemented of the OSI layers. Few protocols have been formally defined for this layer. In the majority of cases, network applications perform the functions that might be associated with the presentation layer.

3.2.7 The Application Layer

The application layer provides the services user applications needed to communicate through the network. Here are several examples of user application layer services:

1. **Electronic mail transport.** A protocol for handling electronic mail can be used by a wide variety of applications. Application designers who use the e-mail protocols do not need to invent their own e-mail handlers. Also, applications that share a common e-mail interface can exchange messages through the e-mail message handler.
2. **Remote file access.** Local applications can be given the capability to access files on remote nodes.
3. **Remote job execution.** Local applications can be given the capability to start and control processes on other nodes.

4. **Directories.** The network can offer a directory of network resources, including logical node names. This directory enables applications to access network sources by logical names rather than abstract numeric node IDs.

5. **Network management.** Network management protocols can enable various applications to access network management information.

You frequently encounter the term application program interface (API) used in conjunction with application layer services. An API is a set of rules that enables user-written applications to access the services of a software system. Developers of program products and protocols frequently provide APIs, which enable programmers to easily adapt their applications to use the services the products provide. A common UNIX API is Berkeley Sockets, which Microsoft has implemented as Windows Sockets.

3.3 Characteristics of Layered Protocols

The OSI reference model illustrates several characteristics of layered protocol stacks. When a device transmits data to the network, each protocol layer processes the data in turn. Figure 3.9 illustrates the steps involved for the sending and receiving devices.

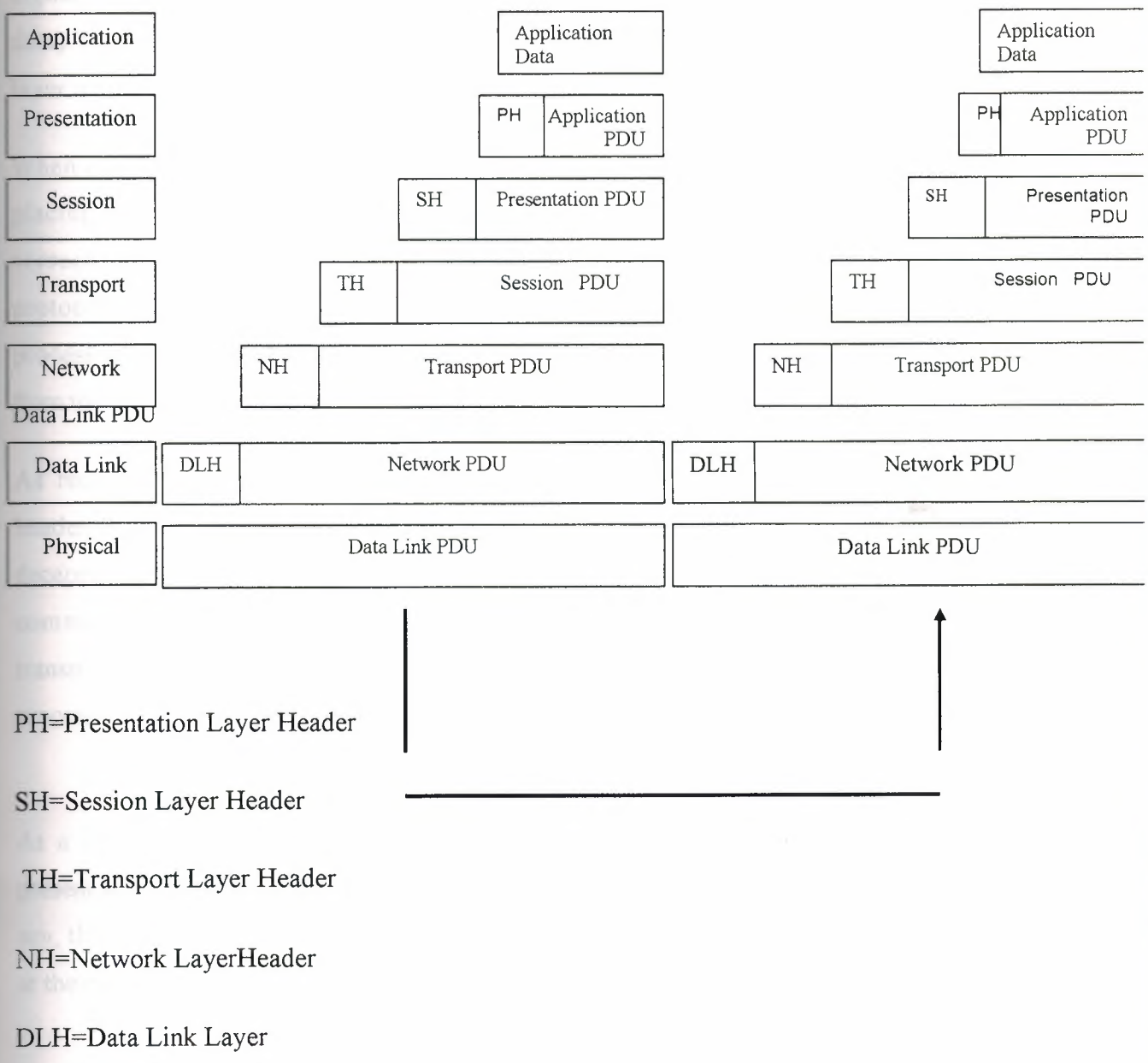


Figure 3.9 Headers and the OSI protocol layers.

Consider the network layer for the sending device. Data to be transmitted are received from the transport layer. The network layer is responsible for routing and must add its routing information to the data. The network layer information is added in the form of a header, which is appended to the beginning of the data. OSI terminology uses the term protocol data unit (PDU) to describe the combination of the control information for a layer with the data from the next higher layer. As figure 3.9 shows, each layer appends a header to the PDU that it receives from the

next higher layer. The data field for each layer consists of the PDU for the next higher layer.(The data link layer also adds a trailer that consists of the error control data).The physical layer does not encapsulate in this manner because the physical layer manages data in bit form.

When a layer adds its header to the data from a higher layer, the process resembles placing mail in an envelope. The envelope can be used to deliver the data to the correct location, where the envelope is opened and the data recovered. When a protocol uses headers or trailers to package the data from another protocol, the process is called encapsulation. It is said that the network layer encapsulates the data from the transport layer.

As received data pass up the protocol stack, each layer strips its corresponding header from the data unit. The process of removing headers from data is called decapsulation.. This mechanism enables each layer in the transmitting device to communicate with the corresponding layer in the receiver. Each layer in the transmitting device communicates with its peer layer in the receiving device, in a process called peer-to-peer communication.

As a consequence of the encapsulation/decapsulation process, the identical PDU is present at corresponding layers in the sending and receiving protocol stacks. That is to say, the network layer PDU at the sending node is identical to the network layer PDU at the receiving node.

3.4 The Internet Model

The protocol architecture for TCP/IP currently is defined by the IETF, which is responsible for establishing the protocols and architecture for the Internet The model dates back to the ARPAnet and often is called the DoD model, The DoD protocol architecture predates the OSI reference model, which was first described in 1979. As such, unambiguous mapping of the DoD model to the OSI reference model is impossible to achieve. Figure 3.10 illustrates the four-layer Internet model, mapping the Internet model layers as closely as possible to the OSI reference model.

The network access layer is responsible for exchanging data between a host and the network and for delivering data between two devices on the same network. Node physical addresses are used to accomplish delivery on the local network. The DoD architecture was designed with the intent of using prevailing network standards, and TCP/IP has been adapted to a wide variety of network types, including circuit-switching(for example, X.21), packet switching {such as X.25), Ethernet, the IEEE 802.x protocols. ATM, and frame relay. Data in the network access layer encode Ether Type information that is used to demultiplex data associated with specific upper-layer protocol stacks.

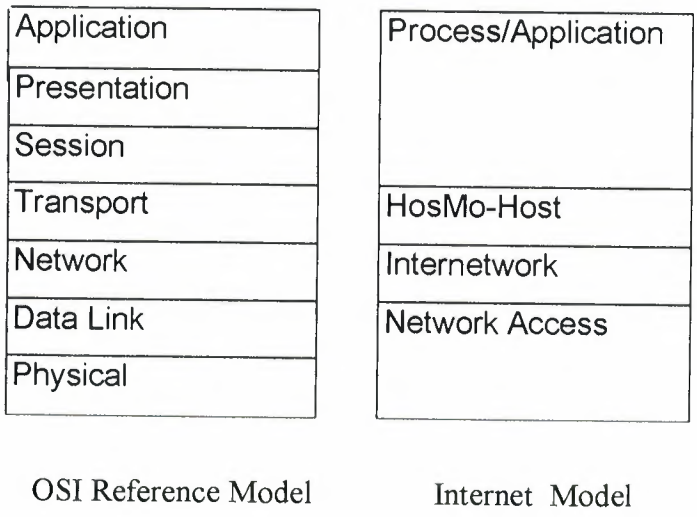


Figure 3.10 A comparision of Internet protocol model and the OSI reference model.

The internet layer corresponds to the OSI network layer and is responsible for routing messages through inter networks. Devices responsible for routing messages between networks are called gateıoays in TCP/IP terminology, although the term router is also used with increasing frequency. The TCP/IP protocol at this layer is the internet protocol (IP). In addition to the physical node addresses utilized at the network access layer, the IP protocol implements a system of logical host addresses called IP addresses. The IP addresses are used by the internet and higher layers to identify devices and to perform inter network routing. The address resolution protocol (AKP) enables IP to identify the physical address that matches a given IP address.

The host-to-host compares closely to the OSI transport layer and is responsible for end-to-end data integrity. Two protocols are employed at this layer: transmission control protocol (TCP) and user datagram protocol (UDP). TCP provides reliable, full-duplex connections and reliable service by ensuring that data is present when transmission results in an error. Also, TCP enables hosts to maintain multiple, simultaneous connections. UDP provides unreliable (datagram) service that enhances network throughput when error correction is not required at the host-to-host layer.

The process /application layer spans the functions of three layers of the OSI reference model: session, presentation, and application. Not surprisingly, therefore, a wide variety of protocols are included in this layer of the DoD model. Examples of protocols at this layer include the following:

1. **File Transfer Protocol** (FTP). Performs basic file transfers between hosts.
2. **Telnet.** Enables users to execute terminal sessions with remote hosts.
3. **Simple Mail Transfer Protocol (SMTP).** Supports basic message delivery.
4. **Simple Network Management Protocol (SNMP).** A protocol that is used to collect management information from network devices.
5. **Network File System** (NFS). A system developed by Sun Microsystems that enables computers to mount drives on remote hosts and operate them as if they were local drives.

Some of these applications encompass functions from several layers of the OSI reference model. NFS. For example, enables hosts to maintain a session (session layer), agree on a data representation (presentation layer), and operate a network file system application layer).

3.5 Delivering Data Through Inter networks

The way data are delivered through inter networks is involved enough that it deserves more thorough discussion. It involves several topics:

1. Methods for carrying multiple data streams on common media, a technique called multiplexing.
2. Methods for switching data through paths on the network.
3. Methods for determining the path to be used.

3.5.1 Multiplexing

LANs typically operate in base band mode, which means that a given cable is carrying a single data signal at any one time. The various devices on the LAN must take turns using the medium. This generally is a workable approach for LANs. Because LAN media offer high performance at low cost.

Long-distance data communications media are expensive to install and maintain, and it would be inefficient. if each media path could support only a single data srieam. Imagine the expense if it was necessary to equip each telephone with its own communication satellite or transatlantic cable to enable it to connect to Europe. WAN's, therefore, tend to use broadband media, which can support two or more simultaneous data streams. Increasingly, as LANs are expected to carry more and different kinds of data, broadband media are being considered for LANs as well.

To enable many data streams to share a high-bandwidth medium, a technique called multiplexing is employed. Figure 3-11 illustrates one method of multiplexing digital signals. The signal-carrying capacity of the medium is divided into time slots, with a time slot assigned to each signal, a technique called time-division-multiplexing

(TDM). Because the .sending arid receiving devices are synchronized to recognize the same time slots, the receiver can identify each data stream and re-create the original signals.

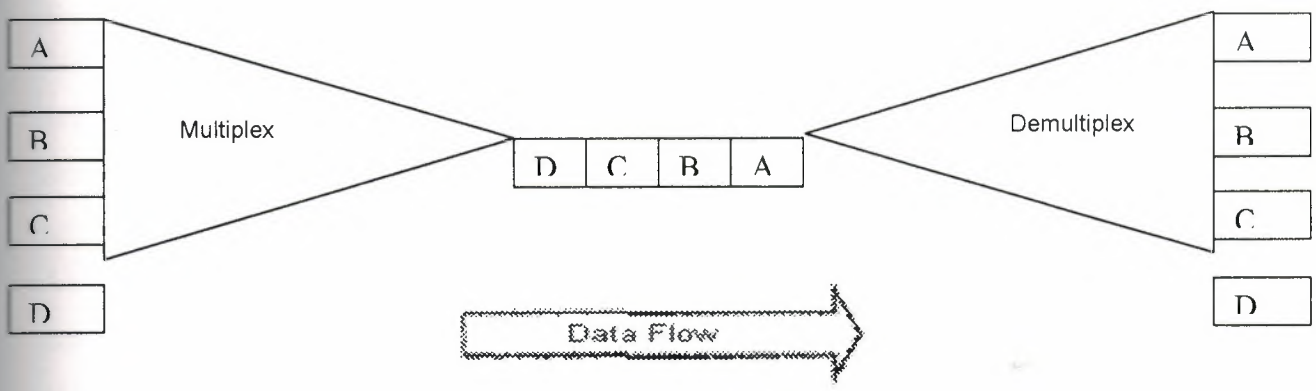


Figure 3.11 Time division multiplexing.

The sending device, which places data into the time slots, is called a multiplexer or mux.The receiving device is called a demultiplexer or demux.

TDM can be inefficient. İf a data stream falls silent, its time slots are not used and the media bandwidth is underutilized. Figure 3.12 depicts a more advanced technique, stactistical time-division multiplexing (stat-TDM). Time slots still are used, but some data streams are allocated more time slots than others. An idle channel is allocated no time slots at all. A device that performs statistical TDM often is called a stat-MUX.

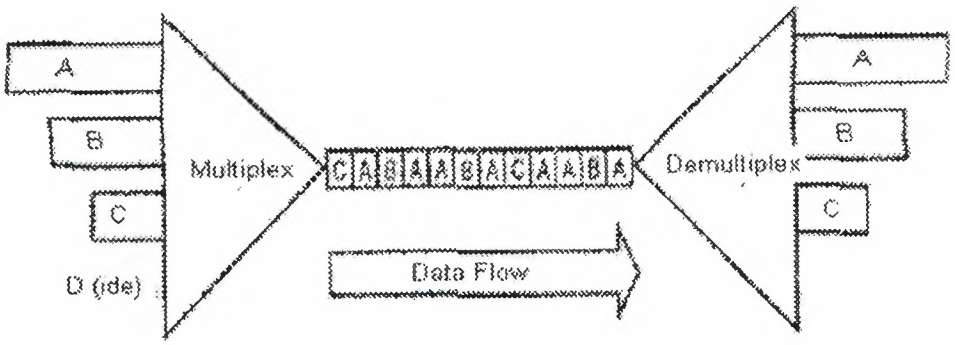


Figure 3.12 Statistical time division multiplexing.

3.5.2 Switching Data

On an inter network, data units must be switched through the various intermediate devices until they are delivered to their destinations. Two contrasting methods of switching data are commonly used: circuit switching and packet switching. Both are used in some form by protocols in common use. (A third, message switching, is useful in some situations).

3.5.2.1 Circuit Switching

Circuit switching is illustrated in figure 3.13. When two devices negotiate the start of a dialog, thev establish a path—called a circuit—-through the network, along with a dedicaied bandwidth through the circuit. After establishing the circuit, all data for the dialog flow through that circuit. This approach resembles a telephone connection in which a voice circuit is established to enable the telephones at the end-points to communicate. The chief disadvantage of circuit switching is that when communication takes place at less than the assigned circuit capacity,

bandwidth is wasted. Also, communicating devices cannot take advantage of other, less busy paths through the network unless the circuit is reconfigured.

Circuit switching does not necessarily mean that a continuous, physical pathway exists for the sole use of the circuit. The message stream may be multiplexed with other message streams in a broadband circuit. In fact, sharing of media is the more likely case with modern telecommunications. The appearance to the end devices, however, is that the network has configured a circuit dedicated to their use.

End devices benefit greatly from circuit switching. Since the path is pre-established, data travel through the network with little processing in transit. And, because multi-part messages travel sequentially through the same path, message segments arrive in order and little effort is required to reconstruct the original message. For that reason, a form of circuit switching was used to design the new. Leading-edge technology ATM.

3.5.2.2 Packet Switching

Packet switching takes a different and generally more efficient approach to switching data through networks. In the late 1960s, packet switching was a new concept that the DoD sought to investigate in the early ARPA network research- As shown in figure 3.14, messages are broken into sections called packets, which are routed individually through the network. At the receiving device, the packets are reassembled to construct the complete message. Messages are divided into packets to ensure that large messages do not monopolize the network. Packets from several messages can be multiplexed through the same communication channel. Thus, packet switching enables devices to share the total network bandwidth efficiently.

Two variations of packet switching can be employed:

Datagram services treat each packet as an independent message. The packets (also called datagrams) are routed through the network using the most efficient

1. Route currently available, enabling the switches to bypass busy segments and use underutilized segments. Datagrams frequently are employed on LANs, and network layer protocols are responsible for routing the datagrams to the appropriate destination. Datagram service is called unreliable, not because it is inherently flawed but because it does not guarantee delivery of data. Recovery of errors is left to upper-layer protocols. Also, if several messages are required to construct a complete message, upper-layer protocols are responsible for reassembling the datagrams in order. Protocols that provide datagram service are called connectionless protocols.

2. Virtual circuits establish a formal connection between two devices, giving the

3. Appearance of a dedicated circuit between the devices. When the connection is

4. Established, issues such as message sizes, buffer capacities, and network paths are considered and mutually agreeable communication parameters are selected. A virtual circuit defines a connection, a communication path through the network, and remains in effect as long as the devices remain in communication. This path functions as a logical connection between the devices. When communication is over, a formal procedure releases the virtual circuit because virtual circuit service guarantees delivery of data, it provides reliable delivery service. Upper-layer protocols need not be concerned with error detection and recovery. Protocols associated with virtual circuits are called connection-oriented.

For the TCP/IP protocol suite, datagram service is provided by the User Datagram Protocol(UDP).The vast majority of system rely on the Transmission Control Protocol (TCP) to provide reliable delivery.

3.5.3 Bridges, Routers, and Switches

Data can be routed through an inter network using the following three types of information:

1. The physical address of the destination device, found at the data link layer. Device that forward messages based on physical address generally are called bridges.
2. The address of the destination network, found at the network layer. Device that use network address to forward message usually are called routers, although the original name, still commonly used in the TCP/IP world, is gateway.
3. The circuit that has been establish for a particular connection. Device that route message based on assigned circuits are called switches.

The following sections discuss each of these devices, respectively.

3.5.3Bridges

Bridges build and maintain a database that lists known addresses of devices and how to reach those devices. When it receives a frame, the bridge consults its database to determine which of its connections should be used to forward the frame. Figure 2.15 illustrates the process in terms of the OSI reference model. A bridge must implement only the physical and data link layers of the protocol stack.

Bridges are fairly simple devices. They receive frames from one connection and forward them to other connections known to be en route, to the destinations. When more than one route is possible, bridges ordinarily cannot determine which route is most efficient. In fact, when multiple routes are available, bridging can result in frames simply traveling in circles. Having multiple paths available on the network is desirable, however, so that failure of one path does not stop the

network. With Ethernet a technique called the spanning-tree algorithm enables bridges networks to contain redundant paths.

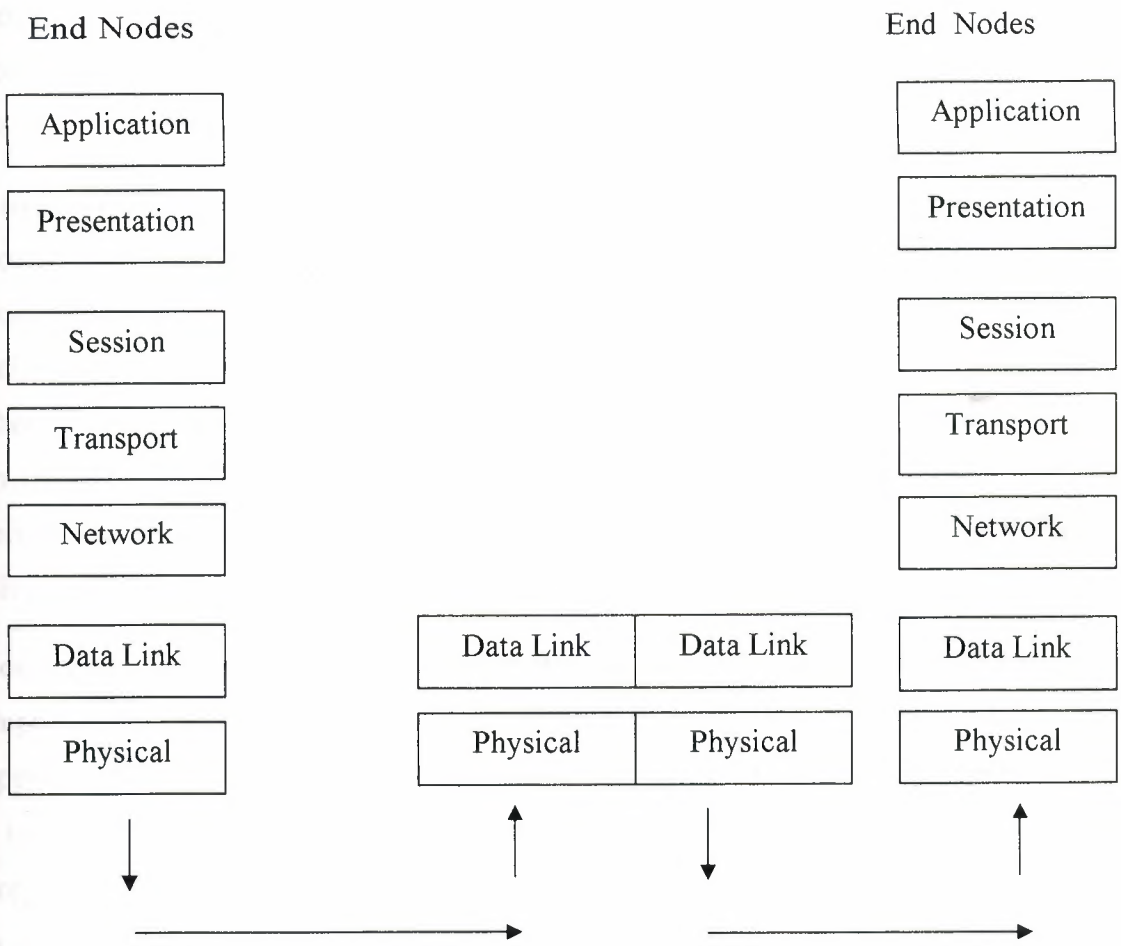


Figure 3.13 The protocol stack model for bridging.

Token ring uses a different approach to bridging. When a device needs to send to another device, it goes through a discovery process to determine a route to the destination. The routing information is stored in each frame transmitted and is used by bridges to forward the frames to the appropriate networks. Although this actually is a data link layer function, the technique token ring uses is called source routing. Notice in figure 2.15 that the bridge mast implements two protocol slacks,

one for each connection. Theoretically, these stacks could belong to different protocols, enabling a bridge to connect different types of networks. Translating data from the data link layer of an Ethernet to which operate at the data link layer of a token ring is difficult(but not impossible). Bridges, therefore. Which operate at the data link layer, generally can join only networks of the same type. You see bridges employed most often in networks that are all Ethernet or all token ring. A few bridges have been marketed that can bridge networks that have different data link layers.

Because bridges ignore network addresses, they cannot be used to create large inter networks consisting of multiple, distinctly identified networks. In many cases, network technologies impose limits on the sizes of networks. TCP/IP, for exarnple, organizes addresses into classes, each of which can support a specific maximum number of nodes. A class C address, for example, can support only 254 nodes on a given network. To expand beyond that limit, additional class C networks must be established in an inter network. Therefore, there is always a limit on the extent to which a bridged network can expand.

In recent years, new LAN devices dubbed switches have been strategically deployed to improve network bandwidth. A LAN switch is essentially a bridge that is equipped with large numbers of ports. Additionally LAN switches use techniques that improve the rapidity with which frames can be forwarded. They often, for example, do not bother to buffer the entire packet before forwarding. Rather, as soon as the destination physical address is identified, a LAN switch can begin to forward the packet immediately without waiting for the end of the frame. Consequently, a LAN switch cannot use the error control field to determine whether the frame has been scrambled. In relatively error-free LAN environments, this lack of an integrity check is typically an acceptable trade off for greater efficiency. Because they rely on data link layer addresses, LAN switches, like bridges, cannot be used to establish inter networks.

3.5.3.2 Routing

A different method of parh determination can be employed using data found at the network layer. At that layer, networks are identified by logical network identifiers. This information can be used to build a picture of the network. This picture can be used to improve the efficiency of the paths that are chosen. Devices that forward data units based on network addresses are called routers. Figure 3.16 illustrates a protocol stack model for routing.

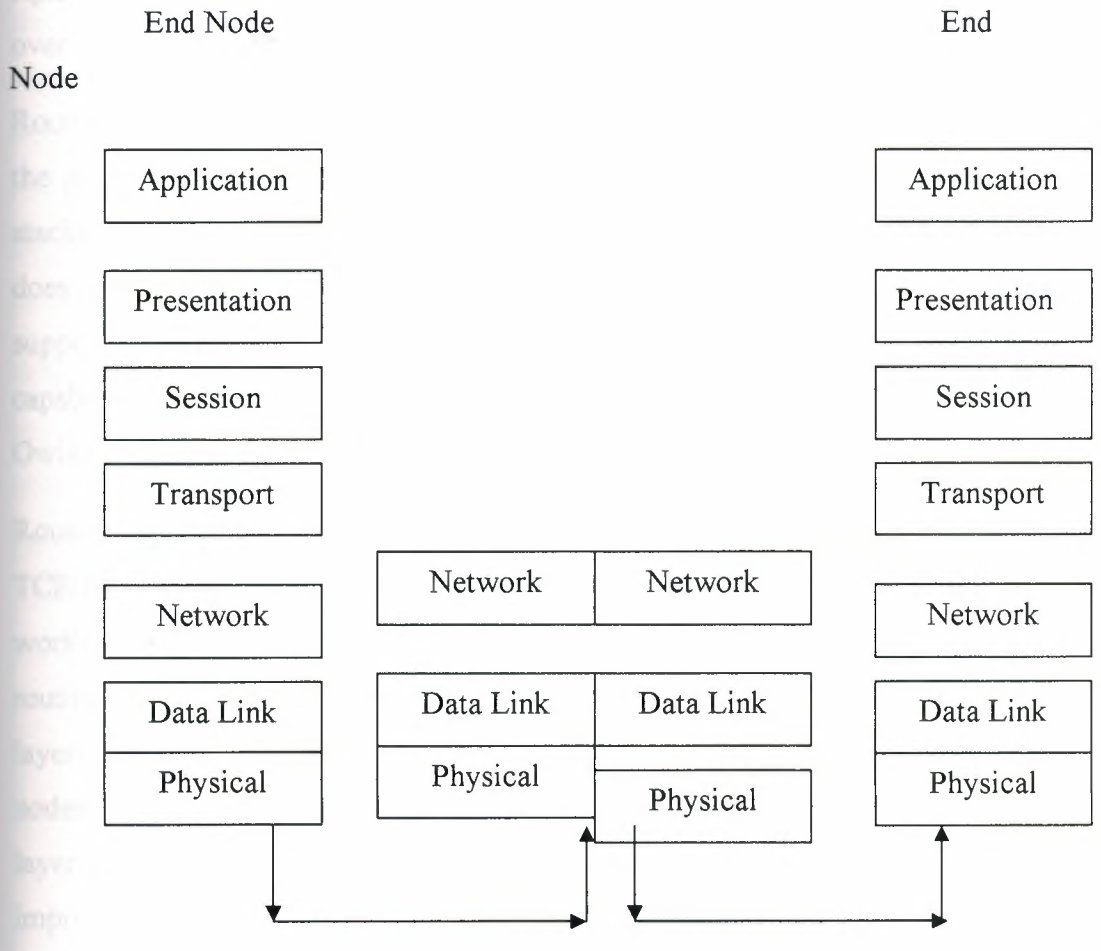


Figure 2.16 A protocol stack model for routing.

With TCP/IP, routing is a function of the internet layer, here is just to briefly illustrating one technique. Figure 2.17 illustrates a network. The hop count describes the number of networks that must be crossed

between two communication end nodes. A wide variety of paths could be identified between A and E:

1. A-B-C-E (5 hops)
2. A-E (3 hops)
3. A-D-E (4 hops)

By this method, A-E is the most efficient route. This assumes that all of the paths between the routers provide the same rate of service. A simple hop-count algorithm would be misleading if A-D and D-E were 1.5 Mbps lines while A-E was a 56 Kbps line. Apart from such extreme cases, however, hop-count routing is a definite improvement over no route planning at all.

Routing operates at the network layer. By the time data reach that layer, all evidence of the physical network has been shorn away. In figure 3.16, therefore, both protocol stacks in the router can share a common network layer protocol. The network layer does not know or care if the network is Ethernet or token ring. Therefore, each stack can support different data link and physical layers. Consequently, routers possess a capability-fairly rare in bridges-to forward traffic between dissimilar types of networks. Owing to that capability, routers often are used to connect LANs to WANs.

Routers can be built around the same protocol stacks that are used at the end-nodes, TCP/IP networks can use routers based on the same IP protocol employed at the workstation. It is not required; however, that routers and end-nodes use the same routing protocol. Because network layer protocols need not communicate with upper-layer protocols, different protocols can be used in routers than are used in the end nodes. Commercial routers (such as Cisco and Wellfleet) employ proprietary network layer protocols to perform routing. These custom protocols are among the keys to the improved routing performance provided by the best routers.

3.5.3.3 Switching

Circuit-based networks operate with high efficiency because the path is established once, when the circuit is established. In figure 3. 18. each switch maintains a table that records how data from different circuits should be switched. Switching is typically performed by lower-level protocols to enhance efficiency, and is associated most closely uilh the data link layer.

Very seldom do real circuits resemble figure 2. 18, with one cable dedicated to each communication circuit. More commonly, many circuits are multiplexed onto a single media channel, with the multiplexing hidden from the end nodes. Such circuits may then be referred to as virtual circuits; for they appear to the end nodes to be physical channels, whereas they actually share network bandwidth with many other virtual circuits.

When data are routed through inter networks using protocols higher than the network layer, the intermediate devices commonly are called gateways. This commonly confuses newcomers to TCP/IP, for historically the TCP/IP term for router has been gateway. You can expect to see the term router used increasingly in discussions of TCP/IP networks, but gateway remains in widespread use as a name for devices that perform routing with the IP protocol.

Chapter4:TCP/IP APPLICATIONS

This chapter examines several common TCP/IP applications. Some, like FTP and Telnet, are tools that end users frequently access. Others, although not often seen by users, are mainstays of TCP/IP networking. Specifically, this chapter examines the following applications:

1. File Transfer Protocol (FTP)
2. Trivial File Transfer Protocol (TFTP)
3. Telnet
4. Simple Mail Transport Protocol (SMTP)
5. Simple Network Management Protocol (SNMP)
6. Network File System (NFS)

Incidentally, DNS name servers also function at the process/application level. Operation of DNS is described above.

4.1 File Transfer Protocol

FTP is both a protocol and a program that can be used to perform basic file operations on remote hosts and to transfer files between hosts. As a program, FTP can be operated by users to do file tasks manually. FTP also can be used as a protocol by applications that require its file services. FTP is a secure, reliable application. To ensure reliability, FTP operates over TCP. Users who access a host through FTP undergo an authentication login that may be secured with usernames and passwords. These usernames and passwords can be tied into the host's security, enabling administrators to restrict access as required.

Although secure, FTP sends passwords as clear text, which can pose problems.

4.1.1 Anatomy of FTP

As figure 4.1 shows, FTP includes client and server components. A host that makes its file system available to users must run an application that operates as an FTP server. Users who access the FTP server must run FTP client software on their computer.

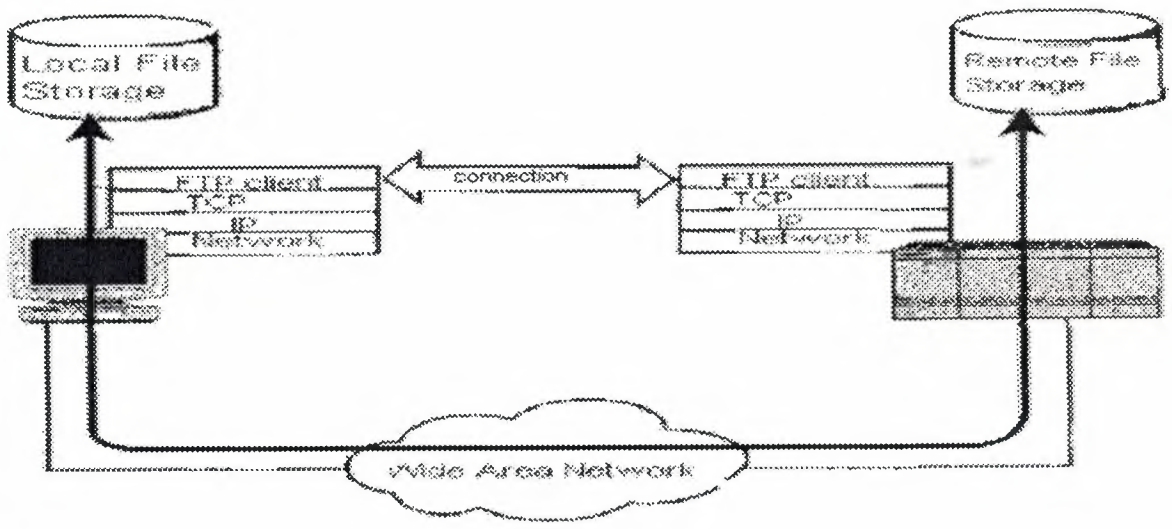


Figure 4.1 Anatomy of an FTP session

When the FTP client opens a connection to an FTP server, a logical pipeline (a virtual circuit defined by two sockets) is established between the client and server hosts. This pipeline enables the FTP components to communicate. The end result is very much like mounting the remote file system for limited local file access. You cannot execute remote files as programs, but you can list directories, type file contents, manipulate local directories, and copy files between the hosts.

A word about capitalization is in order. UNIX commands, directories, and file names are case-sensitive, and the command GET is not interchangeable with get. Therefore,

authors of UNIX books conventionally typeset this and other commands entirely in lowercase.

Windows NT recognizes upper- and lowercase characters in file names, but NT commands are not case sensitive. It does not matter whether you type ftp, FTP, or f Tp. When you are interacting with a remote host, however, you must adhere to the case conventions on that host. Because UNIX hosts expect FTP commands and options to be in lower case, I have stuck to lower case in this chapter.

4.1.2 Using FTP

Chances are high that you connect your Windows NT computers to the Internet in some way. When that happens, you leave the realm of Microsoft networking and need to use FTP to perform remote file operations.

Until fairly recently, FTP has been a text-only application, used from the command line. The new popularity of the Internet, combined with the wide availability of graphical user interfaces, has lead to the development of Windows-based applications that provide a point-and-click interface to FTP.

Windows NT, however, ships with a command-line **ftp** utility only. Fortunately, that is all you need. After doing so, you can use **ftp** as it ships with Windows NT to access FTP sites that can supply you all of the tools you need to set up your Windows NT Internet environment, including the previously mentioned graphical interface to **ftp.**

A World Wide Web browser can be used to retrieve files from FTP sites. Simply specify ftp as the protocol in the URL, for example:

ftp://ds.internic.net

4.1.3 Example of an FTP Session

This section presents an example of an **ftp** session. The example is designed to show you how to retrieve an RFC document from the InterNIC server DS.INTERNIC.NET.

Before beginning, you must establish an Internet connection, using either a dial-in service or a direct network connection.

4.1.3.1 Starting FTP

You can start an FTP session by simply opening a command prompt and typing the command **ftp** as in this dialog (text entered by the user is in bold):

C:\ftp

ftp>

The prompt changes from C:\ to ftp>. At this point, only ftp commands are *accepted.*

4.1.3.2 GettingHelp

To obtain a list of ftp commands, use the? Command:

```
ftp>?
Commands may be abbreviated.                    Commands are:
1          delete     literal    prompt     send
?          debug      ls         put        status
append     dir        mdelete    pwd        trace
ascii      disconnect mdir       quit       type
Bell       get        mget       quote      user
binary     glob       mkdir      recv       verbose
bye        hash       mis        remotehelp
Cd         help       mput       rename
close      led        open       rmdir
ftp>
```

ftp betrays its UNIX origins in many ways. Many of the file and directory management commands originated in the UNIX environment. Some will be familiar to DOS and Windows users (no points for guessing which OS originated the commands). Others, such as pwd (print working directory) and is (list files) are unfamiliar to DOS users, but quite understandable after they are explained.

You can obtain brief help on any of these commands by typing help followed by the command for which you seek advice. The descriptions aren't very elaborate, but they can help, as in this example:

```
ftp> help ls
ls      nlist      contents      of      remote      directory
```

4.1.3.3 Opening a Session

After starting ftp, you can open a session with an FTP server by using the open command, as follows:

```
ftp>    open    ds.internic.net

Connected to ds.internic.net
220-  InternNic Directory and  Database Services
220-
220-Welcome to InterNIC Directory and Database Service provided by AT&T
220-These services are partially supported through a cooperative  agreement
220-with the National Science  Foundation.
220-
220-Your comments and suggestion for improvement are welcome, and can be
220-mailed to admin@ds.internic.net.
220-
```

220-AT&T MAKES NO WARRANTY OR GUARANTEE,OR PROMISE,EXPRESS OR
IMPLIED,
220-CONCERNING THE CONTENT OR ACCURACY OF THE INTERNIC DIRECTORY
ENTRIES
220-AND DATABASE FILES STORED AND MAINTAINED BY AT&T.AT&T
EXPRESSLY
220-DISCLAIMS AND EXCLUDES ALL EXPRESS WARRANTIES AND IMPLIED
WARRANTIES
220-OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
220-
220- **********
220-
220-DSO will be rebooted every Monday morning between 8:00 AM and 8:30 AM est.
220- Please use DS1 or DS2 during this period
220-
220- **********
220-
User(ds.internic.net:(none)):

You can start ftp and open a session with an FTP server by including the domain name for the server as a command-line parameter. The session with InterNIC could also have started using this command:

C:\ftp ds.internic.net

4.1.3.4 Logging in to Anonymous

Like many FTP servers, ds.internic.net accepts anonymous logins. Any user may log in with limited access rights by using the username anonymous and entering a password. By convention, most systems request that users enter their e-mail addresses as their passwords. The login process is shown below:

User (ds.internic.net:(none)): **anonymous**
331 Guest login ok, send ident as password.
Password: (password does not echo)
230 Guest login ok, access restrictions apply.
ftp>

4.1.3.5 Changing Directories

You now are logged in to the root directory of ds.internic.net. At any given time, you will be working with two working directories: one on the FTP server and one on your local computer. You can change these directories if you need to determine where to copy files from and to.
Two commands are available for changing directories:

1. **cd.** Changes your working directory on the FTP server.

2. **led.** Changes your current directory on your local computer.

At any time, you can view your current directory on the FTP server by using the pwd (print working directory) command:

ftp> **pwd**
257 " / " is the current directory

When you complete a login to an ftp server, you enter the realm of that server's operating system, ds.internic.net is a Sun system running on SunOS UNIX—the majority of FTP servers run UNIX—and you must apply UNIX conventions to the names of files and directories.
Remember that case is significant in UNIX. You cannot retrieve the file rfcl 800.txt with the command parameter RFC1800.TXT. Also, UNIX subdirectories are delimited with forward slashes (/) rather than the back slashes (\) with which DOS and Windows users are familiar.

Similarly, in UNIX, case matters in directory names. If a directory named /pub/ Libraries/Programs, you must be sure to match case when you enter the subdirectory names.

RFC files are stored in the / rf c directory. You can access files by specifying the file name with a directory path, or you can change your working directory to /rf c. To change your working directory on the server, use the cd (change directory) command:

```
ftp> cd /rfc
250 CWD command succesful
ftp> pwd
257 "/rfc" is the current directory
```

To change the current directory on your local computer, use the lcd command. To store files in \docs, enter the command:

```
ftp> led \docs
Local directory now c:\docs
```

Figure 4.2 illustrates a directory tree along with some examples of commands for moving around:

1. To move up the directory tree (to the parent directory of the current one) enter the command **cd** .. or **cdup.** The
2. symbol .. simply means "one directory up."
3. To change to a subdirectory of the current directory, use a relative reference. If your working directory is /ietf, you can change to /ietf/95jul simply by typing cd 95 jul. A directory specification that does not begin with a / is a relative reference that begins with the current directory.
4. You can change to any directory on the system by using an absolute reference that begins with the root directory. The command cd /ietf/95apr changes you to the 95apr directory, no matter what your current directory is.

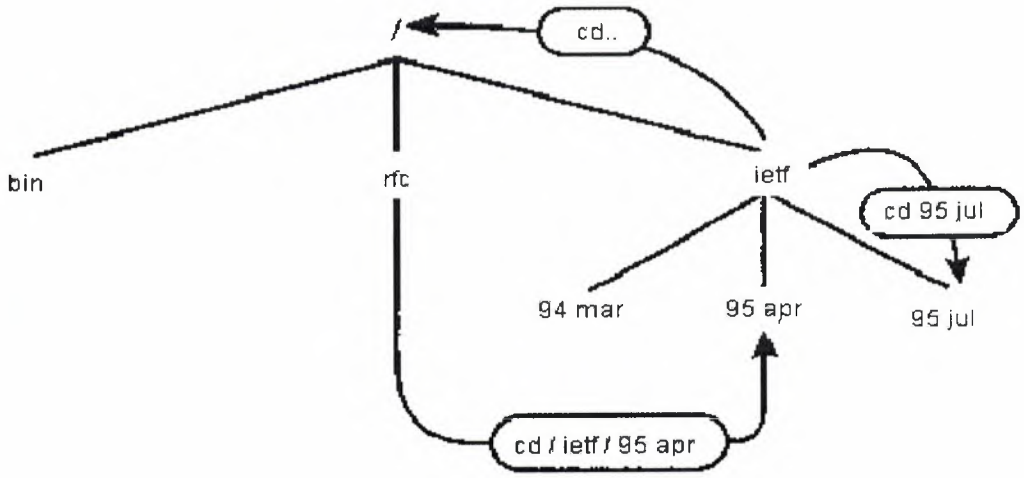


Figure 4.2 Navigating a UNIX directory tree.

4.1.3.6 Listing Directory Contents

This directory contains nearly 2,000 files. You could see them all by typing the is command, but you probably want to use wild cards to delimit the file list.

Directory contents may be shown as file names alone or as detailed file lists. The is (list) command shows only file and directory names. You can use the familiar * and ? wild cards with is. The following example was chosen to highlight several important documents that you should retrieve during your first session with InterNIC.

```
ftp> ls rfc. *  .txt
200 PORT command successful
150 Opening ASCII mode data connection for file list.
rfc – index.txt
rfc – instructions.txt
rfc – retrieval.txt
226  Transfer complete.
56   bytes  received in 0.01 seconds (5.60 kbtes / sec )
ftp >
```

Processing wild card characters is called globing. When globing is turned on, ? and * are treated as wild cards. If required, you can turn globing off by issuing the glob command. Enter the command again to turn globing on.

Even though some of the commands resemble UNIX commands, you cannot freely use UNIX commands while in an FTP session. The commands are processed by ftp, not by a UNIX shell.

When you know you are working on a UNIX host, however, some command options are supported. The is command accepts the following parameters (note the use of uppercase in some options):

-a List all files, including hidden ones
-C Produces a column listing (similar to the DOS/W switch)
-d Lists directory the name only
-F Displays the file type (directory or executable)
-l Produces a long listing
-R Specifies a recursive (continuous) listing

To display more detail, use the dir command. The details you see depend on the OS running on the FTP server. This list originates on a UNIX computer and includes UNIX security information (called permissions) in the leftmost column. Because this server permits unsecured anonymous logins, you are unlikely to see any files that are not secured with read-only (r) permissions. In the following example, a ? wild card was used as a single-character placeholder to display all files starting with the characters "rfcl70".

ftp> dir rfc170?.txt

200 Port command successful.

150 Opening ACII mode data connection for / bin /ls.

- r - - r - - r - - 1 welcome ftpguest 458860 oct 1994 rfc1700.txt

-r--r--r-- 1 welcome ftpguest 15460oct20 1994 rfc1701.txt

-r--r--r-- 1 welcome ftpguest7288oct201994 rfc1702.txt

-r--r--r-- 1 welcome ftpguest17985oct211994 rfc1703.txt

-r--r--r-- 1 welcome ftpguest42335nov 11994 rfc1704.txt

-r--r--r-- 1 welcome ftpguest65222oct 251994 rfc1705.txt

-r--r--r-- 1 welcome ftpguest19721oct211994 rfc1706.txt

-r--r--r-- 1 welcome ftpguest37568oct281994 rfc1707.txt

-r--r--r-- 1 welcome ftpguest26523oct251994 rfc1708.txt

-r--r--r-- 1 welcome ftpguest66659nov221994 rfc1709.txt

226 Transfer complete.

670 bytes received in 0.54 seconds (1.24 Kbytes/sec)

ftp>

If the directory contains subdirectories, you see an entry similar to the following. The d at the beginning of the permissions indicates that the entry is a directory.

drwr – xr – x 3 welcome 1 25600 sep 20 04:07 rfc

4.1.3.7 Retrieving Files

To retrieve an RFC file, use the get command. Keep in mind as you do so that file names in UNIX are case sensitive. The following example gets the file rfcl700.txt from the working directory on the FTP host and retrieves it to the working directory on the local computer:

ftp> get rfcl700.txt

200 PORT command successful.
150 Opening ASCII mode data connection for rfc1700.txt (458860 bytes).
226 Transfer complete.
471743 bytes received in 92.04 seconds (5.13 Kbytes/sec)
ftp>

You do not have permissions to copy files to DS.INTERNIC.NET. For hosts on which you have file write permissions, the put command works very similarly to get.

As disclosed in the ftp dialog, the preceding example was of an ASCII transfer, which generally is preferable for text documents. UNIX and DOS/Windows have different conventions for indicating the ends of lines in text files. UNIX uses the linefeed character alone, while DOS/Windows uses a carriage return/linefeed combination. When ftp transfers in ASCII mode between different host environments, the proper character translations take place.

To turn off end-of-line character translation, enter the cr command. Enter cr again to restore translation. You use the cr command if the file's ultimate destination is a DOS or Windows computer.

If you transfer program or data files, you do not want any character translation or format changes to take place. Before getting the file, enter the command binary to instruct ftp not to convert file contents.
Enter the command ASCII to restore ASCII-mode file transfers.
ftp supports a variety of commands that determine how files transfer and display the information. If you want to know what settings currently are in effect, enter the status command, as in this example:

ftp> status
Connected to ds.internic.net.
Type: ascii; Verbose:On; Bell:Off; Prompting: On
Globbing:On; Debugging:Off; Hash mark printing : Off
ftp>

4.1.3.8 Ending an FTP Session

After you finish your operations on the FTP server, you need to close your connection by entering the close command:

```
ftp>    close
221     Goodbye
ftp>
```

You can then exit ftp by using the command bye. Typing bye while in session with an

FTP host both closes the connection and exits ftp.

```
ftp>bye
C:\
```

4.1.4 FTP Command Reference

You probably will use ftp to obtain files to set up your Internet host or server, and you probably will need to become familiar with command-mode ftp. You might find the command-mode approach preferable to a graphical one. A graphical FTP front-end must transfer entire directory contents to make the file names available for mouse actions. This is unnecessary in command mode, and you can connect to a familiar FTP server and retrieve files with a very few commands while a graphical front-end would still be transferring directory lists.

Table 4.1 includes an alphabetical listing of ftp commands supported on Windows NT with brief explanations of their functions. Many commands are toggles. Each time these commands are executed they reverse the status of the current function (on-to-off or off-to-on).

Table 4.1 Ftp Commands

Name	Function
append local remote	Appends the contents of the local file local to the end of the remote file remote
ASCII	Specifies that file transfers should be performed in 7-bit ASCII text mode.End-of-line characters will be translated as appropriate for the destination host.
binary	Specifies that no translations should be performed when files are transferred between hosts.This mode should be used for data, program, and 8-bit ASCII text files.
bye	Stops ftp.If a host connection is active,the conection is closed first.
case	Toggles conversion of upper-to lowercase characters when using the mget command.
cd path	Changes the working directory on the FTP server to the directory specified in path.
cdup	Changes the working directory on the FTP server to the parent of the current working directory.
close	Closes the host connection but leaves

	FTP running.
cr	Toggles the translation of carriage return characters when transferring text files.
delete file	Deletes the file named file from the FTP server.
dir	Obtains a detailed directory of files in the working directory on the FTP server.
dir path	Obtains a detailed directory of files in the working directory specified by path.
get remote local	Retrieves the file remote from the FTP server and stores it with the name local on the host. Omit local to retain the original file nam.
glob	Toggles use of file name expansion.
help ?	Lists the names of available commands.
help commands	Displays a brief message about command.
lcd path	Changes the working directory on the local computer.
ls	Lists files in the working directory on the FTP server.On a UNIX ls options.
mdelete name	Deletes all files on the FTP server that match name,which may include ? and * wild cards.

mget name	Retrieves files from the FTP server that match name,which may include ? and * wild cards.
mkdir directory	Creates a subdirectory named directory on the FTP server.
mput name	Copies to the ftp server all local files that match name,which may include ? and * wild cards.
open hostname	Opens a connection to the specified hostname, which must be running FTP server software to accept the connection.
prompt	Switches prompting on or off during operations on multiple files(wild cards are used).You may find prompting irritating for routine operations such as mput and mget, but many prefer to have prompting active for deleting files with mdelete.
put local remote	Copies the local file local to the FTP host with the name remote.

put local remote	Copies the local file local to the FTP host with the name remote.
pwd	Prints the working(current) directory on the FTP server.
rename old new	Renames a file on the ftp host from the name old to the name new.

rmdir directory	Removes a directory named directory from the FTP srever.
runique	Toggles uniques file mode when receiving files.To ensure that all received files have unique names,an extansion from.1 through.99 will be appended to each successive file name.One use is to ensure unique file names when copying many files with long names to a system that supports short file names such as the DOS 8.3 format.
status	Shows current session settings
sunique	Toggles unique file mode when sending files. To ensure that all received files have unique names,an extansion from1. through .99 will be appended to each successive file name.
user username	Initiates a login to the FTP host with the username specified ftp prompts for a password.
verbose	Switches verbose mode on or off.Verbose mode provides information you might not care about ,such as file transfer statistics.

4.2 Trivial File Transfer Protocol

FTP was designed to provide robust and secure file operations over unreliable networks and uses TCP as a transport protocol to provide reliable delivery. To operate over TCP virtual circuits, FTP requires hosts to establish a connection before file operations can commence. Part of FTP connection establishment incorporates logins for security.

When the network is reliable, as on a local area network, the overhead of FTP might not be desirable. Consequently, a simpler protocol was developed, Trivial File Transfer Protocol (TFTP; RFC 1350). TFTP uses the unreliable UDP protocol as a transport and does not require establishing a connection or logging on before file transfer requests are possible.

TFTP's lack of security makes offering TFTP services a bit risky for a computer on a public network, because it provides an entree to the computer that could allow an outsider to gain unsecured access. Therefore, system administrators quite commonly disable TFTP on publicly available computers. File access available to TFTP must be carefully secured.

TFTP is a small and efficient protocol easily embedded in a computer's boot ROM. Sun UNIX workstations, for example, use TFTP to download a central operating system image when booting the system on a network.

4.3.1 How Telnet Works

Figure 4.2 illustrates the architecture of Telnet, which like FTP, is based on client and server processes. A Telnet server process on the remote host maintains a virtual terminal, an image running entirely in software of a terminal that can interact with the remote host. A user initiates a Telnet session by running Telnet client software and logging on to the Telnet server. The Telnet server receives keystrokes from the client and applies them to the virtual terminal, which in turn interacts with other processes on the host. The Telnet server also receives screen display data directed to

the virtual terminal and communicates the data to the Telnet client. To the user, it appears that the terminal session is taking place on the local computer, whereas the remote host has the viewpoint that it is interacting with a local terminal.

There is nothing terribly fancy about Telnet, which is based on emulation of one of several models of text-mode terminals, most commonly the Digital VT220, VT100/ VT102, or the VT52. These terminals can perform sophisticated text-based operations such as displaying menus that permit option selection using arrow keys. But they remain text-based.

Another limitation of Telnet is that it does not enable the local computer to participate in processing. All processing takes place on the remote Telnet server, and the local host functions as a "dumb" terminal. Therefore, Telnet cannot be used as a foundation for sophisticated network operations such as file sharing. In fact, Telnet offers no mechanism for file transfer, which must take place under a separate FTP session.

Nevertheless, there are many good reasons to become familiar with Telnet. A great many services on the Internet still are available only through Telnet operation. Even though alternatives exist for other services, Telnet remains a functional way to access many things on the Internet, including:

1. **Archie.** Archie is a database service that catalogs tens of thousands of files on a thousand or so anonymous FTP servers. Archie was developed as a project by McGill University School of Computer Science in Montreal, Canada, and now is widely available as a resource. Several useful documents can be FTP'd from archie.ans.net. Look in the directory pub/archie/doc, and start by obtaining the file whatis.archie.

2. **Gopher.** Originally created to provide a database for documents in a distributed computing environment, Gopher (developed at the University of Minnesota, home of the Golden Gophers) now provides access to a wide variety of files and network services. FTP a guide to Gopher from boombox.micro.umn.edu and retrieve the file pub/gopher/00README.

3. **Veronica.** Gopher provides a database of files on a local computer. Veronica is to Gopher what Archie is to FTP, providing an index to the contents of the majority of Gopher sites. Veronica, incidentally, stands for "very easy rodent-oriented net-wide index to computerized archives." You can FTP a good introduction to Gopher and Veronica from ftp.cso.uiuc.edu. Retrieve the file doc/net/uiucnet/vol6no1.txt.

4. **WAIS.** An Internet tool that enables users to search indexes of cataloged resources. An InterNIC WAIS server is available as a convenient way to find Internet-related documents. To access a WAIS client on the InterNIC server, Telnet to DS.INTERNIC.NET, and log in with the username wais. An online tutorial is available.

You can access Archie, Gopher, Veronica, and WAIS by running local client software. If you do not have the appropriate client installed, however, you can access remote clients using Telnet. The next section provides an example of using Telnet to access a remote Archie client.

4.3.2 Example of a Telnet Session

If you need a file but don't know which FTP servers provide it, or even precisely what the file name is, you need Archie. In this example, Telnet is used to locate sources for Netscape, the popular browser for the World W[r]ide Web. Obtain this file as one step in getting Windows NT well-hooked-into the Internet.
The dialog begins by using Telnet to access an Archie server. Table 4.2 lists a wide variety of Archie servers. The one used in this example is archie.rutgers.edu.

The Telnet that ships with Windows NT, unlike FTP, is Windows-based. You can execute the command telnet from a command prompt, or you can create an icon with the path \winnt\system32\telnet.exe. After starting Windows NT Telnet, select the **Remote System** option in the **Connect** menu to enter a host name and initiate a Telnet connection. This box permits you to specify a terminal type (VT100 or ANSI). It also can be used to access different ports, but the **Port** setting should be left as **telnet.** Choose **Connect** to reach out and touch archie.rutgers.edu.

After establishing a connection, a login dialog appears. To access an Archie client, log in using the username archie. Here is the login sequence:

```
login: archie

Solaris 2 (dogbert.rutgers.edu)  (pts/6)

login: archie
# Bunyip Information Systems, Inc., 1993, 1994, 1995
#Terminal type set to 'vt100 24 80'.
# 'erase ' character is '^?'.
# 'search ' ( type string ) has the value ' sub '.
archie>
```

This listing includes only a representative sample of the locations reported. The majority of entries are directories because the Netscape product distribution file does not contain the text "netscape." These results, however, are sufficient to give you some likely **ftp** servers that should have the file you want.

You probably want a record of the search results. You could use **telnet** logging to capture a record of your dialog with **archie,** but a better way is to e-mail yourself the results of the latest search. As in the preceding example, simply enter the command mail followed by your Internet mail ID.

If you know that the file for the 32-bit version of Netscape, the version most appropriate for Windows NT and Windows 95, begins with the characters "n32", you can conduct a different search:

```
archie> prog n32
#search type: sub.
#your queue position:1
#Estimated time for completion:5 seconds
working...=

Host alfred.ccs.carleton.ca      (134.117.1.1)
```

Last updated 14:56 19 Sep 1995

Location: / pub / civeng / viewers

File -r – r – r - 1269110 bytes 15:00 16 Jun 1994 win32s.zip

Host ftp.bt.net (194.72.6.51)

Last updated 14:55 9 Sep 1995

Location: / pub / computing / information-systems/www/Netscapes/netscape1.2b3

File -r – r – r - 1339223 bytes 15:30 18 Jun 1995 n3212b3.exe

Host venera.isi.edu (128.9.0.32)

Last updated 02:51 19 Sep 1995

Location : / pub / httpd /htdocs/sims/sheila

File -r – r – r - 1198241 bytes 21:49 22 Jun 1995 n32e11n.exe

Again, this listing includes only a representative sample of the files reported. Notice that the files win32s.zip and n3212b3.exe both meet the substring search criteria. After completing each search, you can e-mail the results.

After you finish using **archie,** enter the command bye to end the session:

archie> bye

bye.

You can learn more about Archie from the documents on archie.ans.net. Table 4.2 lists some more Archie servers.

Host Name	Location
Archie.au	Australia
Archi3e.univie.ac.at	Austria
archie. mcgill.ca	Canada
Archie.funet.fi	Finland
Archie.univ-rennesl.fr	France
Archie.th-darmstadt.de	Germany
Archie.ac.il	Israel
Archie.unipi.it	Italy
Archie33.wide.ad.jp	Japan
archie, hama.nm.kr	Korea
Archie.uninett.no	Norway
archie.rediris.es	Spain
archie.luth.se	Sweden
archie.sw3itch.ch	Switzerland
archie. ncu.edu. tw	Taiwan
archie.doc.ic.ac.uk	United Kingdom
archie.unl.edu	USA (Nevada)
archie.internic.net	USA (New Jersey)
archie.rutgers.edu	USA (New Jersey)
archie.ans.net	USA (New York)
archie.sur3a.net	USA (Maryland)

4.4 Simple Mail Transfer Protocol

Electronic mail probably is the single most important application on the Internet. The protocol that supports Internet e-mail is SMTP (RFC 821; RFC 822 defines the message format), which is the protocol used to transfer e-mail messages between different TCP/IP hosts. Although a knowledgeable user quite possibly can communicate directly using the SMTP protocol, doing so is not the standard way of working. Usually, several communication layers are involved.

4.4.1 Architecture of SMTP Mail

Figure 4.3 illustrates the architecture of an SMTP-based mail system. Hosts that support e-mail use a mail transfer agent (MTA) to manage the process. The most *popular MTA in the UNIX community is send mail. Broadly speaking, the MTA has* two responsibilities:

1. Sending messages to and receiving messages from other mail servers
2. Providing an interface that enables user applications to access the mail system

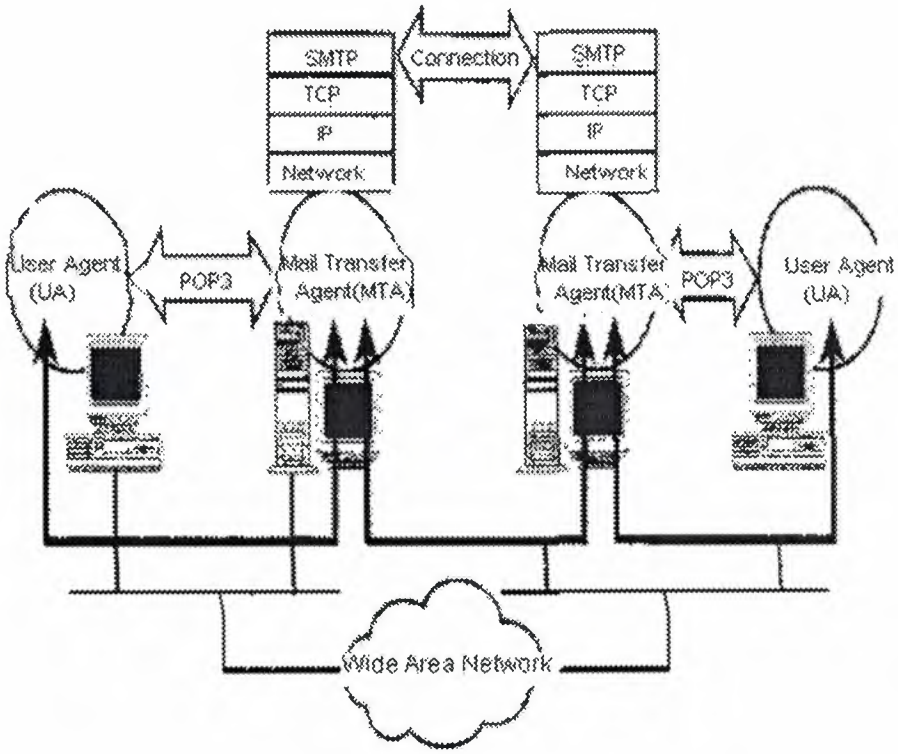


Figure 4.3 Architecture of SMTP-based messaging

The MTA is responsible for providing users with addressable mailboxes. When e-mail is addressed to frodo@bagend.org, bagend.org is the domain name that identifies the host running the MTA. The MTA is responsible for ensuring that messages to frodo arrive in the correct mailbox.

End users interface with the MTA, using one of the many available user agents (UAs). The UA puts a friendly face on the network e-mail, shielding the user from a fairly

complicated process. UAs use a mail protocol to communicate with the MTA, such as Post Office Protocol - Version 3 (POPS; RFC 1460).

A common UA in the UNIX world is the text-based program named mail. In the Windows environment, a popular program is Eudora, available in both shareware and commercial versions for a variety of platforms. But e-mail is so ubiquitous on the Internet that large numbers of Internet applications, such as Web browsers and newsgroup viewers, now provide at least a rudimentary e-mail front end.

4.4.2 Delivering Electronic Mail

Electronic mail systems are not designed to provide real-time message interchange. For that, users can employ tools such as chat programs. Instead, e-mail systems are designed to route large volumes of messages in a reasonable (but not necessarily short) amount of time with as little network overhead as possible. If every mail message were rushed through the Internet via packet switching, the Internet would quickly be bogged down. So e-mail is implemented with the understanding that limited message transit time is permissible if it reduces network loading.

E-mail servers use a store-and-forward method for delivering messages. Figure 6,15 illustrates an example of an e-mail system in which B must forward messages between A and C, B would need to work very hard if it had to route IP datagrams in real time. However, B takes a more relaxed approach. When B receives a message from A. B receives the entire message, storing it on a local hard drive. If B has other priorities, such as receiving other incoming messages, B may wait until things quiet down before forwarding the message to C. B may even be configured to send to C only when several messages are queued or after a time interval has expired. It is more efficient for B to transfer several messages with one connection than to open a connection for every message. These techniques can slow transit time for a message to minutes or hours, but greatly increase overall efficiency.

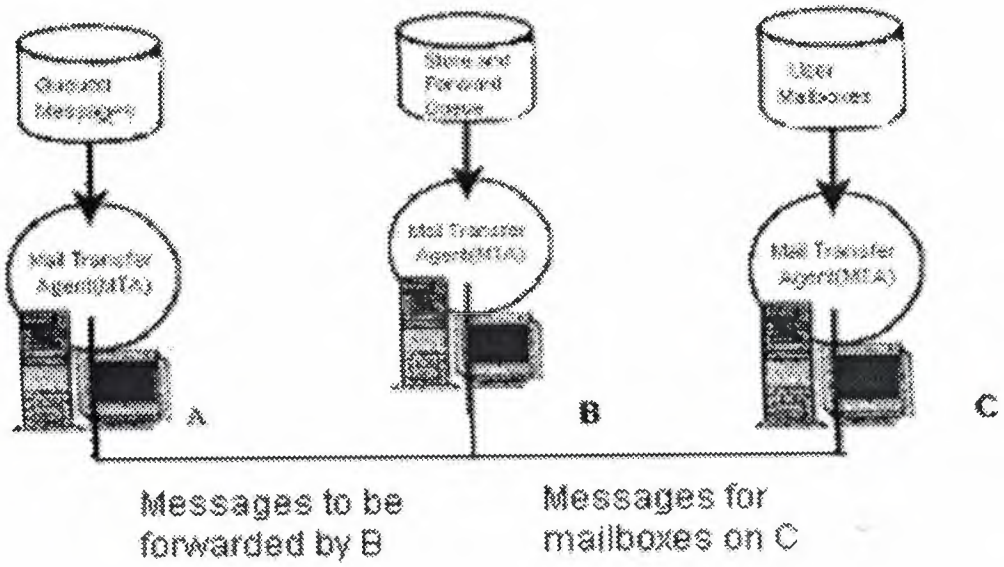


Figure 4.4 Forwarding electronic mail messages.

4.4.3 Characteristics of SMTP

SMTP is a fairly old protocol that predates widespread use of graphic terminals. Many of the data types being transported on modern messaging systems were undreamed of in 1982 when RFC 821 was published. Who could have imagined that we would be e-mailing pictures, voice messages, even video clips? In keeping with then-current technology, SMTP was designed to transfer messages consisting of 7-bit ASCII text.

When binary data must be sent through SMTP mail systems, the data must be encoded to a format compatible with 7-bit character transmission. The receiving site then decodes the message to retrieve the binary data. Many user agents now perform these translations automatically for attached binary files.

The most common encoding scheme in the UNIX community is implemented in the program **uuencode** (UNIX-UNIX encodes), which has a companion program named **uudecode.** Programs are available to perform uuencodes and uudecodes on DOS, Windows, and Macintosh computers.
The Multipurpose Internet Mail Extensions (MIME; RFC 1521) is an Internet elective protocol that supports transfer of non text messages via SMTP.

SMTP was developed to handle non sensitive messages in a fairly close-knit Internet community that operated with a high level of trust. Consequently, SMTP has weak security and does not encrypt messages. A knowledgeable person can readily extract messages from the raw protocol data that travels through the public network. Sensitive data, therefore, should be encrypted before transmission through SMTP mail.

4.5 Simple Network Management Protocol

A wide variety of activities can fairly be described as "network management." With regard to SNMP, network management means collecting, analyzing, and reporting data about the performance of network components. Data collected by SNMP include performance statistics and other routine reports, as well as alerts that report potential or existing network problems.

SNMP is one of a family of protocols that comprise the Internet network management
Strategy;

1. **SNMP.** The protocol that enables network management stations to communicate with managed devices.
2. MIB. The management information base is the database that stores system management information.

3. SMI. Structure and identification of management information describes how each object looks in the MIB.

Before looking at these protocols, you should understand how an SNMP-based network management system is organized.

4.5.1 Organization of SNMP Management

As figure 4.3 illustrates, SNMP is organized around two types of devices:

1. Network management stations serve as central repositories for the collection and analysis of network management data.

2. Managed devices run an SNMP agent, which is a background process that monitors the device and communicates with the network management station. Often, devices that cannot support an SNMP agent can be monitored by a proxy device that communicates with the network management station.

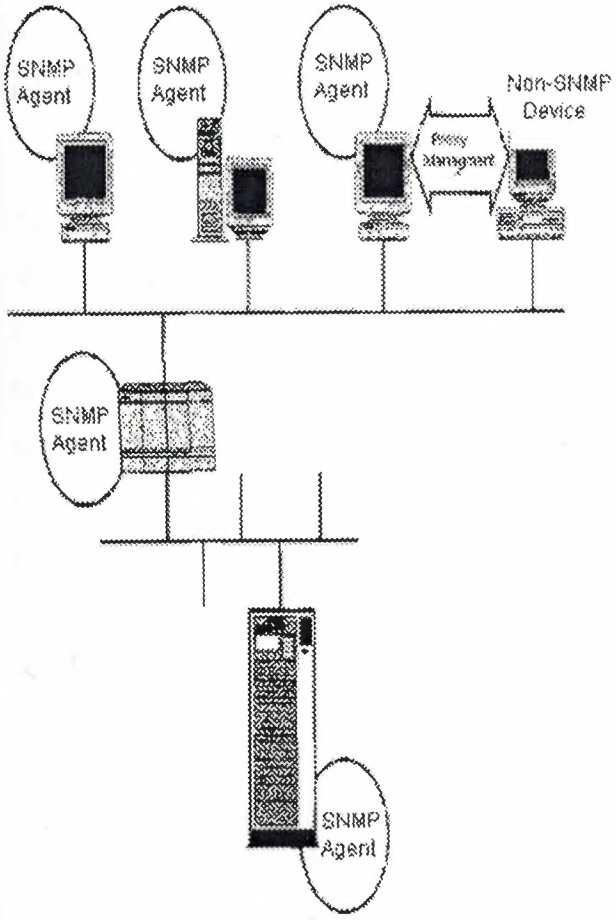


Figure 4.5 Organization of an SMTP- managed network.

Information is obtained from managed devices in two ways: polling and interrupts. These methods support different network management goals.

One goal of network management is to maintain a history of network performance that consists of "snapshots" of various characteristics of the network. These snapshots can be used to analyze trends, anticipate future demand, and isolate problems by comparing the current state of the network to prior conditions. The most efficient way to obtain snapshots is to have the management station poll managed devices, requesting that they provide the required information. Polling (see fig. 4.5) occurs only at defined intervals, rather than continually, to reduce network traffic. A snapshot of the performance characteristics of a healthy network is called a baseline.

Another goal is to quickly notify managers of sudden changes in the network. When operational conditions for a device alter abruptly, waiting for the next poll from the management station is not timely. Managed devices can be configured to send alerts (also called traps) under predefined conditions. Network managers define thresholds for specific network performance characteristics. When a threshold is exceeded, the agent on a managed device immediately sends a trap to the management station (see fig. 4.6).

Essentially, a network management station and an agent can perform the following tasks:

1. The network management station can poll agents for network information.
2. The network management station can update, add, or remove entries in an agent's database. A router's router tables, for example, might be updated from the management console.
3. The network management station can set thresholds for traps.
4. The agent on a managed device can send traps to the network management station.

4.5.2 The Management Information Base

A MIB is a set of objects (types of network entities) that are included in the network management database. The MIB specification states the nature of the objects, whereas the SMI describes the appearances of the objects.

Three Internet MIB standards have been associated with SNMP.

1. **MIB-I** (RFC 1156). This was the first MIB specification, initially published as RFC 1066 in 1988. MIB-I defined eight object groups.
2. **MIB-II (RFC 1213).** This is the current recommended standard, defining 10 object groups and 171 objects.
3. **RMON-MIB (RFC 1513/1217).** The Remote Monitoring MIB is oriented
4. Around monitoring network media rather than network devices.

Additionally, a number of experimental and private (or enterprises) MIBs are available. Some enterprises MIBS are provided by vendors to support unique features of their products.
Experimental MIBs are undergoing trial as they are considered as possible extensions to the MIB standards. MIBs that are proven of value during experimentation may be considered for inclusion in the standard MIB-space.

4.5.3 SNMP

Although SNMP (RFC 1157) was designed for the Internet, SNMP is not dependent on the TCP/IP protocols and can operate above a variety of lower-level protocols such as Novell's IPX/SPX. With TCP/IP, SNMP runs over the UDP datagram protocol to reduce network overhead.

SNMP requests are identified by a community name, a 32-character, case-sensitive identification that serves somewhat as a password. SNMP implementations may impose

limitations on the characters that can appear in community names. Community names serve to identify SNMP messages but are transmitted as open text and, therefore, are not secure. The three types of community names are as follows:

1. **Monitor Community.** This community name grants read access to MIBs and must be included with each MIB query. The default monitor community name is "public."
2. **Control Community.** This community name grants read and write access to MIBs.
3. **Trap Community.** This community name must accompany trap messages. An SNMP console rejects trap messages that do not match its configured trap community name. The default trap community name is "public."

SNMP entities operate asynchronously. A device need not wait for responses before it can send another message. A response is generated for any message except a trap, but network management stations and managed devices communicate fairly informally.

The current SNMP standard, version 1, can perform four operations:

1. **get**. Retrieves a single object from the MIB.
2. **get-next.** Traverses tables in the MIB.
3. **set.** Manipulates MIB objects.
4. **trap.** Reports an alarm.

get, get-next, and set commands originate from the network management station. If a managed device receives the command, a response is generated that is essentially the command message with filled-in blanks.

SNMP vl has several shortcomings, most particularly in the area of security. SNMP names are sent as clear text, which can be observed by anyone who can observe raw network traffic. SNMP v2 (RFC 1441-1452) is a proposed standard that improves security by encrypting messages. SNMP v2 also improves efficiency, for example, by

enabling a management console to retrieve an entire table in one request, rather than record-by-record with get-next.

4.5.4 Network Management Stations

Network management is of little value if the management data is not available for report, analysis, and action. Among the available network management console products are the following:

1. Hewlett-Packard OpenView (DOS, Windows, and UNIX)
2. Sun Microsystems SunNet Manager (UNIX)
3. Novell Network Management System (NMS) (Windows)
4. Synoptics Optivity (Windows)
5. Cabletron Spectrum (Windows, UNIX)

Functions of the network management station include receiving and responding to traps, maintaining a trap history database, and getting object data from managed devices. More sophisticated—and costly—management consoles provide a graphic interface to the network and can construct a logical picture of the network structure. High-end management consoles generally are costly and require fairly powerful hardware.

The record-keeping capability of a management console is critical. It enables you to record baseline measurements of the network when it operates normally. These baseline measurements can be used to set thresholds for traps, which alert you to changes in network operation. They also provide a point of comparison that you can use to identify causes of network performance problems.

4.6 Network File System

Examination of the limitations of FTP and Telnet shows that they leave much to be desired where network computing is concerned. FTP can be used to transfer files, and Telnet can be used to run terminal sessions on a remote computer. But what if you want to run an application on your computer when some or all of the application files are stored on another computer? What if you want, for example, to open a spreadsheet program on your computer and update a spreadsheet on another computer? With FTP, yon need to transfer the file to your computer for work and back to the original computer when changes were complete. To complicate the matter further, what if the spreadsheet program itself is located on the remote computer? Must, you FTP all the application files as well? FTP just isn't up to the job.

Network File System (NFS) provides the TCP/IP equivalent of Microsoft products' file sharing capability. NFS was developed by Sun Microsystems and is widely licensed to other vendors who have implemented NFS on most platforms.

An NFS server can export a portion of its directory tree for use by NFS clients. Clients can mount the exported directories as if they are part of the clients' native file systems. DOS users, for example, access the exported directory as if it is a drive letter in the local DOS file structure. Figure 6.19 furnishes an example oi exporting an NFS directory.

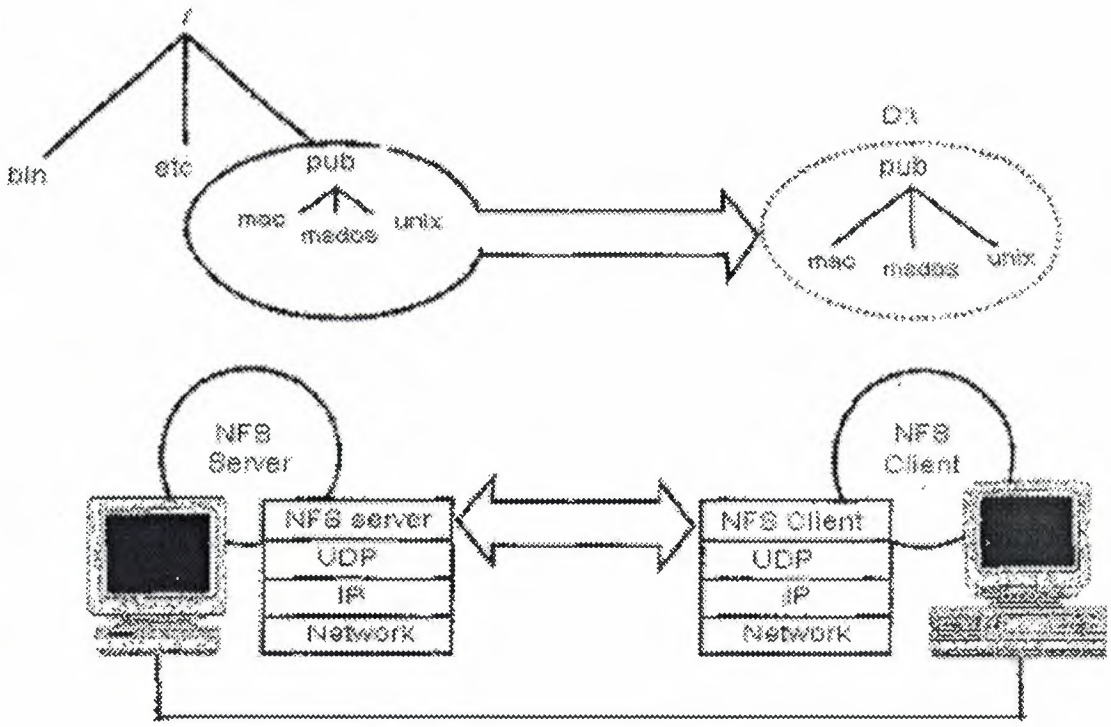


Figure 4.7 Exporting a directory with NFS

NFS is a sophisticated, reliable protocol that does not use TCP as a transport. For efficiency, NFS operates over UDP. NFS performs any required security, message fragmentation, and error recovery.

Microsoft does not offer an NFS product for Windows NT, but NFS is available from third-party vendors. Sources of NFS for Windows NT include the following:

1. Chameleon32NFS from Net Manage 408-973-7171
2. Disk Access for Windows NT from Intergraph Corp. 800-45-4856.
3. Connect NFS for NT from Beame & Whiteside 919-831-8989

computer at Berkeley make much sense? Why not communicate through the Internet? Telnet is a program that makes possible remote terminal access through a network.

4.3.1 How Telnet Works

Figure 4.2 illustrates the architecture of Telnet, which like FTP, is based on client and server processes. A Telnet server process on the remote host maintains a virtual terminal, an image running entirely in software of a terminal that can interact with the remote host. A user initiates a Telnet session by running Telnet client software and logging on to the Telnet server. The Telnet server receives keystrokes from the client and applies them to the virtual terminal, which in turn interacts with other processes on the host. The Telnet server also receives screen display data directed to the virtual terminal and communicates the data to the Telnet client. To the user, it appears that the terminal session is taking place on the local computer, whereas the remote host has the viewpoint that it is interacting with a local terminal.

There is nothing terribly fancy about Telnet, which is based on emulation of one of several models of text-mode terminals, most commonly the Digital VT220, VT100/ VT102, or the VT52. These terminals can perform sophisticated text-based operations such as displaying menus that permit option selection using arrow keys. But they remain text-based.

Another limitation of Telnet is that it does not enable the local computer to participate in processing. All processing takes place on the remote Telnet server, and the local host functions as a "dumb" terminal. Therefore, Telnet cannot be used as a foundation for sophisticated network operations such as file sharing. In fact, Telnet *offers no mechanism for file transfer, which must take place under a separate FTP* session.

Nevertheless, there are many good reasons to become familiar with Telnet. A great many services on the Internet still are available only through Telnet operation. Even though alternatives exist for other services, Telnet remains a functional way to access many things on the Internet, including:

1. **Archie.** Archie is a database service that catalogs tens of thousands of files on a thousand or so anonymous FTP servers. Archie was developed as a project by

McGill University School of Computer Science in Montreal, Canada, and now is widely available as a resource. Several useful documents can be FTP'd from archie.ans.net. Look in the directory pub/archie/doc, and start by obtaining the file what is.archie.

2. **Gopher.** Originally created to provide a database for documents in a distributed computing environment, Gopher (developed at the University of Minnesota, home of the Golden Gophers) now provides access to a wide variety of files and network services. FTP a guide to Gopher from boombox.micro.umn.edu and retrieve the file pub/gopher/00README.

3. **Veronica.** Gopher provides a database of files on a local computer. Veronica is to Gopher what Archie is to FTP, providing an index to the contents of the majority of Gopher sites. Veronica, incidentally, stands for "very easy rodent-oriented net-wide index to computerized archives." You can FTP a good introduction to Gopher and Veronica from ftp.cso.uiuc.edu. Retrieve the file doc/net/uiucnet/vol6no1.txt.

4. **WAIS.** An Internet tool that enables users to search indexes of cataloged resources. An Inter NIC WAIS server is available as a convenient way to find Internet-related documents. To access a WAIS client on the Inter NIC server, Telnet to DS.INTERNIC.NET, and log in with the username wais. An online tutorial is available.

You can access Archie, Gopher, Veronica, and WAIS by running local client software. If you do not have the appropriate client installed, however, you can access remote clients using Telnet. The next section provides an example of using Telnet to access a remote Archie client.

4.3.2 Example of a Telnet Session

If you need a file but don't know which FTP servers provide it, or even precisely what the file name is, you need Archie. In this example, Telnet is used to locate sources for Netscape, the popular browser for the World W[r]ide Web. Obtain this file as one step in getting Windows NT well-hooked-into the Internet.

The dialog begins by using Telnet to access an Archie server. Table 4.2 lists a wide variety of Archie servers. The one used in this example is archie.rutgers.edu. The Telnet that ships with Windows NT, unlike FTP, is Windows-based. You can execute the command telnet from a command prompt, or you can create an icon with the path \winnt\system32\telnet.exe. After starting Windows NT Telnet, select the **Remote System** option in the **Connect** menu to enter a host name and initiate a Telnet connection. This box permits you to specify a terminal type (VT100 or ANSI). It also can be used to access different ports, but the **Port** setting should be left as **telnet.** Choose **Connect** to reach out and touch archie.rutgers.edu.

After establishing a connection, a login dialog appears. To access an Archie client, log in using the username archie. Here is the login sequence:

```
login: archie

Solaris 2 (dogbert.rutgers.edu)  (pts/6)

login: archie
# Bunyip Information Systems, Inc., 1993, 1994, 1995
#Terminal type set to 'vt100 24 80'.
# 'erase ' character is '^?'.
# 'search ' ( type string ) has the value ' sub '.
 archie>
```

This listing includes only a representative sample of the locations reported. The majority of entries are directories because the Netscape product distribution file does not contain the text "Netscape." These results, however, are sufficient to give you some likely **ftp** servers that should have the file you want.

You probably want a record of the search results. You could use **telnet** logging to capture a record of your dialog with **archie,** but a better way is to e-mail yourself the results of the latest search. As in the preceding example, simply enter the command mail followed by your Internet mail ID.

If you know that the file for the 32-bit version of Netscape, the version most

appropriate for Windows NT and Windows 95, begins with the characters "n32", you can conduct a different search:

```
archie> prog n32
#search type: sub.
#your queue position:1
#Estimated time for completion:5 seconds
working...=

Host  alfred.ccs.carleton.ca      (134.117.1.1)
Last  updated  14:56  19  Sep  1995

        Location:    / pub / civeng / viewers
           File  -r – r – r -   1269110 bytes  15:00  16 Jun  1994 win32s.zip

Host  ftp.bt.net   (194.72.6.51)
Last updated  14:55  9 Sep  1995

        Location:              /    pub    /    computing    /    information-
systems/www/Netscapes/netscape1.2b3
        File  -r – r – r -   1339223  bytes  15:30 18 Jun 1995 n3212b3.exe

Host venera.isi.edu     ( 128.9.0.32 )
Last updated 02:51   19  Sep  1995

   Location :   / pub / httpd /htdocs/sims/sheila
           File   -r  –  r  –  r  -   1198241  bytes  21:49  22  Jun  1995  n32e11n.exe
```

Again, this listing includes only a representative sample of the files reported. Notice that the files win32s.zip and n3212b3.exe both meet the substring search criteria. After completing each search, you can e-mail the results.

After you finish using **archie,** enter the command bye to end the session:

archie> bye

bye.

You can learn more about Archie from the documents on archie.ans.net. Table 4.2 lists some more Archie servers.

Host Name	Location
Archie.au	Australia
Archi3e.univie.ac.at	Austria
archie. mcgill.ca	Canada
Archie.funet.fi	Finland
Archie.univ-rennesl.fr	France
Archie.th-darmstadt.de	Germany
Archie.ac.il	Israel
Archie.unipi.it	Italy
Archie33.wide.ad.jp	Japan
archie, hama.nm.kr	Korea
Archie.uninett.no	Norway
archie.rediris.es	Spain
archie.luth.se	Sweden
archie.sw3itch.ch	Switzerland
archie. ncu.edu. tw	Taiwan
archie.doc.ic.ac.uk	United Kingdom
archie.unl.edu	USA (Nevada)
archie.internic.net	USA (New Jersey)
archie.rutgers.edu	USA (New Jersey)
archie.ans.net	USA (New York)
archie.sur3a.net	USA (Maryland)

Table4.2 Archie Servers

4.4 Simple Mail Transfer Protocol

Electronic mail probably is the single most important application on the Internet. The protocol that supports Internet e-mail is SMTP (RFC 821; RFC 822 defines the message format), which is the protocol used to transfer e-mail messages between different TCP/IP hosts. Although a knowledgeable user quite possibly can communicate directly using the SMTP protocol, doing so is not the standard way of working. Usually, several communication layers are involved.

4.4.1 Architecture of SMTP Mail

Figure 4.3 illustrates the architecture of an SMTP-based mail system. Hosts that support e-mail use a *mail transfer agent (MTA)* to manage the process. The most popular MTA in the UNIX community is send mail. Broadly speaking, the MTA has two responsibilities:

1. Sending messages to and receiving messages from other mail servers
2. Providing an interface that enables user applications to access the mail system

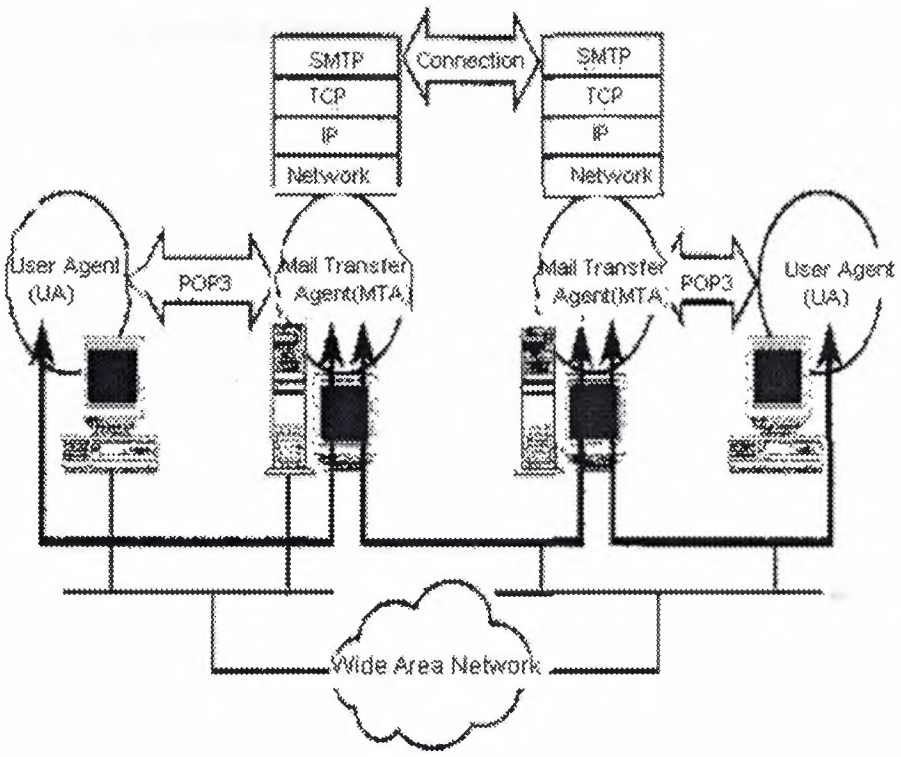


Figure 4.3 Architecture of SMTP-based messaging

The MTA is responsible for providing users with addressable mailboxes. When e-mail is addressed to frodo@bagend.org, bagend.org is the domain name that identifies the host running the MTA. The MTA is responsible for ensuring that messages to frodo arrive in the correct mailbox.

End users interface with the MTA, using one of the many available *user agents (UAs).* The UA puts a friendly face on the network e-mail, shielding the user from a fairly complicated process. UAs use a mail protocol to communicate with the MTA, such as Post Office Protocol - Version 3 (POPS; RFC 1460).

A common UA in the UNIX world is the text-based program named mail. In the Windows environment, a popular program is Eudora, available in both shareware and

commercial versions for a variety of platforms. But e-mail is so ubiquitous on the Internet that large numbers of Internet applications, such as Web browsers and newsgroup viewers, now provide at least a rudimentary e-mail front end.

4.4.2 Delivering Electronic Mail

Electronic mail systems are not designed to provide real-time message interchange. For that, users can employ tools such as chat programs. Instead, e-mail systems are designed to route large volumes of messages in a reasonable (but not necessarily short) amount of time with as little network overhead as possible. If every mail message were rushed through the Internet via packet switching, the Internet would quickly be bogged down. So e-mail is implemented with the understanding that limited message transit time is permissible if it reduces network loading.

E-mail servers use a store-and-forward method for delivering messages. Figure 6,15 illustrates an example of an e-mail system in which B must forward messages between A and C, B would need to work very hard if it had to route IP datagrams in real time. However, B takes a more relaxed approach. When B receives a message from A. B receives the entire message, storing it on a local hard drive. If B has other priorities, such as receiving other incoming messages, B may wait until things quiet down before forwarding the message to C. B may even be configured to send to C only when several messages are queued or after a time interval has expired. It is more efficient for B to transfer several messages with one connection than to open a connection for every message. These techniques can slow transit time for a message to minutes or hours, but greatly increase overall efficiency.

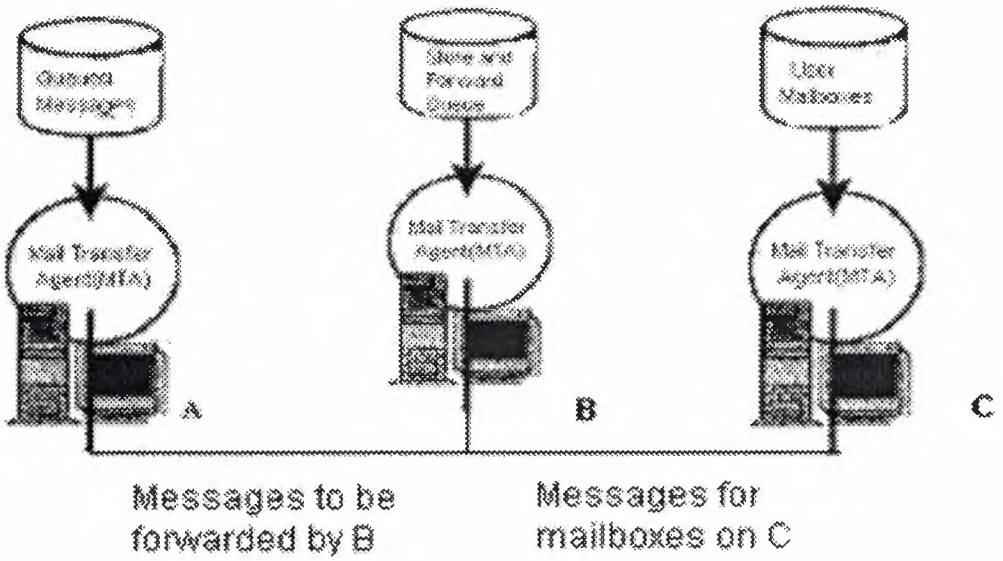


Figure 4.4 Forwarding electronic mail messages.

4.4.3 Characteristics of SMTP

SMTP is a fairly old protocol that predates widespread use of graphic terminals. Many of the data types being transported on modern messaging systems were undreamed of in 1982 when RFC 821 was published. Who could have imagined that we would be e-mailing pictures, voice messages, even video clips? In keeping with then-current technology, SMTP was designed to transfer messages consisting of 7-bit ASCII text.

When binary data must be sent through SMTP mail systems, the data must be encoded to a format compatible with 7-bit character transmission. The receiving site then decodes the message to retrieve the binary data. Many user agents now perform these translations automatically for attached binary files.

The most common encoding scheme in the UNIX community is implemented in the program **uuencode** (UNIX-UNIX encodes), which has a companion program

named **uudecode.** Programs are available to perform uuencodes and uudecodes on DOS, Windows, and Macintosh computers.
The Multipurpose Internet Mail Extensions (MIME; RFC 1521) is an Internet elective protocol that supports transfer of non text messages via SMTP.

SMTP was developed to handle non sensitive messages in a fairly close-knit Internet community that operated with a high level of trust. Consequently, SMTP has weak security and does not encrypt messages. A knowledgeable person can readily extract messages from the raw protocol data that travels through the public network. Sensitive data, therefore, should be encrypted before transmission through SMTP mail.

4.5 Simple Network Management Protocol

A wide variety of activities can fairly be described as "network management." With regard to SNMP, network management means collecting, analyzing, and reporting data about the performance of network components. Data collected by SNMP include performance statistics and other routine reports, as well as alerts that report potential or existing network problems.
SNMP is one of a family of protocols that comprise the Internet network management strategy:

1. **SNMP.** The protocol that enables network management stations to communicate with managed devices.
2. MIB. The management information base is the database that stores system management information.
3. SMI. Structure and identification of management information describes how each object looks in the MIB.

Before looking at these protocols, you should understand how an SNMP-based network management system is organized.

4.5.1 Organization of SNMP Management

As figure 4.3 illustrates, SNMP is organized around two types of devices:

1. Network management stations serve as central repositories for the collection and analysis of network management data.
2. Managed devices run an SNMP *agent,* which is a background process that monitors the device and communicates with the network management station. Often, devices that cannot support an SNMP agent can be monitored by a *proxy* device that communicates with the network management station.

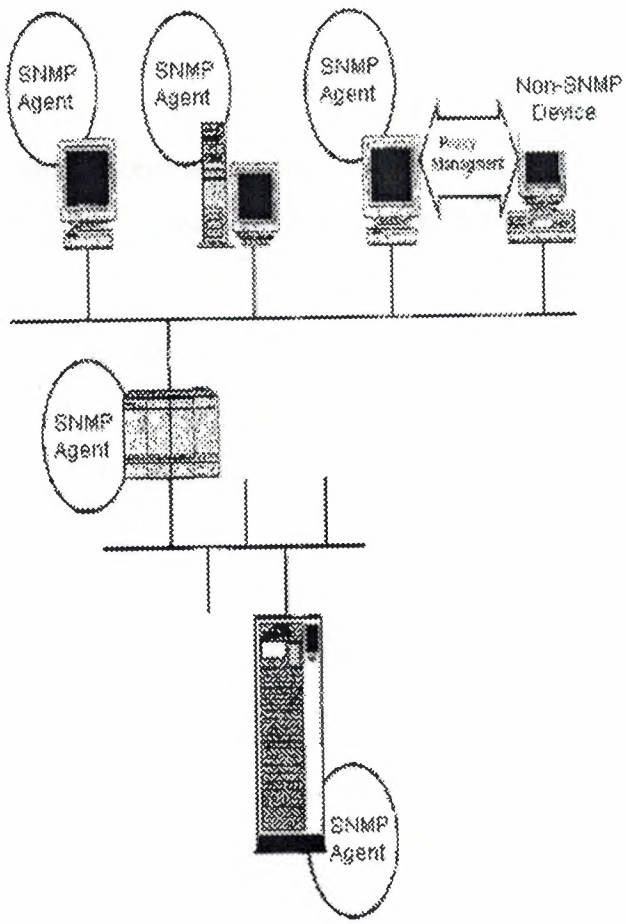


Figure 4.5 Organization of an SNMP- managed network.

Information is obtained from managed devices in two ways: polling and interrupts. These methods support different network management goals.

One goal of network management is to maintain a history of network performance that consists of "snapshots" of various characteristics of the network. These snapshots can be used to analyze trends, anticipate future demand, and isolate problems by comparing the current state of the network to prior conditions. The most efficient way to obtain snapshots is to have the management station poll managed devices, requesting that they provide the required information. Polling (see fig. 4.5) occurs only at defined intervals, rather than continually, to reduce network traffic. A snapshot of the performance characteristics of a healthy network is called a baseline.

Another goal is to quickly notify managers of sudden changes in the network. When operational conditions for a device alter abruptly, waiting for the next poll from the management station is not timely. Managed devices can be configured to send *alerts* (also called *traps)* under predefined conditions. Network managers define thresholds for specific network performance characteristics. When a threshold is exceeded, the agent on a managed device immediately sends a trap to the management station (see fig. 4.6).

Essentially, a network management station and an agent can perform the following tasks:

The network management station can poll agents for network information.

1. The network management station can update, add, or remove entries in an agent's database. A router's router tables, for example, might be updated from the management console.
2. The network management station can set thresholds for traps.
3. The agent on a managed device can send traps to the network management station.

4.5.2 The Management Information Base

A MIB is a set of objects (types of network entities) that are included in the network management database. The MIB specification states the nature of the objects, whereas the SMI describes the appearances of the objects.

Three Internet MIB standards have been associated with SNMP.

1. **MIB-I** (RFC *1*156). This was the first MIB specification, initially published as RFC 1066 in 1988. MIB-I defined eight object groups.
2. **MIB-II (RFC 1213).** This is the current recommended standard, defining 10 object groups and 171 objects.
3. **RMON-MIB (RFC 1513/1217).** The Remote Monitoring MIB is oriented around monitoring network media rather than network devices.

Additionally, a number of experimental and private (or *enterprises)* MIBs are available. Some enterprises MIBS are provided by vendors to support unique features of their products.

Experimental MIBs are undergoing trial as they are considered as possible extensions to the MIB standards. MIBs that are proven of value during experimentation may be considered for inclusion in the standard MIB-space.

4.5.3 SNMP

Although SNMP (RFC 1157) was designed for the Internet, SNMP is not dependent on the TCP/IP protocols and can operate above a variety of lower-level protocols such as Novell's IPX/SPX. With TCP/IP, SNMP runs over the UDP datagram protocol to reduce network overhead.

SNMP requests are identified by a community name, a 32-character, case-sensitive identification that serves somewhat as a password. SNMP implementations may impose limitations on the characters that can appear in community names. Community names

serve to identify SNMP messages but are transmitted as open text and, therefore, are not secure. The three types of community names are as follows:

1. **Monitor Community.** This community name grants read access to MIBs and must be included with each MIB query. The default monitor community name is "public."
2. **Control Community.** This community name grants read and write access to MIBs.
3. **Trap Community.** This community name must accompany trap messages. An SNMP console rejects trap messages that do not match its configured trap community name. The default trap community name is "public."

SNMP entities operate asynchronously. A device need not wait for responses before it can send another message. A response is generated for any message except a trap, but network management stations and managed devices communicate fairly informally. The current SNMP standard, version 1, can perform four operations:

1. **get**. Retrieves a single object from the MIB.
2. **get-next.** Traverses tables in the MIB.
3. **set.** Manipulates MIB objects.
4. **trap.** Reports an alarm.

get, get-next, and set commands originate from the network management station. If a managed device receives the command, a response is generated that is essentially the command message with filled-in blanks.

SNMP vl has several shortcomings, most particularly in the area of security. SNMP names are sent as clear text, which can be observed by anyone who can observe raw network traffic. SNMP v2 (RFC 1441-1452) is a proposed standard that improves security by encrypting messages. SNMP v2 also improves efficiency, for example, by

enabling a management console to retrieve an entire table in one request, rather than record-by-record with get-next.

4.5.4 Network Management Stations

Network management is of little value if the management data is not available for report, analysis, and action. Among the available network management console products are the following:

1. Hewlett-Packard OpenView (DOS, Windows, and UNIX)
2. Sun Microsystems SunNet Manager (UNIX)
3. Novell Network Management System (NMS) (Windows)
4. Synoptics Optivity (Windows)
5. Cabletron Spectrum (Windows, UNIX)

Functions of the network management station include receiving and responding to traps, maintaining a trap history database, and getting object data from managed devices. More sophisticated—and costly—management consoles provide a graphic interface to the network and can construct a logical picture of the network structure. High-end management consoles generally are costly and require fairly powerful hardware.

The record-keeping capability of a management console is critical. It enables you to record baseline measurements of the network when it operates normally. These baseline measurements can be used to set thresholds for traps, which alert you to changes in network operation. They also provide a point of comparison that you can use to identify causes of network performance problems.

4.6 Network File System

Examination of the limitations of FTP and Telnet shows that they leave much to be desired where network computing is concerned. FTP can be used to transfer files, and Telnet can be used to run terminal sessions on a remote computer. But what if you want to run an application on your computer when some or all of the application files are stored on another computer? What if you want, for example, to open a spreadsheet program on your computer and update a spreadsheet on another computer? With FTP, yon need to transfer the file to your computer for work and back to the original computer when changes were complete. To complicate the matter further, what if the spreadsheet program itself is located on the remote computer? Must, you FTP all the application files as well? FTP just isn't up to the job.

Network File System (NFS) provides the TCP/IP equivalent of Microsoft products' file sharing capability. NFS was developed by Sun Microsystems and is widely licensed to other vendors who have implemented NFS on most platforms.

An NFS server can export a portion of its directory tree for use by NFS clients. Clients can mount the exported directories as if they are part of the clients' native file systems. DOS users, for example, access the exported directory as if it is a drive letter in the local DOS file structure. Figure 6.19 furnishes an example oi exporting an NFS directory.

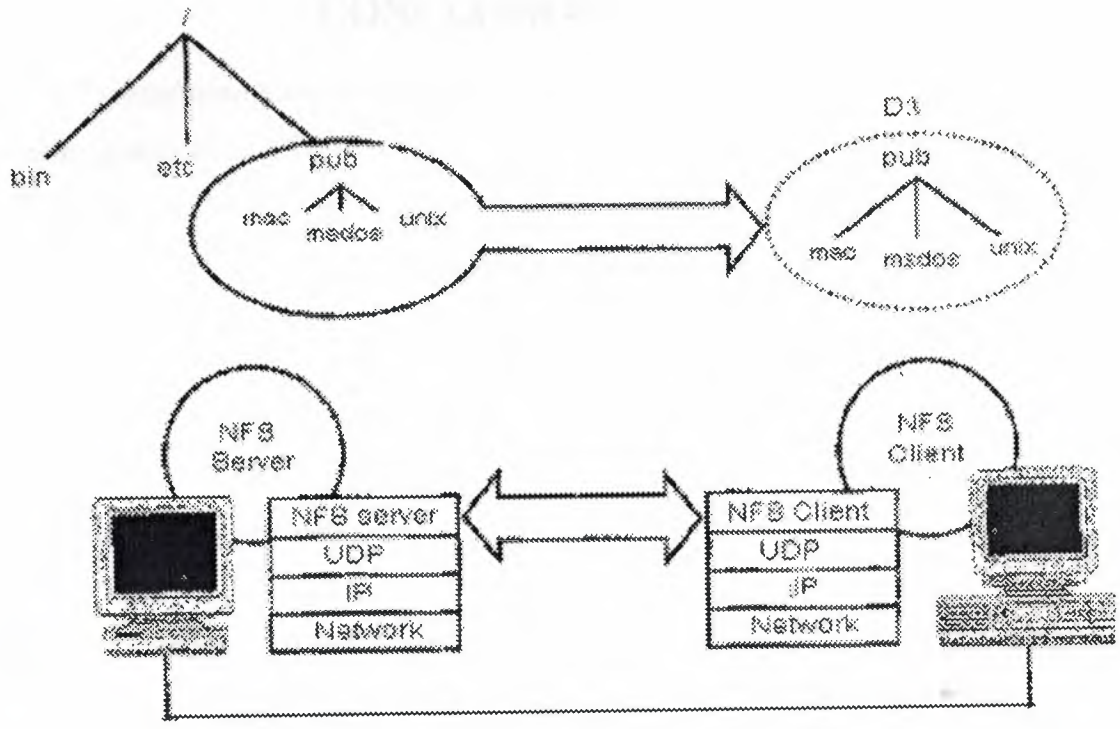


Figure 4.7 Exporting a directory with NFS.

NFS is a sophisticated, reliable protocol that does not use TCP as a transport. For efficiency, NFS operates over UDP. NFS performs any required security, message fragmentation, and error recovery.

Microsoft does not offer an NFS product for Windows NT, but NFS is available from third-party vendors. Sources of NFS for Windows NT include the following:

1. Chameleon32NFS from Net Manage 408-973-7171
2. Disk Access for Windows NT from Intergraph Corp. 800-45-4856.
3. Connect NFS for NT from Beame & Whiteside 919-831-8989

CONCLUSION

Abbreviation of Transmission Control Protocol, and pronounced as separate letters. TCP is one of the main protocols in TCP/IP networks. Whereas the IP protocol deals only with packets, TCP enables two hosts to establish a connection and exchange streams of data. TCP guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent.

TCP/IP (The Transmission Control Protocol/Internet Protocol) is the protocol suite that drives the Internet. Specifically, TCP/IP handles network communications between network nodes (computers, or nodes, connected to the net).

The suite is actually composed of several protocols including IP which handles the movement of data between host computers, TCP which manages the movement of data between applications, UDP which also manages the movement of data between applications but is less complex and reliable than TCP, and ICMP which transmits error messages and network traffic statistics.

REFERENCE

Inside TCP/IP Matthew Flint Arnett
Networking with Microsoft TCP/IP Drew Heywood.