COMPUTER NETWORKS

UNIT-I

SYLLABUS:

OVERVIEW OF THE INTERNET: PROTOCOL, LAYERING SCENARIO, TCP/IP PROTOCOL SUITE: THE OSI MODEL, INTERNET HISTORY STANDARDS AND ADMINISTRATION; COMPARIOSON OF THE OSI AND TCP/IP REFERENCE MODEL.

PHYSICAL LAYER: GUIDED TRANSMISSION MEDIA, WIRELESS TRANSMISSION MEDIA.

NETWORKS

A network is the interconnection of a set of devices capable of communication. In this definition, a device can be a host (or an end system as it is sometimes called) such as a large computer, desktop, laptop, workstation, cellular phone, or security system. A device in this definition can also be a connecting device such as a router (which connects the network to other networks), a switch (which connects devices together), a modem (modulator-demodulator), which changes the form of data, and so on.

(Or)

A network consists of two or more computers that are linked in order to share resources (such as printers and CDs), exchange files, or allow electronic communications. The computers on a network may be linked through cables, telephone lines, radio waves, satellites, or infrared light beams.

COMPUTER NETWORK

A computer network is a set of computers connected together for the purpose of sharing resources. The most common resource shared today is connection to the Internet. Other shared resources can include a printer or a file server. The Internet itself can be considered a computer network.

COMPUTER NETWORK APPLICATIONS

File sharing: Networking of computers helps the network users to share data files.

Hardware sharing: Users can share devices such as printers, scanners, CD-ROM drives, hard drives etc. Without computer networks, device sharing is not possible.

Application sharing: Applications can be shared over the network, and this allows to implement client/server applications

User communication: Networks allow users to communicate using e-mail, newsgroups, and video conferencing etc.

Network gaming: A lot of network games are available, which allow multi-users to play from different locations.

PROTOCOL

Definition: A protocol is a set of rules that govern data communications.

A protocol defines what is communicated, how it is communicated, and when it is communicated.

Elements of Protocol

The key elements of a protocol are syntax, semantics, and timing.

1. Syntax

The term syntax refers to the structure or format of the data, meaning the order in which they are presented. For example, a simple protocol might expect the first 8 bits of data to be the address of the sender, the second 8 bits to be the address of the receiver and the rest of the stream to be the message itself.

SENDER ADDRESS	RECEIVER ADDRESS	MESSAGE
<----------8 bits	<----------8 bits	<----------48 bits

2. Semantics

The word semantics refers to the meaning of each section of bits. How is a particular pattern to be interpreted, and what action is to be taken based on that interpretation? For example, does an address identify the route to be taken or the final destination of the message?

3. Timing

The term timing refers to two characteristics. First when data should be sent and second, how fast they can be sent. For example, if a sender produces data at 100 Mbps but the receiver can process data at only 1 Mbps, the transmission will overload the receiver and some data will be lost.

LAYERING SCENARIO

Protocol layering enables us to divide a complex task into several smaller and simpler tasks. Protocol defines the rules that both the sender and receiver and all intermediate devices need to follow to be able to communicate effectively.

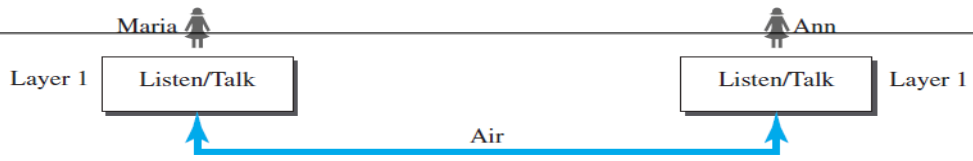
When communication is simple, we may need only one simple protocol; when the communication is complex, we may need to divide the task between different layers, in which case we need a protocol at each layer, or **protocol layering.**

FIRST SCENARIO

In the first scenario, communication is so simple that it can occur in only one layer.

Communication between Maria and Ann takes place in one layer, face to face, in the same language, as shown in Figure 2.1.

Figure 2.1 *A single-layer protocol*



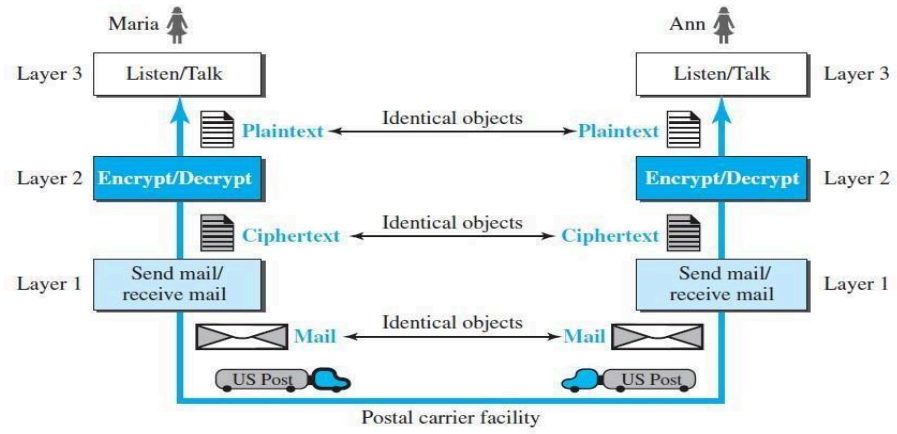
Set of rules needs to be follow

1. First, Maria and Ann should greet each other when they meet.
2. Second, they know that they should confine their vocabulary to the level of their friendship.
3. Third, each party knows that she should refrain from speaking when the other party is speaking.
4. Fourth, each party knows that the conversation should be a dialog, not a monolog: both should have the opportunity to talk about the issue.
5. Fifth, they should exchange some nice words when they leave.

SECOND SCENARIO

In the second scenario, we assume that Ann is located in a city very far from Maria. The two friends still want to continue their communication and exchange ideas. They decide to continue their conversation using regular mail through the post office. However, they do not want their ideas to be revealed by other people if the letters are intercepted. They agree on an encryption/decryption technique. The sender of the letter encrypts it to make it unreadable by an intruder; the receiver of the letter decrypts it to get the original letter.

Figure 2.2 *A three-layer protocol*



Sender side (Maria)

Maria talks to the machine at the third layer as though the machine is Ann and is listening to her. The third layer machine listens to what Maria says and creates the plaintext (a letter in English), which is passed to the second layer machine.

The second layer machine takes the plaintext, encrypts it, and creates the ciphertext, which is passed to the first layer machine.

The first layer machine, presumably a robot, takes the ciphertext, puts it in an envelope, adds the sender and receiver addresses, and mails it.

Receiver side (Ann)

At Ann's side, the first layer machine picks up the letter from Ann's mail box, recognizing the letter from Maria by the sender address. The machine takes out the ciphertext from the envelope and delivers it to the second layer machine.

The second layer machine decrypts the message, creates the plaintext, and passes the plaintext to the third-

layer machine.

The third layer machine takes the plaintext and reads it as though Maria is speaking.

PRINCIPLES OF PROTOCOL LAYERING

Two principles of protocol layering.

First Principle

The first principle dictates that if we want bidirectional communication, we need to make each layer so that it is able to perform two opposite tasks, one in each direction.

Example: the third layer task is to listen (in one direction) and talk (in the other direction). The second layer needs to be able to encrypt and decrypt. The first layer needs to send and receive mail.

Second Principle

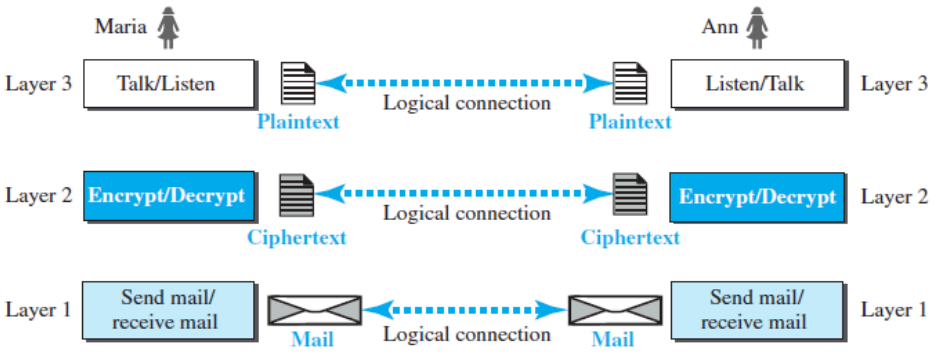
The second principle that we need to follow in protocol layering is that the two objects under each layer at both sites should be identical.

Example: the object under layer 3 at both sites should be a plaintext letter. The object under layer 2 at both sites should be a cipher text letter. The object under layer 1 at both sites should be a piece of mail.

Logical Connections

Maria and Ann can think that there is a logical (imaginary) connection at each layer through which they can send the object created from that layer.

Figure 2.3 *Logical connection between peer layers*



THE OSI MODEL

OSI stands for Open System Interconnection.

Established in 1947, the International Standards Organization (ISO) is a multinational body dedicated to worldwide agreement on international standards.

An ISO standard that covers all aspects of network communications is the Open Systems Interconnection model.

It was first introduced in the late 1970s. In 1984 it was approved as standard model for network communications architecture.

An open system is a set of protocols that allows any two different systems to communicate regardless of their underlying architecture.

The purpose of the OSI model is to show how to facilitate communication between different systems without requiring changes to the logic of the underlying hardware and software.

The OSI model is not a protocol; it is a model for understanding and designing a network architecture that is flexible, robust, and interoperable.

ISO is the organization. OSI is the model.

LAYERED ARCHITECTURE

The OSI model is composed of seven ordered layers: physical (layer 1), data link (layer 2), network (layer 3), transport (layer 4), and session (layer 5), and presentation (layer 6), and application (layer 7).

Peer-to-Peer Processes

Within a single machine, each layer calls upon the services of the layer just below it. Layer 3, for example, uses the services provided by layer 2 and provides services for layer 4.

Between machines, layer *x* on one machine communicates with layer *x* on another machine. This communication is governed by an agreed-upon series of rules and conventions called protocols.

The processes on each machine that communicate at a given layer are called peer-to-peer processes.

The OSI model

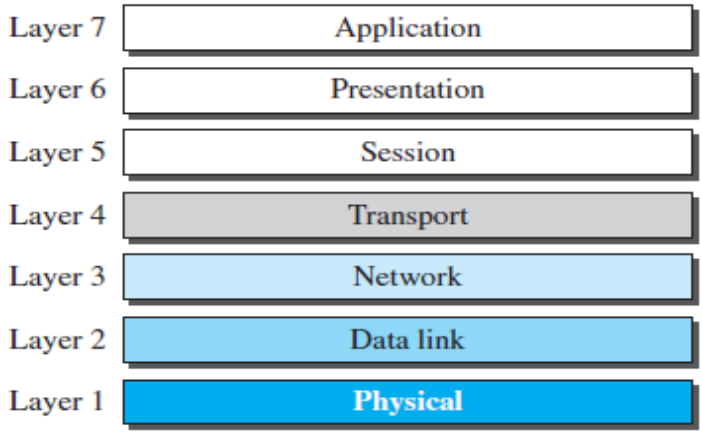
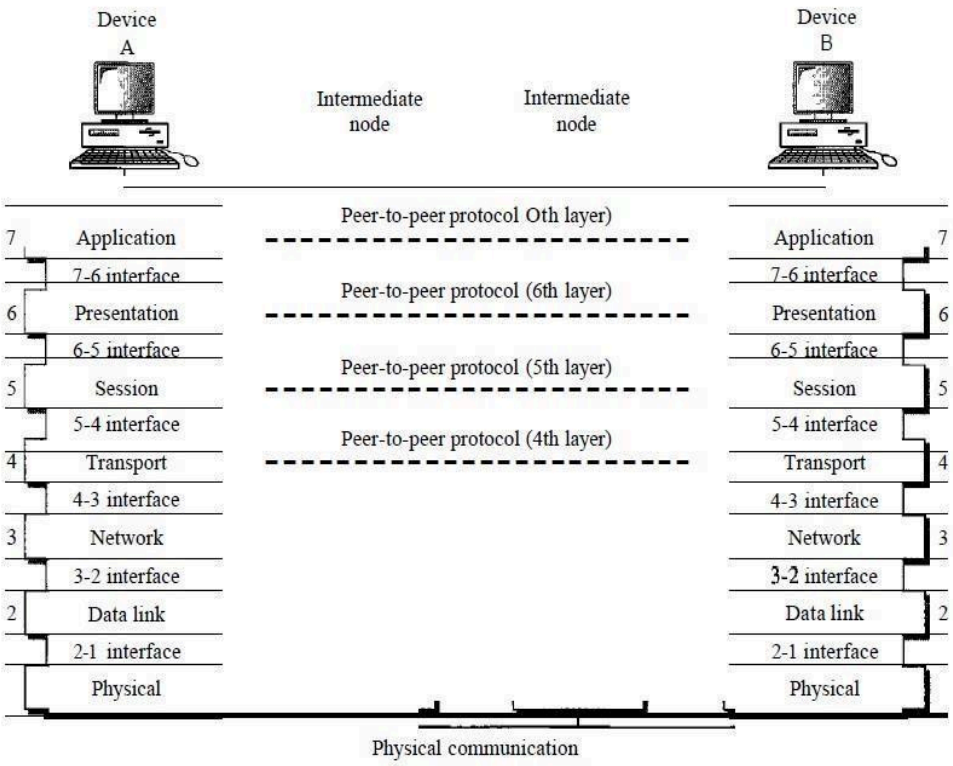


FIG 2.3: SEVEN LAYERS OF OSI MODEL

Interfaces between Layers

The passing of the data and network information down through the layers of the sending device and back up through the layers of the receiving device is made possible by an interface between each pair of adjacent layers. Each interface defines the information and services a layer must provide for the layer above it. Well-defined interfaces and layer functions provide modularity to a network.

Figure 2.3 *The interaction between layers in the OSI model*



Organization of the Layers

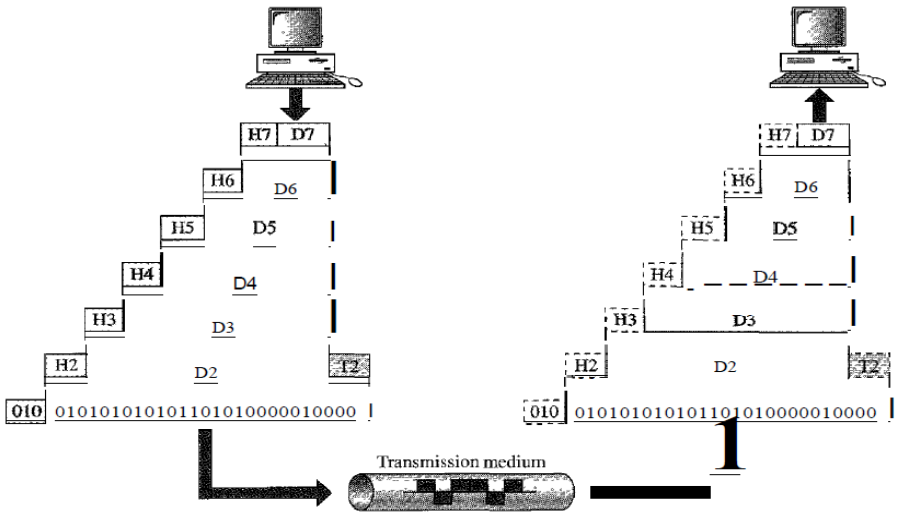
The seven layers can be thought of as belonging to three subgroups.

- ✔ ***Layers 1, 2, and 3- physical, data link, and network-are the network support layers;*** they deal with the physical aspects of moving data from one device to another (such as electrical specifications, physical connections, physical addressing, and transport timing and Reliability).
- ✔ ***Layers 5, 6, and 7-session, presentation, and application-can be thought of as the user support layers;*** they allow interoperability among unrelated software systems.
- ✔ Layer 4, the transport layer, links the two subgroups and ensures that what the lower layers have transmitted is in a form that the upper layers can use.

The upper OSI layers are almost always implemented in software; lower layers are a combination of hardware and software, except for the physical layer, which is mostly hardware.

In Figure 2.4, which gives an overall view of the OSI layers, D7 means the data unit at layer 7, D6 means the data unit at layer 6, and so on. The process starts at layer 7 (the application layer), then moves from layer to layer in descending, sequential order. At each layer, a **header,** or possibly a **trailer,** can be added to the data unit. Commonly, the trailer is added only at layer 2. When the formatted data unit passes through the physical layer (layer 1), it is changed into an electromagnetic signal and transported along a physical link.

Figure 2.4 *An exchange using the OS! model*



Encapsulation

Packet (header and data) at level 7 is encapsulated in a packet at level 6. The whole packet at level 6 is encapsulated in a packet at level 5, and so on. In other words, *the data portion of a packet at level N - 1 carries the whole packet (data and header and maybe trailer) from level N. The concept is called encapsulation.*

LAYERS IN THE OSI MODEL

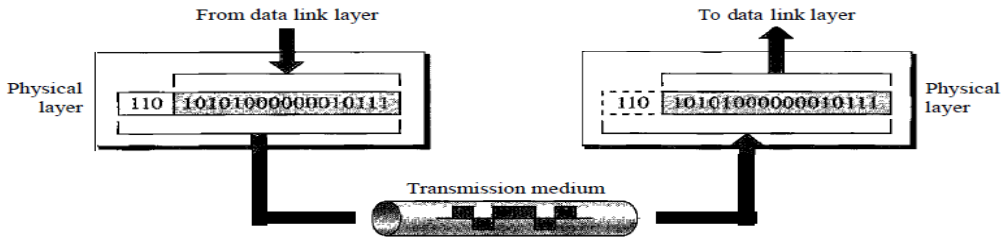
1. PHYSICAL LAYER

It deals with the mechanical and electrical specifications of the interface and transmission medium.

The physical layer is also concerned with the following:

Physical characteristics of interfaces and medium: The physical layer defines the characteristics of the interface between the devices and the transmission medium. It also defines the type of transmission medium.

Figure 2.5 *Physical layer*



The physical layer is responsible for movements of individual bits from one hop (node) to the next.

Representation of bits: The physical layer data consists of a stream of bits (sequence of Os or 1s) with no interpretation. To be transmitted, bits must be encoded into signals--electrical or optical. The physical layer defines the type of encoding (how Os and I s are changed to signals).

Data rate (The transmission rate): the number of bits sent each second-is also defined by the physical layer. In other words, the physical layer defines the duration of a bit, which is how long it lasts.

Synchronization of bits: The sender and receiver not only must use the same bit rate but also must be synchronized at the bit level. In other words, the sender and the receiver clocks must be synchronized.

Line configuration: The physical layer is concerned with the connection of devices to the media. In a point-to-point configuration, two devices are connected through a dedicated link. In a multipoint configuration, a link is shared among several devices.

Physical topology: The physical topology defines how devices are connected to make a network. Devices can be connected by using a mesh topology (every device is connected to every other device), a star topology (devices are connected through a central device), a ring topology (each device is connected to the next, forming a ring), a bus topology (every device is on a common link), or a hybrid topology (this is a combination of two or more topologies).

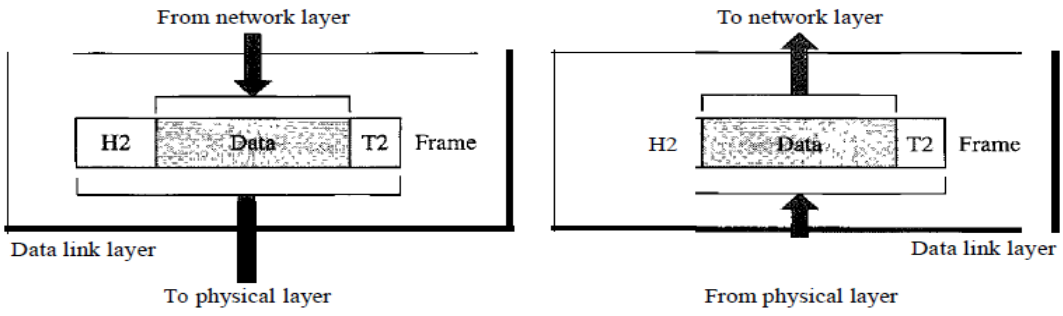
Transmission mode: The physical layer also defines the direction of transmission between two devices: simplex, half-duplex, or full-duplex. In simplex mode, only one device can send; the other can only receive. The simplex mode is a one-way communication. In the half-duplex mode, two devices can send and receive, but not at the same time. In a full-duplex (or simply duplex) mode, two devices can send and receive at the same time.

2. DATALINK LAYER

The data link layer transforms the physical layer, a raw transmission facility, to a reliable link. It makes the physical layer appear error-free to the upper layer (network layer).

The data link layer is responsible for moving frames from one hop (node) to the next.

Figure 2.6 *Data link layer*



Other responsibilities of the data link layer include the following:

Framing: The data link layer divides the stream of bits received from the network layer into manageable data units called frames.

Physical addressing: If frames are to be distributed to different systems on the network, the data link layer adds a header to the frame to define the sender and/or receiver of the frame. If the frame is intended for a system outside the sender's network, the receiver address is the address of the device that connects the network to the next one.

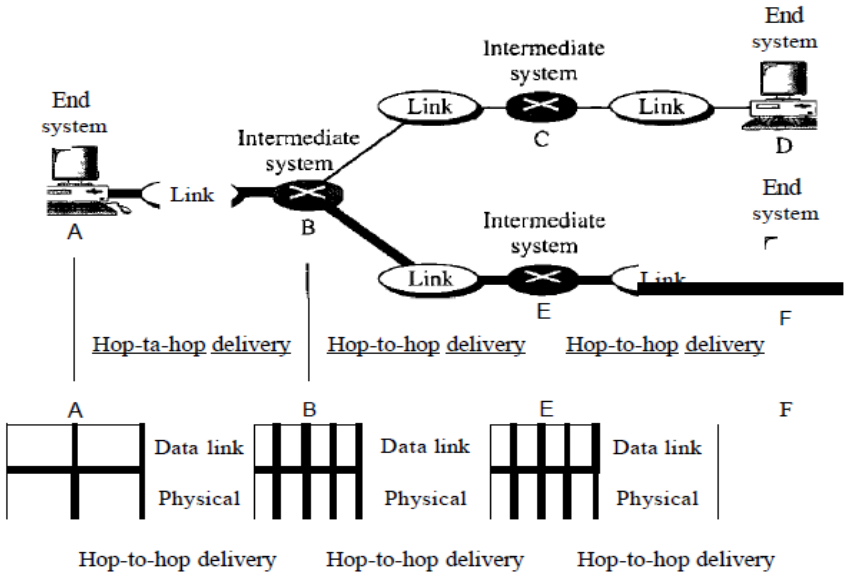
Flow control: If the rate at which the data are absorbed by the receiver is less than the rate at which data are produced in the sender, the data link layer imposes a flow control mechanism to avoid overwhelming the receiver.

Error control: The data link layer adds reliability to the physical layer by adding mechanisms to detect and retransmit damaged or lost frames. It also uses a mechanism to recognize duplicate frames. Error control is normally achieved through a trailer added to the end of the frame.

Access control: When two or more devices are connected to the same link, data link layer protocols are necessary to determine which device has control over the link at any given time.

Figure 2.7 illustrates hop-to-hop (node-to-node) delivery by the data link layer.

Figure 2.7 *Hop-to-hop delivery*



As the figure shows, communication at the data link layer occurs between two adjacent nodes. To send data from A to F, three partial deliveries are made. First, the data link layer at A sends a frame to the data link layer at B (a router). Second, the data link layer at B sends a new frame to the data link layer at E. Finally, the data link layer at E sends a new frame to the data link layer at F.

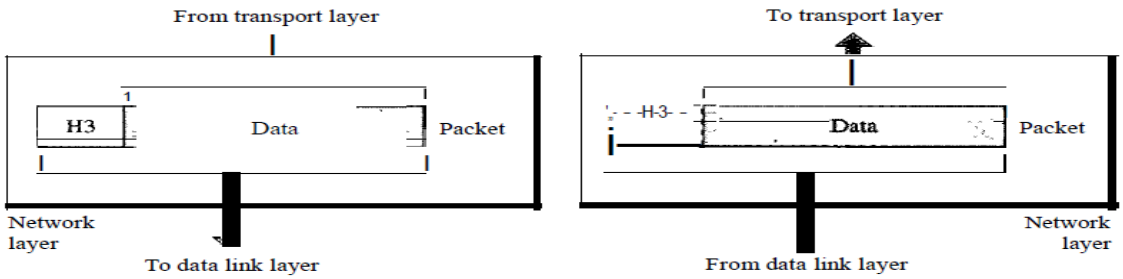
3. NETWORK LAYER

The network layer is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links). Whereas the data link layer oversees the delivery of the packet between two systems on the same network (links), the network layer ensures that each packet gets from its point of origin to its final destination.

Other responsibilities of the network layer include the following:

Logical addressing: The physical addressing implemented by the data link layer handles the addressing problem locally. If a packet passes the network boundary, we need another addressing system to help distinguish the source and destination systems. The network layer adds a header to the packet coming from the upper layer that, among other things, includes the logical addresses of the sender and receiver.

Figure 2.8 *Network layer*

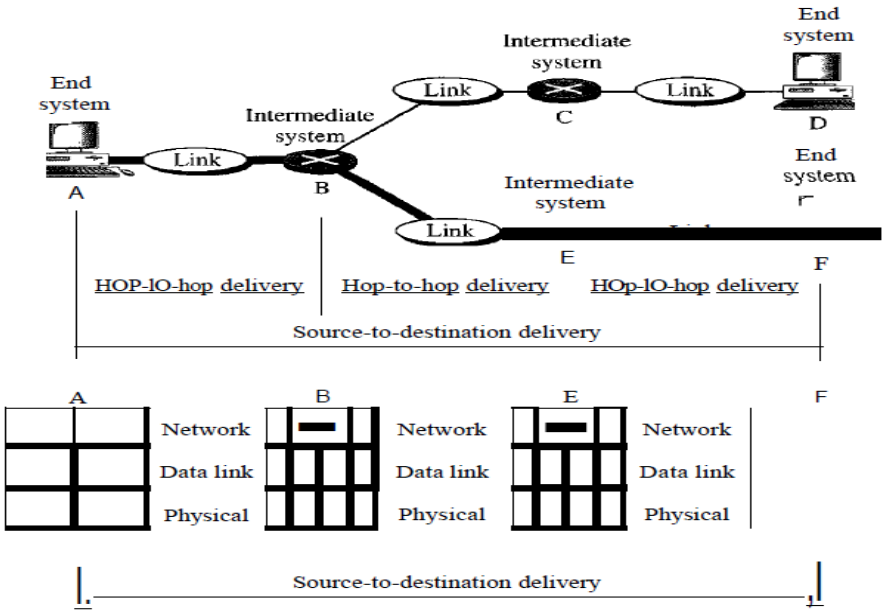


The network layer is responsible for the delivery of individual packets from the source host to the destination host.

Routing: When independent networks or links are connected to create internetworks (network of networks) or a large network, the connecting devices (called routers or switches) route or switch the packets to their final destination. One of the functions of the network layer is to provide this mechanism.

Figure 2.9 illustrates end-to-end delivery by the network layer.

Figure 2.9 *Source-to-destination delivery*

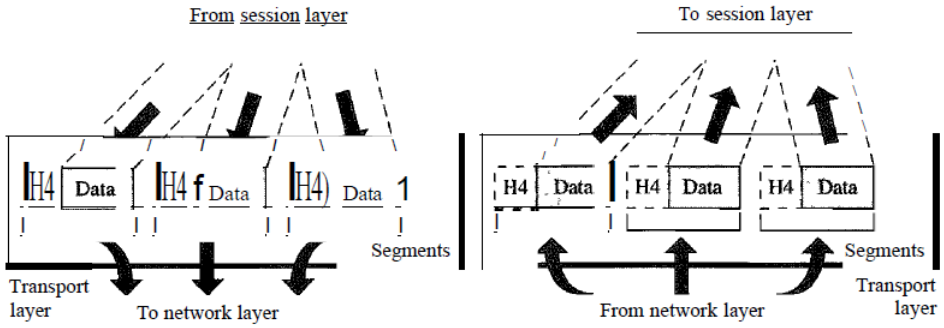


As the figure shows, now we need a source-to-destination delivery. The network layer at A sends the packet to the network layer at B. When the packet arrives at router B, the router makes a decision based on the final destination (F) of the packet router B uses its routing table to find that the next hop is router E. The network layer at B, therefore, sends the packet to the network layer at E. The network layer at E, in turn, sends the packet to the network layer at F.

4. TRANSPORT LAYER

The transport layer is responsible for process-to-process delivery of the entire message. A process is an application program running on a host.

Figure 2.10 *Transport layer*



The transport layer is responsible for the delivery of a message from one process to another.

Other responsibilities of the transport layer include the following:

Service-point addressing: Computers often run several programs at the same time. For this reason, source-to-destination delivery means delivery not only from one computer to the next but also from a specific process (running program) on one computer to a specific process (running program) on the other. The transport Layer header must therefore include a type of address called a *service-point address* (or port address). The network layer gets each packet to the correct computer; the transport layer gets the entire message to the correct process on that computer.

Segmentation and reassembly: A message is divided into transmittable segments, with each segment containing a sequence number. These numbers enable the transport layer to reassemble the message correctly upon arriving at the destination and to identify and replace packets that were lost in transmission.

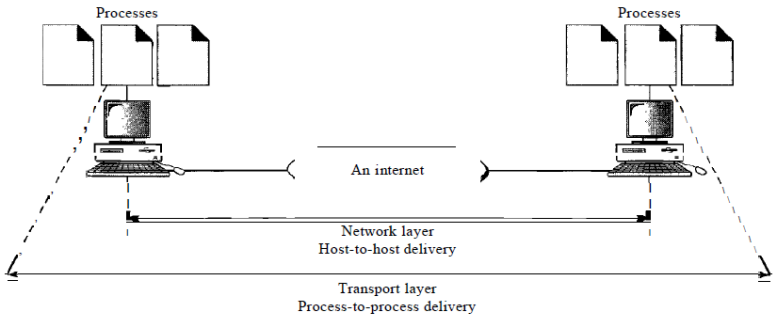
Connection control: The transport layer can be either connectionless or connection oriented. A connectionless transport layer treats each segment as an independent packet and delivers it to the transport layer at the destination machine. A connection oriented transport layer makes a connection with the transport layer at the destination machine first before delivering the packets. After all the data are transferred, the connection is terminated.

Flow control: Like the data link layer, the transport layer is responsible for flow control. However, flow control at this layer is performed end to end rather than across a single link.

Error control: Like the data link layer, the transport layer is responsible for error control. However, error control at this layer is performed process-to process rather than across a single link. The sending transport layer makes sure that the entire message arrives at the receiving transport layer without error (damage, loss, or duplication). Error correction is usually achieved through retransmission.

Figure 2.11 illustrates process-to-process delivery by the transport layer.

Figure 2.11 *Reliable process-to-process delivery of a message*

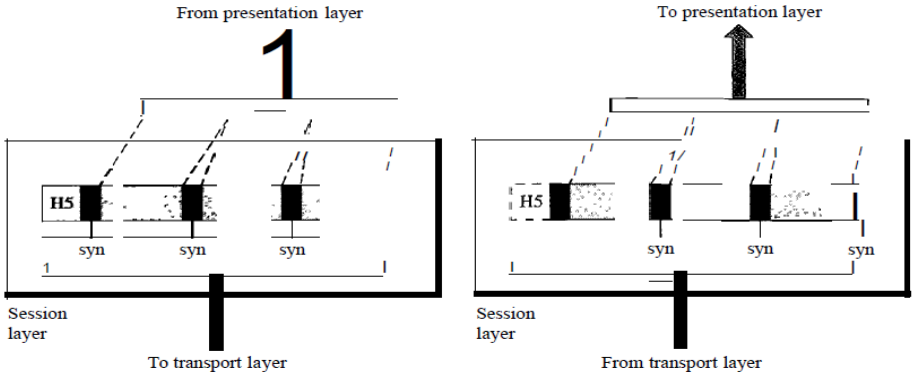


The session layer is responsible for dialog control and synchronization.

5. SESSION LAYER

The session layer is the network *dialog controller*. It establishes, maintains, and synchronizes the interaction among communicating systems. Figure 2.12 illustrates the relationship of the session layer to the transport and presentation layers.

Figure 2.12 *Session layer*



Specific responsibilities of the session layer include the following:

Dialog control: The session layer allows two systems to enter into a dialog. It allows the communication between two processes to take place in either half duplex (one way at a time) or full-duplex (two ways at a time) mode.

Synchronization: The session layer allows a process to add checkpoints, or synchronization points, to a stream of data. For example, if a system is sending a file of 2000 pages, it is advisable to insert checkpoints after every 100 pages to ensure that each 100-page unit is received and acknowledged independently. In this case, if a crash happens during the transmission of page 523, the only pages that need to be resent after system recovery are pages 501 to 523. Pages previous to 501 need not be resent.

6. PRESENTATION LAYER

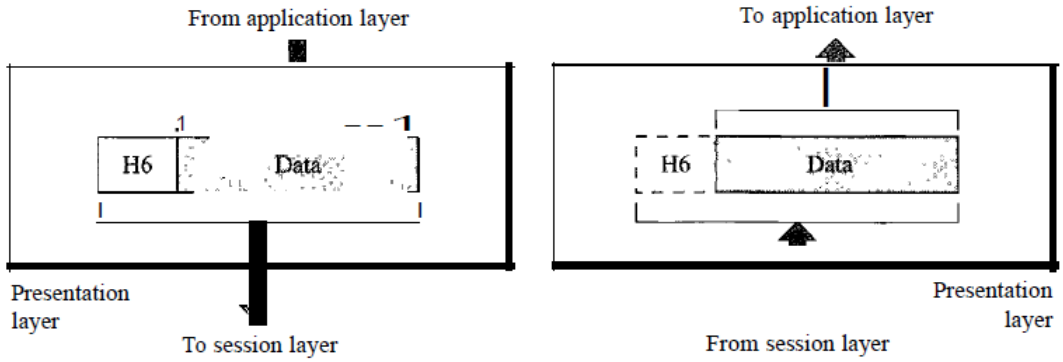
The presentation layer is concerned with the syntax and semantics of the information exchanged between two systems.

Specific responsibilities of the presentation layer include the following:

Compression: Data compression reduces the number of bits contained in the information. Data compression becomes particularly important in the transmission of multimedia such as text, audio, and video.

Translation: The presentation layer at the sender changes the information from its sender-dependent format into a common format. The presentation layer at the receiving machine changes the common format into its receiver-dependent format.

Figure 2.13 *Presentation layer*



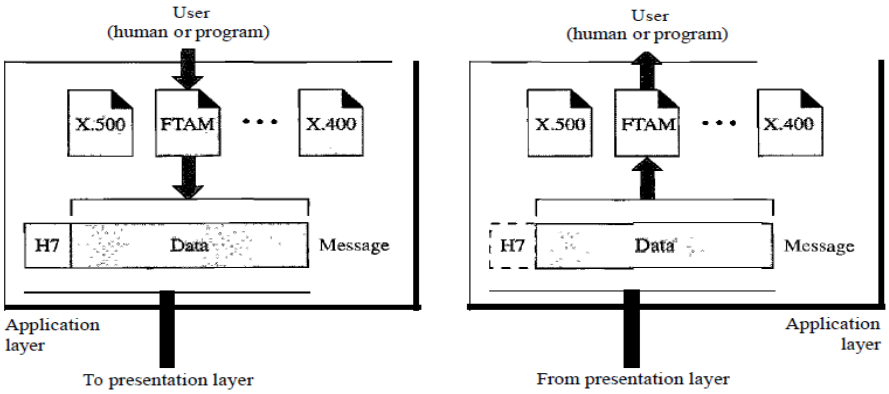
The presentation layer is responsible for translation, compression, and encryption.

Encryption: To carry sensitive information, a system must be able to ensure privacy. Encryption means that the sender transforms the original information to another form and sends the resulting message out over the network. Decryption reverses the original process to transform the message back to its original form.

6. APPLICATION LAYER

The application layer enables the user, whether human or software, to access the network. It provides user interfaces and support for services such as electronic mail, remote file access and transfer, shared database management, and other types of distributed information services.

Figure 2.14 *Application layer*



The application layer is responsible for providing services to the user.

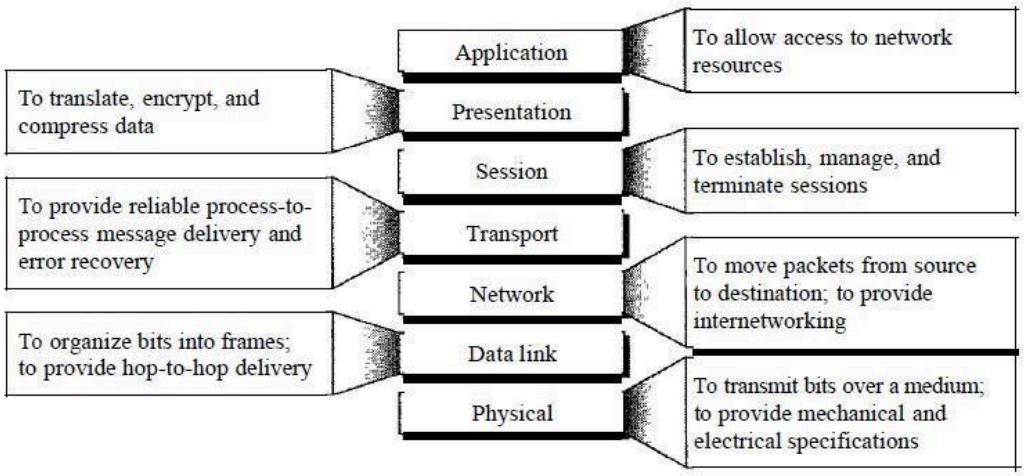
Specific services provided by the application layer include the following:

Network virtual terminal: A network virtual terminal is a software version of a physical terminal, and it allows a user to log on to a remote host.

Mail services: This application provides the basis for e-mail forwarding and storage.

Directory services: This application provides distributed database sources and access for global information about various objects and services.

Figure 2.15 *Summary of layers*



TCP/IP PROTOCAL SUITE

TCP/IP Stands for Transmission Control Protocol/Internet Protocol

It was started in early 1970s, very soon it was popularized. TCP/IP allows computers to communicate resources across networks.

TCP/IP is a protocol suite (a set of protocols organized in different layers) is family of protocols used in the Internet today. It is not a single protocol.

It was funded by US Military (US DoD). ARPANET (Advanced Research Project Agency Network) is the initial version of TCP/IP.

It is a hierarchical protocol made up of interactive modules, each of which provides a specific functionality. The term hierarchical means that each upper level protocol is supported by the services provided by one or more lower level protocols.

TCPIIP protocol suite is made of five layers: physical, data link, network, transport, and application.

The TCPIIP protocol suite was developed prior to the OSI model. Therefore, the layers in the TCP/IP protocol suite do not exactly match those in the OSI model.

Layers in the TCP/IP Protocol Suite

5.APPLICATION LAYER
4.TRANSPORT LAYER
3.NETWORK LAYER
2.DATALINK LAYER
1.PHYSICAL LAYER

Description of Each Layer PHYSICAL LAYER

Physical layer is responsible for carrying individual bits in a frame across the link.

The communication between two devices at the physical layer is still a logical communication because there is another, hidden layer, the transmission media, under the physical layer.

Two devices are connected by a transmission medium (cable or air). Transmission medium does not carry bits; it carries electrical or optical signals. So the bits received in a frame from the data-link layer are transformed and sent through the transmission media, there are several protocols that transform a bit to a signal.

DATA-LINK LAYER

The routers are responsible for choosing the *best* links. However, when the next link to travel is determined by the router, the data-link layer is responsible for taking the datagram and moving it across the link.

The link can be a wired LAN with a link-layer switch, a wireless LAN, a wired WAN, or a wireless WAN.

In each case, the data-link layer is responsible for moving the packet through the link.

TCP/IP does not define any specific protocol for the data-link layer. It supports all the standard and proprietary protocols.

The data-link layer takes a datagram and encapsulates it in a packet called a *frame*.

Each link-layer protocol may provide a different service. Some link-layer protocols provide complete error detection and correction, some provide only error correction.

NETWORK LAYER

The network layer is responsible for host-to-host communication and routing the packet through possible routes.

The network layer is responsible for creating a connection between the source computer and the destination computer.

At the network layer (or, more accurately, the internetwork layer), *TCP/IP* supports the Internetworking Protocol. IP, in turn, uses four supporting protocols: ARP, RARP, ICMP, and IGMP.

INTERNETWORKING PROTOCOL (IP)

- The Internetworking Protocol (IP) is the transmission mechanism used by the TCP/IP protocols.
- IP provides best-effort delivery service. The term *best effort* means that IP provides no error checking or tracking.
- IP is a connectionless protocol that provides no flow control, no error control, and no congestion control services.
- IP defines the format of the packet, called a datagram at the network layer.
- IP defines the format and the structure of addresses used in this layer.
- IP is also responsible for routing a packet from its source to its destination, which is achieved by each router forwarding the datagram to the next router in its path.
- IP transports data in packets called *datagram's,* each of which is transported separately.

The network layer also has some auxiliary protocols that help IP in its delivery and routing tasks.

ARP (Address Resolution Protocol)

The Address Resolution Protocol (ARP) is used to associate a logical address with a physical address. On a typical physical network, such as a LAN, each device on a link is identified by a physical or station address, usually imprinted on the network interface card (NIC). ARP is used to find the physical address of the node when its Internet address is known.

RARP (Reverse Address Resolution Protocol)

The Reverse Address Resolution Protocol (RARP) allows a host to discover its Internet address when it knows only its physical address. It is used when a computer is connected to a network for the first time or when a diskless computer is booted.

ICMP (Internet Control Message Protocol)

The Internet Control Message Protocol (ICMP) is a mechanism used by hosts and gateways to send notification of datagram problems back to the sender. ICMP sends query and error reporting messages. (i.e. ICMP report some problems when routing a packet.)

IGMP(Internet Group Message Protocol)

The Internet Group Message Protocol (IGMP) is used to facilitate the simultaneous transmission of a message to a group of recipients.(i.e helps IP in multitasking).

Dynamic Host Configuration Protocol (DHCP)

It helps IP to get the network-layer address for a host.

TRANSPORT LAYER

The logical connection at the transport layer is also end-to-end.The transport layer at the source host gets the message from the application layer, encapsulates it in a transport layer packet (called a *segment* or a *user datagram* in different protocols) and sends it, through the logical (imaginary) connection, to the transport layer at the destination host.

In other words, the transport layer is responsible for giving services to the application layer: to get a message from an application program running on the source host and deliver it to the corresponding application program on the destination host.

Transport layer was represented in TCP/IP by two protocols: TCP and UDP.

UDP and TCP are transport level protocols responsible for delivery of a message from a process (running program) to another process.

TCP (TRANSMISSION CONTROL PROTOCOL)

- It is a connection-oriented protocol that first establishes a logical connection between transport layers at two hosts before transferring data.
- TCP provides flow control (matching the sending data rate of the source host with the receiving data rate of the destination host to prevent overwhelming the destination), error control (to guarantee that the segments arrive at the destination without error and resending the corrupted ones), and congestion control to reduce the loss of segments due to congestion in the network.

UDP (USER DATAGRAM PROTOCOL)

- It is a connectionless protocol that transmits user datagram's without first creating a logical connection.
- In UDP, each user datagram is an independent entity without being related to the previous or the next one (the meaning of the term connectionless).
- UDP is a simple protocol that does not provide flow, error, or congestion control.

SCTP (STREAM CONTROL TRANSMISSION PROTOCOL): It's a new transport layer protocol has been devised to meet the needs of some newer applications that are emerging in the multimedia.

APPLICATION LAYER

The logical connection between the two application layers is end to- end. The two application layers exchange messages between each other as though there were a bridge between the two layers.

Communication at the application layer is between two processes (two programs running at this layer). To communicate, a process sends a request to the other process and receives a response.

Process-to-process communication is the duty of the application layer.

The application layer in the Internet includes many predefined protocols

Hypertext Transfer Protocol (HTTP) useful in accessing the World Wide Web (WWW).
Simple Mail Transfer Protocol (SMTP) is the main protocol used in electronic mail (e-mail) service.

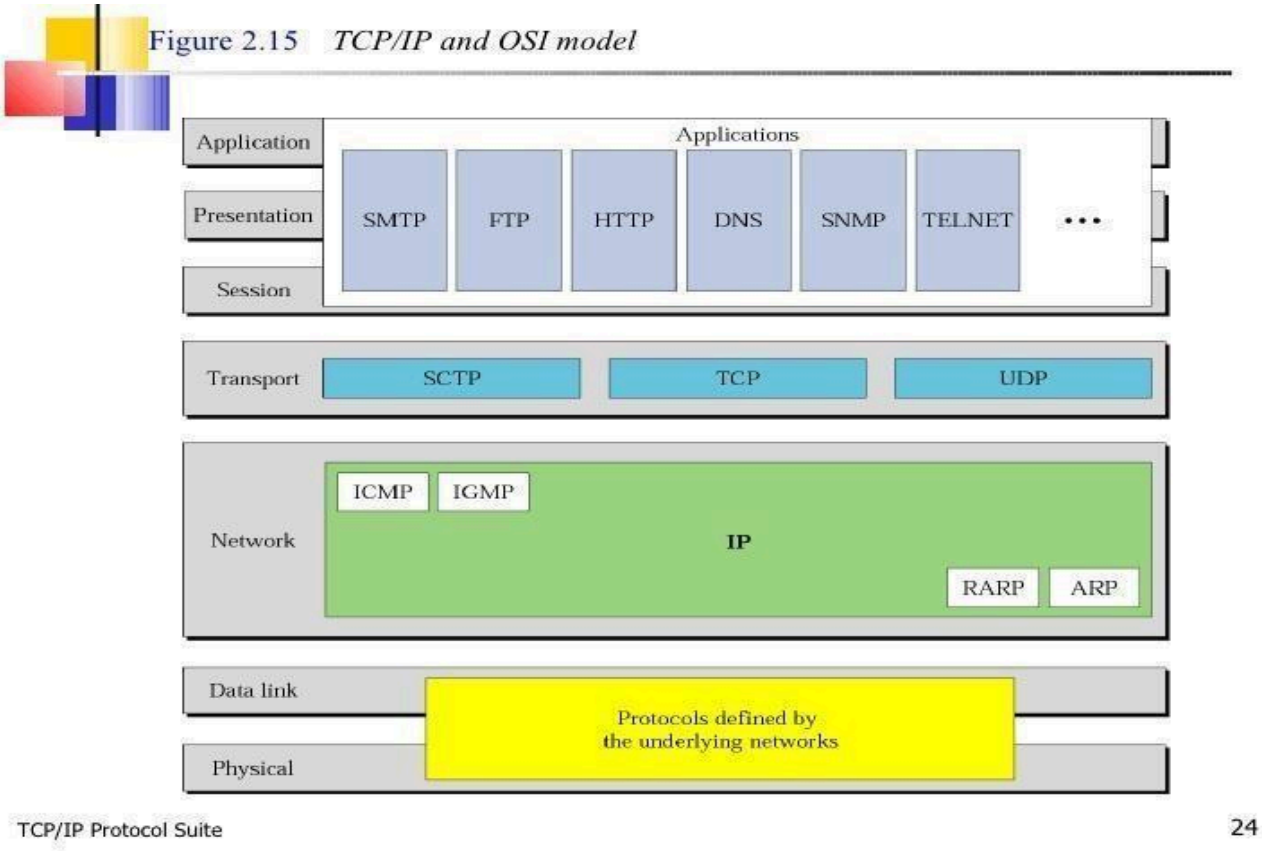
File Transfer Protocol (FTP) is used for transferring files from one host to another.

Terminal Network (TELNET) and **Secure Shell (SSH**) are used for accessing a site remotely.

Simple Network Management Protocol (SNMP) is used by an administrator to manage the Internet at global and local levels.

Domain Name System (DNS) is used by other protocols to find the network-layer address of a computer.

Figure 2.15 *TCP/IP and OSI model*



TCP/IP Protocol Suite 24

LACK OF OSI MODEL'S SUCCESS

First, OSI was completed when TCP/IP was fully in place and a lot of time and money had been spent on the suite; changing it would cost a lot.

Second, some layers in the OSI model were never fully defined.

Example: services provided by the presentation and the session layers were listed in the document, actual protocols for these two layers were not fully defined, nor were they fully described, and the corresponding software was not fully developed.

Third, when OSI was implemented by an organization in a different application, it did not show a high enough level of performance to entice the Internet authority to switch from the TCP/IP protocol suite to the OSI model.

Key Differences between TCP/IP and OSI Model

1. TCP/IP is a client-server model, i.e. when the client requests for service it is provided by the server. Whereas, OSI is a conceptual model.
2. TCP/IP is a standard protocol used for every network including the Internet, whereas, OSI is not a protocol but a reference model used for understanding and designing the system architecture.
3. TCP/IP is a four layered model, whereas, OSI has seven layers.
4. TCP/IP follows Vertical approach. On the other hand, OSI Model supports Horizontal approach.
5. TCP/IP is Tangible, whereas, OSI is not.
6. ~~TCP/IP follows top to bottom approach, whereas, OSI Model follows a bottom-up approach.~~

OVERVIEW OF INTERNET

Internet: (lowercase *i*) It is two or more networks that can communicate with each other.

Internet: (uppercase *I)*, It is composed of thousands of interconnected networks.

INTRANET: Network basically local to company. Users within company can find resources without having go outside. **Ex:** LAN's, Private WAN's, MAN's.

EXTRANET: IT is extended INTRANET. Here certain internal services are made available to known external users or external business partners.

CONCEPTUAL VIEW OF THE INTERNET

The figure 1.15 shows the Internet as several backbones, provider networks, and customer networks.

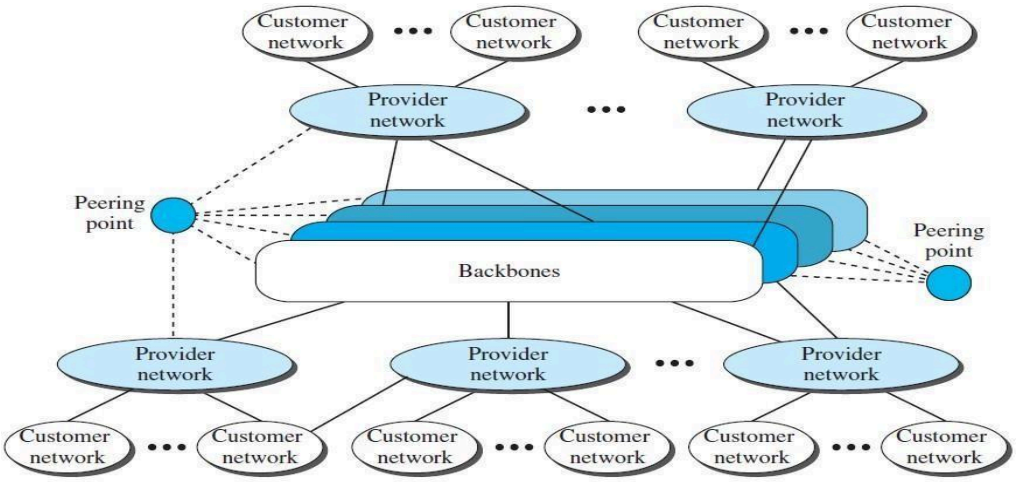
At the top level, the *backbones* are large networks owned by some communication companies such as Sprint, Verizon (MCI), AT&T, and NTT. The backbone networks are connected through some complex switching systems, called *peering points*.

At the second level, there are smaller networks, called *provider networks* that use the services of the backbones for a fee. The provider networks are connected to backbones and sometimes to other provider networks.

The *customer networks* are networks at the edge of the Internet that actually use the services provided by the Internet. They pay fees to provider networks for receiving services.

Backbones and provider networks are also called **Internet Service Providers (ISPs).** The backbones are often referred to as *international ISPs;* the provider networks are often referred to as *national* or *regional ISPs*.

Figure 1.15 *The Internet today*



ACCESSING THE INTERNET

Using Telephone Networks

This can be done in two ways.

- ***Dial-up service.*** Add a modem to the telephone line that converts data to voice. The software installed on the computer dials the ISP and imitates making a telephone connection. Unfortunately, the dial-up service is very slow, and when the line is used for Internet connection, it cannot be used for telephone (voice) connection. It is only useful for small residences.
- ***DSL Service.*** Since the advent of the Internet, some telephone companies have upgraded their telephone lines to provide higher speed Internet services to residences or small businesses. The DSL service also allows the line to be used simultaneously for voice and data communication.

Using Cable Networks

Using Wireless Networks

Wireless connectivity has recently become increasingly popular. A household or a small business can use a combination of wireless and wired connections to access the Internet. With the growing wireless WAN access, a household or a small business can be connected to the Internet through a wireless WAN.

Direct Connection to the Internet

A large organization or a large corporation can itself become a local ISP and be connected to the Internet. This can be done if the organization or the corporation leases a high-speed WAN from a carrier provider and connects itself to a regional ISP. For example, a large university with several campuses can create an internetwork and then connect the internetwork to the Internet.

HISTORY OF INTERNET

Early History

Before 1960*: Communication networks, such as telegraph and telephone networks. These networks were suitable for constant-rate communication at that time, which means that after a connection was made between two users, the encoded message (telegraphy) or voice (telephony) could be exchanged.

A computer network, on the other hand, should be able to handle *bursty* data, which means data received at variable rates at different times. The world needed to wait for the packet-switched network to be invented.

Birth of Packet-Switched Networks

1961*: The theory of packet switching for bursty traffic was first presented by Leonard Klein rock at MIT. At the same time, two other researchers, Paul Baran at Rand Institute and Donald Davies at national Physical Laboratory in England, published some papers about packet-switched networks.

ARPANET

Mid-1960s*: Mainframe computers in research organizations were stand-alone devices. Computers from different manufacturers were unable to communicate with one another.

The Advanced Research Projects Agency (ARPA) in the Department of Defense (DOD) was interested in finding a way to connect computers so that the researchers they funded could share their findings, thereby reducing costs and eliminating duplication of effort.

1967*: At an Association for Computing Machinery (ACM) meeting, ARPA presented its ideas for the Advanced Research Projects Agency Network (ARPANET), a small network of connected computers. The idea was that each host computer (not necessarily from the same manufacturer) would be attached to a specialized computer, called an interface message processor (IMP). The IMPs, in turn, would be connected to each other. Each IMP had to be able to communicate with other IMPs as well as with its own attached host.

1969*: ARPANET was a reality. Four nodes, at the University of California at Los Angeles (UCLA), the University of California at Santa Barbara (UCSB), Stanford Research Institute (SRI), and the University of Utah, were connected via the IMPs to form a network. Software called the Network Control Protocol (NCP) provided communication between the hosts.

1970: ARPANET creates precursor to TCP.
1971: Telnet (remote login) and FTP (transfer files between machines) are made available

Birth of the Internet

1972*

1. Vint Cerf and Bob Kahn, both of whom were part of the core ARPANET group, collaborated on what they called the *Internetting Project*. They wanted to link dissimilar networks so that a host on one network could communicate with a host on another. Cerf and Kahn devised the idea of a device called a ***gateway*** to serve as the intermediary hardware to transfer data from one network to another.
2. 1st version of E-Mail message was sent.

TCP/IP

1973*

1. Cerf and Kahn's landmark 1973 paper outlined the protocols to achieve end-to-end delivery of data. This was a new version of NCP. This paper on transmission control protocol (TCP) included concepts such as encapsulation, the datagram, and the functions of a gateway. (**New version of NCP is available i.e. TCP**)
2. This ARPA Internet now became the focus of the communication effort.

Around this time, responsibility for the ARPANET was handed over to the Defense Communication Agency (DCA).

1974: **TCP** was recognized as Standard & it was used for communication across a system of networks.

1977-81*: Internet consisting of three different networks (ARPANET, Packet radio, and packet satellite) was successfully demonstrated. Communication between networks was now possible.

TCP split into two protocols: **Transmission Control Protocol (TCP)** and **Internet Protocol (IP).**

IP would handle datagram routing while TCP would be responsible for higher level functions such as segmentation, reassembly, and error detection. The new combination became known as TCP/IP.

1982: US DoD starts building defense data networks based on ARPANET Technology. (US DoD developed its own defense network).

1981*: CSNET

CSNET was created. **Computer Science Network (CSNET)** was a network sponsored by the National Science Foundation (NSF). The network was conceived by universities that were ineligible to join ARPANET due to an absence of ties to the Department of Defense. CSNET was a less expensive network; there were no redundant links and the transmission rate was slower.

1983*

TCP/IP became the official protocol for the ARPANET. Those who wanted to use the Internet to access a computer on a different network had to be running TCP/IP.

ARPANET split into two networks: **Military Network (MILNET)** for military users and ARPANET for nonmilitary users.

Internet started in shape.

1986*: NSFNET

With the success of CSNET, the NSF in 1986 sponsored the **National Science Foundation Network (NSFNET),** a backbone that connected five supercomputer centers located throughout the United States. Community networks were allowed access to this backbone, a T-1 line with a 1.544-Mbps data rate, thus providing connectivity throughout the United States.

1990*: ARPANET was officially retired and replaced by NSFNET.

1991*: ANSNET

U.S. government decided that NSFNET was not capable of supporting the rapidly increasing Internet traffic.

Three companies, IBM, Merit, and Verizon, filled the void by forming a nonprofit organization called Advanced Network & Services (ANS) to build a new, high-speed Internet backbone called **Advanced Network Services Network (ANSNET).**

New applications like ARCHIE (developed as FTP search engine) and GOPHER (more intelligent version of ARCHIE) were released.

1992: Internet links more than 17000 networks in 33 countries.

**1993*: ** WWW was launched.

**1995*: ** internet service providers started offering services.

INTERNET TODAY

World Wide Web

The 1990s saw the explosion of Internet applications due to the emergence of the World Wide Web (WWW). The Web was invented at CERN by Tim Berners-Lee. This invention has added the commercial applications to the Internet.

Multimedia

Recent developments in the multimedia applications such as voice over IP (telephony), video over IP (Skype), view sharing (YouTube), and television over IP (PPLive) has increased the number of users and the amount of time each user spends on the network.

Peer-to-Peer Applications

Peer-to-peer networking is also a new area of communication with a lot of potential.

INTERNET APPLICATIONS

Telnet

FTP

E-Mail

WWW

STANDARDS AND ADMINISTRATION

INTERNET STANDARDS

An **Internet standard** is a thoroughly tested specification that is useful to and adhered to by those who work with the Internet. It is a formalized regulation that must be followed.

INTERNET DRAFT:

It is a working document (a work in progress) with no official status and a six-month lifetime.

Request for Comment (RFC)

Upon recommendation from the Internet authorities, a draft may be published as a **Request for Comment (RFC).** Each RFC is edited, assigned a number, and made available to all interested parties. RFCs go through maturity levels and are categorized according to their requirement level.

MATURITY LEVELS

An RFC, during its lifetime, falls into one of six *maturity levels:* proposed standard, draft standard, Internet

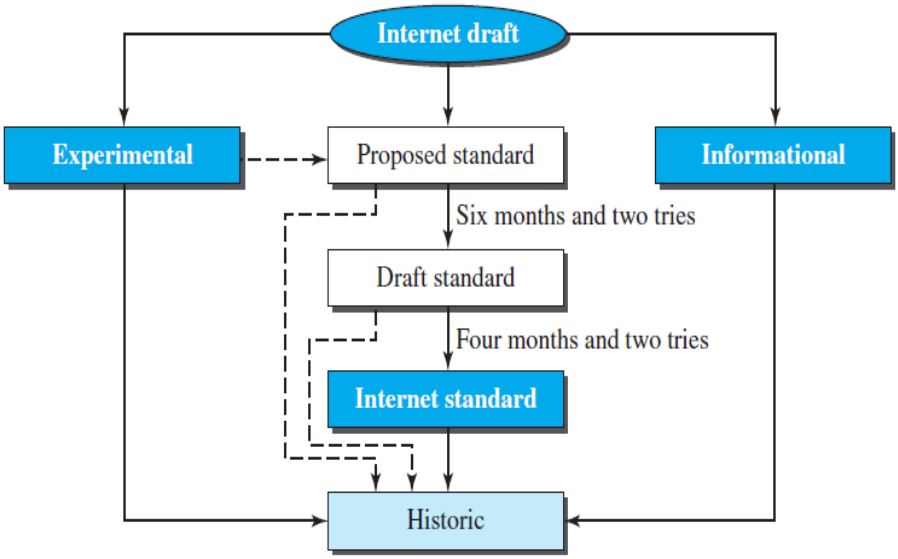
standard, historic, experimental, and informational (see Figure 1.16).

Proposed Standard: A proposed standard is a specification that is stable, well understood, and of sufficient interest to the Internet community. At this level, the specification is usually tested and implemented by several different groups.

Draft Standard: A proposed standard is elevated to draft standard status after at least two successful independent and interoperable implementations. Barring difficulties, a draft standard, with modifications if specific problems are encountered, normally becomes an Internet standard.

Internet Standard: A draft standard reaches Internet standard status after demonstrations of successful implementation.

Figure 1.16 *Maturity levels of an RFC*



Historic: The historic RFCs are significant from a historical perspective. They either have been superseded by later specifications or have never passed the necessary maturity levels to become an Internet standard.

Experimental: An RFC classified as experimental describes work related to an experimental situation that does not affect the operation of the Internet. Such an RFC should not be implemented in any functional Internet service.

Informational: An RFC classified as informational contains general, historical, or tutorial information related to the Internet. It is usually written by someone in a non-Internet organization, such as a vendor.

REQUIREMENT LEVELS

RFCs are classified into five *requirement levels:* required, recommended, elective, limited use, and not recommended.

Required An RFC is labeled *required* if it must be implemented by all Internet systems to achieve minimum conformance. For example, IP and are required protocols.

Recommended An RFC labeled recommended is not required for minimum conformance; it is recommended because of its usefulness. For example, FTP and TELNET are recommended protocols.

Elective An RFC labeled elective is not required and not recommended. However, a system can use it for its own benefit.

Limited Use An RFC labeled limited use should be used only in limited situations. Most of the experimental RFCs fall under this category.

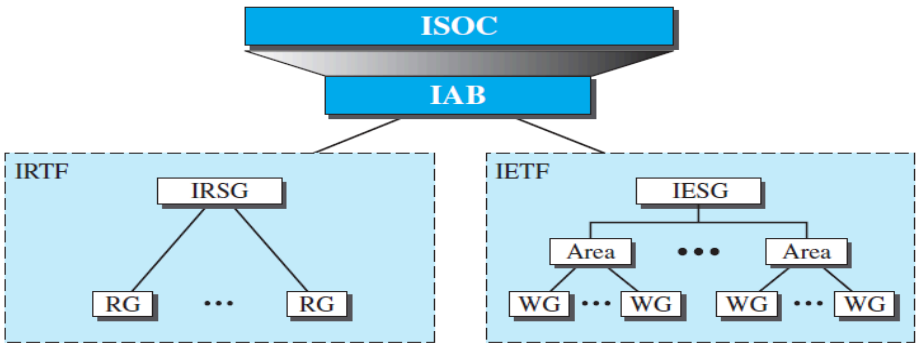
Not Recommended. An RFC labeled not recommended is inappropriate for general use. Normally a historic (deprecated) RFC may fall under this category.

INTERNET ADMINISTRATION

INTERNET SOCIETY (ISOC)

The **Internet Society (ISOC)** is an international, nonprofit organization formed in 1992 to provide support for the Internet standards process. ISOC accomplishes this through maintaining and supporting other Internet administrative bodies such as IAB, IETF, IRTF, and IANA. ISOC also promotes research and other scholarly activities relating to the Internet.

Figure 1.17 *Internet administration*



INTERNET ARCHITECTURE BOARD (IAB)

The **Internet Architecture Board (IAB)** is the technical advisor to the ISOC. The main purposes of the IAB are to oversee the continuing development of the TCP/IP Protocol Suite and to serve in a technical advisory capacity to research members of the Internet community. IAB accomplishes this through its two primary components, the Internet Engineering Task Force (IETF) and the Internet Research Task Force (IRTF).

Another responsibility of the IAB is the editorial management of the RFCs, described earlier. IAB is also the external liaison between the Internet and other standards organizations and forums.

INTERNET ENGINEERING TASK FORCE (IETF)

The **Internet Engineering Task Force (IETF)** is a forum of working groups managed by the Internet Engineering Steering Group (IESG). IETF is responsible for identifying operational problems and proposing solutions to these problems. IETF also develops and reviews specifications intended as Internet standards. The working groups are collected into areas, and each area concentrates on a specific topic. Currently nine areas have been defined. The areas include applications, protocols, routing, network management next generation (IPng), and security.

INTERNET RESEARCH TASK FORCE (IRTF)

The **Internet Research Task Force (IRTF)** is a forum of working groups managed by the Internet Research Steering Group (IRSG). IRTF focuses on long-term research topics related to Internet protocols, applications, architecture, and technology.

TRANSMISSION MEDIA

Transmission media are actually located below the physical layer and are directly controlled by the physical layer. We could say that *transmission media belong to layer zero*.

A **transmission medium** can be broadly defined as anything that can carry information from a source to a destination.

Transmission media can be divided into two broad categories: GUIDED AND UNGUIDED.

Guided media include twisted-pair cable, coaxial cable, and fiber-optic cable. Unguided medium is free space.

The data transmission capabilities of various Medias vary differently depending upon the various factors. These factors are:

1. **Bandwidth**. It refers to the data carrying capacity of a channel or medium. Higher bandwidth communication channels support higher data rates.
2. **Radiation**. It refers to the leakage of signal from the medium due to undesirable electrical characteristics of the medium.
3. **Noise Absorption**. It refers to the susceptibility of the media to external electrical noise that can cause distortion of data signal.
4. **Attenuation**. It refers to loss of energy as signal propagates outwards. The amount of energy lost depends on frequency. Radiations and physical characteristics of media contribute to attenuation.

Factors to be considered while selecting a Transmission Medium

1. Transmission Rate
2. Cost and Ease of Installation
3. Resistance to Environmental Conditions
4. Distances

WIRED (OR) GUIDED MEDIA (OR) BOUND TRANSMISSION MEDIA

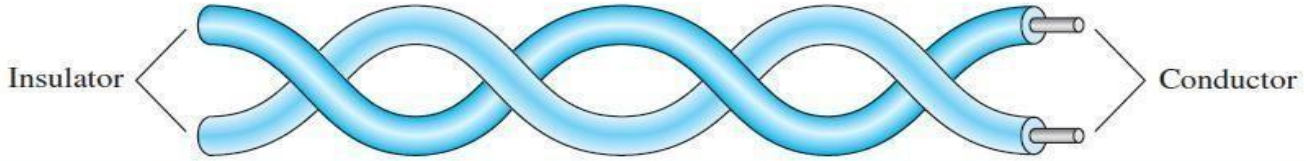
Guided media, which are those that provide a conduit from one device to another, Include **twisted-pair cable, coaxial cable,** and **fiber-optic cable.**

Twisted-pair and coaxial cable use metallic (copper) conductors that accept and transport signals in the form of electric current. **Optical fiber** is a cable that accepts and transports signals in the form of light.

1. TWISTED-PAIR CABLE

A twisted pair consists of two conductors (normally copper), each with its own plastic insulation, twisted together, as shown in Figure 7.3.

Figure 7.3 *Twisted-pair cable*



One of the wires is used to carry signals to the receiver, and the other is used only as a ground reference. The receiver uses the difference between the two. In addition to the signal sent by the sender on one of the wires, interference (noise) and crosstalk may affect both wires and create unwanted signals.

In addition to the signal sent by the sender on one of the wires, interference (noise) and crosstalk may affect both wires and create unwanted signals.

If the two wires are parallel, the effect of these unwanted signals is not the same in both wires because they are at different locations relative to the noise or crosstalk sources (e.g., one is closer and the other is farther). This results in a difference at the receiver. By twisting the pairs, a balance is maintained.

Why to twist the wires?

• Twisting of wires will reduce the effect of noise or external interference.

• Number of twists per unit length will determine the quality of cable. More twists means better quality.

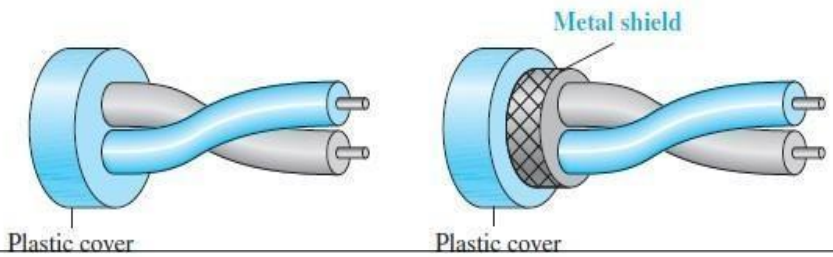
Unshielded Versus Shielded Twisted-Pair Cable

The most common twisted-pair cable used in communications is referred to as ***unshielded twisted-pair* (UTP).** IBM has also produced a version of twisted-pair cable for its use, called ***shielded twisted-pair* (STP).**

STP cable has a metal foil or braided mesh covering that encases each pair of insulated conductors. Although metal casing improves the quality of cable by preventing the penetration of noise or crosstalk, it is bulkier and more expensive.

Figure 7.4 shows the difference between UTP and STP.

Figure 7.4 *UTP and STP cables*



Advantage of STP over UTP

STP is less susceptible to noise as compared to UTP and therefore reduces the cross talk and interference.

Disadvantages of STP

1. It must be properly grounded.
2. It is more expensive than UTP.
3. It is difficult to terminate.

Advantages of Twisted pair cable

1. It can be used to carry both analog and digital data.
2. It is relatively easy to implement and terminate.
3. It is the least expensive media of transmission for short distances.
4. If portion of a twisted pair cable is damaged it does not affect the entire network.

Disadvantages of Twisted pair cable

1. It offers poor noise immunity as a result signal distortion is more?
2. Attenuation is very high.
3. It supports lower bandwidth as compared to other Medias. It supports 10 mbps upto a distance of 100 meters on a 10BASE-T.
4. It offers very poor security and is relatively easy to tap.
5. Being thin in size, they are likely to break easily.

CATEGORIES

The Electronic Industries Association (EIA) has developed standards to classify unshielded twisted-pair cable into seven categories. Categories are determined by cable quality, with 1 as the lowest and 7 as the highest. Each EIA category is suitable for specific uses.

Below Table shows these categories.

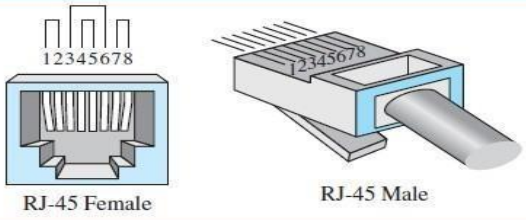
Category	Specification	Data Rate(Mbps)	Use
1	Unshielded twisted-pair used in telephone	<0.1	Telephone
2	Unshielded twisted-pair originally used in T lines	2	T-lines
3	Improved CAT 2 used in LANs	10	LANs
4	Improved CAT 3 used in Token Ring networks	20	LANs
5	Cable wire is normally 24 AWG with a jacket and outside sheath	100	LANs
5E	An extension to category 5 that includes extra features to minimize the crosstalk	125	LANs

	and electromagnetic interference		
6	A new category with matched Components coming from the same manufacturer. The cable must be tested at a 200-Mbps data rate.	200	LANs
7	Sometimes called *SSTP (shielded screen Twisted-pair)*. Each pair is individually Wrapped in a helical metallic foil ollowed by a metallic foil shield in addition to the outside sheath. The shield decreases the effect of crosstalk and increases the data rate.	600	LANs

Connectors

The most common UTP connector is **RJ45** (RJ stands for registered jack), as shown in Figure 7.5. The RJ45 is a keyed connector, meaning the connector can be inserted in only one way.

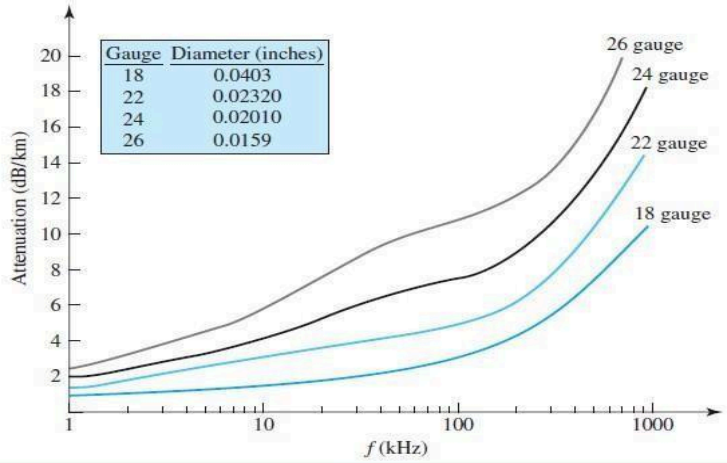
UTP connector



Performance

One way to measure the performance of twisted-pair cable is to compare attenuation versus frequency and distance. With increasing frequency, the attenuation, measured in decibels per kilometer (dB/km), sharply increases with frequencies above 100 kHz.(***Gauge*** is a measure of the thickness of the wire.)

Figure 7.6 *UTP performance*



Applications

Some of the applications of twisted pair cables are as follows:

(1) In telephone lines to carry voice and data channels.

(2) In the local loop.

(3) In the DSL line (ADSL)

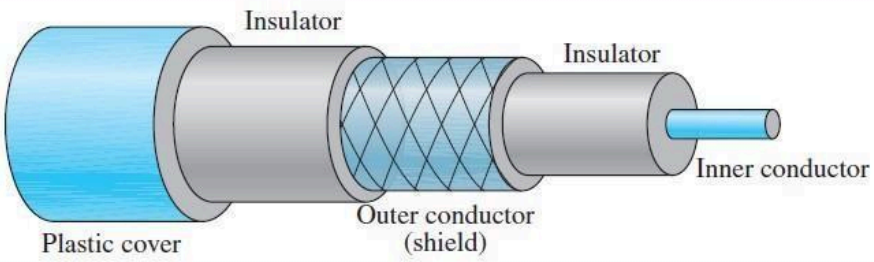
(4) Local area networks such as 10 Base-T and 100 Base-T. Use the twisted pair cables.

(5) In the ISDN (Integrated Services Digital Network).

2. COAXIAL CABLE

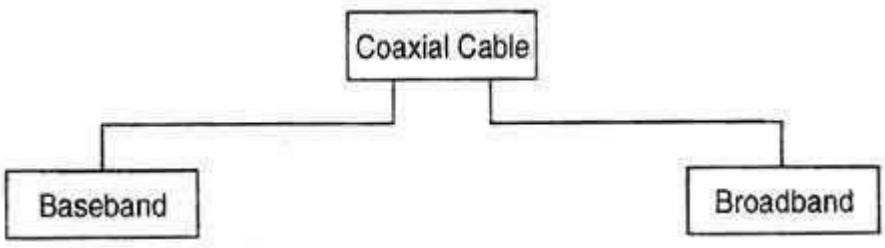
Coaxial cable (or *coax*) carries signals of higher frequency ranges than those in twisted pair cable, Instead of having two wires, coax has a central core conductor of solid or stranded wire (usually copper) enclosed in an insulating sheath, which is, in turn, encased in an outer conductor of metal foil, braid, or a combination of the two. The outer metallic wrapping serves both as a shield against noise and as the second conductor, which completes the circuit. This outer conductor is also enclosed in an insulating sheath, and the whole cable is protected by a plastic cover (see Figure 7.7).

Figure 7.7 *Coaxial cable*



Types of Coaxial Cables

There are two types of coaxial cables:



Baseband Coaxial Cable

A baseband coaxial cable transmits a single signal at a time at very high speed. A baseband cable is mainly used for **LANs**.

Baseband coaxial cables support frequency range of a 4 kHz and are used for digital signaling.

Baseband coaxial cables are 50 ohm cables used for 'digital transmission'.

Broadband Coaxial Cable

A broadband coaxial cable can transmit many simultaneous signals using different frequencies.

Broadband coaxial cables support the frequency range above 4 kHz and are used for analog signals. So it must be used with a modem.
Broadband coaxial cables are 75 ohm cables used for analog transmission.

Coaxial Cable Standards

Coaxial cables are categorized by their **Radio Government (RG)** ratings. Each RG number denotes a unique set of physical specifications, including the wire gauge of the inner conductor, the thickness and type of the inner insulator, the construction of the shield, and the size and type of the outer casing.

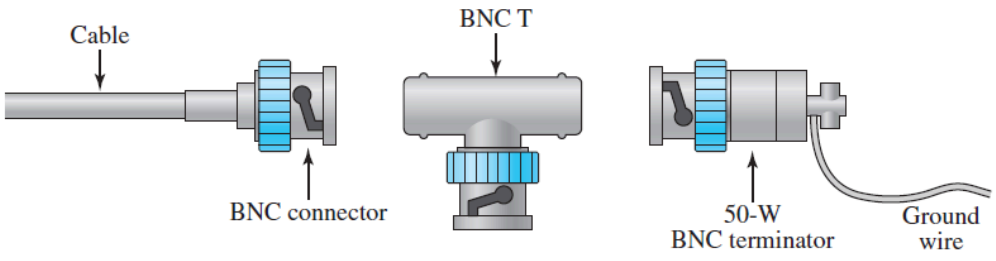
Table 7.2 *Categories of coaxial cables*

Category	*Impedance*	*Use*
RG-59	75 Ω	Cable TV
RG-58	50 Ω	Thin Ethernet
RG-11	50 Ω	Thick Ethernet

Coaxial Cable Connectors

To connect coaxial cable to devices, we need coaxial connectors. The most common type of connector used today is the **Bayonet Neill-Concelman (BNC)** connector. Figure 7.8 shows three popular types of these connectors: the BNC connector, the BNC T connector, and the BNC terminator.

Figure 7.8 *BNC connectors*



- The BNC connector is used to connect the end of the cable to a device, such as a TV set.
- The BNC T connector is used in Ethernet networks to branch out to a connection to a computer or other device.
- The BNC terminator is used at the end of the cable to prevent the reflection of the signal.

Features: It provides better immunity than twisted pair. This cable is able to transmit data at higher rates. **Limitations:** High installation cost. High maintenance cost.

Advantages of Coaxial Cables

1. It can be used for both analog and digital transmission.
2. It offers higher bandwidth as compared to twisted pair cable and can span longer distances.
3. Because of better shielding in coaxial cable, loss of signal or attenuation is less.
4. Better shielding also offers good noise immunity.
5. It is relatively inexpensive as compared to optical fibers.
6. It has lower error rates as compared to twisted pair.
7. It is not as easy to tap as twisted pair because copper wire is contained in plastic jacket.

Disadvantage of Coaxial Cables

It is usually more expensive than twisted pair.

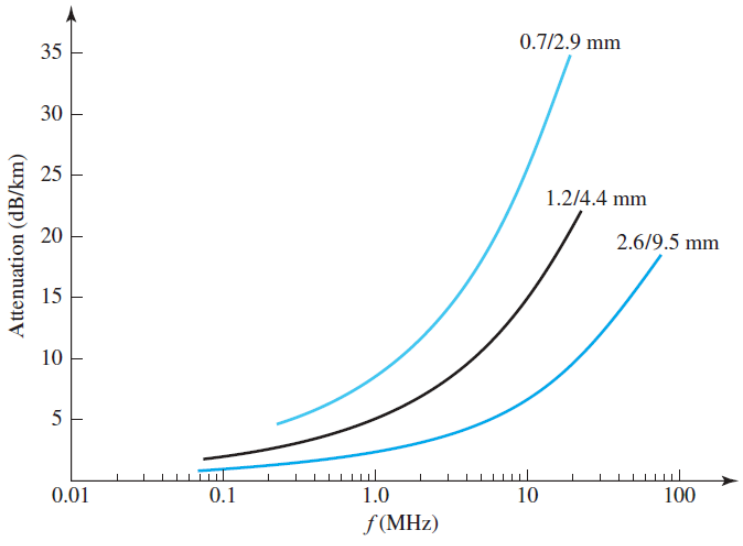
Applications of Co-axial Cables

(1) Analog telephone networks.
(2) Digital telephone network.
(3) Cable TV
(4) Traditional Ethernet LANs
(5) Digital transmission
(6) Thick Ethernet

Performance

We notice in Figure 7.9 that the attenuation is much higher in coaxial cable than in twisted-pair cable. In other words, although coaxial cable has a much higher bandwidth, the signal weakens rapidly and requires the frequent use of repeaters.

Figure 7.9 *Coaxial cable performance*



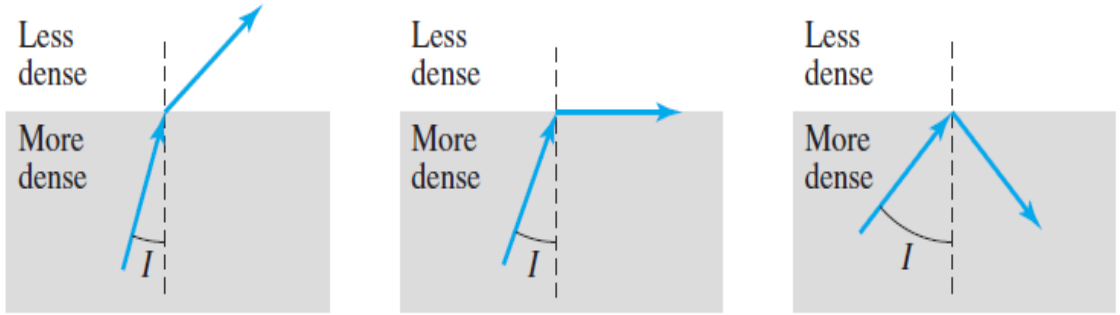
3. FIBER-OPTIC CABLE

A fiber-optic cable is made of glass or plastic and transmits signals in the form of light.

Light travels in a straight line as long as it is moving through a single uniform substance. If a ray of light traveling through one substance suddenly enters another substance (of a different density), the ray changes direction. Figure 7.10 shows how a ray of light changes direction when going from a more dense to a less dense substance.

As the figure shows, if the **angle of incidence** *I* (the angle the ray makes with the line perpendicular to the interface between the two substances) is less than the **critical angle,** the ray **refracts** and moves closer to
the surface. If the angle of incidence is equal to the critical angle, the light bends along the interface. If the angle is greater than the critical angle, the ray **reflects** (makes a turn) and travels again in the denser substance.

Figure 7.10 *Bending of light ray*



I < critical angle, refraction | *I* = critical angle, refraction | *I* > critical angle, reflection

Optical fibers use reflection to guide light through a channel. A glass or plastic **core** is surrounded by a **cladding** of less dense glass or plastic. The difference in density of the two materials must be such that a beam of light moving through the core is reflected off the cladding instead of being refracted into it.

Figure 7.11 *Optical fiber*

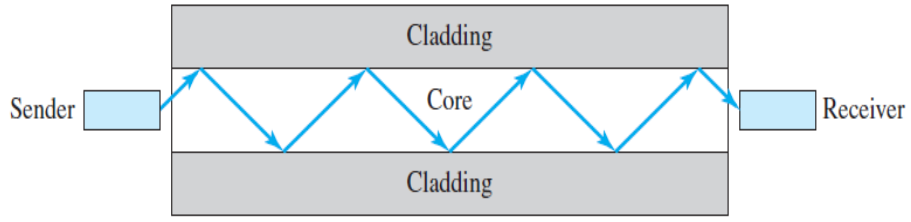
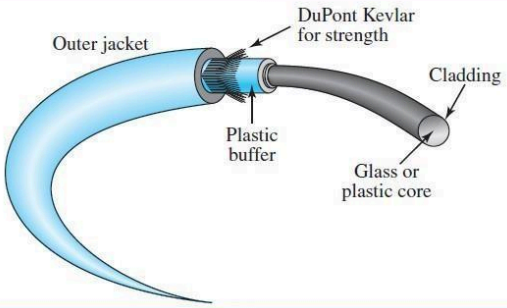


Figure 7.14 *Fiber construction*



Propagation Modes (Types of optical fibers)

Current technology supports two modes (multimode and single mode) for propagating light along optical channels, each requiring fiber with different physical characteristics. Multimode can be implemented in two forms: step-index or graded-index.

Figure 7.12 *Propagation modes*

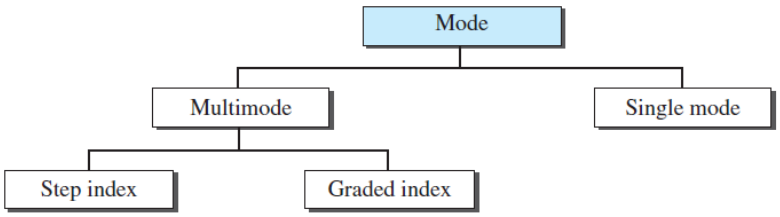


Table 7.3 *Fiber types*

Type	*Core (μm)*	*Cladding (μm)*	*Mode*
50/125	50.0	125	Multimode, graded index
62.5/125	62.5	125	Multimode, graded index
100/125	100.0	125	Multimode, graded index
7/125	7.0	125	Single mode

Advantages and Disadvantages of Optical Fiber

Advantages

Fiber-optic cable has several advantages over metallic cable (twisted-pair or coaxial).

Higher bandwidth Fiber-optic cable can support dramatically higher bandwidths (and hence data rates) than either twisted-pair or coaxial cable. Currently, data rates and bandwidth utilization over fiber-optic cable are limited not by the medium but by the signal generation and reception technology available.

Less signal attenuation Fiber-optic transmission distance is significantly greater than that of other guided media. A signal can run for 50 km without requiring regeneration. We need repeaters every 5 km for coaxial or twisted-pair cable.

Immunity to electromagnetic interference Electromagnetic noise cannot affect fiber-optic cables.

Resistance to corrosive materials Glass is more resistant to corrosive materials than copper.

Light weight. Fiber-optic cables are much lighter than copper cables.

Greater immunity to tapping Fiber-optic cables are more immune to tapping than copper cables. Copper cables create antenna effects that can easily be tapped.

Disadvantages

There are some disadvantages in the use of optical fiber.

Installation and maintenance Fiber-optic cable is a relatively new technology. Its installation and maintenance require expertise that is not yet available everywhere.

Unidirectional light propagation Propagation of light is unidirectional. If we need bidirectional communication, two fibers are needed.

Cost The cable and the interfaces are relatively more expensive than those of other guided media. If the demand for bandwidth is not high, often the use of optical fiber cannot be justified.

Characteristics of Optical Fiber Cables:

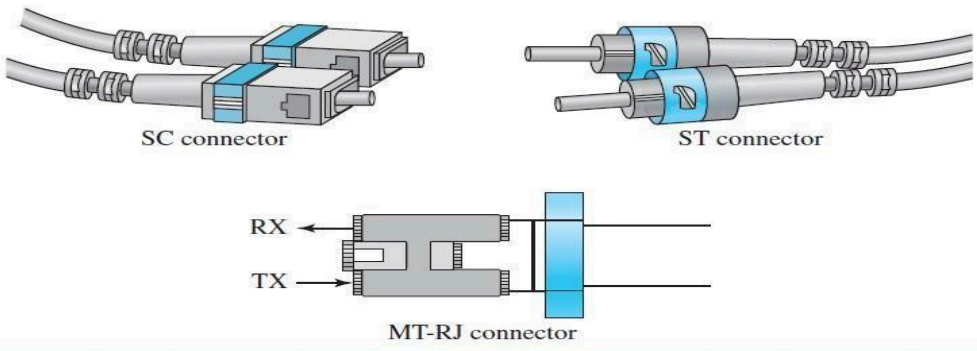
Fiber optic cables have the following characteristics:

1. Fiber optic cabling can provide extremely high bandwidths in the range from 100 mbps to 2 gigabits because light has a much higher frequency than electricity.
2. The number of nodes which a fiber optic can support does not depend on its length but on the hub or hubs that connect cables together.
3. Fiber optic cable has much lower attenuation and can carry signal to longer distances without using amplifiers and repeaters in between.
4. Fiber optic cable is not affected by EMI effects and can be used in areas where high voltages are passing by.
5. The cost of fiber optic cable is more compared to twisted pair and co-axial.
6. The installation of fiber optic cables is difficult and tedious.

Fiber-Optic Cable Connectors

There are three types of connectors for fiber-optic cables, as shown in Figure 7.15.

Figure 7.15 *Fiber-optic cable connectors*



Subscriber channel (SC) connector is used for cable TV. It uses a push/pull locking system.

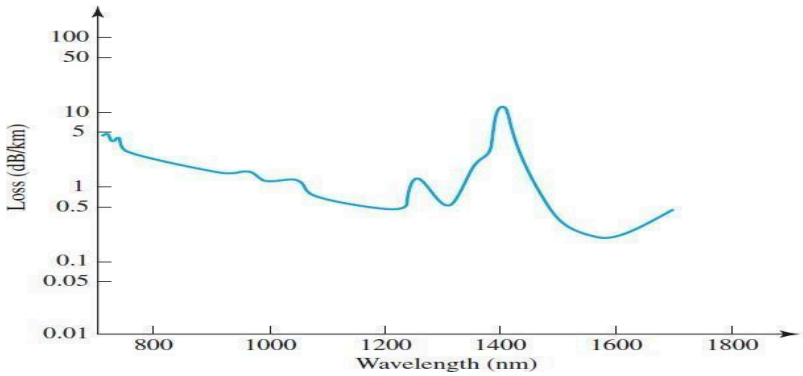
Straight-tip (ST) connector is used for connecting cable to networking devices. It uses a bayonet locking system and is more reliable than SC.

MT-RJ is a connector that is the same size as RJ45.

Performance

The plot of attenuation versus wavelength in Figure 7.16 shows a very interesting phenomenon in fiber-optic cable. Attenuation is flatter than in the case of twisted-pair cable and coaxial cable. The performance is such that we need fewer (actually one tenth as many) repeaters when we use fiber-optic cable.

Optical fiber performance



Applications:

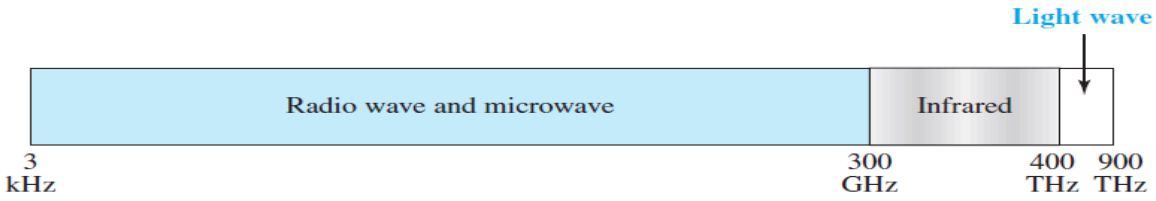
1. Optical fiber transmission systems are widely used in th~ backbone of networks. Current optical fiber systems provide transmission rates from 45 Mb/s to 9.6 Gb/s using the single wavelength transmission.
2. The installation cost of optical fibers is higher than that for the co-axial or twisted wire cables.
3. Optical fibers are now used in the telephone systems.
4. in the Local Area Networks (LANs).

UNGUIDED MEDIA: WIRELESS

Unguided medium transport electromagnetic waves without using a physical conductor. This type of communication is often referred to as ***wireless communication*.** Signals are normally broadcast through free space and thus are available to anyone who has a device capable of receiving them.

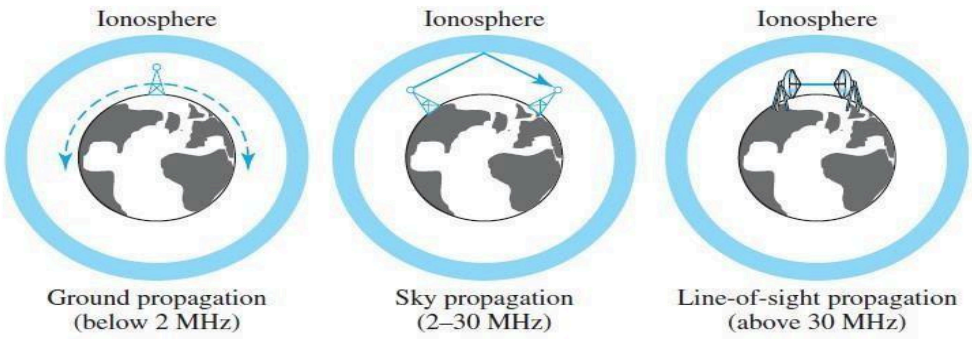
Figure 7.17 shows the part of the electromagnetic spectrum, ranging from 3 kHz to 900 THz, used for wireless communication.

Figure 7.17 *Electromagnetic spectrum for wireless communication*



Unguided signals can travel from the source to the destination in several ways: ground propagation, sky propagation, and line-of-sight propagation, as shown in Figure 7.18.

Figure 7.18 *Propagation methods*



Ground propagation: radio waves travel through the lowest portion of the atmosphere, hugging the earth. These low-frequency signals emanate in all directions from the transmitting antenna and follow the curvature of the planet. Distance depends on the amount of power in the signal: The greater the power, the greater the distance.

Sky propagation: higher-frequency radio waves radiate upward into the ionosphere (the layer of atmosphere where particles exist as ions) where they are reflected back to earth. This type of transmission allows for greater distances with lower output power.

Line-of-sight propagation: very high-frequency signals are transmitted in straight lines directly from antenna to antenna.

1. RADIO WAVES

Electromagnetic waves ranging in frequencies between 3 kHz and 1 GHz are normally called **radio waves.**

Radio waves are Omni directional. When an antenna transmits radio waves, they are propagated in all directions. This means that the sending and receiving antennas do not have to be aligned. A sending antenna sends waves that can be received by any receiving antenna.

Radio waves, particularly those waves that propagate in the sky mode, can travel long distances. This makes radio waves a good candidate for long-distance broadcasting such as AM radio.

Omni directional Antenna

Radio waves use **Omni directional antennas** that send out signals in all directions. Based on the wavelength, strength, and the purpose of transmission, we can have several types of antennas. Figure 7.19 shows an Omni directional antenna.

Figure 7.19 *Omnidirectional antenna*

Disadvantage: The Omni directional property has a disadvantage, too. The radio waves transmitted by one antenna are susceptible to interference by another antenna that may send signals using the same frequency or band.

Applications of Radio Waves:

Radio waves are used for multicast communications, such as radio and television, and paging systems

2. MICROWAVES

Electromagnetic waves having frequencies between 1 and 300 GHz are called microwaves.

Microwaves are unidirectional. When an antenna transmits microwaves, they can be narrowly focused. This means that the sending and receiving antennas need to be aligned

Characteristics of microwave propagation:

Microwave propagation is line-of-sight. Since the towers with the mounted antennas need to be in direct sight of each other, towers that are far apart need to be very tall. The curvatures of the earth as well as other blocking obstacles do not allow two short towers to communicate by using microwaves. Repeaters are often needed for long-distance communication.

Very high-frequency microwaves cannot penetrate walls. This characteristic can be a disadvantage if receivers are inside buildings.

The microwave band is relatively wide, almost 299 GHz. Therefore wider sub bands can be assigned, and a high data rate is possible.

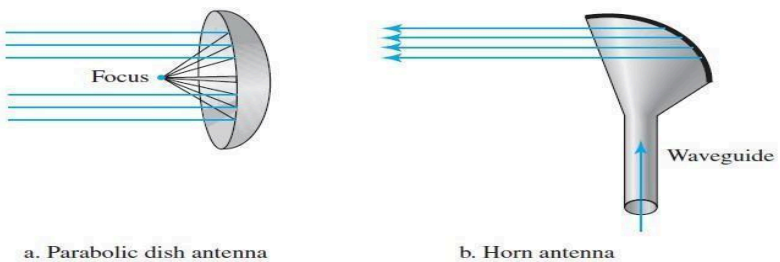
Use of certain portions of the band requires permission from authorities.

Unidirectional Antenna

Microwaves need **unidirectional antennas** that send out signals in one direction. Two types of antennas are used for microwave communications: the parabolic dish and the horn

Applications Microwaves are used for unicast communication such as cellular telephones, satellite networks, and wireless LANs.

Figure 7.20 *Unidirectional antennas*



3. INFRARED

Infrared waves, with frequencies from 300 GHz to 400 THz (wavelengths from 1 mm to 770 nm), can be used for short-range communication.

Infrared waves, having high frequencies, cannot penetrate walls. This advantageous characteristic prevents interference between one system and another.

Applications Infrared signals can be used for short-range communication in a closed area using line-of-sight propagation.

DATA LINK LAYER - DESIGN ISSUES, CRC CODES, ELEMENTARY DATA LINK LAYER PROTOCALS, SLIDING WINDOW PROTOCAL.

DATA LINK LAYER DESIGN ISSUES

The data link layer uses the services of the physical layer to send and receive bits over communication channels.

It has a number of functions, including:

1. Providing a well-defined service interface to the network layer.
2. Dealing with transmission errors.
3. Regulating the flow of data so that slow receivers are not swamped by fast senders.

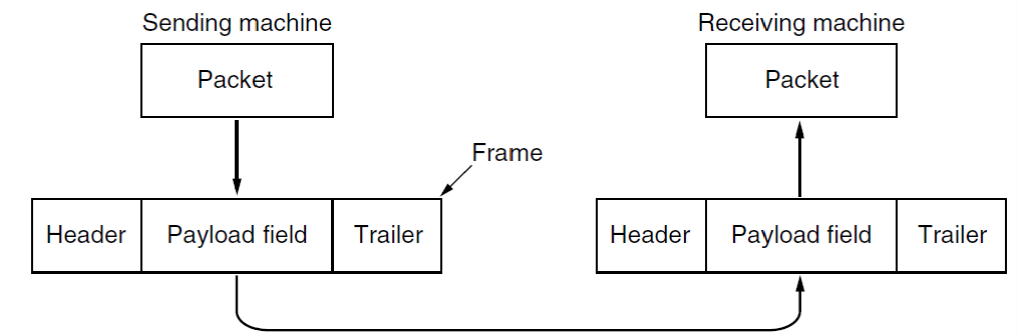


Figure 3-1. Relationship between packets and frames.

To accomplish these goals, the data link layer takes the packets it gets from the network layer and encapsulates them into **frames** for transmission. Each frame contains a frame header, a payload field for holding the packet, and a frame trailer, as illustrated in Fig. 3-1.

SERVICES PROVIDED TO THE NETWORK LAYER

The function of the data link layer is to provide services to the network layer. The principal service is transferring data from the network layer on the source machine to the network layer on the destination machine.

On the source machine is an entity, call it a process, in the network layer that hands some bits to the data link layer for transmission to the destination. The job of the data link layer is to transmit the bits to the destination machine so they can be handed over to the network layer as shown in Fig.

The data link layer can be designed to offer various services vary from protocol to protocol as mentioned below.

1. Unacknowledged connectionless service.
2. Acknowledged connectionless service.
3. Acknowledged connection-oriented service.

The actual transmission follows the path of Fig. 3-2(b), but it is easier to think in terms of two data link layer processes communicating using a data link protocol (Fig. 3-2(a)).

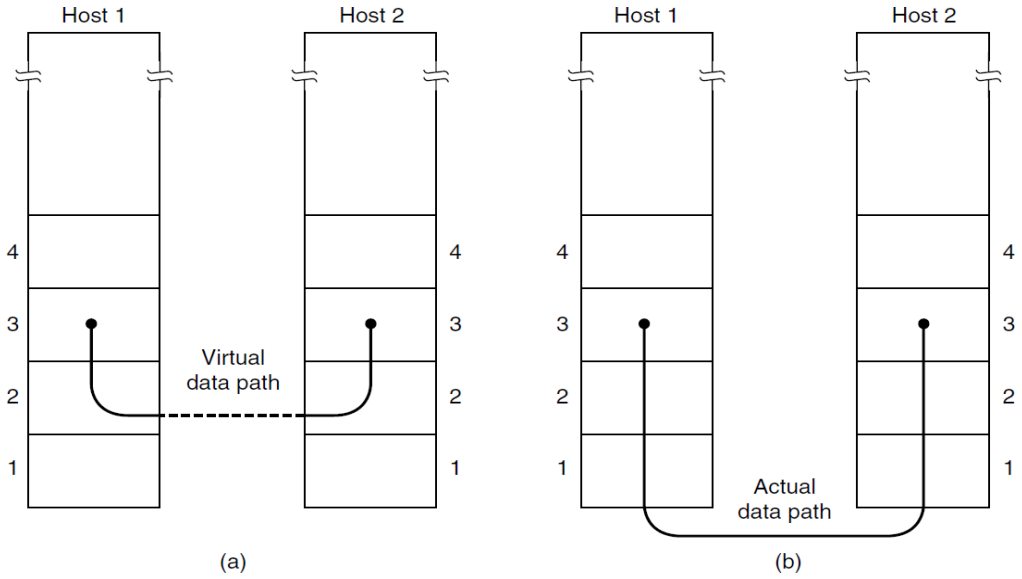


Figure 3-2. (a) Virtual communication. (b) Actual communication.

1. UNACKNOWLEDGED CONNECTIONLESS SERVICE

No logical connection is established beforehand or released afterward. Source machine send independent frames to the destination machine without having the destination machine acknowledge them.

If a frame is lost due to noise on the line, no attempt is made to detect the loss or recover from it in the data link layer.

This class of service is appropriate when the error rate is very low, so recovery is left to higher layers. It is also appropriate for real-time traffic, such as voice, in which late data are worse than bad data.

Example: Ethernet

2. ACKNOWLEDGED CONNECTIONLESS SERVICE

Acknowledged connectionless service there are still no logical connections used, but each frame sent is individually acknowledged. In this way, the sender knows whether a frame has arrived correctly or been lost.

If frame has not arrived within a specified time interval, it can be sent again.

This service is useful over unreliable channels, such as wireless systems.

Example: 802.11 (Wi-Fi)

The network layer can always send a packet and wait for it to be acknowledged by its peer on the remote machine. If the acknowledgement is not forthcoming before the timer expires, the sender can just send the entire message again.

3. ACKNOWLEDGED CONNECTION-ORIENTED SERVICE

This is most sophisticated service the data link layer can provide to the network layer.

The source and destination machines establish a connection before any data are transferred.

Each frame sent over the connection is numbered.

Data link layer guarantees that each frame sent is indeed received and each frame is received exactly once and all frames are received in the right order.

It is appropriate over long, unreliable links such as a satellite channel or long-distance telephone

circuits. In Connection oriented service the data transfer go through three distinct phases.

Phase 1: Connection Establishment (both sender and receiver initialize variables and counters needed to keep track of which frames have been received and which one have not).

Phase 2: The Frames are transmitted.

Phase 3: The Connection release (freeing up the variables, buffers, and other resources used to maintain the connection).

FRAMING

To provide service to the network layer, the data link layer must use the service provided by the physical layer.

Data transmission in the physical layer means moving bits in the form of a signal from the source to the destination. The data-link layer, on the other hand, needs to pack bits into frames, so that each frame is distinguishable from another.

Framing in the data-link layer separates a message from one source to a destination by adding a sender address and a destination address. The destination address defines where the packet is to go; the sender address helps the recipient acknowledge the receipt.

Example of framing: postal system

TYPES OF FRAMES: Frames can be of fixed or variable size.

FIXED-SIZE FRAMING: there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter.

Example: ATM WAN (which uses frames of fixed size called *cells*)

VARIABLE-SIZE FRAMING: In variable-size framing, we need a way to define the end of one frame and the beginning of the next.

Example: Local-area networks

Two approaches were used for this purpose:

Character-oriented approach (Character/ Byte Stuffing), Bit-oriented approach (Bit Stuffing)

The physical layer accepts raw bit stream and attempt to deliver it the destination. If the channel is noisy, the bit received by data link layer is not guaranteed to be error free. Some bits may have different values and

number of bits received may be in correct. It is up to the data link layer to detect and correct the errors.

The data link layer break up the bit stream into discrete frames, compute a short token called a checksum for each frame, and include the checksum in the frame when it is transmitted. When a frame arrives at the destination, the checksum is recomputed. If the newly computed checksum is different from the one contained in the frame, the data link layer knows that an error has occurred and takes steps to deal with it. (e.g., discarding the bad frame and possibly sending back the error report).

The following methods were used by data link layer to form the frames.

1. Byte Count
2. Flag bytes with byte stuffing
3. Flag bits with bit stuffing
4. Physical layer coding violations.

1. BYTE COUNT

In this method the first field in the header to specify the number of bytes in the frame. The data link layer at receiver sees the byte count; it knows how many bytes follow and where to end the frame.

The problem with this algorithm is if the count gets grabbed by a transmission error. Than receiver unable to locate the correct start of the next frame. It has no way of telling where the next frame starts. Sending a frame back to the source asking for a re-transmission does not help either, since the destination does not know how many bytes to skip over to get to the start of the re-transmission.

This method is used very rarely.

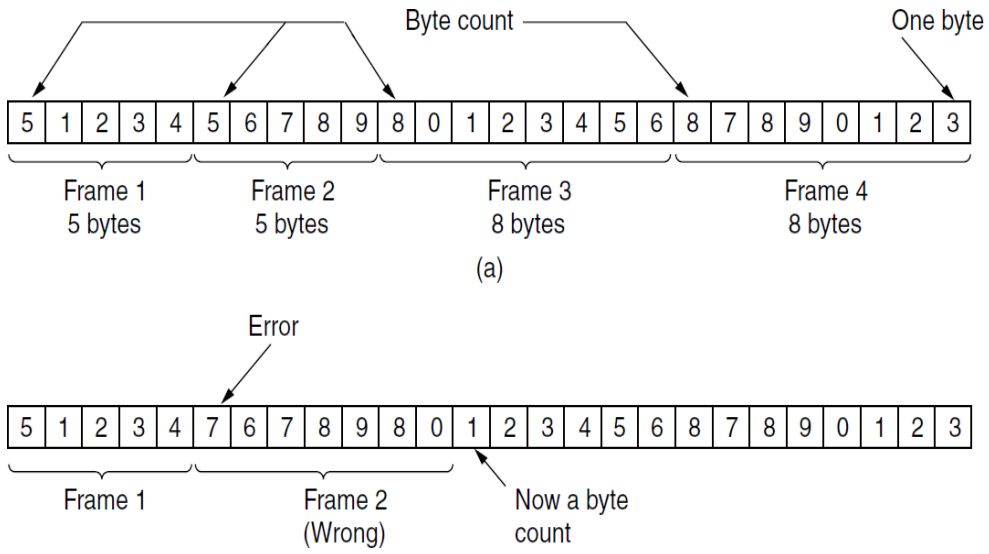


Figure 3-3. A byte stream. (a) Without errors. (b) With one error.

1. FLAG BYTES WITH BYTE STUFFING(CHARACTER STUFFING)

In this method each frame starts and ends with special bytes called a ***FLAG*** byte. Two consecutive ***FLAG*** bytes indicate the end of one frame and the start of the next frame. Thus if the receiver ever loses synchronization it can just search for two flag bytes to find the end of the current frame and the start of the next frame.

If **FLAG** byte occurs in the data, the sender's data link layer insert a special **ESCAPE BYTE (ESC)** just before each "accidental" flag byte in the data. The data link layer on the receiving end removes the escape byte before giving the data to the network layer. This technique is called **byte stuffing.**

If an escape byte occurs in the middle of the data, it is too stuffed with an escape byte. At the receiver, the first escape byte is removed, leaving the data byte that follows it.

EXAMPLE 1:

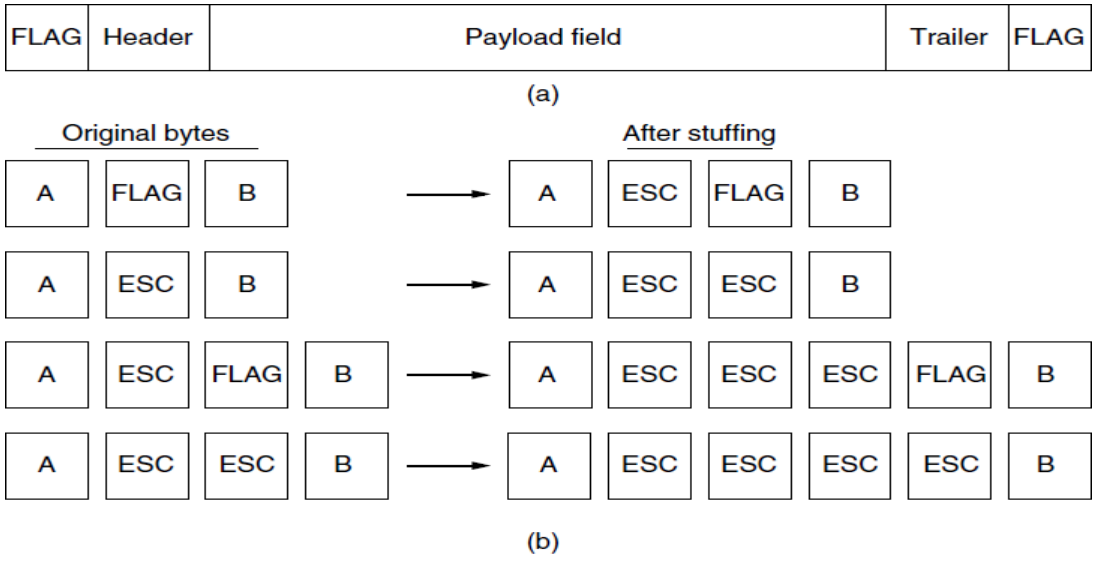


Figure 3-4. (a) A frame delimited by flag bytes. (b) Four examples of byte sequences before and after byte stuffing.

Byte stuffing is the process of adding one extra byte whenever there is a flag or escape character in the text.

EXAMPLE 2:

Figure 11.1 *A frame in a character-oriented protocol*

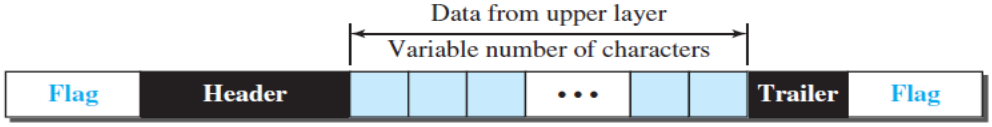
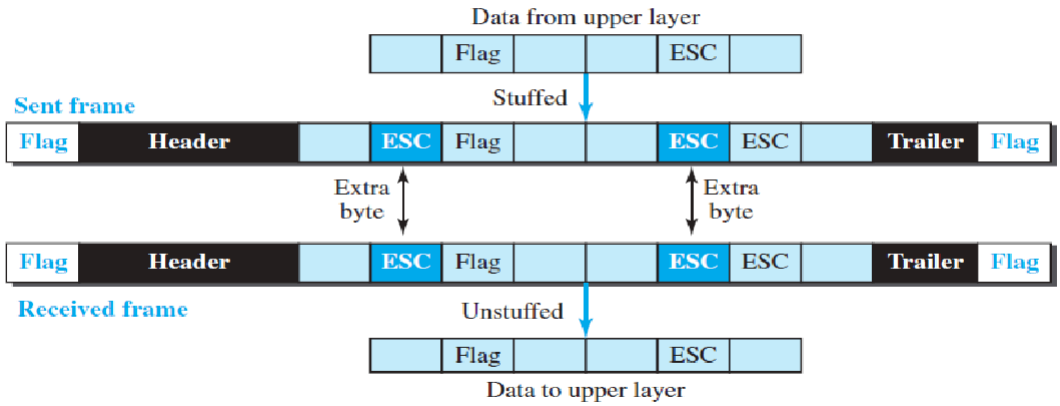


Figure 11.2 *Byte stuffing and unstuffing*



3. FLAG BITS WITH BIT STUFFING

In this algorithm each frame begins and ends with a special bit pattern (flag byte), **01111110** (0x7E in hexadecimal).

Whenever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a 0 bit into the outgoing bit stream to distinguish from flag byte. When the receiver sees five consecutive incoming 1 bits, followed by a 0 bit, it automatically destuffs (i.e., delete) the 0 bit.

If the user data contain the flag pattern, 01111110, this flag is transmitted as 011111010 but stored in the receiver's memory as 01111110.

Figure 3-5 gives an example of bit stuffing.

EXAMPLE 1:

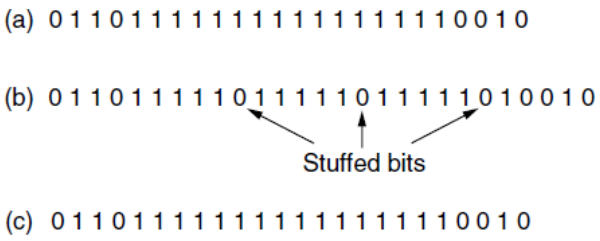


Figure 3-5. Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after destuffing.

EXAMPLE 2:

Figure 11.3 *A frame in a bit-oriented protocol*

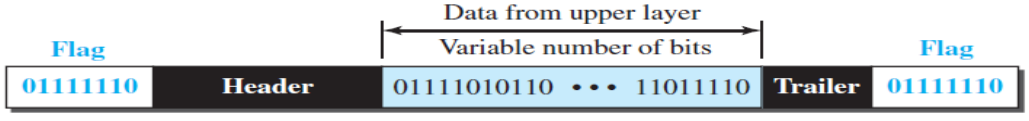
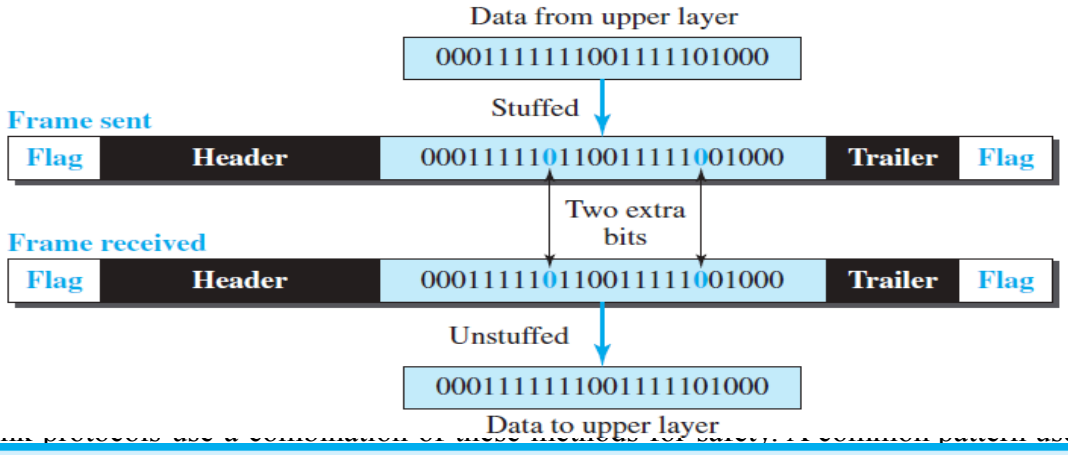


Figure 11.4 *Bit stuffing and unstuffing*



4. P

It i… …coding of bits as … …ignals will not occ… We … …using "coding vio… The … …rt and end of fra… Ma… …d for Ethernet … followed

Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag.

FLOW CONTROL

This is an important data link layer design issue where the sender is running on a fast powerful computer and receiver is running on slow, low-end machine. The receiver may be unable to handle the frames as fast as they are arriving and will lose some.

Two approaches are commonly used.

FEEDBACK-BASED FLOW CONTROL: the receiver sends back information to the sender giving it permission to send more data, or at least telling the sender how the receiver is doing

RATE-BASED FLOW CONTROL: the protocol has a built-in mechanism that limits the rate at which senders may transmit data, without using feedback from the receiver.

ERROR CONTROL

The bit stream transmitted by the physical layer is not guaranteed to be error free. The data link layer is responsible for error detection and correction.

Data link layer deals error control mechanism by following 3 ways.

Feedback based(using acknowledgement): data link layer protocol(at sender) calls the receiver to send back special control frames bearing positive or negative acknowledgements about the incoming frames. If the sender receives a positive acknowledgement about a frame, it knows the frame has arrived safely. On the other hand, a negative acknowledgement means that something has gone wrong and the frame must be transmitted again.

Limitation

Complication comes from the possibility that hardware troubles may cause a frame to vanish completely In this case, the receiver will not react at all, since it has no reason to react. Similarly, if the acknowledgement frame is lost, the sender will not know how to proceed.

Using timers: When the sender transmits a frame, it also starts a timer. The timer is set to expire after an interval long enough for the frame to reach the destination, be processed there, and have the acknowledgement propagate back to the sender. Normally, the frame will be correctly received and the acknowledgement will get back before the timer runs out, in which case the timer will be canceled.

Limitation

If either the frame or the acknowledgement is lost, the timer will go off, alerting the sender to a potential problem. The obvious solution is to just transmit the frame again. When frames transmitted multiple times there is a danger that the receiver will accept the same frame two or more times and pass it to the network layer more than once.

Assign sequence numbers to outgoing frames: Managing the timers and sequence numbers so as to ensure that each frame is ultimately passed to the network layer at the destination exactly once.

Error control at the data-link layer is normally very simple and implemented using one of the following two methods. In both methods, a CRC is added to the frame header by the sender and checked by the receiver.

1. In the first method, if the frame is corrupted, it is silently discarded; if it is not corrupted, the packet is delivered to the network layer. This method is used mostly in wired LANs such as Ethernet.
2. In the second method, if the frame is corrupted, it is silently discarded; if it is not corrupted, an acknowledgment is sent (for the purpose of both flow and error control) to the sender.

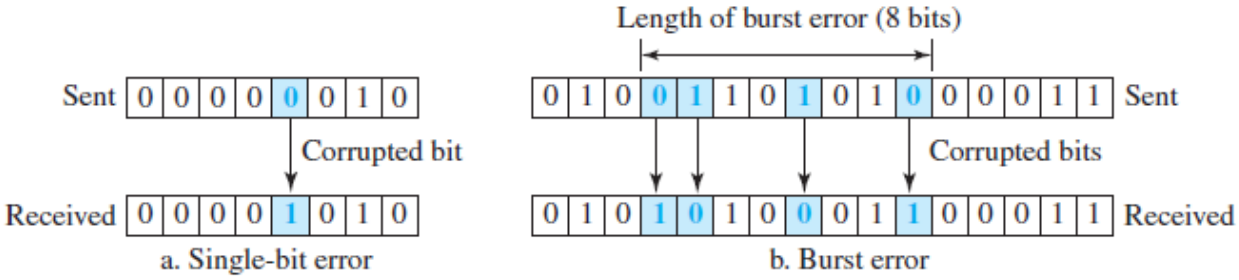
ERROR DETECTION AND CORRECTION

Types of Errors

Single-bit error: only 1 bit of a given data unit (such as a byte, character, or packet) is changed from 1 to 0 or from 0 to 1.

Burst error: 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1.

Figure 10.1 *Single-bit and burst error*



REDUNDANCY

The central concept in detecting or correcting errors is **redundancy**. To be able to detect or correct errors, we need to send some extra bits with our data. These redundant bits are added by the sender and removed by the receiver. Their presence allows the receiver to detect or correct corrupted bits.

ERROR DETECTION VERSUS CORRECTION

The correction of errors is more difficult than the detection.

ERROR DETECTION: we are only looking to see if any error has occurred. The answer is a simple yes or no. We are not even interested in the number of corrupted bits.

ERROR CORRECTION: we need to know the exact number of bits that are corrupted and, more importantly, their location in the message. The number of errors and the size of the message are important factors.

CODING

Redundancy is achieved through various coding schemes.

We can divide coding schemes into two broad categories: **block coding** and **convolution coding (It** is more complex.

BLOCK CODING

In block coding, we divide our message into blocks, each of k bits, called ***datawords*.**

We add r redundant bits to each block to make the length $n=k+r$. The resulting n-bit blocks are called

codewords.

With *k* bits, we can create a combination of 2^k datawords; with *n* bits, we can create a combination of 2^n codewords. Since $n > k$, the number of possible codewords is larger than the number of possible datawords.

Error Detection

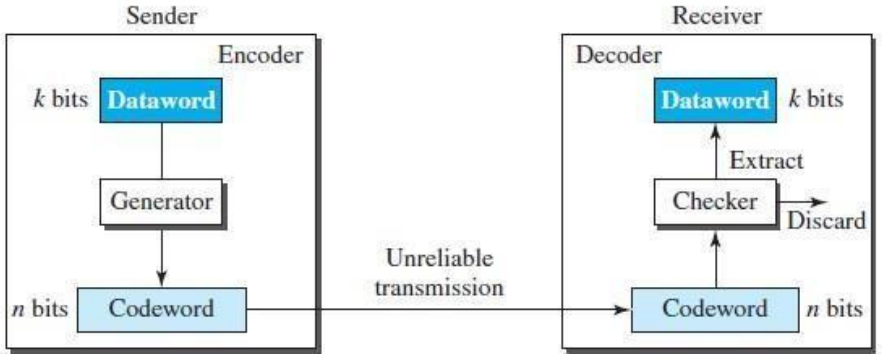
If the following two conditions are met, the receiver can detect a change in the original codeword.

1. The receiver has (or can find) a list of valid codewords.

2. The original codeword has changed to an invalid one.

The sender creates codeword's out of datawords by using a generator that applies the rules and procedures of encoding. Each codeword sent to the receiver may change during transmission. If the received codeword is the same as one of the valid codewords, the word is accepted; the corresponding dataword is extracted for use. If the received codeword is not valid, it is discarded. However, if the codeword is corrupted during transmission but the received word still matches a valid codeword, the error remains undetected.

Figure 10.2 *Process of error detection in block coding*



Example 10.1

Let us assume that *k* =2 and *n*=3. Table 10.1 shows the list of datawords and codewords.

Table 10.1 *A code for error detection in Example 10.1*

Dataword	*Codeword*	*Dataword*	*Codeword*
00	000	10	101
01	011	11	110

Assume the sender encodes the dataword 01 as 011 and sends it to the receiver. Consider the following cases:

1. The receiver receives 011. It is a valid codeword. The receiver extracts the dataword 01 from it.

2. The codeword is corrupted during transmission, and 111 is received (the leftmost bit is corrupted). This is not a valid codeword and is discarded.

3. The codeword is corrupted during transmission, and 000 is received (the right two bits are corrupted). This is a valid codeword. The receiver incorrectly extracts the dataword 00. Two corrupted bits have made the error undetectable.

An error-detecting code can detect only the types of errors for which it is designed; other types of errors may remain undetected.

HAMMING DISTANCE

One of the central concepts in coding for error control is the idea of the Hamming distance.

The **Hamming distance** between two words (of the same size) is the number of differences between the corresponding bits. We show the Hamming distance between two words x and y as $d(x, y)$.

Example: if the codeword 00000 is sent and 01101 is received, 3 bits are in error and the Hamming distance between the two is d (00000, 01101) =3. In other words, if the Hamming distance can easily be found if we apply the XOR operation on the two words and count the number of 1s in the result.

The Hamming distance between two words is the number of differences between corresponding bits.

Example 10.2

Let us find the Hamming distance between two pairs of words.

1. The Hamming distance d (000, 011) is 2 because (000 $\oplus$ 011) is 011 (two 1s).

2. The Hamming distance d (10101, 11110) is 3 because (10101 $\oplus$11110) is 01011 (three 1s).

To guarantee the detection of up to s errors in all cases, the minimum Hamming distance in a block code must be $d_{min} = s + 1$.

Almost all block codes used today belong to a subset of block codes called ***linear block codes***. The use of nonlinear block codes for error detection and correction is not as widespread because their structure makes theoretical analysis and implementation difficult.

Minimum Distance for Linear Block Codes

It is simple to find the minimum Hamming distance for a linear block code. The minimum Hamming distance is the number of 1s in the nonzero valid codeword with the smallest number of 1s.

PARITY-CHECK CODE

The most familiar error-detecting code is the **parity-check code.** It is a linear block code.

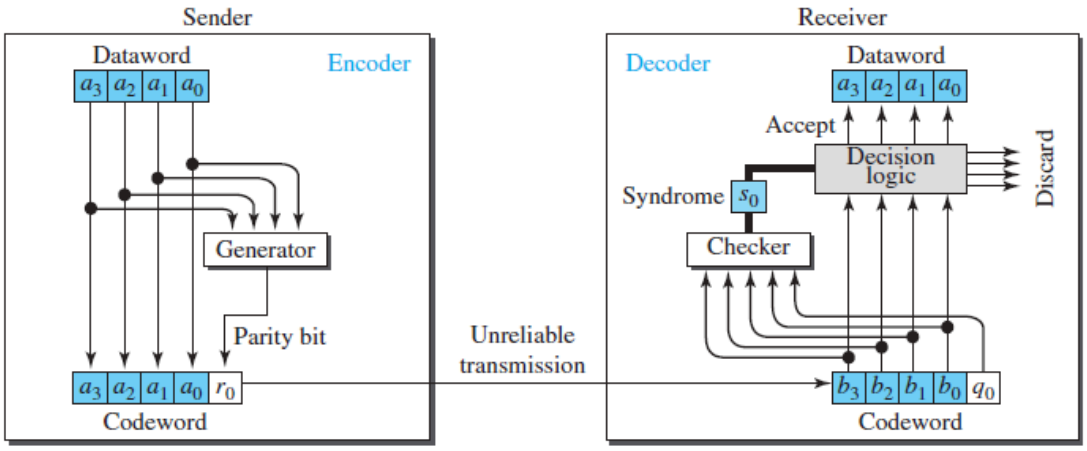
In parity-check code, a *k*-bit dataword is changed to an *n*-bit codeword where $n = k + 1$. The extra bit, called the *parity bit,* is selected to make the total number of 1s in the codeword even.

The encoder uses a generator that takes a copy of a 4-bit dataword ($a0$, $a1$, $a2$, and $a3$) and generates a parity bit $r0$. The dataword bits and the parity bit create the 5-bit codeword. The parity bit that is added makes the number of 1s in the codeword even. This is normally done by adding the 4 bits of the dataword (modulo-2); the result is the parity bit.

In other words, ***r0=a3+a2+a1+a0* (modulo-2)**

If the number of 1s is even, the result is 0; if the number of 1s is odd, the result is 1. In both cases, the total number of 1s in the codeword is even.

Figure 10.4 *Encoder and decoder for simple parity-check code*



The sender sends the codeword, which may be corrupted during transmission. The receiver receives a 5-bit word. The checker at the receiver does the same thing as the generator in the sender with one exception: The addition is done over all 5 bits. The result, which is called the ***syndrome*,** is just 1 bit. The syndrome is 0 when the number of 1s in the received codeword is even; otherwise, it is 1.

***s0=b3+b2+b1+b0* $+q0$ (modulo-2)**

The syndrome is passed to the decision logic analyzer. If the syndrome is 0, there is no detectable error in the received codeword; the data portion of the received codeword is accepted as the dataword; if the syndrome is 1, the data portion of the received codeword is discarded. The dataword is not created.

EXAMPLE (PARITY-CHECK CODE)

Assume the sender sends the dataword 1011. The codeword created from this dataword is 10111, which is sent to the receiver. We examine five cases:

1. No error occurs; the received codeword is 10111. The syndrome is 0. The dataword 1011 is created.

2. One single-bit error changes *a*1. The received codeword is 10**0**11. The syndrome is 1. No dataword is created.

3. One single-bit error changes *r*0. The received codeword is 1011**0**. The syndrome is 1. No dataword is created. Note that although none of the dataword bits are corrupted, no dataword is created because the code is not sophisticated enough to show the position of the corrupted bit.

4. An error changes *r*0 and a second error changes *a*3. The received codeword is **0**011**0**. The syndrome is 0. The dataword 0011 is created at the receiver. Note that here the dataword is wrongly created due to the syndrome value. The simple parity-check decoder cannot detect an even number of errors. The errors cancel each other out and give the syndrome a value of 0.

5. Three bits—*a*3, *a*2, and *a*1—are changed by errors. The received codeword is 01011. The syndrome is 1. The dataword is not created. This shows that the simple parity check, guaranteed to detect one single error, can also find any odd number of errors.

A parity-check code can detect an odd number of errors.

–

CYCLIC CODES

Cyclic codes are special linear block codes with one extra property. In a **cyclic code,** if a codeword is cyclically shifted (rotated), the result is another codeword.

Example: if 1011000 is a codeword and we cyclically left-shift, then 0110001 is also a codeword.

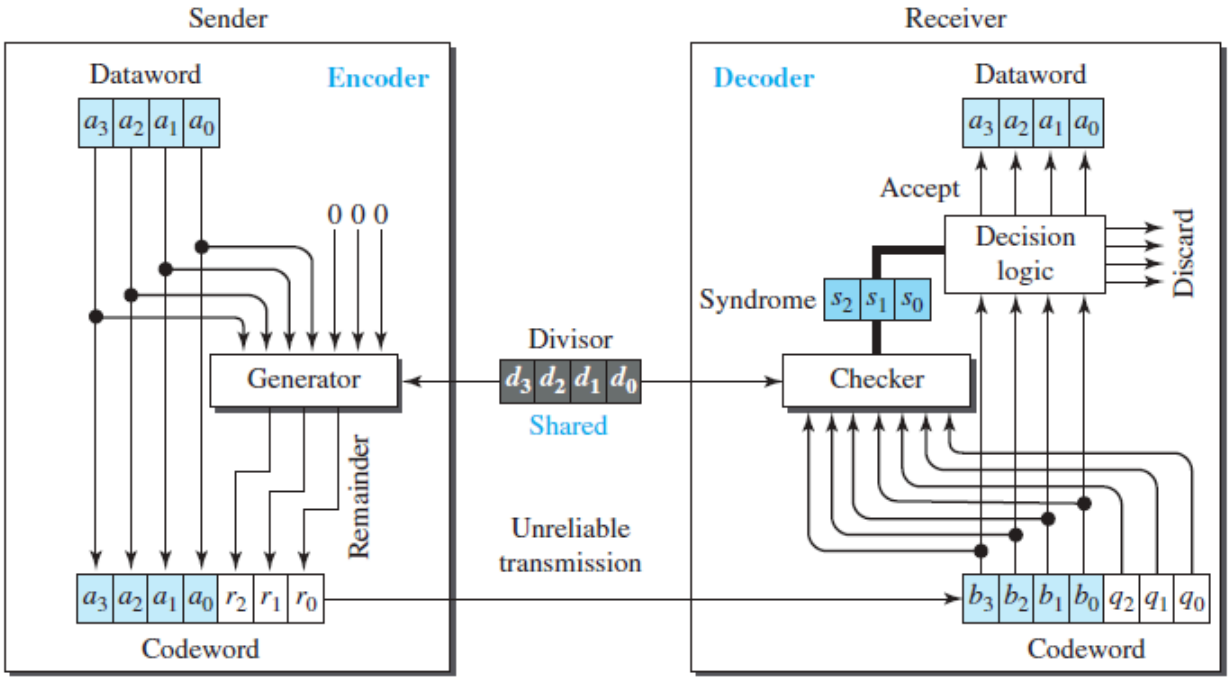
In this case, if we call the bits in the first word *a*0 to *a*6, and the bits in the second word *b*0 to *b*6, we can shift the bits by using the following:

***b*1 =*a*0 *b*2=*a*1 *b*3=*a*2 *b*4=*a*3 *b*5=*a*4 *b*6=*a*5 *b*0= *a*6**

******CYCLIC REDUNDANCY CHECK*****

- We can create cyclic codes to correct errors.
- **Cyclic redundancy check (CRC)** is a subset of cyclic codes, which is used in networks such as LANs and WANs.

Figure 10.5 *CRC encoder and decoder*



ENCODER:

Here the dataword has k bits (4 here); the codeword has n bits (7 here). The size of the dataword is augmented by adding $n - k$ (3 here) 0s to the right-hand side of the word. The n-bit result is fed into the generator. The generator uses a divisor of size $n - k + 1$ (4 here), predefined and agreed upon. The generator divides the augmented dataword by the divisor (modulo-2 division). The quotient of the division is discarded; the remainder ($r2r1r0$) is appended to the dataword to create the codeword.

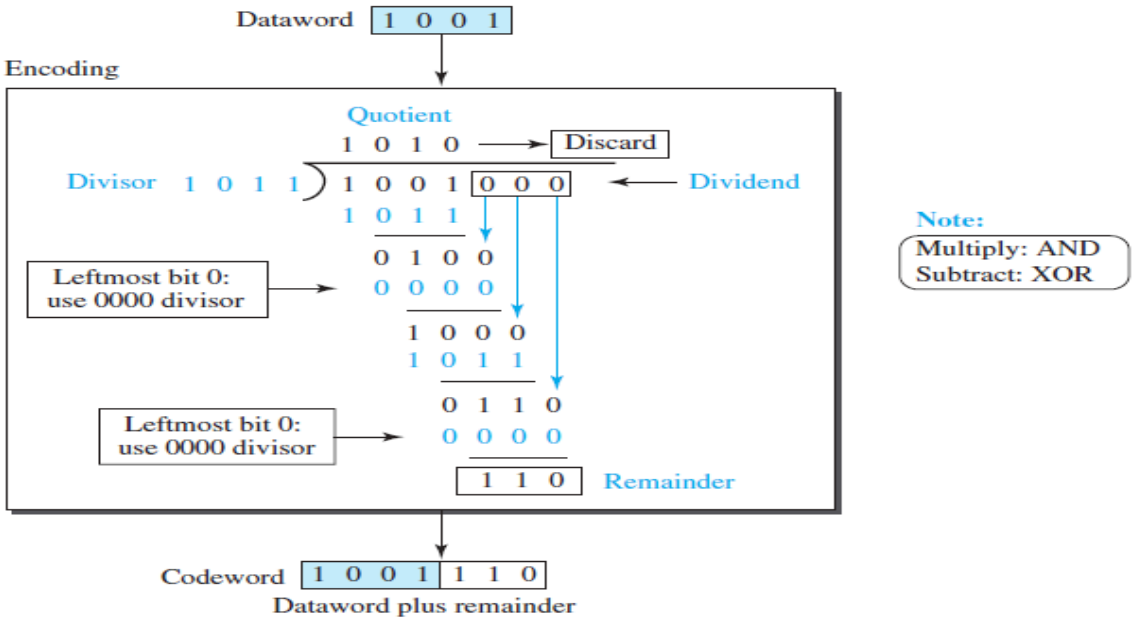
DECODER:

It receives the codeword (possibly corrupted in transition). A copy of all n bits is fed to the checker, which is a replica of the generator. The remainder produced by the checker is a syndrome of $n - k$ (3 here) bits, which is fed to the decision logic analyzer. The analyzer has a simple function. If the syndrome bits are all 0s, the 4 leftmost bits of the codeword are accepted as the dataword (interpreted as no error); otherwise, the 4 bits are discarded (error).

Encoder

The encoder takes a dataword and augments it with $n - k$ number of 0s. It then divides the augmented dataword by the divisor, as shown in Figure 10.6.

Figure 10.6 *Division in CRC encoder*



Decoder

The codeword can change during transmission. The decoder does the same division process as the encoder. The remainder of the division is the syndrome. If the syndrome is all 0s, there is no error with a high probability; the dataword is separated from the received codeword and accepted. Otherwise, everything is discarded.

Figure 10.7 *Division in the CRC decoder for two cases*

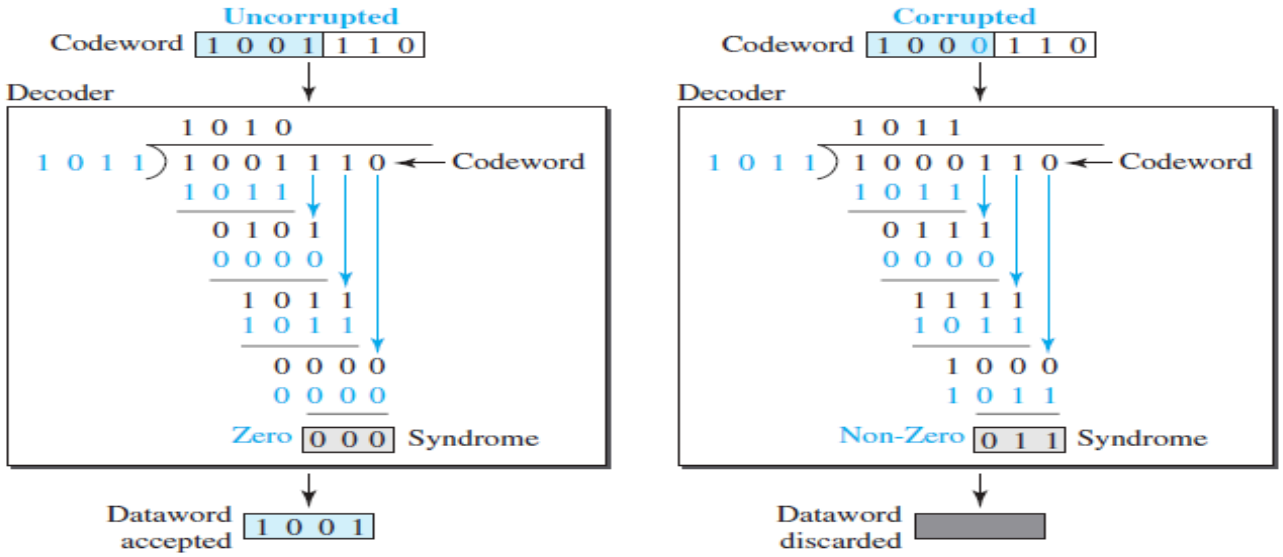


Figure 10.7 show two cases: The left-hand figure shows the value of the syndrome when no error has occurred; the syndrome is 000. The right-hand part of the figure shows the case in which there is a single error. The syndrome is not all 0s (it is 011).

❖ POLYNOMIALS

A better way to understand cyclic codes and how they can be analyzed is to represent them as polynomials. Again, this section is optional. A pattern of 0s and 1s can be represented as a **polynomial** with coefficients of 0 and 1. The power of each term shows the position of the bit; the coefficient shows the value of the bit.

Figure 10.8 *A polynomial to represent a binary word*

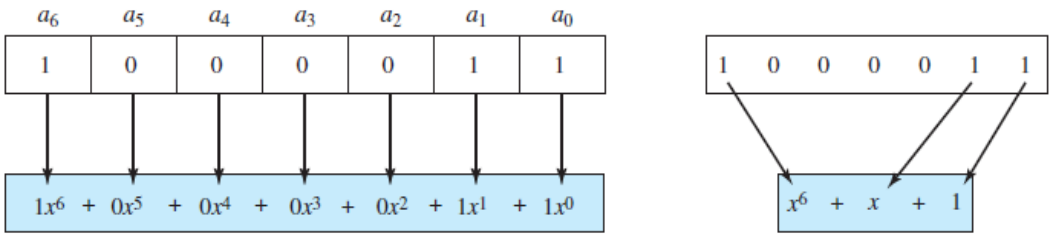


Figure 10.8 shows a binary pattern and its polynomial representation. In Figure 10.8a we show how to translate a binary pattern into a polynomial; in Figure 10.8b we show how the polynomial can be shortened by removing all terms with zero coefficients and replacing $x1$ by x and $x0$ by 1.

Degree of a Polynomial

The degree of a polynomial is the highest power in the polynomial. For example, the degree of the polynomial $x6 + x + 1$ is 6.

Advantages of Cyclic Codes

Cyclic codes have a very good performance in detecting single-bit errors, double errors, an odd number of errors, and burst errors. They can easily be implemented in hardware and software. They are especially fast when implemented in hardware.

CYCLIC CODE ANALYSIS

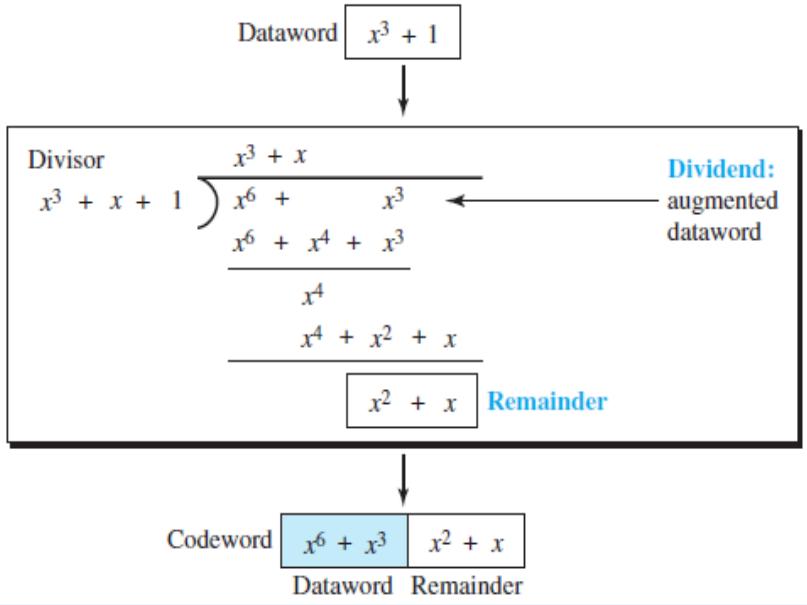
Dataword:$d(x)$ Codeword:$c(x)$ Generator:$g(x)$ Syndrome:$s(x)$ Error:$e(x)$

In a cyclic code,

1. If $s(x) \neq 0$, one or more bits is corrupted.
2. If $s(x) = 0$, either
   a. No bit is corrupted, or
   b. Some bits are corrupted, but the decoder failed to detect them.

CYCLIC CODE ENCODER USING POLYNOMIALS

Figure 10.9 *CRC division using polynomials*



The divisor in a cyclic code is normally called the *generator polynomial* or simply the *generator*.

STANDARD POLYNIMIALS

Table 10.4 *Standard polynomials*

Name	*Polynomial*	*Used in*
CRC-8	$x^8 + x^2 + x + 1$ `100000111`	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$ `11000110101`	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$ `10001000000100001`	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ `10000010011000001000111011011011`	LANs

CHECKSUM

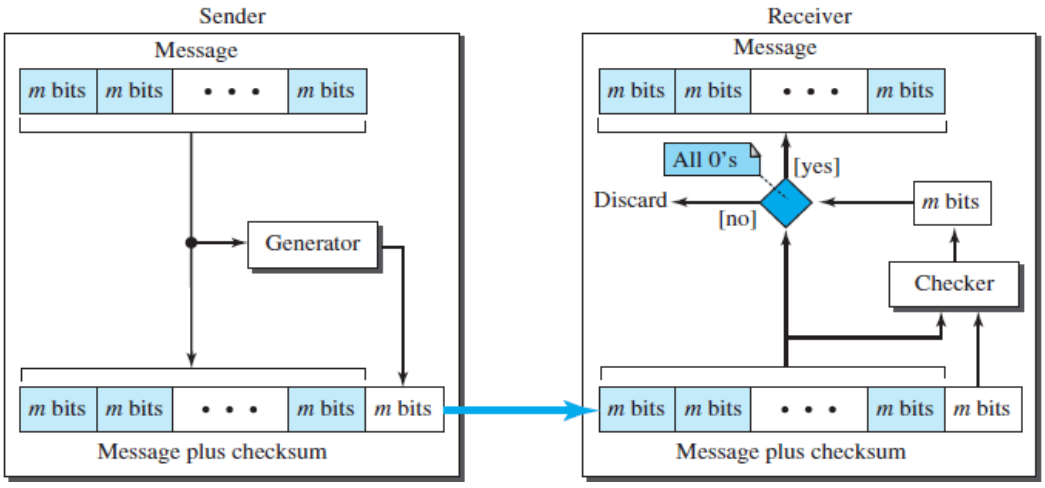
Checksum is an error-detecting technique that can be applied to a message of any length.

In the Internet, the checksum technique is mostly used at the network and transport layer rather than the data-link layer.

At the source, the message is first divided into *m*-bit units. The generator then creates an extra *m*-bit unit called the ***checksum,*** which is sent with the message.

At the destination, the checker creates a new checksum from the combination of the message and sent checksum. If the new checksum is all 0s, the message is accepted; otherwise, the message is discarded

Figure 10.15 *Checksum*



SYLLABUS:

MULTI ACCESS PROTOCOLS - ALOHA, CSMA, COLLISION FREE PROTOCOLS, ETHERNET-PHYSICAL LAYER, ETHERNET MAC SUB LAYER, DATA LINK LAYER SWITCHING & USE OF BRIDGES, LEARNING BRIDGES, SPANNING TREE BRIDGES, REPEATERS, HUBS, BRIDGES, SWITCHES, ROUTERS AND GATEWAYS.

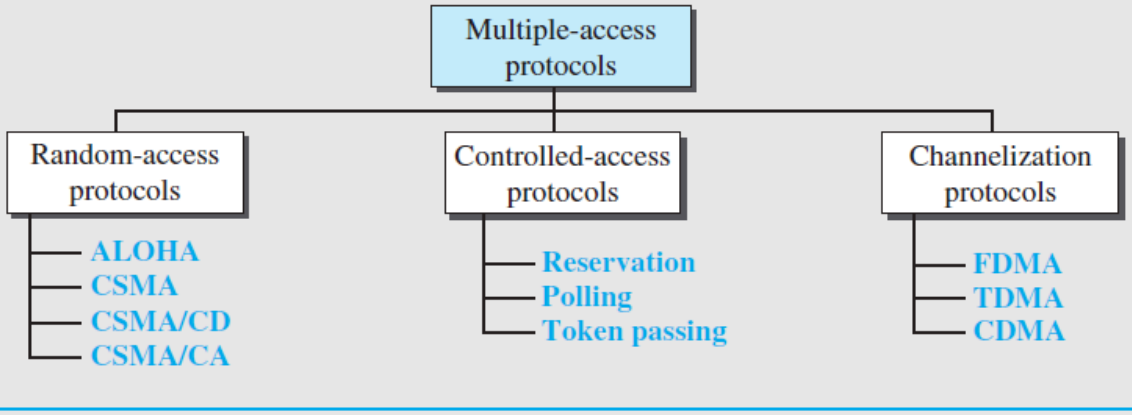
MULTI ACCESS PROTOCOLS

When nodes or stations are connected and use a common link, called a *multipoint* or *broadcast link,* we need a multiple-access protocol to coordinate access to the link.

Many protocols have been devised to handle access to a shared link. All of these protocols belong to a sub layer in the data-link layer called *media access control (MAC).*

We categorize them into three groups, as shown in Figure 12.1.

Figure 12.1 *Taxonomy of multiple-access protocols*



❖ RANDOM ACCESS (OR) CONTENTION METHOD

- No station is superior to another station and none is assigned control over another.
- At each instance, a station that has data to send uses a procedure defined by the protocol to make a decision on whether or not to send. This decision depends on the state of the medium (idle or busy).

  In other words, each station can transmit when it desires on the condition that it follows the predefined procedure, including testing the state of the medium.

Two features give this method its name.

- First, there is no scheduled time for a station to transmit. Transmission is random among the stations. That is why these methods are called random access.
- Second, no rules specify which station should send next. Stations compete with one another to access the medium. That is why these methods are also called contention methods.

In a random-access method, each station has the right to the medium without being controlled by any other station. However, if more than one station tries to send, there is an access conflict—collision—and the frames will be either destroyed or modified. To avoid access conflict or to resolve it when it happens, each station follows a procedure that answers the following questions:

1. When can the station access the medium?
2. What can the station do if the medium is busy?
3. How can the station determine the success or failure of the transmission?
4. What can the station do if there is an access conflict?

- The random-access methods have evolved from a very interesting protocol known as **ALOHA**, which used a very simple procedure called multiple access (MA).
- The method was improved with the addition of a procedure that forces the station to sense the medium before transmitting. This was called **carrier sense multiple access (CSMA).**
  This method later evolved into two parallel methods:
  1. **Carrier sense multiple access with collision detection (CSMA/CD)**, which tells the station what to do when a collision is detected
  2. **Carriers sense multiple access with collision avoidance (CSMA/CA),** which tries to avoid the collision.

ALOHA

- ALOHA, the earliest random access method was developed at the University of Hawaii in early 1970.
- It was designed for a radio (wireless) LAN, but it can be used on any shared medium.

It is obvious that there are potential collisions in this arrangement. The medium is shared between the stations. When a station sends data, another station may attempt to do so at the same time. The data from the two stations collide and become garbled.

Pure ALOHA

The original ALOHA protocol is called pure ALOHA. This is a simple but elegant protocol.

The idea is that each station sends a frame whenever it has a frame to send (multiple access). However, since there is only one channel to share, there is the possibility of collision between frames from different stations. Figure 12.2 shows an example of frame collisions in pure ALOHA.

There are four stations (unrealistic assumption) that contend with one another for access to the shared channel. The figure shows that each station sends two frames; there are a total of eight frames on the shared medium. Some of these frames collide because multiple frames are in contention for the shared channel. Figure 12.2 shows that only two frames survive: one frame from station 1 and one frame from station 3.

Even if one bit of a frame coexists on the channel with one bit from another frame, there is a collision and both will be destroyed. It is obvious that we need to resend the frames that have been destroyed during transmission.

1. The pure ALOHA protocol relies on acknowledgments from the receiver. When a station sends a frame, it expects the receiver to send an acknowledgment. If the acknowledgment does not arrive after a time-out period, the station assumes that the frame (or the acknowledgment) has been destroyed and resends the frame.

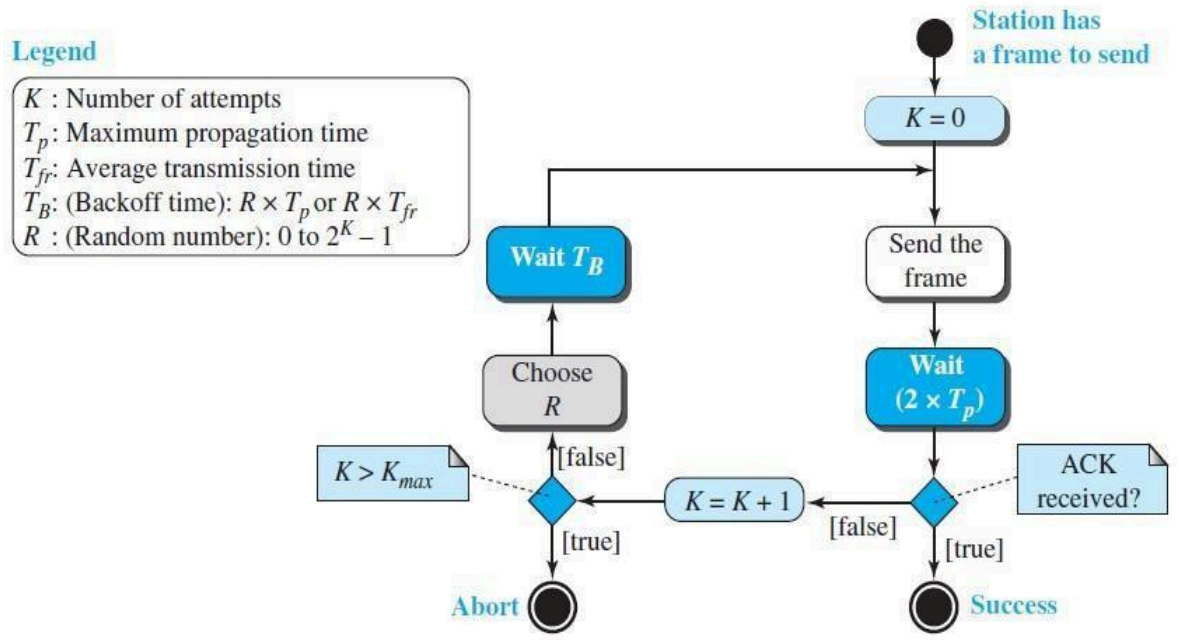
A collision involves two or more stations. If all these stations try to resend their frames after the time- out, the frames will collide again.

Backoff Time (T_B): Pure ALOHA dictates that when the time-out period passes, each station waits a random amount of time before resending its frame. The randomness will help avoid more collisions. We call this time the backoff time (T_B)

2. Pure ALOHA has a second method to prevent congesting the channel with retransmitted frames. After a maximum number of retransmission attempts **K_{max}** a station must give up and try later.

Figure 12.3 shows the procedure for pure ALOHA based on the above strategy.

Figure 12.3 *Procedure for pure ALOHA protocol*



- The time-out period is equal to the maximum possible round-trip propagation delay, which is twice the amount of time required to send a frame between the two most widely separated stations (2 × Tp).
- The value of K_{max} is usually chosen as 15.
- The backoff time T_B is a random value that normally depends on K (the number of attempted unsuccessful transmissions). The formula for T_B depends on the implementation. One common formula is the binary exponential backoff.

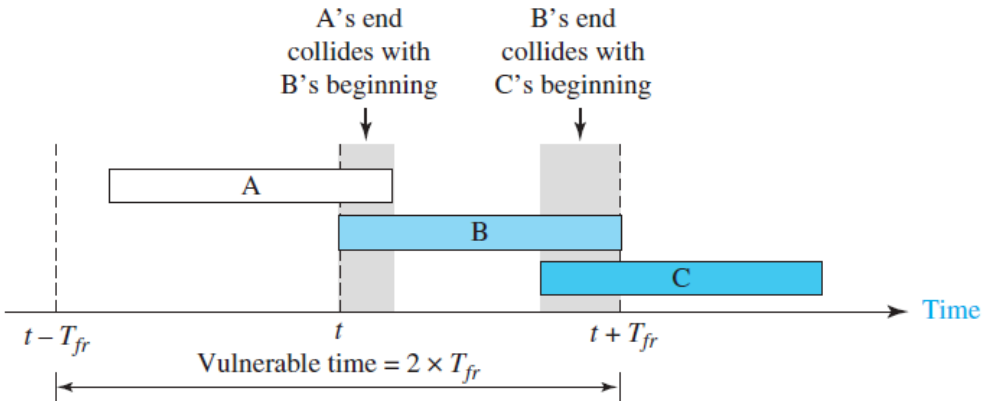
Binary Exponential Backoff: In this method, for each retransmission, a multiplier R = 0 to 2^{K-1} is randomly chosen and multiplied by Tp (maximum propagation time) or T_{fr} (the average time required to send out a frame) to find T_B.

Vulnerable time (Pure ALOHA)

Vulnerable time: *the length of time in which there is a possibility of collision.*

We assume that the stations send fixed-length frames with each frame taking T_{fr} seconds to send. Figure 12.4 shows the vulnerable time for station B.

Figure 12.4 *Vulnerable time for pure ALOHA protocol*



Pure ALOHA vulnerable time = $2 \times T_{fr}$

Throughput (Pure ALOHA)

Let us call G the average number of frames generated by the system during one frame transmission time. Then it can be proven that the average number of successfully transmitted frames for pure ALOHA is S = G × e−2G. The maximum throughput S_{max} is 0.184, for G = 1/2.

In other words, if one-half a frame is generated during one frame transmission time (one frame during two frame transmission times), then 18.4 percent of these frames reach their destination successfully.

The throughput for pure ALOHA is $S = G \times e^{-2G}$.
The maximum throughput $S_{max} = 1/(2e) = 0.184$ when $G = (1/2)$.

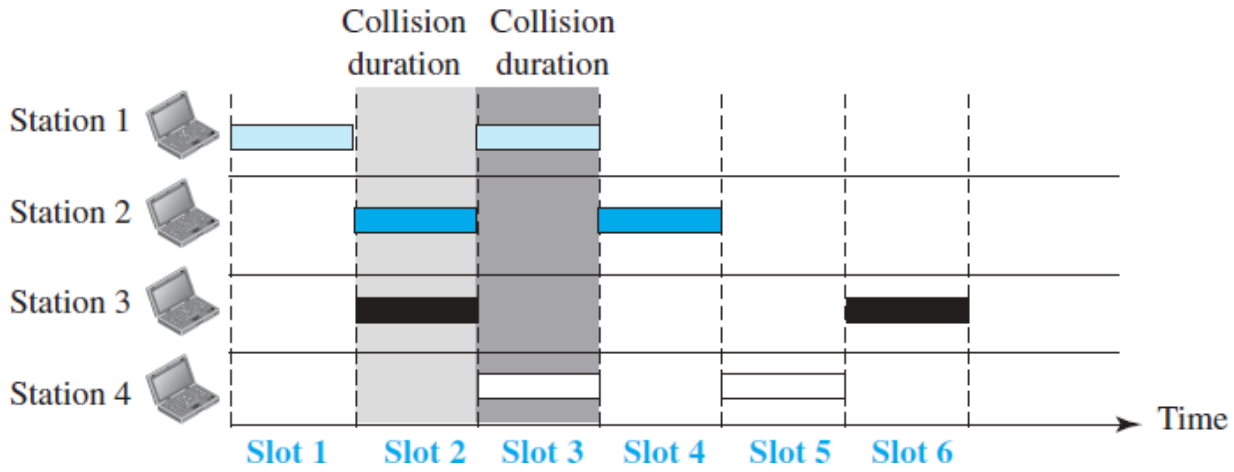
Slotted ALOHA

Slotted ALOHA was invented to improve the efficiency of pure ALOHA.

In slotted ALOHA we divide the time into slots of T_{fr} seconds and force the station to send only at the beginning of the time slot. Figure 12.5 shows an example of frame collisions in slotted ALOHA.

A station is allowed to send only at the beginning of the synchronized time slot, if a station misses this moment, it must wait until the beginning of the next time slot. This means that the station which started at the beginning of this slot has already finished sending its frame. Of course, there is still the possibility of collision if two stations try to send at the beginning of the same time slot.

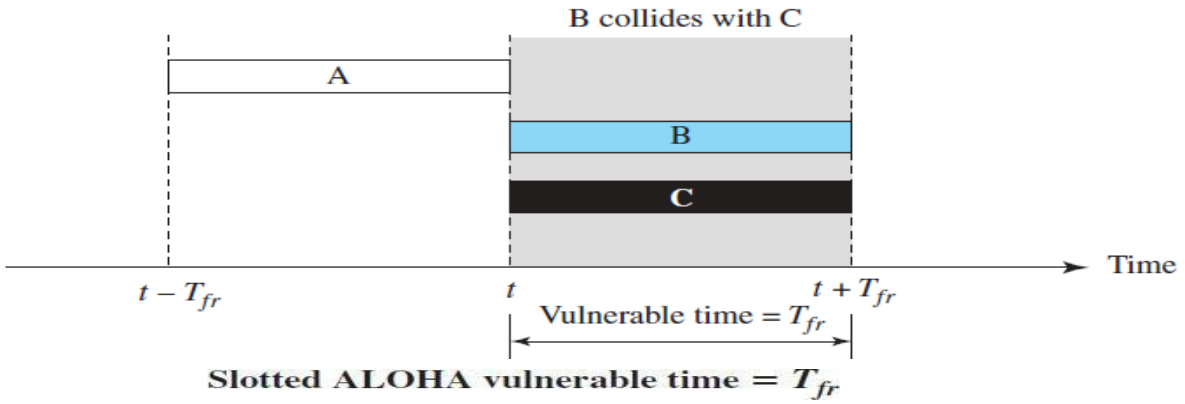
Figure 12.5 *Frames in a slotted ALOHA network*



Vulnerable time (Slotted ALOHA)

The vulnerable time is equal to T_{fr}. (Figure 12.6 shows the situation)

Figure 12.6 *Vulnerable time for slotted ALOHA protocol*



Throughput (Slotted ALOHA)

It can be proven that the average number of successful transmissions for slotted ALOHA is $S = G \times e^{-G}$. The maximum throughput S_{max} is 0.368, when $G = 1$.

In ent of these frames rea

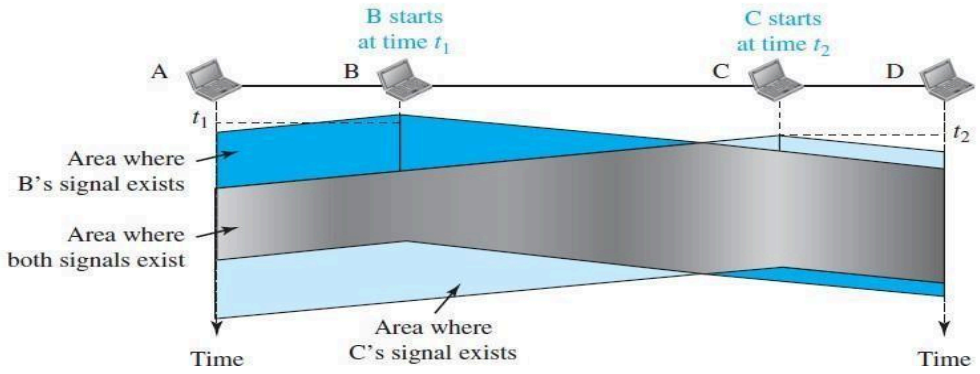
The throughput for slotted ALOHA is $S = G \times e^{-G}$.
The maximum throughput $S_{max} = 0.368$ when $G = 1$.

CSMA (CARRIER SENSE MULTIPLE ACCESS)

- To minimize the chance of collision, increase the performance the CSMA method was developed. (The chance of collision can be reduced if a station senses the medium before trying to use it.)
- **Carrier sense multiple access (CSMA)** requires that each station first listen to the medium (or check the state of the medium) before sending. In other words, CSMA is based on the principle **"sense before transmit"** or **"listen before talk."**
- CSMA can reduce the possibility of collision, but it cannot eliminate it.

The possibility of collision still exists because of propagation delay; when a station sends a frame, it still takes time (although very short) for the first bit to reach every station and for every station to sense it. In other words, a station may sense the medium and find it idle, only because the first bit sent by another station has not yet been received.

Figure 12.7 *Space/time model of a collision in CSMA*



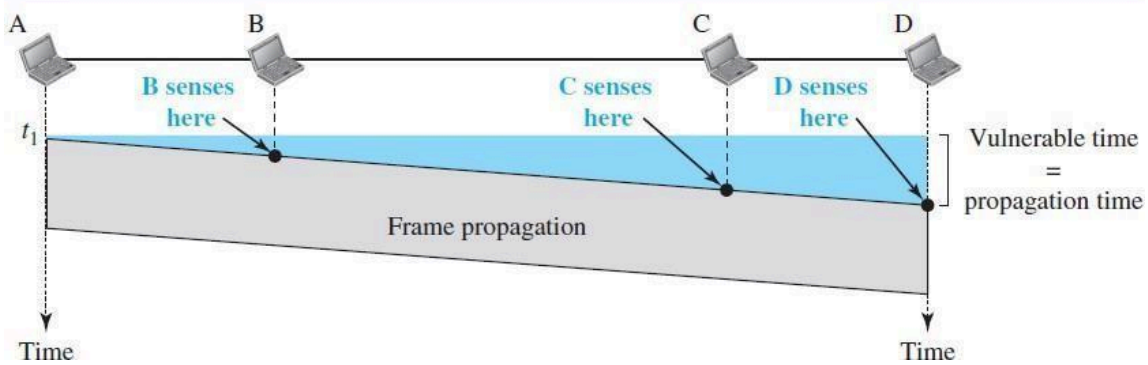
At time t1, station B senses the medium and finds it idle, so it sends a frame. At time t2 (t2 > t1), station C senses the medium and finds it idle because, at this time, the first bits from station B have not reached station C. Station C also sends a frame. The two signals collide and both frames are destroyed.

Vulnerable Time (CSMA

The vulnerable time for CSMA is the propagation time Tp. (This is the time needed for a signal to propagate from one end of the medium to the other).

Figure 12.8 shows the worst case. The leftmost station, A, sends a frame at time t1, which reaches the rightmost station, D, at time t1 + Tp. The gray area shows the vulnerable area in time and space.

Figure 12.8 *Vulnerable time in CSMA*



PERSISTENCE METHODS

What should a station do if the channel is busy? What should a station do if the channel is idle?

Three methods have been devised to answer these questions: the 1-persistent method, the non persistent method, and the p-persistent method.

Figure 12.9 *Behavior of three persistence methods*

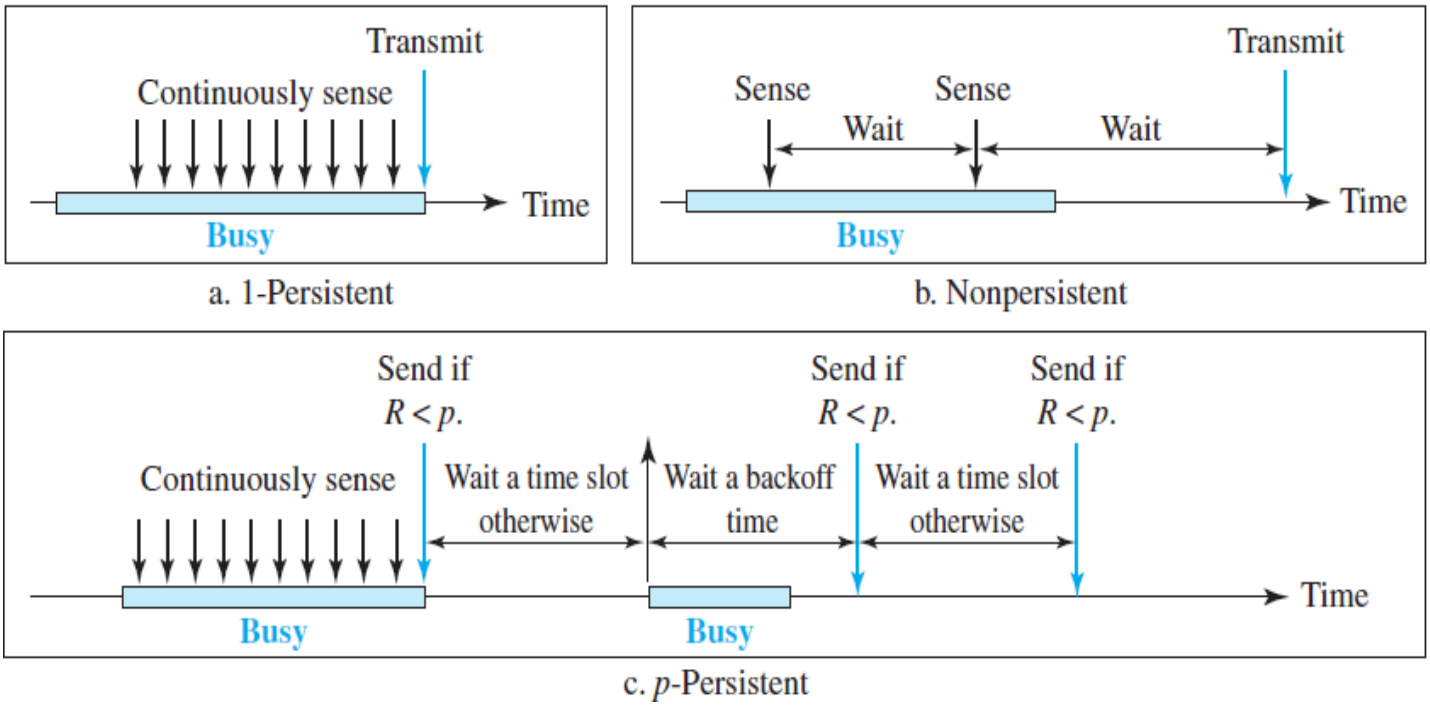


Figure 12.9 shows the behavior of three persistence methods when a station finds a channel busy.

1- PERSISTENT

- The 1-persistent method is simple and straightforward.
- In this method, after the station finds the line idle, it sends its frame immediately (with probability 1).
- It has the highest chance of collision because two or more stations may find the line idle and send their frames immediately.
- **Ethernet** uses this method.

NONPERSISTENT

- In this method, a station that has a frame to send senses the line, If the line is idle, it sends immediately. If the line is not idle, it waits a random amount of time and then senses the line again.
- It reduces the chance of collision because it is unlikely that two or more stations will wait the same amount of time and retry to send simultaneously. However, this method reduces the efficiency of the network because the medium remains idle when there may be stations with frames to send.

P-PERSISTENT

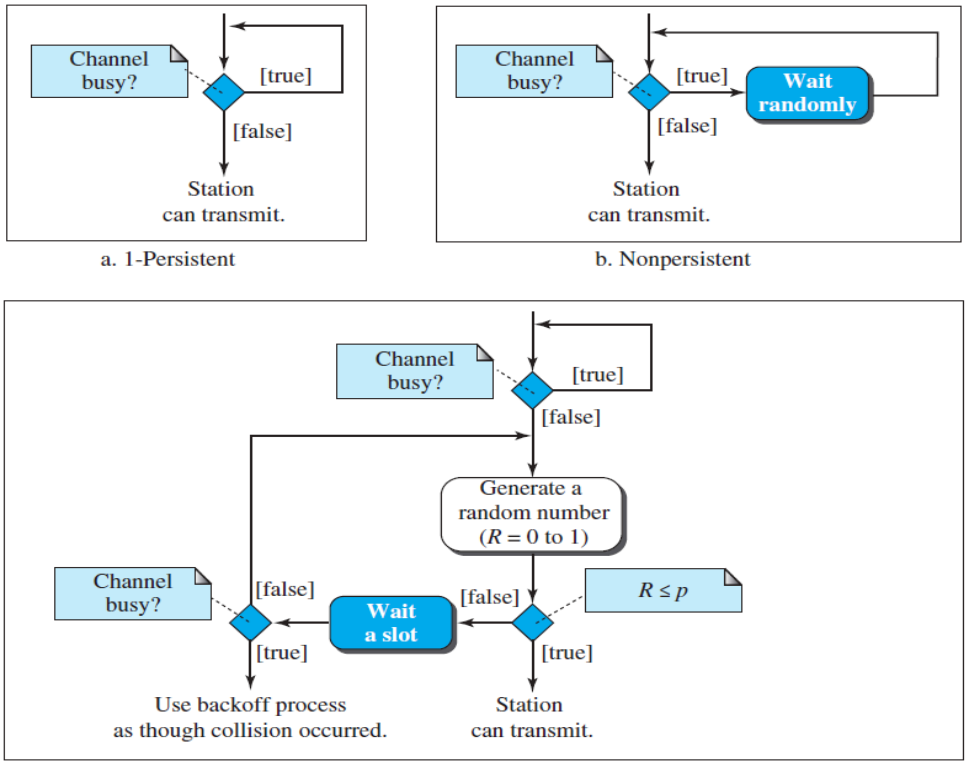
- The p-persistent method is used if the channel has time slots with a slot duration equal to or greater than the maximum propagation time.
- The p-persistent approach combines the advantages of the other two strategies. It reduces the chance of collision and improves efficiency.

In this method, after the station finds the line idle it follows these steps:

1. With probability p, the station sends its frame
2. With probability q = 1 − p, the station waits for the beginning of the next time slot and checks the line again.
   a. If the line is idle, it goes to step 1.

Figure 12.10 *Flow diagram for three persistence methods*

off procedure.



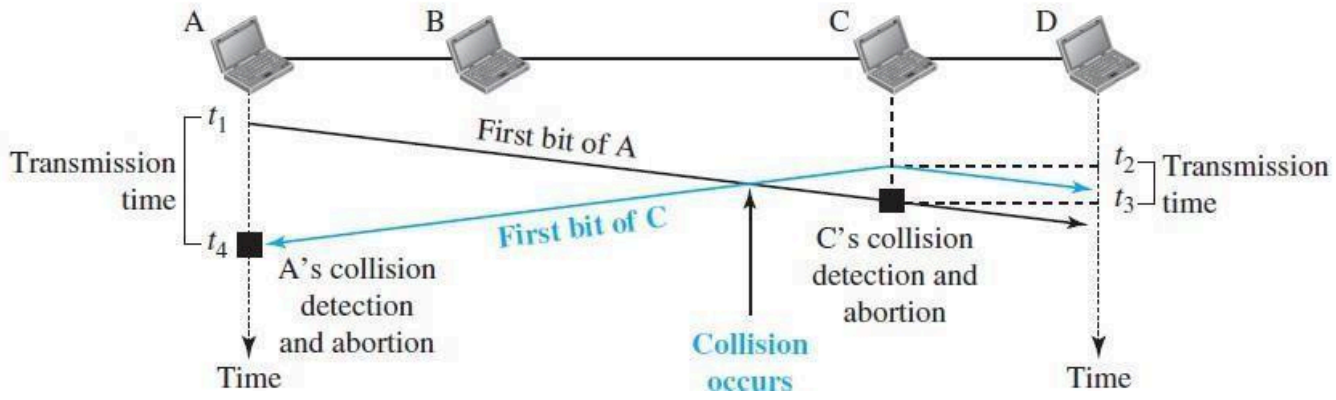
❖ CSMA/CD (CARRIER SENSE MULTIPLE ACCESS WITH COLLISION DETECTION)

The CSMA method does not specify the procedure following a collision.

- Carriers sense multiple access with collision detection (CSMA/CD) augments the algorithm to handle the collision.
- In CSMA/CD, a station monitors the medium after it sends a frame to see if the transmission was successful. If so, the station is finished. If, however, there is a collision, the frame is sent again.

To better understand CSMA/CD, let us look at the first bits transmitted by the two stations involved in the collision. Although each station continues to send bits in the frame until it detects the collision, we show what happens as the first bits collide. In Figure 12.11, stations A and C are involved in the collision.

Figure 12.11 *Collision of the first bits in CSMA/CD*



At time t1, station A has executed its persistence procedure and starts sending the bits of its frame. At time t2, station C has not yet sensed the first bit sent by A. Station C executes its persistence procedure and starts sending the bits in its frame, which propagate both to the left and to the right. The collision occurs sometime after time t2. Station C detects a collision at time t3 when it receives the first bit of A's frame. Station C immediately (or after a short time, but we assume immediately) aborts transmission. Station A detects collision at time t4 when it receives the first bit of C's frame; it also immediately aborts transmission.

Minimum frame size

For CSMA/CD to work, we need a restriction on the frame size. Before sending the last bit of the frame, the sending station must detect a collision, if any, and abort the transmission. This is so because the station, once the entire frame is sent, does not keep a copy of the frame and does not monitor the line for collision detection. Therefore, the frame transmission time T_{fr} must be at least two times the maximum propagation time Tp.

Energy Level

We can say that the level of energy in a channel can have three values: zero, normal, and abnormal.

- At the zero level, the channel is idle.
- At the normal level, a station has successfully captured the channel and is sending its frame.
- At the abnormal level, there is a collision and the level of the energy is twice the normal level.

Throughput

The throughput of CSMA/CD is greater than that of pure or slotted ALOHA. The maximum throughput occurs at a different value of G and is based on the persistence method and the value of p in the p-persistent approach. *For the 1-persistent method, the maximum throughput is around 50 percent when G = 1. For the nonpersistent method, the maximum throughput can go up to 90 percent when G is between 3 and 8.*

Traditional Ethernet

One of the LAN protocols that used CSMA/CD is the traditional Ethernet with the data rate of 10 Mbps.

CSMA/CA(CARRIER SENSE MULTIPLE ACCESS WITH COLLISION AVOIDANCE)

Carrier sense multiple access with collision avoidance (CSMA/CA) was invented for wireless networks. Collisions are avoided through the use of CSMA/CA's three strategies: the interframe space, the contention window, and acknowledgments.

Interframe Space (IFS): First, collisions are avoided by deferring transmission even if the channel is found idle. When an idle channel is found, the station does not send immediately. It waits for a period of time called the interframe space or IFS.

Contention Window: The contention window is an amount of time divided into slots. A station that is ready to send chooses a random number of slots as its wait time. The number of slots in the window changes according to the binary exponential backoff strategy. One interesting point about the contention window is that the station needs to sense the channel after each time slot. However, if the station finds the channel busy, it does not restart the process; it just stops the timer and restarts it when the channel is sensed as idle. This gives priority to the station with the longest waiting time. See Figure 12.16.

Acknowledgment: With all these precautions, there still may be a collision resulting in destroyed data. In addition, the data may be corrupted during the transmission. The positive acknowledgment and the time-out timer can help guarantee that the receiver has received the frame.

COLLISION-FREE PROTOCOLS

A BIT-MAP PROTOCOL

- In bit-map method, each contention period consists of exactly N slots. If station 0 has a frame to send, it transmits a 1 bit during the slot 0. No other station is allowed to transmit during this slot.
- Regardless of what station 0 does, station 1 gets the opportunity to transmit a 1 bit during slot 1, but only if it has a frame queued.

  In general, station j may announce that it has a frame to send by inserting a 1 bit into slot j. After all N slots have passed by; each station has complete knowledge of which stations wish to transmit. At that point, they begin transmitting frames in numerical order (see Fig. 4-6).
- After the last ready station has transmitted its frame, an event all stations can easily monitor, another N-bit contention period is begun.

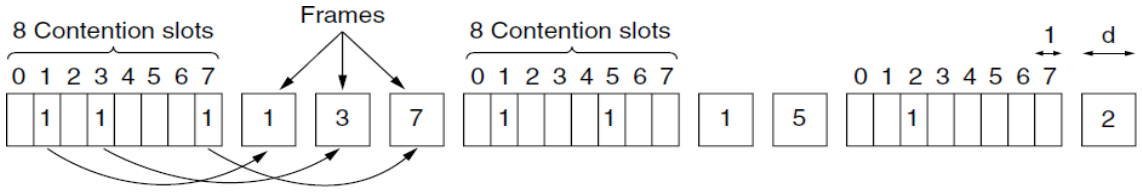


Figure 4-6. The basic bit-map protocol.

Since everyone agrees on who goes next, there will never be any collisions.

Limitation of bit-map protocol

If a station becomes ready just after its bit slot has passed by, it is out of luck and must remain silent until every station has had a chance and the bit map has come around again.

Protocols like this in which the desire to transmit is broadcast before the actual transmission are called reservation protocols because they reserve channel ownership in advance and prevent collisions.

Token Passing

The essence of the bit-map protocol is that it lets every station transmit a frame in turn in a predefined order. Another way to accomplish the same thing is to pass a small message called a token from one station to the next in the same predefined order. The token represents permission to send. If a station has a frame queued for transmission when it receives the token, it can send that frame before it passes the token to the next station. If it has no queued frame, it simply passes the token.

In a token ring protocol, the topology of the network is used to define the order in which stations send. The stations are connected one to the next in a single ring. Passing the token to the next station then simply consists of receiving the token in from one direction and transmitting it out in the other direction, as seen in Fig. 4-7. Frames are also transmitted in the direction of the token. This way they will circulate around the ring and reach whichever station is the destination. However, to stop the frame circulating indefinitely (like the token), some station needs to remove it from the ring. This station may be either the one that originally sent the frame, after it has gone through a complete cycle, or the station that was the intended recipient of the frame.

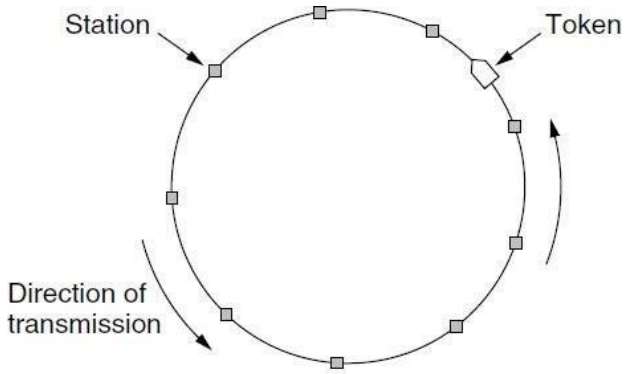


Figure 4-7. Token ring.

Note that we do not need a physical ring to implement token passing. The channel connecting the stations might instead be a single long bus. Each station then uses the bus to send the token to the next station in the predefined sequence. Possession of the token allows a station to use the bus to send one frame, as before. This protocol is called token bus.

Problem with the basic bit-map protocol, and by extension token passing, is that the overhead is 1 bit per station, so bit-map protocol does not scale well to networks with thousands of stations.

BINARY COUNTDOWN

- In binary countdown a station wanting to use the channel broadcasts its address as a binary bit string, starting with the high order bit. All addresses are assumed to be the same length.
- The bits in each address position from different stations are BOOLEAN ORed together by the channel when they are sent at the same time. This protocol is binary countdown.
- It was used in Data kit (Fraser, 1987). It implicitly assumes that the transmission delays are negligible so that all stations see asserted bits essentially instantaneously.

To avoid conflicts, an arbitration rule must be applied: as soon as a station sees that a high-order bit position that is 0 in its address has been overwritten with a 1, it gives up.

Example:

If stations 0010, 0100, 1001, and 1010 are all trying to get the channel, in the first bit time the stations transmit 0, 0, 1, and 1, respectively. These are ORed together to form a 1. Stations 0010 and 0100 see the 1 and know that a higher-numbered station is competing for the channel, so they give up for the current round. Stations 1001 and 1010 continue. The next bit is 0, and both stations continue. The next bit is 1, so station 1001 gives up. The winner is station 1010 because it has the highest address. After winning the bidding, it may now transmit a frame, after which another bidding cycle starts. The protocol is illustrated in Fig. 4-8.

It has the property that higher- numbered stations have a higher priority than lower-numbered stations, which may be either good or bad, depending on the context.

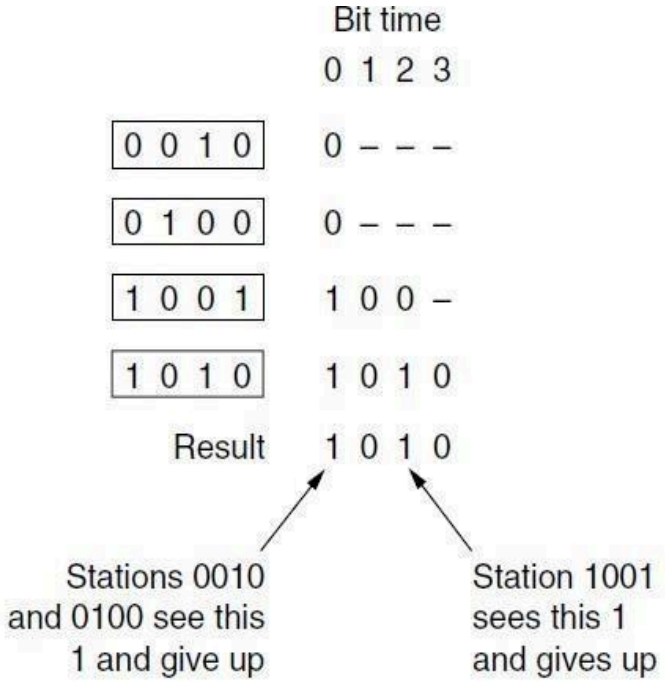


Figure 4-8. The binary countdown protocol. A dash indicates silence.

ETHERNET

- The Ethernet LAN was developed in the 1970s by Robert Metcalfe and David Boggs.
- It is the most widely used local area network (LAN) technology/Architecture.
- Ethernet Uses either Bus topology (Classic Ethernet) or Star topology (switched Ethernet).

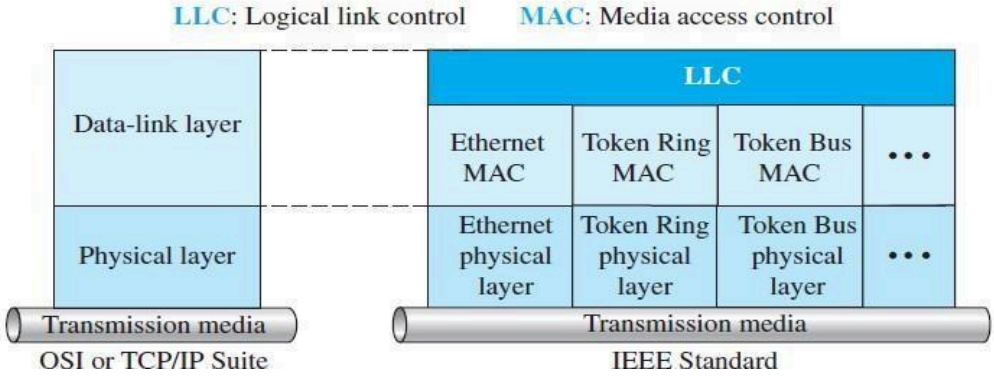
IEEE 802

- IEEE 802 is a family of IEEE standards dealing with local area networks and metropolitan area networks.
- In 1985, the Computer Society of the IEEE started a project, called Project 802, to set standards to enable intercommunication among equipment from a variety of manufacturers.
- Many of the designs for personal, local, and metropolitan area networks have been standardized under the name of IEEE 802.

  The most important of the survivors are 802.3 ***(Ethernet), 802.11 (wireless LAN), 802.15(Bluetooth /wireless PAN), 802.16 (wireless MAN).***

The relationship of the 802 Standard to the TCP/IP protocol suite is shown in Figure 13.1.

Figure 13.1 *IEEE standard for LANs*



The IEEE has subdivided the data-link layer into two sub layers: logical link control (LLC) and media access control (MAC).

Logical Link Control (LLC) / DATA LINK CONTROL (DLC)

- It handles framing, flow control, and error control.
- In IEEE Project 802, flow control, error control, and part of the framing duties are collected into one sublayer called the *logical link control* (LLC). Framing is handled in both the LLC sublayer and the MAC sublayer.

Media Access Control (MAC)

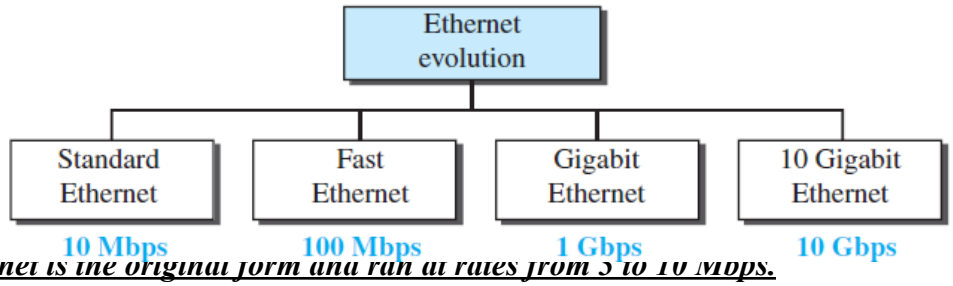
- Framing, Access control (Multi Access Protocols) handled by the MAC layer.
- IEEE Project 802 has created a sub layer called ***media access control*** that defines the specific access method for each LAN.

Example: It defines CSMA/CD as the media access method for Ethernet LANs and defines the Token-passing method for Token Ring and Token Bus LANs.

Ethernet Evolution

Ethernet has gone through four generations: **Standard Ethernet** (10 Mbps), **Fast Ethernet** (100 Mbps), **Gigabit Ethernet** (1 Gbps), and **10 Gigabit Ethernet** (10 Gbps), as shown in Figure 13.2

3.2 *Ethernet evolution through four generations*



TYPES OF E

Two kinds of Eth

1. Classic E
2. Switched

CLASSIC ETH

- ***Classic Ethernet is the original form and ran at rates from 3 to 10 Mbps.***
- To control access to the sharing medium , classic Ethernet chose CSMA/CD with 1-persistent method

SWITCHED ETHERNET

- In this we use devices called switches to connect different computers.
- Switched Ethernet is what Ethernet has become and runs at 100 Mbps (Fast Ethernet), 1000 Mbps (Gigabit Ethernet), and 10,000 Mbps (10 Gigabit Ethernet).
- In practice, only switched Ethernet is used nowadays.

ETHERNET NAME	IEEE STANDARD	DATARATE
STANDARD ETHERNET	802.3	10Mbps
FAST ETHERNET	802.3u	100Mbps
GIGABIT ETHERNET	802.3z	1000Mbps
10GIGABIT ETHERNET	802.3az	10000Mbps

CLASSIC ETHERNET PHYSICAL LAYER

- Bob Metcalfe, together with his colleague David Boggs designed and implemented the first local area. It used a single long, thick coaxial cable and ran at 3 Mbps.
- Ethernet was developed by XEROX Corporation in cooperation with DEC (Digital Equipment Corporation) at PARC (Palo Alto Research Center) in 1976.

- The Xerox Ethernet was so successful that DEC, Intel, and Xerox drew up a standard in 1978 for a 10-Mbps Ethernet, called the DIX standard.
- With a minor change, the DIX standard became the IEEE 802.3 standard in 1983.
- Classic Ethernet uses the 1-persistent CSMA/CD algorithm

Each version of Ethernet has a maximum cable length per segment (i.e., unamplified length) over which the signal will propagate. To allow larger networks, multiple cables can be connected by repeaters. An Ethernet could contain multiple cable segments and multiple repeaters, but no two transceivers could be more than 2.5 km apart and no path between any two transceivers could traverse more than four repeaters.

attache puters were

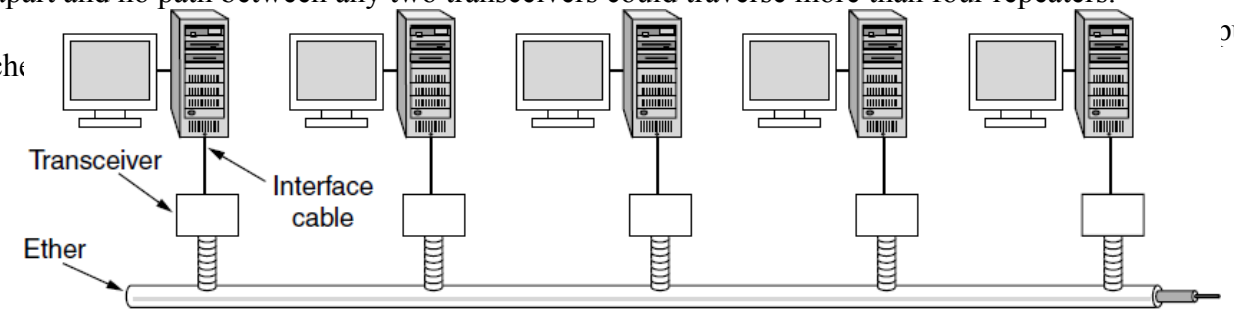


Figure 4-13. Architecture of classic Ethernet.

SWITCHED ETHERNET

Ethernet soon began to evolve away from the single long cable architecture of classic Ethernet. The problems associated with finding breaks or loose connections drove it toward a different kind of wiring pattern, in which each station has a dedicated cable running to a central **hub**. A hub simply connects all the attached wires electrically, as if they were soldered together. This configuration is shown in Fig. 4-17(a).

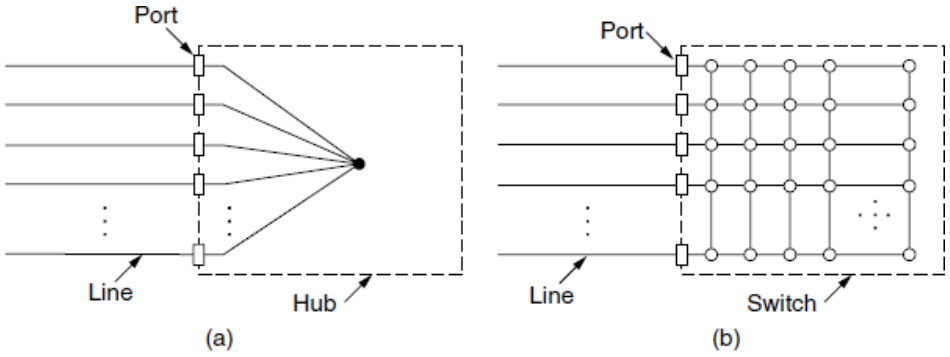


Figure 4-17. (a) Hub. (b) Switch.

Adding or removing a station is simpler in this configuration, and cable breaks can be detected easily. With the advantages of being able to use existing wiring and ease of maintenance, twisted-pair hubs quickly became the dominant form of Ethernet. However, hubs do not increase capacity because they are logically equivalent to the single long cable of classic Ethernet. As more and more stations are added, each station gets a decreasing share of the fixed capacity. Eventually, the LAN will saturate. One way out is to go to a higher speed, say, from 10 Mbps to 100 Mbps, 1 Gbps, or even higher speeds. But with the growth of multimedia and powerful servers, even a 1-Gbps Ethernet can become saturated.

Fortunately, there is another way to deal with increased load: switched Ethernet. The heart of this system is a **switch** containing a high-speed backplane that connects all of the ports, as shown in Fig. 4-17(b). From the outside, a switch looks just like a hub. They are both boxes, typically with 4 to 48 ports, each with a standard RJ-45 connector for a twisted-pair cable. Each cable connects the switch or hub to a single computer, as shown in Fig. 4-18.

SWITCH ADVANTAGES

1. It works like a hub, too.
2. It is easy to add or remove a new station by plugging or unplugging a wire and it is easy to find most faults since a flaky cable or port will usually affect just one station.
3. There is still a shared component that can fail—the switch itself—but if all stations lose connectivity then we can fix the problem by replacing the whole switch.

Switches only output frames to the ports for which those frames are destined. When a switch port receives an Ethernet frame from a station, the switch checks the Ethernet addresses to see which port the frame is destined for.

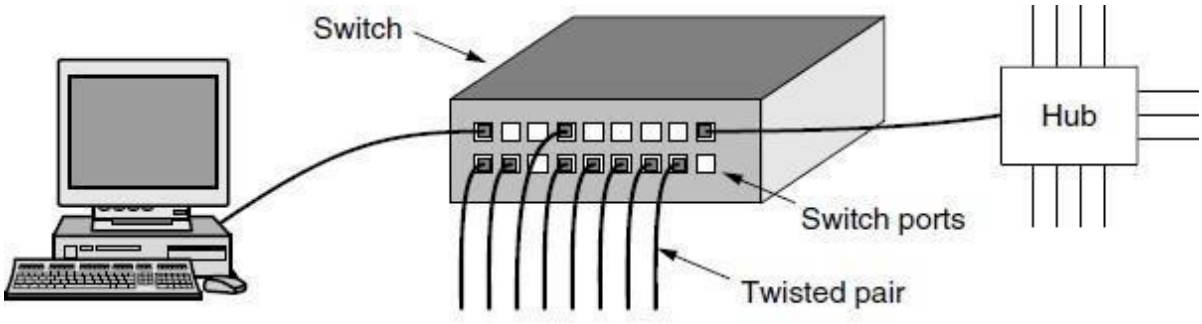


Figure 4-18. An Ethernet switch.

What happens if more than one of the stations or ports wants to send a frame at the same time? Again, switches differ from hubs. In a hub, all stations are in the same **collision domain**. They must use the CSMA/CD algorithm to schedule their transmissions. In a switch, each port is its own independent collision domain. In the common case that the cable is full duplex, both the station and the port can send a frame on the cable at the same time, without worrying about other ports and stations. Collisions are now impossible and CSMA/CD is not needed. However, if the cable is half duplex, the station and the port must contend for transmission with CSMA/CD in the usual way.

A switch improves performance over a hub in two ways.

1. *since there are no collisions, the capacity is used more efficiently.*
2. *With a switch multiple frames can be sent simultaneously (by different stations).*

CLASSIC ETHERNET MAC SUBLAYER PROTOCOL

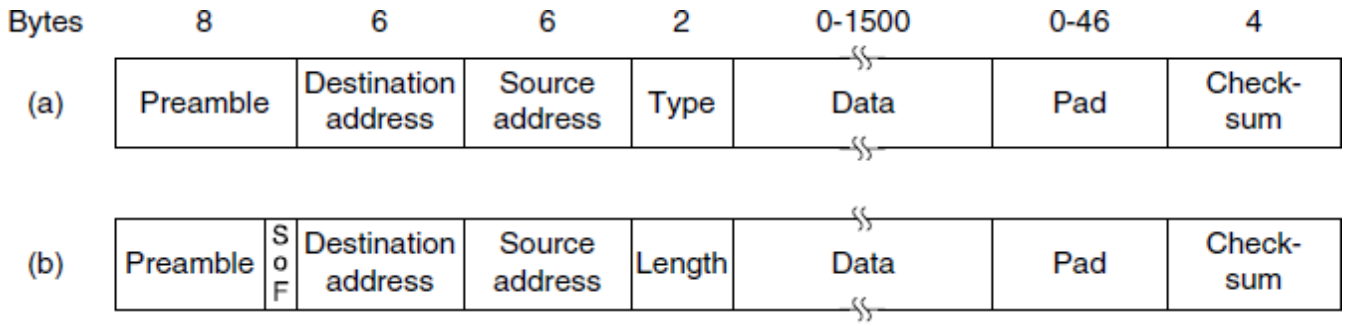


Figure 4-14. Frame formats. (a) Ethernet (DIX). (b) IEEE 802.3.

The format used to send frames is shown in Fig. 4-14.

Preamble

It is of 8 bytes, each containing the bit pattern 10101010 (with the exception of the last byte, in which the last 2 bits are set to 11).

This last byte is called the *Start of Frame* delimiter for 802.3.

The last two 1 bits tell the receiver that the rest of the frame is about to start.

Next come two addresses, one for the destination and one for the source. They are each 6 bytes long.

Destination address

It is 6 bytes long.

The first transmitted bit of the destination address is a 0 for ordinary addresses and a 1 for group addresses.

Group addresses allow multiple stations to listen to a single address. When a frame is sent to a group address, all the stations in the group receive it.

Sending to a group of stations is called **multicasting**. Multicasting is more selective, but it involves group management to define which stations are in the group.

The special address consisting of all 1 bits is reserved for **broadcasting**. A frame containing all 1s in the destination field is accepted by all stations on the network.

Source address

It is 6 bytes long.

Source addresses are globally unique, assigned centrally by IEEE to ensure that no two stations anywhere in the world have the same address.

The idea is that any station can uniquely address any other station by just giving the right 48-bit number.

The first 3 bytes of the address field are used for an **OUI** (**Organizationally Unique Identifier**).Values for this field are assigned by IEEE and indicate a manufacturer.

Manufacturers are assigned blocks of 2^{24} addresses. The manufacturer assigns the last 3 bytes of the address and programs the complete address into the NIC before it is sold.

Type or Length

It tells whether the frame is Ethernet or IEEE 802.3.

Used for MUX/DEMUX.

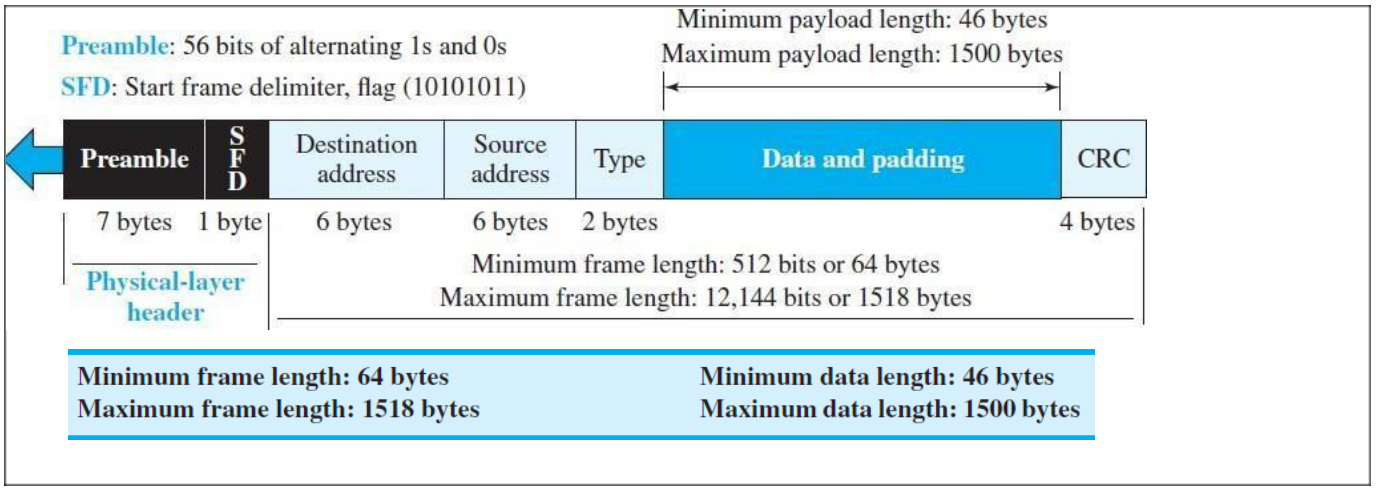
This field defines the upper-layer protocol whose packet is encapsulated in the frame. This protocol can be IP, ARP, OSPF, and so on.

Ethernet uses a Type field to tell the receiver what to do with the frame. (Multiple network-layer protocols may be in use at the same time on the same machine, so when an Ethernet frame arrives, the operating system has to know which one to hand the frame to.)

The *Type* field specifies which process to give the frame to.

Example: a type code of 0x0800 means that the data contains an IPv4 packet.

If the number less than or equal to 0x600 (1536) can be interpreted as Length, and any number greater than 0x600 can be interpreted as Type.



Data

This field carries data encapsulated from the upper-layer protocols.

It is up to 1500 bytes.

It is a minimum of 46 and a maximum of 1500 bytes.

If the data coming from the upper layer is more than 1500 bytes, it should be fragmented and encapsulated in more than one frame. If it is less than 46 bytes, it needs to be padded with extra 0s.

Pad

If data portion of a frame is less than 46 bytes, the *Pad* field is used to fill out the frame to the minimum size.

Checksum

The last field contains error detection information, in this case a CRC-32. The CRC is calculated over the addresses, types, and data field. If the receiver calculates the CRC and finds that it is not zero (corruption in transmission), it discards the frame.

❖ ETHERNET ADDRESSING

- *Each station on an Ethernet network (such as a PC, workstation, or printer) has its own network interface card (NIC). The NIC fits inside the station and provides the station with a link-layer address.*
- *The Ethernet address is 6 bytes (48 bits), normally written in hexadecimal notation, with a colon between the bytes.*

Example: the following shows an Ethernet MAC address:

4A:30:10:21:10:1A

Transmission of Address Bits

The transmission is left to right, byte by byte; however, for each byte, the least significant bit is sent first and the most significant bit is sent last.

Example : Show how the address 47:20:1B:2E:08:EE is sent out online?

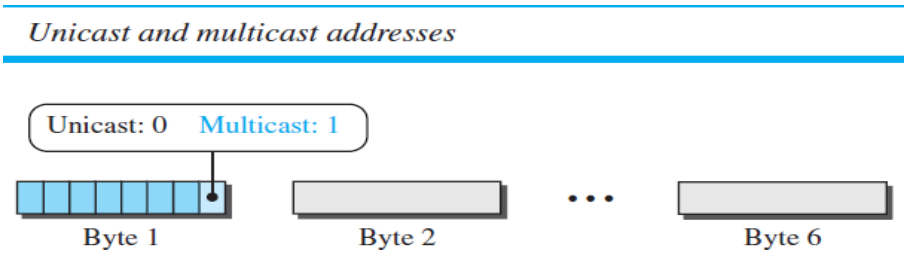
SOLUTION: The address is sent left to right, byte by byte; for each byte, it is sent right to left, bit by bit, as shown below:

Hexadecimal	47	20	1B	2E	08	EE
Binary	01000111	00100000	00011011	00101110	00001000	11101110
Transmitted ←	11100010	00000100	11011000	01110100	00010000	01110111

Unicast, Multicast, and Broadcast Addresses

- *A source address is always a unicast address*—the frame comes from only one station.
- *The destination address, however, can be unicast, multicast, or broadcast.*

Figure shows how to distinguish a unicast address from a multicast address. *If the least significant bit of the first byte in a destination address is 0, the address is unicast; otherwise, it is multicast.*



The broadcast address is a special case of the multicast address: the recipients are all the stations on the LAN. A broadcast destination address is forty-eight 1s.

Example: Define the type of the following destination addresses?

a. 4A:30:10:21:10:1A

b. 47:20:1B:2E:08:EE

c. FF: FF: FF: FF: FF: FF

Solution: To find the type of the address, we need to look at the second hexadecimal digit from the left. If it is even, the address is unicast. If it is odd, the address is multicast. If all digits are Fs, the address is broadcast.

Therefore, we have the following:

a. 4**A**:30:10:21:10:1A (This is a unicast address because A in binary is 1010 (even).)

b. 4**7**:20:1B:2E:08:EE (This is a multicast address because 7 in binary is 0111 (odd).)

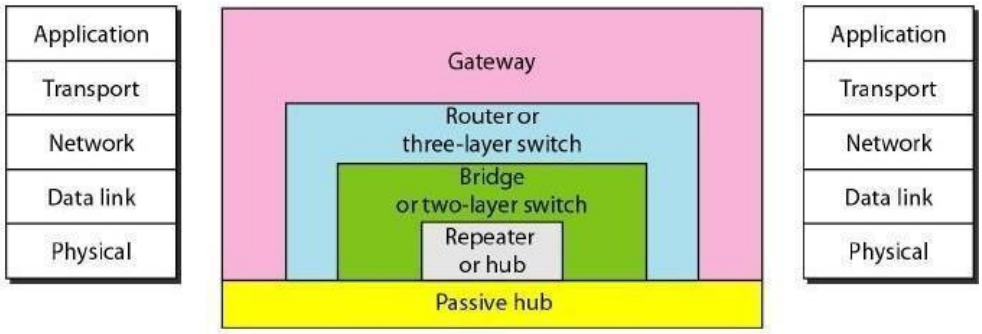
c. **FF: FF: FF: FF: FF: FF** (This is a broadcast address because all digits are Fs in hexadecimal.)

Distinguish Between Unicast, Multicast, and Broadcast Transmission

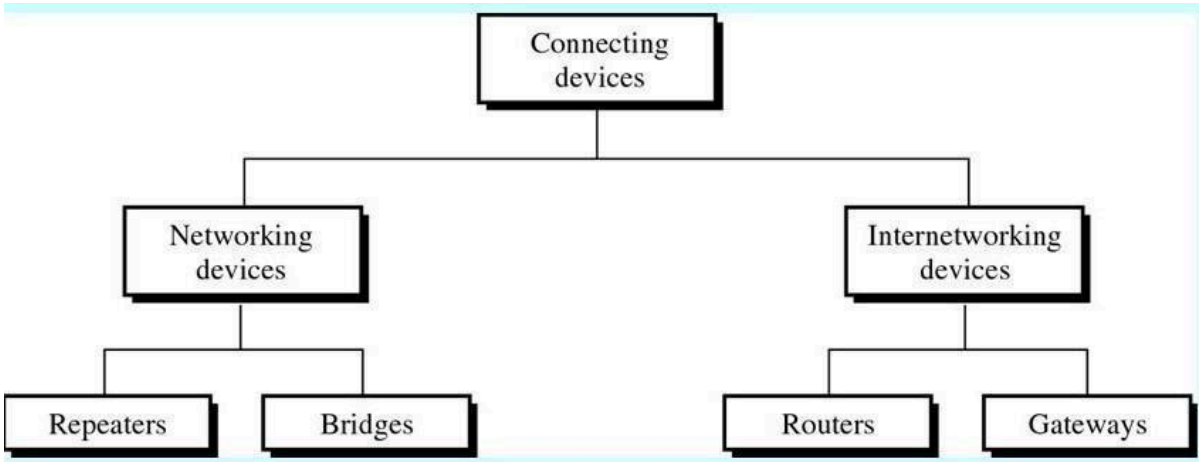
- In a unicast transmission, all stations will receive the frame, the intended recipient keeps and handles the frame; the rest discard it.
- In a multicast transmission, all stations will receive the frame, the stations that are members of the group keep and handle it; the rest discard it.
- In a broadcast transmission, all stations (except the sender) will receive the frame and all stations (except the sender) keep and handle it.

CONNECTING DEVICES (REPEATERS, HUBS, BRIDGES, SWITCHES, ROUTERS, GATEWAYS)

Hosts and networks do not normally operate in isolation. We use connecting devices to connect hosts together to make a network or to connect networks together to make an internet. Connecting devices can operate in different layers of the Internet model.



Five categories of connecting devices



TYPES OF CONNECTING DEVICES

1. Devices which operate below the physical layer.
   Example: Passive hub.
2. Devices which operate at the physical layer.
   Example: Repeater.
3. Devices which operate at the physical and data link layers.
   Example: Bridge.
4. Devices which operate at the physical layer, data link layer and network layer.
   Example: Router.
5. Devices which operate at all five layers.
   Example: Gateway.

❖ PASSIVE HUBS

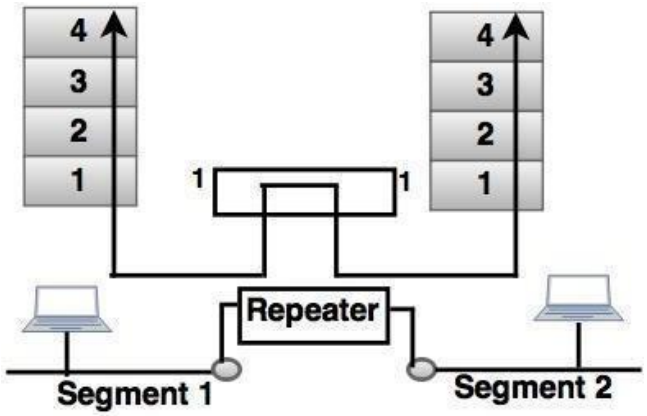
- *A passive hub is just a connector. It connects the wires coming from different branches.*
- *By using passive hub, each computer can receive the signal which is sent from all other computers connected in the hub.*

In a star-topology Ethernet LAN, a passive hub is just a point where the signals coming from different stations collide; the hub is the collision point. This type of a hub is part of the media; its location in the Internet model is below the physical layer.

REPEATERS

- *A repeater is a device that operates only in the physical layer.*
- *A repeater receives a signal and, before it becomes too weak or corrupted, regenerates the original bit pattern. The repeater then sends the refreshed signal.*
- *A repeater can extend the physical length of a LAN.*
- A repeater does not actually connect two LANs; *it connects two segments of the same LAN.* The segments connected are still part of one single LAN.
- The repeater acts as a two-port node, but operates only in the physical layer. When it receives a frame from any of the ports, it regenerates and forwards it to the other port.

A repeater has no filtering capability.



Repeater

Differences between repeater and amplifier

An amplifier cannot discriminate between the intended signal and noise; it amplifies equally everything fed into it.

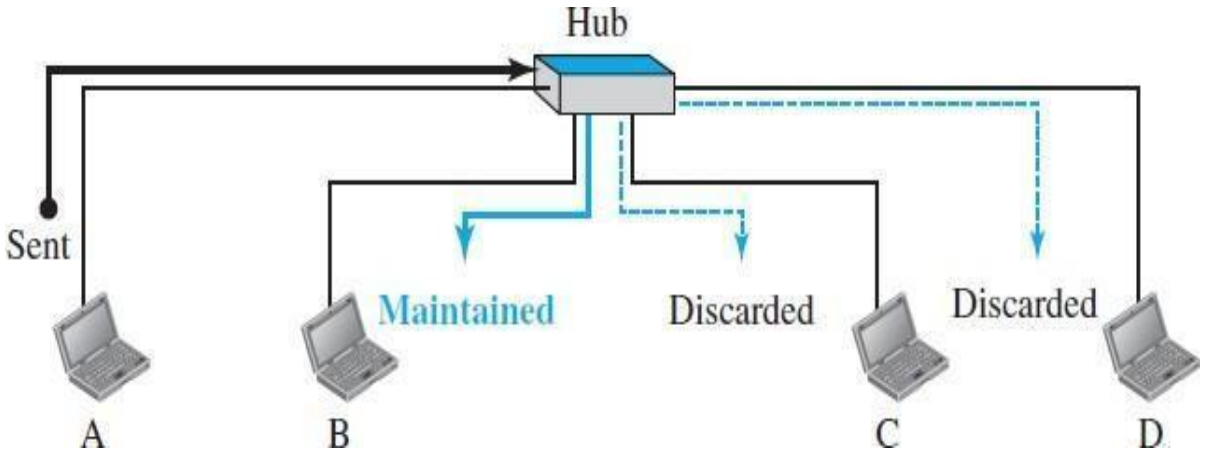
A repeater does not amplify the signal; it regenerates the signal. When it receives a weakened or corrupted signal, it creates a copy, bit for bit, at the original strength.

ACTIVE HUB

It operates in the physical layer.

It does not have a filtering capability. It does not have the intelligence to find from which port the frame should be sent out.

An active hub is actually a multipart repeater. It is normally used to create connections between stations



in a physical star topology.

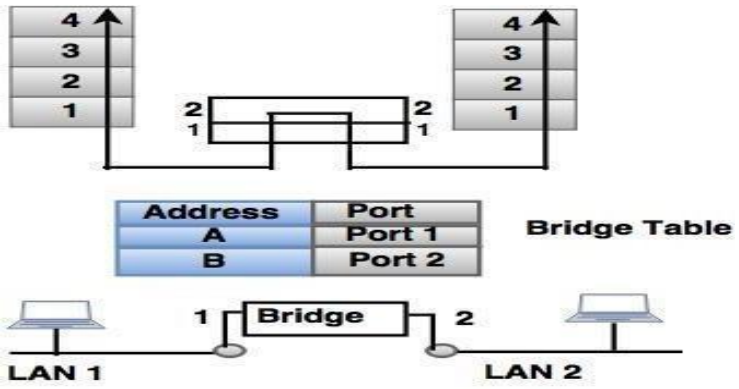
Figure shows that when a packet from station A to station B arrives at the hub, it forwards the packet to all outgoing ports except the one from which the signal was received. In other words, the frame is broadcast. All stations in the LAN receive the frame, but only station B keeps it. The rest of the stations discard it.

The figure definitely shows that a hub does not have a filtering capability; it does not have the intelligence to find from which port the frame should be sent out.

A hub or a repeater is a physical-layer device. They do not have a link-layer address and they do not check the link-layer address of the received frame. They just regenerate the corrupted bits and send them out from every port.

BRIDGES

- Bridge operates in both the physical and the data link layer.
- Bridge as a physical layer device, it regenerates the signal it receives.
- As a data link layer device, the bridge can check the physical (MAC) addresses (source and destination) contained in the frame.



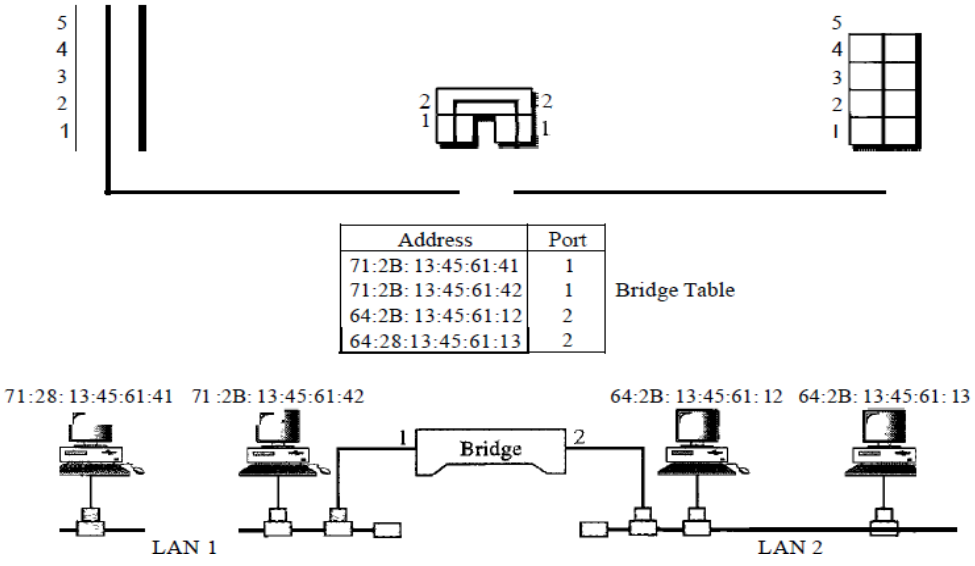
Bridge connecting Two LAN's

Filtering

The difference in functionality between a bridge and a repeater is a bridge has filtering capability. It can check the destination address of a frame and decide if the frame should be forwarded or dropped. If the frame is to be forwarded, the decision must specify the port. A bridge has a table that maps addresses to ports.

A bridge has a table used in filtering decisions.

Figure 15.5 *A bridge connecting two LANs*



In Figure 15.5, two LANs are connected by a bridge. If a frame destined for station 71:2B:13:45:61:42 arrives at port 1, the bridge consults its table to find the departing port. According to its table, frames for 71: 2B:13:45:61:42 leave through port 1; therefore, there is no need for forwarding, and the frame is dropped. On the other hand, if a frame for 71:2B:13:45:61:41 arrives at port 2, the departing port is port 1 and the frame is forwarded.

A bridge does not change the physical (MAC) addresses in a frame.

❖ TYPES OF BRIDGES

There are mainly three types in which bridges can be characterized:

Transparent Bridge: As the name signifies, it appears to be transparent for the other devices on the network. The other devices are ignorant of its existence. It only blocks or forwards the data as per the MAC address.

A transparent bridge is a bridge in which the stations are completely unaware of the bridge's existence. If a bridge is added or deleted from the system, reconfiguration of the stations is unnecessary. According to the IEEE 802.1 d specification, a system equipped with transparent bridges must meet three criteria:

1. Frames must be forwarded from one station to another.

2. The forwarding table is automatically made by learning frame movements in the network.

3. Loops in the system must be prevented.

Source Route Bridge: It derives its name from the fact that the path which packet takes through the network is implanted within the packet. It is mainly used in Token ring networks.

Translational Bridge: The process of conversion takes place via Translational Bridge. It converts the data format of one networking to another. For instance Token ring to Ethernet and vice versa.

SWITCHES (LINK-LAYER SWITCHES)

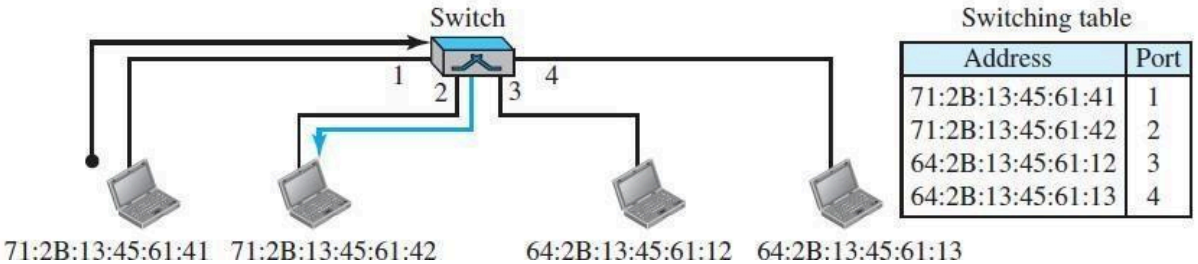
- A **link-layer switch** (or ***switch***) operates in both the physical and the data-link layers.
- As a physical-layer device, it regenerates the signal it receives. As a link-layer device, the link-layer switch can check the MAC addresses (source and destination) contained in the frame.

Filtering

One may ask what the difference in functionality is between a link-layer switch and a hub. A link-layer switch has **filtering** capability. It can check the destination address of a frame and can decide from which outgoing port the frame should be sent.

A link-layer switch has a table used in filtering decisions.

Figure 17.3 *Link-layer switch*



Switching table

Address	Port
71:2B:13:45:61:41	1
71:2B:13:45:61:42	2
64:2B:13:45:61:12	3
64:2B:13:45:61:13	4

In Figure 17.3, we have a LAN with four stations that are connected to a link-layer switch. If a frame destined for station 71:2B:13:45:61:42 arrives at port 1, the link-layer switch consults its table to find the departing port. According to its table, frames for 71:2B:13:45:61:42 should be sent out only through port 2; therefore, there is no need for forwarding the frame through other ports.

A link-layer switch does not change the link-layer (MAC) addresses in a frame.

Transparent Switches

A **transparent switch** is a switch in which the stations are completely unaware of the switch's existence. If a switch is added or deleted from the system, reconfiguration of the stations is unnecessary. According to the IEEE 802.1d specification, a system equipped with transparent switches must meet three criteria:

1. Frames must be forwarded from one station to another.
2. The forwarding table is automatically made by learning frame movements in the network.
3. Loops in the system must be prevented.

2 TYPES OF SWTCHES

Two- Layer Switch (@Datalink layer)

The two-layer switch performs at the physical and the data link layer.

It is a bridge with many ports and design allows faster performs.

A bridge is used to connect different LANs together.

The two- layer switch can make a filtering decision bases on the MAC address of the received frame. However, two- layer switch has a buffer which holds the frame for processing.

Three- Layer Switch (@Network Layer)

The three-layer switch is a router.

The switching fabric in a three-layer allows a faster table lookup and forwarding mechanism.

Advantages of Switches:

A link-layer switch has several advantages over a hub. We discuss only two of them here.

Collision Elimination: link-layer switch eliminates the collision. This means increasing the average bandwidth available to a host in the network. In a switched LAN, there is no need for carrier sensing and collision detection; each host can transmit at any time.
Connecting Heterogeneous Devices: A link-layer switch can connect devices that use different protocols at the physical layer (data rates) and different transmission media.

Spanning Tree Algorithm

To solve the looping problem, the IEEE specification requires that switches use the spanning tree algorithm to create a loop less topology.
Spanning tree: It is a graph in which there is no loop. In a switched LAN, this means creating a topology in which each LAN can be reached from any other LAN through one path only (no loop).

❖ ROUTERS

- A **router** is a three-layer device. It operates in the physical, data-link, and network layers.
- As a physical-layer device, it regenerates the signal it receives.

- As a link-layer device, the router checks the physical addresses (source and destination) contained in the packet.
- As a network-layer device, a router checks the network-layer addresses.
- *Routers route the packets based on their logical addresses (host-to-host addressing).*

A router connects the LANs and WANs on the internet

- *Router has a routing table, which is used to make decision on selecting the route.*
- *The key function of the router is to determine the shortest path to the destination.*

A router is a three-layer (physical, data-link, and network) device.

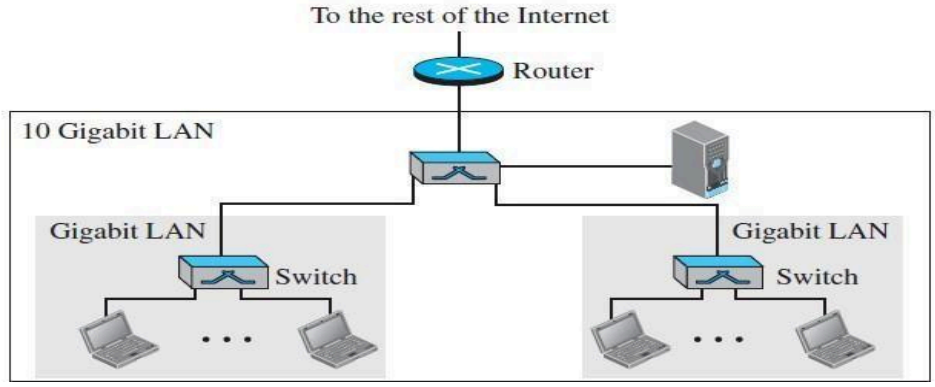
A router can connect networks. In other words, a router is an internetworking device; it connects independent networks to form an internetwork.

According to this definition, two networks connected by a router become an internetwork or an internet. There are three major differences between a router and a repeater or a switch.

1. A router has a physical and logical (IP) address for each of its interfaces.
2. A router acts only on those packets in which the link-layer destination address matches the address of the interface at which the packet arrives.
3. A router changes the link-layer address of the packet (both source and destination) when it forwards the packet.

In the below Figure, assume an organization has two separate buildings with a Gigabit Ethernet LAN installed in each building. The organization uses switches in each LAN. The two LANs can be connected to form a larger LAN using 10 Gigabit Ethernet technology that speeds up the connection to the Ethernet and the connection to the organization server. A router then can connect the whole system to the Internet.

.9 Routing example



A router will change the MAC addresses it receives because the MAC addresses have only local jurisdictions.

A router changes the link-layer addresses in a packet.

GATEWAY

- The Gateway devices work in the Transport layer and above, where the different network technologies are implemented. A gateway is necessary when there are different technologies implemented by the different LAN's which are to be connected together.
- *A gateway is a computer, which operates in all five layers of the internet or seven layers of OSI model.*
- Gateway connects two independent networks.
- *A gateway accepts a packet formatted for one protocol (for example, TCP/IP) and converts it to a packet formatted to another protocol (for example, Apple Talk) before forwarding it.*
- The gateway must adjust the data rate, size and data format.

BROUTER

It is also known as bridging router is a device which combines features of both bridge and router. It can work either at data link layer or at network layer. Working as router, it is capable of routing packets across networks and working as bridge, it is capable of filtering local area network traffic.

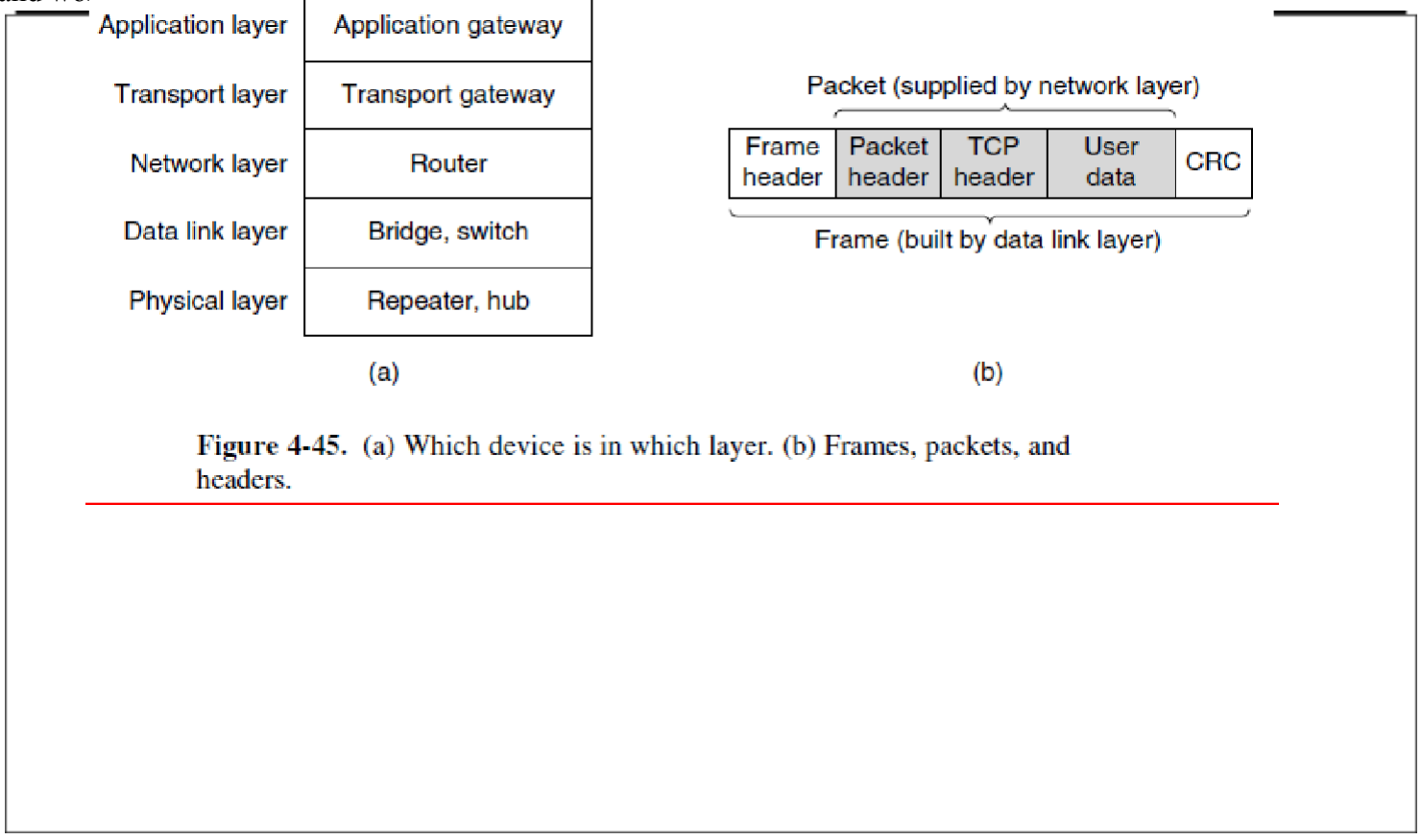


Figure 4-45. (a) Which device is in which layer. (b) Frames, packets, and headers.

HOW CONNECTING DEVICES OPERATE IN DIFFERENT LAYERS

As illustrated in Fig. 4-45(a), the user generates some data to be sent to a remote machine. Those data are passed to the transport layer, which then adds a header (for example, a TCP header) and passes the resulting unit down to the network layer. The network layer adds its own header to form a network layer packet (e.g., an IP packet). In Fig. 4-45(b), we see the IP packet shaded in gray. Then the packet goes to the data link layer, which adds its own header and checksum (CRC) and gives the resulting frame to the physical layer for transmission, for example, over a LAN.

❖ DATA LINK LAYER SWITCHING

Many organizations have multiple LANs and wish to connect them. The connections are made with devices called **bridges**.

Bridges operate in the data link layer, so they examine the data link layer addresses to forward frames. Since they are not supposed to examine the payload field of the frames they forward, they can handle IP packets as well as other kinds of packets, such as AppleTalk packets.

In contrast, *routers* examine the addresses in packets and route based on them, so they only work with the protocols that they were designed to handle.

USES OF BRIDGES

- First, many university and corporate departments have their own LANs to connect their own personal computers, servers, and devices such as printers. Since the goals of the various departments differ, different departments may set up different LANs, without regard to what other departments are doing. Sooner or later, though, there is a need for interaction, so bridges are needed.
- Second, the organization may be geographically spread over several buildings separated by considerable distances. It may be cheaper to have separate LANs in each building and connect them with bridges.
- Third, it may be necessary to split what is logically a single LAN into separate LANs (connected by bridges) to accommodate the load.
- Security
- Reliability

LEARNING BRIDGES

The topology of two LANs bridged together is shown in Fig. 4-41 for two cases.

- On the left-hand side, two multidrop LANs, such as classic Ethernets, are joined by a special station—the bridge—that sits on both LANs.
- On the right-hand side, LANs with point-to-point cables, including one hub, are joined together. The bridges are the devices to which the stations and hub are attached. (If the LAN technology is Ethernet, the bridges are better known as Ethernet switches.)

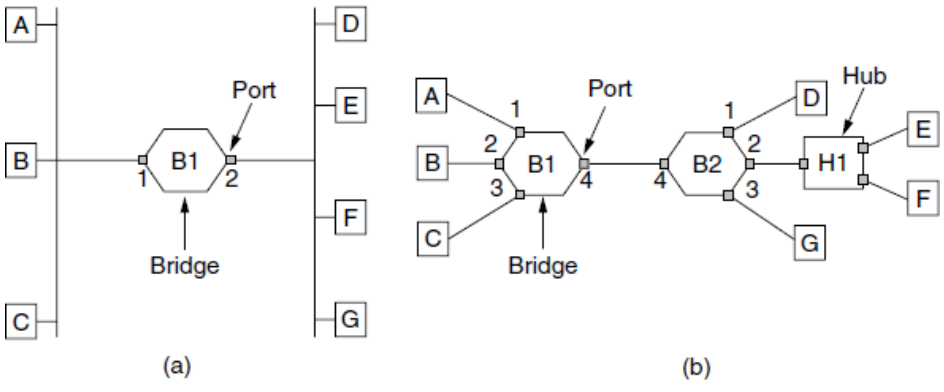


Figure 4-41. (a) Bridge connecting two multidrop LANs. (b) Bridges (and a hub) connecting seven point-to-point stations.

All of the stations attached to the same port on a bridge belong to the same collision domain, and this is different than the collision domain for other ports. *If there is more than one station, as in a classic Ethernet, a hub, or a half-duplex link, the CSMA/CD protocol is used to send frames.*

To bridge multidrop LANs, a bridge is added as a new station on each of the multidrop LANs, as in Fig. 4-41(a). To bridge point-to-point LANs, the hubs are either connected to a bridge or, preferably, replaced with a bridge to increase performance. In Fig. 4-41(b), bridges have replaced all but one hub. Different kinds of cables can also be attached to one bridge.

Example: The cable connecting bridge *B1* to bridge *B2* in Fig. 4-41(b) might be a long-distance fiber optic link, while the cable connecting the bridges to stations might be a short-haul twisted-pair line.

What happens inside the bridges?*

Each bridge operates in promiscuous, i.e. it accepts every frame transmitted by the stations attached to each of its ports. The bridge must decide whether to forward or discard each frame, and, if the former, on which port to output the frame. This decision is made by using the destination address.

Example:

1. **Consider the topology of Fig. 4-41(a).**

   If station *A* sends a frame to station *B*, bridge *B1* will receive the frame on port 1. This frame can be immediately discarded without further ado (trouble or difficulty) because it is already on the correct port.

2. **Consider the topology of Fig. 4-41(b)**

   Suppose that A sends a frame to *D*. Bridge *B1* will receive the frame on port 1 and output it on port 4. Bridge *B2* will then receive the frame on its port 4 and output it on its port 1.

A simple way to implement this scheme is to have a big (hash) table inside the bridge. The table can list each possible destination and which output port it belongs on.

Example: in Fig. 4-41(b), the table at *B1* would list *D* as belonging to port 4, since all *B1* has to know is which port to put frames on to reach *D*.

FLOODING

- When the bridges are first plugged in, all the hash tables are empty. None of the bridges know where any of the destinations are, so they use a flooding algorithm.
- Every incoming frame for an unknown destination is output on all the ports to which the bridge is connected except the one it arrived on.
- As time goes on, the bridges learn where destinations are. Once a destination is known, frames destined for it are put only on the proper port; they are not flooded.

BACKWARD LEARNING

The algorithm used by the bridges is backward learning. As mentioned above, the bridges operate in promiscuous mode, so they see every frame sent on any of their ports. By looking at the source addresses, they can tell which machines are accessible on which ports.

Example: if bridge *B1* in Fig. 4-41(b) sees a frame on port 3 coming from *C*, it knows that *C* must be reachable via port 3, so it makes an entry in its hash table. Any subsequent frame addressed to *C* coming in to *B1* on any other port will be forwarded to port 3.

The topology can change as machines and bridges are powered up and down and moved around**. To handle dynamic topologies, whenever a hash table entry is made, the arrival time of the frame is noted in the entry.** Whenever a frame whose source is already in the table arrives, its entry is updated with the current time. Thus, the time associated with every entry tells the last time a frame from that machine was seen.

ROUTING (BRIDGE)

The routing procedure for an incoming frame depends on the port it arrives on (the source port) and the address to which it is destined (the destination address). The procedure is as follows.

1. If the port for the destination address is the same as the source port, discard the frame.
2. If the port for the destination address and the source port are different, forward the frame on to the destination port.
3. If the destination port is unknown, use flooding and send the frame on all ports except the source port.

PROTOCOL PROCESSING AT BRIDGE

Bridges only look at the MAC addresses to decide how to forward frames; it is possible to start forwarding as soon as the destination header field has come in, before the rest of the frame has arrived (provided the output line is available, of course). This design reduces the latency of passing through the bridge, as well as the number of frames that the bridge must be able to buffer. It is referred to as **cut-through switching** or **wormhole routing** and is usually handled in hardware.

We can look at the operation of a bridge in terms of protocol stacks to understand what it means to be a link layer device. Consider a frame sent from station *A* to station *D* in the configuration of Fig. 4-41(a), in which the LANs are Ethernet. The frame will pass through one bridge. The protocol stack view of processing is shown in Fig. 4-42.

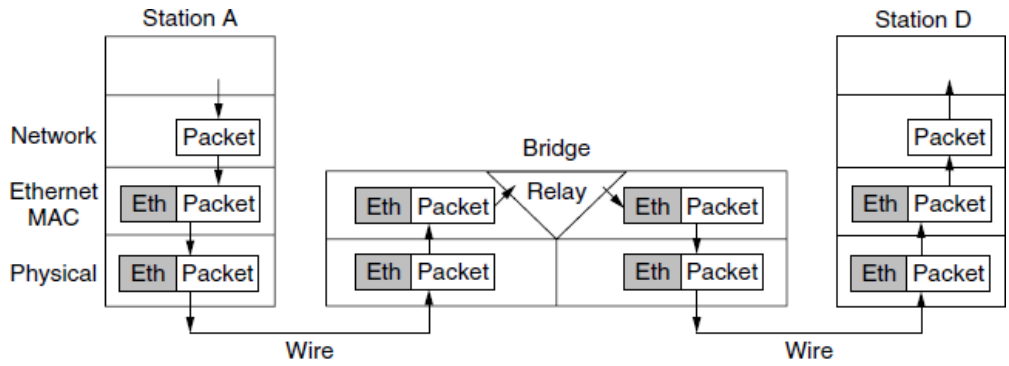


Figure 4-42. Protocol processing at a bridge.

The packet comes from a higher layer and descends into the Ethernet MAC layer. It acquires an Ethernet header (and also a trailer, not shown in the figure). This unit is passed to the physical layer, goes out over the cable, and is picked up by the bridge.

In the bridge, the frame is passed up from the physical layer to the Ethernet MAC layer. This layer has extended processing compared to the Ethernet MAC layer at a station. It passes the frame to a relay, still within the MAC layer. The bridge relay function uses only the Ethernet MAC header to determine how to handle the frame. In this case, it passes the frame to the Ethernet MAC layer of the port used to reach station *D*, and the frame continues on its way. In the general case, relays at a given layer can rewrite the headers.

SPANNING TREE BRIDGES

To increase reliability, redundant links can be used between bridges. In the example of Fig. 4-43, there are two links in parallel between a pair of bridges. This design ensures that if one link is cut, the network will not be partitioned into two sets of computers that cannot talk to each other.

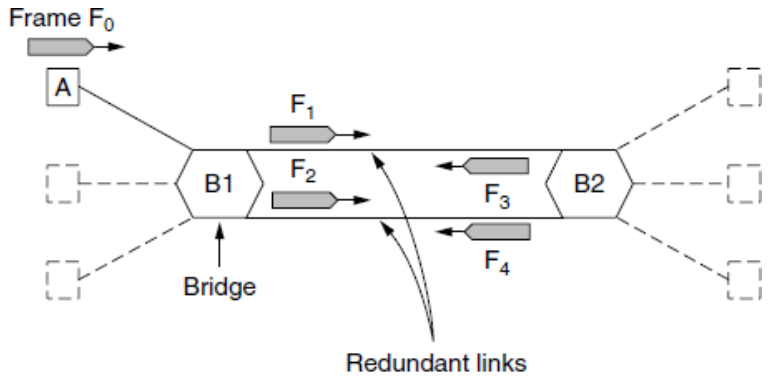


Figure 4-43. Bridges with two parallel links.

However, this redundancy introduces some additional problems because it creates loops in the topology. An example of these problems can be seen by looking at how a frame sent by *A* to a previously unobserved destination is handled in Fig. 4-43.

Each bridge follows the normal rule for handling unknown destinations, which is to flood the frame. Call the frame from *A* that reaches bridge *B1* frame F_0. The bridge sends copies of this frame out all of its other ports. We will only consider the bridge ports that connect *B1* to *B2* (though the frame will be sent out the other ports, too). Since there are two links from *B1* to *B2*, two copies of the frame will reach *B2*. They are shown in Fig. 4-43 as *F*1 and *F*2. Shortly thereafter, bridge *B2* receives these frames. However, it does not (and cannot) know that they are copies of the same frame, rather than two different frames sent one after the other. So bridge *B2* takes *F*1 and sends copies of it out all the other ports, and it also takes *F*2 and sends copies of it out all the other ports. This produces frames *F*3 and *F*4 that are sent along the two links back to *B1*. Bridge *B1* then sees two new frames with unknown destinations and copies them again. This cycle goes on forever.

The solution to this difficulty is for the bridges to communicate with each other and overlay the actual topology with a spanning tree that reaches every bridge.

Example: in Fig. 4-44 we see five bridges that are interconnected and also have stations connected to them. Each station connects to only one bridge. There are some redundant connections between the bridges so that frames will be forwarded in loops if all of the links are used.

This topology can be thought of as a graph in which the bridges are the nodes and the point-to-point links are the edges. The graph can be reduced to a spanning tree, which has no cycles by definition, by dropping the links shown as dashed lines in Fig. 4-44.

Using this spanning tree, there is exactly one path from every station to every other station. Once the bridges have agreed on the spanning tree, all forwarding between stations follows the spanning tree. Since there is a unique path from each source to each destination, loops are impossible.

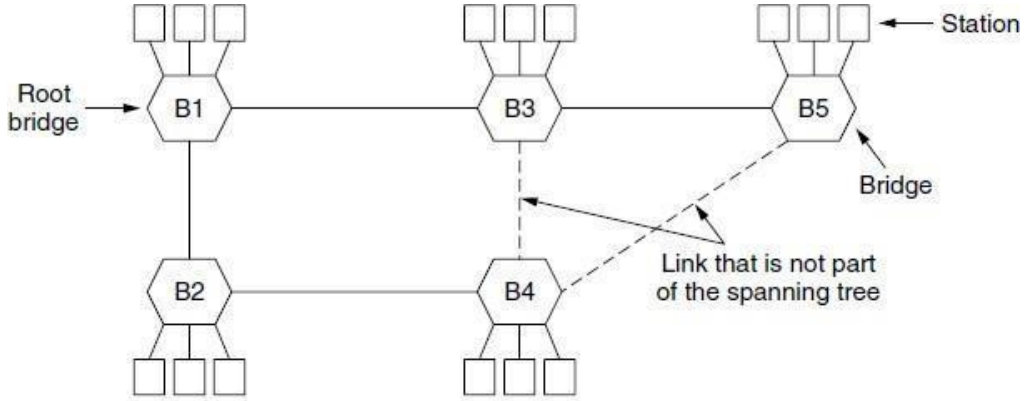


Figure 4-44. A spanning tree connecting five bridges. The dashed lines are links that are not part of the spanning tree.

Figure 4-44. A spanning tree connecting five bridges. The dashed lines are links that are not part of the spanning tree. To build the spanning tree, the bridges run a distributed algorithm.

Each bridge periodically broadcasts a configuration message out all of its ports to its neighbors and processes the messages it receives from other bridges. These messages are not forwarded, since their purpose is to build the tree, which can then be used for forwarding. The bridges must first choose one bridge to be the root of the spanning tree. To make this choice, they each include an identifier based on their MAC address in the configuration message, as well as the identifier of the bridge they believe to be the root. MAC addresses are installed by the manufacturer and guaranteed to be unique worldwide, which makes these identifiers convenient and unique. The bridges choose the bridge with the lowest identifier to be the root. After enough messages have been exchanged to spread the news, all bridges will agree on which bridge is the root.

In Fig. 4-44, bridge *B1* has the lowest identifier and becomes the root. Next, a tree of shortest paths from the root to every bridge is constructed. In Fig. 4-44, bridges *B2* and *B3* can each be reached from bridge *B1* directly, in one hop that is a shortest path. Bridge *B4* can be reached in two hops, via either *B2* or *B3*. To break this tie, the path via the bridge with the lowest identifier is chosen, so *B4* is reached via *B2*. Bridge *B5* can be reached in two hops via *B3*.

To find these shortest paths, bridges include the distance from the root in their configuration messages. Each bridge remembers the shortest path it finds to the root. The bridges then turn off ports that are not part of the shortest path. Although the tree spans all the bridges, not all the links (or even bridges) are necessarily present in the tree. This happens because turning off the ports prunes some links from the network to prevent loops. Even after the spanning tree has been established, the algorithm continues to run during normal operation to automatically detect topology changes and update the tree.

The algorithm for constructing the spanning tree was invented by Radia Perlman.

UNIT – 3 NETWORK LAYER

NETWORK LAYER DESIGN ISSUES

In the following sections we will provide an introduction to some of the issues that the designers of the network layer must grapple with. These issues include the service provided to the transport layer and the internal design of the subnet.

STORE-AND-FORWARD PACKET SWITCHING

The network layer protocols operation can be seen in Fig. 5-1.

The major components of the system are the carrier's equipment (routers connected by transmission lines), shown inside the shaded oval.

The customers' equipment, shown outside the oval. Host *H1* is directly connected to one of the carrier's routers, *A*, by a leased line. In contrast, *H2* is on a LAN with a router, *F*, owned and operated by the customer. This router also has a leased line to the carrier's equipment.

We have shown *F* as being outside the oval because it does not belong to the carrier, but in terms of construction, software, and protocols, it is probably no different from the carrier's routers. Whether it belongs to the subnet is arguable, but for the purposes of this chapter, routers on customer premises are considered part of the subnet.

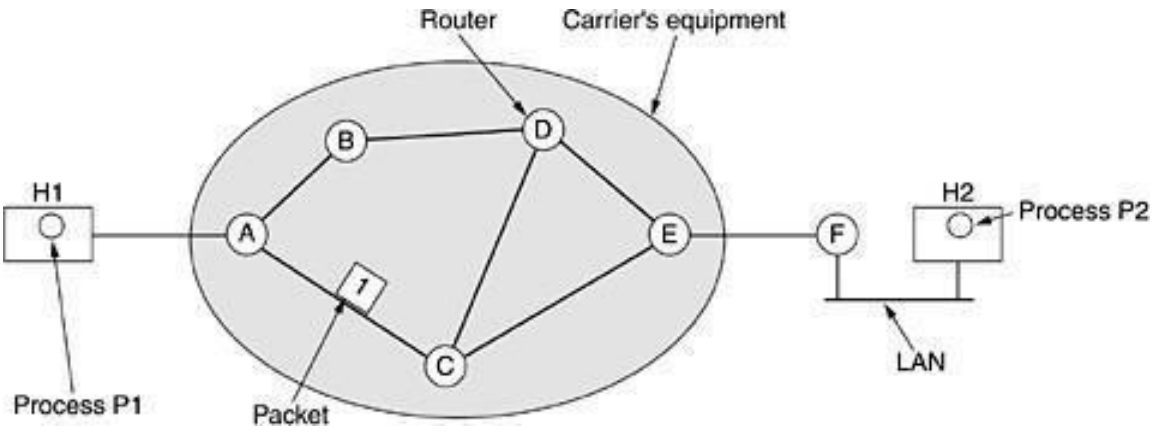


Figure 5-1. The environment of the network layer protocols.

A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-point link to the carrier.

The packet is stored there until it has fully arrived so the checksum can be verified. Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered.

This mechanism is **store-and-forward packet switching**.

SERVICES PROVIDED TO THE TRANSPORT LAYER

The network layer provides services to the transport layer at the network layer/transport layer interface. The network layer services have been designed with the following goals in mind.

1. The services should be independent of the router technology.
2. The transport layer should be shielded from the number, type, and topology of the routers present.
3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

There is a discussion centers on whether the network layer should provide connection-oriented service or connectionless service.

In their view (based on 30 years of actual experience with a real, working computer network), the subnet is inherently unreliable, no matter how it is designed. Therefore, the hosts should accept the fact that the network is unreliable and do error control (i.e., error detection and correction) and flow control themselves.

This viewpoint leads quickly to the conclusion that the network service should be connectionless, with primitives SEND PACKET and RECEIVE PACKET and little else.

In particular, no packet ordering and flow control should be done, because the hosts are going to do that anyway, and there is usually little to be gained by doing it twice.

Furthermore, each packet must carry the full destination address, because each packet sent is carried independently of its predecessors, if any.

The other camp (represented by the telephone companies) argues that the subnet should provide a reliable, connection-oriented service.

These two camps are best exemplified by the Internet and ATM. The Internet offers connectionless network-layer service; ATM networks offer connection-oriented network-layer service. However, it is interesting to note that as quality-of-service guarantees are becoming more and more important, the Internet is evolving. In particular, it is starting to acquire properties normally associated with connection-oriented service, as we will see later.

IMPLEMENTATION OF CONNECTIONLESS SERVICE

Two different organizations are possible, depending on the type of service offered.

If connectionless service is offered, packets are injected into the subnet individually and routed independently of each other. No advance setup is needed. In this context, the packets are frequently called **datagrams** (in analogy with telegrams) and the subnet is called a

datagram subnet.

If connection-oriented service is used, a path from the source router to the destination router must be established before any data packets can be sent. This connection is called a **VC** (**virtual circuit**), in analogy with the physical circuits set up by the telephone system, and the subnet is called a **virtual-circuit subnet**.

Let us now see how a datagram subnet works. Suppose that the process *P1* in **Fig. 5-2** has

a long message for *P2*. It hands the message to the transport layer with instructions to deliver it to process *P2* on host *H2*. The transport layer code runs on *H1*, typically within the operating system. It prepends a transport header to the front of the message and hands the result to the network layer, probably just another procedure within the operating system.

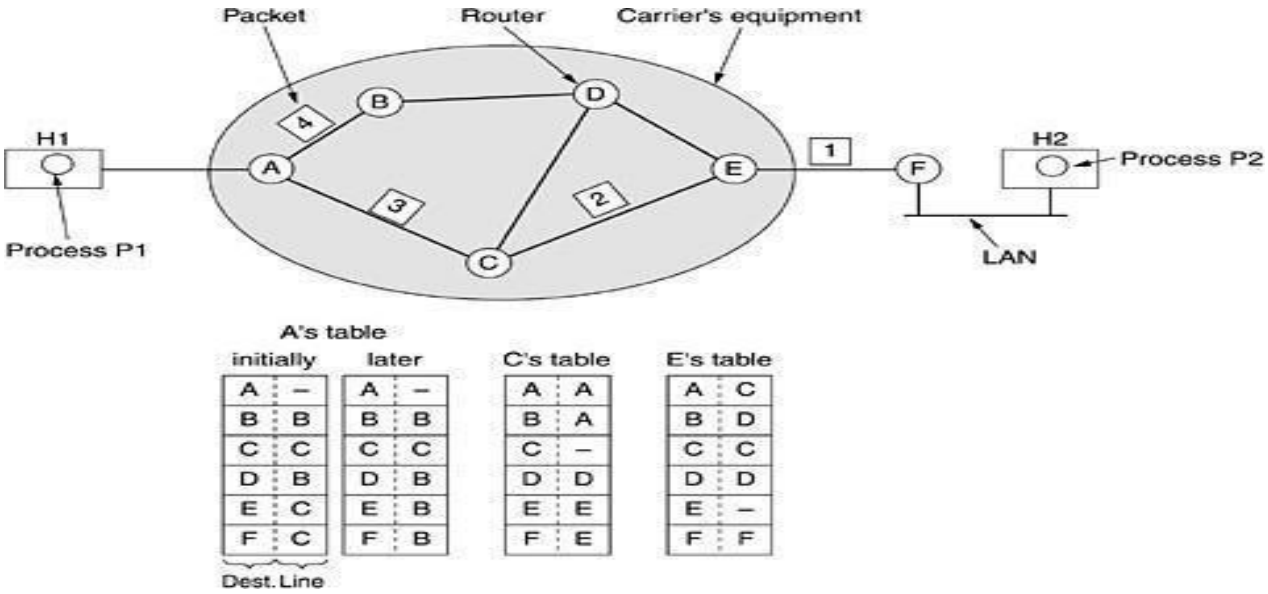


Figure 5-2. Routing within a datagram subnet

Let us assume that the message is four times longer than the maximum packet size, so the network layer has to break it into four packets, 1, 2, 3, and 4 and sends each of them in turn to router *A* using some point-to-point protocol,

For example, PPP. At this point the carrier takes over. Every router has an internal table telling it where to send packets for each possible destination. Each table entry is a pair consisting of a destination and the outgoing line to use for that destination. Only directly-connected lines can be used.

For example, in Fig. 5-2, *A* has only two outgoing lines—to *B* and *C*—so every incoming packet must be sent to one of these routers, even if the ultimate destination is some other router. *A*'s initial routing table is shown in the figure under the label "initially". As they arrived at *A*, packets 1, 2, and 3 were stored briefly (to verify their checksums). Then each was forwarded to *C* according to *A*'s table. Packet 1 was then forwarded to *E* and then to *F*. When it got to *F*, it was encapsulated in a data link layer frame and sent to *H2* over the LAN. Packets 2 and 3 follow the same route.

However, something different happened to packet 4. When it got to *A* it was sent to router *B*, even though it is also destined for *F*. For some reason, *A* decided to send packet 4 via a different route than that of the first three. Perhaps it learned of a traffic jam somewhere along the *ACE* path and updated its routing table, as shown under the label "later."

The algorithm that manages the tables and makes the routing decisions is called the **routing algorithm**.

IMPLEMENTATION OF CONNECTION-ORIENTED SERVICE

For connection-oriented service, we need a virtual-circuit subnet.

Let us see how that works.

The idea behind virtual circuits is to avoid having to choose a new route for every packet sent, as in Fig. 5-2. Instead, when a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers.

That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works.

When the connection is released, the virtual circuit is also terminated. With connection-oriented service, each packet carries an identifier telling which virtual circuit it belongs to.

As an example, consider the situation of **Fig. 5-3**. Here, host *H1* has established connection 1 with host *H2*. It is remembered as the first entry in each of the routing tables. The first line of *A*'s table says that if a packet bearing connection identifier 1 comes in from *H1*, it is to be sent to router *C* and given connection identifier 1. Similarly, the first entry at *C* routes the packet to *E*, also with connection identifier 1.

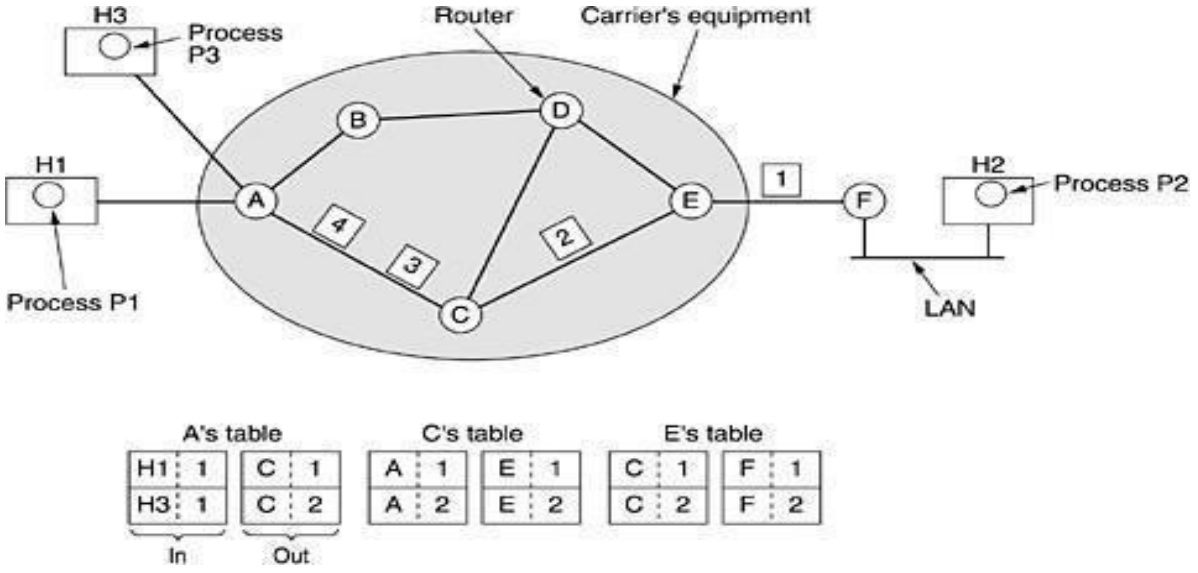


Figure 5-3. Routing within a virtual-circuit subnet.

Now let us consider what happens if *H3* also wants to establish a connection to *H2*. It chooses connection identifier 1 (because it is initiating the connection and this is its only connection) and tells the subnet to establish the virtual circuit. This leads to the second row in the tables. Note that we have a conflict here because although *A* can easily distinguish connection 1 packets from *H1* from connection 1 packets from *H3*, *C* cannot do this. For this reason, *A* assigns a different connection identifier to the outgoing traffic for the second connection. Avoiding conflicts of this kind is why routers need the ability to replace connection

identifiers in outgoing packets. In some contexts, this is called label switching.

COMPARISON OF VIRTUAL-CIRCUIT AND DATAGRAM SUBNETS

The major issues are listed in Fig. 5-4, although purists could probably find a counter example for everything in the figure.

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Figure 5-4. Comparison of datagram and virtual-circuit subnets.

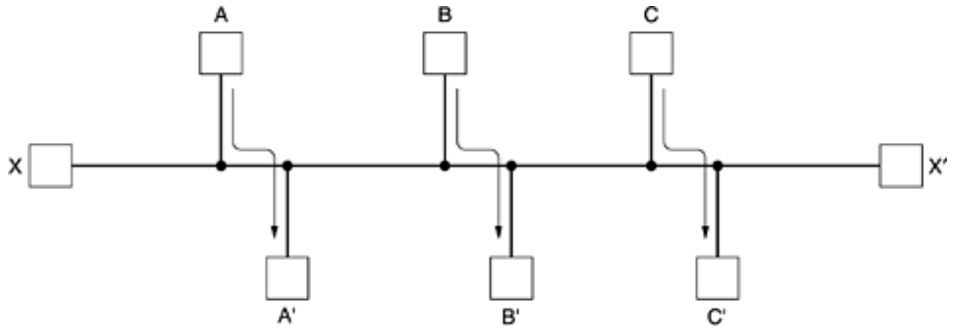
ROUTING ALGORITHMS

The **routing algorithm** is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on.

PROPERTIES OF ROUTING ALGORITHM:

Correctness, simplicity, robustness, stability, fairness, and optimality

FAIRNESS AND OPTIMALITY.



Fairness and optimality may sound obvious, but as it turns out, they are often contradictory goals. There is enough traffic between A and A', between B and B', and between C and C' to saturate the horizontal links. To maximize the total flow, the X to X' traffic should be shut off altogether. Unfortunately, X and X' may not see it that way. Evidently, some compromise between global efficiency and fairness to individual connections is needed.

CATEGORY OF ALGORITHM

Routing algorithms can be grouped into two major classes: **nonadaptive and adaptive.**

Nonadaptive algorithms do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from I to J is computed in advance, off-line, and downloaded to the routers when the network is booted.

This procedure is sometimes called **Static routing**.

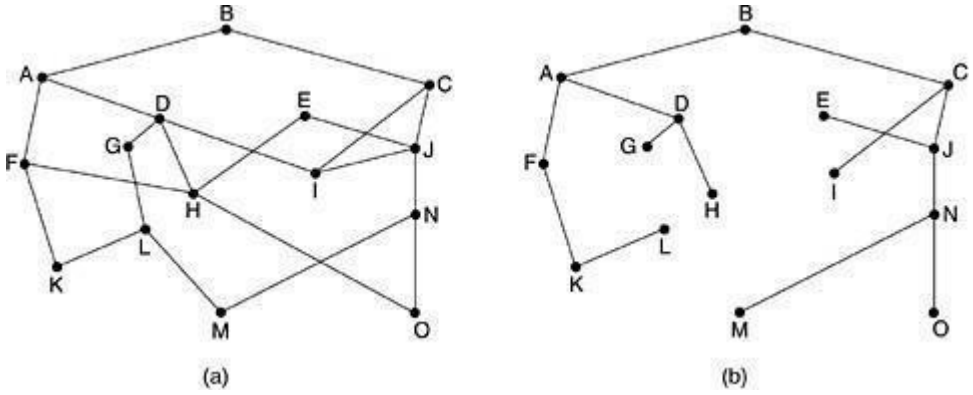
Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well

This procedure is sometimes called **dynamic routing**

THE OPTIMALITY PRINCIPLE

(a) **I**f router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.

(b) The set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a sink tree.



***(c) A subnet. (b) A sink tree for router* B.**

As a direct consequence of the optimality principle, we can see that the set of optimal routes from all sources to a given destination form a tree rooted at the destination.

Such a tree is called a **sink tree** where the distance metric is the number of hops. Note that a sink tree is not necessarily unique; other trees with the same path lengths may exist.

The goal of all routing algorithms is to discover and use the sink trees for all routers.

SHORTEST PATH ROUTING

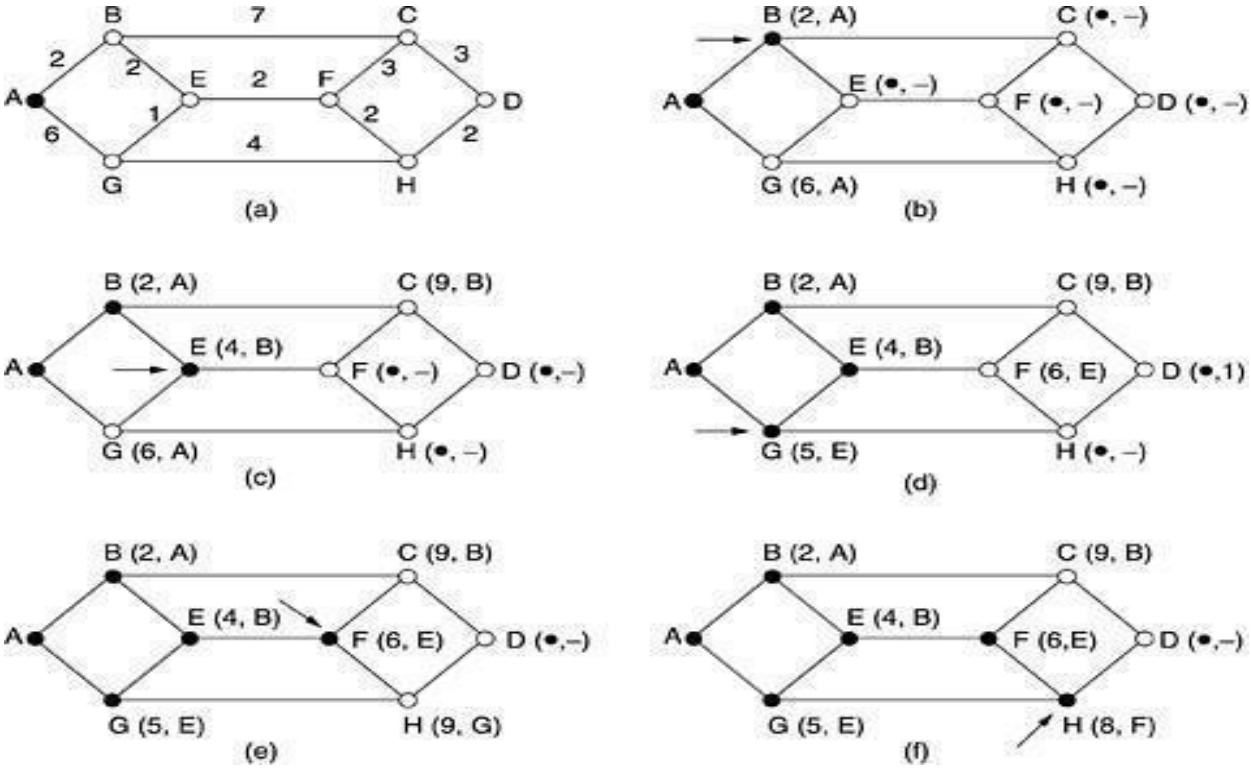
- A technique to study routing algorithms: The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line (often called a link).

- To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.
- One way of measuring path length is the number of hops. Another metric is the geographic distance in kilometers. Many other metrics are also possible. For example,

each arc could be labeled with the mean queuing and transmission delay for some standard test packet as determined by hourly test runs.

- In the general case, the labels on the arcs could be computed as a function of the distance, bandwidth, average traffic, communication cost, mean queue length, measured

delay, and other factors. By changing the weighting function, the algorithm would then compute the "shortest" path measured according to any one of a number of criteria or to a combination of criteria.



The first five steps used in computing the shortest path from* A *to* D. *The arrows indicate the working node.

To illustrate how the labelling algorithm works, look at the weighted, undirected graph of Fig. 5-7(a), where the weights represent, for example, distance.

We want to find the shortest path from *A* to *D*. We start out by marking node *A* as permanent, indicated by a filled-in circle.

Then we examine, in turn, each of the nodes adjacent to *A* (the working node), relabeling each one with the distance to *A*.

Whenever a node is relabelled, we also label it with the node from which the probe was made so that we can reconstruct the final path later.

Having examined each of the nodes adjacent to *A*, we examine all the tentatively labelled nodes in the whole graph and make the one with the smallest label permanent, as shown in Fig. 5-7(b).

This one becomes the new working node.

We now start at B and examine all nodes adjacent to it. If the sum of the label on B and the distance from B to the node being considered is less than the label on that node, we have a shorter path, so the node is relabelled.

After all the nodes adjacent to the working node have been inspected and the tentative labels changed if possible, the entire graph is searched for the tentatively-labelled node with the smallest value. This node is made permanent and becomes the working node for the next round. Figure 5-7 shows the first five steps of the algorithm.

To see why the algorithm works, look at Fig. 5-7(c). At that point we have just made *E* permanent. Suppose that there were a shorter path than *ABE*, say *AXYZE*. There are two possibilities: either node *Z* has already been made permanent, or it has not been. If it has, then *E* has already been probed (on the round following the one when *Z* was made permanent), so the *AXYZE* path has not escaped our attention and thus cannot be a shorter path.

Now consider the case where *Z* is still tentatively labelled. Either the label at *Z* is greater than or equal to that at *E*, in which case *AXYZE* cannot be a shorter path than *ABE*, or it is less than that of *E*, in which case *Z* and not *E* will become permanent first, allowing *E* to be probed from *Z*.

This algorithm is given in Fig. 5-8. The global variables *n* and *dist* describe the graph and are initialized before *shortest path* is called. The only difference between the program and the algorithm described above is that in Fig. 5-8, we compute the shortest path starting at the terminal node, *t*, rather than at the source node, *s*. Since the shortest path from *t* to *s* in an undirected graph is the same as the shortest path from *s* to *t*, it does not matter at which end we begin (unless there are several shortest paths, in which case reversing the search might discover a different one). The reason for searching backward is that each node is labelled with its predecessor rather than its successor. When the final path is copied into the output variable, *path*, the path is thus reversed. By reversing the search, the two effects cancel, and the answer is produced in the correct order.

CONGESTION CONTROL ALGORITHMS

When too many packets are present in (a part of) the subnet, performance degrades. This situation is called **congestion**.

Figure 5-25 depicts the symptom. When the number of packets dumped into the subnet by the hosts is within its carrying capacity, they are all delivered (except for a few that are afflicted with transmission errors) and the number delivered is proportional to the number sent.

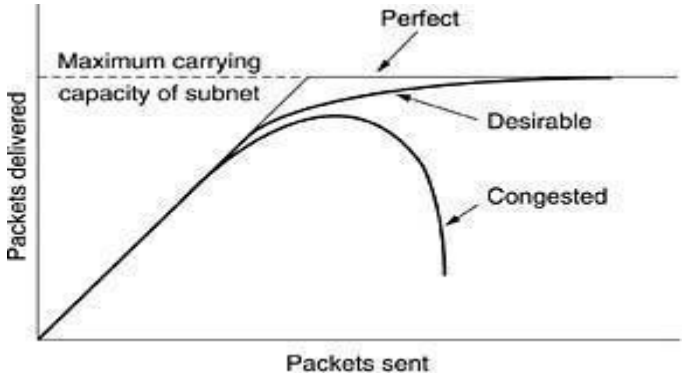


Figure 5-25. When too much traffic is offered, congestion sets in and performance degrades sharply.

However, as traffic increases too far, the routers are no longer able to cope and they begin losing packets. This tends to make matters worse. At very high traffic, performance collapses completely and almost no packets are delivered.

Congestion can be brought on by several factors. If all of a sudden, streams of packets begin arriving on three or four input lines and all need the same output line, a queue will build up.

If there is insufficient memory to hold all of them, packets will be lost.

Slow processors can also cause congestion. If the routers' CPUs are slow at performing the bookkeeping tasks required of them (queuing buffers, updating tables, etc.), queues can build up, even though there is excess line capacity. Similarly, low-bandwidth lines can also cause congestion.

APPROACHES TO CONGESTION CONTROL

Many problems in complex systems, such as computer networks, can be viewed from a control theory point of view. This approach leads to dividing all solutions into two groups: open loop and closed loop.

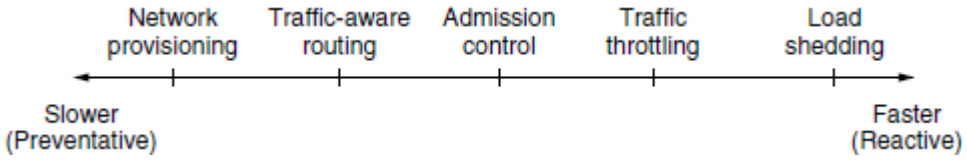


Figure: Timescales Of Approaches To Congestion Control

Open loop solutions attempt to solve the problem by good design.

Tools for doing open-loop control include deciding when to accept new traffic, deciding when to discard packets and which ones, and making scheduling decisions at various points in the network.

Closed loop solutions are based on the concept of a feedback loop.

This approach has three parts when applied to congestion control:

1. Monitor the system to detect when and where congestion occurs.
2. Pass this information to places where action can be taken.
3. Adjust system operation to correct the problem.

A variety of metrics can be used to monitor the subnet for congestion. Chief among these are the percentage of all packets discarded for lack of buffer space, the average queue lengths,

the number of packets that time out and are retransmitted, the average packet delay, and the standard deviation of packet delay. In all cases, rising numbers indicate growing congestion.

The second step in the feedback loop is to transfer the information about the congestion from the point where it is detected to the point where something can be done about it.

In all feedback schemes, the hope is that knowledge of congestion will cause the hosts to take appropriate action to reduce the congestion.

The presence of congestion means that the load is (temporarily) greater than the resources (in part of the system) can handle. Two solutions come to mind: increase the resources or decrease the load.

CONGESTION PREVENTION POLICIES

The methods to control congestion by looking at open loop systems. These systems are designed to minimize congestion in the first place, rather than letting it happen and reacting after the fact. They try to achieve their goal by using appropriate policies at various levels. In Fig. 5-26 we see different data link, network, and transport policies that can affect congestion (Jain, 1990).

Layer	Policies
Transport	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy • Timeout determination
Network	• Virtual circuits versus datagram inside the subnet • Packet queueing and service policy • Packet discard policy • Routing algorithm • Packet lifetime management
Data link	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy

Figure 5-26. Policies that affect congestion.

The **data link layer Policies.**

The **retransmission policy** is concerned with how fast a sender times out and what it transmits upon timeout. A jumpy sender that times out quickly and retransmits all outstanding packets using go back n will put a heavier load on the system than will a leisurely sender that uses selective repeat.

Closely related to this is the **buffering policy**. If receivers routinely discard all out-of- order packets, these packets will have to be transmitted again later, creating extra load. With respect to congestion control, selective repeat is clearly better than go back n.

Acknowledgement policy also affects congestion. If each packet is acknowledged immediately, the acknowledgement packets generate extra traffic. However, if acknowledgements are saved up to piggyback onto reverse traffic, extra timeouts and retransmissions may result. A tight flow control scheme (e.g., a small window) reduces the data rate and thus helps fight congestion.

The **network layer Policies.**

The choice between using **virtual circuits and using datagrams** affects congestion since many congestion control algorithms work only with virtual-circuit subnets.

Packet queueing and service policy relates to whether routers have one queue per input line, one queue per output line, or both. It also relates to the order in which packets are processed (e.g., round robin or priority based).

Discard policy is the rule telling which packet is dropped when there is no space.

A good **routing algorithm** can help avoid congestion by spreading the traffic over all the lines, whereas a bad one can send too much traffic over already congested lines.

Packet lifetime management deals with how long a packet may live before being discarded.

If it is too long, lost packets may clog up the works for a long time, but if it is too short, packets may sometimes time out before reaching their destination, thus inducing retransmissions.

The **transport layer Policies**,

The **same issues occur as in the data link layer**, but in addition, determining the **timeout interval** is harder because the transit time across the network is less predictable than the transit time over a wire between two routers. If the timeout interval is too short, extra packets will be sent unnecessarily. If it is too long, congestion will be reduced but the response time will suffer whenever a packet is lost.

ADMISSION CONTROL

One technique that is widely used to keep congestion that has already started from getting worse is **admission control**.

Once congestion has been signaled, no more virtual circuits are set up until the problem has gone away.

An alternative approach is to allow new virtual circuits but carefully route all new virtual circuits around problem areas. For example, consider the subnet of Fig. 5-27(a), in which two routers are congested, as indicated.

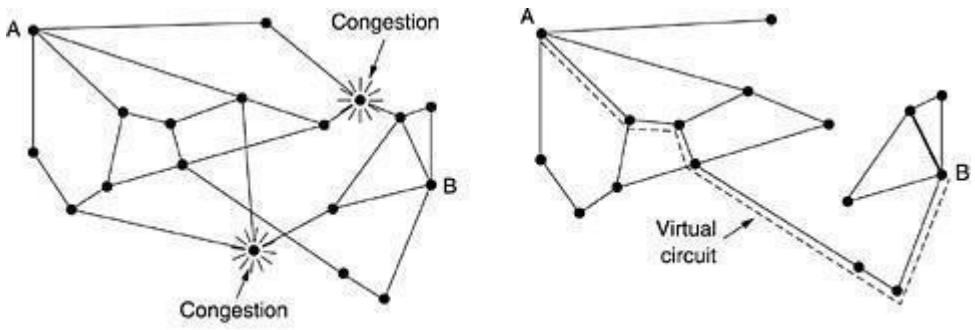


Figure 5-27. (a) A congested subnet. (b) A redrawn subnet that eliminates the congestion. A virtual circuit from* A *to* B *is also shown.

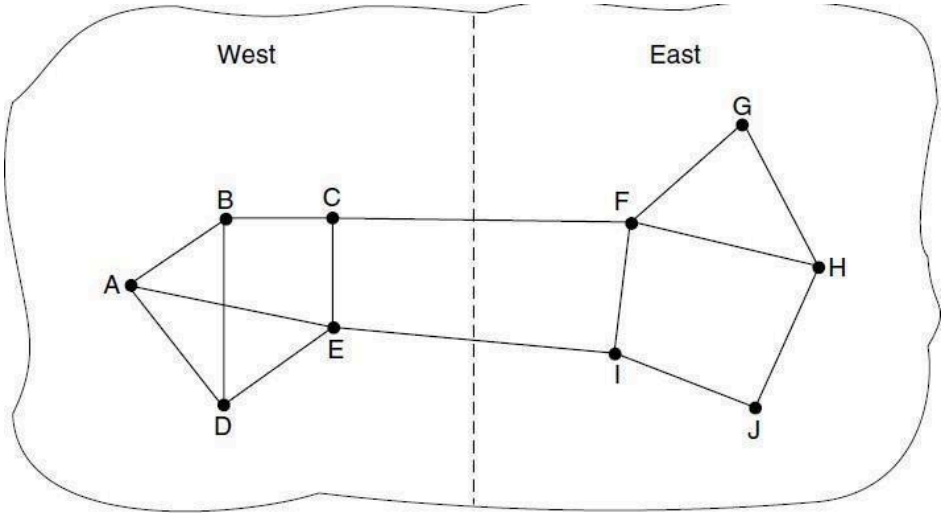
Suppose that a host attached to router *A* wants to set up a connection to a host attached to router *B*. Normally, this connection would pass through one of the congested routers. To avoid this situation, we can redraw the subnet as shown in Fig. 5-27(b), omitting the congested routers and all of their lines. The dashed line shows a possible route for the virtual circuit that avoids the congested routers.

TRAFFIC AWARE ROUTING

These schemes adapted to changes in topology, but not to changes in load. The goal in taking load into account when computing routes is to shift traffic away from hotspots that will be the first places in the network to experience congestion.

The most direct way to do this is to set the link weight to be a function of the (fixed) link bandwidth and propagation delay plus the (variable) measured load or average queuing delay. Least-weight paths will then favor paths that are more lightly loaded, all else being equal.

Consider the network of Fig. 5-23, which is divided into two parts, East and West, connected by two links, *CF* and *EI*. Suppose that most of the traffic between East and West is using link *CF*, and, as a result, this link is heavily loaded with long delays. Including queueing delay in the weight used for the shortest path calculation will make *EI* more attractive. After the new routing tables have been installed, most of the East-West traffic will now go over *EI*, loading this link. Consequently, in the next update, *CF* will appear to be the shortest path. As a result, the routing tables may oscillate wildly, leading to erratic routing and many potential problems.



If load is ignored and only bandwidth and propagation delay are considered, this problem does not occur. Attempts to include load but change weights within a narrow range only slow down routing oscillations. Two techniques can contribute to a successful solution. The first is multipath routing, in which there can be multiple paths from a source to a destination. In our example this means that the traffic can be spread across both of the East to West links. The second one is for the routing scheme to shift traffic across routes slowly enough that it is able to converge.

TRAFFIC THROTTLING

Each router can easily monitor the utilization of its output lines and other resources. For example, it can associate with each line a real variable, *u*, whose value, between 0.0 and 1.0, reflects the recent utilization of that line. To maintain a good estimate of *u*, a sample of the instantaneous line utilization, *f* (either 0 or 1), can be made periodically and *u* updated according to

$$u_{new} = au_{old} + (1 - a)f$$

where the constant *a* determines how fast the router forgets recent history.

Whenever *u* moves above the threshold, the output line enters a "warning" state. Each newly-arriving packet is checked to see if its output line is in warning state. If it is, some action is taken. The action taken can be one of several alternatives, which we will now discuss.

THE WARNING BIT

The old DECNET architecture signaled the warning state by setting a special bit in the packet's header.

When the packet arrived at its destination, the transport entity copied the bit into the next acknowledgement sent back to the source. The source then cut back on traffic.

As long as the router was in the warning state, it continued to set the warning bit, which meant that the source continued to get acknowledgements with it set.

The source monitored the fraction of acknowledgements with the bit set and adjusted its transmission rate accordingly. As long as the warning bits continued to flow in, the source continued to decrease its transmission rate. When they slowed to a trickle, it increased its transmission rate.

Note that since every router along the path could set the warning bit, traffic increased only when no router was in trouble.

CHOKE PACKETS

In this approach, the router sends a **choke packet** back to the source host, giving it the destination found in the packet.

The original packet is tagged (a header bit is turned on) so that it will not generate any more choke packets farther along the path and is then forwarded in the usual way.

When the source host gets the choke packet, it is required to reduce the traffic sent to the specified destination by *X* percent. Since other packets aimed at the same destination are

probably already under way and will generate yet more choke packets, the host should ignore choke packets referring to that destination for a fixed time interval. After that period has expired, the host listens for more choke packets for another interval. If one

arrives, the line is still congested, so the host reduces the flow still more and begins ignoring choke packets again. If no choke packets arrive during the listening period, the host may increase the flow again.

The feedback implicit in this protocol can help prevent congestion yet not throttle any flow unless trouble occurs.

Hosts can reduce traffic by adjusting their policy parameters.

Increases are done in smaller increments to prevent congestion from reoccurring quickly.

Routers can maintain several thresholds. Depending on which threshold has been crossed, the choke packet can contain a mild warning, a stern warning, or an ultimatum.

HOP-BY-HOP BACK PRESSURE

At high speeds or over long distances, sending a choke packet to the source hosts does not work well because the reaction is so slow.

Consider, for example, a host in San Francisco (router *A* in Fig. 5-28) that is sending traffic to a host in New York (router *D* in Fig. 5-28) at 155 Mbps. If the New York host begins to run out of buffers, it will take about 30 msec for a choke packet to get back to San Francisco to tell it to slow down. The choke packet propagation is shown as the second, third, and fourth steps in Fig. 5-28(a). In those 30 msec, another 4.6 megabits will have been sent. Even if the host in San Francisco completely shuts down immediately, the 4.6 megabits in the pipe will continue to pour in and have to be dealt with. Only in the seventh diagram in Fig. 5- 28(a) will the New York router notice a slower flow.

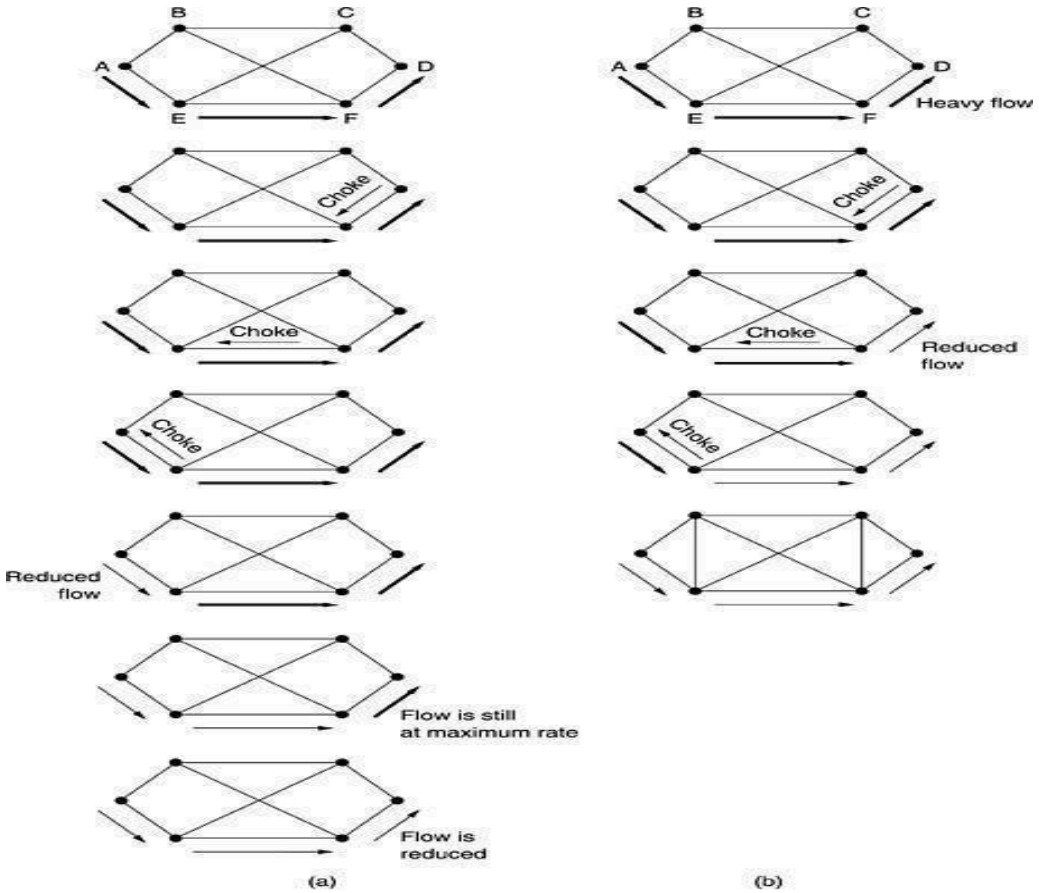


Figure 5-28. (a) A choke packet that affects only the source. (b) A choke packet that affects each hop it passes through.

An alternative approach is to have the choke packet take effect at every hop it passes through, as shown in the sequence of Fig. 5-28(b). Here, as soon as the choke packet reaches *F*, *F* is required to reduce the flow to *D*. Doing so will require *F* to devote more buffers to the flow, since the source is still sending away at full blast, but it gives *D* immediate relief, like a headache remedy in a television commercial. In the next step, the choke packet reaches *E*, which tells *E* to reduce the flow to *F*. This action puts a greater demand on *E*'s buffers but gives *F* immediate relief. Finally, the choke packet reaches *A* and the flow genuinely slows down.

The net effect of this hop-by-hop scheme is to provide quick relief at the point of congestion at the price of using up more buffers upstream. In this way, congestion can be nipped in the bud without losing any packets.

LOAD SHEDDING

When none of the above methods make the congestion disappear, routers can bring out the heavy artillery: load shedding.

Load shedding is a fancy way of saying that when routers are being in undated by packets that they cannot handle, they just throw them away.

A router drowning in packets can just pick packets at random to drop, but usually it can do better than that.

Which packet to discard may depend on the applications running.

To implement an intelligent discard policy, applications must mark their packets in priority classes to indicate how important they are. If they do this, then when packets have to be discarded, routers can first drop packets from the lowest class, then the next lowest class, and so on.

RANDOM EARLY DETECTION

It is well known that dealing with congestion after it is first detected is more effective than letting it gum up the works and then trying to deal with it. This observation leads to the idea of discarding packets before all the buffer space is really exhausted. A popular algorithm for doing this is called **RED** (**Random Early Detection)**.

In some transport protocols (including TCP), the response to lost packets is for the source to slow down. The reasoning behind this logic is that TCP was designed for wired networks and wired networks are very reliable, so lost packets are mostly due to buffer overruns rather than transmission errors. This fact can be exploited to help **reduce congestion**.

By having routers drop packets before the situation has become hopeless (hence the "early" in the name), the idea is that there is time for action to be taken before it is too late. To determine when to start discarding, routers maintain a running average of their queue lengths. When the average queue length on some line exceeds a threshold, the line is said to

be congested and action is taken.

JITTER CONTROL

The variation (i.e., standard deviation) in the packet arrival times is called **jitter**.

High jitter, for example, having some packets taking 20 msec and others taking 30 msec to arrive will give an uneven quality to the sound or movie. Jitter is illustrated in Fig. 5-29. In contrast, an agreement that 99 percent of the packets be delivered with a delay in the range of 24.5 msec to 25.5 msec might be acceptable.

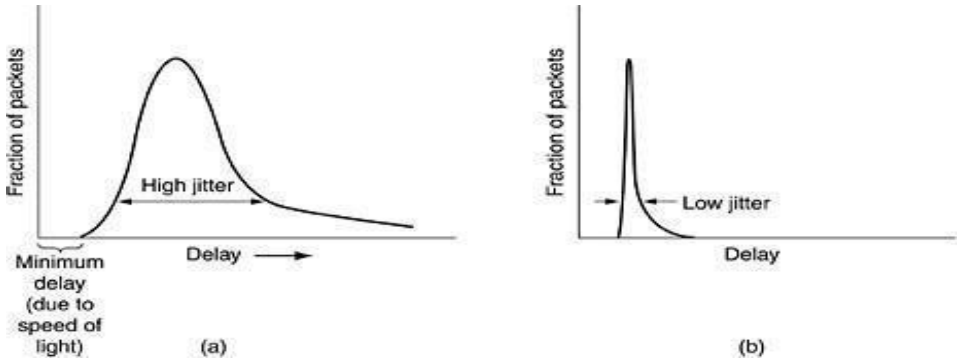


Figure 5-29. (a) High jitter. (b) Low jitter.

The jitter can be bounded by computing the expected transit time for each hop along the path. When a packet arrives at a router, the router checks to see how much the packet is behind or ahead of its schedule. This information is stored in the packet and updated at each hop. If the packet is ahead of schedule, it is held just long enough to get it back on schedule. If it is behind schedule, the router tries to get it out the door quickly.

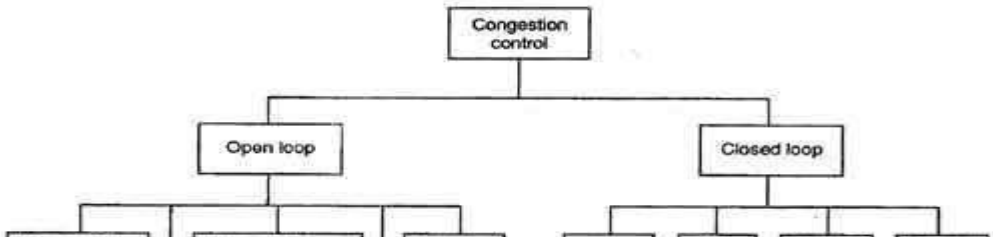
In fact, the algorithm for determining which of several packets competing for an output line should go next can always choose the packet furthest behind in its schedule.

In this way, packets that are ahead of schedule get slowed down and packets that are behind schedule get speeded up, in both cases reducing the amount of jitter.

In some applications, such as video on demand, jitter can be eliminated by buffering at the receiver and then fetching data for display from the buffer instead of from the network in real time. However, for other applications, especially those that require real- time interaction between people such as Internet telephony and videoconferencing, the delay inherent in buffering is not acceptable.

How to correct the Congestion Problem:

Congestion Control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened. Congestion control mechanisms are divided into two categories, one category prevents the congestion from happening and the other category removes congestion after it has taken place.



These two categories are:

1. Open loop
2. Closed loop

Open Loop Congestion Control

• In this method, policies are used to prevent the congestion before it happens.

• Congestion control is handled either by the source or by the destination.

• The various methods used for open loop congestion control are:

1. Retransmission Policy

• The sender retransmits a packet, if it feels that the packet it has sent is lost or corrupted.

• However retransmission in general may increase the congestion in the network. But we need to implement good retransmission policy to prevent congestion.

• The retransmission policy and the retransmission timers need to be designed to optimize efficiency and at the same time prevent the congestion.

2. Window Policy

• To implement window policy, selective reject window method is used for congestion control.

• Selective Reject method is preferred over Go-back-n window as in Go-back-n method, when timer for a packet times out, several packets are resent, although some may have arrived safely at the receiver. Thus, this duplication may make congestion worse.

• Selective reject method sends only the specific lost or damaged packets.

3. Acknowledgement Policy

• The acknowledgement policy imposed by the receiver may also affect congestion.

• If the receiver does not acknowledge every packet it receives it may slow down the sender and help prevent congestion.

• Acknowledgments also add to the traffic load on the network. Thus, by sending fewer acknowledgements we can reduce load on the network.

• To implement it, several approaches can be used:

1. A receiver may send an acknowledgement only if it has a packet to be sent.
2. A receiver may send an acknowledgement when a timer expires.
3. A receiver may also decide to acknowledge only *N* packets at a time.

4. Discarding Policy

• A router may discard less sensitive packets when congestion is likely to happen.

• Such a discarding policy may prevent congestion and at the same time may not harm the integrity of the transmission.

5. Admission Policy

- An admission policy, which is a quality-of-service mechanism, can also prevent congestion in virtual circuit networks.

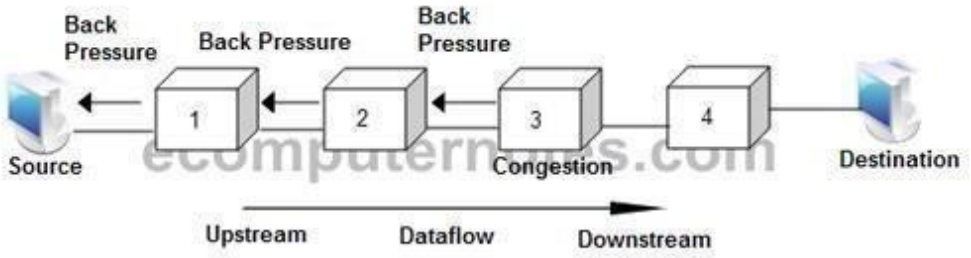
- Switches in a flow first check the resource requirement of a flow before admitting it to the network.
- A router can deny establishing a virtual circuit connection if there is congestion in the "network or if there is a possibility of future congestion.

Closed Loop Congestion Control

- Closed loop congestion control mechanisms try to remove the congestion after it happens.
- The various methods used for closed loop congestion control are:

1. Backpressure

- Backpressure is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow.



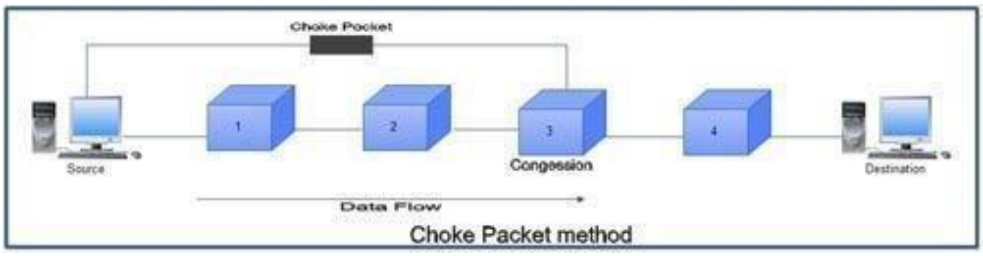
Backpressure Method

- The backpressure technique can be applied only to virtual circuit networks. In such virtual circuit each node knows the upstream node from which a data flow is coming.
- In this method of congestion control, the congested node stops receiving data from the immediate upstream node or nodes.
- This may cause the upstream node on nodes to become congested, and they, in turn, reject data from their upstream node or nodes.

As shown in fig node 3 is congested and it stops receiving packets and informs its upstream node 2 to slow down. Node 2 in turns may be congested and informs node 1 to slow down. Now node 1 may create congestion and informs the source node to slow down. In this way the congestion is alleviated. Thus, the pressure on node 3 is moved backward to the source to remove the congestion.

2. Choke Packet

- In this method of congestion control, congested router or node sends a special type of packet called choke packet to the source to inform it about the congestion.
- Here, congested node does not inform its upstream node about the congestion as in backpressure method.
- In choke packet method, congested node sends a warning directly to the source station *i.e.* the intermediate nodes through which the packet has traveled are not warned.



3. Implicit Signaling

- In implicit signaling, there is no communication between the congested node or nodes and the source.

- The source guesses that there is congestion somewhere in the network when it does not receive any acknowledgment. Therefore the delay in receiving an acknowledgment is interpreted as congestion in the network.

- On sensing this congestion, the source slows down.

- This type of congestion control policy is used by TCP.

4. Explicit Signaling

- In this method, the congested nodes explicitly send a signal to the source or destination to inform about the congestion.

- Explicit signaling is different from the choke packet method. In choke packed method, a separate packet is used for this purpose whereas in explicit signaling method, the signal is included in the packets that carry data .

- Explicit signaling can occur in either the forward direction or the backward direction .

- In backward signaling, a bit is set in a packet moving in the direction opposite to the congestion. This bit warns the source about the congestion and informs the source to slow down.

- In forward signaling, a bit is set in a packet moving in the direction of congestion. This bit warns the destination about the congestion. The receiver in this case uses policies such as slowing down the acknowledgements to remove the congestion.

UNIT -4
TRANSPORT LAYER

The Internet Transport Protocols

The Internet has two main protocols in the transport layer, **a connectionless protocol** and a **connection- oriented** one. The protocols complement each other. The connectionless protocol is **UDP.** It does almost nothing beyond sending packets between applications, letting applications build their own protocols on top as needed.

The connection-oriented protocol is **TCP.** It does almost everything. It makes connections and adds reliability with retransmissions, along with flow control and congestion control, all on behalf of the applications that use it. Since UDP is a transport layer protocol that typically runs in the operating system and protocols that use UDP typically run in user s pace, these uses might be considered applications.

UDP

INTROUCTION TO UDP

The Internet protocol suite supports a connectionless transport protocol called UDP (User Datagram Protocol). UDP provides a way for applications to send encapsulated IP datagrams without having to establish a connection.

UDP transmits segments consisting of an 8-byte header followed by the pay-load. The two ports serve to identify the end-points within the source and destination machines.

When a UDP packet arrives, its payload is handed to the process attached to the destination port. This attachment occurs when the BIND primitive. Without the port fields, the transport layer would not know what to do with each incoming packet. With them, it delivers the embedded segment to the correct application.

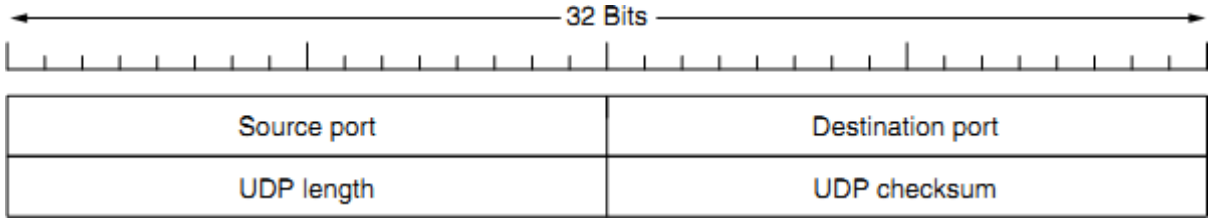


Fig 4.9: The UDP header

Source port, destination port: Identifies the end points within the source and destination machines.

UDP length: Includes 8-byte header and the data

UDP checksum: Includes the UDP header, the UDP data padded out to an even number of bytes if need be. It is an optional field

REMOTE PROCEDURE CALL

- In a certain sense, sending a message to a remote host and getting a reply back is like making a function call in a programming language. This is to arrange request-reply interactions on networks to be cast in the form of procedure calls.

- For example, just imagine a procedure named *get IP address* (*host name*) that works by sending a UDP packet to a DNS server and waiting or the reply, timing out and trying again if one is not forthcoming quickly enough. In this way, all the details of networking can be hidden from the programmer.

- RPC is used to call remote programs using the procedural call. When a process on machine 1 calls a procedure on machine 2, the calling process on 1 is suspended and execution of the called procedure takes place on 2.

- Information can be transported from the caller to the callee in the parameters and can come back in the procedure result. No message passing is visible to the application programmer. This technique is known as **RPC** (**Remote Procedure Call**) and has become the basis for many networking applications.

  Traditionally, the calling procedure is known as the **client** and the called procedure is known as the server.

- In the simplest form, to call a remote procedure, the client program must be bound with a small library procedure, called the **client stub**,that represents the server procedure in the client's address space. Similarly, the server is bound with a procedure called the **server stub**. These procedures hide the fact that the procedure call from the client to the server is not local.

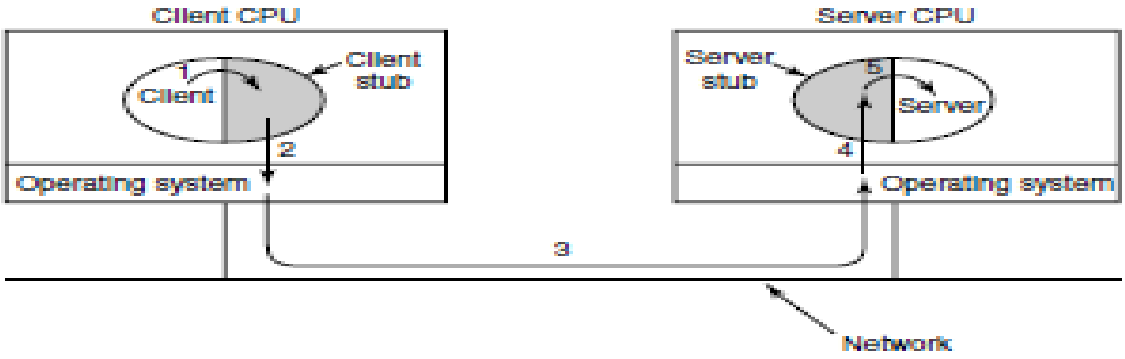


Fig 4.10: Steps in making a RPC

Step 1 is the client calling the client stub. This call is a local procedure call, with the parameters pushed onto the stack in the normal way.

Step 2 is the client stub packing the parameters into a message and making a system call to send the message. Packing the parameters is called **marshaling**.

Step 3 is the operating system sending the message from the client machine to the server machine.

Step 4 is the operating system passing the incoming packet to the server stub.
Step 5 is the server stub calling the server procedure with the **unmarshaled** parameters. The reply traces the same path in the other direction.

The key item to note here is that the client procedure, written by the user, just makes a normal (i.e., local) procedure call to the client stub, which has the same name as the server procedure. Since the client procedure and client stub are in the same address space, the parameters are passed in the usual way.

Similarly, the server procedure is called by a procedure in its address space with the parameters it expects. To the server procedure, nothing is unusual. In this way, instead of I/O being done on sockets, network communication is done by faking a normal procedure call. With RPC, passing pointers is impossible because the client and server are in different address spaces.

UDP Applications

UDP does not provide error control; it provides an unreliable service. Most applications expect reliable service from a transport-layer protocol. Although a reliable service is desirable.

UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control

UDP is suitable for a process with internal flow- and error-control mechanisms. For example, the Trivial File Transfer Protocol (TFIP)

UDP is a suitable transport protocol for multicasting. Multicasting capability is embedded in the UDP software

UDP is used for management processes such as SNMP

UDP is used for some route updating protocols such as Routing Information Protocol (RIP)

UDP is normally used for interactive real-time applications that cannot tolerate uneven delay between sections of a received message

TCP

It was specifically designed to provide a reliable end-to end byte stream over an unreliable network. It was designed to adapt dynamically to properties of the inter network and to be robust in the face of many kinds of failures.

Each machine supporting TCP has a TCP transport entity, which accepts user data streams from local processes, breaks them up into pieces not exceeding 64kbytes and sends each piece as a separate IP datagram. When these datagrams arrive at a machine, they are given to TCP entity, which reconstructs the original byte streams. It is up to TCP to time out and retransmits them as needed, also to reassemble datagrams into messages in proper sequence.

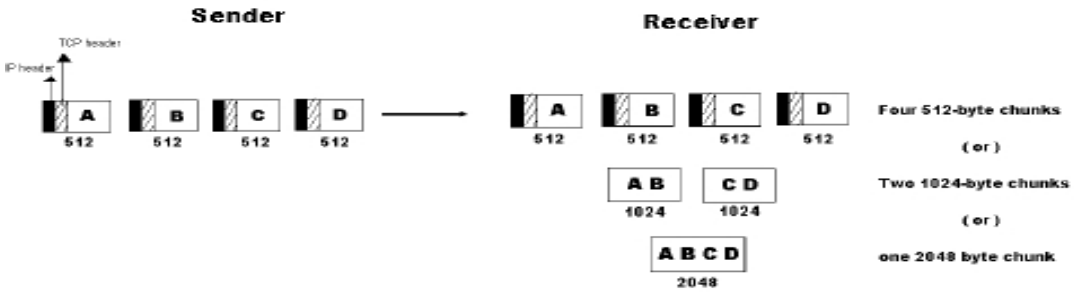
The different issues to be considered are:

1. The TCP Service Model
2. The TCP Protocol
3. The TCP Segment Header
4. The Connection Management
5. TCP Transmission Policy
6. TCP Congestion Control
7. TCP Timer Management.

The TCP Service Model

- TCP service is obtained by having both the sender and receiver create end points called **SOCKETS**
  - Each socket has a socket number(address)consisting of the IP address of the host, called a **"PORT"** (= TSAP)
  - To obtain TCP service a connection must be explicitly established between a socket on the sending machine and a socket on the receiving machine
  - All TCP connections are full duplex and point to point i.e., multicasting or broadcasting is not supported.
  - A TCP connection is a byte stream, not a message stream i.e., the data is delivered as chunks

E.g.: 4 * 512 bytes of data is to be transmitted.



Sockets:

A socket may be used for multiple connections at the same time. In other words, 2 or more connections may terminate at same socket. Connections are identified by socket identifiers at same socket. Connections are identified by socket identifiers at both ends. Some of the sockets are listed below:

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Ports: Port numbers below 256 are called Well- known ports and are reserved for standard services.

***Eg*:**

PORT-21	To establish a connection to a host to transfer a file using FTP
PORT-23	To establish a remote login session using TELNET

The TCP Protocol

- A key feature of TCP, and one which dominates the protocol design, is that every byte on a TCP connection has its own 32-bit sequence number.
- When the Internet began, the lines between routers were mostly 56-kbps leased lines, so a host blasting away at full speed took over 1 week to cycle through the sequence numbers.
- The basic protocol used by TCP entities is the **sliding window protocol**.
- When a sender transmits a segment, it also starts a timer.
- When the segment arrives at the destination, the receiving TCP entity sends back a segment (with data if any exist, otherwise without data) bearing an acknowledgement number equal to the next sequence number it expects to receive.
- If the sender's timer goes off before the acknowledgement is received, the sender transmits the segment again.

The TCP Segment Header

Every segment begins with a fixed-format, 20-byte header. The fixed header may be followed by header options. After the options, if any, up to 65,535 - 20 - 20 = 65,495 data bytes may follow, where the first 20 refer to the IP header and the second to the TCP header. Segments without any data are legal and are commonly used for acknowledgements and control messages.

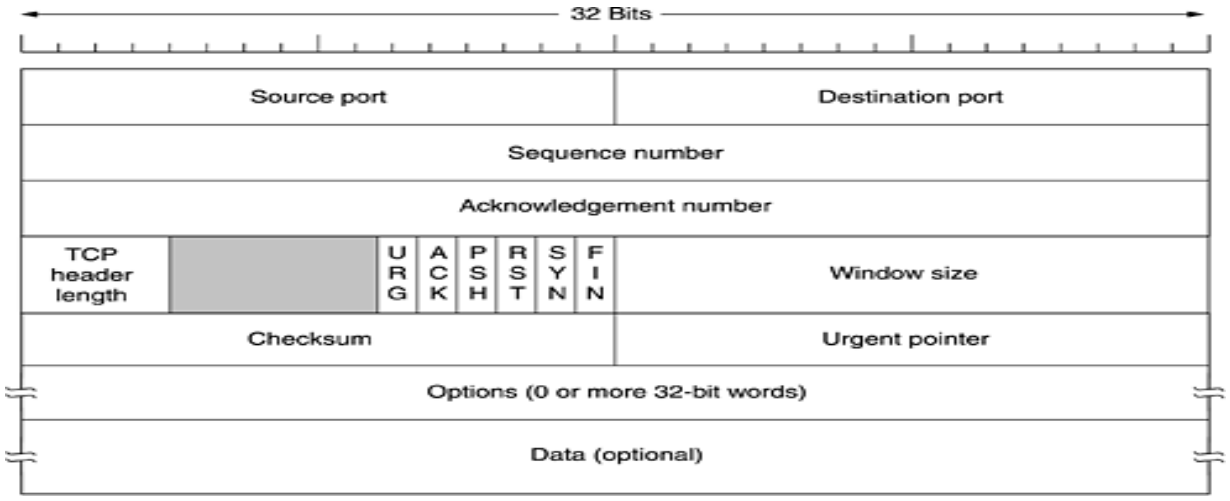


Fig 4.11: The TCP Header

Source Port, Destination Port : Identify local end points of the connections **Sequence number:** Specifies the sequence number of the segment **Acknowledgement Number:** Specifies the next byte expected.

TCP header length: Tells how many 32-bit words are contained in TCP header **URG: I**t is set to 1 if URGENT pointer is in use, which indicates start of urgent data. **ACK:** It is set to 1 to indicate that the acknowledgement number is valid.

PSH: Indicates pushed data

RST: It is used to reset a connection that has become confused due to reject an invalid segment or refuse an attempt to open a connection.

FIN: Used to release a connection.

SYN: Used to establish connections.

TCP Connection Establishment

To establish a connection, one side, say, the server, passively waits for an incoming connection by executing the LISTEN and ACCEPT primitives, either specifying a specific source or nobody in particular.

The other side, say, the client, executes a CONNECT primitive, specifying the IP address and port to which it wants to connect, the maximum TCP segment size it is willing to accept, and optionally some user data (e.g., a password).

The CONNECT primitive sends a TCP segment with the *SYN* bit on and *ACK* bit off and waits for a response.

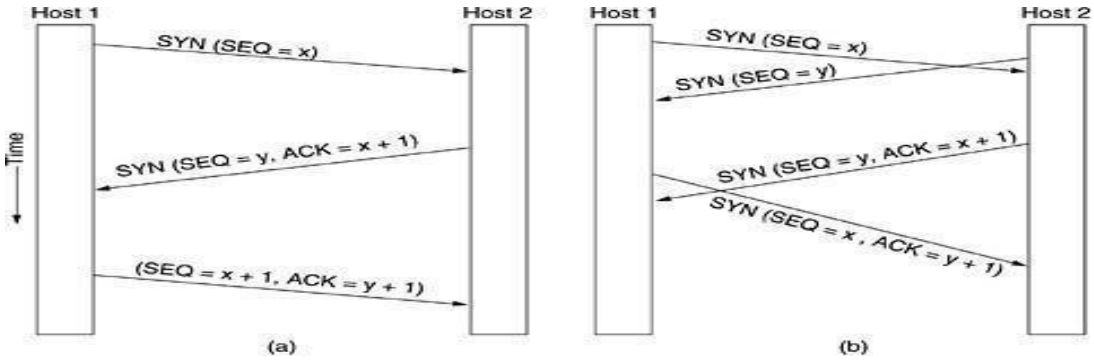


Fig 4.12: a) TCP Connection establishment in the normal case b) Call Collision

TCP Connection Release

- Although TCP connections are full duplex, to understand how connections are released it is best to think of them as a pair of simplex connections.
- Each simplex connection is released independently of its sibling. To release a connection, either party can send a TCP segment with the *FIN* bit set, which means that it has no more data to transmit.
- When the *FIN* is acknowledged, that direction is shut down for new data. Data may continue to flow indefinitely in the other direction, however.
- When both directions have been shut down, the connection is released.
- Normally, four TCP segments are needed to release a connection, one *FIN* and one *ACK* for each direction. However, it is possible for the first *ACK* and the second *FIN* to be contained in the same segment, reducing the total count to three.

TCP Connection Management Modeling

The steps required establishing and release connections can be represented in a finite state machine with the 11 states listed in Fig. 4.13. In each state, certain events are legal. When a legal event happens, some action may be taken. If some other event happens, an error is reported.

State	Description
CLOSED	No connection is active or pending
LISTEN	The server is waiting for an incoming call
SYN RCVD	A connection request has arrived; wait for ACK
SYN SENT	The application has started to open a connection
ESTABLISHED	The normal data transfer state
FIN WAIT 1	The application has said it is finished
FIN WAIT 2	The other side has agreed to release
TIMED WAIT	Wait for all packets to die off
CLOSING	Both sides have tried to close simultaneously
CLOSE WAIT	The other side has initiated a release
LAST ACK	Wait for all packets to die off

Figure 4.13. The states used in the TCP connection management finite state machine.

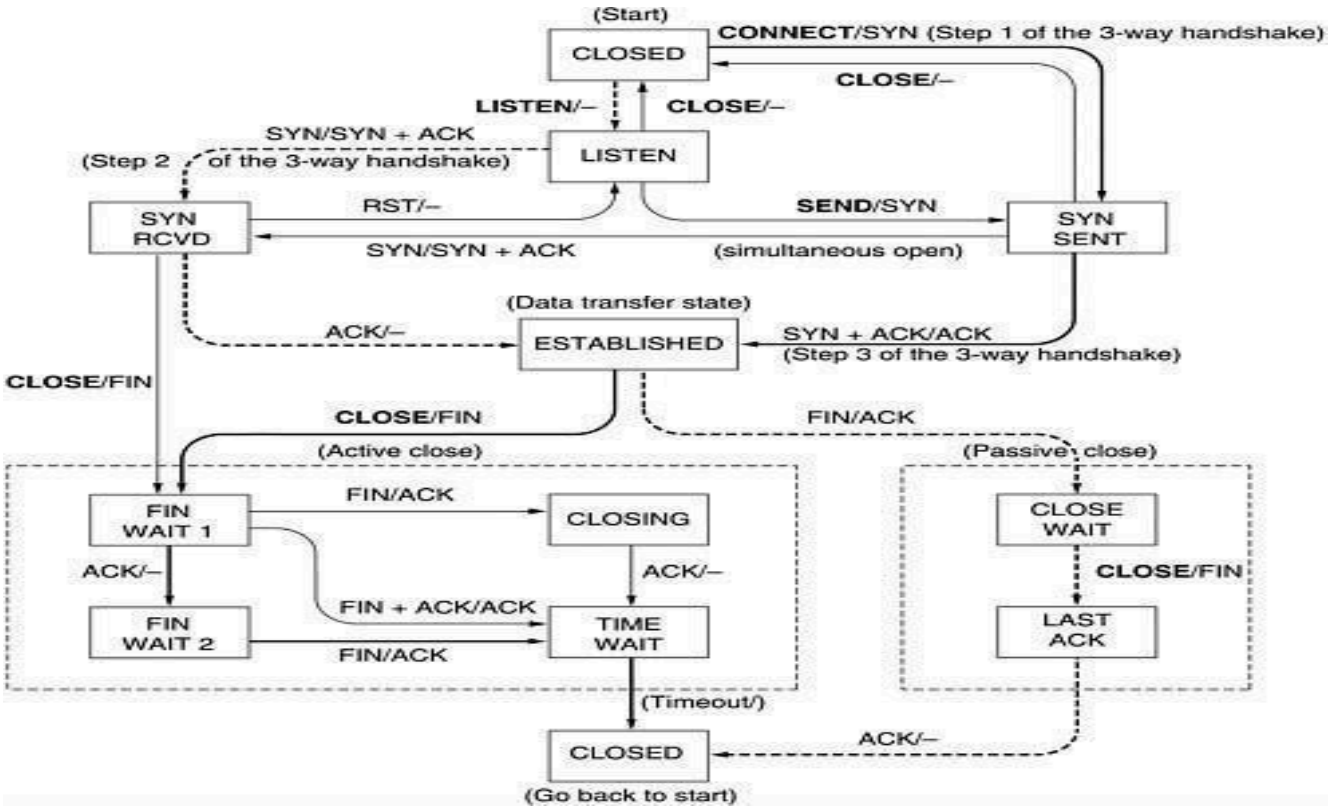
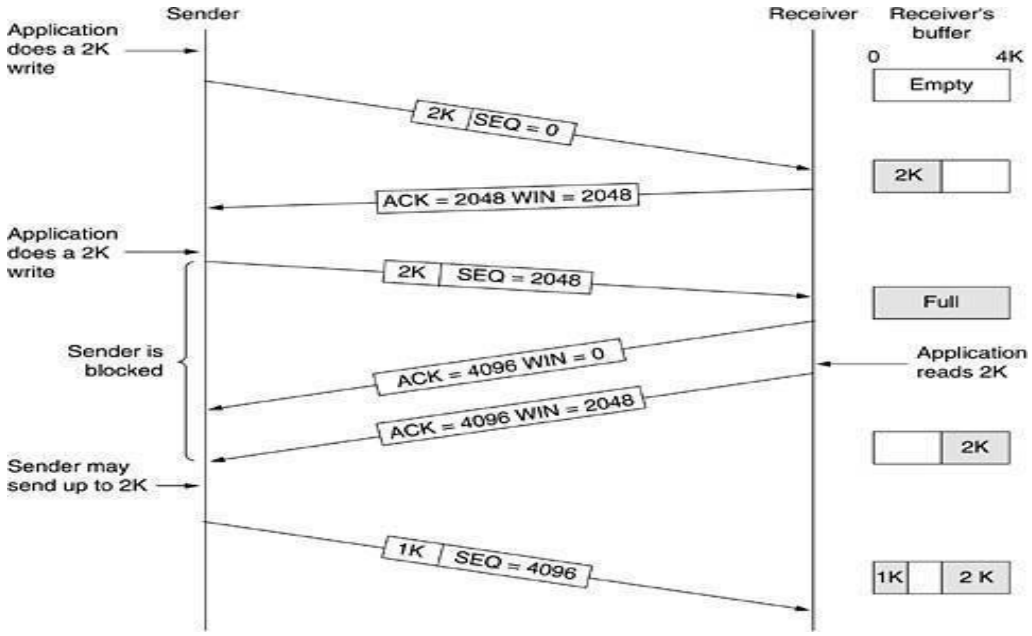


Figure 4.14 - TCP connection management finite state machine.

TCP Connection management from server's point of view:

1. The server does a **LISTEN** and settles down to see who turns up.
2. When a **SYN** comes in, the server acknowledges it and goes to the **SYNRCVD** state
3. When the servers **SYN** is itself acknowledged the 3-way handshake is complete and server goes to the **ESTABLISHED** state. Data transfer can now occur.
4. When the client has had enough, it does a close, which causes a **FIN** to arrive at the server [dashed box marked passive close].
5. The server is then signaled.
6. When it too, does a **CLOSE**, a **FIN** is sent to the client.
7. When the client's acknowledgement shows up, the server releases the connection and deletes the connection record.

TCP Transmission Policy

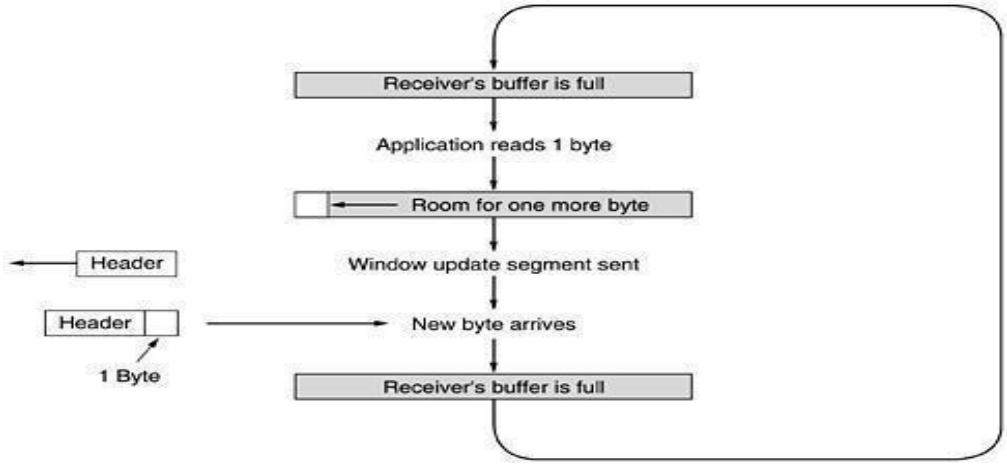


1. In the above example, the receiver has 4096-byte buffer.
2. If the sender transmits a 2048-byte segment that is correctly received, the receiver will acknowledge the segment.
3. Now the receiver will advertise a window of 2048 as it has only 2048 of buffer space, now.
4. Now the sender transmits another 2048 bytes which are acknowledged, but the advertised window is'0'.
5. The sender must stop until the application process on the receiving host has removed some data from the buffer, at which time TCP can advertise a layer window.

SILLY WINDOW SYNDROME:

This is one of the problems that ruin the TCP performance, which occurs when data are passed to the sending TCP entity in large blocks, but an interactive application on the receiving side reads 1 byte at a time.

- Initially the TCP buffer on the receiving side is full and the sender knows this(win=0)
- Then the interactive application reads 1 character from tcp stream.
- Now, the receiving TCP sends a window update to the sender saying that it is all right to send 1 byte.
- The sender obligates and sends 1 byte.
- The buffer is now full, and so the receiver acknowledges the 1 byte segment but sets window to zero. This behavior can go on forever.



TCP CONGESTION CONTROL:

TCP does to try to prevent the congestion from occurring in the first place in the following way:

When a connection is established, a suitable window size is chosen and the receiver specifies a window based on its buffer size. If the sender sticks to this window size, problems will not occur due to buffer overflow at the receiving end. But they may still occur due to internal congestion within the network. Let's see this problem occurs.

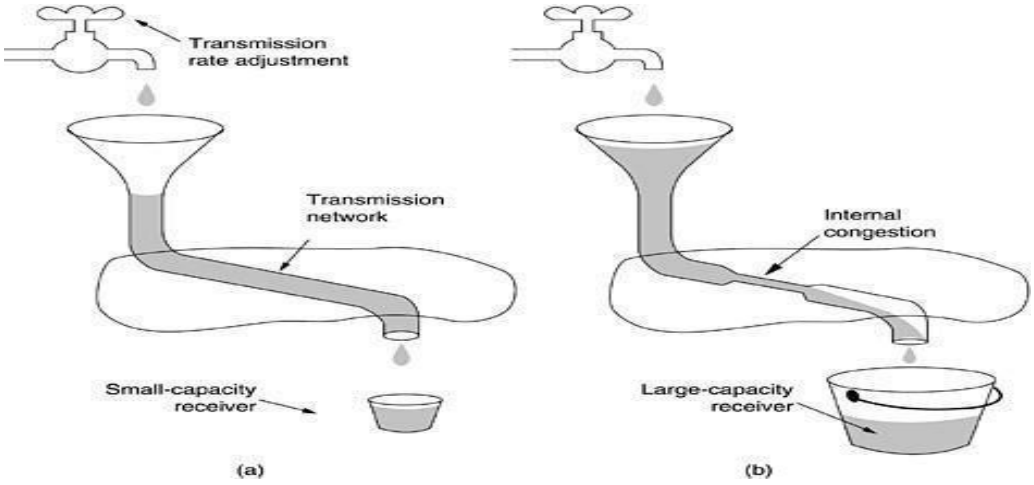


Figure 4.16. (a) A fast network feeding a low-capacity receiver. (b) A slow network feeding a high-capacity receiver.

In fig (a): We see a thick pipe leading to a small- capacity receiver. As long as the sender does not send more water than the bucket can contain, no water will be lost.

In fig (b): The limiting factor is not the bucket capacity, but the internal carrying capacity of the n/w. if too much water comes in too fast, it will backup and some will be lost.

- When a connection is established, the sender initializes the congestion window to the size of the max segment in use our connection.
- It then sends one max segment .if this max segment is acknowledged before the timer

goes off, it adds one segment s worth of bytes to the congestion window to make it two maximum size segments and sends 2 segments.

- As each of these segments is acknowledged, the congestion window is increased by one max segment size.
- When the congestion window is ‘n’ segments, if all ‘n’ are acknowledged on time, the congestion window is increased by the byte count corresponding to ‘n’ segments.
- The congestion window keeps growing exponentially until either a time out occurs or the receiver’s window is reached.
- The internet congestion control algorithm uses a third parameter, the **“threshold”** in addition to receiver and congestion windows.

Different congestion control algorithms used by TCP are:

- RTT variance Estimation.
- Exponential RTO back-off Re-transmission Timer Management
- Karn’s Algorithm
- Slow Start
- Dynamic window sizing on congestion
- Fast Retransmit Window Management
- Fast Recovery

TCP TIMER MANAGEMENT:

TCP uses 3 kinds of timers:

1. Retransmission timer
2. Persistence timer
3. Keep-Alive timer.

1. **Retransmission timer:** When a segment is sent, a timer is started. If the segment is acknowledged before the timer expires, the timer is stopped. If on the other hand, the timer goes off before the acknowledgement comes in, the segment is retransmitted and the timer is started again. The algorithm that constantly adjusts the time-out interval, based on continuous measurements of n/w performance was proposed by JACOBSON and works as follows:

- for each connection, TCP maintains a variable RTT, that is the best current estimate of the round trip time to the destination inn question.
- When a segment is sent, a timer is started, both to see how long the acknowledgement takes and to trigger a retransmission if it takes too long.
- If the acknowledgement gets back before the timer expires, TCP measures how long the measurements took say M
- It then updates RTT according to the formula

$$\mathbf{RTT = \alpha RTT + (1-\alpha)\, M}$$

Where α *= a smoothing factor that determines how much weight is given to the old value. Typically,* α ***=7/8***

Retransmission timeout is calculated as

$$\mathbf{D = \alpha\, D + (1-\alpha)\, |RTT-M|}$$

Where D = another smoothed variable, Mean RTT = expected acknowledgement value M = observed acknowledgement value

$$\mathbf{Timeout = RTT+(4*D)}$$

2. **Persistence timer**:
It is designed to prevent the following deadlock:

- The receiver sends an acknowledgement with a window size of '0' telling the sender to wait later, the receiver updates the window, but the packet with the update is lost now both the sender and receiver are waiting for each other to do something
- when the persistence timer goes off, the sender transmits a probe to the receiver the response to the probe gives the window size
- if it is still zero, the persistence timer is set again and the cycle repeats
- if it is non zero, data can now be sent

3. **Keep-Alive timer: W**hen a connection has been idle for a long time, this timer may go off to cause one side to check if other side is still there. If it fails to respond, the connection is terminated.

UNIT-5
APPLICATION LAYER

DOMAIN NAME SYSTEM

This is primarily used for mapping host and e-mail destinations to IP addresses but can also be used other purposes. DNS is defined in RFCs 1034 and 1035.

Working:-

- To map a name onto an IP address, an application program calls a library procedure called Resolver, passing it the name as a parameter.
- The resolver sends a UDP packet to a local DNS server, which then looks up the name and returns the IP address to the resolver, which then returns it to the caller.
- Armed with the IP address, the program can then establish a TCP connection with the destination, or send it UDP packets.

1. **The DNS name space.**
2. **Resource Records.**
3. **Name Servers.**

1. THE DNS NAME SPACE:

The Internet is divided into several hundred top level domains, where each domain covers many hosts. Each domain is partitioned into sub domains, and these are further partitioned as so on. All these domains can be represented by a tree, in which the leaves represent domains that have no sub domains. A leaf domain may contain a single host, or it may represent a company and contains thousands of hosts. Each domain is named by the path upward from it to the root. The components are separated by periods (pronounced "dot")

Eg: Sun Microsystems Engg. Department = eng.sun.com.

The top domain comes in 2 flavours:-

- **Generic:** com(commercial), edu(educational instructions), mil(the U.S armed forces, government), int (certain international organizations), net(network providers), org (non profit organizations).
- **Country:** include 1 entry for every country. Domain names can be either absolute (ends with a period
  e.g. eng.sum.com) or relative (doesn't end with a period). Domain names are case sensitive and the component names can be up to 63 characters long and full path names must not exceed 255 characters.

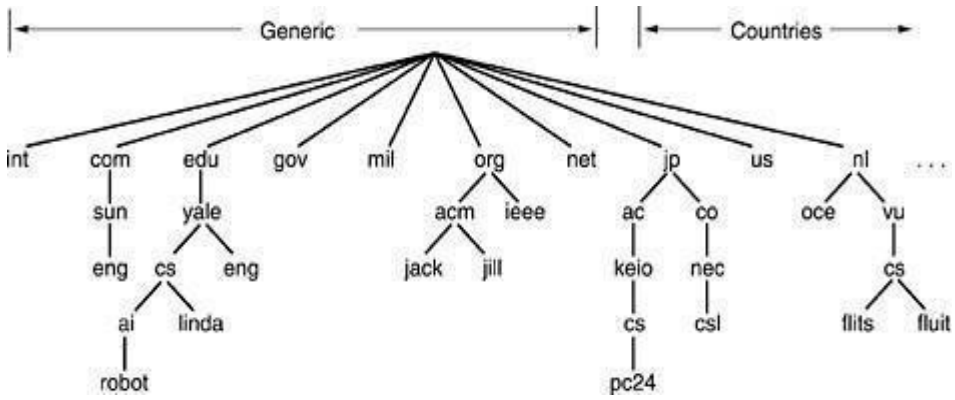


Figure 5-1. A portion of the Internet domain name space.

Insertions of a domain into the tree can be done in 2 days:-

- Under a generic domain (Eg: cs.yale.edu)
- Under the domain of their country (E.g: cs.yale.ct.us)

2. RESOURCE RECORDS:

Every domain can have a sent of resource records associated with it. For a single host, the most common resource record is just its IP address. When a resolver gives a domain name to DNS, it gets both the resource records associated with that name i.e., the real function of DNS is to map domain names into resource records. A resource record is a 5- tuple and its format is as follows:

Domain	Name	Time to live	Type	Class	Value

Domain _name : Tells the domain to which this record applies.

Time- to- live : Gives an identification of how stable the record is (High Stable = 86400 i.e. no. of seconds
/day) (High Volatile = 1 min)
Type: Tells what kind of record this is.
Class: It is IN for the internet information and codes for non internet information
Value: This field can be a number a domain name or an ASCII string

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

3. NAME SERVERS:

It contains the entire database and responds to all queries about it. DNS name space is divided up into non- overlapping zones, in which each zone contains some part of the tree and also contains name servers holding the authoritative information about that zone.

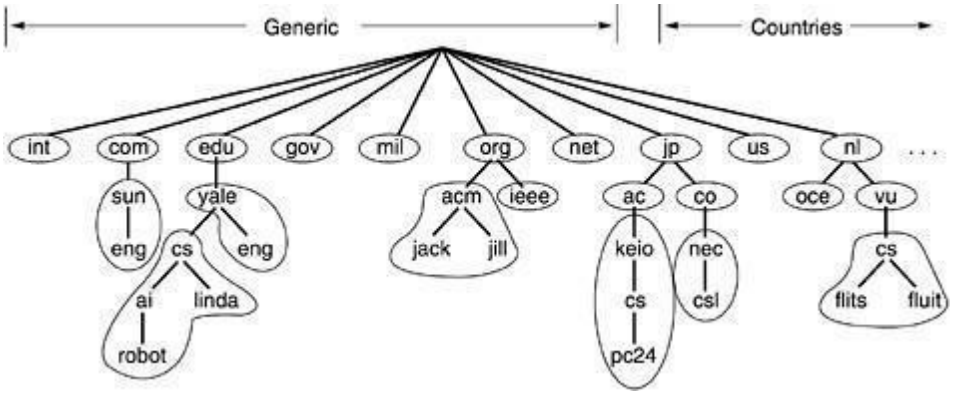


Figure 5-2. Part of the DNS name space showing the division into zones.

When a resolver has a query about a domain name, it passes the query to one of the local name servers:

1. If the domain being sought falls under the jurisdiction of name server, it returns the authoritative resource records (that comes from the authority that manages the record, and is always correct).
2. If the domain is remote and no information about the requested domain is available locally the name server sends a query message to the top level name server for the domain requested.

E.g.: A resolver of flits.cs.vle.nl wants to know the IP address of the host Linda.cs.yale.edu

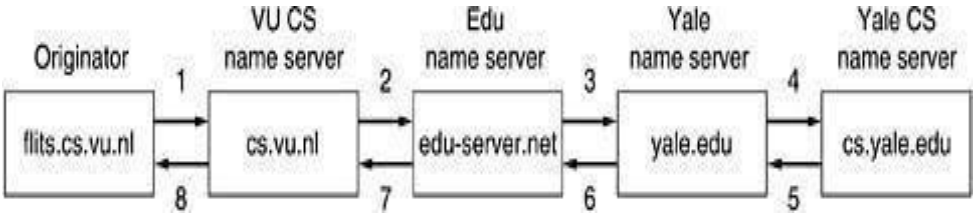


Figure 5-3. How a resolver looks up a remote name in eight steps.

Step 1: Resolver sends a query containing domain name sought the type and the class to local name server, cs.vu.nl.

Step 2: Suppose local name server knows nothing about it, it asks few others nearby name servers. If none of them know, it sends a UDP packet to the server for edu-server.net.

Step 3: This server knows nothing about Linda.cs.yale.edu or cs.yale.edu and so it forwards the request to the name server for yale.edu.

Step 4: This one forwards the request to cs.yale.edu which must have authoritative resource records.

Step 5 to 8: The resource record requested works its way back in steps 5-8 This query method is known as Recursive Query

3. When a query cannot be satisfied locally, the query fails but the name of the next server along the line to try is returned.

ELECTRONIC MAIL

1. ARCHITECTURE AND SERVICES:

E-mail systems consist of two subsystems. They are:-

(1). **User Agents**, which allow people to read and send e-mail

(2). **Message Transfer Agents**, which move messages from source to destination E-mail systems support 5 basic functions:-

a. Composition
b. Transfer
c. Reporting
d. Displaying
e. Disposition

(a). **Composition:** It refers to the process of creating messages and answers. Any text editor is used for body of the message. While the system itself can provide assistance with addressing and numerous header fields attached to each message.

(b). **Reporting:** It has to do with telling the originator what happened to the message that is, whether it was delivered, rejected (or) lost.

(c). **Transfer:** It refers to moving messages from originator to the recipient.

(d). **Displaying:** Incoming messages are to be displayed so that people can read their email.

(e). **Disposition:** It concerns what the recipient dose with the message after receiving it. Possibilities include throwing it away before reading (or) after reading, saving it and so on.

Most systems allow users to create **mailboxes** to store incoming e-mail. Commands are needed to create and destroy mailboxes, inspect the contents of mailboxes, insert and delete messages from mailboxes, and so on.

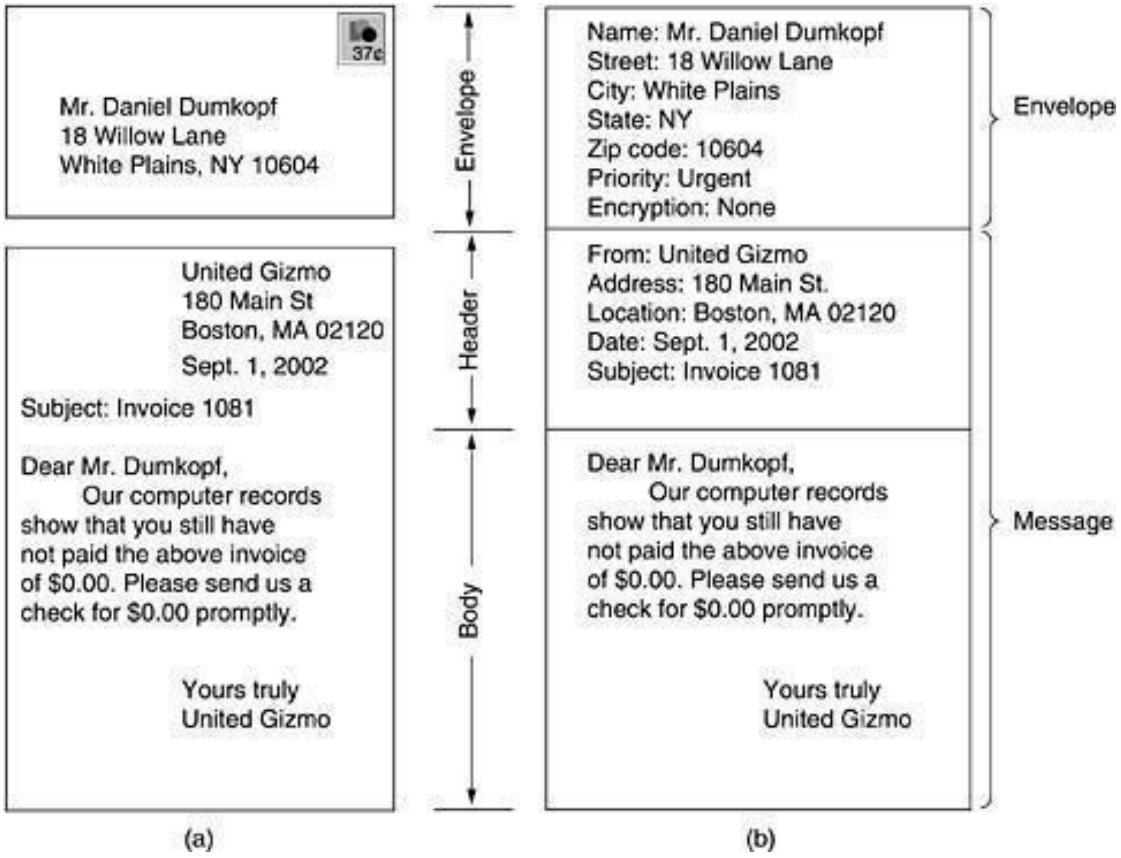


Figure 5-4: Envelopes and messages. (a) Paper mail. (b) Electronic mail.

(1) THE USER AGENT

A user agent is normally a program (sometimes called a mail reader) that accepts a variety of commands for composing, receiving, and replying to messages, as well as for manipulating mailboxes.

SENDING E-MAIL

To send an e-mail message, a user must provide the message, the destination address, and possibly some other parameters. The message can be produced with a free-standing text editor, a word processing program, or possibly with a specialized text editor built into the user agent. The destination address must be in a format that the user agent can deal with. Many user agents expect addresses of the form *user@dns-address*.

READING E-MAIL

When a user agent is started up, it looks at the user's mailbox for incoming e-mail before displaying anything on the screen. Then it may announce the number of messages in the mailbox or display a one-line summary of each one and wait for a command.

(2) MESSAGE FORMATS

RFC 822

Messages consist of a primitive envelope (described in RFC 821), some number of header fields, a blank line, and then the message body. Each header field (logically) consists of a single line of ASCII text containing the field name, a colon, and, for most fields, a value.

Header	Meaning
To:	E-mail address(es) of primary recipient(s)
Cc:	E-mail address(es) of secondary recipient(s)
Bcc:	E-mail address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	E-mail address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

Figure 5-5: RFC 822 header fields related to message transport

MIME — The Multipurpose Internet Mail Extensions

RFC 822 specified the headers but left the content entirely up to the users. Nowadays, on the worldwide Internet, this approach is no longer adequate. The problems include sending and receiving

1. Messages in languages with accents (e.g., French and German).
2. Messages in non-Latin alphabets (e.g., Hebrew and Russian).
3. Messages in languages without alphabets (e.g., Chinese and Japanese).
4. Messages not containing text at all (e.g., audio or images).

A solution was proposed in RFC 1341 called **MIME** (**Multipurpose Internet Mail Extensions**)

The basic idea of MIME is to continue to use the RFC 822 format, but to add structure to the message body and define encoding rules for non-ASCII messages. By not deviating from RFC 822, MIME messages can be sent using the existing mail programs and protocols. All that has to be changed are the sending and receiving programs, which users can do for themselves.

Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

Figure 5-6: RFC 822 headers added by MIME

MESSAGE TRANSFER

The message transfer system is concerned with relaying messages from the originator to the recipient. The simplest way to do this is to establish a transport connection from the source machine to the destination machine and then just transfer the message.

SMTP—THE SIMPLE MAIL TRANSFER PROTOCOL

SMTP is a simple ASCII protocol. After establishing the TCP connection to port 25, the sending machine, operating as the client, waits for the receiving machine, operating as the server, to talk first. The server starts by sending a line of text giving its identity and telling whether it is prepared to receive mail. If it is not, the client releases the connection and tries again later.

Even though the SMTP protocol is completely well defined, a **few problems** can still arise.

One problem relates to message length. Some older implementations cannot handle messages exceeding 64 KB.

Another problem relates to timeouts. If the client and server have different timeouts, one of them may give up while the other is still busy, unexpectedly terminating the connection.

Finally, in rare situations, infinite mailstorms can be triggered.

For example, if host 1 holds mailing list *A* and host 2 holds mailing list *B* and each list contains an entry for the other one, then a message sent to either list could generate a never-ending amount of e-mail traffic unless somebody checks for it.

FINAL DELIVERY

With the advent of people who access the Internet by calling their ISP over a modem, it breaks down.

One solution is to have a message transfer agent on an ISP machine accept e-mail for its customers and store it in their mailboxes on an ISP machine. Since this agent can be on-line all the time, e-mail can be sent to it 24 hours a day.

POP3

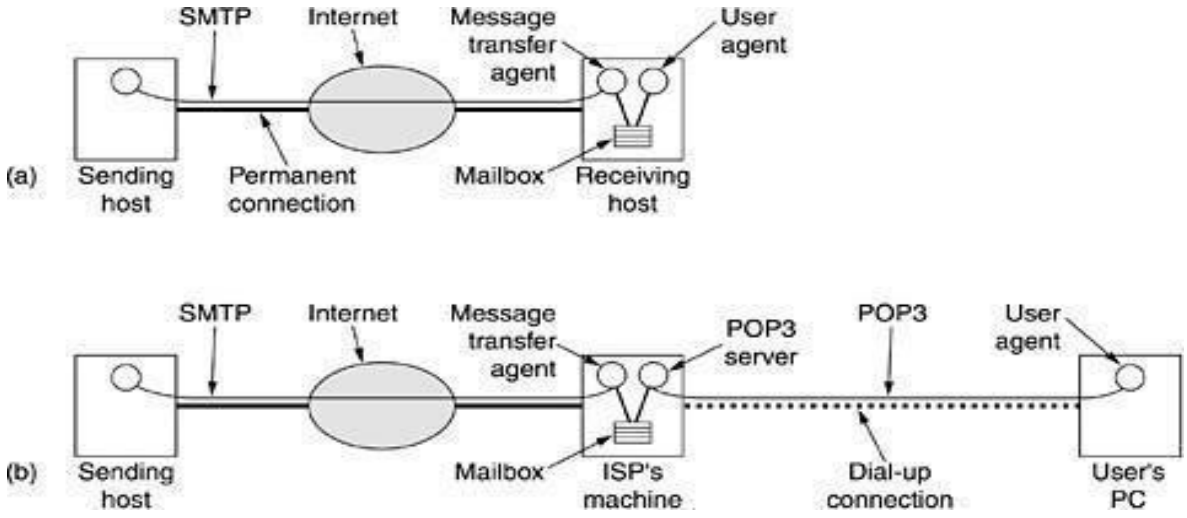


Figure:5-7

(a) Sending and reading mail when the receiver has a permanent Internet connection and the user agent runs on the same machine as the message transfer agent.

(b) Reading e-mail when the receiver has a dial-up connection to an ISP

POP3 begins when the user starts the mail reader. The mail reader calls up the ISP (unless there is already a connection) and establishes a TCP connection with the message transfer agent at port 110. Once the connection has been established, the POP3 protocol goes through three states in sequence:

1. Authorization.
2. Transactions.
3. Update.

The authorization state deals with having the user log in.

The transaction state deals with the user collecting the e-mails and marking them for deletion from the mailbox. The update state actually causes the e-mails to be deleted.

IMAP (**Internet Message Access Protocol**).

POP3 normally downloads all stored messages at each contact, the result is that the user's e-mail quickly gets spread over multiple machines, more or less at random; some of them not even the user's.

This disadvantage gave rise to an alternative final delivery protocol, **IMAP** (**Internet Message Access Protocol**).

IMAP assumes that all the e-mail will remain on the server indefinitely in multiple mailboxes. IMAP provides extensive mechanisms for reading messages or even parts of messages, a feature useful when using a slow modem to read the text part of a multipart message with large audio and video attachments.

DNS PROTOCOL

DNS is an abbreviation of Domain Name System or Domain Name Service. It is an application layer protocol.

Basically, a Domain name system is a supporting program that is used by other programs such as an E-mail.

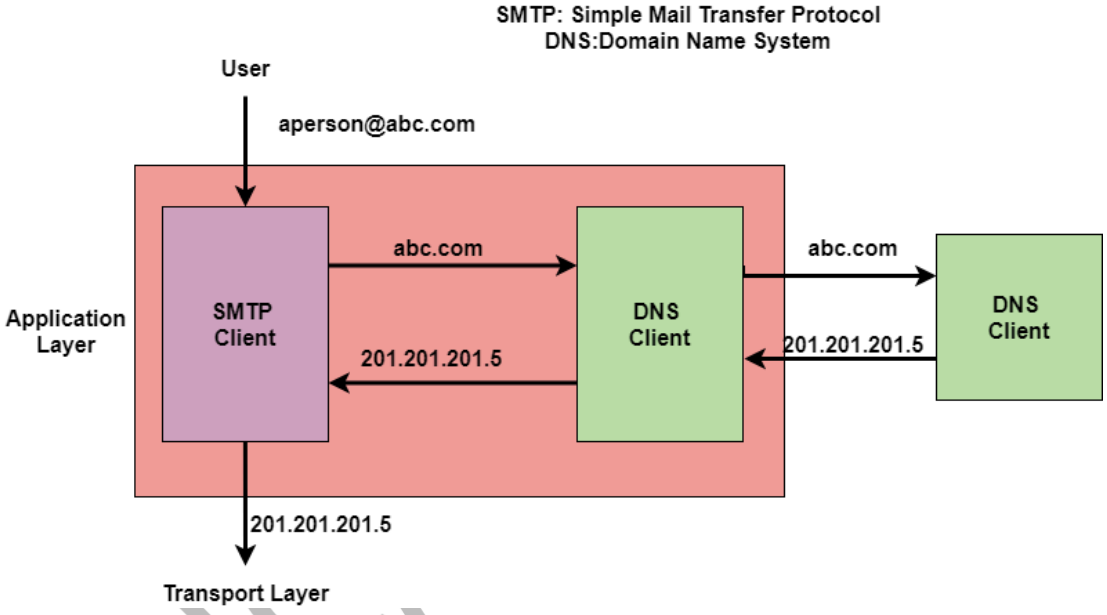
The user of the email program knows the email address of the recipient; the Internet protocol needs the IP address.

Mainly the DNS client program sends a request to the DNS server in order to map the e-mail address to the corresponding IP address.

In order to identify an entity, the TCP/IP protocols also make use of an IP address that uniquely identifies the connection of the host to the internet. But people usually prefer to use names instead of numeric addresses. Thus there is a need for the system that can map a name to an address or an address to a name. Domain Name System is a system that can map a name to an address or an address to a name.

Examp

Given



Name Space

Namespace basically maps each address to a unique name. The names assigned to the machines must be unique because addresses are unique.

It is further categorized into two:

Flat Name Space

Hierarchical Name Space

Flat Name Space

In the Flat Name Space basically, a name is assigned to an address.

A name in this space is basically a sequence of characters without any structure.

Also, the names may or may not have a common section. In case if they have a common section then it has no meaning.

One of the main disadvantages of this system is that it cannot be used in the case of large systems; because there is no central control and it will lead to ambiguity and duplication.

Hierarchical Name Space

In Hierarchical Name Space each name consists of several parts.

The first part mainly indicates the nature of the organization.

The second part mainly indicates the name of the organization.

The third part mainly defines the departments in the organization and so on.

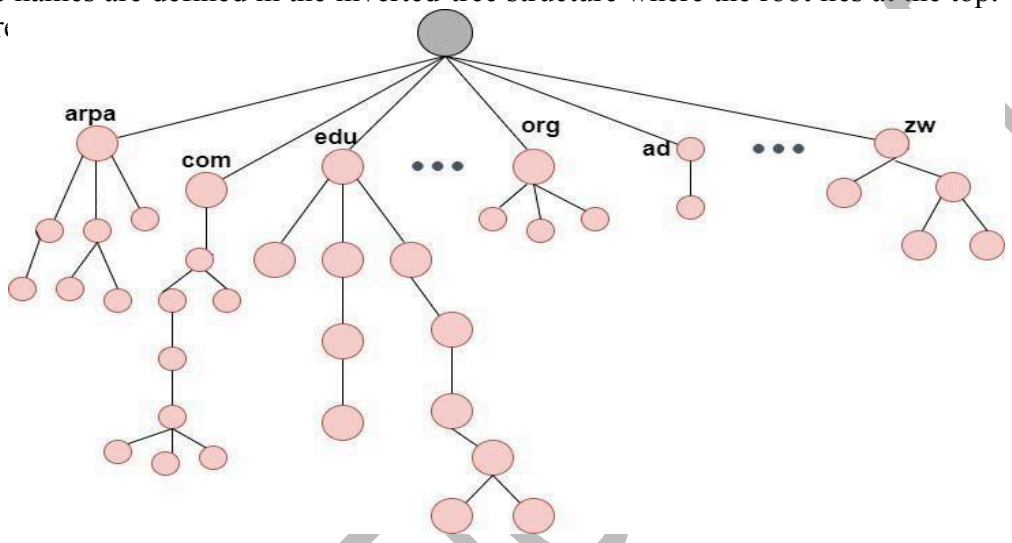
The central authority can assign the part of the name that indicates the name and nature of the organization and the responsibility of the rest of the name is given to the organization itself.

An organization can also add suffixes (or prefixes) to the name in order to define the host or resources.

Domain Name Space

When we use the hierarchical Name Space in that case we need to design the **Domain Name Space**. In this Design, the names are defined in the inverted-tree structure where the root lies at the top.

Also, the tr



Label

Each node of the tree must have a label. A Label is a string having a maximum of 63 characters.

The root label is basically a null string(means an empty string).

Domain Name Space requires that the children of the node that means branches from the same node should have different labels and this guarantees the uniqueness of the domain names.

Domain Name

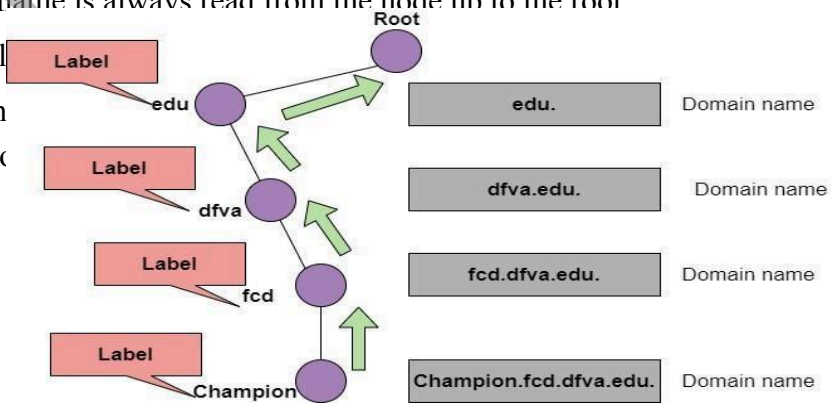
Each node of the tree has a domain name.

A Full domain name is basically a sequence of labels that are usually separated by dots (.).

The domain name is always read from the node up to the root

The last label

All this mean



neans that the last character is always a d

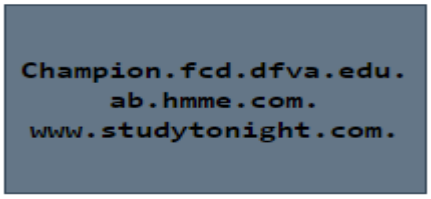
The figure shows the domain names and labels

Domain Names are further categorized into two:

1. Fully Qualified Domain Name

- If the label is terminated by the null string then it is known as a fully qualified domain name. This domain name contains the full name of the host.
- FQDN mainly consists of two parts: **hostname** and **domain name**.
- The FQDN mainly contains all the labels from the most specific one to the most general one that helps to uniquely define the name of the host.
- Example: Champion.fcd.dfva.edu. in this the hostname is

Champion. Given below are some Fu



2. Partially Qualified Domain Name

If the label is not terminated by the null string then it is known as Partially Qualified Domain Name.

- This name starts from the node but does not reach the root.
- It is mainly used when the name to be resolved belongs to the same site as the client and in this case, the resolver can supply the missi reate an FQDN.

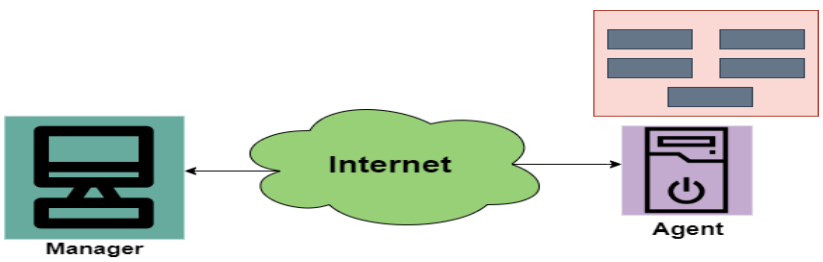
Domain

A Domain is basically the sub tree of the Domain Name Space. The name of the domain is usually the domain name of the node that is at the top of the sub tree.

SNMP Protocol

- SNMP mainly stands for Simple Network Management protocol.
- It is basically a framework that is used for managing the devices on the internet by using the TCP/IP protocol suite.
- Basically, SNMP provides a set of fundamental operations in order to monitor and maintain the Internet.
- It is an application layer protocol that was defined by the Internet engineering task force.
- This protocol is mainly used to monitor the network, detect the faults in the Network, and sometimes it is also used to configure the remote devices.

Concept of SNMP



The SNMP protocol makes the use of Manager and Agent; where the manager is usually a host that controls and monitors the set of agents.

- The SNMP is an application-level protocol and it consists of a few manager stations that mainly controls a set of agents. This protocol is mainly designed at the application level so that it can monitor the devices that are mainly made by different manufacturers and that are installed on different physical networks.
  Thus there are three components in the architecture of the SNMP:
- SNMP Manager
- SNMP Agent
- Management Information Base

SNMP Manager

It is basically a centralized system and it is mainly used to monitor and manage devices that are connected with the network. SNMP manager is typically a computer and it is used to run one or more network management systems.

Given below are the main functions of SNMP Manager:

1. Collects response from the agents.
2. To acknowledge asynchronous events from the agents.
3. To set variables in the agent.
4. Queries the Agent

SNMP Agent

SNMP Agent is basically a software program that is packaged within the network element. It is mainly installed on a managed device where managed devices can be switches, servers, routers, PC, etc.

Mainly the agents keep the information in the database also the manager has the access to the values present in the database.

Given below are the main responsibilities of the SNMP Agent:

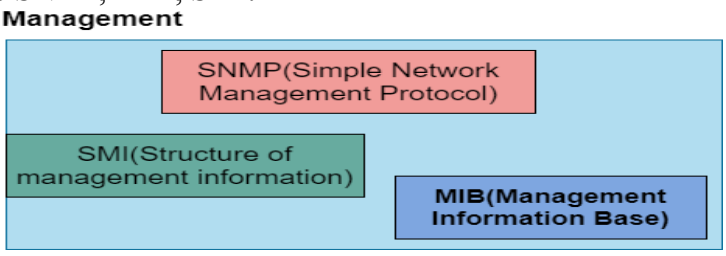
- SNMP agents mainly collect the management information about its local environment
- The SNMP agent mainly signals an event to the manager.
- The SNMP agents also act as a proxy for some non–SNMP manageable network

nodes. Thus the management with SNMP is mainly based on these given ideas:

1. An SNMP manager checks the agent by requesting information that mainly reflects the behavior of the SNMP agent.
2. The SNMP manager also forces the agent to perform the task by resetting the values in the database of the agent.
3. Management process is also contributed by the agent just by warning the SNMP manager about an unusual situation.

Management Components

In order to perform the Management tasks, the SNMP protocol makes the use of two other protocols and are SMI and MIB. We can also say that the Management on the Internet is done by the cooperation of three protocols and these are SNMP, MIB, SMI.



Let us discuss their roles one by one;

Role of SNMP

The SNMP protocol performs some specific roles in Network Management;

- It mainly defines the format of the packet that needs to be sent from the manager to the agent or vice-versa.
- SNMP is also used to interpret the result and create the statistics.
- The packets that are exchanged between the manager and agent contains the name of the object(variable) and their status(values).
- The SNMP is also responsible for reading and changing these values.

Role of SMI

In order to use the SNMP, there is a need for some rules and these rules are for naming the objects. Now its time to take a look at the roles of SMI:

- SMI (Structure of Management Information) is mainly used to define the general rules for naming the objects.
- It is also used to define the type of objects that includes (range and length).
- This is also used to show how to encode the objects and values.
- The SMI does not define the number of objects that should be managed by an entity.
- It also does not define the association between the objects and their values.

Role of MIB

In order to manage each entity, this protocol is mainly used to define the number of objects and then to name them according to the rules defined by the SMI and after that associate a type to each named object.

- MIB (Management Information Base) is mainly used to create a set of objects that are defined for each entity that is similar to the database.
- Thus MIB mainly creates a collection of named objects, their types.

Advantages of SNMP Protocol

Given below are some of the benefits of using SNMP :

1. It is the standard network management protocol.
2. This protocol is independent of the operating system and programming language.
3. The functional design of this protocol is Portable.
4. The SNMP is basically a core set of operations and it remains the same on all managed devices. Thus SNMP supports extendibility.
5. SNMP is a universally accepted protocol.
6. It is a lightweight protocol.
7. This protocol allows distributed management access.

Disadvantages

Some of the drawbacks of SNMP are as follows:

- This protocol leads to the reduction of the bandwidth of the network.
- Access control, authentication, and privacy of data are some largest security issues using this.
- SNMP deals with information that is neither detailed nor enough well organized

ELECTRONIC MAIL

Electronic mail is often referred to as E-mail and it is a method used for **exchanging digital messages**.

- Electronic mail is mainly designed for **human use**.
- It allows a message to includes **text, image, audio** as well as **video**.
- This service allows one message to be **sent to one or more than one recipient**.
- The E-mail systems are mainly based on the **store-and-forward model** where the E-mail server system accepts, forwards, deliver and store the messages on behalf of users who only need to connect to the infrastructure of the Email.
- The Person who **sends the email** is referred to as **the Sender** while the person who receives an email is referred to as **the Recipient**.

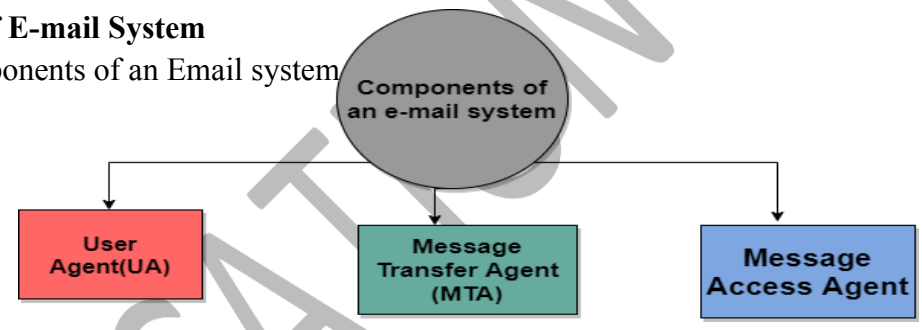
Need of an Email

By making use of Email, we can send any message at any time to anyone.

- We can send the same message to several peoples at the same time.
- It is a very fast and efficient way of transferring information.
- The email system is very fast as compared to the Postal system.
- Information can be easily forwarded to coworkers without retyping it.

Components of E-mail System

The basic Components of an Email system



1. User Agent (UA)

It is a program that is mainly used to send and receive an email. It is also known as an email reader. User-Agent is used to compose, send and receive emails.

- It is the first component of an Email.
- User-agent also handles the mailboxes.
- The User-agent mainly provides the services to the user in order to make the sending and receiving process of message easier.

Given below are some services provided by the User-Agent:

1. Reading the Message 2.Replying the Message 3.Composing the Message 4.Forwarding the Message. 5.Handling the Message.

2. Message Transfer Agent

The actual process of transferring the email is done through the Message Transfer Agent (MTA).

- In order to send an Email, a system must have an MTA client.
- In order to receive an email, a system must have an MTA server.
- The protocol that is mainly used to define the MTA client and MTA server on the internet is called SMTP

(Simple Mail Transfer Protocol).

- The SMTP mainly defines how the commands and responses must be sent back and forth

3. Message Access Agent

In the first and second stages of email delivery, we make use of SMTP.

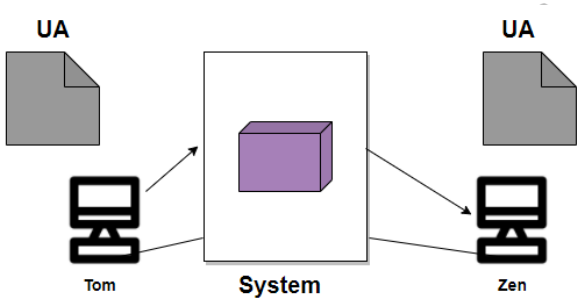
- SMTP is basically a Push protocol.
- The third stage of the email delivery mainly needs the pull protocol, and at this stage, the message access agent is used.
- The two protocols used to access messages are POP and IMAP4.

Architecture of Email

Now its time to take a look at the architecture of e-mail with the help of four scenarios:

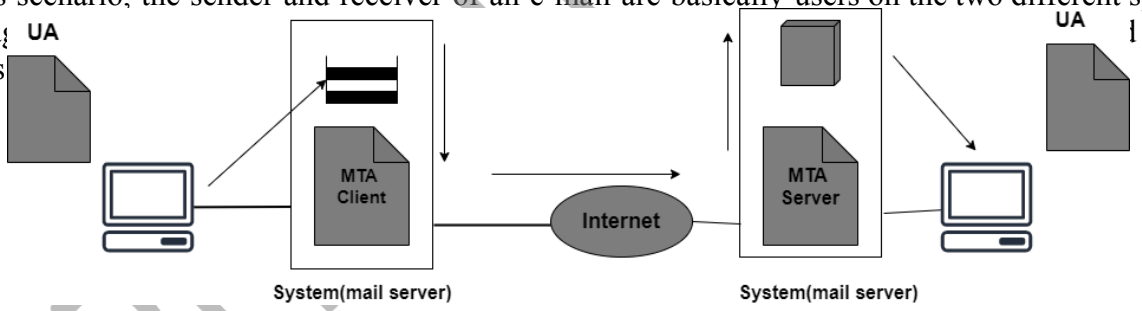
First Scenario

When the sender and the r there is the need for only two user agents.



Second Scenario

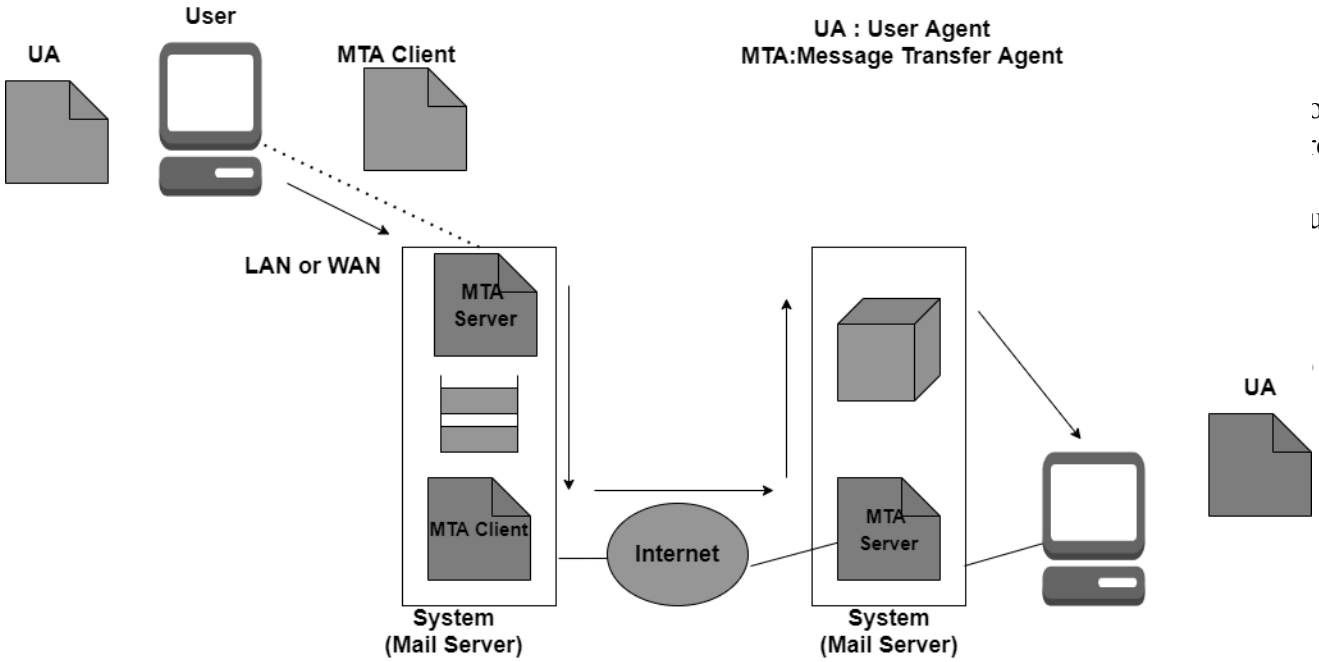
In this scenario, the sender and receiver of an e-mail are basically users on the two different systems. Also, the messa Message transfer agents



Third Scenario

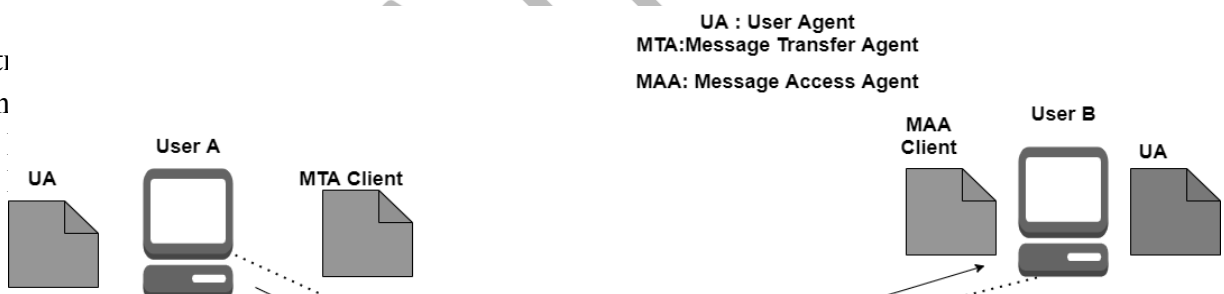
In this scenario, the sender is connected to the system via a point-to-point WAN it can be either a dial-up modem or a cable modem. While the receiver is directly connected to the system like it was connected in the second scenario.

Also in this case sender needs a User agent(UA) in order to prepare the message. After preparing the message the sender sends the message via a pair of MTA through LAN or WAN.



other set of
retrieve the

uest for the



Header

The header part of the email generally contains the sender's address as well as the receiver's address and the subject of the message.

Body

The Body of the message contains the actual information that is meant for the receiver.

Email Address

In order to deliver the email, the mail handling system must make use of an addressing system with unique addresses.

The address consists of two parts:

- Local part
- Domain Name

Local Part

It is used to define the name of the special file, which is commonly called a user mailbox; it is the place where all the mails received for the user is stored for retrieval by the Message Access Agent.

Domain Name

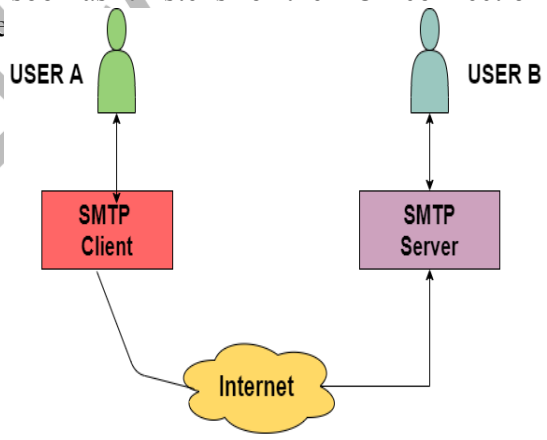
It is the second part of the address is Domain Name.

Both local part and domain name are separated with the help of @.

SMTP PROTOCOL

SMTP mainly stands for Simple Mail Transfer Protocol. Basically, the actual transfer of mail is done through the message transfer agents(MTA). Thus in order to send the mail, the system must have the **client MTA** and in order to receive the mail, the system must have a server MTA.

- In order to define the **MTA client** and **server** on the Internet, there is a formal way and it is known as **Simple Mail Transfer Protocol (SMTP)**.
- SMTP also makes the use of TCP/IP for sending and receiving e-mail.
- SMTP is based on the client/server model.
- The original standard port for SMTP is Port 25.
- Using this protocol, the client who wants to send the e-mail first opens a TCP connection to the SMTP server and then sends the e-mail across the TCP connection. It is important to note that the SMTP server is always in listening mode. As soon as it listens for the TCP connection from any client then the connection is Initiated on port 25 and afte ls the e-mail/message immediately.



SMTP is used two times while sending an Email:

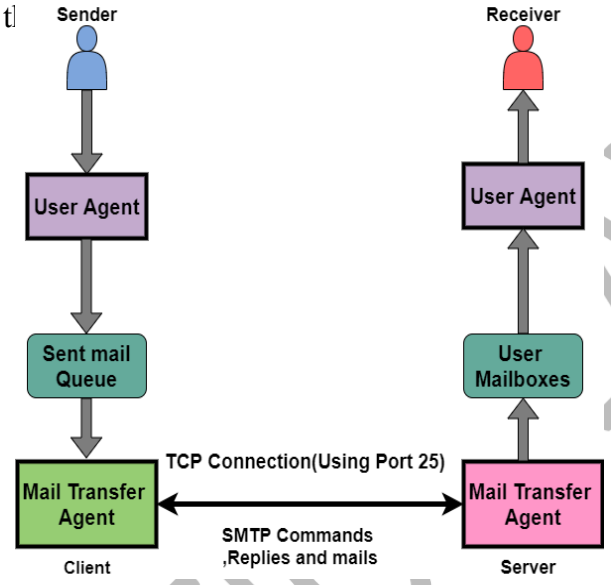
1. Between the Sender and Sender's mail server
2. Between the Sender's mail server and the Receiver's mail server It is important to note that in order to receive or download the email,

- There is a need for another protocol between the mail server of receiver and the receiver.
- Commonly used protocols are POP3 and IMAP. Thus these two are mail access agents.

Architecture of SMTP

All the users make use of **User Agent (UA)**. The Mail Transfer Agent **(MTA)** mainly helps to exchange all the messages in between both sender and receiver using the TCP/IP. The system administrator has the authority to configure the set up of local MTA, thus the users who are sending the email do not need to deal with the MTA. The MTA keeps the queue in the pool of messages; if the receiver is not available at that moment then MTA can schedule the repeat delivery of all the messages.

MTA (Mail User Agent) forwards the emails into mailboxes of the user's local system, and then the user agent (UA) can download t



The SMTP Client as well as the SMTP server both has two main components and these are:

- UA(User-Agent)
- MTA(Mail Transfer Agent)

Let us now take a look at communication between the sender and the receiver:
The user agent at the sender side prepares the message and then sent it to the MTA. The task of the MTA is to transfer the Email across the network to the Receiver MTA. Also in order to send the Email, a system must have the client MTA and in order to receive the email, a system must have a server MTA.

Sending the Email

An email is sent between the sender and receiver using a series of request and response messages. An Email mainly consists of two parts **a header and body**. The body part of an email indicates the main message area. It is the actual information that is to be read by the receiver. The header mainly contains the address of the sender and recipient and it also contains the subject of the email.
In order to terminate the header of the email, there is a NULL line, everything after the NULL line is considered as the body of the message.

Receiving the Email

Mailboxes are checked by the user agent at the server side at a particular interval of time. In case if any information is received then it informs the receiver about the email.
At the time when the user tries to read the email then MTA mainly displays a list of emails with their short description in the mailbox. If the user selects any of the emails then can easily view the contents inside the email.

SMTP Protocol Method

1. **Store-and-Forward Method** The store and forward method is used within an organization.
2. **End-to-End Method** Mainly the end-to-end method is used to communicate between the different organizations

An SMTP client is the one who wants to send the mail and will definitely contact the destination's host SMTP directly in the order to send the Email to the destination. Also, the session is initiated by the client SMPT.
On the other hand, the SMTP server will keep the mail to itself until it is successfully copied to the SMTP at the receiver. The server SMTP mainly responds to the session request.
Thus the session is started by the client-SMTP and the server-SMTP will respond to the request of the sender.

Characteristics of SMTP

Let us take a look at the characteristics of the SMTP:

- SMTP makes use of Port 25.
- It makes use of persistent TCP connections and thus can send multiple emails all at once.
- It is a stateless protocol.
- It is a connection-oriented protocol.
- It makes use of TCP at the transport layer.
- It is a push control protocol.

Advantages of SMTP

Let us take a look at the advantages offered by the simple mail transfer protocol(SMTP):

- SMTP offers reliability in terms of the outgoing email messages.
- It is the simplest form of communication between various computers in a network via Email.
- In those cases where a particular message was not **delivered successfully** then, the SMTP server always tries to re-send the same message until the **transmission** becomes **successful**.

Disadvantages of SMTP

- SMTP does not provide good security.
- It is only limited to 7-bit ASCII characters.
- Beyond some specific length, email messages are rejected by SMTP servers.
- The usefulness of SMTP is limited by its simplicity.
- With the help of SMTP, the transmission of executable files and binary files is not possible until they get converted into text files.

POP PROTOCOL

POP is a short form of Post Office Protocol. It is another protocol present at the Application Layer of the OSI reference model.

- POP is mainly a message access protocol.
- POP is basically an internet standard protocol and as we already told you it works on the application layer and is used by the local email software in order to retrieve emails from the remote email server over the TCP/IP connection.
- The Post office Protocol (POP) does not allow any search facility.
- This protocol mainly allows one protocol to be created on the server.
- As this protocol supports offline access to the messages and so less internet usage time is required by this.
- Non-email data is not accessed by this protocol.
- Some of the common clients that make use of POP3 are Gmail, Netscape, Internet Explorer, Eudora.

History of POP

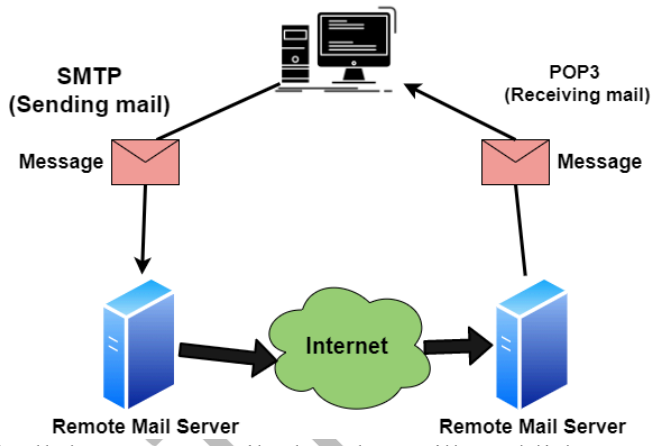
The POP (post office protocol) was published in 1984 by Internet Engineering Task Force. After that, it has

been updated two times, because the backend developers want to make the layout simple.

The second version of POP was developed in 1985 and known as POP2 and this version needs the SMTP protocol in order to push the emails.
Then after the third version of POP was released in 1988 and known as POP3, this version does not require the SMTP protocol. The **POP (Post office protocol version 3)** is also integrated into famous e-mail software, like Eudora and Outlook Express.
And since then (1988) the POP3 is the active version.

Working of POP

All the incoming messages are stored on the POP server until the user login by using an email client and download the message to their computer. After the message is downloaded by the user it gets deleted from the server.
As we know that the SMTP is used to transfer the email message from the server to the server, basically POP is used to collect the email with an email client from the server and it does not include means to send messages.



If any user tries to check all the recent emails then they will establish a connection with the **POP3** at the server-side. The user sends the username and password to the server machine for getting the proper authentication. After getting the connection, users can receive all text-based emails and store them on their local terminal (machine), then finally discard all server copies and then breaks the connection from the server machine.
In order to retrieve a message from the server following steps are taken;

- Firstly a TCP connection is established by the client using port 110.
- The client identifies itself to the server.
- After that client issues a series of POP3 commands.

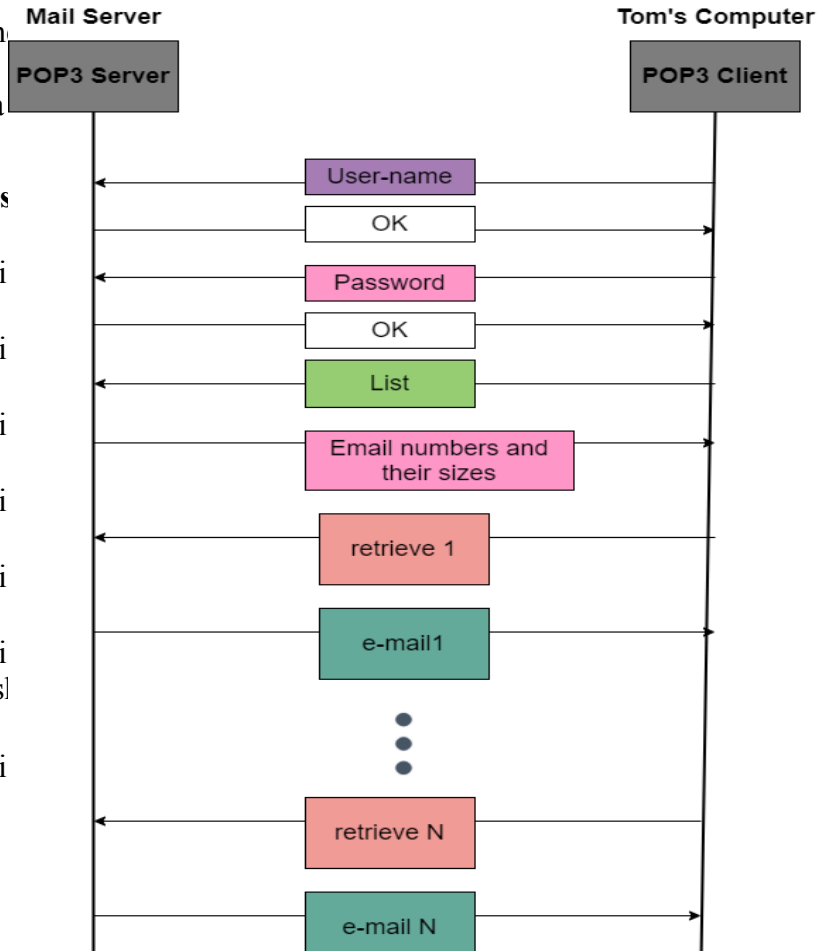
Features of POP protocol

Given below are some of the features provided by the POP protocol:

- The POP protocol uses PORT 110.
- It makes the use of a Persistent TCP connection.
- It is a Pull protocol.
- It is a connection-oriented protocol.
- The POP protocol is a stateful protocol until the mail is downloaded and across the sessions, it is a stateless protocol.

Th ses in the POP3

Let us now take a



Commands	Des
LOGIN	Thi
STAT	Thi the mailbox.
DELE	Thi
RSET	Thi
QUIT	Thi
LIST	Thi where each message summary is s
RETR	Thi the messages.

Advantages of POP

Given below are the advantages offered by the POP :

- This protocol does not require any internet connection in order to access the downloaded emails.
- In order to receive emails on a single device, POP3 is very useful.
- The Configuration of this protocol is simple and it is easy to use.
- Less storage space is needed in order to store emails on the hard disk.
- This protocol is much better for the ones who hardly check their email on any other computer.

Disadvantages of POP

Now it's time to take a look at the drawbacks of Post office Protocol(POP):

- The same email account cannot be accessed from multiple computers or devices.
- The spread of the virus is easily using this protocol because it is possible that the file attached with the email contains the virus.
- The transfer of the local email folder to another email client terminal point is a difficult task.

WORLD WIDE WEB

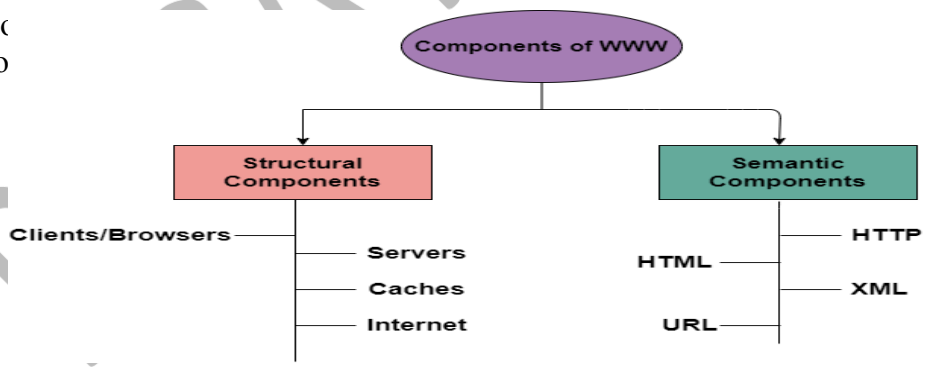
The **World Wide Web** or Web is basically a collection of information that is linked together from points all over the world. It is also abbreviated as **WWW.**

- World Wide Web provides flexibility, portability, and user-friendly features.
- It mainly consists of a worldwide collection of electronic documents (i.e, Web Pages).
- It is basically a way of exchanging information between computers on the Internet.
- The WWW is mainly the network of pages consists of images, text, and sounds on the Internet which can be simply viewed on the browser by using the browser software.
- It was invented by Tim Berners-Lee.

Components of WWW

The Components of WWW mainly falls into two categories:

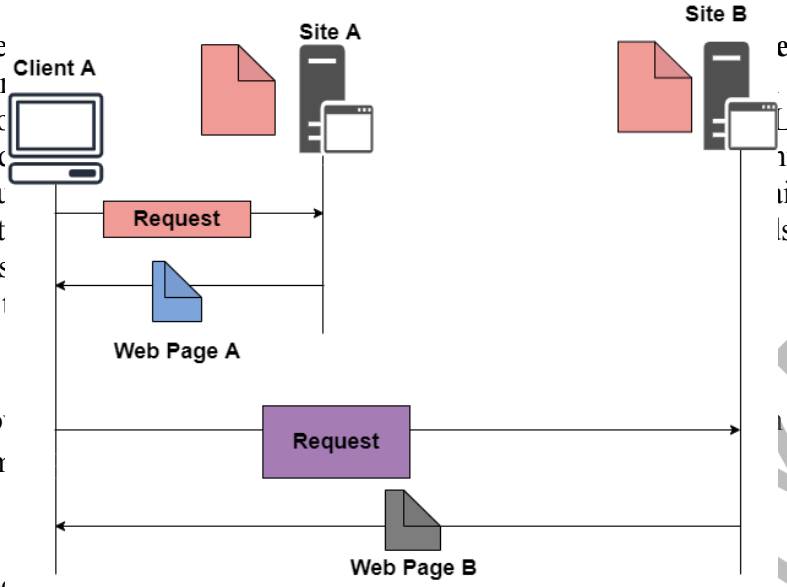
1. Structural Cc
2. Semantic Co



Architecture of WWW

The **WWW** is mainly a distributed **client/server** service where a client using the browser can access the service using a server. The Service that is provided is distributed over many different locations commonly known as **sites/websites.**

- Each website holds one or more documents that are generally referred to as **web pages.**
- Where each web page contains a link to other pages on the same site or at other sites.
- These pages can be retrieved and viewed by using browsers.



In the above e **A**. It generally sends a request through its browse the web) and also the request generally contains L).The server at **site A** finds the document then sen nt finds the reference to another document that inclu ains the URL of site B. And the client is interested t s the request to the new site and then the new page is Now we will cover

1. Client/Browser

The Client/Web bro the webserver on the Internet.

- Each browser n
  - o Controller
  - o Interpreter
  - o Client Protocols
    - The Controller mainly receives the input from the input device, after that it uses the client programs in order to access the documents.
    - After accessing the document, the controller makes use of an interpreter in order to display the document on the screen.
    - An interpreter can be Java, HTML, and JavaScript mainly depending upon the type of the document.
    - The Client protocol can be FTP, HTTP, and TELNET.

2. Server

The Computer that is mainly available for the network resources and in order to provide services to the other computer upon request is generally known as the **server.**

- The Web pages are mainly stored on the server.
- Whenever the request of the client arrives then the corresponding document is sent to the client.
- The connection between the client and the server is TCP.
- It can become more efficient through multithreading or multiprocessing. Because in this case, the server can answer more than one request at a time.

3. URL

URL is an abbreviation of **the Uniform resource locator.**

- It is basically a standard used for specifying any kind of information on the Internet.
- In order to access any page the client generally needs an address.
- To facilitate the access of the documents throughout the world HTTP generally makes use of Locators. URL mainly defines the four things:
- **Protocol** It is a client/server program that is mainly used to retrieve the document. A commonly used protocol is HTTP.
- **Host Computer** It is the computer on which the information is located. It is not mandatory because it is the name given to any computer that hosts the web page.
- **Port** The URL can optionally contain the port number of the server. If the port number is included then it is generally inserted in between the host and path and is generally separated from the host by the colon.
- **Path** It indicates the pathname of the file where the information is located.

4. HTML

HTML is an abbreviation of Hypertext Markup Language.

- It is generally used for creating web pages.
- It is mainly used to define the contents, structure, and organization of the web page.

5. XML

XML is an abbreviation of Extensible Markup Language. It mainly helps in order to define the common syntax in the semantic web.

Features of WWW

Given below are some of the features provided by the World Wide Web:

- Provides a system for Hypertext information
- Open standards and Open source
- Distributed.
- Mainly makes the use of Web Browser in order to provide a single interface for many services.
- Dynamic
- Interactive
- Cross-Platform

Advantages of WWW

Given below are the benefits offered by WWW:

- It mainly provides all the information for Free.
- Provides rapid Interactive way of Communication.
- It is accessible from anywhere.
- It has become the Global source of media.
- It mainly facilitates the exchange of a huge volume of data.

Disadvantages of WWW

There are some drawbacks of the WWW and these are as follows;

- It is difficult to prioritize and filter some information.
- There is no guarantee of finding what one person is looking for.
- There occurs some danger in case of overload of Information.
- There is no quality control over the available data.
- There is no regulation.

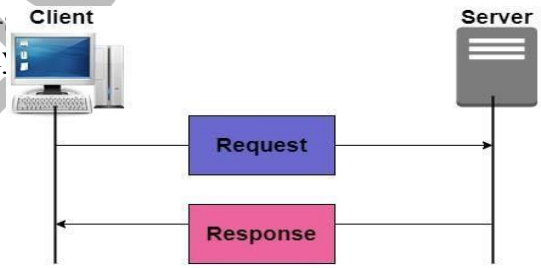
HTTP PROTOCOL

HTTP stands for Hypertext Transfer Protocol and is mainly used to access the data on the world wide web i.e **(WWW)**. The **HTTP** mainly functions as the combination of **FTP**(File Transfer Protocol) and **SMTP**(Simple Mail Transfer Protocol).

- **HTTP** is one of the protocols used at the **Application Layer**.
- The **HTTP** is similar to **FTP** because **HTTP** is used to transfer the files and it mainly uses the services of **TCP.**
- Also, **HTTP** is much simpler than **FTP** because there is only **one TCP connection**.
- In **HTTP,** there is no separate control connection, as only data is transferred between the client and the server.
- The **HTTP** is like SMTP because the transfer of data between the client and server simply looks like **SMTP** messages. But there is a difference unlike **SMTP,** the **HTTP** messages are not destined to be read **by humans** as they are read and interpreted by **HTPP Client**(that is browser) and HTTP server.
- Also, **SMTP** messages are **stored and then forwarded** while the **HTTP** messages are **delivered immediately.**
- The **HTTP** mainly uses the services of the **TCP** on the well-known port that is **port 80.**
- **HTTP** is a **stateless protocol.**
- In **HTTP, the** client initializes the transaction by sending a request message and the server replies by sending a response.
- This protocol is used to transfer the data in the form of plain text, hypertext, audio as well as video, and so on.

Working of HTTP

The HTTP makes use of Client-Server Architecture. We already told you that the browser acts as the HTTP client and this client mainly requests the server which is hosting the website.



The figure shows the HTTP transaction

The format of the request and the response message is similar. The Request Message mainly consists of a request line, a header, and a body sometimes. A Response message consists of the status line, a header, and sometimes a body.

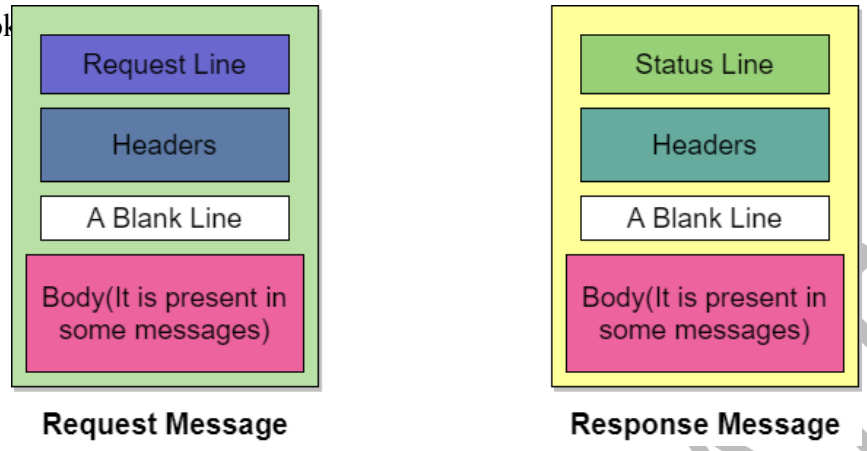
At the time when a client makes a request for some information (say client clicks on the hyperlink) to the web server. The browser then sends a request message to the HTTP server for the requested objects.

After that the following things happen:

- There is a connection that becomes open between the client and the webserver through the TCP.
- After that, the HTTP sends a request to the server that mainly collects the requested data.
- The response with the objects is sent back to the client by HTTP

- At last, HTTP closes the connection.

Let us take a look



Request Line and Status line

The first line in the Request message is known as the request line, while the first line in the Response message is known as

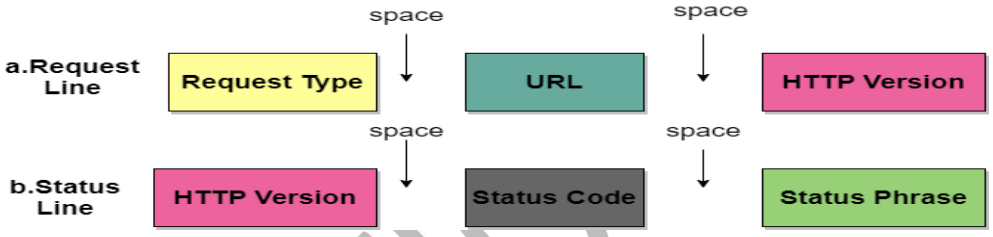


Figure: Request Line and Status Line

Request Type

This field is used in the request line. The are several request types that are defined and these are mentioned in the table given below;

Name of Method	Actions
GET	This method is used to request a document from the server.
HEAD	This method mainly requests information about a document and not the document itself
POST	This method sends some information from the client to the server.
PUT	This method sends a document from the server to the client.
TRACE	This method echoes the incoming request.
CONNECT	This method means reserved

Name of Method	Actions
OPTION	In order to inquire about the available options.

URL

URL is a Uniform Resource locator and it is mainly a standard way of specifying any kind of information on the Internet.

HTTP Version

The current version of the HTTP is 1.1.

Status Code

The status code is the field of the response message.The status code consists of three digits.

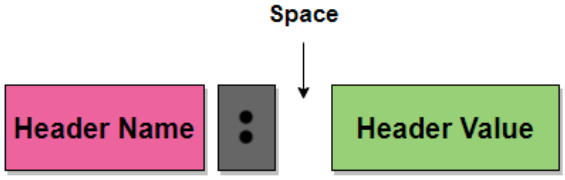
Status Phrase

This field is also used in the response message and it is used to explain the status code in the form of text.

Header

The header is used to exchange the additional information between the client and the server. The header mainly consists of one or more header lines. Each header line has a header name, a colon, space, and a header value. The header line is further categorized into four:

- **General Header** It provides general information about the message and it can be present in both request and response.
- **Request Header** It is only present in the request message and is used to specify the configuration of the client and the format of the document preferred by the client
- **Response Header** This header is only present in the response header and mainly specifies the configuration of the server and also the special information about the request.
- **Entity Header** It is u ocument.



Body

It can be present in the request message or in the response message. The body part mainly contains the document to be sent or received.

Features of HTTP

The HTTP offers various features and these are as follows:

1. **HTTP is simple** The HTTP protocol is designed to be plain and human-readable.
2. **HTTP is stateless** Hypertext transfer protocol (HTTP) is a stateless protocol, which simply means that there is no connection among two requests that are being consecutively carried out on the same connection. Also, both the client and the server know each other only during the current requests and thus the core of the HTTP is itself a stateless one, On the other hand, the HTTP cookies provide in making use of stateful sessions.
3. **HTTP is extensible** The HTTP can be integrated easily with the new functionality by providing a simple agreement between the client and the server.

4. **HTTP is connectionless** As the HTTP request is initiated by the browser (HTTP client) and as per the request information by the user, after that the server processes the request of the client and then responds back to the client

Advantages of HTTP

Given below are the benefits of using HTTP:

1. There is no runtime support required to run properly.
2. As it is connectionless so there is no overhead in order to create and maintain the state and information of the session.
3. HTTP is usable over the firewalls and global application is possible.
4. HTTP is platform-independent.
5. HTTP reports the errors without closing the TCP connection.
6. Offers Reduced Network congestions.

Disadvantages of HTTP

There are some drawbacks of using the HTTP protocol:

- HTTP is not optimized for mobile.
- HTTP is too verbose.
- It can be only used for point-to-point connections.
- This protocol does not have push capabilities.
- This protocol does not offer reliable exchange without the retry logic.

The HTTP supports proxy servers. A proxy server is basically a computer that keeps the copies of the responses to recent requests. The proxy server mainly reduces the load on the originals server. In order to use the proxy server, the client must be configured in order to access the proxy instead of the target server.

HTTP Connections

HTTP connections can be further classified into two:

- Persistent Connection
- Nonpersistent Connection

Let us discuss them one by one:

1. Persistent Connection

In the persistent HTTP connection, all the requests and their corresponding responses are sent over the same TCP connections. The 1.1 version of the HTTP specifies a persistent connection by default.

In this type of connection, the server leaves the connection open for more requests after sending a response. Also, the server can close the connection at the request of the client or upon reaching the time-out.

In a Persistent connection, a single TCP connection is mainly used for sending multiple objects one after the other.

Usually, the length of the data is sent along with each response. There are some cases when the server does not know the length of the data this happens when the document is created dynamically and in such cases, the server informs the client that length is not known and closes the connection after sending the data so in order let the client Inform about the end of the data.

2. Nonpersistent Connection

In the Nonpersistent HTTP connection, one TCP connection is made for each request/response; it means there is a separate for each object.

Following are the steps used;

- The client opens a TCP connection and then sends a request.

- After that, the server sends the response and then closes the connection.
- Then the client reads the data and until it encounters an end-of-file marker then it closes the connection.

This connection imposes a high overhead on the server because N different buffers are required by the server, and the start procedure is slow each time when a connection is opened.
The nonpersistent connection is supported by the HTTP 1.0 version.