srivenkateshwaraa
College of Engineering & Technology
(Approved by AICTE, New Delhi & Affiliated to Pondicherry University, Puducherry)
13-A, Pondy - Villupuram Main Road, Ariyur, Puducherry - 605 102.
ASPIRE TO EXCEL

ISO
21001

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

EC T52   Data Communication Networks

III YEAR/ V SEM

UNIT-I

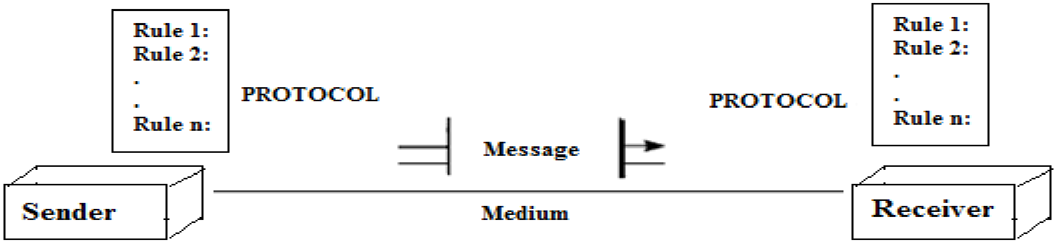
Network Models

DATA COMMUNICATIONS: Data communications are the exchange of data between two devices via some form of transmission medium such as a wire cable. For data communications to occur, the communicating devices must be part of a communication system made up of a combination of hardware (physical equipment) and software (programs). The word *data* refers to information presented in whatever form is agreed upon by the parties creating and using the data. The effectiveness of a data communications system depends on four fundamental characteristics:

1. **Delivery.** The system must deliver data to the correct destination. Data must be received by the intended device or user and only by that device or user.
2. **Accuracy.** The system must deliver the data accurately. Data that have been altered in transmission and left uncorrected are unusable.
3. **Timeliness.** The system must deliver data in a timely manner. Data delivered late are useless. In the case of video and audio, timely delivery means delivering data as they are produced, in the same order that they are produced, and without significant delay. This kind of delivery is called *real-time* transmission.
4. **Jitter.** Jitter refers to the variation in the packet arrival time. It is the uneven delay in the delivery of audio or video packets.

A data communications system has five components (see Figure 1.1).

Figure 1.1 Five components of Data Communication



1. **Message.** The message is the information (data) to be communicated. Popular forms of information include text, numbers, pictures, audio, and video.
2. **Sender.** The sender is the device that sends the data message. It can be a computer, workstation, telephone handset, video camera, and so on.
3. **Receiver.** The receiver is the device that receives the message. It can be a computer, workstation, telephone handset, television, and so on.
4. **Transmission medium**. The transmission medium is the physical path by which a message travels from sender to receiver. Some examples of transmission media include twisted-pair wire, coaxial cable, fiber-optic cable, and radio waves.
5. **Protocol.** A protocol is a set of rules that govern data communications. It represents an agreement between the communicating devices. Without a protocol, two devices may be connected but not communicating.

Data Representation

Information presents in different forms such as text, numbers, images, audio, and video.

Text: In data communications, text is represented as a bit pattern, a sequence of bits (0s or 1s). Different sets of bit patterns have been designed to represent text symbols. Each set is called a code, and the process of representing symbols is called coding. E.g. ASCII

Numbers: Numbers are also represented by bit patterns. However, a code such as ASCII is not used to represent numbers; the number is directly converted to a binary number to simplify mathematical operations.

Images: Images are also represented by bit patterns and is composed of a matrix of pixels (picture elements), where each pixel is a small dot. The size of the pixel depends on the *resolution.* After an image is divided into pixels, each pixel is assigned a bit pattern. For a black and white image (e.g., a chessboard), a I-bit pattern is enough to represent a pixel. For color images RGB (*red*, green, and blue) and YCM (yellow, cyan, and magenta) are used.

Audio: Audio refers to the recording or broadcasting of sound or music. Audio is by nature different from text, numbers, or images. It is continuous, not discrete. Even when we use a microphone to change voice or music to an electric signal, we create a continuous signal.

Video: Video refers to the recording or broadcasting of a picture or movie. Video can either be produced as a continuous entity (e.g., by a TV camera), or it can be a combination of images, each a discrete entity, arranged to convey the idea of motion.

Data Flow

Communication between two devices can be simplex, half-duplex, or full-duplex.

Simplex

In simplex mode, the communication is unidirectional, as on a one-way street. Only one of the two devices on a link can transmit; the other can only receive (see Figure 1.2a). Keyboards and traditional monitors are examples of simplex devices.

Half-Duplex

In half-duplex mode, each station can both transmit and receive, but not at the same time. When one device is sending, the other can only receive, and vice versa (see Figure 1.2b). The half-duplex mode is like a one-lane road with traffic allowed in both directions. Walkie-talkies and CB (citizens band) radios are both half-duplex systems.

Full-Duplex

In full-duplex (also called duplex), both stations can transmit and receive simultaneously. The full-duplex mode is like a two-way street with traffic flowing in both directions at the same time. One common example is the telephone network.

NETWORKS

A network is a set of devices (often referred to as *nodes)* connected by communication Links. A node can be a computer, printer, or any other device capable of sending and/or receiving data generated by other nodes on the network.

Network Criteria

A network must be able to meet a certain number of criteria. The most important of these are performance, reliability, and security.

Performance-Performance can be measured in many ways, including transit time and response time. Transit time is the amount of time required for a message to travel from one device to another. Response time is the elapsed time between an inquiry and a response. Performance is often evaluated by two networking metrics: throughput and delay.

Reliability-In addition to accuracy of delivery, network reliability is measured by the frequency of failure, the time it takes a link to recover from a failure, and the network's robustness in a catastrophe.

Security- Network security issues include protecting data from unauthorized access, protecting data from damage and development, and implementing policies and procedures for recovery from breaches and data losses.

NETWORK ATTRIBUTES

Before discussing networks, we need to define some network attributes.

Type of Connection- A network is two or more devices connected through links. For communication to occur, two devices must be connected in some way to the same link at the same time. There are two possible types of connections: point-to-point and multipoint.

Point-to-Point- A point-to-point connection provides a dedicated link between two devices. The entire capacity of the link is reserved for transmission between those two devices. Most point-to-point connections use an actual length of wire or cable to connect the two ends, but other options, such as microwave or satellite links.

Multipoint- A multipoint (also called multi drop) connection is one in which more than two specific devices share a single link. In a multipoint environment, the capacity of the channel is shared, either spatially or temporally.

Link- A link is a communications pathway that transfers data from one device to another. For visualization purposes, it is simplest to imagine any link as a line drawn between two points.

Categories of Networks

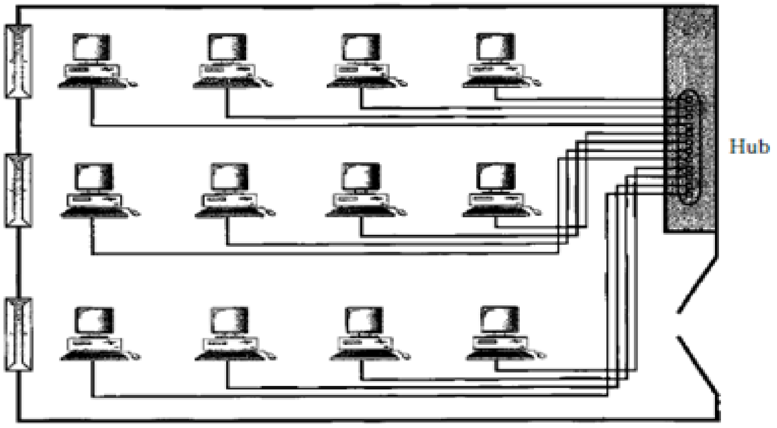
Today when we speak of networks, we are generally referring to the following primary categories: LAN, WAN, MAN

Local Area Network

Local area networks, generally called LANs, are privately-owned networks within a single building or campus of up to a few kilometers in size. LANs are designed to allow resources to be shared between personal computers or workstations. The resources to be shared can include hardware (e.g., a printer), software (e.g., an application program), or data. LAN can exchange information or it can extend throughout a company and include audio and video peripherals. LANs are distinguished from other kinds of networks by three characteristics: **(1) their size, (2) their transmission technology, and (3) their topology.** Examples of a LAN are in many business environment links, a workgroup of task-related computers, Engineering workstations or accounting PCs.

LANs may use a transmission technology consisting of a cable to which all the machines are attached, like the telephone company party lines once used in rural areas. LANs are restricted in size, which means that the worst-case transmission time is bounded and known in advance. It also simplifies network management. Traditional LANs run at speeds of 10 Mbps to 100 Mbps, have low delay (Micro or Nano seconds), and make very few errors. Newer LANs operate at up to 10 Gbps.

Figure 1.2 An Isolated LAN connecting 12 Computers to a hub in a closet



One of the computers may be given a large capacity disk drive and may become a server to clients. Software can be stored on this central server and used as needed by the whole group. In this example, the size of the LAN may be determined by licensing restrictions on the number of users per copy of software, or by restrictions on the number of users licensed to access the operating system.

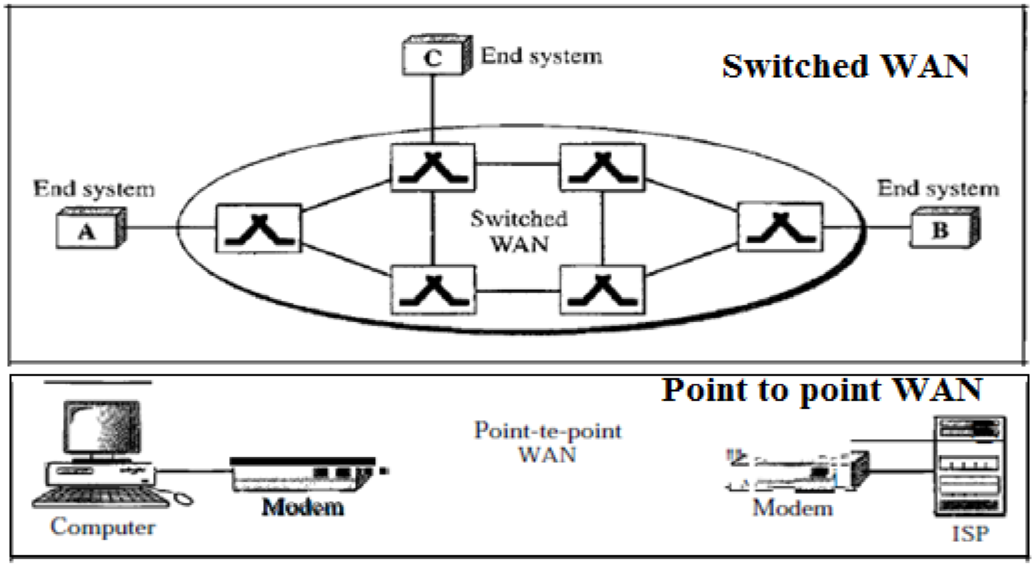
In addition to size, LANs are distinguished from other types of networks by their transmission media and topology. In general, a given LAN will use only one type of transmission medium. The most common LAN topologies are bus, ring, and star. Early LANs had data rates in the 4 to 16 megabits per second (Mbps) range. Today, however, speeds are normally 100 or 1000 Mbps. Wireless LANs are the newest evolution in LAN technology.

Wide Area Network

A wide area network (WAN) provides long-distance transmission of data, image, audio, and video information over large geographic areas that may comprise a country, a continent, or even the whole world. It contains a collection of machines intended for running user (i.e., application) programs. We will follow traditional usage and call these machines hosts. The hosts are connected by a communication subnet, or just subnet for short. The hosts are owned by the customers (e.g., people's personal computers), whereas the communication subnet is typically owned and operated by a telephone company or Internet service provider.

A WAN can be as complex as the backbones that connect the Internet or as simple as a dial-up line that connects a home computer to the Internet. We normally refer to the first as a switched WAN and to the second as a point-to-point WAN (Figure 1.3). The switched WAN connects the end systems, which usually comprise a router (internetworking connecting device) that connects to another LAN or WAN. The point-to-point WAN is normally a line leased from a telephone or cable TV provider that connects a home computer or a small LAN to an Internet service provider (lSP). This type of WAN is often used to provide Internet access.

Figure 1.3 WANs: a switched WAN and a point to point WAN

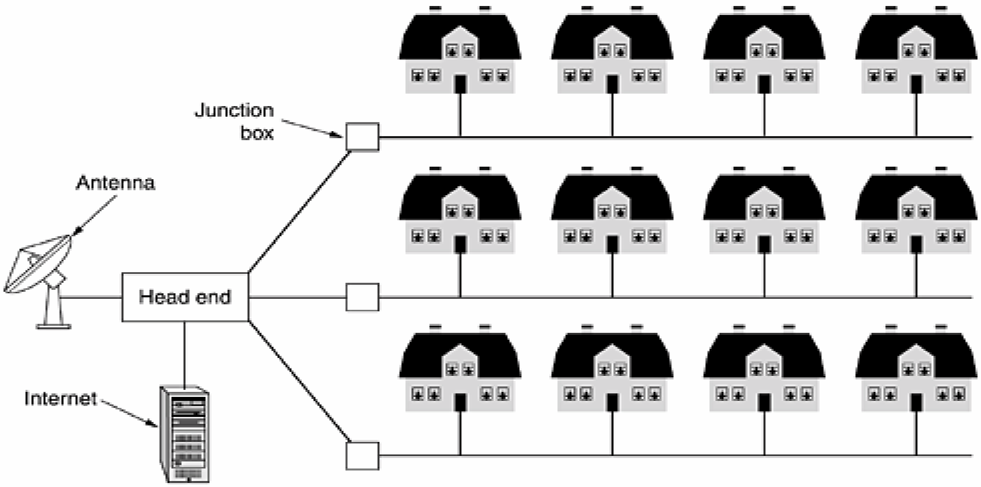


An early example of a switched WAN is X.25, but X.25 is being gradually replaced by a high-speed, more efficient network called Frame Relay. A good example of a switched WAN is the asynchronous transfer mode (ATM) network. Another example is the wireless WAN.

Metropolitan Area Networks

A metropolitan area network (MAN) is a network with a size between a LAN and a WAN. It normally covers the area inside a town or a city. It is designed for customers who need a high-speed connectivity, normally to the Internet, and have endpoints spread over a city or part of city.

Figure 1.4 Metro politan Area Network (MAN)



A good example of a MAN is the part of the telephone company network that can provide a high-speed DSL line to the customer. Another example is the cable TV network, but today can also be used for high-speed data connection to the Internet.

Personal Area Network (802.15)

A **personal area network** (**PAN**) is a computer network used for data transmission among devices such as computers, telephones and personal digital assistants. PANs can be used for communication among the personal devices themselves (intrapersonal communication), or for connecting to a higher level network and the Internet (an uplink).

The data cable is an example of the above PAN. This is also a Personal Area Network because that connection is for the user's personal use. PAN is used for personal use only.

Recently, low-power wireless networking standards like 802.15.1 (Bluetooth) have driven consumer interest in personal area networks (PANs). These networks are designed for inexpensively connecting low-power devices located within 1 m to 100 m of each other. The emerging PAN standards: Bluetooth (802.15.1), Zig Bee (802.15.4) and Ultra-Wide Band (UWB).

Intranet

A private network based on Internet protocols such as TCP/IP but designed for information management within a company or organization. One of the key advantages of an intranet is the broad availability and use of software applications unique to the needs of a corporation. It is also a computer network and includes some of the same technologies as the Internet. Intranet uses include providing access to software applications; document distribution; software Distribution; access to databases; and training.

An intranet is so named because it looks like a World Wide Web site and is based on the same technologies, yet is strictly internal and confidential to the organization and is not connected to the Internet proper. Some intranets also offer access to the Internet, but such connections are directed through a firewall that protects the internal network from the external Web

Extranet

An extension of some combination of corporate, public, and private intranet using World Wide Web technology to facilitate communication with the corporation's suppliers, customers, and associates. An extranet allows customers, suppliers, and business partners to gain limited access to a company's intranet in order to enhance the speed and efficiency of their business relationship.

INTERCONNECTION OF NETWORKS: INTERNETWORK

Today, it is very rare to see a LAN, a MAN, or a LAN in isolation; they are connected to one another. An internet (note the lowercase letter i) is two or more networks that can communicate with each other. The most notable internet is called the Internet (uppercase letter I), a collaboration of more than hundreds of thousands of interconnected networks.

Internet: The Internet has revolutionized many aspects of our daily lives. The Internet is a communication system that has brought a wealth of information to our fingertips and organized it for our use. The Internet is a structured, organized system. Private individuals as well as various organizations such as government agencies, schools, research facilities, corporations, and libraries in more than 100 countries use the Internet. Millions of people are users. Yet this extraordinary communication system only came into being in 1969.

A Brief History of Internet: In the mid-1960s, mainframe computers in research organizations were standalone 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. In 1967, at an Association for Computing Machinery (ACM) meeting, ARPA presented its ideas for 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 one another. Each IMP had to be able to communicate with other IMPs as well as with its own attached host.

By 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

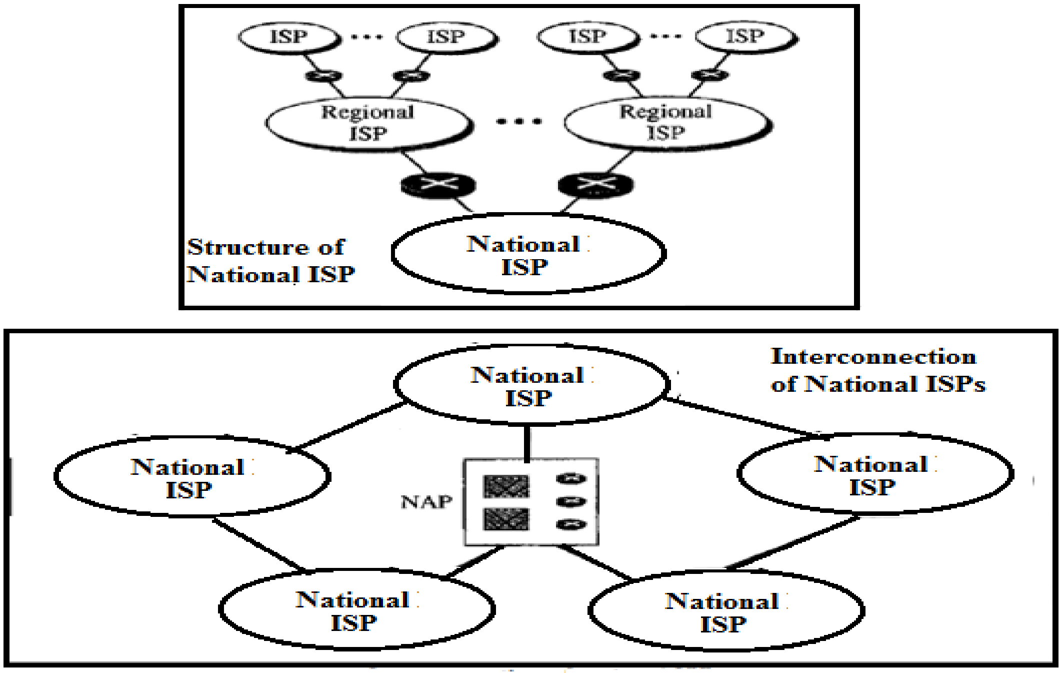
Institute (SRI), and the University of Utah, were connected via the IMPs to form a network. Software called the *Network Control Protocol* (NCP) provided communication between the hosts. In 1972, Vint Cerf and Bob Kahn, both of whom were part of the core ARPANET group, collaborated on what they called the *Internetting Project.* Cerf and Kahn's landmark 1973 paper outlined the protocols to achieve end-to-end delivery of packets. This paper on Transmission Control Protocol (TCP) included concepts such as encapsulation, the datagram, and the functions of a gateway. Shortly thereafter, TCP split into two protocols: Transmission Control Protocol (TCP) and Internetworking Protocol (lP). IP would handle datagram routing while TCP would be responsible for higher-level functions such as segmentation, reassembly, and error detection. The internetworking protocol became known as TCP/IP.

The Internet Today: The Internet today is not a simple hierarchical structure. It is made up of many wide and local-area networks joined by connecting devices and switching stations. Today most end users who want Internet connection use the services of Internet service providers (lSPs). There are international, national, regional service providers and local service providers. The Internet today is run by private companies, not the government.

International Internet Service Providers: At the top of the hierarchy are the international service providers that connect nations together.

National Internet Service Providers: The national Internet service providers are backbone networks created and maintained by specialized companies. E.g. Sprint Link, PSINet, UUNet, AGIS, and internet Mel (North America).

Figure 1.5 Hierarchical Organization of the Internet



To provide connectivity between the end users, these backbone networks are connected by complex switching stations (normally run by a third party) called network access points

(NAPs). Some national ISP networks are also connected to one another by private switching stations called *peering points.* These normally operate at a high data rate (up to 600 Mbps).

Regional Internet Service Providers: Regional internet service providers or regional ISPs are smaller ISPs that are connected to one or more national ISPs. They are at the third level of the hierarchy with a smaller data rate.

Local Internet Service Providers: Local Internet service providers provide direct service to the end users. The local ISPs can be connected to regional ISPs or directly to national ISPs. Most end users are connected to the local ISPs. Local ISP can be a company with Internet services, a corporation with a network to its own employees, or a nonprofit organization, such as a college or a university that runs its own network. Each of these local ISPs can be connected to a regional or national service provider.

PROTOCOLS AND STANDARDS

Protocols

In computer networks, communication occurs between entities in different systems. An entity is anything capable of sending or receiving information. For communication to occur, the entities must agree on a protocol. 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. The key elements of a protocol are syntax, semantics, and timing.

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.

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?

Timing: The term *timing* refers to two characteristics: when data should be sent and how fast they can be sent.

Standards

Standards are essential in creating and maintaining an open and competitive market for equipment manufacturers and in guaranteeing national and international interoperability of data and telecommunications technology and processes. Standards provide guidelines to manufacturers, vendors, government agencies, and other service providers to ensure the kind of interconnectivity necessary in today's marketplace and in international communications.

Standards Organizations

Standards are developed through the cooperation of standards creation committees, forums, and government regulatory agencies.

International Organization for Standardization (ISO)-1990: The ISO is a multinational body whose membership is drawn mainly from the standards creation committees of various governments throughout the world. The ISO is active in developing cooperation in the realms of scientific, technological, and economic activity.

International Telecommunication Union-Telecommunication Standards Sector (ITU-T) 1993: The United Nations responded by forming, as part of its International Telecommunication Union (ITU), a committee, the Consultative Committee for International Telegraphy and Telephony (CCITT). This committee was devoted to the research and establishment of standards for telecommunications in general and for phone and data systems in particular. On March 1, 1993, the name of this committee was changed (ITU-T).

American National Standards Institute (ANSI): Despite its name, the American National Standards Institute is a completely private, nonprofit corporation not affiliated with the U.S. federal government. However, all ANSI activities are undertaken with the welfare of the United States and its citizens occupying primary importance.

Institute of Electrical and Electronics Engineers (IEEE): The Institute of Electrical and Electronics Engineers is the largest professional engineering society in the world. International in scope, it aims to advance theory, creativity, and product quality in the fields of electrical engineering, electronics, and radio as well as in all related branches of engineering. As one of its goals, the IEEE oversees the development and adoption of international standards for computing and communications.

Electronic Industries Association (EIA): Aligned with ANSI, the Electronic Industries Association is a nonprofit organization devoted to the promotion of electronics manufacturing concerns. Its activities include public awareness education and lobbying efforts in addition to standards development

Forums and Regulatory Agencies

Forums are developed to facilitate the standardization process by special interest groups from corporations and it work with universities and users to test, evaluate, and standardize new technologies. The purpose of government regulatory agencies such as the **Federal Communications Commission** (FCC in the United States) is to protect the public interest by regulating radio, television, and wire/cable communications. The FCC has authority over interstate and international commerce as it relates to communications.

Internet Standards

An **Internet standard** is a thoroughly tested specification and there is a strict procedure by which a specification attains Internet standard status. A specification begins as an Internet draft. An **Internet draft** is a working document (a work in progress) with no official status and a 6-month lifetime. Upon recommendation from the Internet authorities, a draft may be published as a **Request for Comment** (RFC).

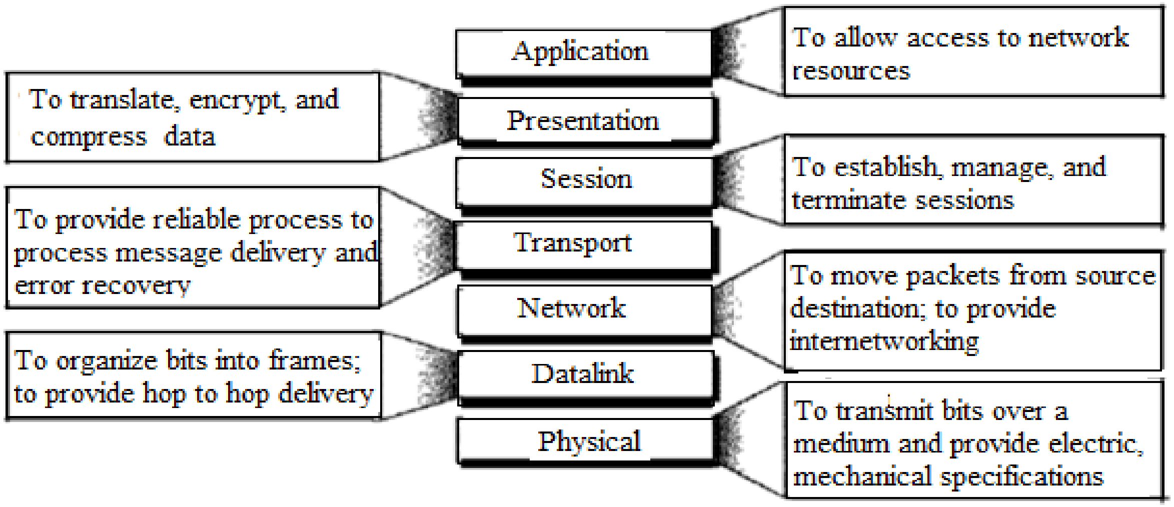
Network Models

A network is a set of devices (often referred to as *nodes)* connected by communication Links. A node can be a computer, printer, or any other device capable of sending and/or receiving data generated by other nodes on the network. A network is a combination of hardware and software that sends data from one location to another. The hardware consists of the physical equipment that carries signals from one point of the network to another. The software consists of instruction sets that make possible the services that we expect from a network.

THE OSI MODEL

An ISO standard that covers all aspects of network communications is the Open Systems Interconnection (OSI) model. It was first introduced in the late 1970s. 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.

Figure 1.6 Seven Layers of OSI Model


Application
To allow access to network resources
To translate, encrypt, and compress data
Presentation
Session
To establish, manage, and terminate sessions
To provide reliable process to process message delivery and error recovery
Transport
Network
To move packets from source destination; to provide internetworking
To organize bits into frames; to provide hop to hop delivery
Datalink
Physical
To transmit bits over a medium and provide electric, mechanical specifications

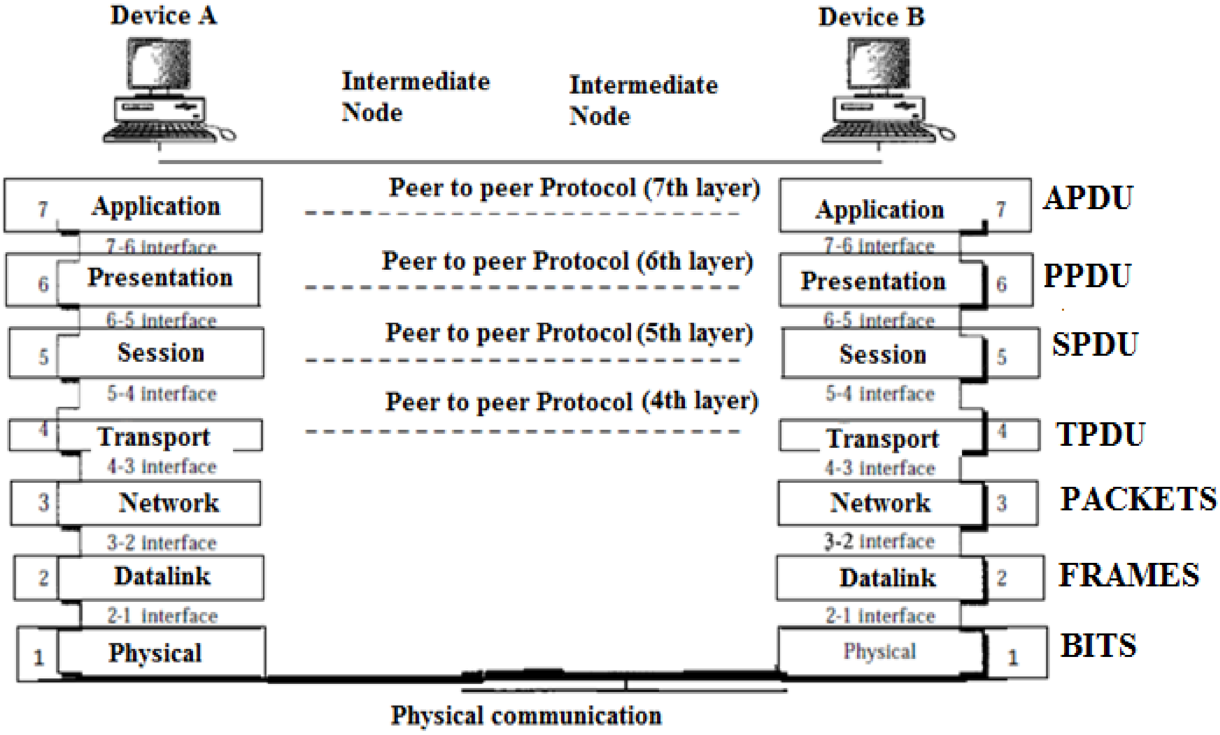
Layered Architecture

A network model can be explained with the layered tasks. According to the layered task, the functions of network from sender to the receiver are split into many layers. The layered model that dominated data communications and networking literature before 1990 was the Open Systems Interconnection (OSI) model. Everyone believed that the OSI model would become the ultimate standard for data communications, but this did not happen. The TCP/IP protocol suite became the dominant commercial architecture because it was used and tested extensively in the Internet; the OSI model was never fully implemented.

LAYERS IN THE OSI MODEL

Figure 1.6 shows the layers involved when a message is sent from device A to device B. As the message travels from A to B, it may pass through many intermediate nodes. These intermediate nodes usually involve only the first three layers of the OSI model.

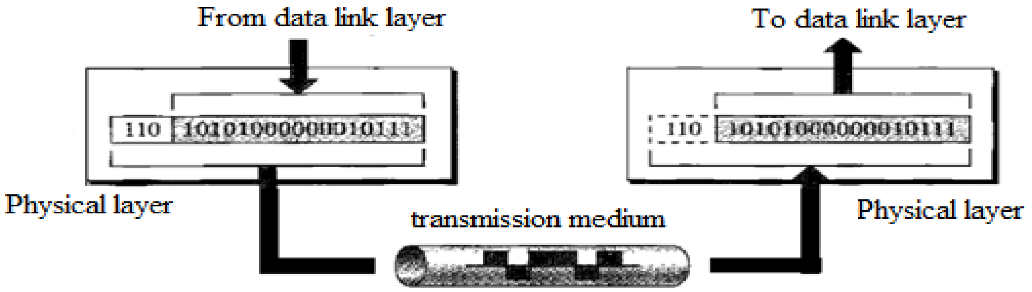
Figure 1.7 The interaction between layers in the OSI model



Physical Layer

The physical layer coordinates the functions required to carry a bit stream over a physical medium. It deals with the mechanical and electrical specifications of the interface and transmission medium. It also defines the procedures and functions that physical devices and interfaces have to perform for transmission to occur.

Figure 1.8 Physical Layer



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

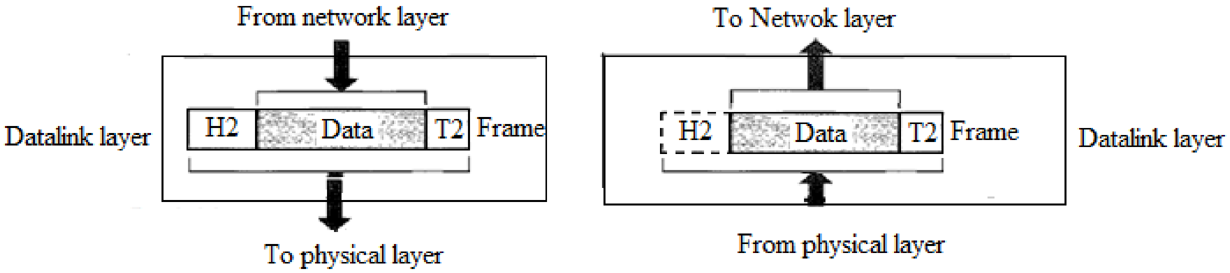
The physical layer is also concerned with the following:

1. **Physical characteristics of interfaces and medium**: It also defines the type of transmission medium.
2. **Representation of bits:** The physical layer defines the type of encoding (how 0s and 1s are changed to signals).
3. **Data rate**: The physical layer defines the duration of a bit, which is how long it lasts.
4. **Synchronization of bits:** The sender and the receiver clocks must be synchronized.
5. **Line configuration**: The physical layer is concerned with the connection of devices to the media. It may be dedicated link (point-to-point) or Shared link (Multi point).
6. **Physical topology:** The physical topology defines how devices are connected to make a network. Devices can be connected by using a mesh, star, a ring, a bus or a hybrid topology
7. **Transmission mode:** The physical layer also defines the direction of transmission between two devices: It may be simplex, half-duplex, or full-duplex.

Data Link Layer

The data link layer transforms the physical layer, a raw transmission facility, to a reliable link. The data link layer is responsible for moving frames from one hop (node) to the next. Figure 1.9 shows the relationship of the data link layer to the network and physical layers.

Figure 1.9 Datalink Layer



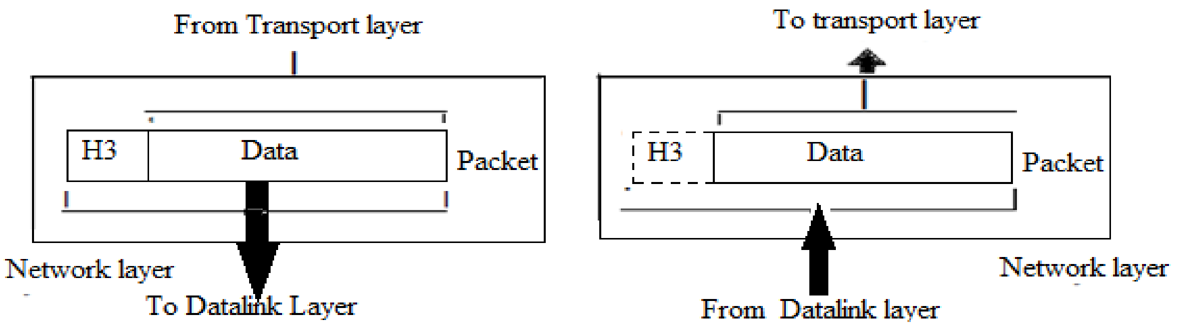
Other responsibilities of the data link layer include the following:

1. **Framing:** The data link layer divides the stream of bits received from the network layer into manageable data units called frames.
2. **Physical addressing:** 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, then receiver address connects the next network.
3. **Flow control:** If the receiver rate is less than the sender rate, then data link layer imposes a flow control mechanism to avoid overwhelming the receiver.
4. **Error control:** The data link layer adds reliability to the physical layer by detecting and retransmitting damaged or lost frames. It also recognizes duplicate frames.
5. **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.

Network Layer

The network layer is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links). If the two systems are attached to different networks (links) with connecting devices between the networks (links), there is often a need for the network layer to accomplish source-to-destination delivery. Figure 1.10 shows the relationship of the network layer to the data link and transport layers.

Figure 1.10 Network Layer



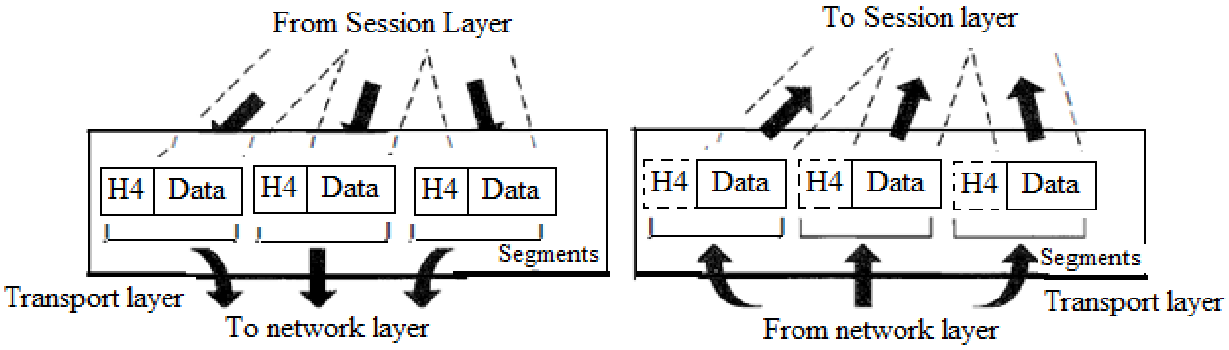
The network layer is responsible for the delivery of individual packets from the source host to the destination host.

Other responsibilities of the network layer include the following:

1. **Logical addressing:** The network layer adds a header to the packet which includes the logical addresses of the sender and receiver. This address used to distinguish the source and destination systems.
2. **Routing:** When independent networks or links are connected to create *intemetworks* (network of networks) or a large network, the connecting devices (called *routers* or *switches)* route or switch the packets to their final destination.

Transport Layer

Figure 1.11 Transport Layer



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

The transport layer is responsible for process-to-process delivery of the entire message. A process is an application program running on a host. The transport layer ensures that the whole message arrives intact and in order, overseeing both error control and flow control at the source-to-destination level.

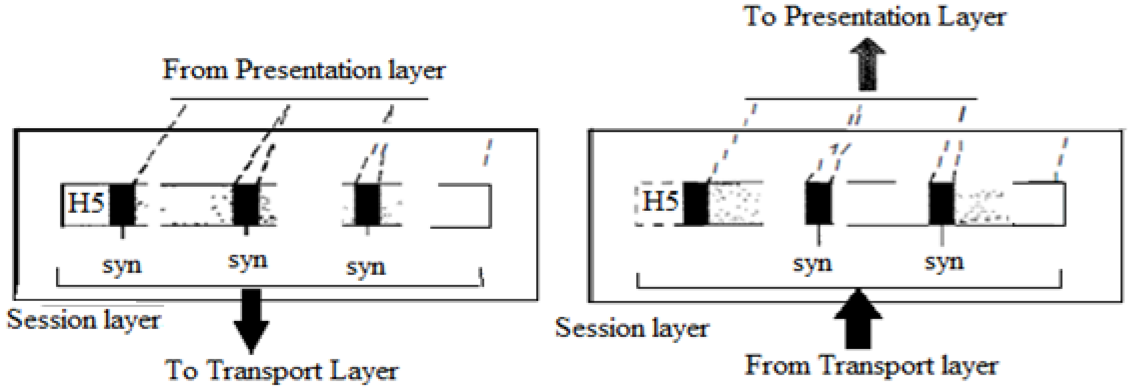
Other responsibilities of the transport layer include the following:

1. **Service-point addressing:** To achieve source-to-destination delivery, the transport layer uses a *service-point address* (or port address) in the transport layer header. The network layer gets each packet to the correct computer; the transport layer gets the entire message to the correct process on that computer.
2. **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.
3. **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. After all the data are transferred, the connection is terminated.
4. **Flow control:** The transport layer is responsible for end-to-end flow control rather than across a single link.
5. **Error control:** The transport layer is responsible for end-to-end error control, 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.

Session Layer

The services provided by the first three layers (physical, data link, and network) are not sufficient for some processes. The session layer is the network *dialog controller.*

Figure 1.12 Session Layer



Session layer establishes, maintains, and synchronizes the interaction among communicating systems. The session layer is responsible for dialog control and synchronization.

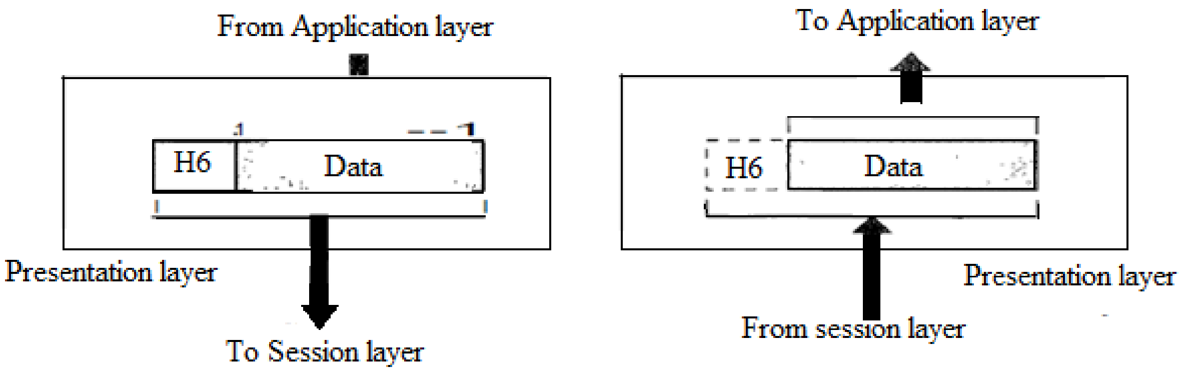
Specific responsibilities of the session layer include the following:

1. **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.
2. **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.

Presentation Layer

The presentation layer is concerned with the syntax and semantics of the information exchanged between two systems. Figure 1.13 shows the relationship between the presentation layer and the application and session layers.

Figure 1.13 Presentation layer



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

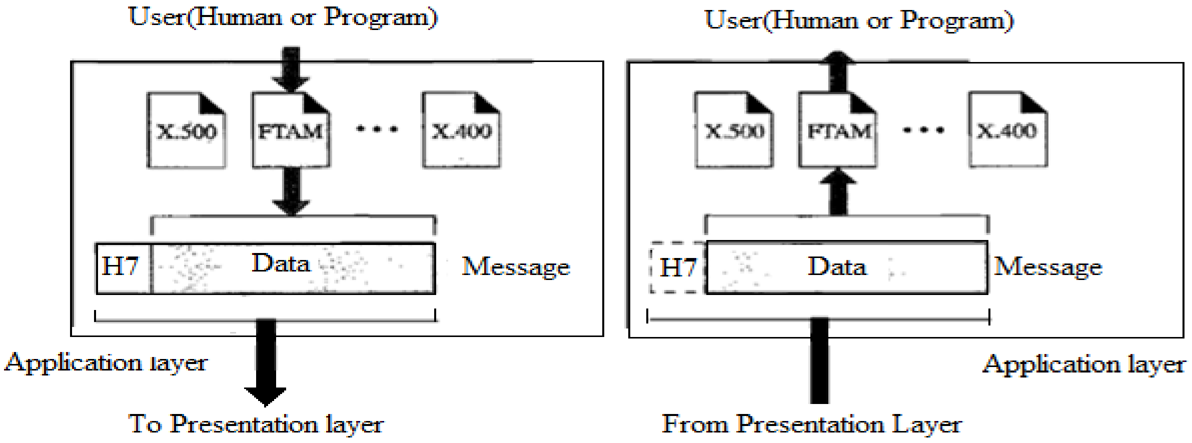
Specific responsibilities of the presentation layer include the following:

1. **Translation:** The presentation layer is responsible for interoperability between these different encoding methods. 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.
2. **Encryption:** To carry sensitive information, a system must be able to ensure privacy. Encryption means that the sender transforms the message into coded form and sends over the network. Decryption reverses the process to transform message back to its original.
3. **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.

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, etc. The figure shows only three: *XAOO* (message-handling services), X.500 (directory services), and file transfer, access, and management (FTAM).

Figure 1.14 Application layer



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

Specific services provided by the application layer include the following:

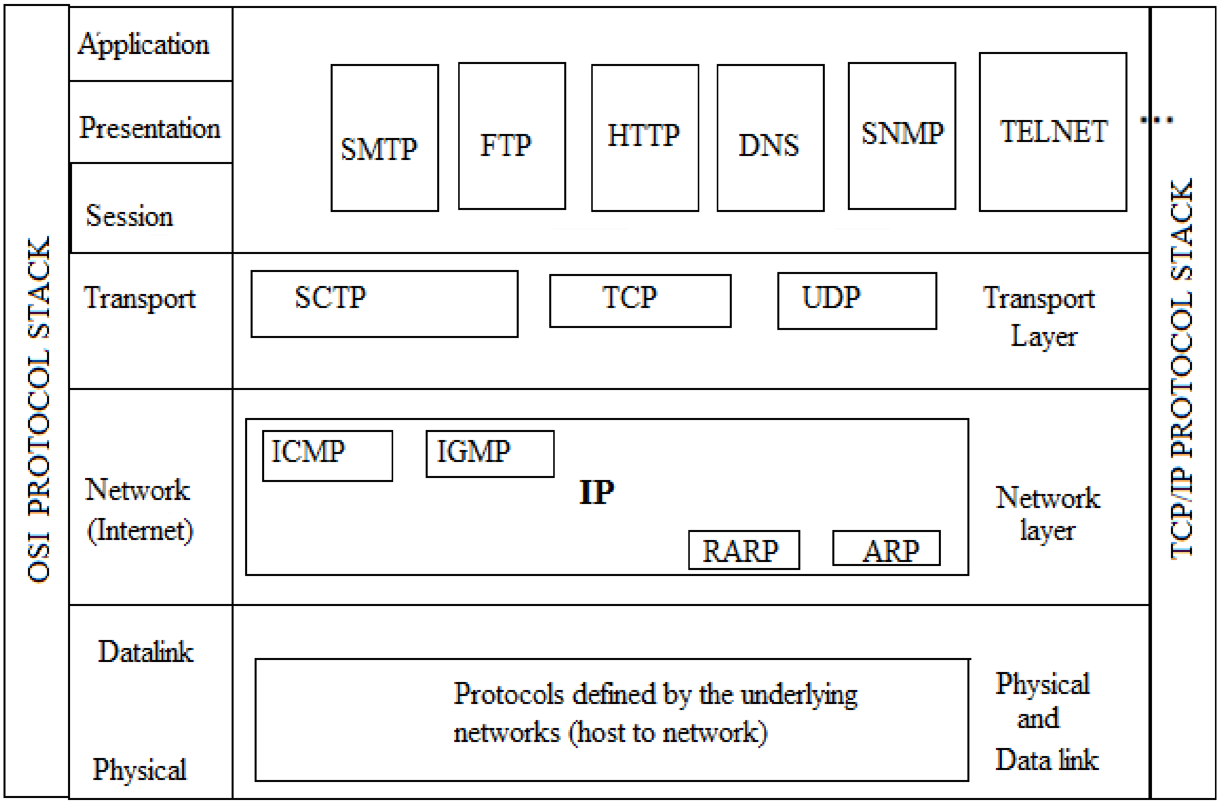
1. **Network virtual terminal**: A network virtual terminal is a software version of a physical terminal, and it allows a user to log on to a remote host. To do so, the application creates a software emulation of a terminal at the remote host. The user's computer talks to the software terminal which, in turn, talks to the host, and vice versa. The remote host believes it is communicating with one of its own terminals and allows the user to log on.
2. **File transfer, access, and management:** This application allows a user to access files in a remote host (to make changes or read data), to retrieve files from a remote computer for use in the local computer, and to manage or control files in a remote computer locally.
3. **Mail services:** This application provides the basis for e-mail forwarding and storage.
4. **Directory services:** This application provides distributed database sources and access for global information about various objects and services.

TCP/IP PROTOCOL SUITE

The TCP/IP 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. The original TCP/IP protocol suite was defined as having four layers: host-to-network, internet, transport, and application.

When TCP/IP is compared to OSI, we can say that the host-to-network layer is equivalent to the combination of the physical and data link layers. TCP/IP is made of five layers: physical, data link, network, transport, and application. The first four layers provide physical standards, network interfaces, internetworking, and transport functions that correspond to the first four layers of the OSI model. The internet layer is equivalent to the network layer, and the application layer is roughly doing the job of the session, presentation, and application layers with the transport layer in TCP/IP taking care of part of the duties of the session layer (see Figure 1.15).

Figure 1.15 TCP/IP and OSI Model



TCP/IP is a hierarchical protocol made up of interactive modules, each of which provides a specific functionality; however, the modules are not necessarily interdependent. Whereas the OSI model specifies which functions belong to each of its layers, the layers of the *TCP/IP* protocol suite contain relatively independent protocols that can be mixed and matched depending on the needs of the system. The term *hierarchical* means that each upper-level protocol is supported by one or more lower-level protocols.

Physical and Data Link Layers

At the physical and data link layers, TCP/IP does not define any specific protocol. It supports all the standard and proprietary protocols. A network in a TCP/IP internetwork can be a local-area network or a wide-area network.

Network Layer

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. It is an unreliable and connectionless protocol-a best-effort delivery service. The term *best effort* means that IP provides no error checking or tracking. IP assumes the unreliability of the underlying layers and does its best to get a transmission through to its destination, but with no guarantees. IP transports data in packets called *data grams*, each of which is transported separately. Data grams can travel along different routes and can arrive out of sequence or be duplicated.

IP does not keep track of the routes and has no facility for reordering data grams once they arrive at their destination. The limited functionality of IP should not be considered a weakness, however. IP provides bare-bones transmission functions that free the user to add only those facilities necessary for a given application and thereby allows for maximum efficiency.

Address Resolution Protocol (ARP): The Address Resolution Protocol is used to associate a logical address with a physical address. ARP is used to find the physical address of the node when its Internet address is known. It is also used to translate IP addresses to Ethernet addresses. The translation is done only for outgoing IP packets, because this is when the IP header and the Ethernet header are created.

Reverse Address Resolution Protocol (RARP): The Reverse Address Resolution Protocol 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.

Internet Control Message Protocol (ICMP): The Internet Control Message Protocol 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.

Internet Group Message Protocol (IGMP): The Internet Group Message Protocol is used to facilitate the simultaneous transmission of a message to a group of recipients.

Transport Layer

Traditionally the transport layer was represented in *TCP/IP* by two protocols: TCP and UDP. IP is a host-to-host protocol, meaning that it can deliver a packet from one physical device to another. UDP and TCP are transport level protocols responsible for delivery of a message from a process (running program) to another process. A new transport layer protocol, SCTP, has been devised to meet the needs of some newer applications.

User Datagram Protocol (UDP): User Datagram Protocol is an unreliable, connectionless protocol for applications that do not want TCP's sequencing or flow control and wish to provide their own. It is also widely used for one-shot, client-server-type request-reply queries and applications in which prompt delivery is more important than accurate delivery, such as transmitting speech or video. It is a process-to-process protocol that adds only port addresses, checksum error control, and length information to the data from the upper layer.

Transmission Control Protocol (TCP): The Transmission Control Protocol (TCP) provides full transport-layer services to applications. TCP is a reliable stream transport protocol. The term *stream,* in this context, means connection-oriented: A connection must be established between both ends of a transmission before either can transmit data.

At the sending end of each transmission, TCP divides a stream of data into smaller units called *segments.* Each segment includes a sequence number for reordering after receipt, together with an acknowledgment number for the segments received.

Segments are carried across the internet inside of IP data grams. At the receiving end, TCP collects each datagram as it comes in and reorders the transmission based on sequence numbers.

Stream Control Transmission Protocol: The Stream Control Transmission Protocol (SCTP) provides support for newer applications such as voice over the Internet. It is a transport layer protocol that combines the best features of UDP and TCP.

Application Layer

The *application layer* in TCP/IP is equivalent to the combined session, presentation, and application layers in the OSI model many protocols are defined at this layer. The TCP/IP model does not have session or presentation layers. No need for them was perceived, so they were not included. Experience with the OSI model has proven this view correct: they are of little use to most applications.

On top of the transport layer is the application layer. It contains all the higher-level protocols. The early ones included virtual terminal (TELNET), file transfer (FTP), and electronic mail (SMTP). The virtual terminal protocol allows a user on one machine to log onto a distant machine and work there.

The file transfer protocol provides a way to move data efficiently from one machine to another. Electronic mail was originally just a kind of file transfer, but later a specialized protocol (SMTP) was developed for it. Many other protocols have been added to these over the years: the Domain Name System (DNS) for mapping host names onto their network addresses, NNTP, the protocol for moving USENET news articles around, and HTTP, the protocol for fetching pages on the World Wide Web, and many others.

Broadband ISDN

The need for a Broadband ISDN service sprung from the growing needs of the customers. The planned Broadband ISDN services can broadly be categorized as follows:

Interactive services

These are services allowing information flow between two end users of the network, or between the user and the service provider. Such services can be subdivided:

Conversational services: These are basically end-to-end, real-time communications, between users or between a user and a service provider, e.g. telephone-like services. Indeed, B-ISDN will support N-ISDN type services. (Note also that the user-to-user signaling, user-to-network signaling, and inter-change signaling are also provided but outside our scope.) Also the additional bandwidth offered will allow such services as video telephony, video conferencing and high volume, high speed data transfer.

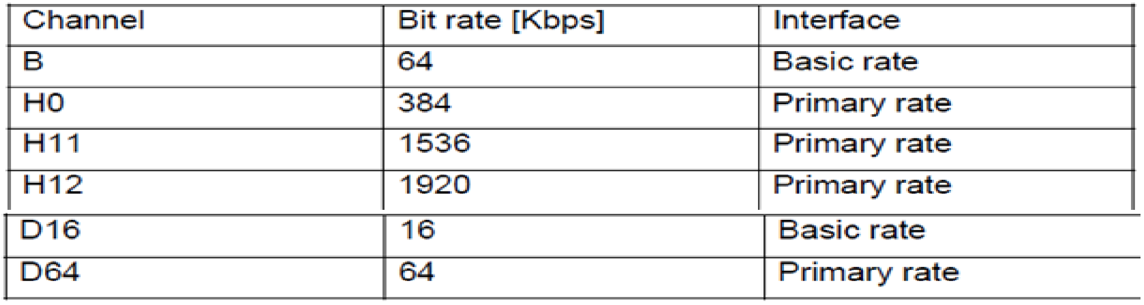
Messaging services: This differs from conversational services in that it is mainly a store-and-forward type of service. Applications could include voice and video mail, as well as multi-media mail and traditional electronic mail.

Retrieval services: This service provides access to (public) information stores, and information is sent to the user on demand only. This includes things like tele-shopping, video tex services, still and moving pictures, tele-software and entertainment.

Distribution services

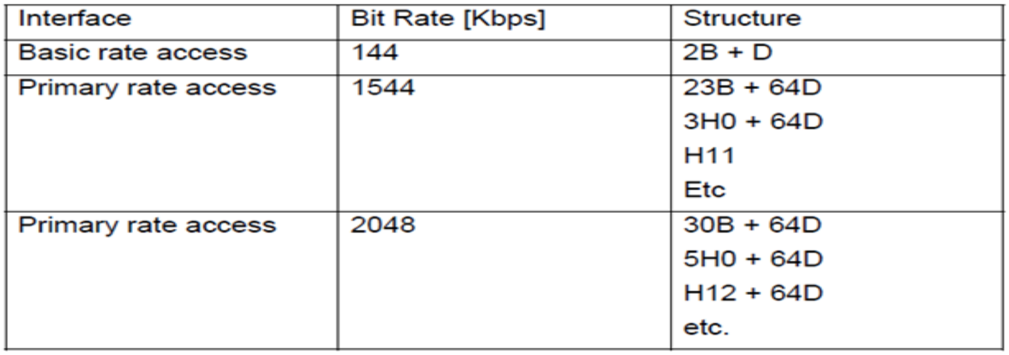
These are mainly broadcast services, are intended for mainly one way interaction from a service provider to a user:

No user control of presentation. This would be for instance, a TV broadcast, where the user can choose simply either to view or not. It is expected that cable TV companies will become interested in Broadband ISDN as a carrier for the high definition TV (HDTV) services that are foreseen for the future. However, many of these services have very high throughput requirements, as shown in Table below. The business is the ratio of the peak bit rate to average bit rate.



Channel	Bit rate [Kbps]	Interface
B	64	Basic rate
H0	384	Primary rate
H11	1536	Primary rate
H12	1920	Primary rate
D16	16	Basic rate
D64	64	Primary rate

It is clear that high network capacity is required if this kind of service is to be offered to many user simultaneously. The N-ISDN can currently offer interfaces which aggregate B-Channels to give additional throughput, as shown in Tables below. However, these are not sufficient for our Broadband service requirements.



Interface	Bit Rate [Kbps]	Structure
Basic rate access	144	2B + D
Primary rate access	1544	23B + 64D 3H0 + 64D H11 Etc
Primary rate access	2048	30B + 64D 5H0 + 64D H12 + 64D etc.

B-ISDN NETWORK ARCHITECTUIRE

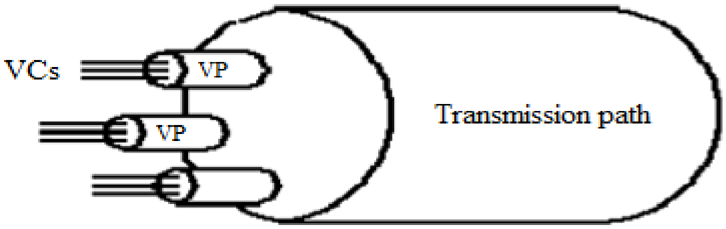
The B-ISDN needs to provide:

1. Broadband services 2.Narrowband services (for backwards compatibility).

3. User-to-network signaling 4.Inter-exchange signaling within the network

5. User-to-user signaling 6.Management facilities for controlling and operating network.

It is intended that the B-ISDN will offer both connection oriented (CO) and connectionless (CL) services, however, the CO mode of operation is receiving the greatest attention at the moment, while CL service definitions mature.



The broadband information transfer is provided by the use of asynchronous transfer mode (ATM), in both cases, using end-to-end logical connections. ATM makes use of small, fixed size (53 octets) cells in which the information is transferred, along the logical connections. Each logical connection is accessed as a **virtual channel (VC)**. Many VCs may be used to a single destination and they may be associated by use of a **virtual path (VP)**. Their relationship between VCs and VPs with respect to the **transmission path** is shown in Figure.

The transmission path is the logical connection between the two end-points, and consists in reality of many **links** between network exchanges and switches. The VCs are identified at each end of the connection by a **virtual channel identifier (VCI)** and user-to-user data VCs are unidirectional. Similarly, the VP is identified by a **virtual path identifier (VPI)**. VCIs and VPIs are used within the network for switching purposes, with **virtual channel links** and **virtual path links** being defined as the connection between two points where the either the VC or the VP is switched, respectively, i.e. the link is defined to exist between the two points where the VCI or VPI value is removed or translated (switched).

Asynchronous Transfer Mode (ATM)

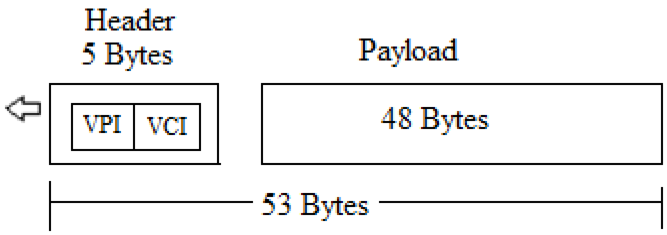
Asynchronous Transfer Mode (ATM) is the cell relay protocol designed by the ATM Forum and adopted by the ITU-T. The combination of ATM and SONET will allow high-speed interconnection of all the world's networks.

ATM is a cell-switched network. The user access devices, called the endpoints, are connected through a user-to-network interface (UNI) to the switches inside the network. The switches are connected through network-to-network interfaces (NNIs). Figure shows an example of an ATM network.

An ATM Cell

The basic data unit in an ATM network is called a cell. A cell is only 53 bytes long with bytes allocated to the header and 48 bytes payload (user data may be less than 48 bytes).

Figure 1.16 An ATM Cell



The most of cell header is occupied by the VPI and VCI that define the virtual connection through which a cell should travel from an endpoint to a switch or from a switch to another switch. Figure shows the cell structure.

ATM Protocol Reference Model

Unlike the earlier two-dimensional reference models, the ATM model is defined as being three-dimensional, as shown in Figure.

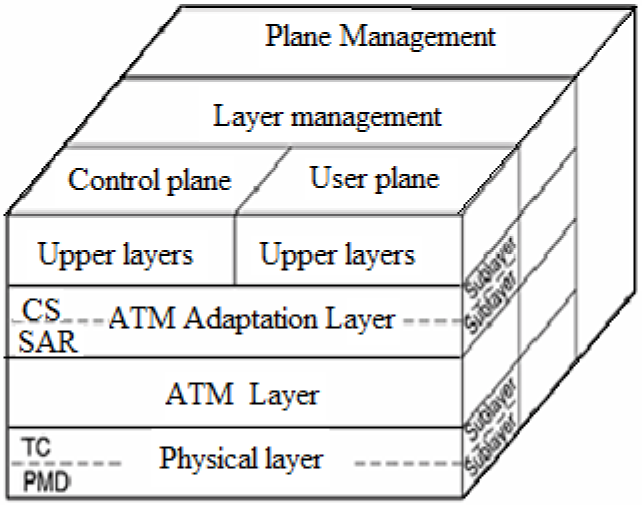


Figure 1.17 ATM Protocol Reference Model

CS: Convergence Sublayer
SAR: Segmentation and Reassembly
TC: Transmission Control Sublayer
PMD: Physical Medium Dependent Sublayer

The user plane deals with data transport, flow control, error correction, and other user functions. In contrast, the control plane is concerned with connection management. The layer and plane management functions relate to resource management and interlayer coordination.

ATM Layers

The ATM standard defines three layers. They are, from top to bottom; the application adaptation layer, the ATM layer, and the physical layer see Figure. The endpoints use all three layers while the switches use only the two bottom layers.

Figure 1.18 ATM Layers

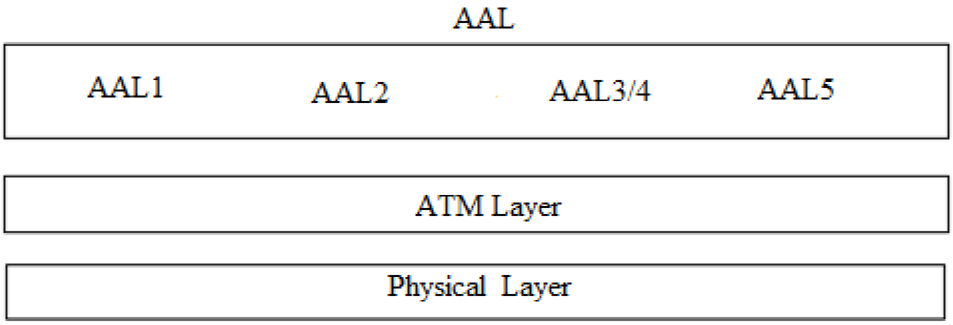
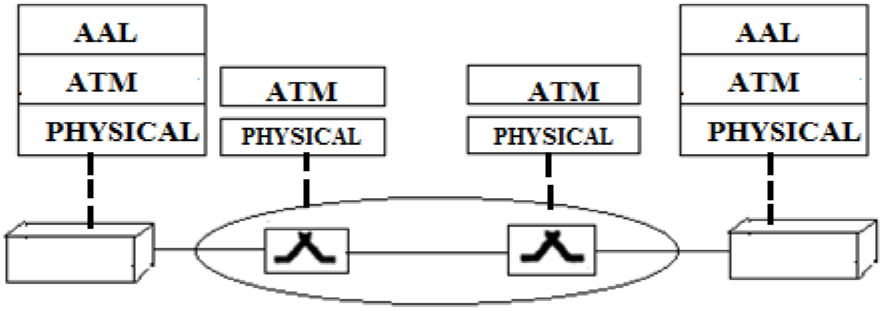


Figure 1.19 ATM Layers in End point Devices



Physical Layer

Like Ethernet and wireless LANs, ATM cells can be carried by any physical layer carrier. It deals with the physical medium: voltages, bit timing, and various other issues. ATM has been designed to be independent of the transmission medium.

The original design of ATM was based on *SONET* as the physical layer carrier. SONET is preferred for two reasons. First, the high data rate of SONET's carrier and second using SONET, the boundaries of cells can be clearly defined.

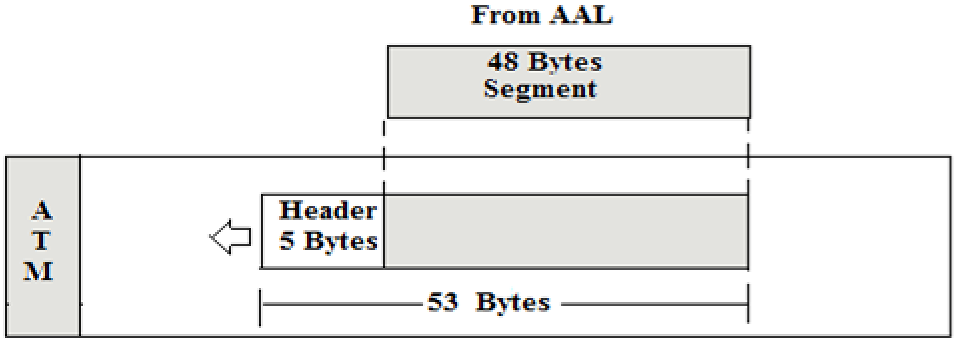
Other Physical Technologies ATM does not limit the physical layer to SONET. Other technologies, even wireless, may be used. However, the problem of cell boundaries must be solved. One solution is for the receiver to guess the end of the cell and apply the CRC to the 5-byte header.

ATM Layer

The ATM layer deals with cells and cell transport. It defines the layout of a cell and tells what the header fields mean. It also deals with establishment and release of virtual circuits. Congestion control is also located here.

The ATM layer provides routing, traffic management, switching, and multiplexing services. It processes outgoing traffic by accepting 48-byte segments from the AAL sub-layers and transforming them into 53-byte cells by the addition of a 5-byte header.

Figure 1.20 ATM Layer

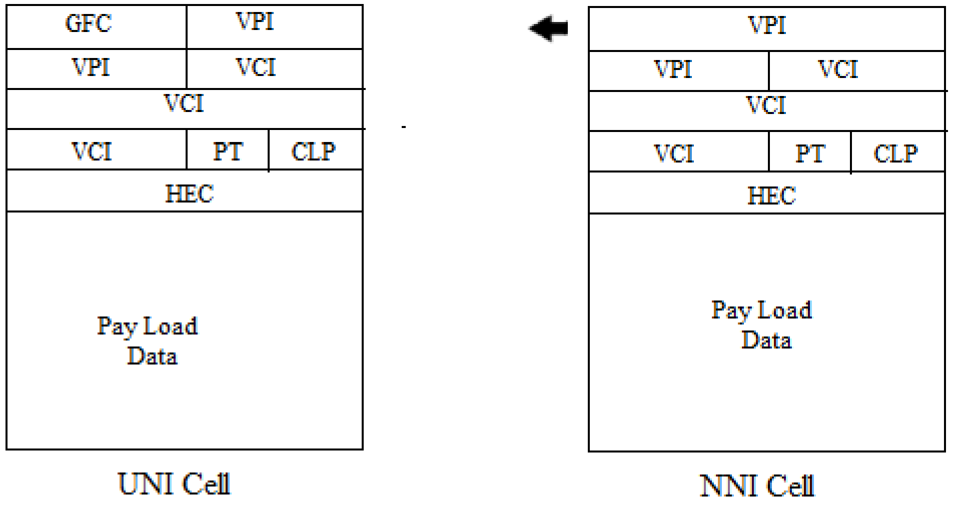


Header Format ATM uses two formats for this header, one for user-to-network interface (UNI) cells and another for network-to-network interface (NNI) cells. Figure below shows these headers in the byte-by-byte format preferred by the ITU-T (each row represents a byte).

Figure 1.21 ATM Headers

GFC: Generic Flow Control
VPI: Virtual Path Identifier
VCI: Virtual connection Identifier

PT: Payload
CLP: Cell Loss Priority
HEC: Header error connection



Generic flow control (GFC). The 4-bit GFC field provides flow control at the UNI level. The ITU-T has determined that this level of flow control is not necessary at the NNI level. In the NNI header, therefore, these bits are added to the VPI. The longer VPI allows more virtual paths to be defined at the NNI level. The format for this additional VPI has not yet been determined.

- **Virtual path identifier (VPI):** The VPI is an 8-bit field in a UNI cell and a 12-bit field in an NNI cell (see above).
- **Virtual circuit identifier (VCI):** The VCI is a 16-bit field in both frames.
- **Payload type (PT):** In the 3-bit PT field, the first bit defines the payload as user data or managerial information. The interpretation of the last 2 bits depends on the first bit.
- **Cell loss priority (CLP):** The 1st-bit CLP field is provided for congestion control. A cell with its CLP bit set to 1 must be retained as long as there are cells with a CLP of 0.
- **Header error correction (HEC):** The HEC is a code computed for the first 4 bytes of the header. It is a CRC with the divisor *x8* + *x2* + *x* + 1 that is used to correct single-bit errors and a large class of multiple-bit errors.

Application Adaptation Layer

The application adaptation layer (AAL) was designed to enable two ATM concepts. First, ATM must accept any type of payload, both data frames and streams of bits. A data frame can come from an upper-layer protocol that creates a clearly defined frame to be sent to a carrier network such as ATM. A good example is the Internet. ATM must also carry multimedia payload. It can accept continuous bit streams and break them into chunks to be encapsulated into a cell at the ATM layer. AAL uses two sub layers to accomplish these tasks.

The AAL defines a sub layer, called a segmentation and reassembly (SAR) sub layer, to do so. Segmentation is at the source; reassembly, at the destination. Before data are segmented by SAR, they must be prepared to guarantee the integrity of the data. This is done by a sub layer called the convergence sub layer (CS).

The CS sub layer divides the bit stream into 47-byte segments and passes them to the SAR sub layer below. Note that the CS sub layer does not add a header. The SAR sub layer adds 1 byte of header and passes the 48-byte segment to the

ATM layer header has two fields:

1. Sequence number (SN). This 4-bit field defines a sequence number to order the bits. The first bit is sometimes used for timing, which leaves 3 bits for sequencing (modulo 8).
2. Sequence number protection (SNP). The second 4-bit field protects the first field. The first 3 bits automatically correct the SN field. The last bit is a parity bit that detects error over all 8 bits.

ATM defines four versions of the AAL: AALl, AAL2, *AAL3/4*, and AAL5. The common versions today are AAL1 and AAL5. The first reason is used in streaming audio and video communication; the second, in data communications.

AAL1: AAL1 supports applications that transfer information at constant bit rates, such as video and voice. It allows ATM to connect existing digital telephone networks such as voice channels and T lines.

AAL2: Originally AAL2 was intended to support a variable-data-rate bit stream, but it has been redesigned. It is now used for low-bit-rate traffic and short-frame traffic such as audio (compressed or uncompressed), video, or fax.

AAL3/4: AAL3/4 Initially, AAL3 was intended to support connection-oriented data services and AAL4 to support connectionless services. As they evolved, however, it became evident that the fundamental issues of the two protocols were the same. They have therefore been combined into a single format calledAAL3/4.

AAL5: AALS AAL3/4 provides comprehensive sequencing and error control mechanisms that are not necessary for every application. For these applications, the designers of ATM have provided a fifth AAL sub layer, called the simple and efficient adaptation layer (SEAL). AAL5 assumes that all cells belonging to a single message travel sequentially and that control functions are included in the upper layers of the sending application.

SONET/SDH

The ANSI standard is called the Synchronous Optical Network (SONET). The ITU-T standard is called the Synchronous Digital Hierarchy (SOH). SONET was developed by ANSI; SDH was developed by ITU-T. SONET/SDH is a synchronous network using synchronous TDM multiplexing. All clocks in the system are locked to a master clock.

ARCHITECTURE

Architecture of a SONET system contains: signals, devices, and connections.

Signals: SONET defines a hierarchy of electrical signaling levels called synchronous transport signals (STSs). Each STS level (STS-l to STS-192) supports a certain data rate, specified in megabits per second. The corresponding optical signals are called optical carriers (OCs). SDH specifies a similar system called a synchronous transport module (STM).

SONET Devices: SONET transmission relies on three basic devices: STS multiplexers/demultiplexers, regenerators, add/drop multiplexers and terminals.

<u>STS Multiplexer/Demultiplexer:</u> It marks the beginning points and endpoints of a SONET link. They provide the interface between an electrical tributary network and the optical network. An STS multiplexer multiplexes signals from multiple electrical sources and creates the corresponding OC signal. An STS demultiplexer demultiplexes an optical OC signal into corresponding electric signals.

Regenerator: Regenerators extend the length of the links. A regenerator is a repeater that takes a received optical signal *(OC-n),* demodulates it into the corresponding electric signal *(STS-n),* regenerates the electric signal, and finally modulates the electric signal into its correspondent *OC-n* signal. A SONET regenerator replaces some of the existing overhead information (header information) with new information.

Add/drop Multiplexer: It allows insertion and extraction of signals. An **add/drop multiplexer (ADM)** can add STSs coming from different sources into a given path or can remove a desired signal from a path and redirect it without demultiplexing the entire signal. Instead of relying on timing and bit positions, add/drop multiplexers use header information such as addresses and pointers (described later in this section) to identify individual streams.

In the simple configuration shown by Figure, a number of incoming electronic signals are fed into an STS multiplexer, where they are combined into a single optical signal. The optical signal is transmitted to a regenerator, where it is recreated without the noise it has picked up in transit. The regenerated signals from a number of sources are then fed into an add/drop multiplexer. The add/drop multiplexer reorganizes these signals, if necessary, and sends them out as directed by information in the data frames. These demultiplexed signals are sent to another regenerator and from there to the receiving STS demultiplexer, where they are returned to a format usable by the receiving links.

Terminals: A **terminal** is a device that uses the services of a SONET network. For example, in the Internet, a terminal can be a router that needs to send packets to another router at the other side of a SONET network.

Connections: The devices are connected using *sections, lines,* and *paths.*

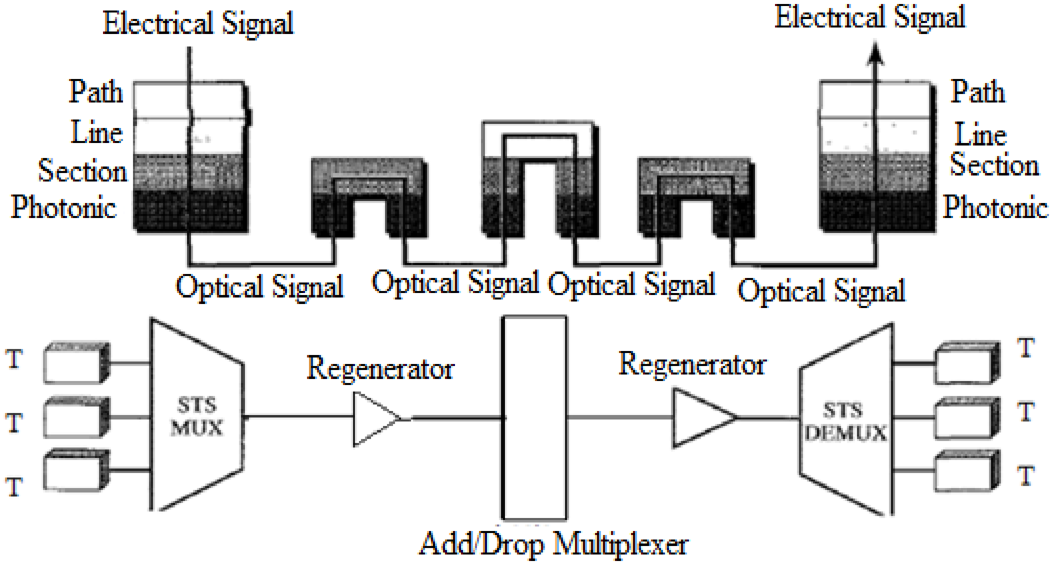
> ***Sections:*** A section is the optical link connecting two neighbor devices: multiplexer to multiplexer, Multiplexer to regenerator, or regenerator to regenerator.
>
> ***Lines:*** A line is the portion of the network between two multiplexers: STS multiplexer to add/drop multiplexer, two add/drop multiplexers, or two STS multiplexers.
>
> ***Paths:*** A path is the end-to-end portion of the network between two STS multiplexers. In a simple SONET of two STS multiplexers linked directly to each other, the section, line, and path are the same.

SONET LAYERS

The SONET standard includes four functional layers: the photonic, the section, the line, and the path layer. The headers added to the frame at the various layers are discussed later in this chapter. SONET defines four layers: path, line, section, and photonic.

Path Layer: The path layer is responsible for the movement of a signal from its optical source to its optical destination. At the optical source, the signal is changed from an electronic form into an optical form, multiplexed with other signals, and encapsulated in a frame. At the optical destination, the received frame is demultiplexed, and the individual optical signals are changed back into their electronic forms. Path layer overhead is added at this layer. STS multiplexers provide path layer functions.

Figure 1.22 Device- Layer Relationship in SONET



Line Layer: The **line layer** is responsible for the movement of a signal across a physical line. Line layer overhead is added to the frame at this layer. STS multiplexers and add/drop multiplexers provide line layer functions.

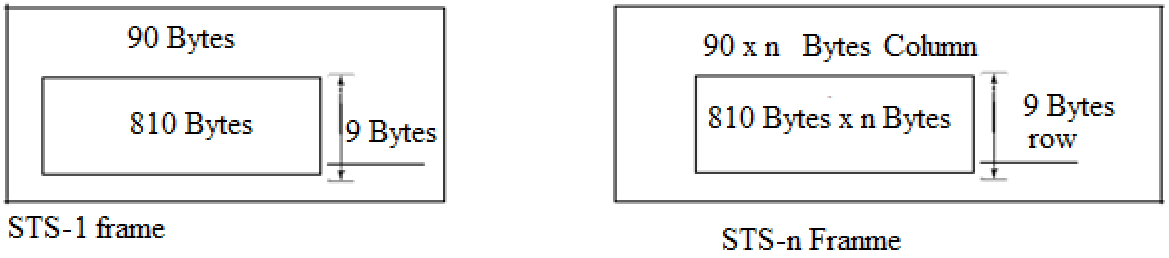
Section Layer: The **section layer** is responsible for the movement of a signal across a physical section. It handles framing, scrambling, and error control. Section layer overhead is added to the frame at this layer.

Photonic Layer: The **photonic layer** corresponds to the physical layer of the OSI model. It includes physical specifications for the optical fiber channel, the sensitivity of the receiver, multiplexing functions, and so on. SONET uses NRZ encoding with the presence of light representing 1 and the absence of light representing O.

SONET FRAMES

Each synchronous transfer signal *STS-n* is composed of 8000 frames. Each frame is a two-dimensional matrix of bytes with 9 rows by 90 x *n* columns. For example, STS-l frame is 9 rows by 90 columns (810 bytes), and an STS-3 is 9 rows by 270 columns (2430 bytes). Figure 17.4 shows the general format of an STS-l and an *STS-n.*

Figure 1.23 An STS-1 and STS-n Frame



A SONET *STS-n* signal is transmitted at 8000 frames per second. If we sample a voice signal and use 8 bits (l byte) for each sample, we can say that each byte in a SONET frame can carry information from a digitized voice channe1. In other words, an STS-l signal can carry 774 voice channels simultaneously (810 minus required bytes for overhead). Each byte in a SONET frame can carry a digitized voice channel.

BLUETOOTH

Bluetooth is a wireless LAN technology designed to connect devices of different functions such as telephones, notebooks, computers (desktop and laptop), cameras, printers, coffee makers, and so on. A Bluetooth LAN is an ad hoc network, formed spontaneously; the devices called gadgets find each other and make a network called a Pico-net. A Bluetooth LAN can even be connected to the Internet if one of the gadgets has this capability. A Bluetooth LAN, by nature, cannot be large. If there are many gadgets that try to connect, there is chaos.

Bluetooth Applications: Peripheral devices such as a wireless mouse or keyboard can communicate with the computer through this technology. Monitoring devices can communicate with sensor devices in a small health care center. Home security devices can use this technology to connect different sensors to the main security controller. Conference attendees can synchronize their laptop computers at a conference.

Bluetooth was originally started as a project by the Ericsson Company. It is named for Harald Blaatand, the king of Denmark (940-981) who united Denmark and Norway. *Blaatand* translates to *Bluetooth* in English. Today, Bluetooth technology is the implementation of a protocol defined by the IEEE 802.15 standard. The standard defines a wireless personal-area network (PAN) operable in an area the size of a room or a hall.

Bluetooth Layers and Architecture

Bluetooth defines two types of networks: Piconet and Scatternet.

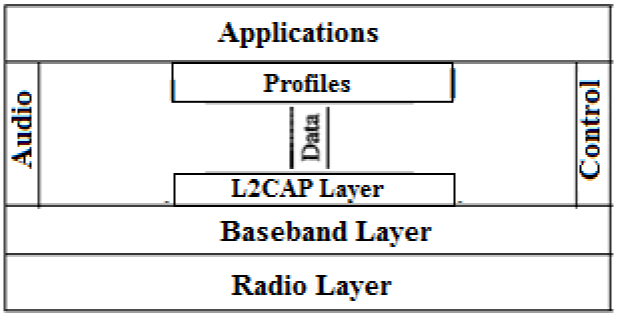
Piconets: A Bluetooth network is called a Piconet, or a small net. A Piconet can have up to eight stations, one of which is called the primary; the rest are called secondaries. All the secondary stations synchronize their clocks and hopping sequence with the primary. Note that a Piconet can have only one primary station.

Scatternet: Piconets can be combined to form what is called a Scatternet. A secondary station in

One Piconet can be the primary in another Piconet. This station can receive messages from the primary in the first Piconet (as a secondary) and, acting as a primary, deliver them to secondaries in the second Piconet. A station can be a member of two Piconets.

Bluetooth Devices: A Bluetooth device has a built-in short-range radio transmitter. The current data rate is 1 Mbps with a 2.4-GHz bandwidth. This means that there is a possibility of interference between the IEEE 802.11b wireless LANs and Bluetooth LANs. Bluetooth uses several layers that do not exactly match those of the Internet model we have defined in this book. Figure 1.24 shows these layers.

Figure 1.24 Bluetooth Layers



Radio Layer

The radio layer is roughly equivalent to the physical layer of the Internet model. Bluetooth devices are low-power and have a range of 10 m.

Band: Bluetooth uses a 2.4-GHz ISM band divided into 79 channels of 1 MHz each.
FHSS: Bluetooth uses the frequency-hopping spread spectrum (FHSS) method in the physical layer to avoid interference from other devices or other networks. Bluetooth hops 1600 times per second (Device changes its modulation frequency 1600 times per second).
Modulation: To transform bits to a signal, Bluetooth uses a sophisticated version of FSK, called GFSK (FSK with Gaussian bandwidth filtering).

The Bluetooth Baseband Layer

It is roughly equivalent to the MAC sub layer in LANs. The access method is TDMA. The primary and secondary communicate with each other using time slots. It turns the raw bit stream into frames and defines some key formats.

In the simplest form, the master in each Piconet defines a series of 625 µsec time slots, with the master's transmissions starting in the even slots and the slaves' transmissions starting in the odd ones. This is traditional time division multiplexing, with the master getting half the slots and the slaves sharing the other half. Frames can be 1, 3, or 5 slots long.

FHSS: The frequency hopping timing allows a settling time of 250–260 µsec per hop to allow the radio circuits to become stable. Faster settling is possible, but only at higher cost.

TDMA: Bluetooth uses a form of TDMA (see Chapter 12) that is called TDD-TDMA (time division duplex TDMA). TDD-TDMA is a kind of half-duplex communication in which the secondary and receiver send and receive data, but not at the same time (half duplex).

Physical Links: Each frame is transmitted over a logical channel, called a **link**, between the master and a slave. Two kinds of links exist. The first is the **ACL** (**Asynchronous Connection-Less**) link, which is used for packet-switched data available at irregular intervals. ACL traffic is delivered on a best-efforts basis. No guarantees are given. Frames can be lost and may have to be retransmitted. A slave may have only one ACL link to its master.

The other is the **SCO** (**Synchronous Connection Oriented**) link, for real-time data, such as telephone connections. This type of channel is allocated a fixed slot in each direction. Due to the time-critical nature of SCO links, frames sent over them are never retransmitted. Instead, Forward error correction can be used to provide high reliability. A slave may have up to three SCO links with its master. Each SCO link can transmit one 64,000 bps PCM audio channel.

The Bluetooth L2CAP Layer

The Logical Link Control and Adaptation Protocol, or L2CAP (L2 here means LL), is roughly equivalent to the LLC sub layer in LANs. It is used for data exchange on an ACL link; SCO channels do not use L2CAP. The L2CAP has specific duties: multiplexing, segmentation and reassembly, quality of service (QoS), and group management.

Multiplexing: The L2CAP can do multiplexing. At the sender site, it accepts data from one of the upper-layer protocols, frames them, and delivers them to the baseband layer. At the receiver site, it accepts a frame from the baseband layer, extracts the data, and delivers them to the appropriate protocol layer. It creates a kind of virtual channel that we will discuss in later chapters on higher-level protocols.

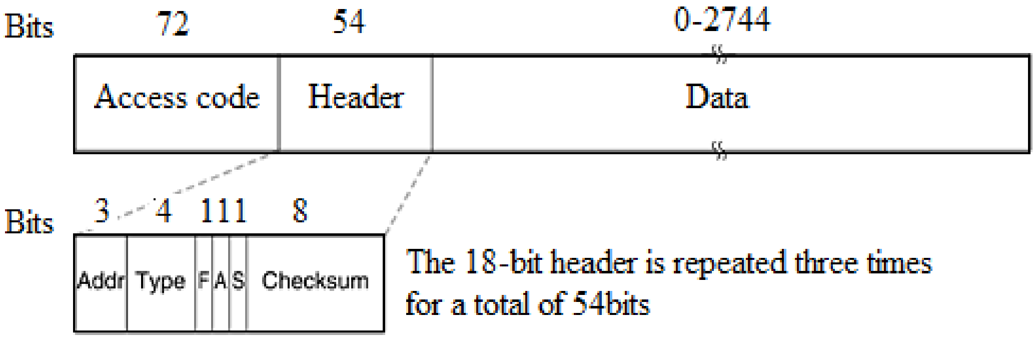
Segmentation and Reassembly: The maximum size of the payload field in the baseband layer is 2774 bits, or 343 bytes. This includes 4 bytes to define the packet and packet length. Therefore, the size of the packet that can arrive from an upper layer can only be 339 bytes. However, application layers sometimes need to send a data packet that can be up to 65,535 bytes (an Internet packet, for example). The L2CAP divides these large packets into segments and adds extra information to define the location of the segments in the original packet. The L2CAP segments the packet at the source and reassembles them at the destination.

QoS: Bluetooth allows the stations to define a quality-of-service level. We discuss quality of service in Chapter 24. For the moment, it is sufficient to know that if no quality-of-service level is defined, Bluetooth defaults to what is called *best-effort* service; it will do its best under the circumstances.

Group Management: Another functionality of L2CAP is to allow devices to create a type of logical addressing between themselves. This is similar to multicasting. For example, two or three secondary devices can be part of a multicast group to receive data from the primary.

Bluetooth Frame Structure: The frame begins with an access code that usually identifies the master so that slaves within radio range of two masters can tell which traffic is for them. The next 54-bit header contains typical MAC sub layer fields. The payload has 2744 bits for a five-slot transmission and 240 bits for a single time slot.

Figure 1.25 Bluetooth Frame Format



Access code: This 72-bit field normally contains synchronization bits and the identifier of the primary to distinguish the frame of one Piconet from another.

Header: The header has three identical 18-bit sections. Each pattern has the subfields given below. The receiver compares these three sections, bit by bit. If each of the corresponding bits is the same, the bit is accepted; if not, the majority opinion rules. This double error control is needed because the nature of the communication, via air, is very noisy. Note that there is no retransmission in this sub layer.

1. Address. The 3-bit address subfield can define up to seven secondaries (l to 7). If the address is zero, it is used for broadcast from the primary to all secondaries.

2. Type. The 4-bit type subfield defines the type of data coming from the upper layers.

3. F. This I-bit subfield is for flow control. When set (I), it indicates that the device is unable to receive more frames (buffer is full).

4. A. This I-bit subfield is for acknowledgment. Bluetooth uses Stop-and-Wait ARQ; I bit is sufficient for acknowledgment.

5. S. This I-bit subfield holds a sequence number. Bluetooth uses Stop-and-Wait ARQ; I bit is sufficient for sequence numbering.

6. HEC. The 8-bit header error correction subfield is a checksum to detect errors in each 18-bit header section.

Payload: This subfield can be 0 to 2740 bits long. It contains data or control information corning from the upper layers.

Other Upper Layers

Bluetooth defines several protocols for the upper layers that use the services of L2CAP; these protocols are specific for each purpose.

Ultra-Wide Band:

Ultra-wide band (UWB) radios take a drastically different approach from Bluetooth and 802.15.4. Where the latter two radios emit signals over long periods using a small part of the spectrum, UWB takes the opposite approach: UWB uses short pulses (in the ps to ns range) over a large bandwidth (often many GHz).

UWB radios offer very high data rates (hundreds of Mbps or even several Gbps) with relatively low power consumption. The use of short pulses over a wide spectrum also means that the signal is below the average power output defined as noise by the FCC (-41.3 dBm/MHz), and that UWB signals are not susceptible to noise or jamming. UWB is a much simpler technology than Bluetooth and ZigBee, since there are currently no mandatory or optional middleware layers that build on top of the basic PHY and MAC layers.

There are currently two major competing UWB standards.

Direct Sequence-UWB (UWB Forum): It is the more straightforward of the two approaches. DS-UWB radios use a single pulse in one of two different spectra. These pulses may occur in the spectrum from 3.1 GHz - 4.85 GHz, or at 6.2 GHz - 9.7 GHz.

Multi-Band OFDM (WiMedia): Multi-Band Orthogonal Frequency Division Multiplexing (MB-OFDM) uses a slightly different approach to signaling from DS-UWB. Rather than using a single pulse over a wide band, MB-OFDM divides the spectrum into multiple sub-bands. MB-OFDM's frequency-hopping required complicated synchronization schemes and made it less susceptible to interference from neighboring UWB PANs.

UWB Security: UWB radios are somewhat inherently secure, because their low output power and short pulses make their transmissions appear to be white noise from a distance. Nevertheless, UWB signals could potentially be sniffed by a determined attacker who is located close to the transmitter; this mandates the use of security at the MAC layer.

Standardization Efforts: The IEEE 802.15.3a Task Group [IEEE802.15.3a] was formed in 2003 to create a common, industry-wide standard for UWB devices. Unfortunately, the group quickly divided into opposing camps.

Applications and Future Outlook: UWB is mainly advocated as a cable-replacement technology like Bluetooth, except for devices with much higher data-rate requirements. Examples are wireless USB hub using the DS-UWB [Belkin06]. The USB Implementers Forum is developing an official Wireless USB standard, which will sit on top of the WiMedia stack and provide USB 2.0-like speeds of 480 Mbps when devices are within 3 m [USB06]. Finally, because UWB's data rate is high enough to support HDTV streams, it is replacement for audio/video cables [Nekoogar05].

Wi-Fi

Wi-Fi means Wireless Fidelity. It describes only narrow range of connectivity ensuring Wireless Local Area Network with IEEE 802.11 Standard. Establish and enforce standards for Interoperability and backward compatibility

IEEE Standard of Wi-Fi

Wi-Fi Networks use Radio Technologies to transmit & receive data at high speed:

IEEE 802.11a

- Introduced in 2001
- Operates at 5 GHz (less popular)
- 54 Mbps (theoretical speed)
- 15-20 Mbps (Actual speed)
- 50-75 feet range
- More expensive
- Not compatible with 802.11b

IEEE 802.11b

- Appear in late 1999
- Operates at 2.4GHz radio spectrum
- 11 Mbps (theoretical speed) - within 30 m Range
- 4-6 Mbps (actual speed) and 100 -150 feet range
- Most popular, Least Expensive
- Has 11 channels, with 3 non-overlapping
- Interference from mobile phones and Bluetooth devices which can reduce the transmission speed.

IEEE 802.11g

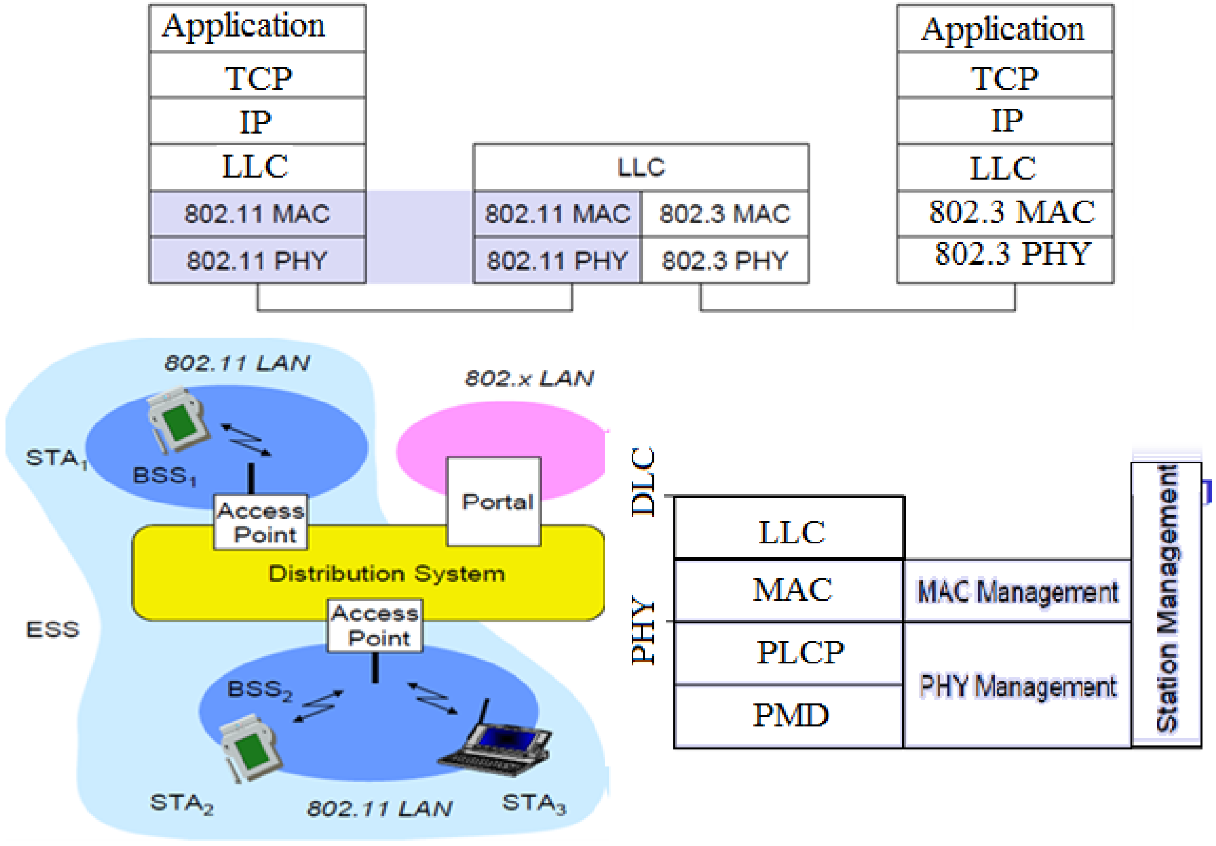
- Introduced in 2003
- Combine the feature of both standards (a,b)
- 100-150 feet range
- 54 Mbps Speed
- 2.4 GHz radio frequencies
- Compatible with ‘b’

IEEE 802.11n

- Introduced in 2009
- Improve Network throughput over 802.11a and 802.11g
- 175 feet range and 300 Mbps speed
- Multiple Input Multiple Output (MIMO) added
- 40 MHz channels to the PHY (physical layer), and frame aggregation to the MAC layer
- 2.4/5 GHz radio frequencies

802.11 - Architecture of an infrastructure network

Figure 1.26a IEEE 802.11 Architecture and protocol for Wi-Fi



Station (STA): Access mechanisms to the wireless medium and radio contact to the access point

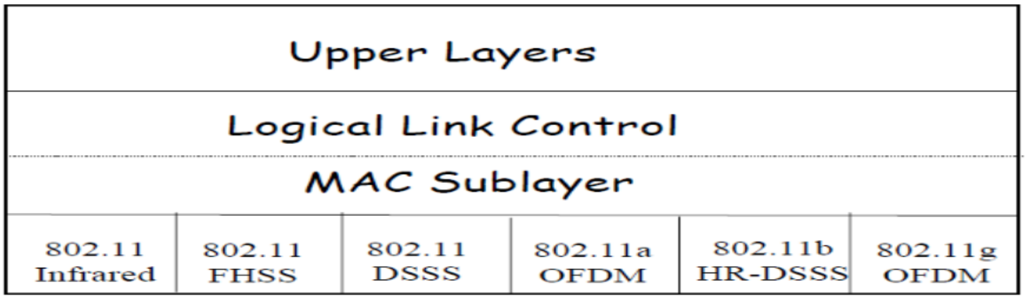
Basic Service Set (BSS): Group of stations using the same radio frequency

Access Point: Station integrated into the wireless LAN and the distribution system

Portal: bridge to other (wired) networks

Distribution System: Interconnection network to form one logical network (EES: Extended Service Set) based on several BSS

802.11 - Layers and functions



Upper Layers					
Logical Link Control					
MAC Sublayer					
802.11 Infrared	802.11 FHSS	802.11 DSSS	802.11a OFDM	802.11b HR-DSSS	802.11g OFDM

PLCP: Physical Layer Convergence Protocol provides Clear channel assessment signal.

PMD: Physical Medium Dependent provides the Modulation and coding function.
PHY Management: Channel selection and MIB are performed here. It also provides the Station Management for the Coordination of all functions. Management functions includes MAC Access mechanisms, fragmentation and Encryption
MAC Management: It includes Synchronization, roaming, MIB, power management

802.11 Physical Layer

- Physical layer corresponds to OSI stack well
- Five different physical layers are proposed
- Data link layer split in two or more sub layers e.g. MAC and Logical link control sub layers

  MAC allocates the channel
  LLC hides differences between different physical layers to network layer

WiMAX

WiMAX stands for Worldwide Interoperability for Microwave Access and is an IP based, wireless broadband access technology that provides performance similar to 802.11/Wi-Fi networks with the coverage and QOS (quality of service) of cellular networks.

WiMAX is a Protocol or A standard based technology that provides fixed and mobile Internet with the delivery of last mile wireless broadband access as an alternative to DSL.

Features of WiMAX

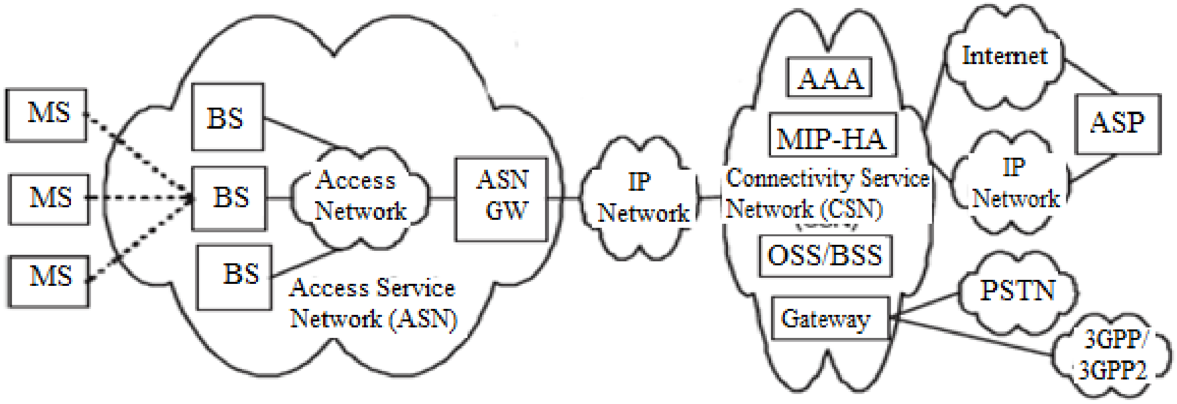
It provides fixed, nomadic, portable and eventually mobile wireless broadband without the need for direct LOS to base station. Current WiMAX revision provides up to 40Mbps in typical 3-10 km base station radius. Current WiMAX revision is based upon IEEE Std 802.16e-2005. Actual Standard is IEEE STD 802.16d-2004, IEEE 802.16e-2005 improves upon IEEE 802.16-2004 by:

- Adding Support for Mobility
- Scaling of the Fast Fourier Transform (FFT) to the channel bandwidth
- Adaptive Antenna Systems (AAS) and MIMO Technology
- Adding an extra QOS for VOIP Applications
- Introducing downlink sub-channelization

WiMAX Network Architecture and Protocol Model

WiMAX architecture consists of two types of fixed (non mobile) stations:
Subscriber Stations (SS): serves a building (business or residence)
Base station (BS): connects to public network and provide SS with first-mile access to public networks

Figure 1.26 WiMAX Network Architecture



The communication path between SS and BS has two directions:

- Uplink (from SS to BS)
- Downlink (from BS to SS)

Mobile Stations (MS) used by the end user to access the network. The access service network (ASN), which comprises one or more base stations and one or more ASN gateways that form the radio access network at the edge. Connectivity service network (CSN), which provides IP connectivity and all the IP core network functions.

WiMAX Physical Layer

Physical layer functions are encoding/decoding of signals, preamble generation/removal, and bit transmission/reception.

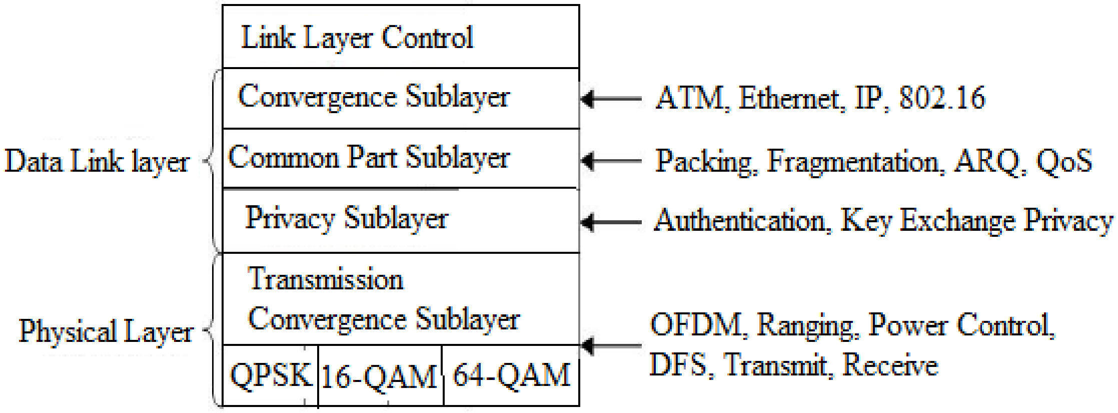
The physical layer supports:

OFDM: Orthogonal Frequency Division Multiplexing TDD: Time Division Duplex
FDD: Frequency Division Duplex QPSK: Quadrature Phase Shift Keying

Some features of Physical layer:

- Based on orthogonal frequency division multiplexing (OFDM)
- OFDM is the transmission scheme of choice to enable high-speed data, video, and multimedia communications and is used by a variety of commercial broadband systems
- OFDM is an elegant and efficient scheme for high data rate transmission in a non-line-of-sight or multipath radio environment.

Figure 1.27 WiMAX Layers



WiMAX MAC Layer

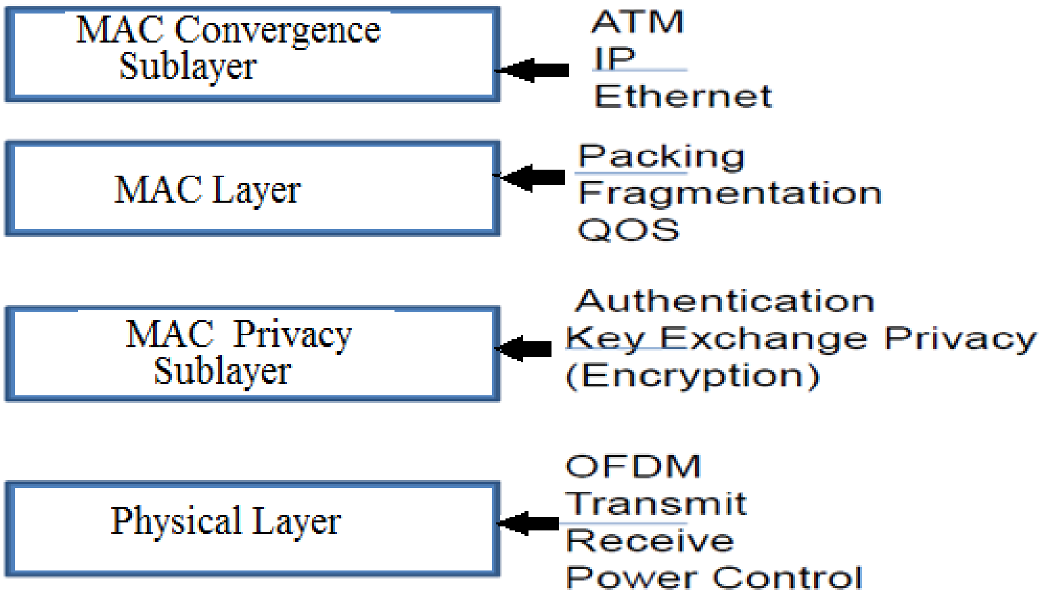
WiMAX MAC layer is a point to multipoint protocol (P2MP). It supports high bandwidth and hundreds of users per channel. It utilizes spectrum efficiently by supporting bursty traffic. The MAC convergence sub layer offers support for ATM, Ethernet, 802.1Q, IPv4, IPv6 (a possible future support for PPP, MPLS etc). The core MAC layer provides packet fragmentation, ARQ and QOS. The MAC Privacy Sub layer integrates security features in WiMAX. Authentication, encryption and Key exchange functionality are provided in MAC sub layer.

In the *Data link layer*, medium access control functions are:

□ On transmission, assemble data into a frame with address and error detection fields

□ On reception, disassemble frame, and perform address recognition and error detection

□ Govern access to the wireless transmission medium



Privacy Sub layer functions

1.Encrypt or decrypt data	2.Secure distribution of keying data (BS to SS)
Privacy Key Management (PKM)	1) Security Association (SA)
3.Encapsulation protocol	4.Identified by SAID

<u>The convergence layer, functions are:</u> Encapsulate PDU framing of upper layers into native 802.16 MAC/PHY frames, map upper layers addresses into 802.16 addresses, translate upper layer QoS parameters into native 802.16 MAC format, and adapt time dependencies of upper layer traffic into equivalent MAC service.

Other responsibilities:

- The IEEE 802.16 MAC was designed for point-to-multipoint broadband wireless access applications.
- Provide an interface between the higher transport layers and the physical layer.
- MAC service data units (MSDUs).and organizes them into MAC protocol data units (MPDUs) for transmission over the air.
- Broadcast and multicast support and Manageability primitives.
- High-speed handover and mobility management primitives.
- Three power management levels, normal operation, sleep and idle.
- Header suppression, packing and fragmentation for efficient use of spectrum.

WiMAX Applications

WiMAX provides Portable broadband connectivity across cities through variety of devices. The other applications include:

- Wireless alternative for DSL and cable
- Providing data communications (VOIP) and IPTV Service (Tripple Play)
- Providing source of Internet connectivity as part of a business continuity plan
- Enterprise Data Service and Peer to Peer access and Varieties VAS.

Cognitive Radio (CR)

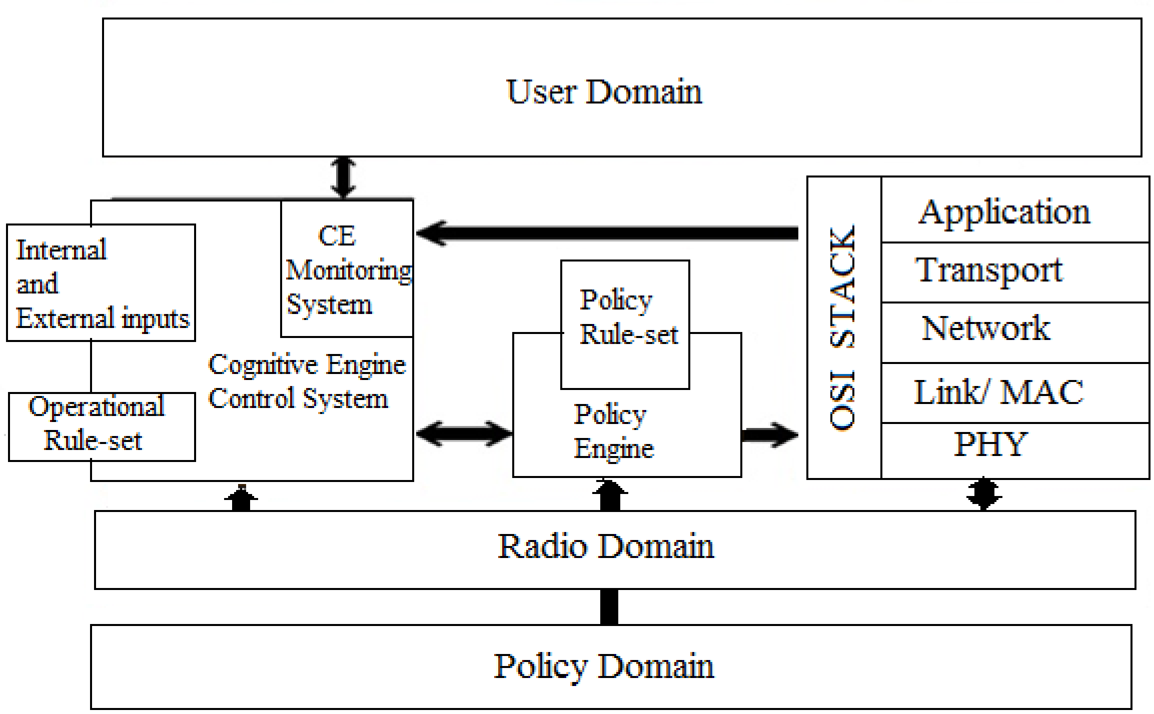
The term cognitive radio first was used publicly in an article by Joseph Mitola, where it was defined as *"The point in which wireless personal digital assistants (PDAs) and the related networks are sufficiently computationally intelligent about radio resources and related computer-to-computer communications to detect user communications needs as a function of use context, and to provide radio resources and wireless services most appropriate to those needs."*

The definition was developed in the context of a software-defined radio (SDR), where the radio could easily be reconfigured to operate on different frequencies with different protocols by software re-programming. Software-Defined Radio (SDR) Forum and International Telecommunications Union-Radio Sector (ITU-R) are working in this area.

Cognitive Radio (CR) Concept and Architecture

A simple CR system might have a single reconfigurable radio component accepting sensing information from a single local node and no external data sources. There are two major subsystems in a cognitive radio; a cognitive unit that makes decisions based on various inputs and a flexible SDR unit whose operating software provides a range of possible operating modes. A separate spectrum sensing subsystem is also often included in the architectural a cognitive radio to measure the signal environment to determine the presence of other services or users. It is important to note that these subsystems do not necessarily define a single piece of equipment, but may instead incorporate components that are spread across an entire network. As a result, cognitive radio is often referred to as a cognitive radio system or a cognitive network.

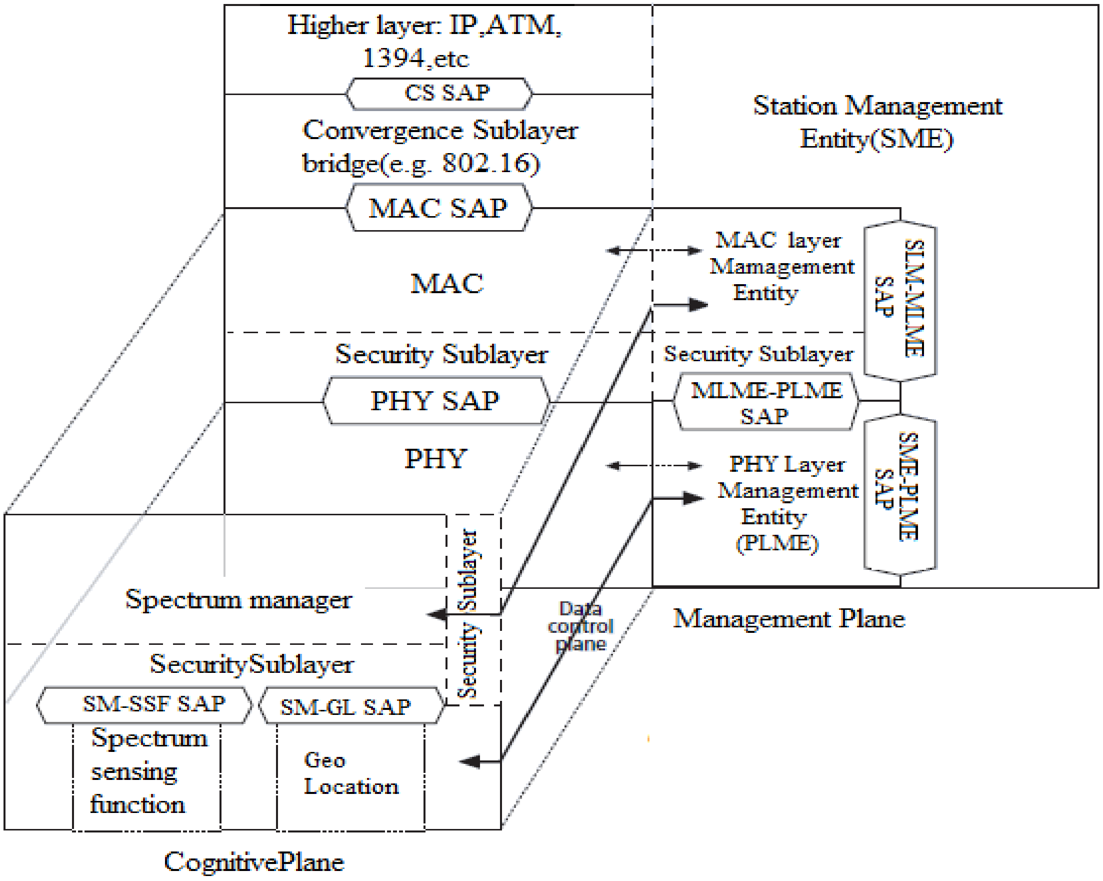
Figure 1.3 Cognitive Radio Architecture



The cognitive unit is further separated into two parts as shown in the block diagram below. The first labeled the “cognitive engine” tries to find a solution or optimize a performance goal based on inputs received defining the radio’s current internal state and operating environment. The second engine is the “policy engine” and is used to ensure that the solution provided by the “cognitive engine” is in compliance with regulatory rules and other policies external to the radio.

Cognitive Radio Protocol Stack (IEEE 802.22)

In May 2004 (LAN/MAN) Standards committee created the 802.22 WG on wireless regional area networks (WRANs) with a CR-based air interface for use by license-exempt devices on a non-interfering basis in VHF and UHF (54–862 MHz) bands. IEEE 802.22 will be the first complete cognitive radio-based international standard with frequency bands allocated for its use. Figure 4a shows the protocol referenced model (PRM) for a CR node that is likely to be adopted by the 802.22 WG.

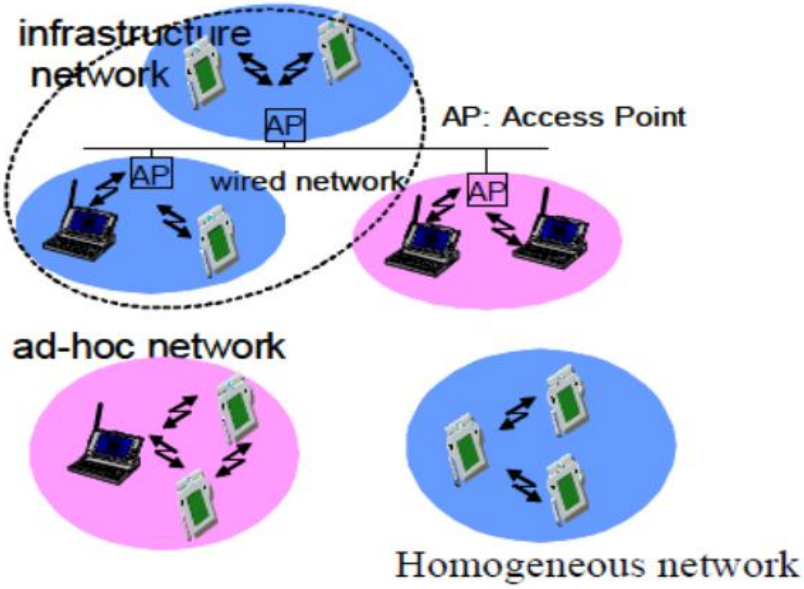


PRM defines the system architecture, functionalities of various blocks, and their mutual interactions. The PRM separates the system into the cognitive, data/control, and management planes. The spectrum-sensing function (SSF) and geo-location function that interface with the RF stage of the device provide information to the spectrum manager (SM) on the presence of incumbent signals, as well as its current location. The SM function makes decisions on transmission of the information-bearing signals. The SM at the subscriber location is called the spectrum automaton (SA), because it is assumed that almost all of the intelligence and the decision-making capability will reside at the SM of the base station.

The PHY, MAC, and convergence layers are essentially the same as in 802.16. Security sub-layers are added between service access points (SAPs) to provide enhanced protection. The 802.22 WG has adopted many of the PHY, MAC, security, and quality of service (QoS) features from the IEEE 802.16-2004 and 802.16e standards with some essential modifications due to the different propagation and operational scenario characteristics for WRANs. Because signals at VHF/UHF travel longer distances than those at higher frequencies, various WRAN cells using similar frequencies are likely to create co-channel interference. Hence, as compared to other standards, 802.22 have been quite proactive in addressing the issue of self coexistence. Cognitive radio still is an active area of research, and as research progresses and comes into practice, further standardization work will be required to facilitate adoption of new techniques.

Ad Hoc Networks

Ad Hoc" is actually a Latin phrase that means "for this purpose." It is often used to describe solutions that are developed on-the-fly for a specific purpose. In computer networking, an ad hoc network refers to a network connection established for a single session and does not require a router or a wireless base station. For example, If you need to share files with more than one computer, you could set up a mutli-hop ad hoc network, which can transfer data over multiple nodes.



Basically, an ad hoc network is a temporary network connection created for a specific purpose (such as transferring data from one computer to another). If the network is set up for a longer period of time, it is just a plain old local area network (LAN). An ad hoc network typically refers to any set of networks where all devices have equal status on a network and are free to associate with any other ad hoc network devices in link range. Very often, ad hoc network refers to a mode of operation of IEEE 802.11 wireless networks. Two topologies: Heterogeneous (Differences in capabilities) and Homogeneous or fully symmetric (Right).

A wireless ad-hoc network is a decentralized type of wireless network. The network is ad hoc because it does not rely on a preexisting infrastructure, such as routers in wired networks or access points in managed (infrastructure) wireless networks. Instead, each node participates in routing by forwarding data for other nodes, and so the determination of which nodes forward data is made dynamically based on the network connectivity. In addition to the classic routing, ad hoc networks can use flooding for forwarding the data.

CLASSIFICATION OF Ad Hoc NETWORK:

Wireless ad hoc networks can be further classified by their application:

1. **Wireless Mesh Network (WMN):** WMN made up of radio nodes organized in a mesh topology and consist of mesh clients, mesh routers and gateways. The mesh clients are often laptops, cell phones and other wireless devices while the mesh routers forward traffic to and from the gateways which may but need not connect to the Internet.

2. **Wireless Sensor Networks (WSN**): WSN consists of spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, humidity, motion or pollutants and to cooperatively pass their data through the network to a main location. The more modern networks are bi-directional, also enabling control of sensor activity. Applications include battlefield surveillance (Military), industrial process monitoring and control, machine health monitoring, and so on.

3. **Mobile Ad Hoc Network (MANET)**: A mobile ad-hoc network is a self-configuring infrastructure less network of mobile devices connected by wireless links. Ad-hoc is Latin and means "for this purpose". Each device in a MANET is free to move independently in any direction, and will therefore change its links to other devices frequently. Each must forward traffic unrelated to its own use, and therefore be a router. The primary challenge in building a MANET is equipping each device to continuously maintain the information required to properly route traffic.

Types of MANET

Vehicular Ad-hoc Networks (VANETs): A Vehicular Ad-Hoc Network or VANET is a technology that uses moving cars as nodes in a network to create a mobile network. As cars fall out of the signal range and drop out of the network, other cars can join in, connecting vehicles to one another so that a mobile Internet is created.

Internet Based Mobile Ad-hoc Networks (iMANET): iMANET are used to link mobile nodes and fixed Internet-gateway nodes. In such type of networks normal ad hoc routing algorithms don't apply directly. By nature these types of networks are suitable for situations where either no fixed infrastructure exists, or to deploy one is not possible.

Intelligent vehicular ad-hoc networks (InVANETs): InVANET integrates on multiple ad-hoc networking technologies such as WiFi IEEE 802.11, WiMAX IEEE 802.16, Bluetooth, IRA, ZigBee for easy, accurate communication between vehicles on dynamic mobility. InVANET

Applications include Intelligent Transportation Systems (ITS), infotainment and telematics, Dedicated Short Range Communications (DSRC), Cellular, Satellite, and WiMAX.

Sensor Networks

A sensor network is composed of a large number of sensor nodes that are densely deployed. To list just a few venues, sensor nodes may be deployed in an open space; on a battlefield in front of, or beyond, enemy lines; in the interior of industrial machinery; at the bottom of a body of water; in a biologically and/or chemically contaminated field; in a commercial building; in a home; or in or on a human body.

A sensor node typically has embedded processing capabilities and onboard storage; the node can have one or more sensors operating in the acoustic, seismic, radio (radar), infrared, optical, magnetic, and chemical or biological domains. The node has communication interfaces, typically wireless links, to neighboring domains. The sensor node also often has location and positioning knowledge that is acquired through a global positioning system (GPS) or local positioning algorithm. Figure 1.2 depicts a typical WSN arrangement.

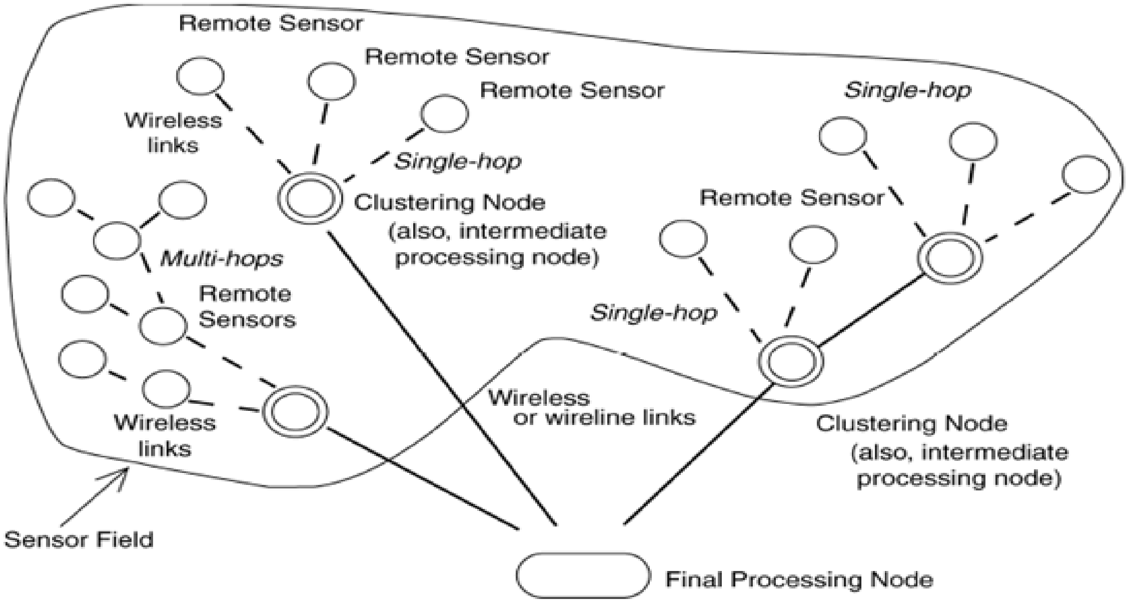


Figure 1.28 Typical Sensor Network

The components of a (remote) sensing node include the following:
_ A sensing and actuation unit (single element or array)
_ A processing unit
_ A communication unit
_ A power unit
_ Other application-dependent units

In addition to (embedded) sensing there is a desire to build, deploy, and manage unattended or un tethered embedded control and actuation systems, sometimes called control networks. Such a control system acts on the environment either in a self-autonomous manner or under the telemetry of a remote or centralized node.

Software (Operating Systems and Middleware): To support the node operation, it is important to have open-source operating systems designed specifically for WSNs. Such operating systems typically utilize a component-based architecture that enables rapid implementation and innovation while minimizing code size as required by the memory constraints endemic in sensor networks.

Standards for Transport Protocols: The goal of WSN engineers is to develop a cost-effective standards-based wireless networking solution that supports low-to medium data rates, has low power consumption, and guarantees security and reliability. The position of sensor nodes does not have been predetermined, allowing random deployment in inaccessible terrains or dynamic situations; however, this also means that sensor network protocols and algorithms must possess self-organizing capabilities.

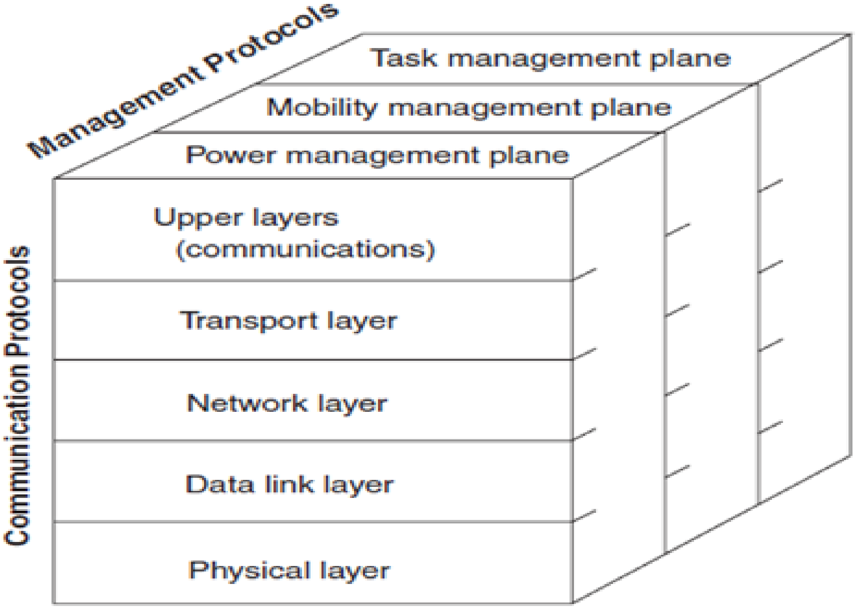


Figure 1.29 Generic Protocol Stack for Sensor Network

Upper layers: In-network applications, including application processing, data aggregation, external querying query processing, and external database
Layer 4: Transport, including data dissemination and accumulation, caching, and storage
Layer 3: Networking, including adaptive topology management and topological routing
Layer 2: Link layer (contention): channel sharing (MAC), timing, and locality
Layer 1: Physical medium, communication channel, sensing, actuation, and signal processing

Routing and Data Dissemination: Routing and data dissemination issues deal with data dissemination mechanisms for large-scale wireless networks, directed diffusion, data-centric routing, adaptive routing, and other specialized routing mechanism. Routing protocols for WSNs generally fall into three groups: data-centric, hierarchical, and location-based. The concept of data aggregation is to combine the data arriving from different sources along way.

Sensor Network Organization and Tracking: Areas of interest involving network organization and tracking include distributed group management (maintaining organization in large-scale sensor networks); self-organization, including authentication, registration, and session establishment; and entity tracking: target detection, classification, and tracking.

Computation: Computation deals with data aggregation, data fusion, data analysis, computation hierarchy, grid computing (utility-based decision making in wireless sensor networks), and signal processing.

Data Management Data management: It deals with data architectures; database management, including querying mechanisms; and data storage and warehousing. Security Security deals with confidentiality (encryption), integrity (e.g., identity management, digital signatures), and availability (protection from denial of service).

Network Design Issues: We have already noted that in sensor networks, issues relate to reliable transport (possibly including encryption), bandwidth-and power limited transmission, data-centric routing, in-network processing, and self configuration.

GREEN COMMUNICATION

In the past few years, the cellular network sector has developed rapidly. This rapid growth is due to the increases in the numbers of mobile subscribers, multimedia applications, and data rates. The data transmission rate doubles by a factor of approximately ten every five years. The increase in the number of mobile subscribers has led to an increase in data traffic; as a result, the number of base stations (BSs) has increased to meet the needs of customers. This increasing energy demand has prompted considerable research on the subject of "*green communications.*"

Perhaps the two most important reasons to pursue the development of green communications networks are increases in carbon dioxide emissions (CO2) and increases in operational expenditures (OPEX). CO2 emissions are mainly associated with off-grid sites that provide coverage for remote areas.

Cellular networks represent the largest component of the ICT sector. Energy consumption by cellular networks is expected to increase rapidly in the future if no measures are taken to alter this trend. The above-mentioned statistics have motivated researchers in both academia and industry to develop techniques to reduce the energy consumption of cellular networks, thereby maintaining profitability and making cellular networks "greener."

The goal of Green Communication associated with green cellular networks is:

(i) Improvement of energy efficiency,

(ii) Improvement of the intelligence of the network through tradeoffs between energy consumption and external conditions, that is, traffic loads,

(iii) Integration of the network infrastructure and network services to enable the network to be more responsive and to require less power to operate,

(iv) reduced carbon emissions.

Therefore, an effort is required to reduce the energy consumption of base stations, while continuing to provide the expected quality of service, taking into account the associated cost. The solutions have two components: first, the hardware solution, for which the focus is on improving the energy consumption in the BS components, such as power amplifiers (PAs), digital signal processors (DSPs), cooling systems, and feeder cables and second is the software solution includes intelligent management of network elements based on traffic load variations.

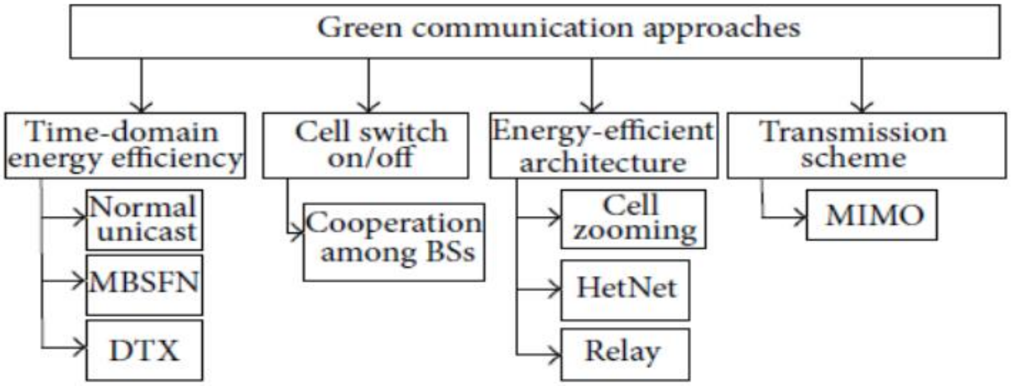


FIGURE 5: Classifications of energy-saving techniques.

Energy Efficiency Metrics

Energy efficiency metrics provide information that can be used to assess and compare the energy consumption of various components of a cellular network and of the network as a whole. These metrics also help us to set long-term research goals for reducing energy consumption.

Power Amplifier Improvement

It has been noted that many factors must be taken into account in PA design:

(i) High linearity, to satisfy higher-order modulation schemes,

(ii) Greater average output power levels,

(iii) Broader operating bandwidths, and

(iv) OPEX reductions achieved by decreasing BS energy consumption.

Time Domain Energy Efficiency

Time-domain solutions seek to reduce the PA operating time by reducing control signals during low traffic or in the idle case situation. Therefore, the amount of energy that can be saved using this approach depends on the PA off time.

Cell Switch On/Off

Switching off unused wireless resources and devices has become the most popular approach to reducing power consumption by cellular networks because it can save large amounts of energy. Cell switching is based on the traffic load conditions: if the traffic is low in a given area, some cells will be switched off, and the radio coverage and service will be provided by the remaining active cells.

Energy-Efficient Architecture

This approach is considered a special case of the cell switch off approach. The principle of this approach depends on the cooperation of the BSs to provide service to the users and provide energy savings in the event of low traffic. It includes cell zooming, heterogeneous networks (HetNets), and relay techniques.

Cell Zooming: Cell zooming is the capability of cell to allow the adjustment of the cell size according to the traffic load. When congestion occurs at the cell due to an increase in the number of UEs, the congested cell could zoom in, while the neighboring cells with smaller amounts of traffic could zoom out to provide the coverage for UEs that cannot be served by the congested cell. The major component of this design is a cell-zooming server (CS).
Advantages: This technique can improve the throughput and lengthen the UE's battery life.
Shortcomings: The BS's maximum transmission power is limited. In addition, Inter-cell interference occurs when all the neighboring cells zoom out during the same time interval.

Heterogeneous Networks (HetNets): Heterogeneous networks have a layered structure that combines different networks to serve the same mobile devices. Heterogeneous networks are intended to improve both throughput and energy consumption through the deployment of a network of cells. ***Advantages:*** A smaller cell size leads to improvement in coverage and capacity with lower costs. ***Shortcomings:*** it includes management of the interfaces between heterogeneous environments and the dead zone problem.

Relay: Relay techniques are other means of saving energy, while improving network performance. The principle of this class of techniques is based on the deployment of relay nodes between the source (BS) and the destination (UEs). ***Advantages:*** *L*ow transmission power, low interference, reduced path loss, extended handset life. ***Shortcomings:*** There is a tradeoff in achieving energy efficiency in terms of the number of nodes that are required in a relay scheme. The second issue is to select an appropriate user to act as a relay node.

Transmission Scheme: MIMO has the advantages of reducing fading and increasing throughput without it being necessary to increase either the bandwidth or the transmission power. These advantages can be achieved by introducing space-time coding (STC), which exploits spatial diversity to overcome fading by sending the signal that carries the same information through different paths.

Unit II

Data Link Control and Media Access Control

INTRODUCTION

The data link layer transforms the physical layer, a raw transmission facility, to a link responsible for node-to-node (hop-to-hop) communication. Specific responsibilities of the data link layer include framing, addressing, flow control, error control, and media access control. The data link layer divides the stream of bits received from the network layer into manageable data units called frames. The data link layer adds a header to the frame to define the addresses of the sender and receiver of the frame. 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.

Networks must be able to transfer data from one device to another with acceptable accuracy. For most applications, a system must guarantee that the data received are identical to the data transmitted. Any time data are transmitted from one node to the next, they can become corrupted in passage. Many factors can alter one or more bits of a message. Some applications require a mechanism for detecting and correcting errors

Data can be corrupted during transmission. Some applications require that errors be detected and corrected.

Some applications can tolerate a small level of error. For example, random errors in audio or video transmissions may be tolerable, but when we transfer text, we expect very high level of accuracy.

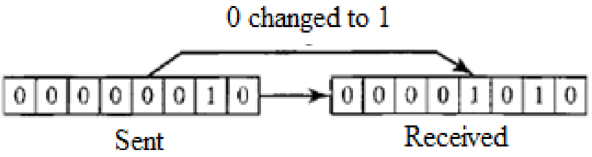
Types of Errors

Whenever bits flow from one point to another, they are subject to unpredictable changes because of interference. This interference can change the shape of the signal. In a single-bit error a and 0 is changed to a 1 or a 1 to a O. In a burst error, multiple bits are changed. For example, a 11100 s burst of impulse noise on a transmission with a data rate of 1200 bps might change all or some of the12 bits of information.

Single-Bit Error

The term *single-bit error* means that 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. Figure 10.1 shows the effect of a single-bit error on a data unit. To understand the impact of the change, imagine that each group of 8 bits is an ASCII character with a 0 bit added to the left. In Figure 10.1,00000010 (ASCII *STX)* was sent, meaning *start of text*, but 00001010 (ASCII *LF)* was received, meaning *line feed.*

Figure 2.1 Single Bit Error

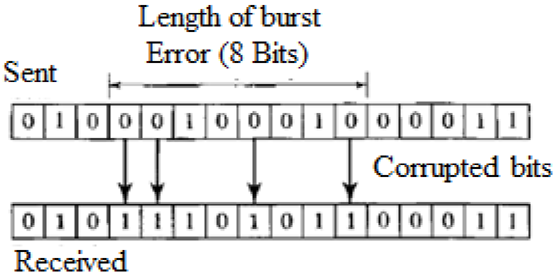


Single-bit errors are the least likely type of error in serial data transmission. To understand why, imagine data sent at 1 Mbps. This means that each bit lasts only 1/1,000,000 s, or 1)ls. For a single-bit error to occur, the noise must have a duration of only 1) ls, which is very rare; noise normally lasts much longer than this.

Burst Error:

The term *burst error* means that 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1. Figure 2.2 shows the effect of a burst error on a data unit. In this case, 0100010001000011 was sent, but 0101110101100011 was received. Note that a burst error does not necessarily mean that the errors occur in consecutive bits. The length of the burst is measured from the first corrupted bit to the last corrupted bit. Some bits in between may not have been corrupted.

Figure 2.2Burst errorof Length 8



A burst error is more likely to occur than a single-bit error. The duration of noise is normally longer than the duration of 1 bit, which means that when noise affects data, it affects a set of bits. The number of bits affected depends on the data rate and duration of noise.

Redundancy

The central concept in detecting or correcting errors is redundancy. To be able 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.

Coding: Redundancy is achieved through various coding schemes. The sender adds redundant bits through a process that creates a relationship between the redundant bits and the actual data bits. The receiver checks the relationships between the two sets of bits to detect or correct the errors. The ratio of redundant bits to the data bits and the robustness of the process are important factors in any coding scheme.

Detection versus Correction

The correction of errors is more difficult than the detection. In error detection, we are looking only to see if any error has occurred. The answer is a simple yes or no. In 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 the errors and the size of the message are important factors.

Forward Error Correction versus Retransmission

There are two main methods of error correction. Forward error correction is the process in which the receiver tries to guess the message by using redundant bits. Correction by retransmission is a technique in which the receiver detects the occurrence of an error and asks the sender to resend the message. Resending is repeated until a message arrives that the receiver believes is error-free.

CHECKSUM

The checksum is used in the Internet by several protocols although not at the data link layer. Like linear and cyclic codes, the checksum is based on the concept of redundancy. Several protocols still use the checksum for error detection, although the tendency is to replace it with a CRC. The concept of the checksum is illustrated with a few examples.

Example 1: Suppose our data is a list of five 4-bit numbers that we want to send to a destination. In addition to sending these numbers, we send the sum of the numbers. For example, if the set of numbers is (7, 11, 12, 0, 6), we send (7, 11, 12, 0, 6, 36), where 36 is the sum of the original numbers. The receiver adds the five numbers and compares the result with the sum. If the two are the same, the receiver assumes no error, accepts the five numbers, and discards the sum. Otherwise, there is an error somewhere and the data are not accepted.

Example 2: We can make the job of the receiver easier if we send the negative (complement) of the sum, called the *checksum.* In this case, we send (7, 11, 12, 0, 6, -36). The receiver can add all the numbers received (including the checksum). If the result is 0, it assumes no error; otherwise, there is an error.

Internet Checksum

Traditionally, the Internet has been using a 16-bit checksum. The sender calculates the checksum by following these steps.

Sender site:

1. The message is divided into 16-bit words.
2. The value of the checksum word is set to O.
3. All words including the checksum are added using one's complement addition.
4. The sum is complemented and becomes the checksum.
5. The checksum is sent with the data.

The receiver uses the following steps for error detection.

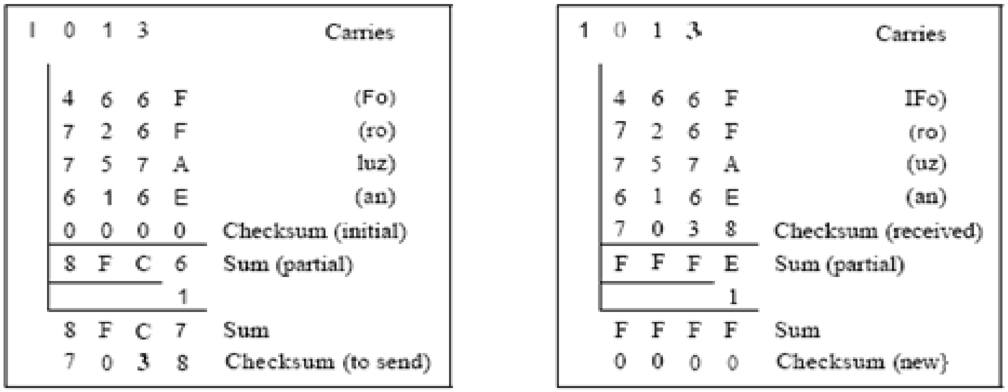
Receiver site:

1. The message (including checksum) is divided into 16-bit words.
2. All words are added using one's complement addition.
3. The sum is complemented and becomes the new checksum.
4. If the value of checksum is 0, the message is accepted; otherwise, it is rejected.

The nature of the checksum (treating words as numbers and adding and complementing them) is well-suited for software implementation. Short programs can be written to calculate the checksum at the receiver site or to check the validity of the message at the receiver site.

Example 3: Let us calculate the checksum for a text of 8 characters ("Forouzan"). The text needs to be divided into 2-byte (l6-bit) words.

Figure 2.3 Checksum illustration



I	0	1	3	Carries
4	6	6	F	(Fo)
7	2	6	F	(ro)
7	5	7	A	luz)
6	1	6	E	(an)
0	0	0	0	Checksum (initial)
8	F	C	6	Sum (partial)
			1	
8	F	C	7	Sum
7	0	3	8	Checksum (to send)

a. Checksum at the sendersite

1	0	1	3	Carries
4	6	6	F	IFo)
7	2	6	F	(ro)
7	5	7	A	(uz)
6	1	6	E	(an)
7	0	3	8	Checksum (received)
F	F	F	E	Sum (partial)
			1	
F	F	F	F	Sum
0	0	0	0	Checksum (new}

b. Checksum at the receicer site

FRAMING

The data link layer needs to pack bits into frames, so that each frame is distinguishable from another. Our postal system practices a type of framing. **Framing in the data link layer separates a message from one source to a destination, or from other messages to other destinations, 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.**

Although the whole message could be packed in one frame that is not normally done one reason is that a frame can be very large, making flow and error control very inefficient. When a message is carried in one very large frame, even a single bit error would require the retransmission of the whole message. When a message is divided into smaller frames, a single-bit error affects only that small frame.

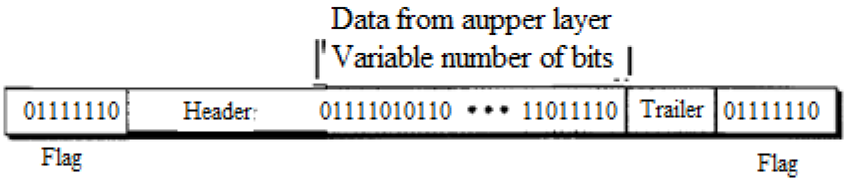
Fixed-Size Framing: Frames can be of fixed or variable size. In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. An example of this type of framing is the ATM wide-area network, which uses frames of fixed size called cells.

Variable-Size Framing: In variable-size framing, we need a way to define the end of the frame and the beginning of the next. Historically, two approaches were used for this purpose: a character oriented approach and a bit-oriented approach.

Character-Oriented Protocols: In a character-oriented protocol, data to be carried are 8-bit characters from a coding system such as ASCII. The header, which normally carries the source and destination addresses and other control information, and the trailer, which carries error detection or error correction redundant bits, are also multiples of 8 bits.

Bit-Oriented Protocols: In a bit-oriented protocol, the data section of a frame is a sequence of bits to be interpreted by the upper layer as text, graphic, audio, video, and so on. However, in addition to headers (and possible trailers), we still need a delimiter to separate one frame from the other. Most protocols use a special 8-bit pattern flag 01111110 as the delimiter to define the beginning and the end of the frame, as shown in Figure 2.4.

Figure 2.4 A frame in a Bit Oriented Protocol

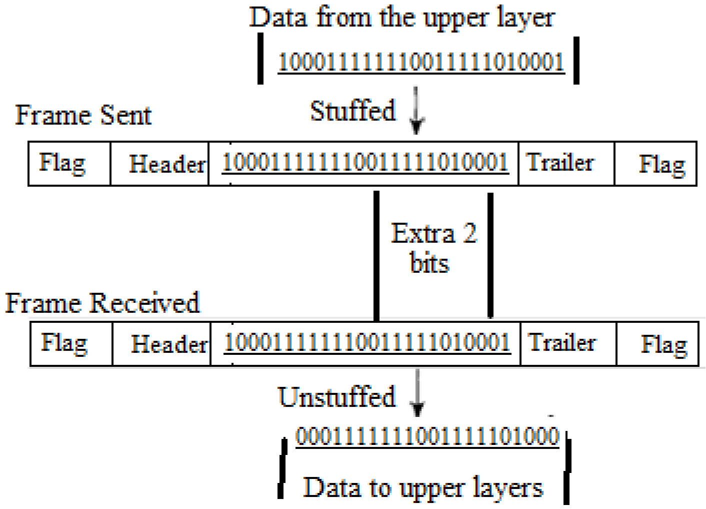


Bit Stuffing: If the flag pattern appears in the data, we need to somehow inform the receiver that this is not the end of the frame. We do this by stuffing 1 single bit (instead of 1 byte) to prevent the pattern from looking like a flag. The strategy is called bit stuffing.

Bit stuffing is the process of adding one extra 0 whenever five consecutive 1's follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag. This guarantees that the flag field sequence does not inadvertently appear in the frame.

Figure 2.5 shows bit stuffing at the sender and bit removal at the receiver. Note that even if we have a 0 after five 1s, we still stuff a 0. The 0 will be removed by the receiver. This means that if the flag like pattern 01111110 appears in the data, it will change to 011111010 (stuffed) and is not mistaken as a flag by the receiver. The real flag 01111110 is not stuffed by the sender and is recognized by the receiver.

Figure 2.5 Bit Stuffing and Unstuffing



FLOW AND ERROR CONTROL

Data communication requires at least two devices working together, one to send and the other to receive. Even such a basic arrangement requires a great deal of coordination for an intelligible exchange to occur. The most important responsibilities of the data link layer are flow control and error control. Collectively, these functions are known as data link control.

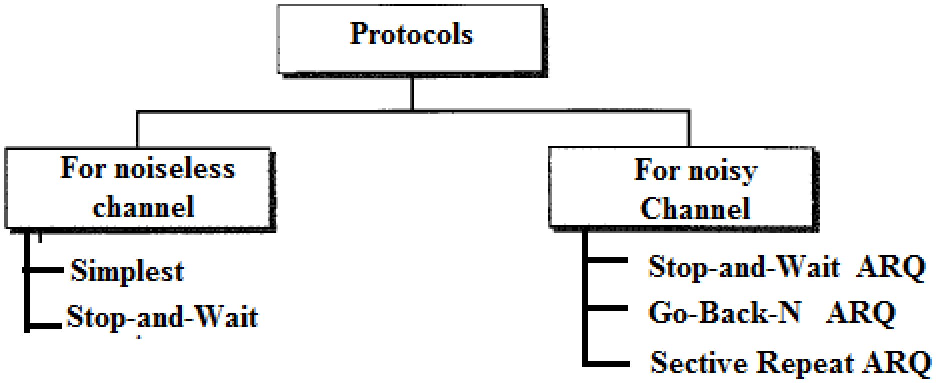
Flow Control: Flow control coordinates the amount of data that can be sent before receiving an acknowledgment and is one of the most important duties of the data link layer. In most protocols, **flow control is a set of procedures that tells the sender how much data it can transmit before it must wait for an acknowledgment from the receiver**. The flow of data must not be allowed to overwhelm the receiver.

Any receiving device has a limited speed at which it can process incoming data and a limited amount of memory in which to store incoming data. The receiving device must be able to inform the sending device before those limits are reached and to request that the transmitting device send fewer frames or stop temporarily. Incoming data must be checked and processed before they can be used.

Error Control: Error control is both error detection and error correction. It allows the receiver to inform the sender of any frames lost or damaged in transmission and coordinates the retransmission of those frames by the sender. In the data link layer, the term *error control* refers primarily to methods of error detection and retransmission. Error control in the data link layer is often implemented simply: Any time an error is detected in an exchange, specified frames are retransmitted. This process is called automatic repeat request (ARQ).

PROTOCOLS FOR FLOW AND ERROR CONTROL

We divide the discussion of protocols into those that can be used for noiseless (error-free) channels and those that can be used for noisy (error-creating) channels. The protocols in the first category cannot be used in real life, but they serve as a basis for understanding the protocols of noisy channels.



Although special frames, called acknowledgment (ACK) and negative acknowledgment (NAK) can flow in the opposite direction for flow and error control purposes, data flow in only one direction. In a real-life network, the data link protocols are implemented as bidirectional; data flow in both directions. In these protocols the flow and error control information such as ACKs and NAKs is included in the data frames in a technique called piggybacking. Because bidirectional protocols are more complex than unidirectional ones, we chose the latter for our discussion. If they are understood, they can be extended to bidirectional protocols. We leave this extension as an exercise.

NOISELESS CHANNELS

Let us first assume we have an ideal channel in which no frames are lost, duplicated, or corrupted. We introduce two protocols for this type of channel. The first is a protocol that does not use flow control; the second is the one that does. Of course, neither has error control because we have assumed that the channel is a perfect noiseless channel.

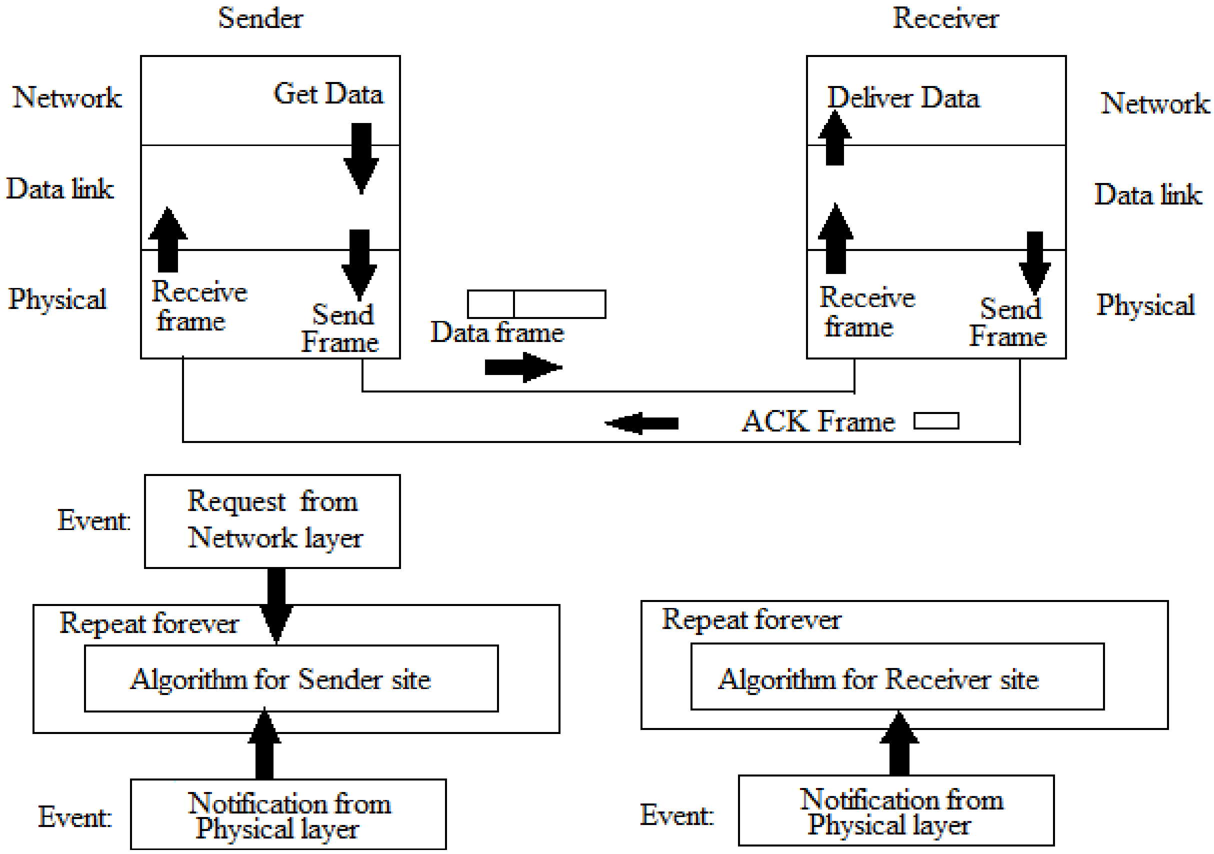
Stop-and-Wait Protocol

If data frames arrive at the receiver site faster than they can be processed, the frames must be stored until their use. Normally, the receiver does not have enough storage space, especially if it is receiving data from many sources. This may result in either the discarding of frames or denial of service. To prevent the receiver from becoming overwhelmed with frames, we somehow need to tell the sender to slow down. There must be feedback from the receiver to the sender. The protocol we discuss now is called the Stop-and-Wait Protocol because the sender sends one frame, stops until it receives confirmation from the receiver (okay to go ahead), and then sends the next frame.

Design

Figure 2.6 illustrates the mechanism. Comparing this figure with Figure 11.6, we can see the traffic on the forward channel (from sender to receiver) and the reverse channel. At any time, there is either one data frame on the forward channel or one ACK frame on the reverse channel. We therefore need a half-duplex link.

Figure 2.6 Design of Stop and Wait Protocol



Analysis Here two events can occur: a request from the network layer or an arrival notification from the physical layer. The responses to these events must alternate. In other words, after a frame is sent, the algorithm must ignore another network layer request until that frame is acknowledged. We know that two arrival events cannot happen one after another because the channel is error-free and does not duplicate the frames. The requests from the network layer, however, may happen one after another without an arrival event in between.

We need somehow to prevent the immediate sending of the data frame. Although there are several methods, we have used a simple *canSend* variable that can either be true or false. When a frame is sent, the variable is set to false to indicate that a new network request cannot be sent until *canSend* is true. When an ACK is received, canSend is set to true to allow the sending of the next frame.

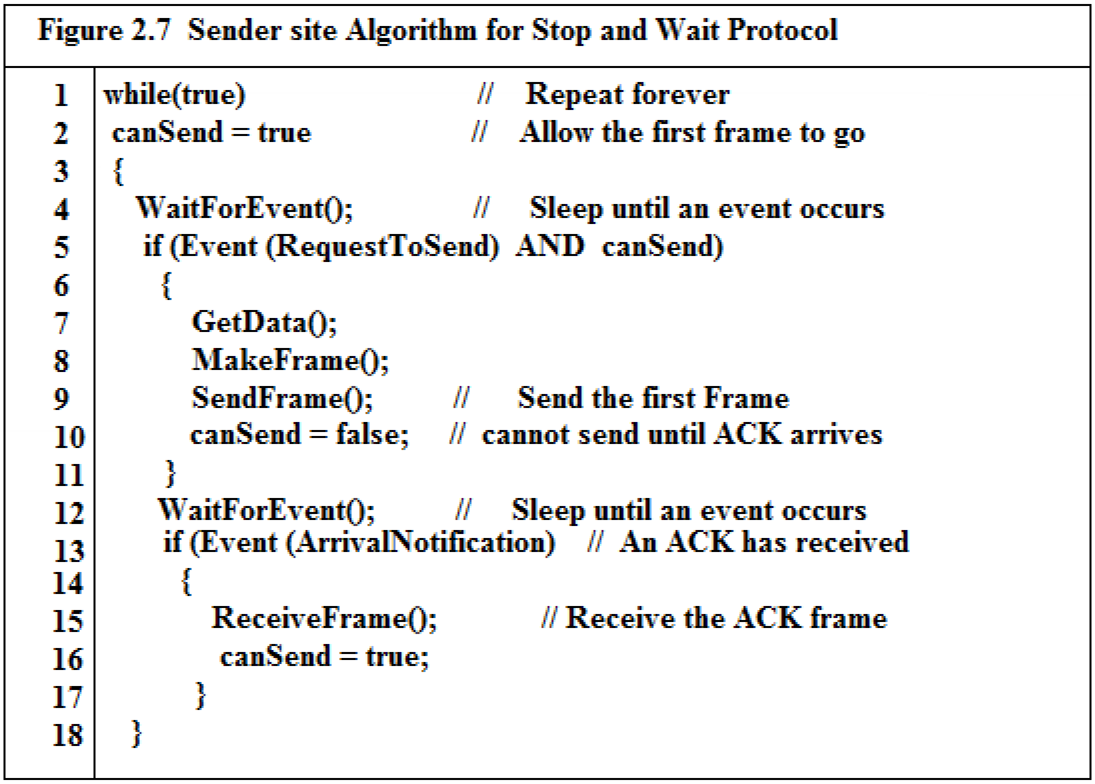


Figure 2.7 Sender site Algorithm for Stop and Wait Protocol

```
while(true)                    //   Repeat forever
 canSend = true                //   Allow the first frame to go
 {
   WaitForEvent();             //   Sleep until an event occurs
    if (Event (RequestToSend)  AND  canSend)
     {
        GetData();
        MakeFrame();
        SendFrame();          //     Send the first Frame
        canSend = false;     //  cannot send until ACK arrives
     }
     WaitForEvent();         //    Sleep until an event occurs
      if (Event (ArrivalNotification)    //  An ACK has received
       {
          ReceiveFrame();          // Receive the ACK frame
           canSend = true;
        }
  }
```

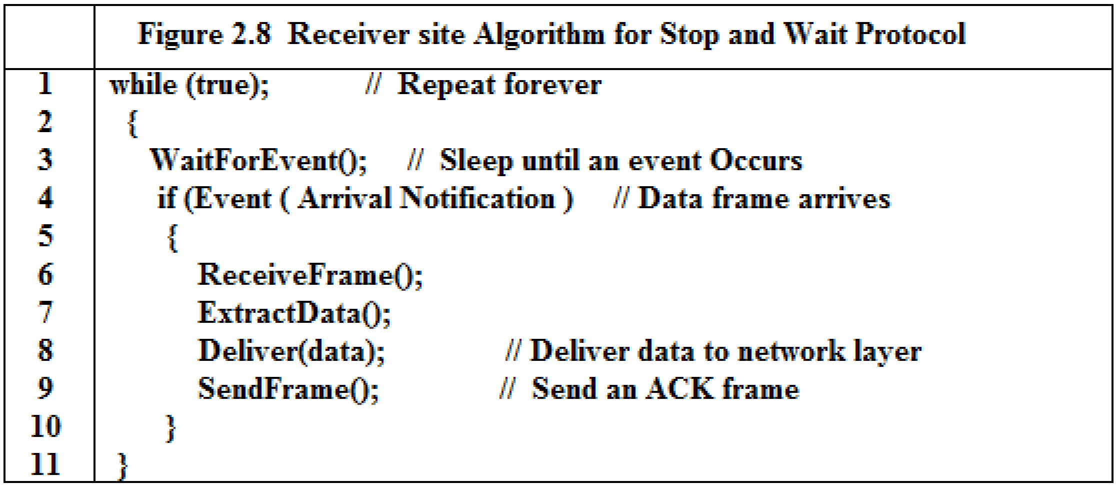


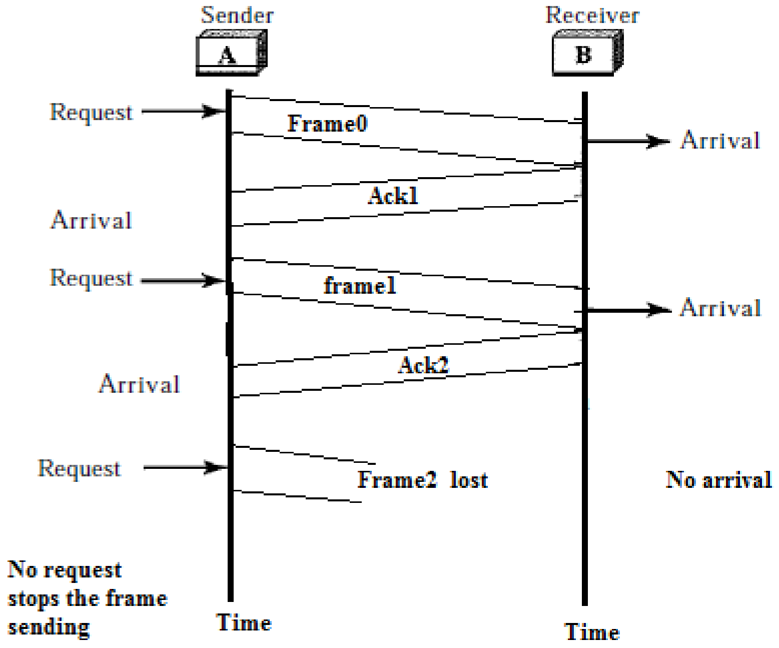
Figure 2.8 Receiver site Algorithm for Stop and Wait Protocol

```
1   while (true);          //  Repeat forever
2    {
3      WaitForEvent();   //  Sleep until an event Occurs
4       if (Event ( Arrival Notification )     // Data frame arrives
5        {
6          ReceiveFrame();
7          ExtractData();
8          Deliver(data);             // Deliver data to network layer
9          SendFrame();              //  Send an ACK frame
10       }
11   }
```

Analysis this is very similar to Algorithm 2.7 with one exception. After the data frame arrives, the receiver sends an ACK frame (line 9) to acknowledge the receipt and allow the sender to send the next frame.

Figure 11.9 shows an example of communication using this protocol. It is still very simple. The sender sends one frame and waits for feedback from the receiver. When the ACK arrives, the sender sends the next frame. Note that sending two frames in the protocol involves the sender in four events and the receiver in two events. When frame2 lost, the transmitter stops the communication. It is because of the sender wait for acknowledgement and stopped the communication.

Figure 2.9 Flow Diagram for Stop-and-Wait Protocol



NOISY CHANNELS

We discuss three protocols in this section that use error control.

Stop-and-Wait ARQ (Automatic Repeat Request)

Stop-and-Wait Automatic Repeat Request (Stop-and Wait ARQ), adds a simple error control mechanism to the Stop-and-Wait Protocol. When the frame arrives at the receiver site, it is checked and if it is corrupted, it is silently discarded. The detection of errors in this protocol is manifested by the silence of the receiver.

In stop and wait protocols, there was no way to identify a frame. The received frame could be the correct one, or a duplicate. The solution is to number the frames. The corrupted and lost frames need to be resent in this protocol. If the receiver does not respond when there is an error, how can the sender know which frame to resend? To remedy this problem, the sender keeps a copy of the sent frame. At the same time, it starts a timer. If the timer expires and there is no ACK for the sent frame, the frame is resent, the copy is held, and the timer is restarted.

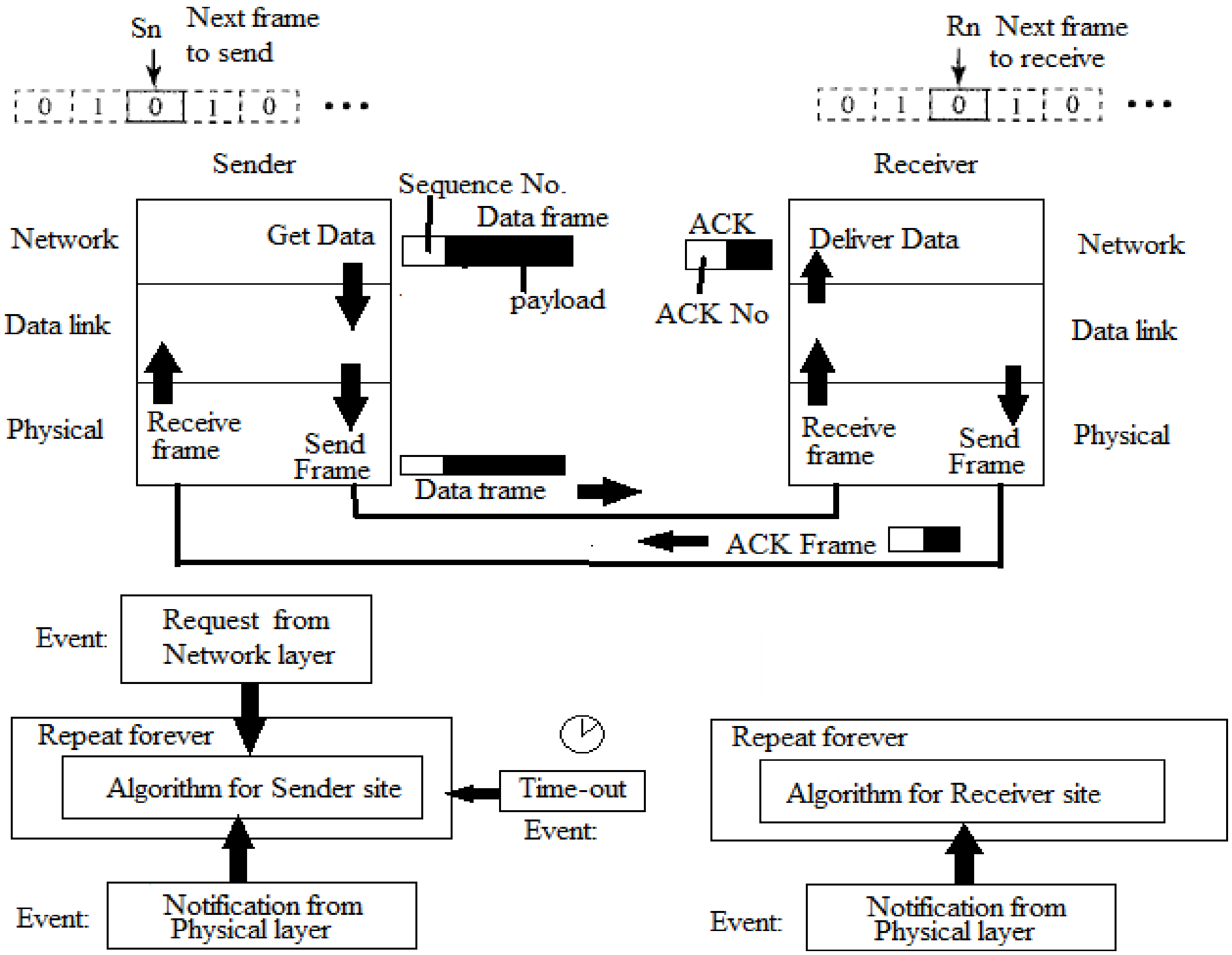
Sequence Numbers: In Stop-and-Wait ARQ we use sequence numbers to number the frames. The sequence numbers are based on modulo-2 arithmetic (2^m – 1). A field is added to the data frame to hold the sequence number of that frame.

Acknowledgment Numbers: In Stop-and-Wait ARQ the acknowledgment number always announces in modulo-2 arithmetic, the sequence number of the next frame expected. The acknowledgment numbers always announce the sequence number of the next frame expected by the receiver.

Design

Figure 2.10 shows the design of the Stop-and-Wait ARQ Protocol. The sending device keeps a copy of the last frame transmitted until it receives an acknowledgment for that frame. A data frames uses a seqNo (sequence number); an ACK frame uses an ackNo (acknowledgment number). The sender has a control variable, which we call S_n (next frame to send), that holds the sequence number for the next frame to be sent (0 or 1).

Figure 2.10 Design of Stop-and-Wait ARQ

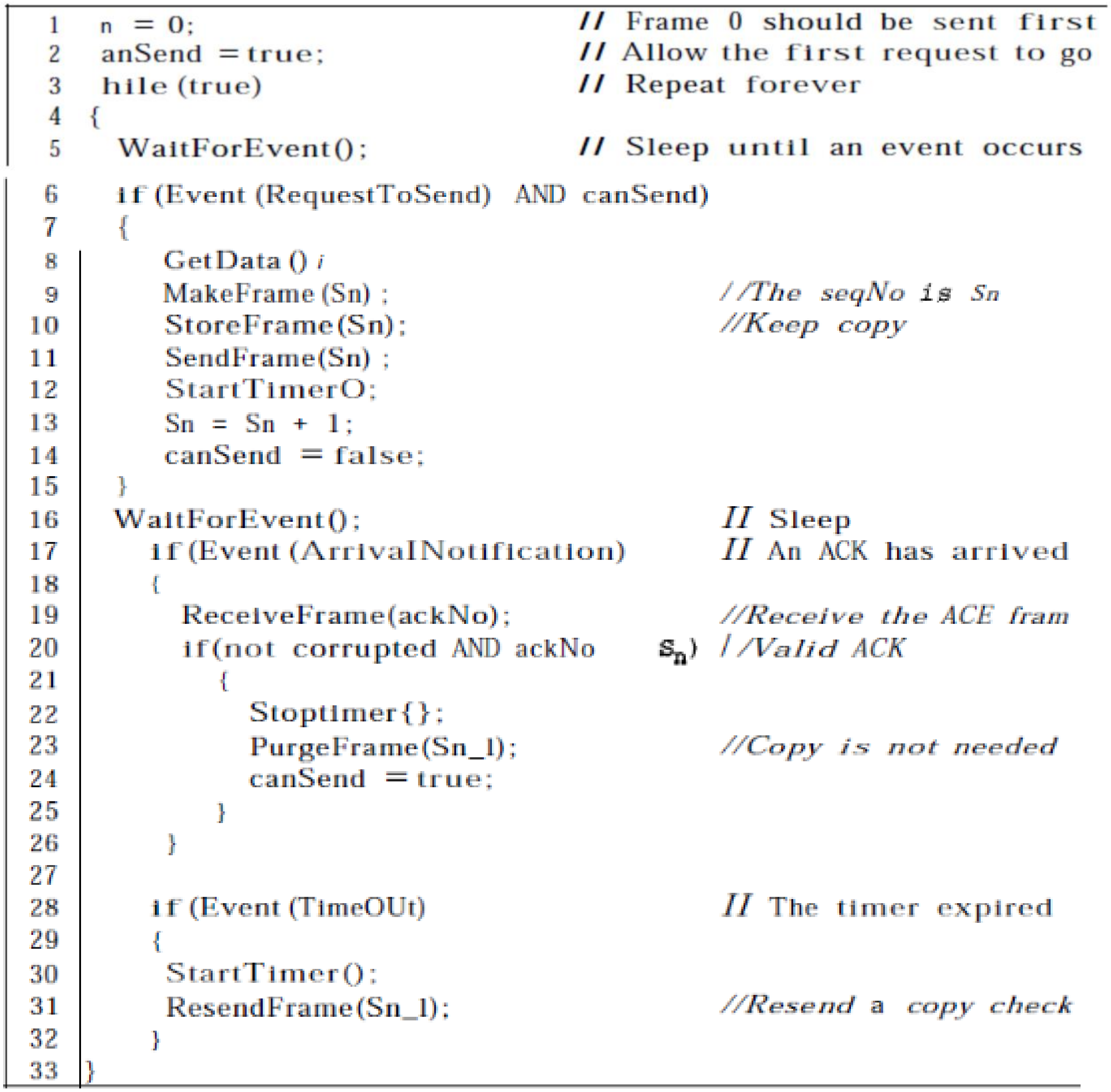


The receiver has a control variable, which we call R_n (next frame expected), that holds the number of the next frame expected. When a frame is sent, the value of S_n is incremented (modulo-2), which means if it is 0, it becomes 1 and vice versa. When a frame is received, the value of R_n is incremented (modulo-2), which means if it is 0, it becomes 1 and vice versa. Three events can happen at the sender site; one event can happen at the receiver site. Variable S_n points to the slot that matches the sequence number of the frame that has been sent, but not acknowledged; R_n points to the slot that matches the sequence number of the expected frame.

Sender-site algorithm for Stop-and- Wait ARQ

We first notice the presence of S_n' the sequence number of the next frame to be sent. This variable is initialized once (line 1), but it is incremented every time a frame is sent (line 13) in preparation for the next frame. However, since this is modulo-2 arithmetic, the sequence numbers are 0, 1,0, 1, and so on. Note that the processes in the first event (SendFrame, StoreFrame, and PurgeFrame) use a S_n defining the frame sent out. We need at least one buffer to hold this frame until we are sure that it is received safe and sound. Line 10 shows that before the frame is sent, it is stored. The copy is used for resending a corrupt or lost frame. We are still using the canSend variable to prevent the network layer from making a request before the previous frame is received safe and sound.

Sender-site algorithm for Stop-and- Wait ARQ



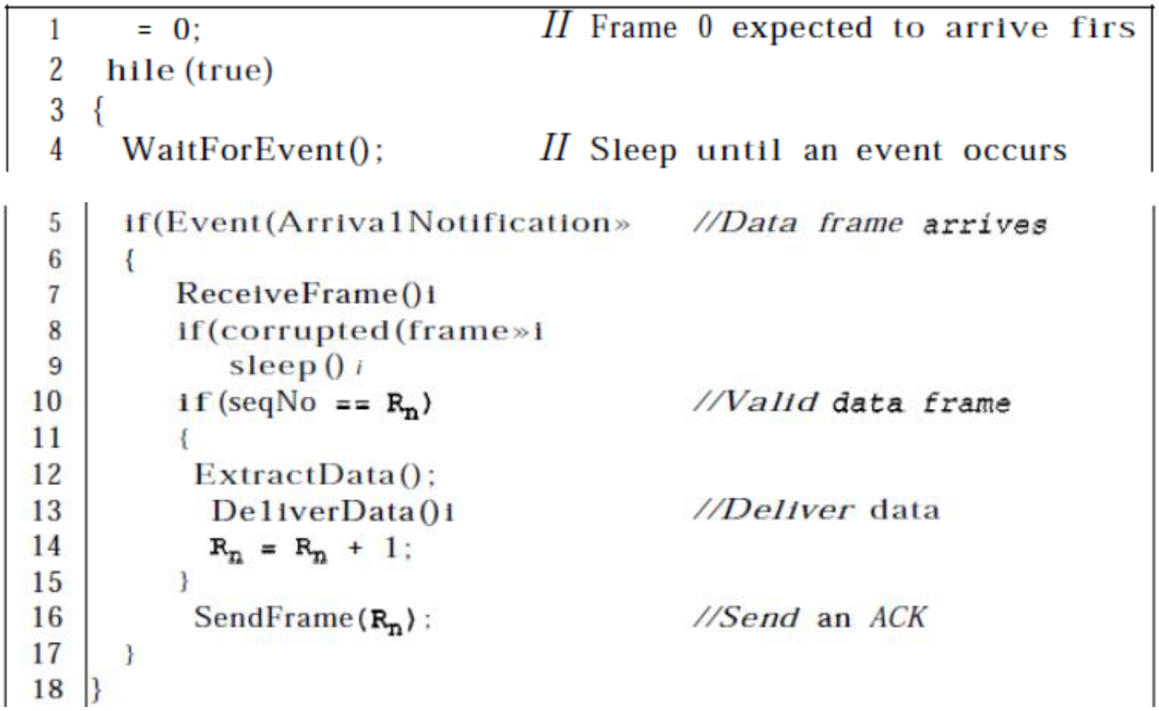
```
 1  n = 0;                               // Frame 0 should be sent first
 2  anSend =true;                        // Allow the first request to go
 3  hile (true)                          // Repeat forever
 4  {
 5    WaitForEvent();                    // Sleep until an event occurs
 6    if (Event (RequestToSend)  AND canSend)
 7    {
 8        GetData () i
 9        MakeFrame (Sn) ;                        / /The seqNo is Sn
10        StoreFrame(Sn);                         //Keep copy
11        SendFrame(Sn) ;
12        StartTimerO;
13        Sn = Sn + 1;
14        canSend = false;
15    }
16   WaitForEvent();                              II Sleep
17      if(Event (ArrivalNotification)            II An ACK has arrived
18      {
19        ReceiveFrame(ackNo);                    //Receive the ACE fram
20        if(not corrupted AND ackNo     Sn)      / /Valid ACK
21           {
22             Stoptimer{};
23             PurgeFrame(Sn_l);                  //Copy is not needed
24             canSend =true;
25           }
26       }
27
28      if (Event (TimeOUt)                       II The timer expired
29      {
30       StartTimer():
31       ResendFrame(Sn_l);                       //Resend a copy check
32      }
33 }
```

If the frame is not corrupted and the ackNo of theACK frame matches the sequence number of the next frame to send, we stop the timer and purge the copy of the data frame we saved. Otherwise, we just ignore this event and wait for the next event to happen. After each frame is sent, a timer is started. When the timer expires (line 28), the frame is resent and the timer is restarted.

Receiver-site algorithm for Stop-and-Wait ARQ Protocol

First, all arrived data frames that are corrupted are ignored. If the seqNo of the frame is the one that is expected *(R_n)* the frame is accepted, the data are delivered to the network layer, and the value of R_n is incremented. However, there is one subtle point here. Even if the sequence number of the data frame does not match the next frame expected, an ACK is sent to the sender. This ACK, however, just reconfirms the previous ACK instead of confirming the frame received. This is done because the receiver assumes that the previous ACK might have been lost; the receiver is sending a duplicate frame. The resent ACK may solve the problem before the time-out does it.

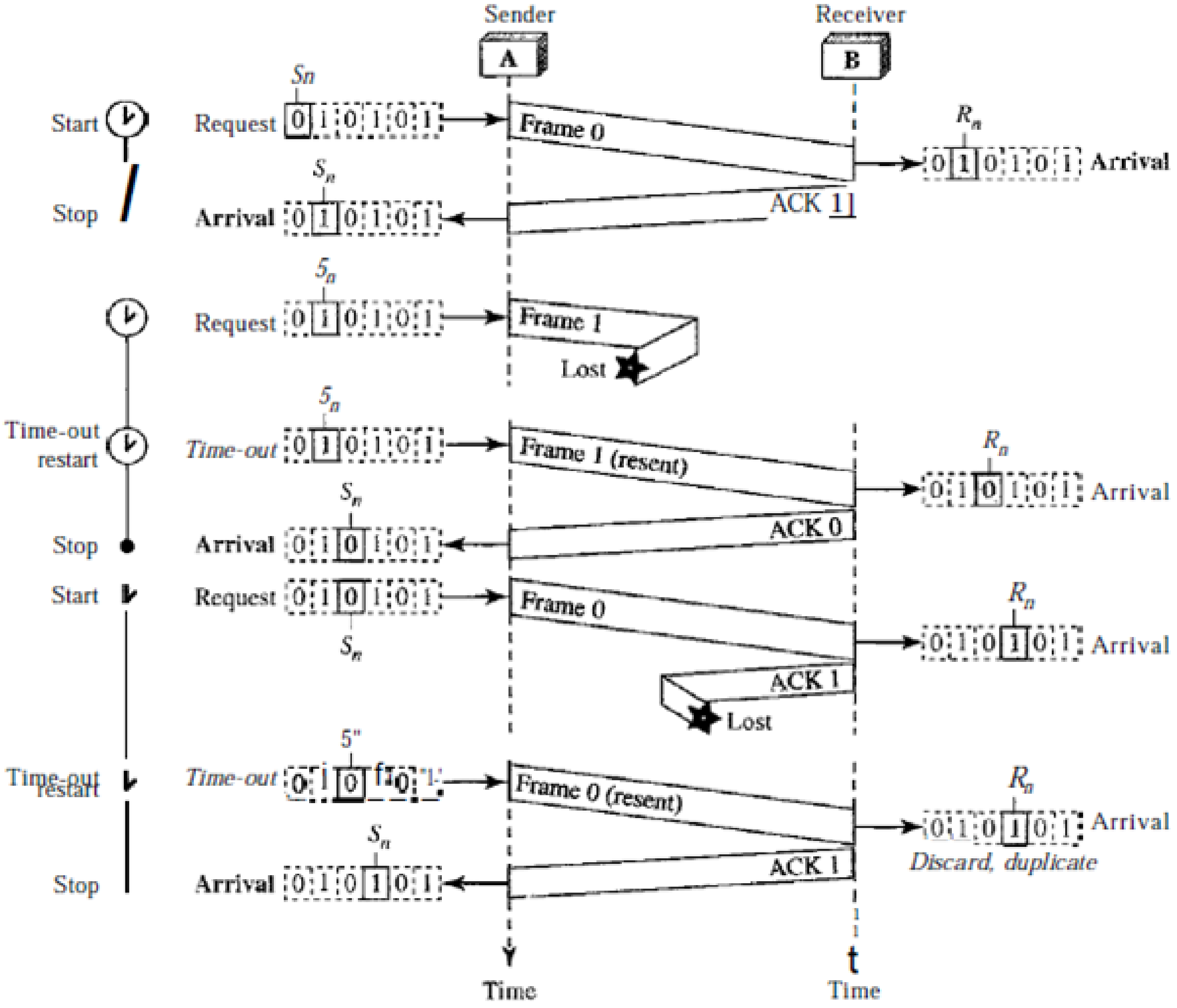
Receiver-site algorithm for Stop-and-Wait ARQ Protocol



```
  = 0;                          II Frame 0 expected to arrive firs
hile (true)
{
  WaitForEvent();               II Sleep until an event occurs
  if(Event(Arriva1Notification»     //Data frame arrives
  {
     ReceiveFrame()i
     if(corrupted(frame»i
        sleep() i
     if(seqNo == $R_n$)                //Valid data frame
     {
      ExtractData();
       De1iverData()i                //Deliver data
       $R_n$ = $R_n$ + 1;
     }
      SendFrame($R_n$);               //Send an ACK
  }
}
```

Figure 2.11 shows an example of Stop-and-Wait ARQ. Frame 0 is sent and acknowledged. Frame 1 is lost and resent after the time-out. The resent frame 1 is acknowledged and the timer stops. Frame 0 is sent and acknowledged, but the acknowledgment is lost. The sender has no idea if the frame or the acknowledgment is lost, so after the time-out, it resends frame 0, which is acknowledged.

Figure 2.11 Data flow for Stop-and-Wait ARQ Protocol



Efficiency

The Stop-and-Wait ARQ is very inefficient if our channel is *thick* and *long*. By *thick,* we mean that our channel has a large bandwidth; by *long,* we mean the round-trip delay is long. The product of these two is called the bandwidth delay product. The system can send 20,000 bits during the time it takes for the data to go from the sender to the receiver and then back again. However, the system sends only 1000 bits. We can say that the link utilization is only 1000/20,000, or 5 percent. For this reason, for a link with a high bandwidth or long delay, the use of Stop-and-Wait ARQ wastes the capacity of the link.

Pipelining

In networking and in other areas, a task is often begun before the previous task has ended. This is known as pipelining. There is no pipelining in Stop-and-Wait ARQ because we need to wait for a frame to reach the destination and be acknowledged before the next frame can be sent. Pipelining improves the efficiency of the transmission if the number of bits in transition is large with respect to the bandwidth-delay product.

Go-Back-N with ARQ (Automatic Repeat Request)

To improve the efficiency of transmission (filling the pipe), multiple frames must be in transition while waiting for acknowledgment. In other words, we need to let more than one frame be outstanding to keep the channel busy while the sender is waiting for acknowledgment. In Go-Back-N Automatic Repeat Request (the rationale for the name will become clear later), we can send several frames before receiving acknowledgments; we keep a copy of these frames until the acknowledgments arrive.

Sequence Numbers

Frames from a sending station are numbered sequentially. However, because we need to include the sequence number of each frame in the header, we need to set a limit. If the header of the frame allows *m* bits for the sequence number, the sequence numbers are modulo-2^m and its range from 0 to 2^m - 1. For example, if *m* is 4, the only sequence numbers are 0 through 15 inclusive. 0, 1,2,3,4,5,6, 7,8,9, 10, 11, 12, 13, 14, 15,0, 1,2,3,4,5,6,7,8,9,10, 11, ...

Sliding Window

In this protocol (and the next), the sliding window is an abstract concept that defines the range of sequence numbers that is the concern of the sender and receiver. In other words, the sender and receiver need to deal with only part of the possible sequence numbers. The range which is the concern of the sender is called the send sliding window; the range that is the concern of the receiver is called the receive sliding window.

The send window is an imaginary box covering the sequence numbers of the data frames which can be in transit. The send window is an abstract concept defining an imaginary box of size 2^m - 1 with three variables: S_f S_n and $S_{size'}$. The send window can slide one or more slots when a valid acknowledgment arrives.

The receive window is an abstract concept defining an imaginary box of size 1 with one single variable R_n. The window slides when a correct frame has arrived; sliding occurs one slot at a time.

Timers

Although there can be a timer for each frame that is sent, in our protocol we use only one. The reason is that the timer for the first outstanding frame always expires first; we send all outstanding frames when this timer expires.

Acknowledgment

The receiver sends a positive acknowledgment if a frame has arrived safe and sound and in order. If a frame is damaged or is received out of order, the receiver is silent and will discard all subsequent frames until it receives the one it is expecting. The silence of the receiver causes the timer of the unacknowledged frame at the sender site to expire. This, in turn, causes the sender to go back and resend all frames, beginning with the one with the expired timer. The receiver does not have to acknowledge each frame received. It can send one cumulative acknowledgment for several frames.

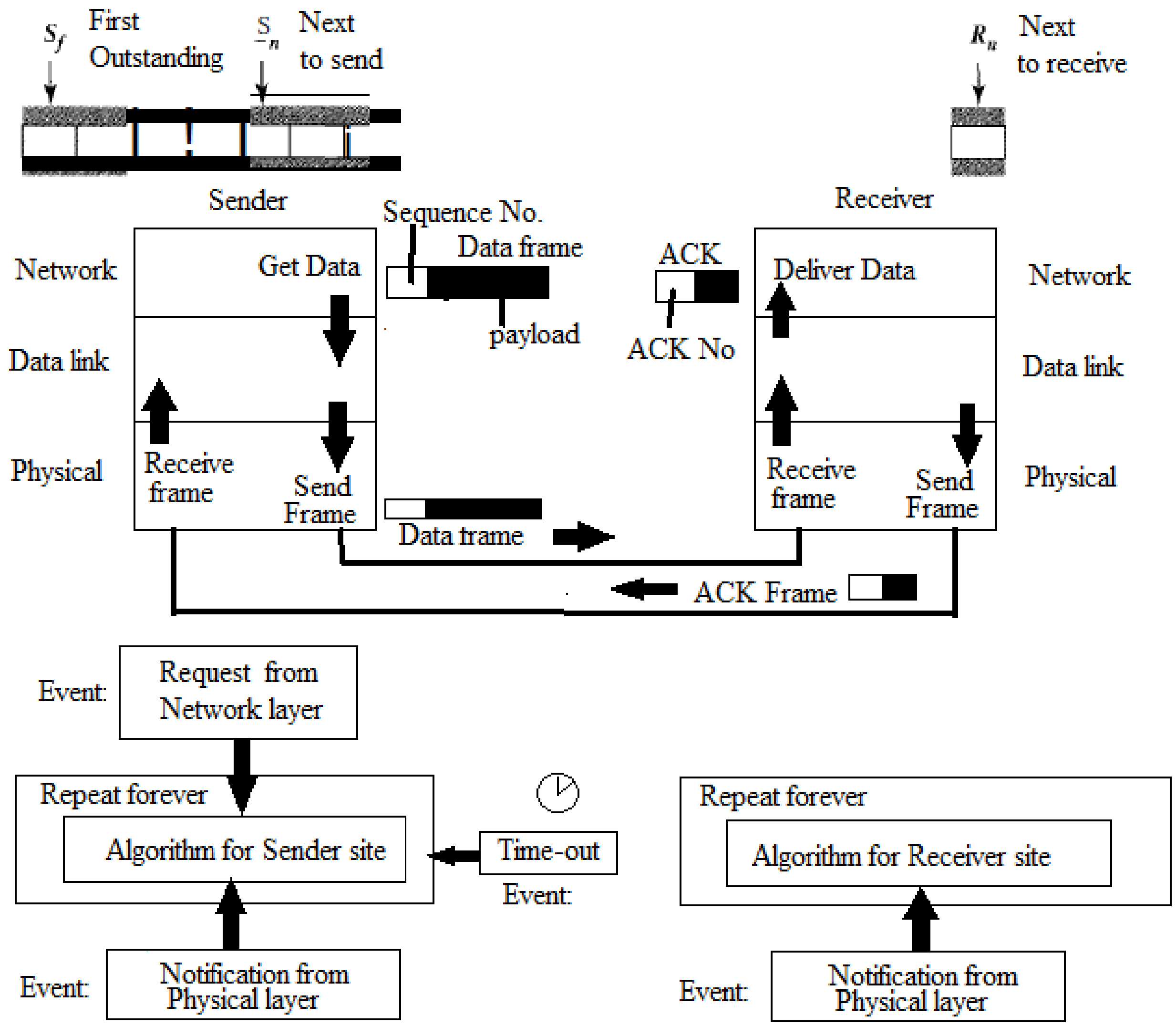
Resending a Frame

When the timer expires, the sender resends all outstanding frames. For example, suppose the sender has already sent frame 6, but the timer for frame 3 expires. This means that frame 3 has not been acknowledged; the sender goes back and sends frames 3, 4,5, and 6 again. That is why the protocol is called *Go-Back-N* ARQ.

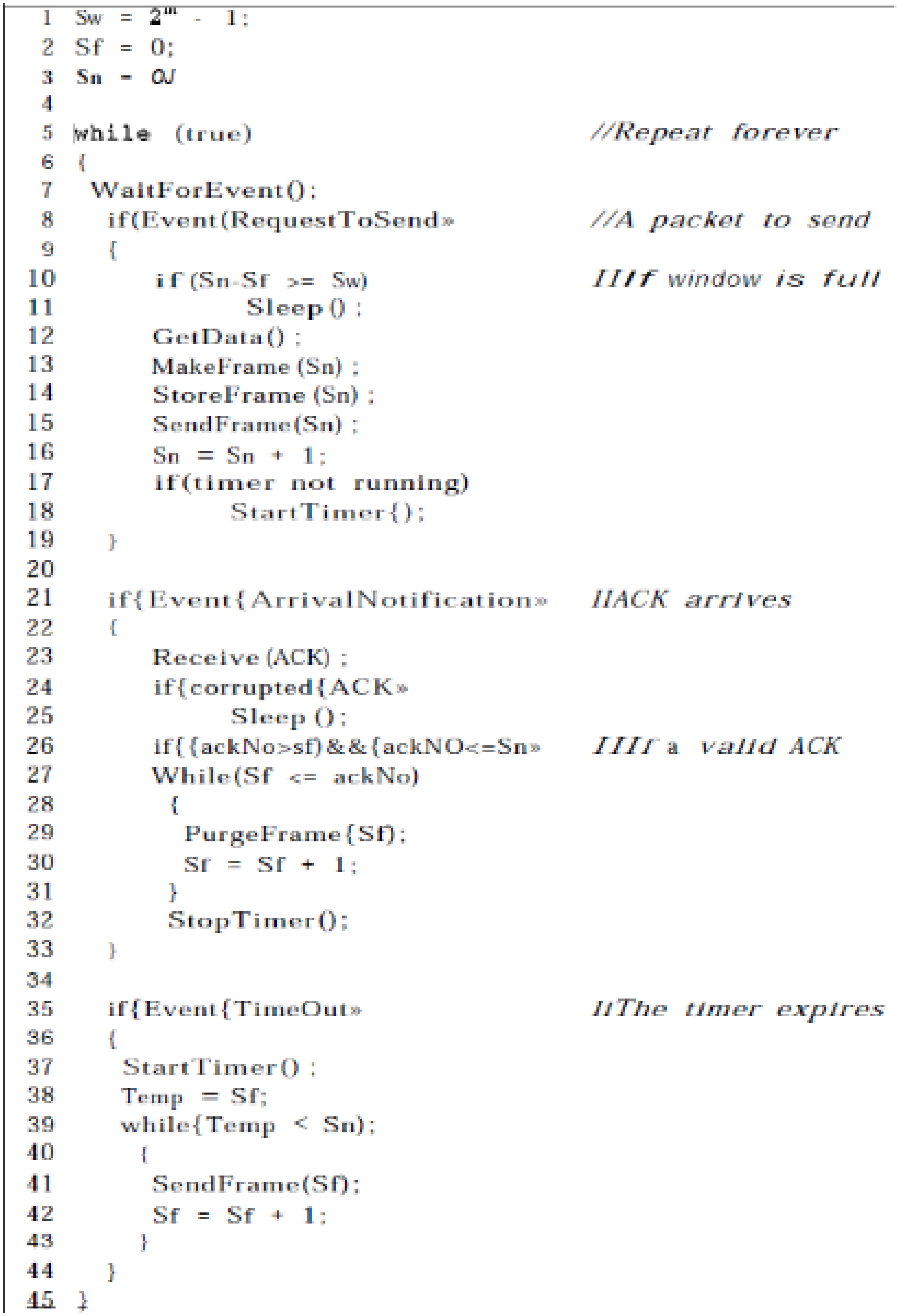
Design

Figure 2.12 shows the design for this protocol. As we can see, multiple frames can be in transit in the forward direction, and multiple acknowledgments in the reverse direction. The idea is similar to Stop-and-Wait ARQ; the difference is that the send window allows us to have as many frames in transition as there are slots in the send window.

Figure 2.12 Design of Go -Back -N ARQ



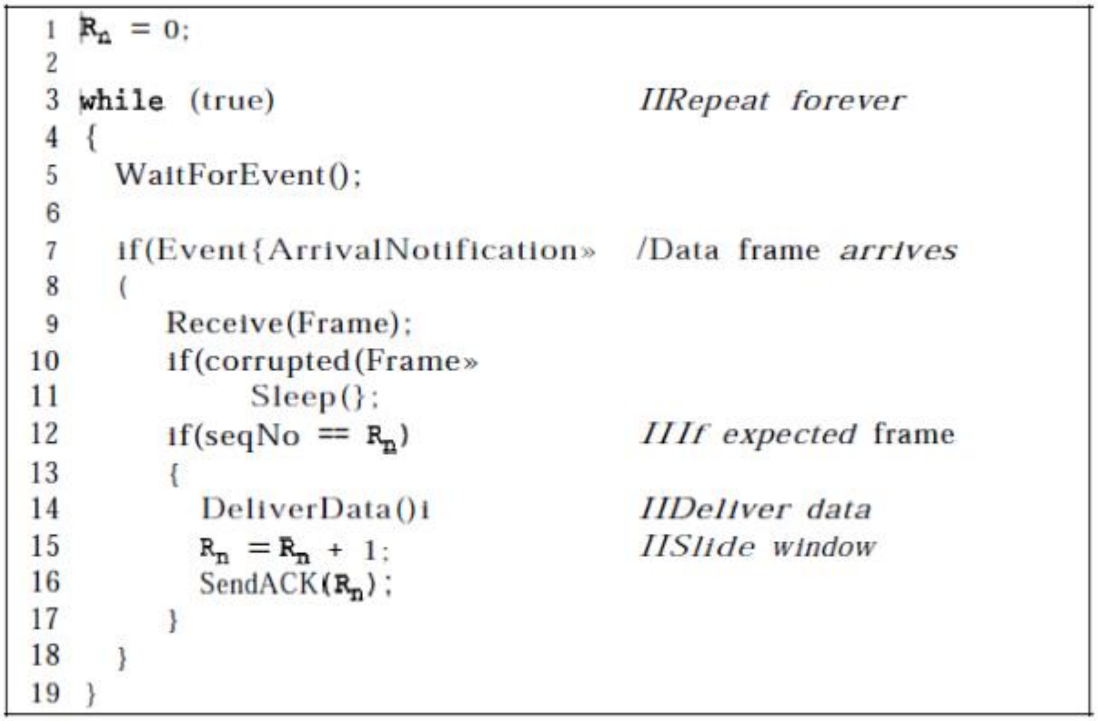
Algorithm 11.7 *Go-Back-N sender algorithm*



```
Sw = 2m - 1;
Sf = 0;
Sn - OJ

while (true)                          //Repeat forever
{
 WaitForEvent();
  if(Event(RequestToSend»             //A packet to send
  {
     if (Sn-Sf >= Sw)                 IIIf window is full
           Sleep () ;
     GetData() ;
     MakeFrame (Sn) ;
     StoreFrame (Sn) ;
     SendFrame(Sn) ;
     Sn = Sn + 1;
     if(timer not running)
          StartTimer{);
  }

  if{Event{ArrivalNotification»       IIACK arrives
  {
     Receive (ACK) ;
     if{corrupted{ACK»
          Sleep ();
     if{{ackNo>sf)&&{ackNO<=Sn»       IIIr a valid ACK
     While(Sf <= ackNo)
      {
       PurgeFrame{Sf);
       Sf = Sf + 1;
      }
      StopTimer();
  }

  if{Event{TimeOut»                   IIThe timer expires
  {
   StartTimer() ;
   Temp = Sf;
   while{Temp < Sn);
    {
     SendFrame(Sf);
     Sf = Sf + 1;
    }
  }
}
```

Analysis: This algorithm first initializes three variables. Unlike Stop-and-Wait ARQ, this protocol allows several requests from the network layer without the need for other events to occur; we just need to be sure that the window is not full (line 12). In our approach, if the window is full, the request is just ignored and the network layer needs to try again. Some implementations use other methods such as enabling or disabling the network layer. The handling of the arrival event is more complex than in the previous protocol. If we receive a corrupted ACK, we ignore it. If the ackNo belongs to one of the outstanding frames, we use a loop to purge the buffers and move the left wall to the right. The time-out event is also more complex. We first start a new timer. We then resend all outstanding frames.

Receiver Site Algorithm for Go-Back-N ARQ: This algorithm is simple. We ignore a corrupt or out-of-order frame. If a frame arrives with an expected sequence number, we deliver the data, update the value of R_n, and send an ACK with the ackNo showing the next frame expected

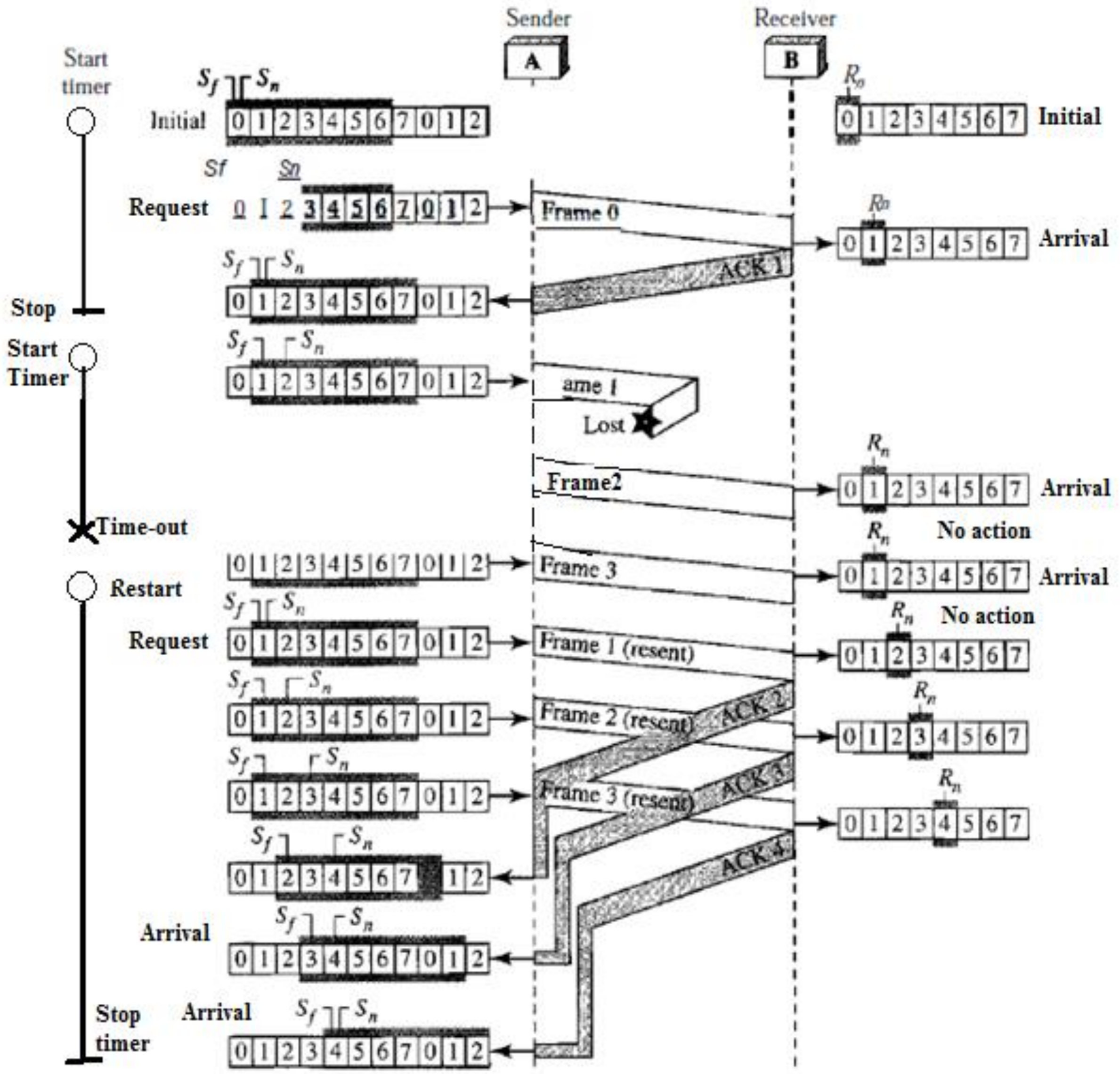


```
R_n = 0;

while (true)                          IIRepeat forever
{
  WaitForEvent();

  if(Event{ArrivalNotification»       /Data frame arrives
  (
     Receive(Frame);
     if(corrupted(Frame»
          Sleep(};
     if(seqNo == R_n)                 IIIf expected frame
     {
       DeliverData()i                 IIDeliver data
       R_n = R_n + 1;                 IISlide window
       SendACK(R_n);
     }
  }
}
```

Data flow of Go-Back-N ARQ with example

Figure 2.13 shows what happens when a frame is lost. Frames 0, 1, 2, and 3 are sent. However, frame 1 is lost. The receiver receives frames 2 and 3, but they are discarded because they are received out of order (frame 1 is expected). The sender receives no acknowledgment about frames 1, 2, or 3. Its timer finally expires. The sender sends all outstanding frames (1, 2, and 3) because it does not know what is wrong. Note that the resending of frames l, 2, and 3 is the response to one single event. When the sender is responding to this event, it cannot accept the triggering of other events. This means that when ACK 2 arrives, the sender is still busy with sending frame 3.

Figure 2.13 Data flow of Go-Back-N ARQ Protocol



The physica1layer must wait until this event is completed and the data link layer goes back to its sleeping state. We have shown a vertical line to indicate the delay. It is the same story with ACK 3; but when ACK 3 arrives, the sender is busy responding to ACK 2. It happens again when ACK 4 arrives. Note that before the second timer expires, all outstanding frames have been sent and the timer is stopped.

Go-Back-N ARQ versus Stop-and- Wait ARQ

The reader may find that there is a similarity between *Go-Back-N*ARQ and Stop-and-Wait ARQ. We can say that the Stop-and-WaitARQ Protocol is actually a *Go-Back-N*ARQ in which there are only two sequence numbers and the send window size is 1. Stop-and-Wait ARQ is a special case of Go-Back-NARQ in which the size of the send window is 1.

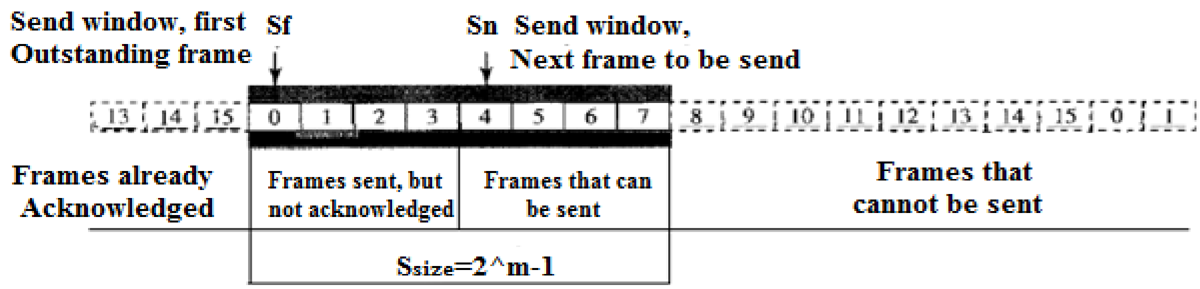
Selective Repeat Automatic Repeat Request:

Go-Back-N ARQ simplifies the process at the receiver site. However, this protocol is very inefficient for a noisy link. In a noisy link a frame has a higher probability of damage, which means the resending of multiple frames. This resending uses up the bandwidth and slows down the transmission. For noisy links, there is a more efficient mechanism that does not resend *N* frames when just one frame is damaged; only the damaged frame is resent. This mechanism is called Selective Repeat ARQ. But the processing at the receiver is more complex.

Windows

The Selective Repeat Protocol also uses two windows: a send window and a receive window. However, there are differences between the windows in this protocol and the ones in Go-Back-N. First, the size of the send window is much smaller; it is $2^m - 1$. Second, the receive window is the same size as the send window. The send window maximum size can be $2^m - 1$.

Figure 2.14 Send Window for Selective Repeat ARQ



The Selective Repeat Protocol allows as many frames as the size of the receive window to arrive out of order and be kept until there is a set of in-order frames to be delivered to the network layer. Because the sizes of the send window and receive window are the same, all the frames in the send frame can arrive out of order and be stored until they can be delivered.

Figure 2.15 Receive Window for selective Repeat ARQ

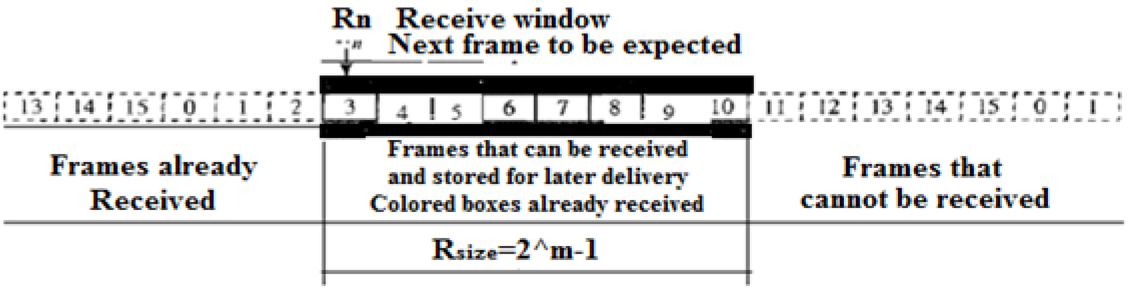
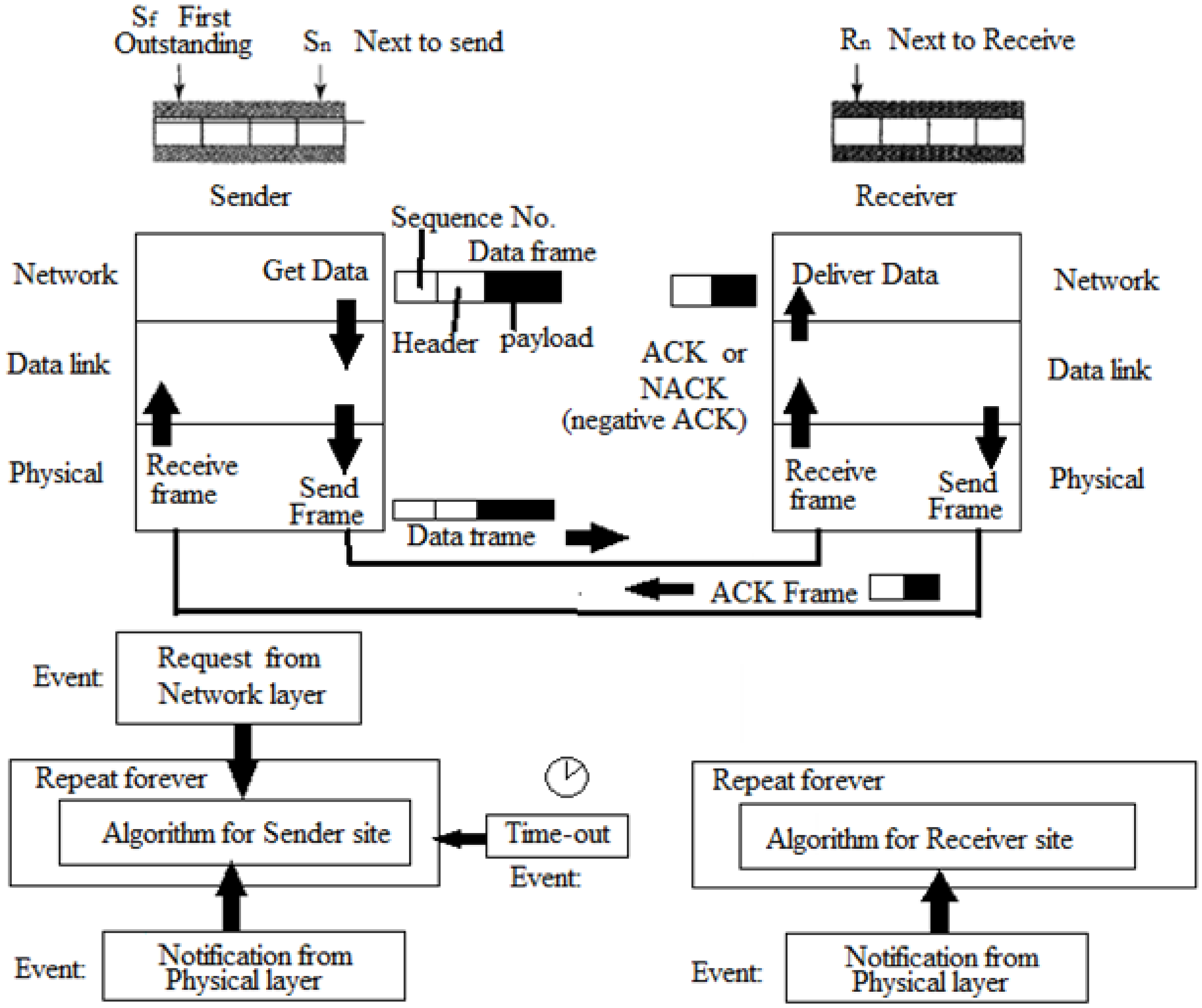

Rn Receive window
Next frame to be expected
13
14
15
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
0
1
Frames already
Received
Frames that can be received
and stored for later delivery
Colored boxes already received
Frames that
cannot be received
Rsize=2^m-1

Figure 2.15 shows the receive window in this protocol. Those slots inside the window that are colored define frames that have arrived out of order and are waiting for their neighbors to arrive before delivery to the network layer.

Design

The design in this case is to some extent similar to the one we described for the Go-Back-N, but more complicated, as shown in Figure 2.16.

Figure 2.16 Design of Selective Repeat Protocol ARQ

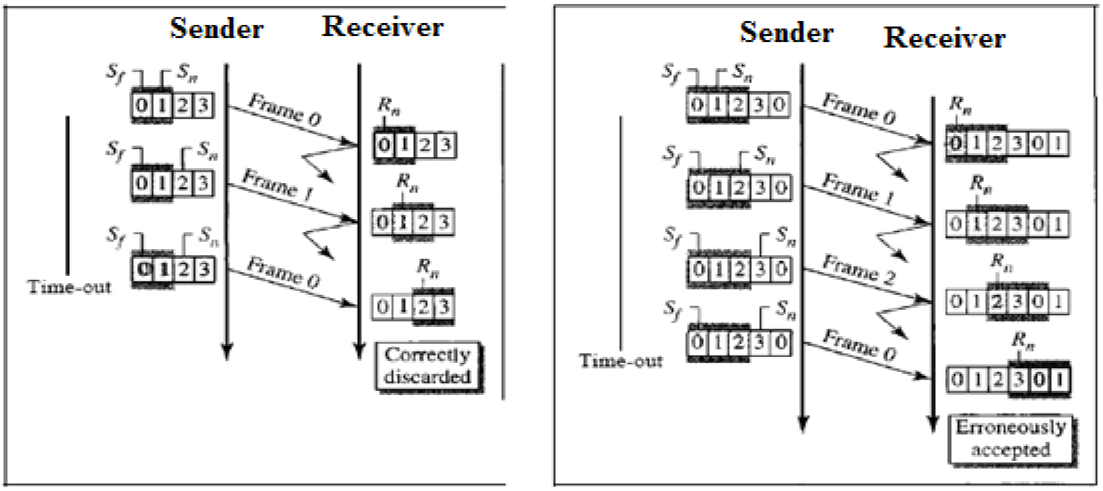


Window Sizes

We can now show why the size of the sender and receiver windows must be at most one half of *2m*. For an example, we choose *m* = 2, which means the size of the window is *2m/2*, or 2. Figure 11.21 compares a window size of 2 with a window size of 3. If the size of the window is 2 and all acknowledgments are lost, the timer for frame 0 expires and frame 0 is resent. However, the window of the receiver is now expecting frame 2, not frame 0, so this duplicate frame is correctly discarded. When the size of the window is 3 and all acknowledgments are lost, the sender sends a duplicate of frame 0.

However, this time, the window of the receiver expects to receive frame 0 (0 is part of the window), so it accepts frame 0, not as a duplicate, but as the first frame in the next cycle. This is clearly an error. **In Selective Repeat ARQ, the size of the sender and receiver window must be at most one-half of *2m*.**

Figure 2.17 Selective Repeat ARQ window size



a) Window size = 2^m-1 b) Window size = greater than 2^m-1

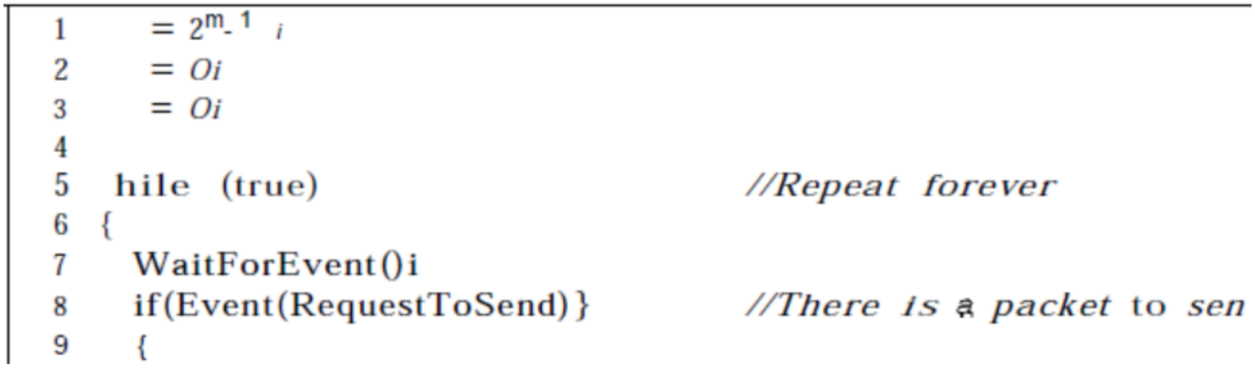
Algorithms

Algorithm 11.9 shows the procedure for the sender.

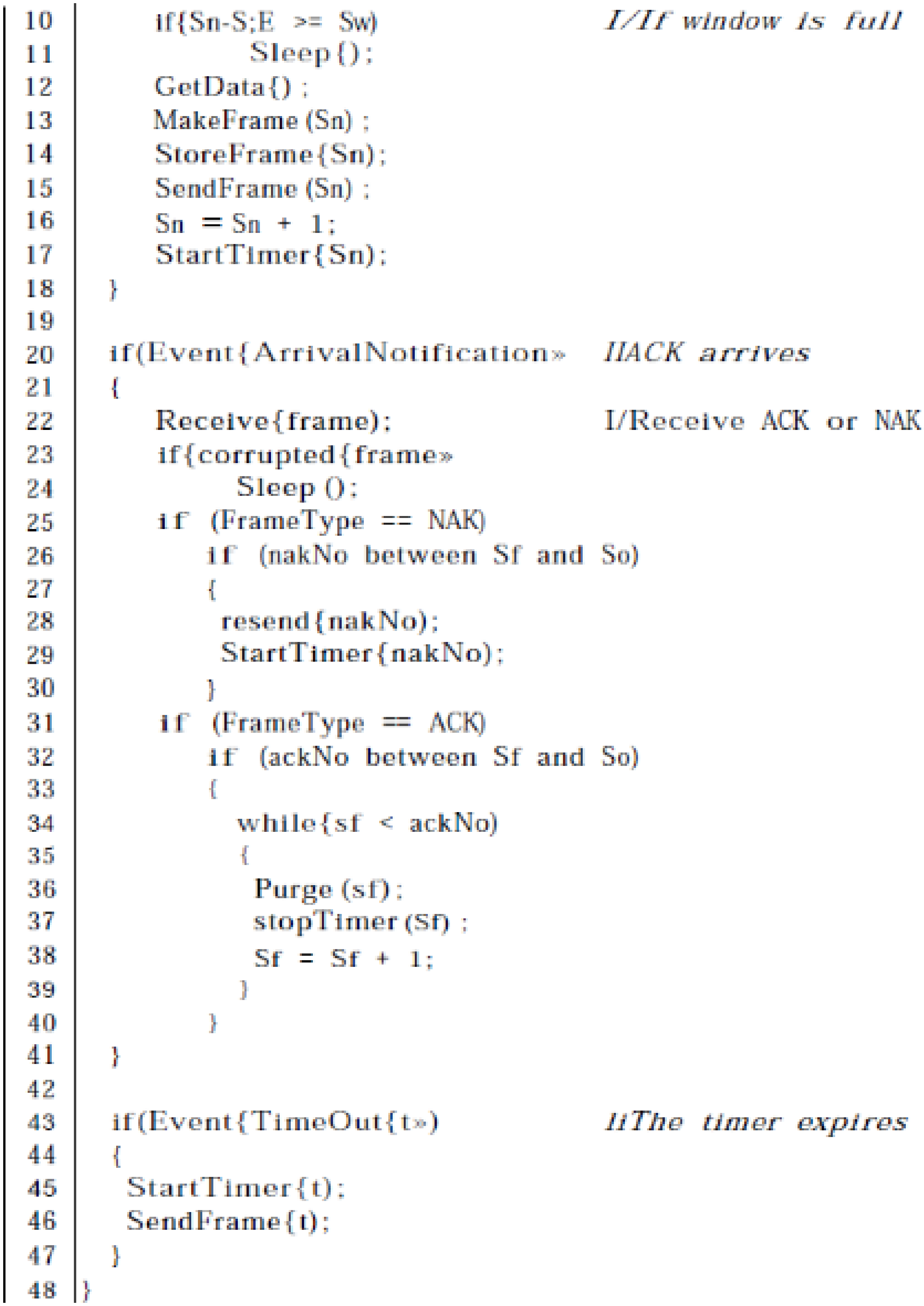
Analysis of Sender site Selective Repeat ARQ:

The handling of the request event is similar to that of the previous protocol except that one timer is started for each frame sent. The arrival event is more complicated here. An ACK or a NAK frame may arrive. If a valid NAK frame arrives, we just resend the corresponding frame. If a valid ACK arrives, we use a loop to purge the buffers, stop the corresponding timer and move the left wall of the window. The time-out event is simpler here; only the frame which times out is resent.

Algorithm for Sender site Selective Repeat ARQ



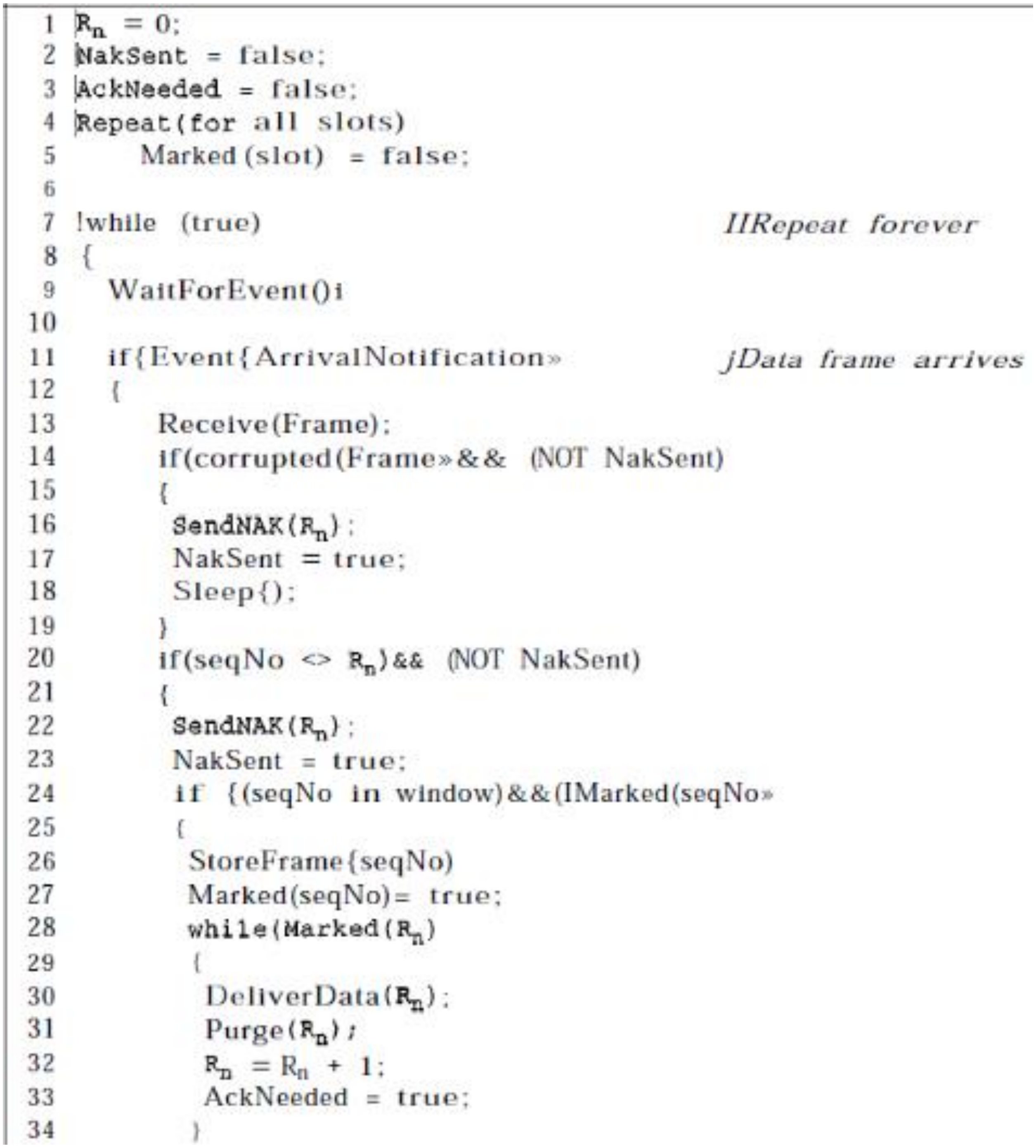
```
      = 2m- 1   i
      = Oi
      = Oi

 hile  (true)                          //Repeat  forever
{
  WaitForEvent()i
  if(Event(RequestToSend)}             //There  is a  packet  to  sen
  {
```



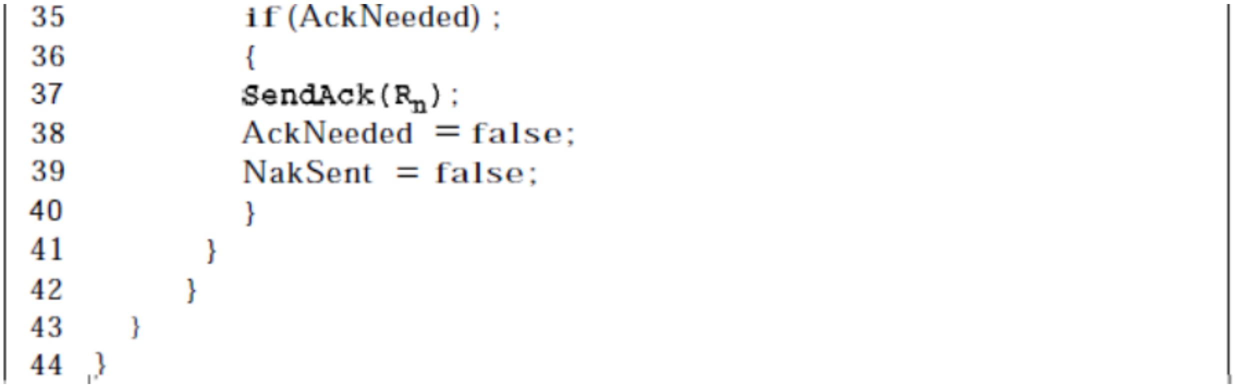
```
    if{Sn-S;E  >=  Sw)                    I/If window is full
          Sleep{};
    GetData{} ;
    MakeFrame (Sn) ;
    StoreFrame{Sn);
    SendFrame (Sn) ;
    Sn  = Sn  +  1;
    StartTimer{Sn);
 }

 if(Event{ArrivalNotification»   IIACK arrives
 {
    Receive{frame);                       I/Receive ACK or NAK
    if{corrupted{frame»
         Sleep ();
    if  (FrameType  ==  NAK)
       if  (nakNo  between  Sf  and  So)
       {
        resend{nakNo);
        StartTimer{nakNo);
       }
    if  (FrameType  ==  ACK)
       if  (ackNo  between  Sf  and  So)
       {
         while{sf  <  ackNo)
         {
          Purge (sf);
          stopTimer (Sf) ;
          Sf  =  Sf  +  1;
         }
       }
 }

 if(Event{TimeOut{t»)                     liThe timer expires
 {
  StartTimer{t);
  SendFrame{t);
 }
}
```

Analysis of Receiver site Selective Repeat ARQ

Here we need more initialization. In order not to overwhelm the other side with NAKs, we use a variable called Nak Sent. To know when we need to send an ACK, we use a variable called Ack needed. Both of these are initialized to false. We also use a set of variables to mark the slots in the receive window once the corresponding frame has arrived and is stored. If we receive a corrupted frame and a NAK has not yet been sent, we send a NAK to tell the other site that we have not received the frame we expected.



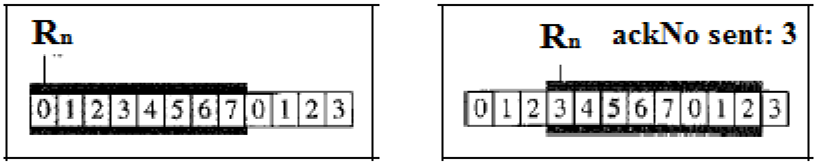
```
$R_n$ = 0;
NakSent = false;
AckNeeded = false;
Repeat(for all slots)
     Marked(slot) = false;

!while (true)                                //Repeat forever
{
  WaitForEvent()i

  if{Event{ArrivalNotification»              jData frame arrives
  {
      Receive(Frame);
      if(corrupted(Frame»&& (NOT NakSent)
      {
       SendNAK($R_n$);
       NakSent = true;
       Sleep{);
      }
      if(seqNo <> $R_n$)&& (NOT NakSent)
      {
       SendNAK($R_n$);
       NakSent = true;
       if {(seqNo in window)&&(IMarked(seqNo»
       {
        StoreFrame{seqNo)
        Marked(seqNo)= true;
        while(Marked($R_n$)
        {
         DeliverData($R_n$);
         Purge($R_n$);
         $R_n$ = $R_n$ + 1;
         AckNeeded = true;
        }
```



```
            if(AckNeeded);
            {
            SendAck(Rn);
            AckNeeded = false;
            NakSent = false;
            }
         }
      }
   }
}
```

If the frame is not corrupted and the sequence number is in the window, we store the frame and mark the slot. If contiguous frames, starting from R_n have been marked, we deliver their data to the network layer and slide the window.

Figure 2.18 Delivery of Data in Selective Repeat ARQ



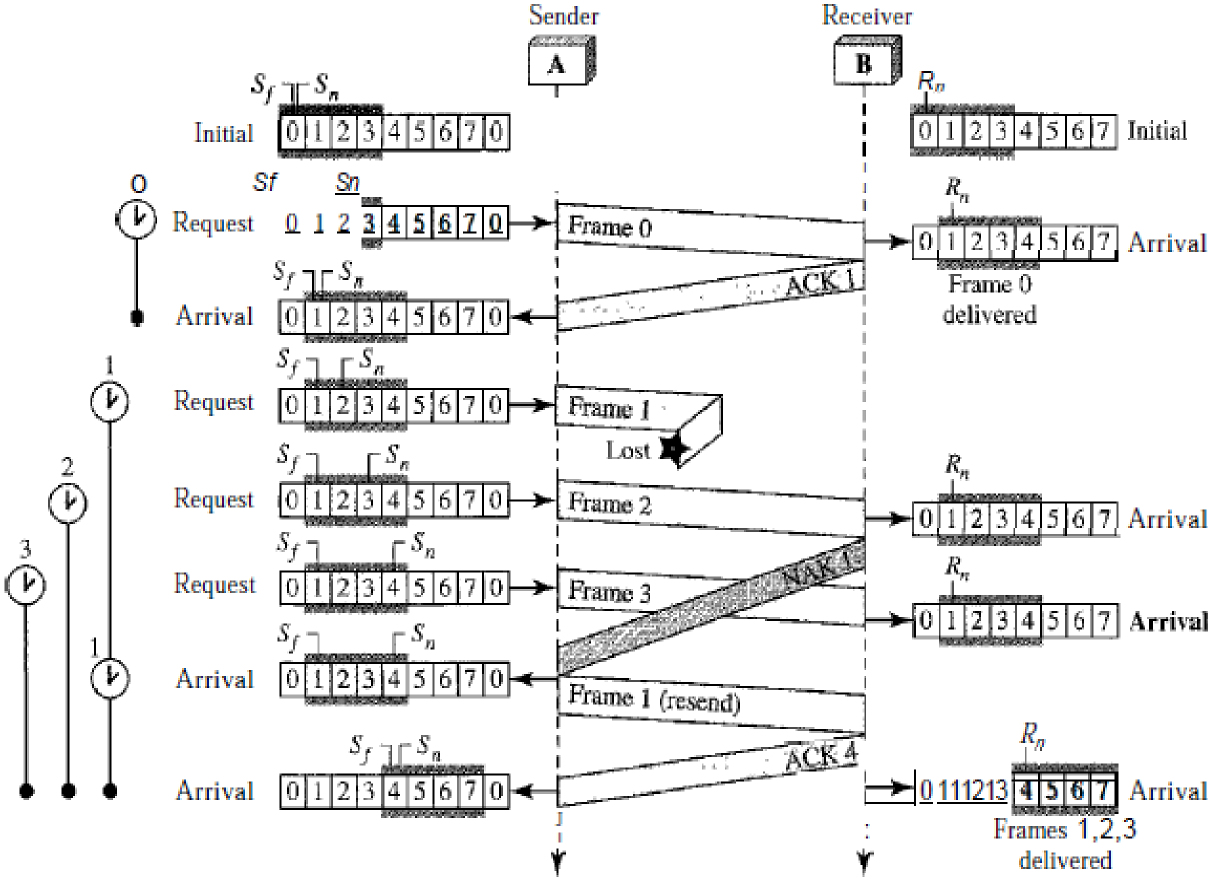
a)Before delivery **b)After delivery**

Data flow of Selective Repeat ARQ with an example

This example is similar to Example 2.19 in which frame 1 is lost. We show how Selective Repeat behaves in this case. Figure 2.19 shows the situation. One main difference is the number of timers. Here, each frame sent or resent needs a timer, which means that the timers need to be numbered (0, 1, 2, and 3). The timer for frame °starts at the first request, but stops when the ACK for this frame arrives. The timer for frame 1 starts at the second request restarts when a NAK arrives, and finally stops when the last ACK arrives. The other two timers start when the corresponding frames are sent and stop at the last arrival event.

At the receiver site we need to distinguish between the acceptance of a frame and its delivery to the network layer. At the second arrival, frame 2 arrives and is stored and marked (colored slot), but it cannot be delivered because frame I is missing. At the next arrival, frame 3 arrives and is marked and stored, but still none of the frames can be delivered. Only at the last arrival, when finally a copy of frame 1 arrives, can frames 1, 2, and 3 be delivered to the network layer. There are two conditions for the delivery of frames to the network layer: First, a set of consecutive frames must have arrived. Second, the set starts from the beginning of the window. After the first arrival, there was only one frame and it started from the beginning of the window. After the last arrival, there are three frames and the first one starts from the beginning of the window.

Figure 2.19 Data flow of Selective Repeat ARQ Protocol



Another important point is that a NAK is sent after the second arrival, but not after the third, although both situations look the same. The reason is that the protocol does not want to crowd the network with unnecessary NAKs and unnecessary resent frames. The second NAK would still be NAKI to inform the sender to resend frame 1 again; this has already been done. The first NAK sent is remembered (using the nakSent variable) and is not sent again until the frame slides. A NAK is sent once for each window position and defines the first slot in the window.

The next point is about the ACKs. Notice that only two ACKs are sent here. The first one Repeat, ACKs are sent when data are delivered to the network layer. If the data belonging to *n* frames are delivered in one shot, only one ACK is sent for all of them.

Piggybacking

The data link control protocols are all unidirectional: data frames flow in only one direction although control information such as ACK and NAK frames can travel in the other direction. In real life, data frames are normally flowing in both directions: from node A to node B and from node B to node A. This means that the control information also needs to flow in both directions.

A technique called **piggybacking** is used to improve the efficiency of the bidirectional protocols. When a frame is carrying data from A to B, it can also carry control information about arrived (or lost) frames from B; when a frame is carrying data from B to A, it can also carry control information about the arrived (or lost) frames from A. An important point about piggybacking is that both sites must use the same algorithm. This algorithm is complicated because it needs to combine two arrival events into one.

HDLC

High-level Data Link Control (HDLC) is a bit-oriented protocol for communication over point-to-point and multipoint links. It implements the ARQ mechanisms.

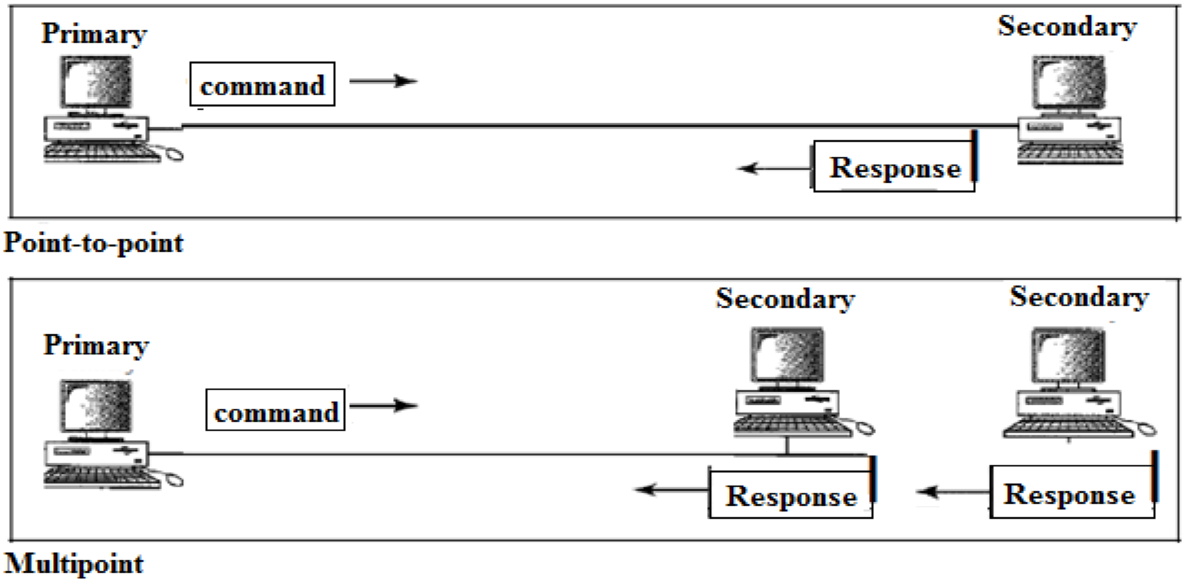
Configurations and Transfer Modes

HDLC provides two common transfer modes that can be used in different configurations: normal response mode (NRM) and asynchronous balanced mode (ABM).

Normal Response Mode

In normal response mode (NRM), the station configuration is unbalanced. We have one primary station and multiple secondary stations. A primary station can send commands; a secondary station can only respond. The NRM is used for both point-to-point and multiple-point links, as shown in Figure 2.20.

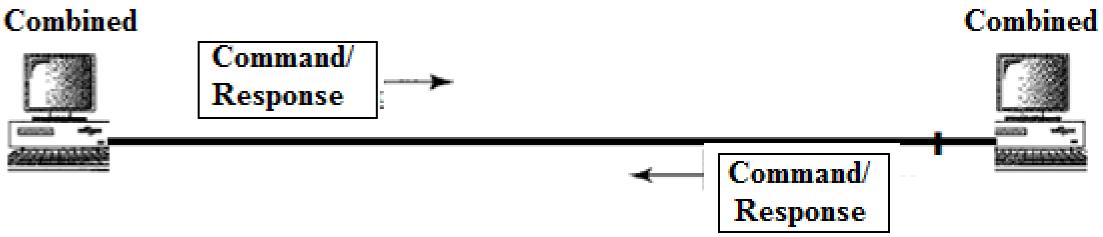
Figure 2.20 Normal response mode



Asynchronous Balanced Mode

In asynchronous balanced mode (ABM), the configuration is balanced. The link is point-to-point, and each station can function as a primary and a secondary (acting as peers), as shown in Figure 2.21. This is the common mode today.

Figure 2.21 Asynchronous balanced mode



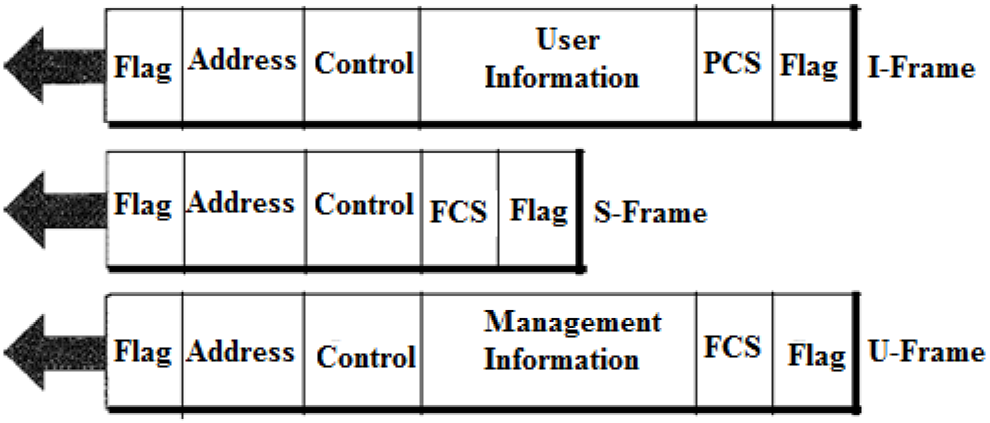
Frames for HDLC

To provide the flexibility necessary to support all the options possible in the modes and configurations just described, HDLC defines three types of frames: information frames (I-frames), supervisory frames (S-frames), and unnumbered frames (V-frames). Each type of frame serves as an envelope for the transmission of a different type of message. I-frames are used to transport user data and control information relating to user data (piggybacking). S-frames are used only to transport control information. U-frames are reserved for system management. Information carried by U-frames is intended for managing the link itself.

Frame Format

Each frame in HDLC may contain up to six fields, as shown in Figure 2.22. Beginning flag field, an address field, a control field, an information field, a frame check sequence (FCS) field, and an ending flag field. In multiple-frame transmissions, the ending flag of one frame can serve as the beginning flag of the next frame.

Figure 2.22 HDLC Frames



Fields

Let us now discuss the fields and their use in different frame types.

Flag field: The flag field of an HDLC frame is an 8-bit sequence with the bit pattern 01111110 that identifies both the beginning and the end of a frame and serves as a synchronization pattern for the receiver.

Address field: The second field of an HDLC frame contains the address of the secondary station. If a primary station created the frame, it contains a *to* address. If a secondary creates the frame, it contains *from* address. An address field can be 1 byte or several bytes long, depending on the needs of the network. One byte can identify up to 128 stations (l bit is used for another purpose).

Control field: The control field is a 1- or 2-byte segment of the frame used for flow and error control. The interpretation of bits in this field depends on the frame type.

Information field: The information field contains the user's data from the network layer or management information. Its length can vary from one network to another.

FCS field: The frame check sequence (FCS) is the HDLC error detection field. It can contain either a 2- or 4-byte ITU-T CRC.

RANDOM ACCESS

In random access or contention methods, no station is superior to another station and none is assigned the control over another. No station permits, or does not permit, another station to send. 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. These decisions transmit when it desires on the condition that it follows the predefined procedure, including the testing of 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.

The random access methods we study in this chapter have evolved from a very interesting protocol known as ALOHA, which used a very simple procedure called multiple accesses (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: CSMA with collision detection (CSMA/CD) and CSMA with collision avoidance *(CSMA/CA).*

ALOHA

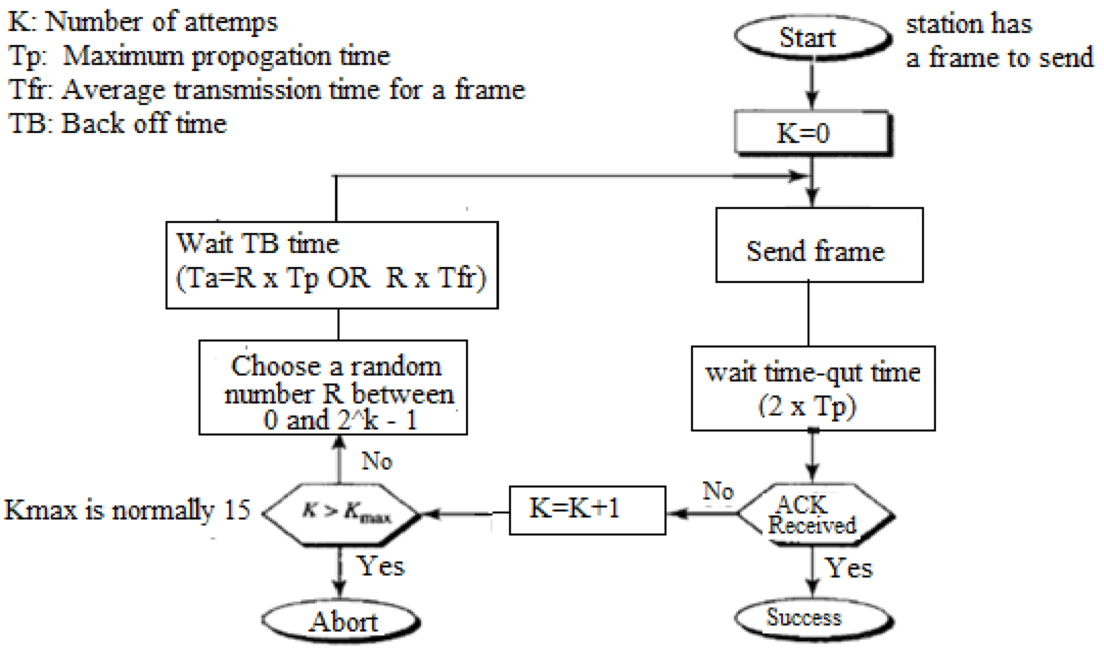
ALOHA, the earliest random access method was developed at the University of Hawaii in early 1970. It was designed for a radio (wireless) LAN, but it can be used on any shared medium. It is obvious that there are potential collisions in this arrangement. The medium is shared between the stations. When 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. However, since there is only one channel to share, there is the possibility of collision between frames from different stations. It is obvious that we need to resend the frames that have been destroyed during transmission. 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 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. 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 back-off time *TB.* Pure ALOHA has a second method to prevent congesting the channel with retransmitted frames.

Figure 2.23 Procedure for pure ALOHA

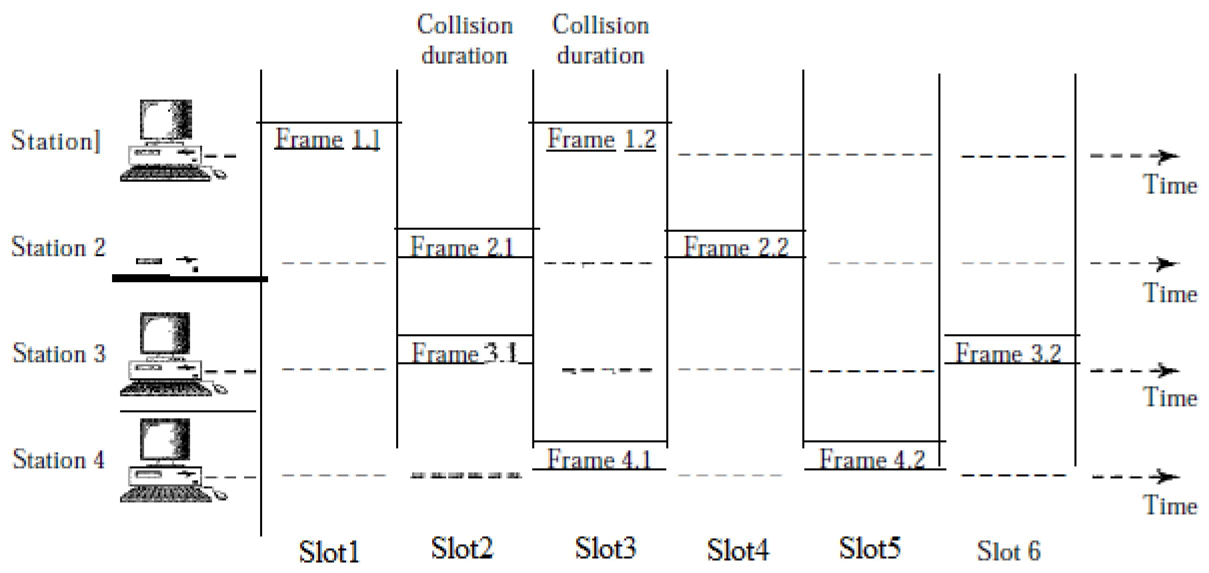


The time-out period is equal to the maximum possible round-trip propagation delay, which is twice the amount of time required to send a frame between the two most widely separated stations (2 x *Tp)'* The back-off time *TB* is a random value that normally depends on *K* (the number of attempted unsuccessful transmissions). The formula for *TB* depends on the implementation. One common formula is the **binary exponential back-off.** The value of *Kmax* is usually chosen as 15. **The Throughput for Pure ALOHA is $S = G \times e^{-2G}$. The maximum throughput is $S_{max} = 0.184$ when $G = 1/2$**

Slotted ALOHA

Pure ALOHA has a vulnerable time of 2 x T_{fr} . This is so because there is no rule that defines when the station can send. A station may send soon after another station has started or soon before another station has finished. Slotted ALOHA was invented to improve the efficiency of pure ALOHA. In slotted ALOHA we divide the time into slots of T_{fr} s and force the station to send only at the beginning of the time slot. Figure 12.6 shows an example of frame collisions in slotted ALOHA.

Figure 2.24 Frames in a Slotted ALOHA



Because 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.

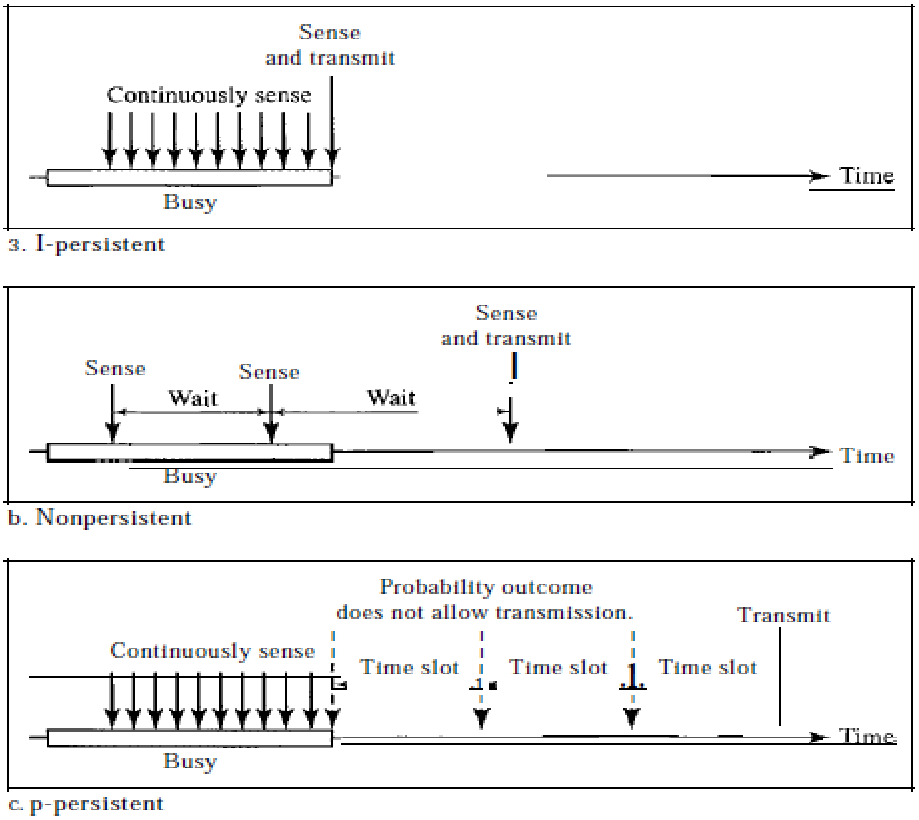
Carrier Sense Multiple Access (CSMA)

To minimize the chance of collision and, therefore, 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. 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.

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 I-persistent method, the non-persistent method, and the p-persistent method. Figure 12.10 shows the behavior of three persistence methods when a station finds a channel busy.

Figure 2.25 Three persistent methods



3. I-persistent

b. Nonpersistent

c. p-persistent

Vulnerable Time

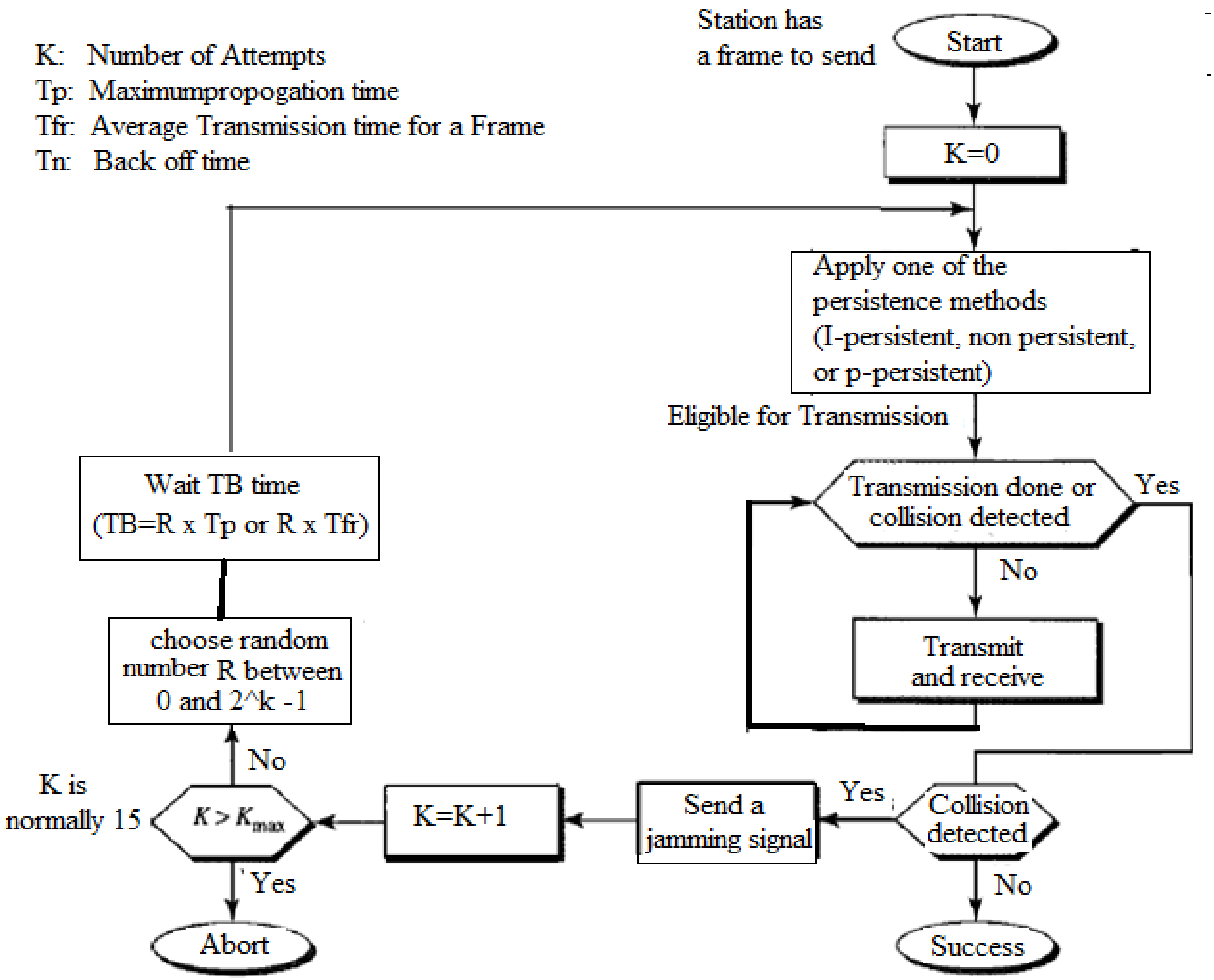
The vulnerable time for CSMA is the propagation time *Tp*. This is the time needed for a signal to propagate from one end of the medium to the other. When a station sends a frame, and any other station tries to send a frame during this time, a collision will result. But if the first bit of the frame reaches the end of the medium, every station will already have heard the bit and will refrain from sending.

Carrier Sense Multiple Access with Collision Detection (CSMA/CD)

The CSMA method does not specify the procedure following a collision. Carrier sense Multiple Access with collision detection (CSMA/CD) augments the algorithm to handle the collision. In this method, 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 12.12, stations A and C are involved in the collision.

Figure 2.26 Procedure for CSMA with Collision Detection (CSMA/CD)



K: Number of Attempts
Tp: Maximumpropogation time
Tfr: Average Transmission time for a Frame
Tn: Back off time
Station has a frame to send
Start
K=0
Apply one of the persistence methods (I-persistent, non persistent, or p-persistent)
Eligible for Transmission
Transmission done or collision detected
Yes
No
Transmit and receive
Collision detected
Yes
No
Success
Send a jamming signal
K=K+1
$K > K_{max}$
K is normally 15
No
Yes
Abort
choose random number R between 0 and 2^k -1
Wait TB time (TB=R x Tp or R x Tfr)

Minimum Frame Size

For *CSMAlCD* 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 T_p. To understand the reason, let us think about the worst-case scenario. If the two stations involved in a collision are the maximum distance apart, the signal from the first takes time T_p to reach the second and the effect of the collision takes another time T_p to reach the first. So the requirement is that the first station must still be transmitting after $2T_p$.

Procedure

The flow diagram for *CSMA/CD is shown in* Figure 2.26. It is similar to the one for the ALOHA protocol, but there are differences. The first difference is the addition of the persistence process. The corresponding box can be replaced by one of the persistence. The second difference is the frame transmission. In ALOHA, we first transmit the entire frame and then wait for an acknowledgment. In *CSMA/CD,* transmission and collision detection is a continuous process. We do not send the entire frame and then look for a collision. The station transmits and receives continuously and simultaneously with the help of a loop. When we come out of the loop, if a collision has not been detected, it means that transmission is complete; the entire frame is transmitted. Otherwise, a collision has occurred. The third difference is the sending of a short jamming signal that enforces the collision in case other stations have not yet sensed the collision.

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 I-persistent method the maximum throughput is around 50 percent when G =1. For non persistent method, the maximum throughput can go up to 90 percent when G is between 3 and 8.

Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)

In a wireless network, much of the sent energy is lost in transmission. The received signal has very little energy. Therefore, a collision may add only 5 to 10 percent additional energy. This is not useful for effective collision detection. We need to avoid collisions on wireless networks because they cannot be detected. Carrier sense multiple access with collision avoidance *(CSMA/CA)* was invented for this network. Collisions are avoided through the use of CSMAICA's three strategies: the inter-frame space, the contention window, and acknowledgments.

Inter-frame 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 inter-frame space or IFS. Even though the channel may appear idle when it is sensed, a distant station may have already started transmitting. The IFS time allows the front of the transmitted signal by the distant station to reach this station. The IFS variable can also be used to prioritize stations or frame types.

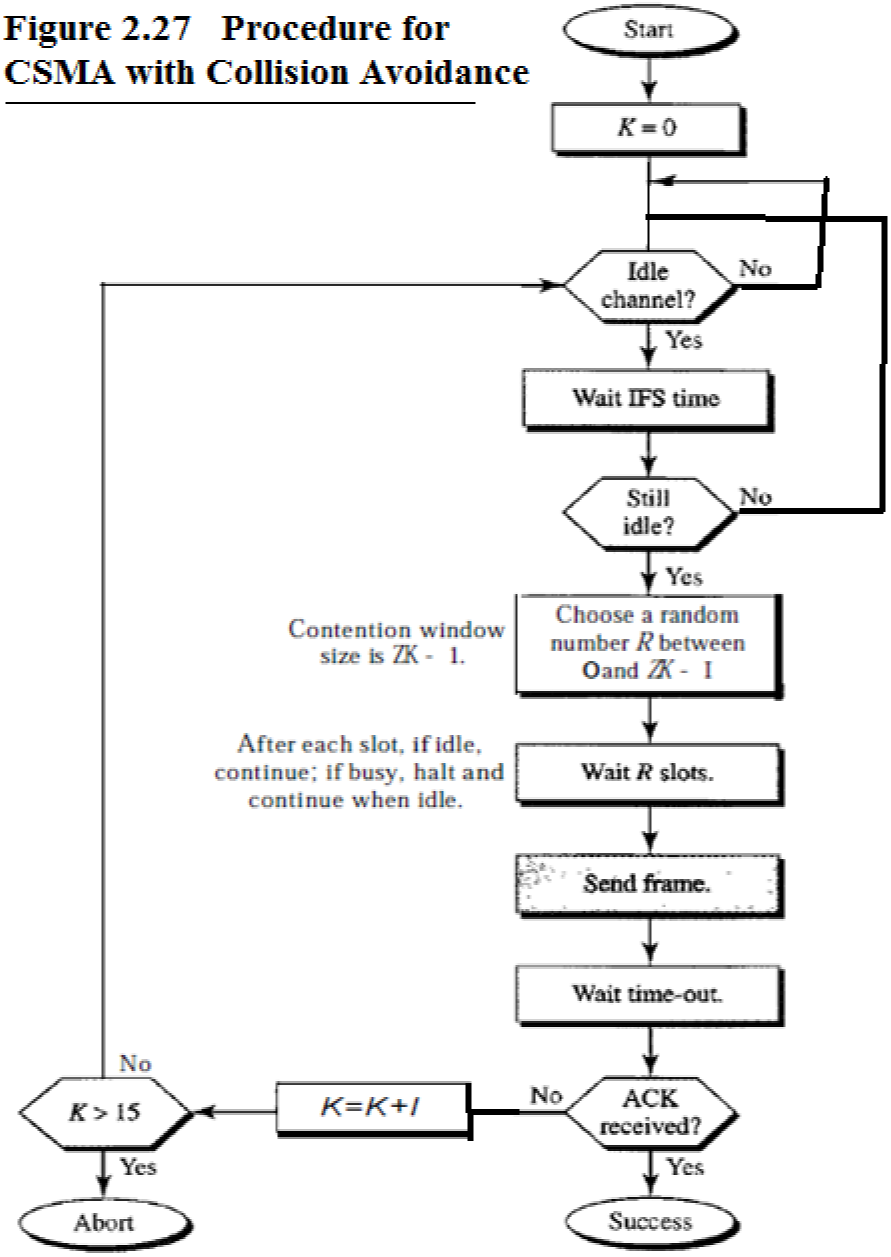
Contention Window: The contention window is an amount of time divided into slots. A station that is ready to send chooses a random number of slots as its wait time. The number of slots in the window changes according to the binary exponential back-off strategy. One interesting point about the contention window is that the station needs to sense the channel after each time slot. In CSMA/CA, if the station finds the channel busy, it does not restart the timer of the contention window; it stops the timer and restarts it when the channel becomes idle.

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.

Procedure

Figure shows the procedure. Note that the channel needs to be sensed before and after the IFS. The channel also needs to be sensed during the contention time. For each time slot of the contention window, the channel is sensed. If it is found idle, the timer continues; if the channel is found busy, the timer is stopped and continues after the timer becomes idle again.

Figure 2.27 Procedure for CSMA with Collision Avoidance


Start
K = 0
Idle channel?
No
Yes
Wait IFS time
Still idle?
No
Yes
Contention window size is ZK - 1.
Choose a random number R between Oand ZK - I
After each slot, if idle, continue; if busy, halt and continue when idle.
Wait R slots.
Send frame.
Wait time-out.
ACK received?
No
Yes
K=K+I
K > 15
No
Yes
Abort
Success

CSMA/CA and Wireless Networks

CSMAICA was mostly intended for use in wireless networks. The procedure described above, however, is not sophisticated enough to handle some particular issues related to wireless networks, such as hidden terminals or exposed terminals. We will see how these issues are solved by augmenting the above protocol with hand-shaking features.

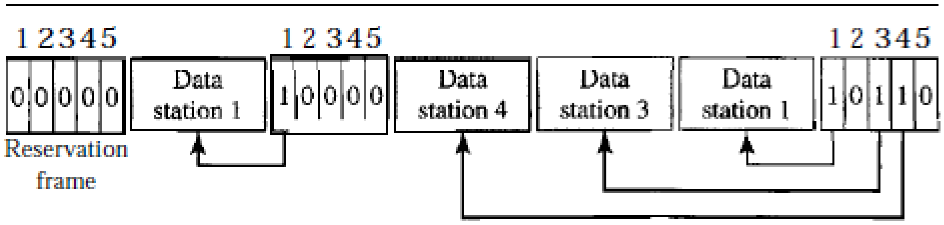
CONTROLLED ACCESS

In controlled access, the stations consult one another to find which station has the right to send. A station cannot send unless it has been authorized by other stations. We discuss three popular controlled-access methods.

Reservation

In the reservation method, a station needs to make a reservation before sending data. Time is divided into intervals. In each interval, a reservation frame precedes the data frames sent in that interval. If there are *N* stations in the system, there are exactly *N* reservation mini slots in the reservation frame. Each mini slot belongs to a station.

Figure 2.28 Reservation access method



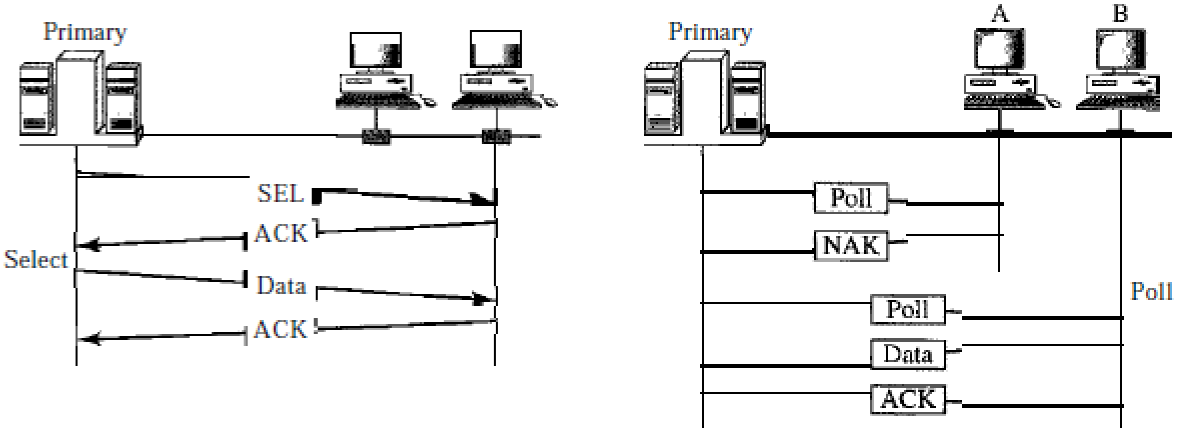
When a station needs to send a data frame, it makes a reservation in its own mini slot. The stations that have made reservations can send their data frames after the reservation frame. Figure 2.28 shows a situation with five stations and a five-mini slot reservation frame. In the first interval, only stations 1, 3, and 4 have made reservations. In the second interval, only station 1 has made a reservation.

Polling

Polling works with topologies in which one device is designated as a primary station and the other devices are secondary stations. All data exchanges must be made through the primary device even when the ultimate destination is a secondary device. The primary device controls the link; the secondary devices follow its instructions. It is up to the primary device to determine which device is allowed to use the channel at a given time. The primary device, therefore, is always the initiator of a session (see Figure 2.29).

If the primary wants to receive data, it asks the secondaries if they have anything to send; this is called poll function. If the primary wants to send data, it tells the secondary to get ready to receive; this is called select function.

Figure 2.29 Select and poll function in polling access method



Select: The *select* function is used whenever the primary device has something to send. Remember that the primary controls the link. If the primary is neither sending nor receiving data, it knows the link is available. If it has something to send, the primary device sends it. Before sending data, the primary creates and transmits a select (SEL) frame, one field of which includes the address of the intended secondary.

Poll: The *poll* function is used by the primary device to solicit transmissions from the secondary devices. When the primary is ready to receive data, it must ask (poll) each device in turn if it has anything to send. If the response is negative (NAK), then the primary polls the next secondary in the same manner until it finds one with data to send. When the response is positive, the primary reads the frame and returns an acknowledgment (ACK frame), verifying its receipt.

Token Passing

In the token-passing method, the stations in a network are organized in a logical ring. In other words, for each station, there is a *predecessor* and a *successor*. The predecessor is the station which is logically before the station in the ring; the successor is the station which is after the station in the ring. In this method, a special packet called a token circulates through the ring. The possession of the token gives the station the right to access the channel and send its data. Token management is needed for this access method.

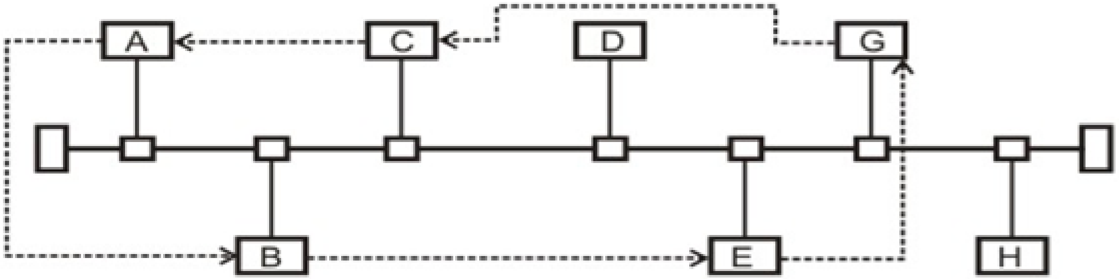
Logical Ring: In a token-passing network, stations do not have to be physically connected in a ring; the ring can be a logical one. Figure 2.30 show four different physical topologies that can create a logical ring. In the physical ring topology, when a station sends the token to its successor, the token cannot be seen by other stations; the successor is the next one in line. This means that the token does not have to have the address of the next successor. The high-speed Token Ring networks called FDDI (Fiber Distributed Data Interface) and CDDI (Copper Distributed Data Interface) use this topology.

IEEE STANDARDS

IEEE 802.4 (Token Bus)

Token bus is a network implementing the token ring protocol over a "virtual ring" on a coaxial cable. A token is passed around the network nodes and only the node possessing the token may transmit. If a node doesn't have anything to send, the token is passed on to the next node on the virtual ring. Each node must know the address of its neighbour in the ring, so a special protocol is needed to notify the other nodes of connections to, and disconnections from, the ring. IEEE 802.4 describes a token bus LAN standard.

Figure 2.31 Token Bus



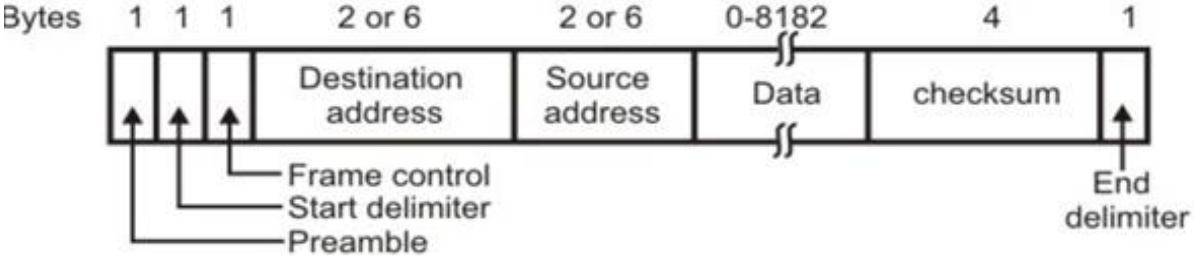
Functions of a Token Bus: It is the technique in which the station on bus or tree forms a logical ring that is the stations are assigned positions in an ordered sequence, with the last number of the sequence followed by the first one as shown in Fig.2.27. Each station knows the identity of the station following it and preceding it.

A control packet known as a *Token* regulates the right to access. When a station receives the token, it is granted control to the media for a specified time, during which it may transmit one or more packets. It may poll stations and receive responses when the station is done, or if its time has expired then it passes token to next station in logical sequence. Token propagates through the logical ring (dotted line) with only the token holder being permitted to transmit frames (in the fig2.31, node A,B,E,G, and C only having frames).

IEEE 802.4 MAC sub-layer protocol

The frame format of the Token Bus is shown in Fig.2.28. It consists of the following field

Figure 2.32 Frame format of IEEE 802.4

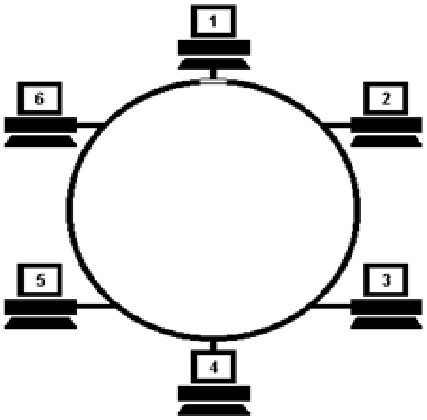


Preamble (1 byte): The preamble is an at least one octet to establish bit synchronization.
Start delimiter (1 byte): Alerts each station of the arrival of a token (or data/command frame). This field includes signals that distinguish the byte from the rest of the frame by violating the encoding scheme used elsewhere in the frame.
Frame-control byte (1 byte): Indicates whether the frame contains data or control information. In control frames, this byte specifies the type of control information.
Destination and source addresses (2-6 bytes): Consists of two 6-byte address fields that identify the destination and source station addresses.
Frame-check sequence (FCS- 4 byte): Is filed by the source station with a calculated value dependent on the frame contents. The destination station recalculates the value to determine whether the frame was damaged in transit. If so, the frame is discarded.
End delimiter (1 byte): Signals the end of the token or data/command frame. This field also contains bits to indicate a damaged frame and identify the frame that is the last in a logical sequence.

IEEE 802.5 (Token Ring)

Token Ring and IEEE802.5 are based on token passing MAC protocol with ring topology. They resolve the uncertainty by giving each station a turn on by one. Each node takes turns sending the data; each station may transmit data during its turn. The technique that coordinates this turn mechanism is called Token passing; as a Token is passed in the network and the station that gets the token can only transmit. As one node transmits at a time, there is no chance of collision.
Token Ring Operation: Token-passing networks move a small frame, called a *token*, around the network as shown in Fig2.33. Possession of the token grants the right to transmit. If a node receiving the token has no information to send, it passes the token to the next end station. Each station can hold the token for a maximum period of time.

Figure 2.33 Token ring



If a station possessing the token does have information to transmit, it seizes the token, alters 1 bit of the token, appends the information that it wants to transmit, and sends this information to the next station on the ring. *Collisions cannot occur in Token Ring networks*. The step by step explanation is given below.

1. Initially a free token is circulating on the ring; if machine 1 wants to send some data to machine 4, so it first has to capture the free token.
2. It then writes its data and the recipient's address onto the token. The packet of data is then sent to machine 2 who reads the address, realizes it is not its own, so passes it on to 3.
3. Machine 3 does the same and passes the token on to machine 4.
4. Now the machine 4 reads the message, it cannot release a free token on to the ring. The machine 4 must first send the message back to number 1 with an acknowledgement to say that it has received the data.
5. The receipt is then sent to machine 5 who checks the address, realizes that it is not its own and so forwards it on to the next machine in the ring, number 6.
6. Machine 6 does the same and forwards the data to number 1, who sent the original message. Machine 1 recognizes the address, reads the acknowledgement from number 4 and then releases the free token back on to the ring ready for the next machine to use.

Advantage of token ring

1. It is flexible control over access that it provides.
2. It used to regulate access to provide for priority and for guaranteed bandwidth services.

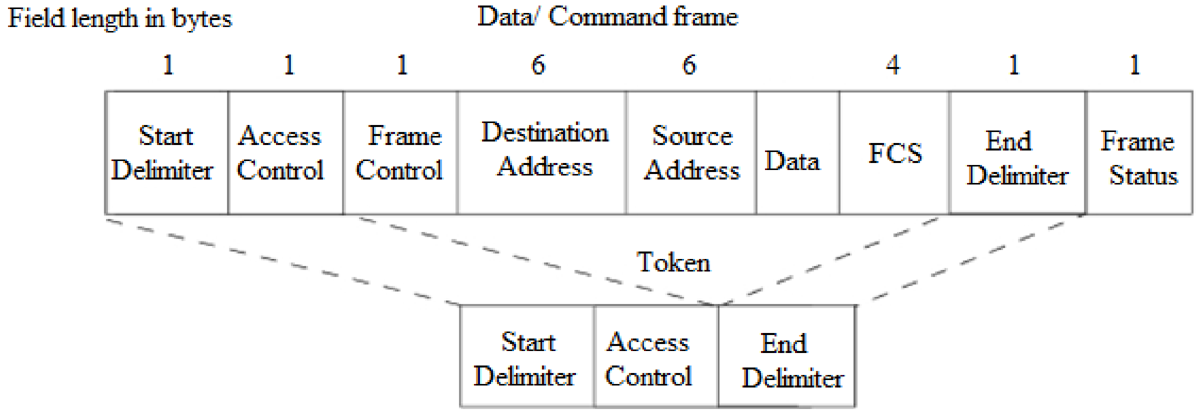
Disadvantage of token ring

1. Token maintenance is very difficult.
2. Duplication of the token can also disrupt ring operation

IEEE 802.5 Frame Format

Token Ring and IEEE 802.5 support two basic frame types: tokens and data/command frames. Tokens are 3 bytes in length and consist of a start delimiter, an access control byte, and an end delimiter.

Figure 2.34 IEEE 802.5 and Token Ring Specify Tokens and Data/Command Frames

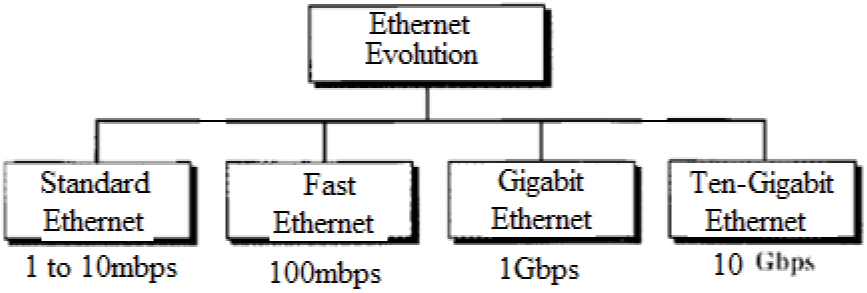


Data/command frames vary in size, depending on the size of the Information field. Data frames carry information for upper-layer protocols, while command frames contain control information and have no data for upper-layer protocols.

Start delimiter—Alerts each station of the arrival of a token (or data/command frame). This field includes signals that distinguish the byte from the rest of the frame by violating the encoding scheme used elsewhere in the frame.
Access-control byte—Contains the Priority field (the most significant 3 bits) and the Reservation field (the least significant 3 bits), as well as a token bit and a monitor bit.
End delimiter—Signals the end of the token or data/command frame. This field also contains bits to indicate a damaged frame and identify the frame that is the last in a logical sequence.
Start delimiter: Alerts each station of the arrival of a token. This field includes signals that distinguish the byte from the rest of the frame by violating the encoding.
Access-control-byte: Contains the Priority field and the Reservation field, as well as a token and a monitor bit.
Frame-control bytes: Indicates whether the frame contains data or control information. In control frames, this byte specifies the type of control information.

IEEE 802.3 (ETHERNET)

The original Ethernet was created in 1976 at Xerox's Palo Alto Research Center (PARC). Since then, it has gone through four generations: Standard Ethernet (lot Mbps), Fast Ethernet (100 Mbps), Gigabit Ethernet (l Gbps), and Ten-Gigabit Ethernet (l0 Gbps).



MAC Sub-layer Protocol

The frame format as per IEEE 802.3 standard is shown in below figure. The frame contains two addresses, one for the destination and one for the source.

Field size	7	1	2/6	2/6	2	0 – 1500	0-46	4	octets
	Preamble	SFD	DA	SA	L	Data	PAD	FCS	

SFD – start of frame delimiter   L – Length of data field
DA – Destination address   FCS – frame check sequence
SA – Source address

Preamble: Each frame starts with a preamble. The preamble is a 7 octets (7–bytes) long pattern to establish bit synchronization.

Start of frame delimiter: It is one octet long unique bit pattern which marks the start of the frame. It is 1 byte long.

Destination Address: The destination address field address is 2 or 6 octets.
Source Address: The source address field address is 2 or 6 octets.
Length (L): This field is two octets long and indicates the number of octets in the data fields.
Data Field: It can have 46-1500 octets if the address field option is octets. If data octets are less than 46, the PAD field makes up the difference. This ensures that minimum size of the frame.
Frame Check Sequence (FCS): The frame check sequence is 4 octets long and contains the CRC code. It checks on DA, SA, L and PAD fields.

FAST ETHERNET

Fast Ethernet was designed to compete with LAN protocols such as FDDI or Fiber Channel (or Fiber Channel, as it is sometimes spelled). IEEE created Fast Ethernet under the name 802.3u. Fast Ethernet is backward-compatible with Standard Ethernet, but it can transmit data 10 times faster at a rate of 100 Mbps. The goals of Fast Ethernet can be summarized as follows:

1. Upgrade the data rate to 100 Mbps.
2. Make it compatible with Standard Ethernet.
3. Keep the same 48-bit address.
4. Keep the same frame format.
5. Keep the same minimum and maximum frame lengths.

MAC Sub layer

A main consideration in the evolution of Ethernet from 10 to 100 Mbps was to keep the MAC sub layer untouched. However, a decision was made to drop the bus topologies and keep only the star topology. For the star topology, there are two choices, as we saw before: half duplex and full duplex. In the half-duplex approach, the stations are connected via a hub; in the full-duplex approach, the connection is made via a switch with buffers at each port.

The access method is the same *(CSMA/CD)* for the half-duplex approach; for full duplex Fast Ethernet, there is no need for *CSMA/CD.* However, the implementations keep *CSMA/CD* for backward compatibility with Standard Ethernet.

Auto negotiation: Auto negotiation allows two devices to negotiate the mode or data rate of operation. It was designed particularly for the following purposes:

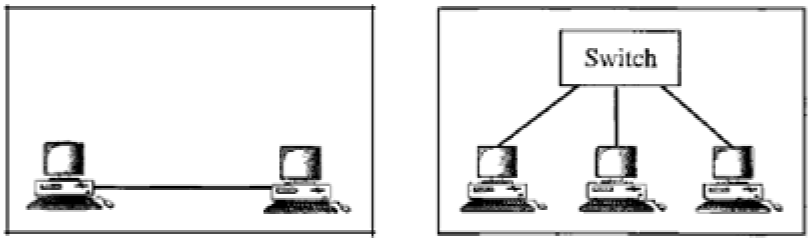
- To allow incompatible devices to connect to one another.
- To allow one device to have multiple capabilities.
- To allow a station to check a hub's capabilities.

Physical Layer

The physical layer in Fast Ethernet is more complicated than the one in Standard Ethernet. We briefly discuss some features of this layer.

Topology: Fast Ethernet is designed to connect two or more stations together. If there are only two stations, they can be connected point-to-point. Three or more stations need to be connected in a star topology with a hub or a switch at the center, as shown in Figure 2.35.

Figure 2.35 Fast Ethernet Topology

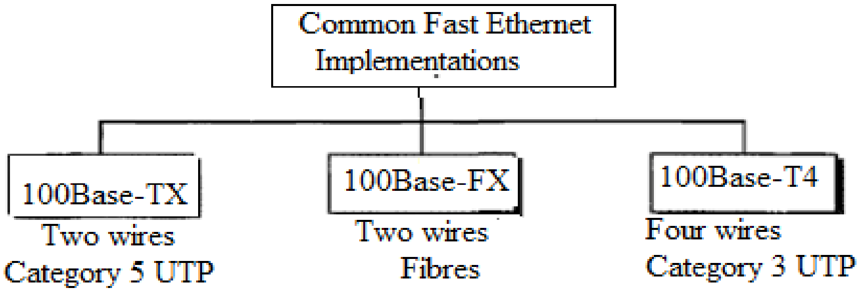


a. Point-to-point

b. Star

Encoding: Manchester encoding needs a 200-Mbaud bandwidth for a data rate of 100 Mbps, which makes it unsuitable for a medium such as twisted-pair cable. For this reason, the Fast Ethernet designers sought some alternative encoding/decoding scheme.

Implementation: Fast Ethernet implementation at the physical layer can be categorized as either two-wire or four-wire. The two-wire implementation can be either category 5 UTP (100Base-TX) or fiber-optic cable (l00 Base-FX). The four-wire implementation is designed only for category 3 UTP (l00Base-T4).



100Base-TX: uses two pairs of twisted-pair cable (either category 5 UTP or STP). For this implementation, the MLT-3 scheme was selected since it has good bandwidth performance (see Chapter 4). However, since MLT-3 is not a self-synchronous line coding scheme, 4B/5B block coding is used to provide bit synchronization by preventing the occurrence of a long sequence of 0s and 1s. This creates a data rate of 125 Mbps, which is fed into MLT-3 for encoding.

100Base-FX uses two pairs of fiber-optic cables. Optical fiber can easily handle high bandwidth requirements by using simple encoding schemes. The designers of 100Base-FX selected the NRZ-I encoding scheme for this implementation. The block encoding increases the bit rate from 100 to 125 Mbps, which can easily be handled by fiber-optic cable. A 100Base-TX network can provide a data rate of 100 Mbps, but it requires the use of category 5 UTP or STP cable. This is not cost-efficient for buildings that have already been wired for voice-grade twisted-pair.

100Base-T4: A new standard, called 100Base-T4 was designed to use category 3 or higher UTP. The implementation uses four pairs of UTP for transmitting 100 Mbps. Encoding/decoding in 100Base-T4 is more complicated. As this implementation uses category 3 UTP, each twisted-pair cannot easily handle more than 25 Mbaud. In this design, one pair switches between sending and receiving. Three pairs of UTP category 3, however, can handle only 75 Mbaud (25 Mbaud) each. In 8B/6T, eight data elements are encoded as six signal elements. This means that 100 Mbps uses only (6/8) x 100 Mbps, or 75 Mbaud.

GIGABIT ETHERNET

The need for an even higher data rate resulted in the design of the Gigabit Ethernet protocol (1000 Mbps). The IEEE committee calls the Standard 802.3z. The goals of the Gigabit Ethernet design can be summarized as follows:

1. Upgrade the data rate to 1 Gbps.
2. Make it compatible with Standard or Fast Ethernet.
3. Use the same 48-bit address.
4. Use the same frame format.
5. Keep the same minimum and maximum frame lengths.
6. To support auto negotiation as defined in Fast Ethernet.

MAC Sub-layer

A main consideration in the evolution of Ethernet was to keep the MAC sub layer untouched. However, to achieve a data rate 1 Gbps, this was no longer possible. Gigabit Ethernet has two distinctive approaches for medium access: half-duplex and full-duplex. Almost all implementations of Gigabit Ethernet follow the full-duplex approach.

Full-Duplex Mode: In full-duplex mode, there is a central switch connected to all computers or other switches. In this mode, each switch has buffers for each input port in which data are stored until they are transmitted. There is no collision; the maximum length of the cable is determined by the signal attenuation in the cable.

Half-Duplex Mode: Gigabit Ethernet can also be used in half-duplex mode, although it is rare. In this case, a switch can be replaced by a hub, which acts as the common cable in which a collision might occur. The half-duplex approach uses *CSMA/CD.* However, as we saw before, the maximum length of the network in this approach is totally dependent on the minimum frame size. Three methods have been defined: traditional, carrier extension, and frame bursting.

Traditional: In the traditional approach, we keep the minimum length of the frame as in traditional Ethernet (512 bits). However, because the length of a bit is 11100 shorter in Gigabit Ethernet than in l0-Mbps Ethernet, the slot time for Gigabit Ethernet is 512 bits x 111000 *J/S*, which is equal to 0.512 J/S. The maximum length of the network is 25 m.

Carrier Extension: To allow for a longer network, we increase the minimum frame length. The carrier extension approach defines the minimum length of a frame as 512 bytes (4096 bits).

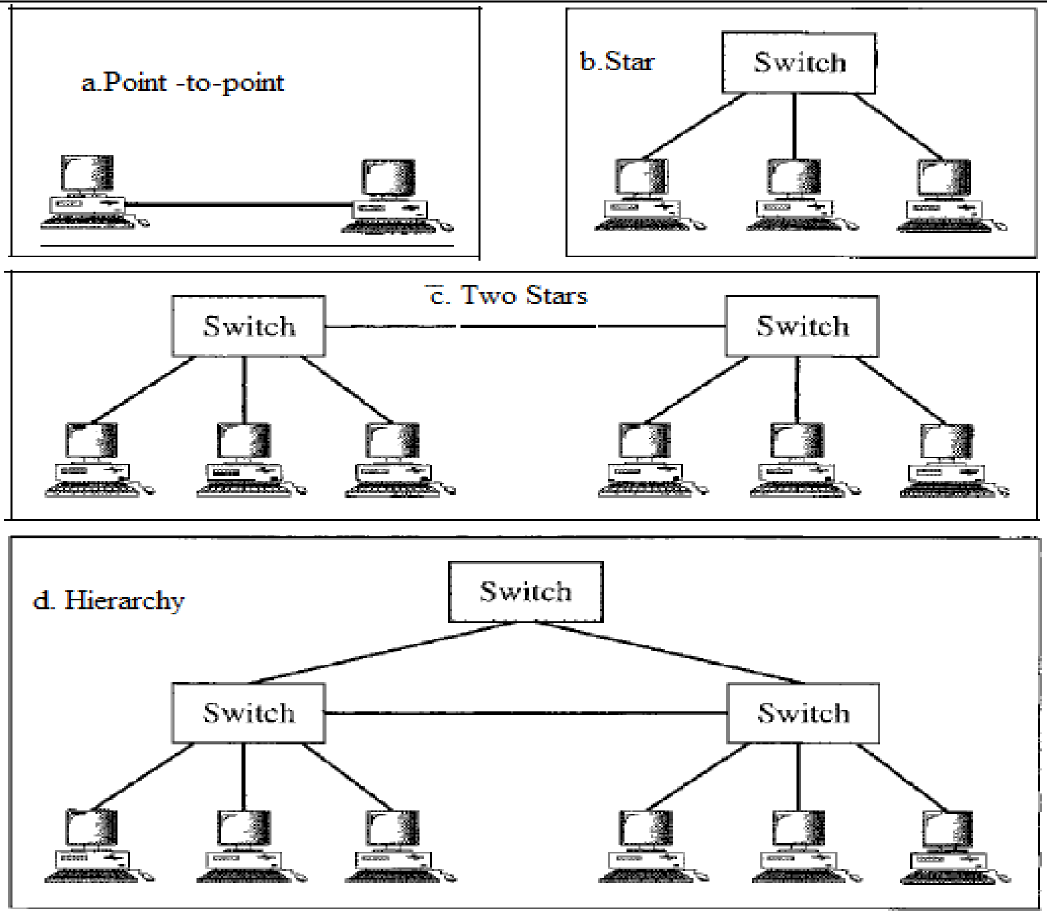
Frame Bursting: Carrier extension is very inefficient if we have a series of short frames to send; each frame carries redundant data. To improve efficiency, frame bursting was proposed. Instead of adding an extension to each frame, multiple frames are sent.

Physical Layer

The physical layer in Gigabit Ethernet is more complicated than that in Standard or Fast Ethernet. We briefly discuss some features of this layer.

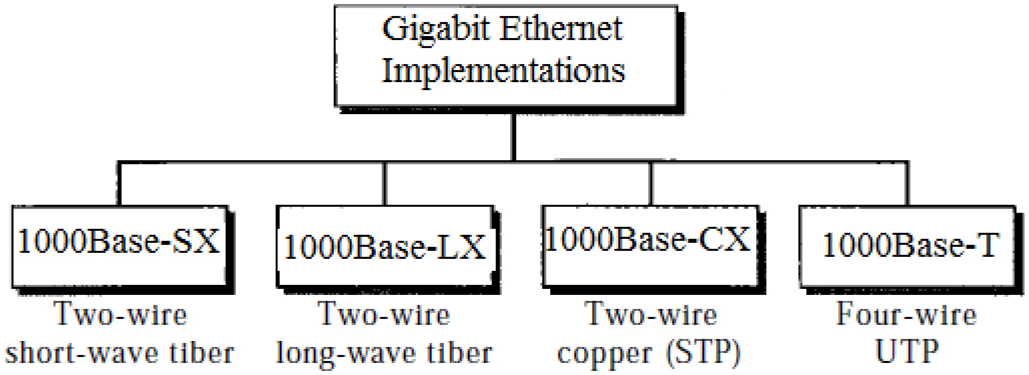
Topology: Gigabit Ethernet is designed to connect two or more stations. If there are only two stations, they can be connected point-to-point.

Figure 2.36 Topologies of Gigabit Ethernet



Three or more stations need to be connected in a star topology with a hub or a switch at the center. Another possible configuration is to connect several star topologies or let a star topology be part of another as shown in Figure 2.36.

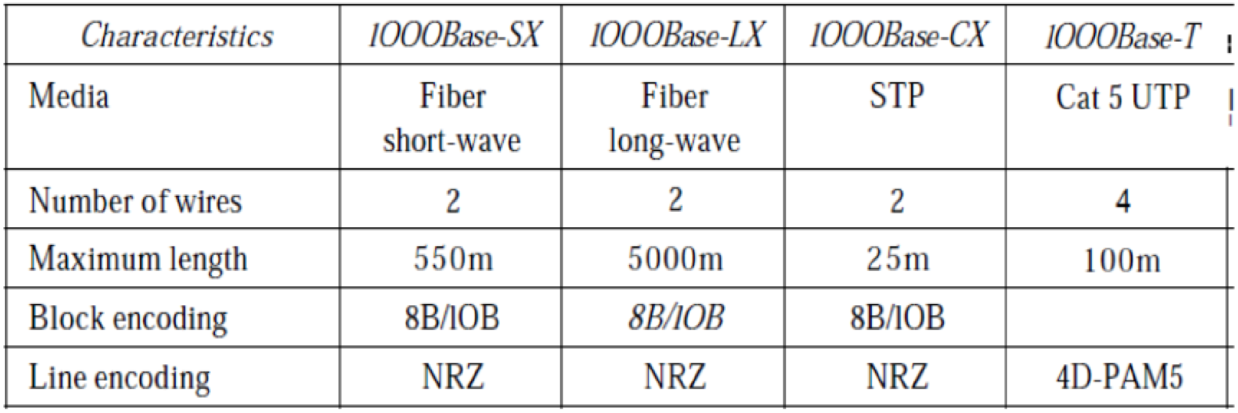
Implementation: Gigabit Ethernet can be categorized as either a two-wire or a four-wire implementation. The two-wire implementations use fiber-optic cable (1000Base-SX, short-wave, or 1000Base-LX, long-wave), or STP (1000Base-CX). The four-wire version uses category 5 twisted-pair cable (l000Base-T).



Encoding: Gigabit Ethernet cannot use the Manchester encoding scheme because it involves a very high bandwidth (2 GBaud). The two-wire implementations use an NRZ scheme, but NRZ does not self-synchronize properly. This block encoding prevents long sequences of 0s or 1s in the stream, but the resulting stream is 1.25 Gbps.

In the four-wire implementation it is not possible to have 2 wires for input and 2 for output, because each wire would need to carry 500 Mbps, which exceeds the capacity for category 5 UTP. As a solution, 4D-PAM5 encoding is used to reduce the bandwidth. Thus, all four wires are involved in both input and output; each wire carries 250 Mbps, which is in the range for category 5 UTP cable.

Table: Summary of Gigabit Ethernet implementations



Characteristics	*1OOOBase-SX*	*1OOOBase-LX*	*1OOOBase-CX*	*1OOOBase-T*
Media	Fiber short-wave	Fiber long-wave	STP	Cat 5 UTP
Number of wires	2	2	2	4
Maximum length	550m	5000m	25m	100m
Block encoding	8B/IOB	*8B/IOB*	8B/IOB	
Line encoding	NRZ	NRZ	NRZ	4D-PAM5

Ten-Gigabit Ethernet

The IEEE committee created Ten-Gigabit Ethernet and called it Standard 802.3ae. The goals of the Ten-Gigabit Ethernet design can be summarized as follows:

1. Upgrade the data rate to 10 Gbps.
2. Make it compatible with Standard, Fast, and Gigabit Ethernet.
3. Use the same 48-bit address.
4. Use the same frame format.
5. Keep the same minimum and maximum frame lengths.
6. Allow the interconnection of existing LANs into a MAN or a WAN.
7. Make Ethernet compatible with technologies such as Frame Relay and ATM.

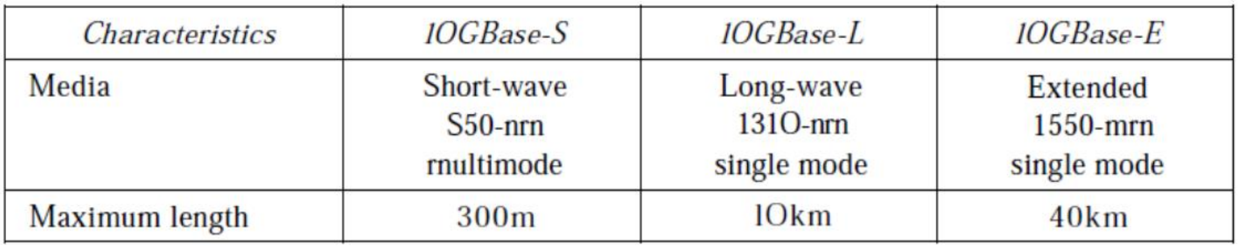
MAC Sub layer

Ten-Gigabit Ethernet operates only in full duplex mode which means there is no need for contention; *CSMA/CD* is not used in Ten-Gigabit Ethernet.

Physical Layer

The physical layer in Ten-Gigabit Ethernet is designed for using fiber-optic cable over long distances. Three implementations are the most common: l0GBase-S, l0GBase-L, and l0GBase-E. Table 13.4 shows a summary of the Ten-Gigabit Ethernet implementations.

Table: ***Summary of Ten-Gigabit Ethernet implementations***



Characteristics	*1OGBase-S*	*1OGBase-L*	*1OGBase-E*
Media	Short-wave S50-nrn rnultimode	Long-wave 131O-nrn single mode	Extended 1550-mrn single mode
Maximum length	300m	1Okm	40km

Wireless LANs

Wireless communication is one of the fastest-growing technologies. The demand for connecting devices without the use of cables is increasing everywhere. Wireless LANs can be found on college campuses, in office buildings, and in many public areas. There are two promising wireless technologies for LANs: IEEE 802.11 wireless LANs (Wireless Ethernet) and Bluetooth (a technology for small wireless LANs).

IEEE 802.11 Architecture

IEEE has defined the specifications for a wireless LAN, called IEEE 802.11, which covers the physical and data link layers. The standard defines two kinds of services: the basic service set (BSS) and the extended service set (ESS).

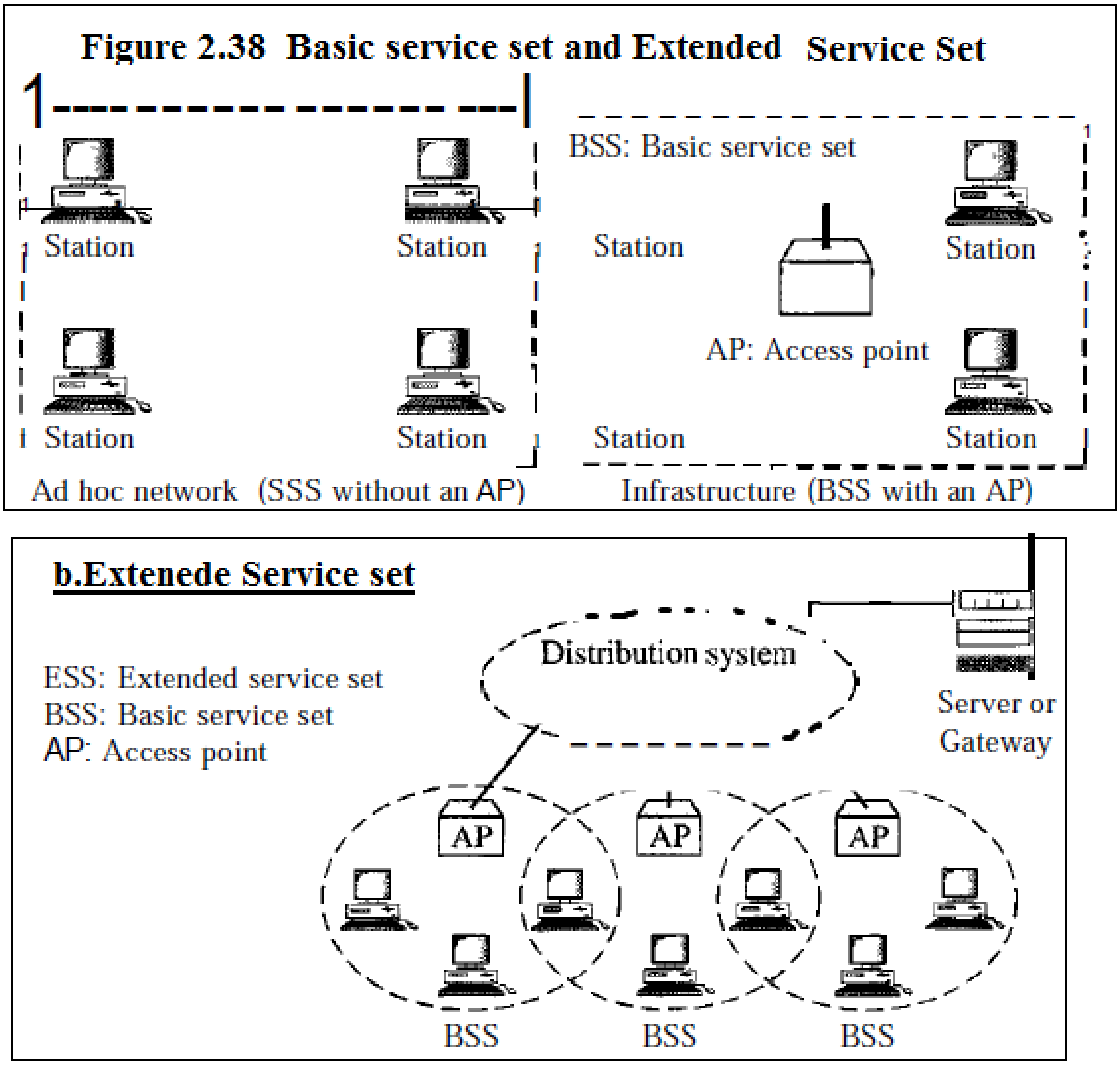


Figure 2.38 Basic service set and Extended Service Set

Basic Service Set: A Basic Service Set (BSS) is made of stationary or mobile wireless stations and an optional central base station, known as the access point (AP). The BSS without an AP is a stand-alone network and cannot send data to other BSSs. It is called an *ad hoc architecture*. In this architecture, stations can form a network without the need of an AP; they can locate one another and agree to be part of a BSS. A BSS with an AP is sometimes referred to as an *infrastructure* network.

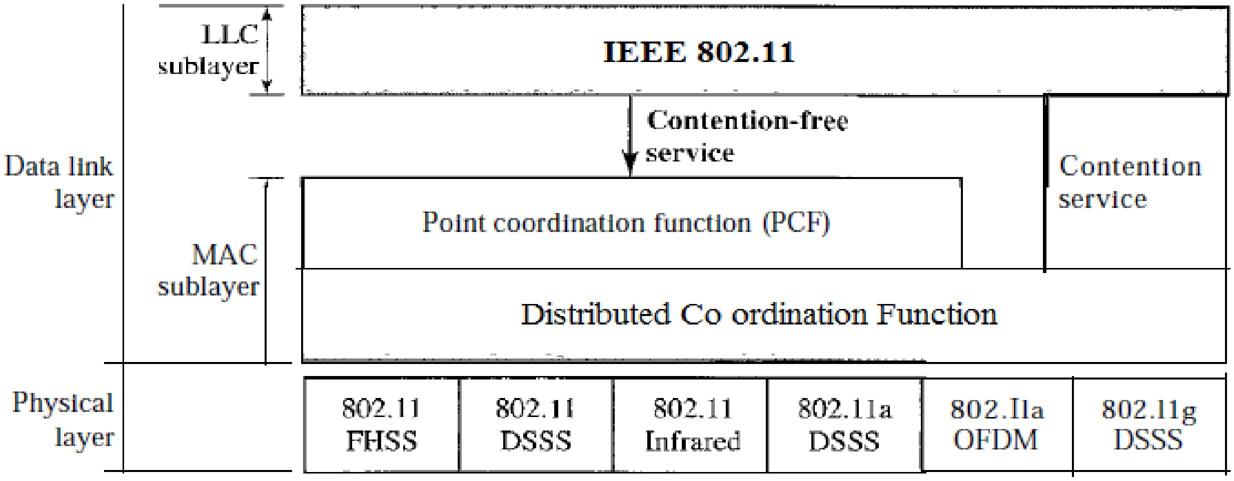
Extended Service Set: An extended service set (ESS) is made up of two or more BSSs with APs. In this case, the BSSs are connected through a *distribution system,* which is usually a wired LAN. The distribution system connects the APs in the BSSs. IEEE 802.11 does not restrict the distribution system; it can be any IEEE LAN such as an Ethernet. Note that the extended service set uses two types of stations: mobile and stationary. The mobile stations are normal stations inside a BSS. The stationary stations are AP stations that are part of a wired LAN. Figure 2.38 shows an ESS.

Station Types: IEEE 802.11 defines three types of stations based on their mobility in a wireless LAN: no-transition, BSS ·transition, and ESS-transition mobility. A station with no-transition mobility is either stationary (not moving) or moving only inside a BSS. A station with BSS-transition mobility can move from one BSS to another, but the movement is confined inside one ESS. A station with ESS-transition mobility can move from one ESS to another. However, IEEE 802.11 does not guarantee that communication is continuous during the move.

MAC Sub layer

IEEE 802.11 defines two MAC sub layers: the distributed coordination function (DCF) and point coordination function (PCF). Figure 2.39 shows the relationship between the two MAC sub layers, the LLC sub layer, and the physical layer.

Figure 2.39 MAC Layers in IEEE 802.11 Standards



.

Frame Types: A wireless LAN defined by IEEE 802.11 have three categories of frames: management frames, control frames, and data frames. Management Frames Management frames is used for the initial communication between stations and access points. Control Frames Control frames are used for accessing the channel and acknowledging frames. Data Frames Data frames are used for carrying data and control information.

Addressing Mechanism: The IEEE 802.11 addressing mechanism specifies four cases, defined by the value of the two flags in the FC field, *To DS* and *From DS.* Each flag can be either 0 or 1, resulting in four different situations. The interpretation of the four addresses (address I to address 4) in the MAC frame depends on the value of these flags.

Physical Layer: All implementations, except the infrared, operate in the *industrial, scientific,* and *medical (ISM)* band, which defines three unlicensed bands in the three ranges 902-928 MHz, 2.400--4.835 GHz, and 5.725-5.850 GHz.

IEEE 802.11 FHSS: It uses the frequency-hopping spread spectrum (FHSS) method and uses the 2.4-GHz ISM band. The band is divided into 79 sub bands of 1 MHz (and some guard bands). A pseudorandom number generator selects the hopping sequence. The modulation technique in this specification is either two-level FSK or four-level FSK with 1 or 2 bits baud, which results in a data rate of 1 or 2 Mbps.

IEEE 802.11 DSSS: IEEE 802.11 DSSS uses the direct sequence spread spectrum (DSSS) method and uses the 2.4-GHz ISM band. The modulation technique in this specification is PSK at 1 Mbaud/s. The system allows 1 or 2 bits l baud (BPSK or QPSK), which results in a data rate of 1 or 2 Mbps.

IEEE 802.11 Infrared

IEEE 802.11 infrared uses infrared light in the range of 800 to 950 nm. The modulation technique is called pulse position modulation (PPM). For a 1-Mbps data rate, a 4-bit sequence is first mapped into a 16-bit sequence in which only one bit is set to 1 and the rest are set to 0. For a 2-Mbps data rate, a 2-bit sequence is first mapped into a 4-bit sequence in which only one bit is set to 1 and the rest are set to 0.

IEEE 802.11a OFDM

IEEE 802.Ila OFDM describes the orthogonal frequency-division multiplexing (OFDM) method for signal generation in a 5-GHz ISM band. OFDM is similar to FDM with one major difference: All the sub bands are used by one source at a given time. Sources contend with one another at the data link layer for access. The band is divided into 52 sub bands, with 48 sub bands for sending 48 groups of bits at a time and 4 sub bands for control information. OFDM uses PSK and QAM for modulation. The common data rates are 18 Mbps (PSK) and 54 Mbps (QAM).

IEEE 802.11b DSSS

IEEE 802.11 b DSSS describes the high-rate direct sequence spread spectrum (HRDSSS) method for signal generation in the 2.4-GHz ISM band. HR-DSSS is similar to DSSS except for the encoding method, which is called complementary code keying (CCK). CCK encodes 4 or 8 bits to one CCK symbol. To be backward compatible with DSSS, HR-DSSS defines four data rates: 1, 2, 5.5, and 11 Mbps. The first two use the same modulation techniques as DSSS.

IEEE 802.11g

This new specification defines forward error correction and OFDM using the 2.4-GHz ISM band. The modulation technique achieves a 22- or 54-Mbps data rate. It is back ward compatible with 802.11b, but the modulation technique is OFDM.

Unit-III

Network Layer: Logical Addressing

Communication at the network layer is host-to-host (computer-to-computer); the packet transmitted by the sending computer may pass through several LANs or WANs before reaching the destination computer.

For this level of communication, we need a global addressing scheme; we called this as logical addressing. Today, we use the term IP address to mean a logical address in the network layer of the TCP/IP protocol suite.

There are two types of internet addresses: one is IPv4 (IP version 4) addresses or simply IP addresses and the new generation of IP or IPv6 (IP version 6).

IPv4 Protocol

- An **IPv4** address is a 32-bit address that *uniquely* and *universally* defines the connection of a device (for example, a computer or a router) to the Internet. An IPv4 address is 32 bits long. IPv4 addresses are unique. They are unique in the sense that each address defines one, and only one, connection to the Internet. Two devices on the Internet can never have the same address at the same time.
- IPv4 is an unreliable and connectionless datagram protocol-a best-effort delivery service. The term *best-effort* means that IPv4 provides no error control or flow control (except for error detection on the header).
- IPv4 is also a connectionless protocol for a packet-switching network that uses the datagram approach. This implies that datagrams sent by the same source to the same destination could arrive out of order.

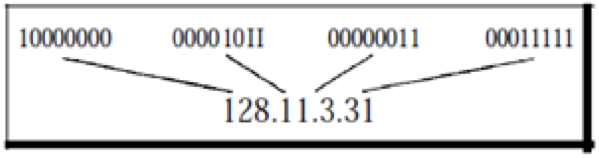
Address Space

IPv4 uses 32-bit addresses, which means that the address space is 2^{32} or 4,294,967,296 (more than 4 billion). This means that, theoretically, if there were no restrictions, more than 4 billion devices could be connected to the Internet.

Notations: There are two prevalent notations to show an IPv4 address: binary notation and dotted decimal notation.

Binary Notation: In binary notation, the IPv4 address is displayed as 32 bits. Each octet is often referred to as a byte. So it is common to hear an IPv4 address referred to as a 32-bit address or a 4-byte address. The following is an example of an IPv4 address in binary notation: 01110101 10010101 00011101 00000010

Figure 3.1 Dotted decimal notation and binary notation for IPv4 address

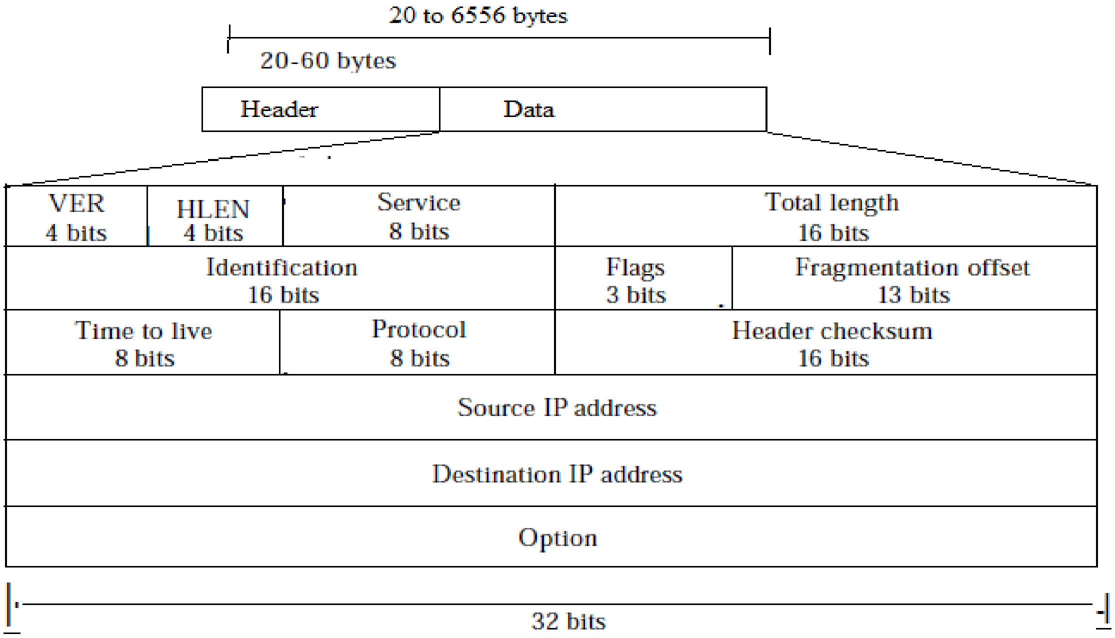


Dotted-Decimal Notation: To make the IPv4 address more compact and easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separating the bytes. The following is the dotted decimal notation of the above address: 117.149.29.2

Figure 3.1 shows an IPv4 address in both binary and dotted-decimal notation. Note that each byte (octet) is 8 bits; each number in dotted-decimal notation is a value ranging from 0 to 255.

Datagram of IPv4

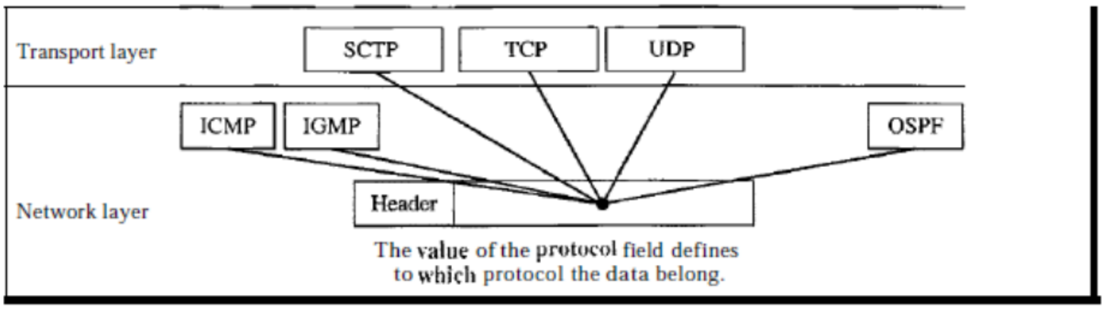
Packets in the IPv4 layer are called datagrams. A datagram is a variable-length packet consisting of two parts: header and data. The header is 20 to 60 bytes in length and contains information essential to routing and delivery.



The fields of IPv4 are described as follows.

Version (VER): This 4-bit field defines the version of the IPv4 protocol. Currently the version is 4. However, version 6 (or IPng) may totally replace version 4 in the future. This field tells the IPv4 software running in the processing machine that the datagram has the format of version 4.

Header length (HLEN): This 4-bit field defines the total length of the datagram header in 4-byte words. This field is needed because the length of the header is variable (between 20 and 60 bytes).
Services: IETF has changed the interpretation and name of this 8-bit field. This field, previously called service type, is now called differentiated services.
Identification: This field is used in fragmentation (discussed in the next section).
Flags: This field is used in fragmentation (discussed in the next section).
Fragmentation offset: This field is used in fragmentation (discussed in the next section).
Time to live: A datagram has a limited lifetime in its travel through an internet. This field was originally designed to hold a timestamp, which was decremented by each visited router. The datagram was discarded when the value became zero.
Protocol: This 8-bit field defines the higher-level protocol that uses the services of the IPv4 layer. An IPv4 datagram can encapsulate data from several higher-level protocols such as TCP, UDP, ICMP, and IGMP. This field specifies the final destination protocol to which the IPv4 datagram is delivered.


Transport layer
SCTP
TCP
UDP
ICMP
IGMP
OSPF
Network layer
Header
The value of the protocol field defines
to which protocol the data belong.

Checksum: First, the value of the checksum field is set to O. Then the entire header is divided into 16-bit sections and added together. The result (sum) is complemented and inserted into the checksum field. The checksum in the IPv4 packet covers only the header, not the data. There are two good reasons for this. First, all higher-level protocols that encapsulate data in the IPv4 datagram have a checksum field that covers the whole packet. Second, the header of the IPv4 packet changes with each visited router, but the data do not.

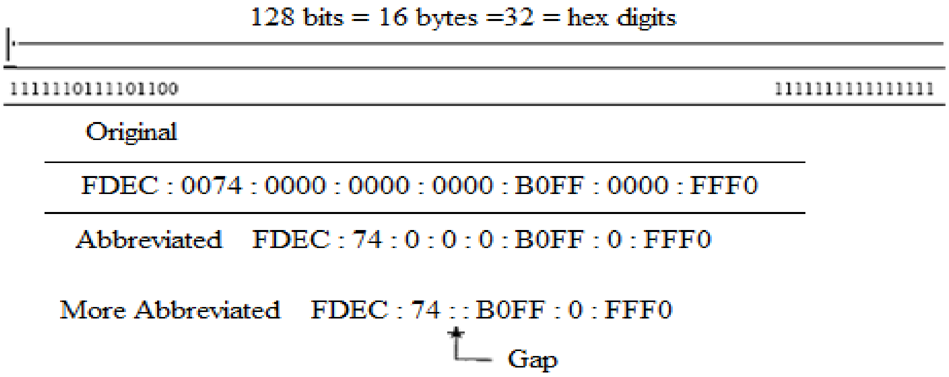
IPv6 ADDRESSES

Address depletion is still a long-term problem for the Internet. This and other problems in the IP protocol itself, such as lack of accommodation for real-time audio and video transmission, and encryption and authentication of data for some applications, have been the motivation for IPv6.

Structure

An IPv6 address consists of 16 bytes (octets); 128 bits long.

Figure 3.2 IPv6 address in binary and hexadecimal notation


128 bits = 16 bytes =32 = hex digits
1111110111101100
1111111111111111
Original
FDEC : 0074 : 0000 : 0000 : 0000 : B0FF : 0000 : FFF0
Abbreviated
FDEC : 74 : 0 : 0 : 0 : B0FF : 0 : FFF0
More Abbreviated
FDEC : 74 : : B0FF : 0 : FFF0
Gap

Hexadecimal Colon Notation: To make addresses more readable, IPv6 specifies hexadecimal colon notation. In this notation, 128 bits is divided into eight sections, each 2 bytes in length. The address consists of 32 hexadecimal digits, with every four digits separated by a colon.

Address Space: IPv6 has a much larger address space; 2^{128} addresses are available. The designers of IPv6 divided the address into several categories.

IPv6 Protocol

IPv4 has some deficiencies (listed below) that make it unsuitable for the fast-growing Internet.

- Despite all short-term solutions, such as subnetting, classless addressing, and NAT, address depletion is still a long-term problem in the Internet.
- The Internet must accommodate real-time audio and video transmission. This type of transmission requires minimum delay strategies and reservation of resources not provided in the IPv4 design.
- The Internet must accommodate encryption and authentication of data for some applications. No encryption or authentication is provided by IPv4.

To overcome these deficiencies, Internetworking Protocol version 6 (IPv6) was proposed and is now a standard.

Advantages of IPv6 over IPv4

The next-generation IP, or IPv6, has some advantages over IPv4 that can be summarized as follows:

Larger address space: An IPv6 address is 128 bits long.

Better header format: IPv6 uses a new header format in which options are separated from the base header. This simplifies and speeds up the routing process.

New options: IPv6 has new options to allow for additional functionalities.

Allowance for extension: IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.

Support for resource allocation: Mechanism to enable the source to request special handling of the packet which is used to support traffic such as real-time audio and video.

Support for more security: The encryption and authentication options in IPv6 provide confidentiality and integrity of the packet.

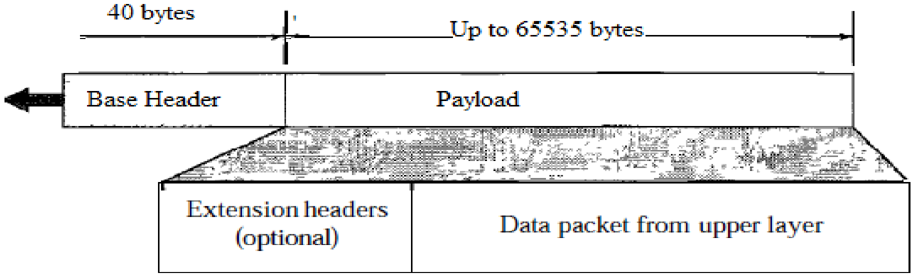
Packet Format

Each packet is composed of a mandatory base header followed by the payload. The payload consists of two parts: optional extension headers and data from an upper layer. The base header occupies 40 bytes, whereas the extension headers and data from the upper layer contain up to 65,535 bytes of information.

Base Header

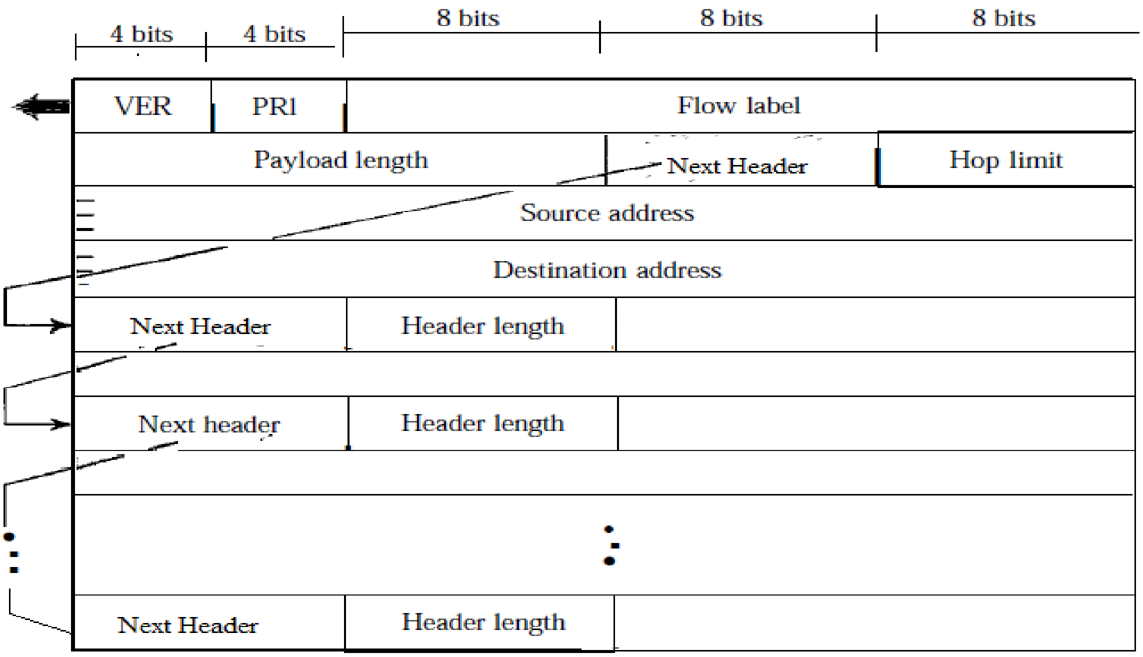
Figure shows the base header with its eight fields. These fields are as follows:

Figure 20.15 *IPv6 datagram header and payload*



Version: This 4-bit field defines the version number of the IP. For IPv6, the value is 6.
Priority: The 4-bit priority field defines the priority of the packet with respect to traffic congestion. We will discuss this field later.
Flow label. The flow label is a 3-byte (24-bit) field that is designed to provide special handling for a particular flow of data. We will discuss this field later.
Payload length: The 2-byte payload length field defines the length of the IP datagram excluding the base header.

Figure 20.16 *Format ofan IPv6 datagram*



Next header: The next header is an 8-bit field defining the header that follows the base header in the datagram. The next header is either one of the optional extension headers used by IP or the header of an encapsulated packet such as UDP or TCP.

Hop limit: This 8-bit hop limit field serves the same purpose as the TIL field in IPv4.

Source address: The source address field is a 16-byte (128-bit) Internet address that identifies the original source of the datagram.

Destination address: The destination address field is a 16-byte (128-bit) Internet address that usually identifies the final destination of the datagram. However, if source routing is used, this field contains the address of the next router.

Priority: The priority field of the IPv6 packet defines the priority of each packet with respect to other packets from the same source. IPv6 divides traffic into two broad categories: congestion-controlled and non congestion-controlled.

Flow Label: A sequence of packets, sent from a particular source to a particular destination that needs special handling by routers is called a *flow* of packets. The combination of the source address and the value of the *flow label* uniquely define a flow of packets.

To a router, a flow is a sequence of packets that share the same characteristics, such as traveling the same path, using the same resources, having the same kind of security, and so on. A router that supports the handling of flow labels has a flow label table. To allow the effective use of flow labels, three rules have been defined:

1. The flow label is assigned to a packet by the source host. The label is a random number between 1 and 224 - 1. A source must not reuse a flow label for a new flow while the existing flow is still active.

2. If a host does not support the flow label, it sets this field to zero. If a router does not support the flow label, it simply ignores it.

3. All packets belonging to the same flow have the same source, same destination, same priority, and same options.

Fragmentation: In IPv6, only the original source can fragment. A source must use a path MTU discovery technique to find the smallest MTU supported by any network on the path. The source then fragments using this knowledge.

Authentication: The authentication extension header has a dual purpose: it validates the message sender and ensures the integrity of data.

TRANSITION FROM IPv4 TO IPv6

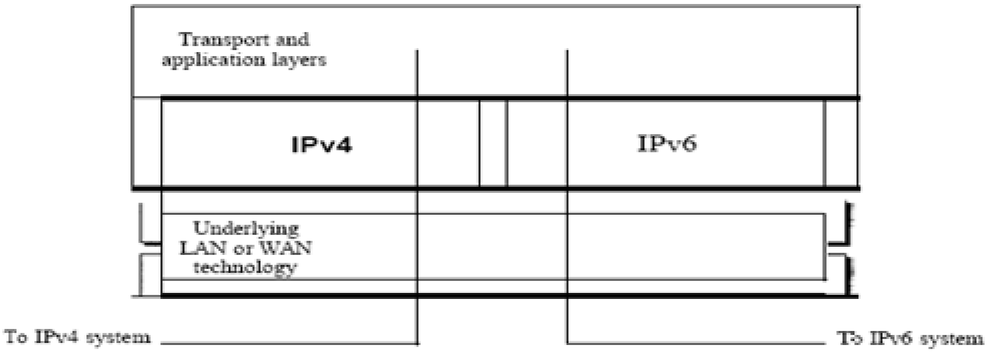
Because of the huge number of systems on the Internet, the transition from IPv4 to IPv6 cannot happen suddenly. It takes a considerable amount of time before every system in the Internet can move from IPv4 to IPv6. The transition must be smooth to prevent any problems between IPv4 and IPv6 systems. There are three types of transition strategies:

1. Dual Stack 2.Tunneling 3.Header Translation

Dual Stack

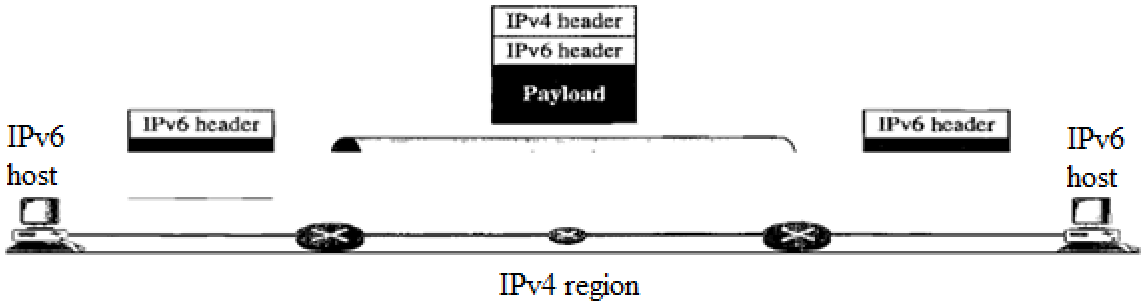
It is recommended that all hosts, before migrating completely to version 6, have a **dual** stack of protocols. In other words, a station must run IPv4 and IPv6 simultaneously until all the Internet uses IPv6. See Figure for the layout of a dual-stack configuration.

To determine which version to use when sending a packet to a destination, the source host queries the DNS. If the DNS returns an IPv4 address, the source host sends an IPv4 packet. If the DNS returns an IPv6 address, the source host sends an IPv6 packet.


Transport and application layers
IPv4
IPv6
Underlying LAN or WAN technology
To IPv4 system
To IPv6 system

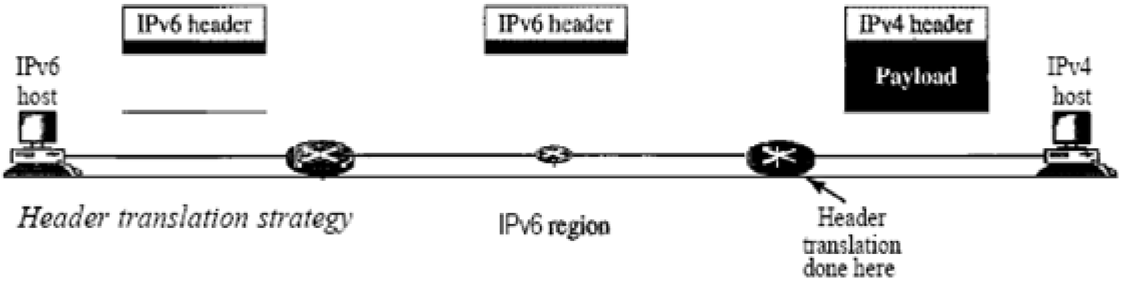
Tunneling

Tunneling is a strategy used when two computers using IPv6 want to communicate with each other and the packet must pass through a region that uses IPv4. To pass through this region, the packet must have an IPv4 address. So the IPv6 packet is encapsulated in an IPv4 packet when it enters the region, and it leaves its capsule when it emerges at the other end.



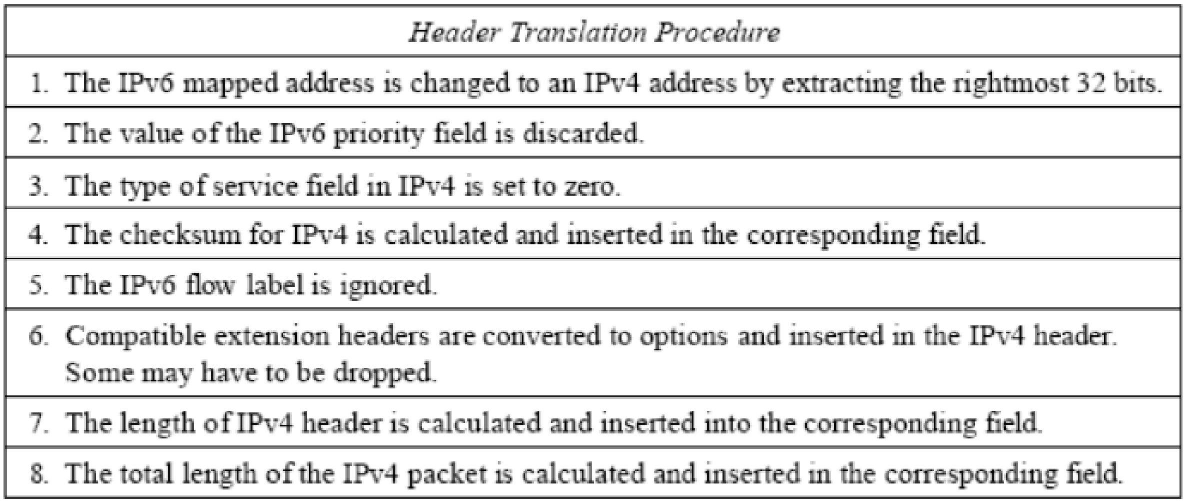
Header Translation

Header translation is necessary when the majority of the Internet has moved to IPv6 but some systems still use IPv4. The sender wants to use IPv6, but the receiver does not understand IPv6. Tunneling does not work in this situation because the packet must be in the IPv4 format to be understood by the receiver. In this case, the header format must be totally changed through header translation. The header of the IPv6 packet is converted to an IPv4.



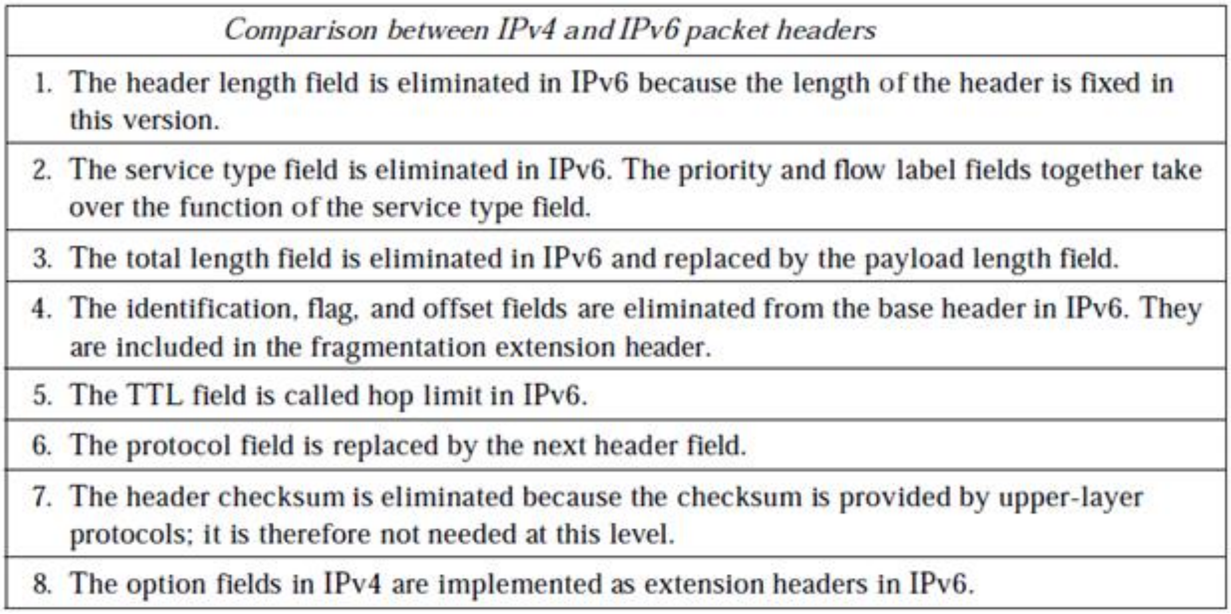
Header translation strategy

Table 20.11 *Header translation*



Header Translation Procedure
1. The IPv6 mapped address is changed to an IPv4 address by extracting the rightmost 32 bits.
2. The value of the IPv6 priority field is discarded.
3. The type of service field in IPv4 is set to zero.
4. The checksum for IPv4 is calculated and inserted in the corresponding field.
5. The IPv6 flow label is ignored.
6. Compatible extension headers are converted to options and inserted in the IPv4 header. Some may have to be dropped.
7. The length of IPv4 header is calculated and inserted into the corresponding field.
8. The total length of the IPv4 packet is calculated and inserted in the corresponding field.

Header translation uses the mapped address to translate an IPv6 address to an IPv4 address. Table 20.11 lists some rules used in transforming an IPv6 packet header to an IPv4 packet header.



Comparison between IPv4 and IPv6 packet headers
1. The header length field is eliminated in IPv6 because the length of the header is fixed in this version.
2. The service type field is eliminated in IPv6. The priority and flow label fields together take over the function of the service type field.
3. The total length field is eliminated in IPv6 and replaced by the payload length field.
4. The identification, flag, and offset fields are eliminated from the base header in IPv6. They are included in the fragmentation extension header.
5. The TTL field is called hop limit in IPv6.
6. The protocol field is replaced by the next header field.
7. The header checksum is eliminated because the checksum is provided by upper-layer protocols; it is therefore not needed at this level.
8. The option fields in IPv4 are implemented as extension headers in IPv6.

Mapping Logical to Physical Address: ARP

Anytime a host or a router has an IP datagram to send to another host or router, it has the logical (IP) address of the receiver. The logical (IP) address is obtained from the DNS if the sender is the host or it is found in a routing table if the sender is a router. But the IP datagram must be encapsulated in a frame to be able to pass through the physical network.

This means that the sender needs the physical address the receiver. The host or the router sends an ARP query packet. The packet includes the physical and IP addresses of the sender and the IP address of the receiver. Because the sender does not know the physical address of the receiver, the query is broadcast over the network.

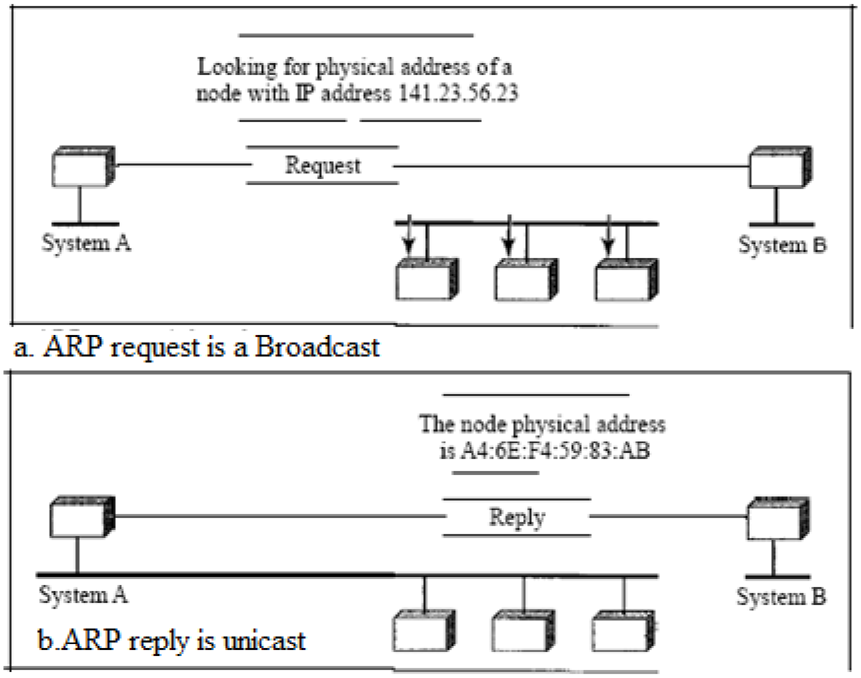
In Figure 21.1a, the system on the left (A) has a packet that needs to be delivered to another system (B) with IP address 141.23.56.23. System A needs to pass the packet to its data link layer for the actual delivery, but it does not know the physical address of the recipient. It uses the services of ARP by asking the ARP protocol to send a broadcast ARP request packet to ask for the physical address of a system with an IF address of 141.23.56.23.

This packet is received by every system on the physical network, but only system B will answer it, as shown in Figure 21.1 b. System B sends an ARP reply packet that includes its physical address. Now system A can send all the packets it has for this destination by using the physical address it received.

Cache Memory: Using ARP is inefficient if system A needs to broadcast an ARP request for each IP packet it needs to send to system B. It could have broadcast the IP packet itself. ARP can be useful if the ARP reply is cached (kept in cache memory for a while) because a normally sends several packets to the same destination.

Packet Format of ARP: Figure 21.2 shows the format of an ARP packet.

Figure 3.8 ARP Operation


Looking for physical address of a node with IP address 141.23.56.23
Request
System A
System B
The node physical address is A4:6E:F4:59:83:AB
Reply
System A
System B

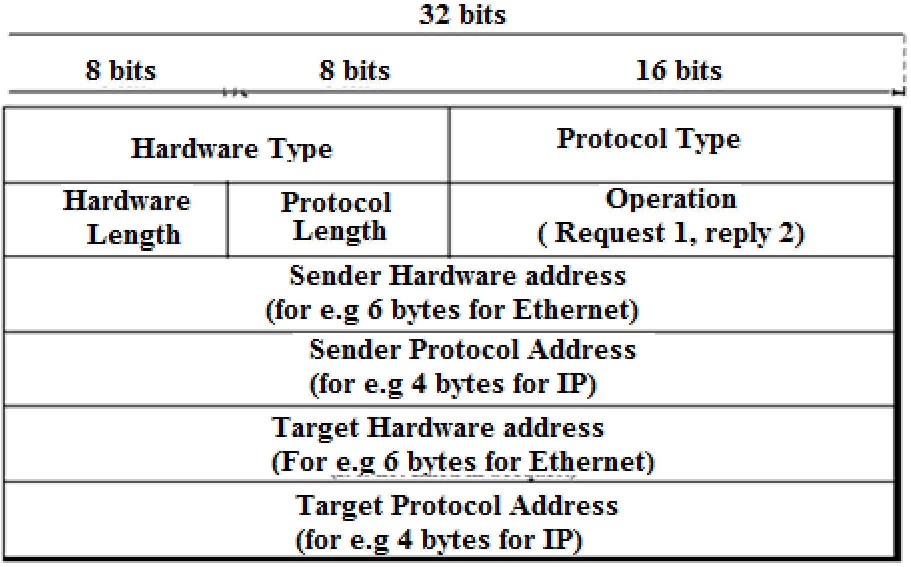
a. ARP request is a Broadcast

b.ARP reply is unicast

Hardware type: This is a 16-bit field defining the type of the network on which ARP is running. Each LAN has been assigned an integer based on its type. **Protocol type:** This is a 16-bit field defining the protocol.

Hardware length: This is an 8-bit field defining the length of the physical address in bytes. For example, for Ethernet the value is 6. **Protocol length:** This is an 8-bit field defining the length of the logical address in bytes. For example, for the IPv4 protocol the value is 4.

ARP Packet

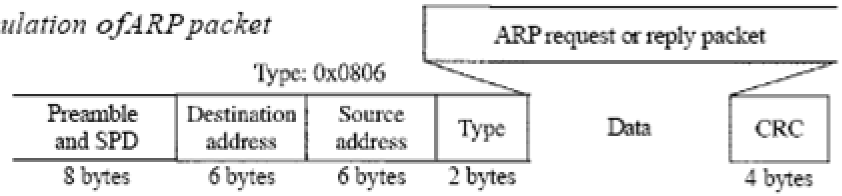


32 bits

8 bits	8 bits	16 bits
Hardware Type		Protocol Type
Hardware Length	Protocol Length	Operation (Request 1, reply 2)
Sender Hardware address (for e.g 6 bytes for Ethernet)		
Sender Protocol Address (for e.g 4 bytes for IP)		
Target Hardware address (For e.g 6 bytes for Ethernet)		
Target Protocol Address (for e.g 4 bytes for IP)		

Encapsulation: An ARP packet is encapsulated directly into a data link frame. For example, in Figure an ARP packet is encapsulated in an Ethernet frame. Note that the type field indicates that the data carried by the frame are an ARP packet.

Encapsulation ofARP packet



Operation

Let us see how ARP functions on a typical internet. First we describe the steps involved.

1. The sender knows the IP address of the target. We will see how the sender obtains this shortly.
2. IP asks ARP to create an ARP request message, filling in the sender physical address, the sender IP address, and the target IP address. The target physical address field is filled with Os.
3. The message is passed to the data link layer where it is encapsulated in a frame by using the physical address of the sender as the source address and the physical broadcast address as the destination address.
4. Every host or router receives the frame. Because the frame contains a broadcast destination address, all stations remove the message and pass it to ARP. All machines except the one targeted drop the packet. The target machine recognizes its IP address.
5. The target machine replies with an ARP reply message that contains its physical address. The message is unicast.
6. The sender receives the reply message. It now knows the physical address of the target machine.
7. The IP datagram, which carries data for the target machine, is now encapsulated in a frame and is unicast to the destination.

Physical to Logical Mapping: RARP, BOOTP, and DHCP

There are occasions in which a host knows its physical address, but needs to know its logical address. This may happen in two cases:

1. A diskless station is just booted. The station can find its physical address by checking its interface, but it does not know its IP address.

2. An organization does not have enough IP addresses to assign to each station; it needs to assign IP addresses on demand. The station can send its physical address and ask for a short time lease.

RARP: Reverse Address Resolution Protocol (RARP) finds the logical address for a machine that knows only its physical address. Each host or router is assigned one or more logical (IP) addresses, which are unique and independent of the physical (hardware) address of the machine. To create an IP datagram, a host or a router needs to know its own IP address or addresses. The IP address of a machine is usually read from its configuration file stored on a disk file.

However, a diskless machine is usually booted from ROM, which has minimum booting information. The ROM is installed by the manufacturer. It cannot include the IP address because the IP addresses on a network are assigned by the network administrator.

There is a serious problem with RARP: Broadcasting is done at the data link layer. If an administrator has several networks or several subnets, it Need to assign a RARP server for each network or subnet. This is the reason that RARP is almost obsolete. BOOTP and DHCP are replacing RARP.

BOOTP: The Bootstrap Protocol (BOOTP) is a client/server protocol designed to provide physical address to logical address mapping. BOOTP is an application layer protocol. The administrator may put the client and the server on the same network or on different networks. BOOTP messages are encapsulated in a UDP packet, and the UDP packet itself is encapsulated in an IP packet.

The reader may ask how a client can send an IP datagram when it knows neither its own IP address (the source address) nor the server's IP address (the destination address). The client simply uses all as the source addresses and as the destination address.

One of the advantages of BOOTP over RARP is that the client and server are application-layer processes. As in other application-layer processes, a client can be in one network and the server in another, separated by several other networks. A broadcast IP datagram cannot pass through any router. To solve the problem, there is a need for an intermediary.

DHCP: The Dynamic Host Configuration Protocol (DHCP) has been devised to provide static and dynamic address allocation that can be manual or automatic.

Dynamic Address Allocation DHCP has a second database with a pool of available IP addresses. This second database makes DHCP dynamic. When a DHCP client requests a temporary IP address, the DHCP server goes to the pool of available (unused) IP addresses and assigns an IP address for a negotiable period of time.

The dynamic aspect of DHCP is needed when a host moves from network to network or is connected and disconnected from a network. DHCP provides temporary IP addresses for a limited time.

DHCP allows both manual and automatic configurations. Static addresses are created manually dynamic addresses are created automatically.

ICMP

The IP protocol has no error-reporting or error-correcting mechanism. The IP protocol also lacks a mechanism for host and management queries. A host sometimes needs to determine if a router or another host is alive. And sometimes a network administrator needs information from another host or router.

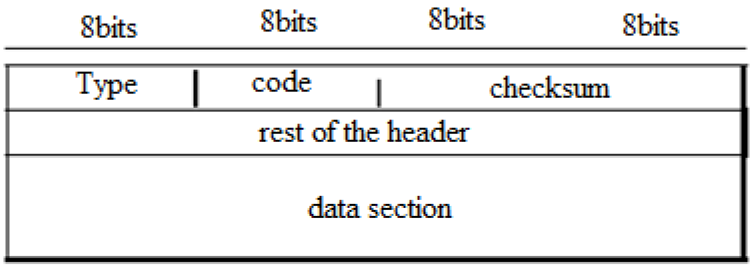
The Internet Control Message Protocol (ICMP) has been designed to compensate for the above two deficiencies. It is a companion to the IP protoco1.

Message Format

An ICMP message has an 8-byte header and a variable-size data section. Although the general format of the header is different for each message type, the first 4 bytes are common to all. As Figure 21.8 shows, the first field, ICMP type, defines the type of the message.

The code field specifies the reason for the particular message type. The last common field is the checksum field. The rest of the header is specific for each message type.

Figure General format of ICMP messages



The data section in error messages carries information for finding the original packet that had the error. In query messages, the data section carries extra information based on the type of the query.

Types of Messages

ICMP messages are divided into two broad categories: **error-reporting messages and query messages.**

The error-reporting messages report problems that a router or a host (destination) may encounter when it processes an IP packet.

The query messages, which occur in pairs, help a host or a network manager get specific information from a router or another host.

Error Reporting Messages

One of the main responsibilities of ICMP is to report errors. Although technology has produced increasingly reliable transmission media, errors still exist and must be handled. IP is an unreliable protocol. This means that error checking and error control are not a concern of IP. ICMP was designed, in part, to compensate for this shortcoming.

ICMP always reports error messages to the original source. Five types of errors are handled: destination unreachable, source quench, time exceeded, parameter problems, and redirection. The following are important points about ICMP error messages:

- No ICMP error message will be generated in response to a datagram carrying an ICMP error message.
- No ICMP error message will be generated for a fragmented datagram that is not the first fragment.

- No IeMP error message will be generated for a datagram having a multicast address.
- No ICMP error message will be generated for a datagram having a special address such as 127.0.0.0 or 0.0.0.0.

Note that all error messages contain a data section that includes the IP header of the original datagram plus the first 8 bytes of data in that datagram. The original datagram header is added to give the original source, which receives the error message, information about the datagram itself.

There are five types of reporting messages of ICMP.

1. Destination Unreachable: When a router cannot route a datagram or a host cannot deliver a datagram, the datagram is discarded and the router or the host sends a destination-unreachable message back to the source host that initiated the datagram. Note that destination-unreachable messages can be created by either a router or the destination host.

2. Source Quench: The source-quench message in ICMP was designed to add a kind of flow control to the IP. When a router or host discards a datagram due to congestion, it sends a source-quench message to the sender of the datagram.

This message has two purposes. First, it informs the source that the datagram has been discarded. Second, it warns the source that there is congestion somewhere in the path and that the source should slow down (quench) the sending process.

3. Time Exceeded: When a datagram visits a router, the value of this field is decremented by 1. When the time-to-live value reaches 0, after decrementing, the router discards the datagram. However, when the datagram is discarded, a time-exceeded message must be sent by the router to the original source. Second, a time-exceeded message is also generated when not all fragments that make up a message arrive at the destination host within a certain time limit.

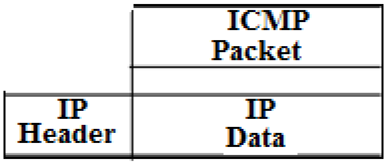
4. Parameter Problem: Any ambiguity in the header part of a datagram can create serious problems as the datagram travels through the Internet. If a router or the destination host discovers an ambiguous or missing value in any field of the datagram, it discards the datagram and sends a parameter-problem message back to the source.

5. Redirection: When a router needs to send a packet destined for another network, it must know the IP address of the next appropriate router. The same is true if the sender is a host. Both routers and hosts, then, must have a routing table to find the address of the router or the next router. Routers take parts in the routing update process are supposed to be updated constantly. Routing is dynamic.

Query Messages of ICMP

In addition to error reporting, ICMP can diagnose some network problems. This is accomplished through the query messages, a group of four different pairs of messages. In ICMP query message, a node sends a message that is answered in a specific format by the destination node. A query is encapsulated in an IP packet, which in turn is encapsulated in a data link layer frame. There four types of Query Messages in ICMP.

1. Echo Request and Reply: The echo-request and echo-reply messages are designed for diagnostic purposes. Network managers and users utilize this pair of messages to identify network problems.



The combination of echo-request and echo-reply messages determines whether two systems can communicate with each other. The echo-request and echo-reply messages can be used to determine if there is communication at the IP level. Because ICMP messages are encapsulated in IP datagrams, the receipt of an echo-reply message by the machine that sent the echo request is proof that the IP protocols in the sender and receiver are communicating with each other using the IP datagram.

2. Timestamp Request and Reply: Two machines (hosts or routers) can use the timestamp request and timestamp reply messages to determine the round-trip time needed for an IP datagram to travel between them. It can also be used to synchronize the clocks in two machines.

3. Address-Mask Request and Reply: A host may know its IP address, but it may not know the corresponding mask. To obtain mask, a host sends an address-mask-request message to a router on the LAN. If the host knows the address of the router, it sends the request directly to the router. If it does not know, it broadcasts the message.

4. Router Solicitation and Advertisement

A router can also periodically send router-advertisement messages even if no host has solicited. Note that when a router sends out an advertisement, it announces not only its own presence but also the presence of all routers on the network of which it is aware.

Debugging Tools

There are several tools that can be used in the Internet for debugging. We can determine the viability of a host or router. We can trace the route of a packet. We introduce two tools that use ICMP for debugging: *ping* and *traceroute*. We will introduce more tools in future chapters after we have discussed the corresponding protocols.

Ping:* P*ing is a program to find if a host is alive and responding. We use *ping* here to see how it uses ICMP packets. Note that *ping* can calculate the round-trip time. It inserts the sending time in the data section of the message. When the packet arrives, it subtracts the arrival time from the departure time to get the round-trip time (RTT). *Ping* defines the number of packets sent, the number of packets received, the total time, and the RTT minimum, maximum, and average. Some systems may print more information.

Trace-route Program: It can be used to trace the route of a packet from the source to the destination. We have seen an application of the *traceroute* program to simulate the loose source route and strict source route options of an IP datagram. The program elegantly uses two ICMP messages; time exceeded and destination unreachable, to find the route of a packet. This is a program at the application level that uses the services of UDP.

Direct Versus Indirect Delivery

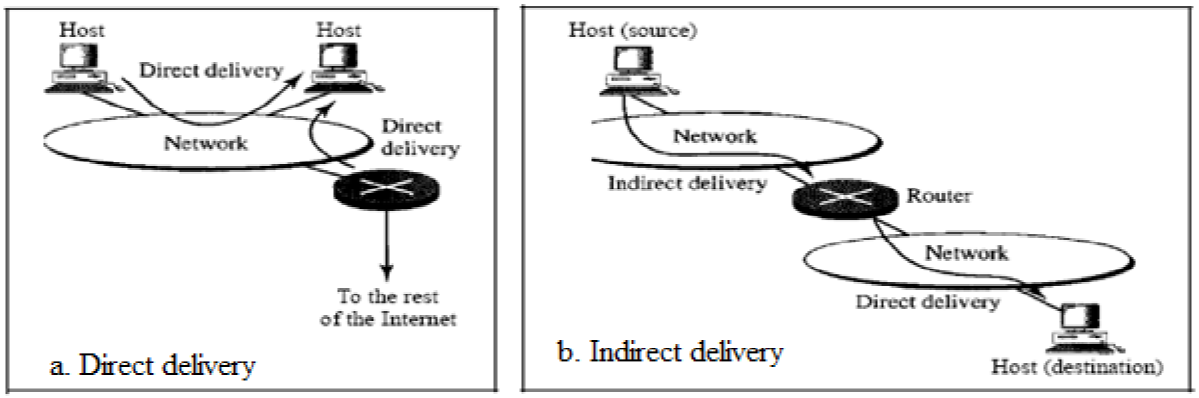
The delivery of a packet to its final destination is accomplished by using two different methods of delivery, direct and indirect.

Direct Delivery: In a direct delivery, the final destination of the packet is a host connected to the same physical network as the deliverer. Direct delivery occurs when the source and destination of the packet are located on the same physical network or when the delivery is between the last router and the destination host. The sender can easily determine if the delivery is direct. It can extract the network address of the destination (using the mask) and compare this address with the addresses of the networks to which it is connected. If a match is found, the delivery is direct.

Indirect Delivery

If the destination host is not on the same network as the deliverer, the packet is delivered indirectly. In an indirect delivery, the packet goes from router to router until it reaches the one connected to the same physical network as its final destination. Note that a delivery always involves one direct delivery but zero or more indirect deliveries. Note also that the last delivery is always a direct delivery.

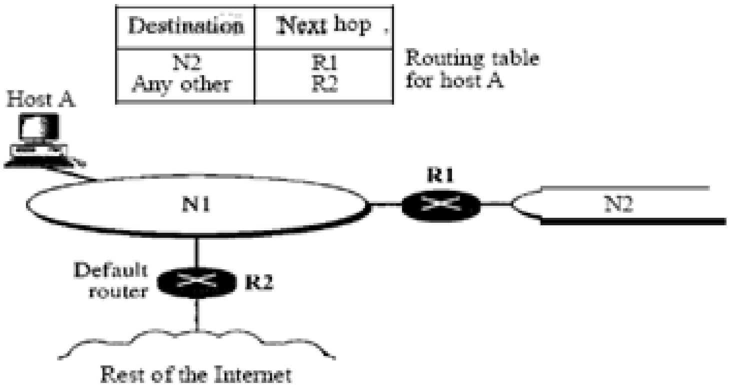
Figure Direct and Indirect Delivery



FORWARDING

Forwarding means to place the packet in its route to its destination. Forwarding requires a host or a router to have a routing table. When a host has a packet to send or when a router has received a packet to be forwarded, it looks at this table to find the route to the final destination. However, this simple solution is impossible today in an internetwork such as the Internet because the number of entries needed in the routing table would make table lookups inefficient. Several techniques can make the size of the routing table manageable and also handle issues such as security.

Figure 22.4 *Default method*



Destination	Next hop
N2	R1
Any other	R2

Next-Hop Method versus Route Method: One technique to reduce the contents of a routing table is called the next-hop method. In this technique, the routing table holds only the address of the next hop instead of information about the complete route (route method). The entries of a routing table must be consistent with one another.

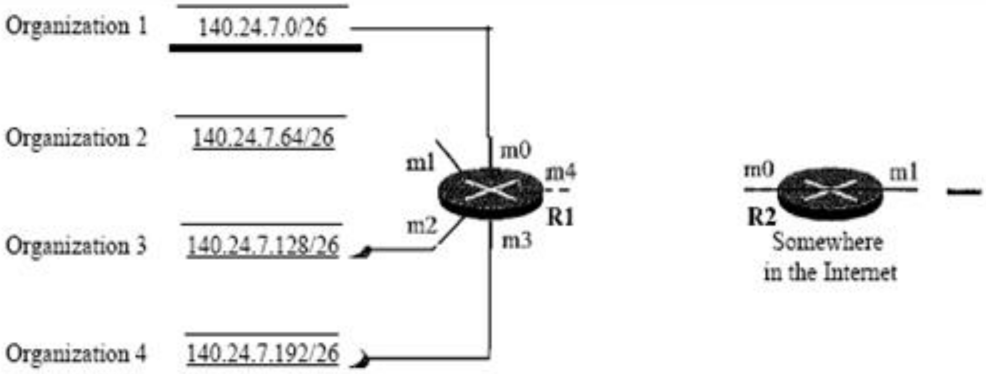
Network-Specific Method versus Host-Specific Method: A second technique to reduce the routing table and simplify the searching process is called the network-specific method. Here, instead of having an entry for every destination host connected to the same physical network.

Default Method: Another technique to simplify routing is called the default method. Host A is connected to a network with two routers. Router Rl routes the packets to hosts connected to network N2. However, for the rest of the Internet, router R2 is used. So instead of listing all networks in the entire Internet, host A can just have one entry called the *default* (normally defined as network address 0.0.0.0).

Forwarding Process: In classless addressing, the routing table needs to have one row of information for each block involved. The table needs to be searched based on the network address (first address in the block). Unfortunately, the destination address in the packet gives no clue about the network address. To solve the problem, we need to include the mask *(In)* in the table; we need to have an extra column that includes the mask for the corresponding block.

Address Aggregation: When we use classless addressing, it is likely that the number of routing table entries will increase. This is so because the intent of classless addressing is to divide up the whole address space into manageable blocks. The increased size of the table results in an increase in the amount of time needed to search the table. To alleviate the problem, the idea of address aggregation was designed. In Figure 22.7 we have two routers. Router Rl is connected to networks of four organizations that each use 64 addresses. Router R2 is somewhere far from Rl. Router Rl has a longer routing table because each packet must be correctly routed to the appropriate organization. Router R2, on the other hand, can have a very small routing table.

Figure 22.7 *Address aggregation*

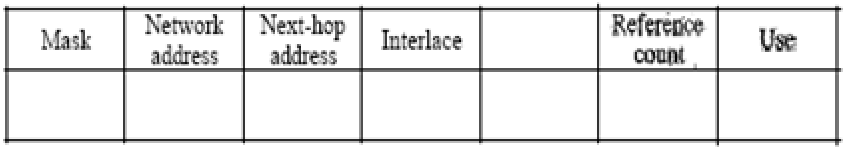


Hierarchical Routing: To solve the problem of gigantic routing tables, we can create a sense of hierarchy in the routing tables. National ISPs are divided into regional ISPs, and regional ISPs are divided into local ISPs. If the routing table has a sense of hierarchy like the Internet architecture, the routing table can decrease in size.

Geographical Routing: To decrease the size of the routing table even further, we need to extend hierarchical routing to include geographical routing. We must divide the entire address space into a few large blocks.

Routing Table: A host or a router has a routing table with an entry for each destination, or a combination of destinations, to route IP packets. The routing table can be either static or dynamic.

Routing table Format: As mentioned previously, a routing table for classless addressing has a minimum of four columns. However, some of today's routers have even more columns. We should be aware that the number of columns is vendor-dependent, and not all columns can be found in all routers.



Mask	Network address	Next-hop address	Interlace		Reference count	Use

Mask: This field defines the mask applied for the entry.

Network address: It defines the network address to which the packet is finally delivered. In the case of host-specific routing, this field defines the address of the destination host.

Next-hop address: It defines the address of the next-hop router to which the packet is delivered.

D Interface: This field shows the name of the interface.

Flags: This field defines up to five flags are U (up), G (gateway), H (host-specific), D (added by redirection), and M (modified by redirection).

Static routing table: It contains information entered manually. The administrator enters the route for each destination into the table. When a table is created, it cannot update automatically when there is a change in the Internet. The table must be manually altered by the administrator.

A static routing table can be used in a small internet that does not change very often, or in an experimental internet for troubleshooting. It is poor strategy to use a static routing table in a big internet such as the Internet.

Dynamic Routing Table: A dynamic routing table is updated periodically by using one of the dynamic routing protocols such as RIP, OSPF, or BGP. Whenever there is a change in the Internet, such as a shutdown of a router or breaking of a link, the dynamic routing protocols update all the tables in the routers automatically. The routers in a big internet such as the Internet need to be updated dynamically for efficient delivery of the IP packets.

UNICAST ROUTING PROTOCOLS

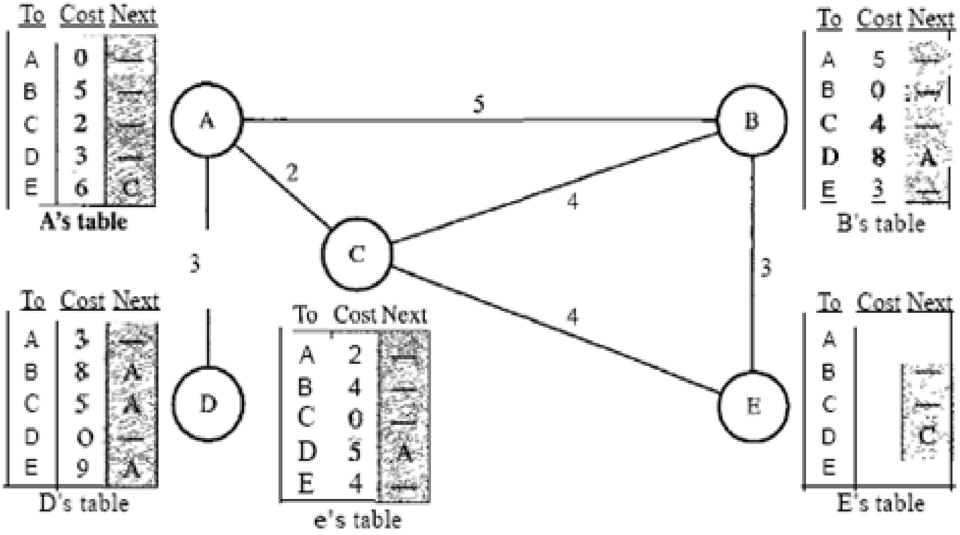
Routing protocols have been created in response to the demand for dynamic routing tables. A routing protocol is a combination of rules and procedures that let routers in the internet inform each other of changes. It allows routers to share whatever they know the internet or their neighborhood. The routing protocols also include procedures for combining information received from other routers.

Distance Vector Routing

In distance vector routing, the least-cost route between any two nodes is the route with minimum distance. In this protocol, as the name implies, each node maintains a vector (table) of minimum distances to every node.

The table at each node also guides the packets to the desired node by showing the next stop in the route (next-hop routing). **In** distance vector routing, each node shares its routing table with its immediate neighbors periodically and when there is a change. In Figure 22.14, we show a system of five nodes with their corresponding tables.

Figure 22.14 *Distance vector routing tables*



To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

A's table

To	Cost	Next
A	5	—
B	0	—
C	4	—
D	8	A
E	3	—

B's table

To	Cost	Next
A	3	—
B	8	A
C	5	A
D	0	—
E	9	A

D's table

To	Cost	Next
A	2	—
B	4	—
C	0	—
D	5	A
E	4	—

e's table

To	Cost	Next
A		
B		—
C		—
D		C
E		

E's table

Updating: When a node receives a two-column table from a neighbor, it needs to update its routing table. Updating takes three steps:

1. The receiving node needs to add the cost between itself and the sending node to each value in the second column. The logic is clear. If node C claims that its distance to a destination is *x* mi, and the distance between A and C is *y* mi, then the distance between A and that destination, via C, is *x* + *y* mi.

2. The receiving node needs to add the name of the sending node to each row as the third column if the receiving node uses information from any row. The sending node is the next node in the route.

3. The receiving node needs to compare each row of its old table with the corresponding row of the modified version of the received table.

a. If the next-node entry is different, the receiving node chooses the row with the smaller cost. If there is a tie, the old one is kept.

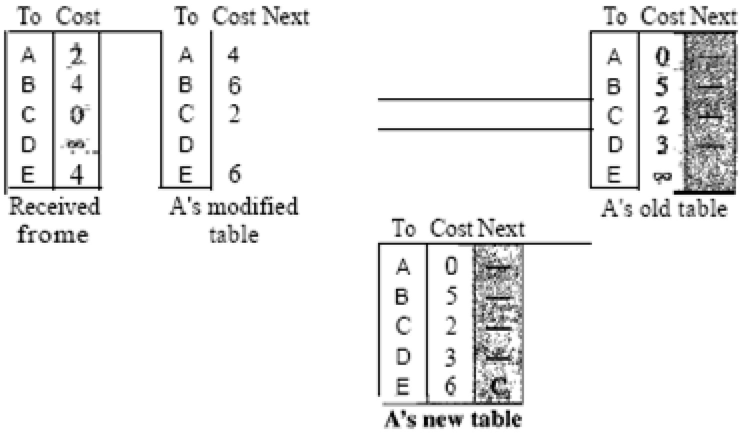
b. If the next-node entry is the same, the receiving node chooses the new row. For example, suppose node C has previously advertised a route to node X with distance

3. Suppose that now there is no path between C and X; node C now advertises route with a distance of infinity. Node A must not ignore this value even though its old entry is smaller. The old route does not exist anymore. The new route has a distance of infinity.

Figure 22.16 shows how node A updates its routing table after receiving the partial table from node C. There are several points we need to emphasize here. First, as we know from mathematics, when we add any number to infinity, the result is still infinity. Second, the modified table shows how to reach A from A via C. If A needs to reach itself via C, it needs to go to C and come back, a distance of 4. Third, the only benefit from this updating of node A is the last entry, how to reach E.

Each node can update its table by using the tables received from other nodes. In a short time, if there is no change in the network itself, such as a failure in a link, each node reaches a stable condition in which the contents of its table remains the same.

Figure 22.16 *Updating in distance vector routing*



Received frome

To	Cost
A	2
B	4
C	0
D	∞
E	4

A's modified table

To	Cost	Next
A	4	
B	6	
C	2	
D		
E	6	

A's old table

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	∞	

A's new table

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

When to Share

Periodic Update A node sends its routing table, normally every 30 s, in a periodic update. The period depends on the protocol that is using distance vector routing. This is called a triggered update. The change can result from the following.

1. A node receives a table from a neighbor, resulting in changes in its own table after updating.
2. A node detects some failure in the neighboring links which results in a distance change to infinity.

The Routing Information Protocol (RIP)

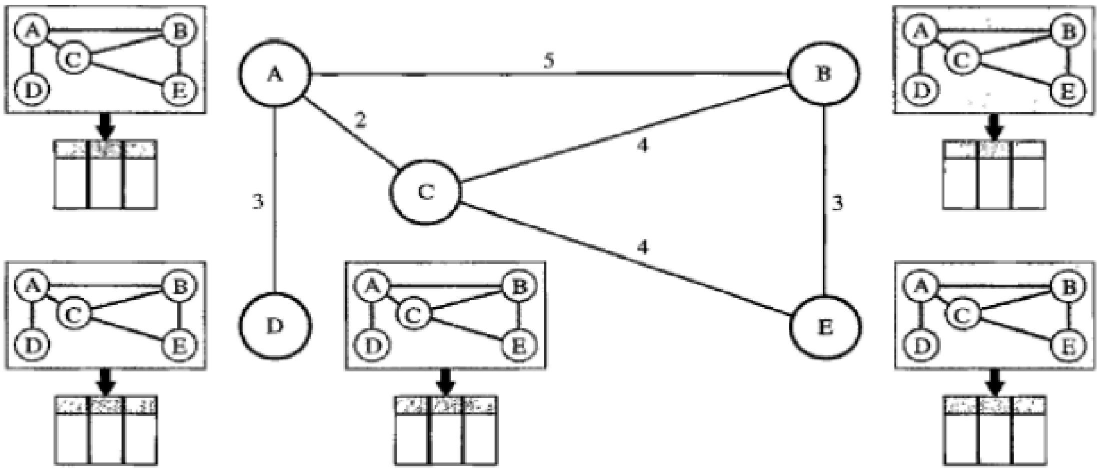
It is an intra-domain routing protocol used inside an autonomous system. It is a very simple protocol based on distance vector routing. RIP implements distance vector routing directly with some considerations:

1. In an autonomous system, we are dealing with routers and networks (links). The routers have routing tables; networks do not.
2. The destination in a routing table is a network, which means the first column defines a network address.
3. The metric used by RIP is very simple; the distance is defined as the number of links to reach the destination. For this reason, the metric in RIP is called a hop count.
4. Infinity is defined as 16, which means that any route in an autonomous system using RIP cannot have more than 15 hops.
5. The next-node column defines the address of the router to which the packet is to be sent to reach its destination.

Link State Routing Protocol

Link state routing has a different philosophy from that of distance vector routing. In link state routing, if each node in the domain has the entire topology of the domain the list of nodes and links, how they are connected including the type, cost (metric), and condition of the links (up or down)-the node can use Dijkstra's algorithm to build a routing table.

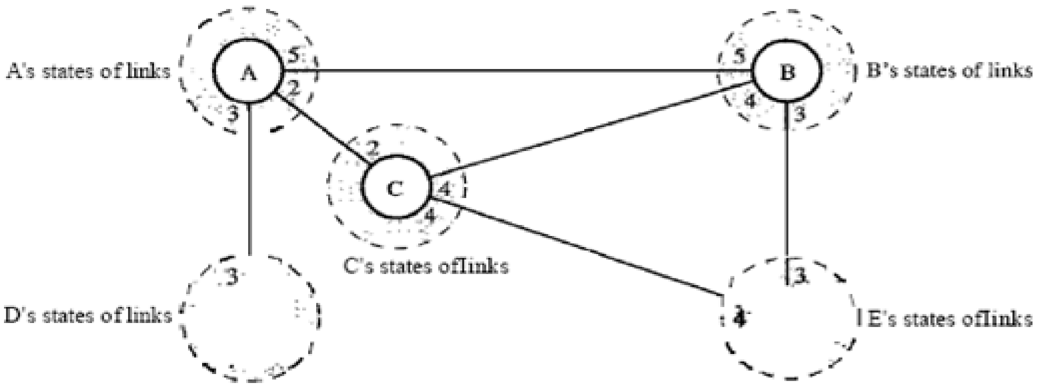
Figure 22.20 *Concept oflink state routing*



The figure shows a simple domain with five nodes. Each node uses the same topology to create a routing table, but the routing table for each node is unique because the calculations are based on different interpretations of the topology. This is analogous to a city map. While each person may have the same map, each needs to take a different route to reach her specific destination.

The topology must be dynamic, representing the latest state of each node and each link. If there are changes in any point in the network (a link is down, for example), the topology must be updated for each node.

Figure 22.21 *Link state knowledge*



Node A knows that it is connected to node B with metric 5, to node C with metric 2, and to node D with metric 3. Node C knows that it is connected to node A with metric 2, to node B with metric 4, and to node E with metric 4. Node D knows that it is connected only to node A with metric 3. And so on. Although there is an overlap in the knowledge, the overlap guarantees the creation of a common topology-a picture of the whole domain for each node.

Building Routing Tables

In link state routing, four sets of actions are required to ensure that each node has the routing table showing the least-cost node to every other node.

1. Creation of the states of the links by each node, called the link state packet (LSP).

2. Dissemination of LSPs to every other router, called **flooding,** in an efficient and reliable way.

3. Formation of a shortest path tree for each node.

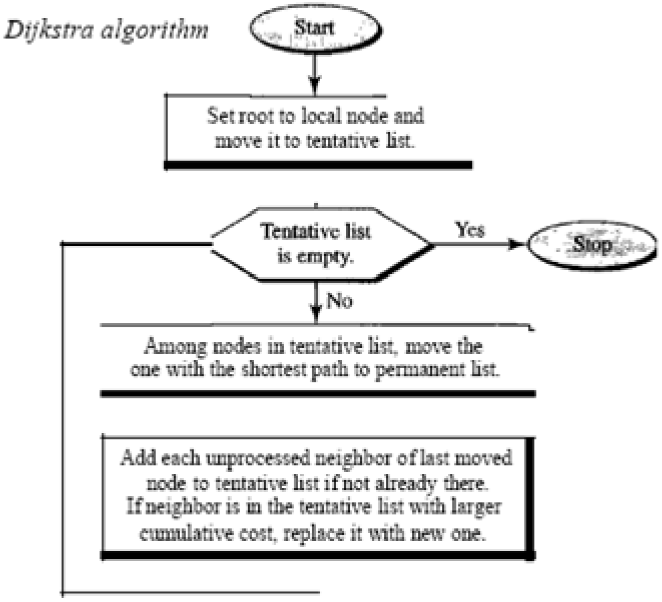
4. Calculation of a routing table based on the shortest path tree.

Creation of Link State Packet (LSP) A link state packet can carry a large amount of information. LSPs are generated on two occasions:

1. When there is a change in the topology of the domain. Triggering of LSP dissemination is the main way of quickly informing any node in the domain to update its topology.

2. on a periodic basis. The period in this case is much longer compared to distance vector routing. As a matter of fact, there is no actual need for this type of LSP dissemination. It is done to ensure that old information is removed from the domain.

Formation of Shortest Path Tree: Dijkstra Algorithm After receiving all LSPs, each node will have a copy of the whole topology. However, the topology is not sufficient to find the shortest path to every other node; a shortest path tree is needed.


Dijkstra algorithm
Start
Set root to local node and move it to tentative list.
Tentative list is empty.
Yes
Stop
No
Among nodes in tentative list, move the one with the shortest path to permanent list.
Add each unprocessed neighbor of last moved node to tentative list if not already there. If neighbor is in the tentative list with larger cumulative cost, replace it with new one.

A tree is a graph of nodes and links; one node is called the root. All other nodes can be reached from the root through only one single route. A shortest path tree is a tree in which the path between the root and every other node is the shortest. What we need for each node is a shortest path tree with that node as the root.

The Dijkstra algorithm creates a shortest path tree from a graph. The algorithm divides the nodes into two sets: tentative and permanent. It finds the neighbors of a current node, makes them tentative, examines them, and if they pass the criteria, makes them permanent. We can informally define the algorithm by using the flowchart.

To find the shortest path in each step, we need the cumulative cost from the root to each node, which is shown next to the node. The following shows the steps. At the end of each step, we show the permanent (filled circles) and the tentative (open circles) nodes and lists with the cumulative costs.

1. We make node A the root of the tree and move it to the tentative list. Our two lists are Permanent list: empty Tentative list: A(O)

2. Node A has the shortest cumulative cost from all nodes in the tentative list. We move A to the permanent list and add all neighbors of A to the tentative list. Our new lists are Permanent list: A(O) Tentative list: B(5), C(2), D(3)

3. Node C has the shortest cumulative cost from all nodes in the tentative list. We move C to the permanent list. Node C has three neighbors, but node A is already processed, which makes the unprocessed neighbors just B and E. Our new lists are Permanent list: A(O), e(2) Tentative list: B(5), 0(3), E(6)

4. Node D has the shortest cumulative cost of all the nodes in the tentative list. We move D to the permanent list. Node D has no unprocessed neighbor to be added to the tentative list. Our new lists are Permanent list: A(O), C(2), 0(3) Tentative list: B(5), E(6)

5. Node B has the shortest cumulative cost of all the nodes in the tentative list. We move B to the permanent list. The cumulative cost to node E, as the neighbor of B, is 8. We keep node E(6) in the tentative list. Our new lists are Permanent list: A(O), B(5), C(2), 0(3) Tentative list: E(6)

6. Node E has the shortest cumulative cost from all nodes in the tentative list. We move E to the permanent list. Node E has no neighbor. Now the tentative list is empty. We stop; our shortest path tree is ready.

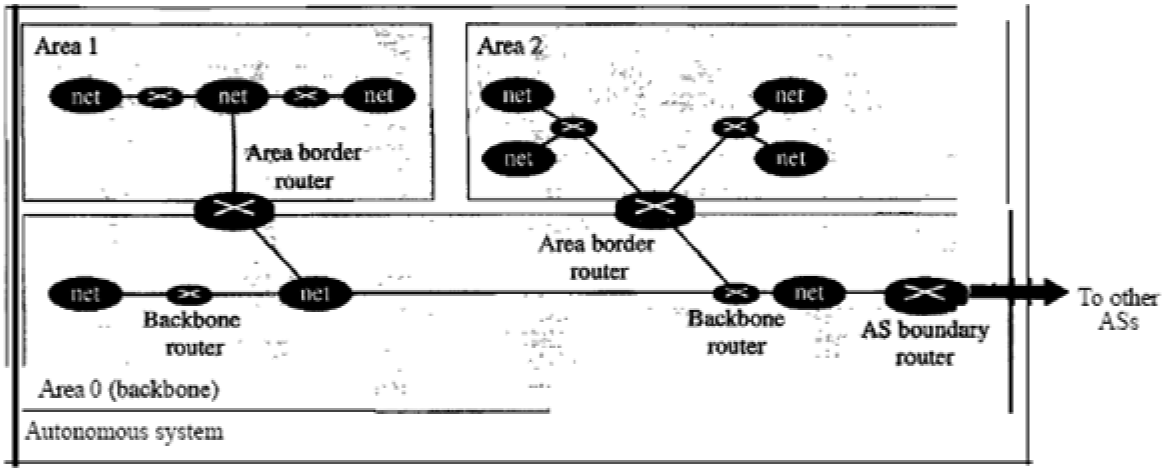
The final lists are Permanent list: A(O), B(5), C(2), D(3), E(6) Tentative list: empty Calculation of Routing Table from Shortest Path Tree Each node uses the shortest path tree protocol to construct its routing table.

OSPF

The Open Shortest Path First or OSPF protocol is an intra-domain routing protocol based on link state routing. Its domain is also an autonomous system. Areas to handle routing efficiently and in a timely manner, OSPF divide an autonomous system into areas. An area is a collection of networks, hosts, and routers all contained within an autonomous system. An autonomous system can be divided into many different areas. All networks inside an area must be connected.

Routers inside an area flood the area with routing information. At the border of an area, special routers called area border routers summarize the information about the area and send it to other areas. The backbone serves as a primary area and the other areas as secondary areas. This does not mean that the routers within areas cannot be connected to each other, however. The routers inside the backbone are called the backbone routers. Note that a backbone router can also be an area border router.

Figure 22.24 *Areas in an autonomous system*

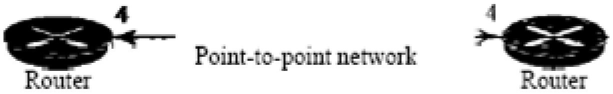


If, because of some problem, the connectivity between a backbone and an area is broken, a virtual link between routers must be created by an administrator to allow continuity of the functions of the backbone as the primary area.

Metric The OSPF protocol allows the administrator to assign a cost, called the metric, to each route. The metric can be based on a type of service (minimum delay, maximum throughput, and so on). As a matter of fact, a router can have multiple routing tables, each based on a different type of service.

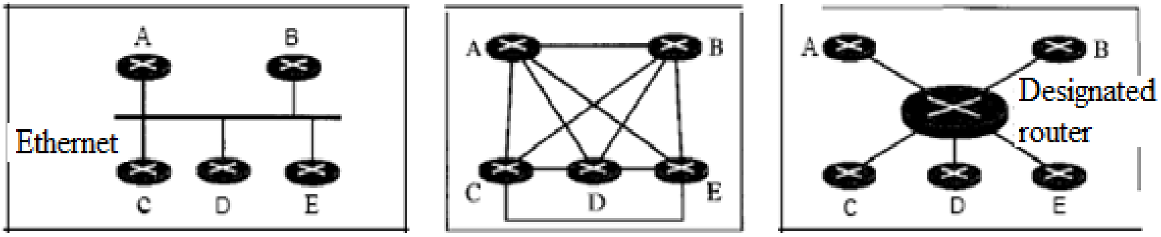
Types of Links In OSPF terminology, a connection is called a *link.* Four types of links have been defined: point-to-point, transient, stub, and virtual.

A point-to-point link: It connects two routers without any other host or router in between. In other words, the purpose of the link (network) is just to connect the two routers. An example of this type of link is two routers connected by a telephone line or a T line. There is no need to assign a network address to this type of link. Graphically, the routers are represented by nodes, and the link is represented by a bidirectional edge connecting the nodes.



A transient link: It is a network with several routers attached to it. The data can enter through any of the routers and leave through any router. All LANs and some WANs with two or more routers are of this type. In this case, each router has many neighbors. For example, consider the Ethernet in Figure a. Router A has routers B, C, D, and E as neighbors. Router B has routers A, C, D, and E as neighbors. If we want to show the neighborhood relationship in this situation, we have the graph shown in Figure b.

Figure Transient Link

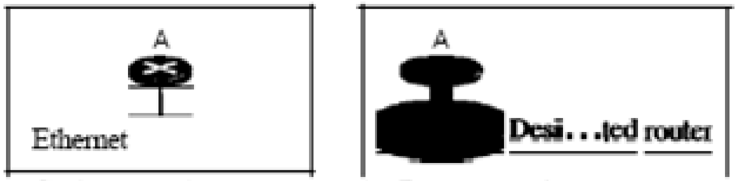


a. Transient Network b.Unrealistic representation c.Realistic representation

It is not realistic because there is no single network (link) between each pair of routers; there is only one network that serves as a crossroad between all five routers. However, because a network is not a machine, it cannot function as a router. One of the routers in the network takes this responsibility. It is assigned a dual purpose; it is a true router and a designated router.

We can use the topology shown in Figure c to show the connections of a transient network. While there is a metric from each node to the designated router, there is no metric from the designated router to any other node. The reason is that the designated router represents the network. When a packet enters a network, we assign a cost; when a packet leaves the network to go to the router, there is no charge.

A **stub link:** It is a network that is connected to only one router. The data packets enter the network through this single router and leave the network through this same router. This is a special case of the transient network using the router as a node and using the designated router for the network. However, the link is only one-directional, from the router to the network.



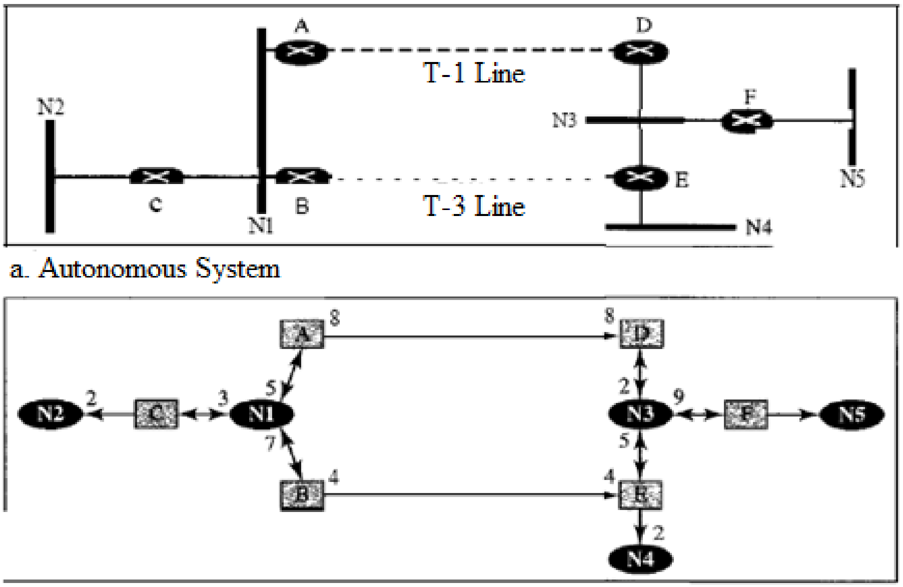
a. Stub network b. Representation

When the link between two routers is broken, the administration may create a **virtual link** between them, using a longer path that probably goes through several routers.

Graphical Representation: We use symbols such as Nl and N2 for transient and stub networks. There is no need to assign an identity to a point-to-point network. The figure also shows the graphical representation of the AS as seen by OSPF.

We have used square nodes for the routers and ovals for the networks (represented by designated routers). However, OSPF sees both as nodes. Note that we have three stub networks.

Example of Graphical representation of OSPF

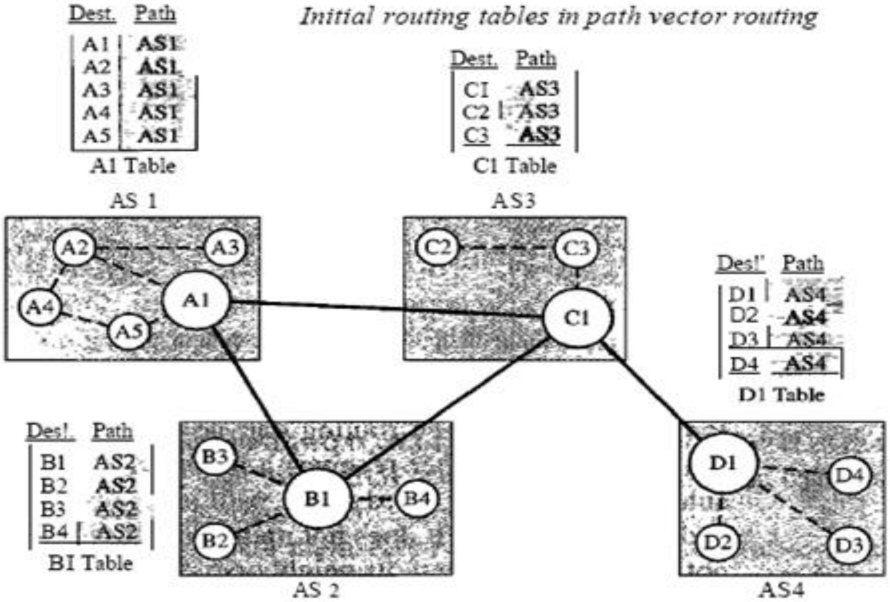


a. Autonomous System

b.Graphical reosentation

Path Vector Routing

Distance vector and link state routing are both intra-domain routing protocols. They can be used inside an autonomous system, but not between autonomous systems. These two protocols are not suitable for inter-domain routing mostly because of scalability.



Initial routing tables in path vector routing

Dest.	Path
A1	AS1
A2	AS1
A3	AS1
A4	AS1
A5	AS1

A1 Table

Dest.	Path
C1	AS3
C2	AS3
C3	AS3

C1 Table

Des!'	Path
D1	AS4
D2	AS4
D3	AS4
D4	AS4

D1 Table

Des!.	Path
B1	AS2
B2	AS2
B3	AS2
B4	AS2

BI Table

Path vector routing proved to be useful for inter-domain routing. The principle of path vector routing is similar to that of distance vector routing. In path vector routing, we assume that there is one node in each autonomous system that acts on behalf of the entire autonomous system. Let us call it the speaker node. The speaker node in an AS creates a routing table and advertises it to speaker nodes in the neighboring ASs. The idea is the same as for distance vector routing except that only speaker nodes in each AS can communicate with each other.

Initialization: At the beginning, each speaker node can know only the reachability of nodes inside its autonomous system. Figure shows the initial tables for each speaker node in a system made of four ASs.

Node Al is the speaker node for ASl, Bl for AS2, Cl for AS3, and Dl for AS4. Node Al creates an initial table that shows Al to A5 are located in ASI and can be reached through it. Node Bl advertises that Bl to B4 are located in AS2 and can be reached through Bl. And so on.

Sharing Just as in distance vector routing, in path vector routing, a speaker in an autonomous system shares its table with immediate neighbors. In Figure node Al shares its table with nodes Bl and Cl. Node Cl shares its table with nodes Dl, Bl, and Al. Node Bl shares its table with Cl and Al. Node Dl shares its table with Cl. Updating when a speaker node receives a two-column table from a neighbor, it updates its own table by adding the nodes that are not in its routing table and adding its own autonomous system and the autonomous system that sent the table.

BGP

Border Gateway Protocol (BGP) is an inter-domain routing protocol using path vector routing. It first appeared in 1989 and has gone through four versions.

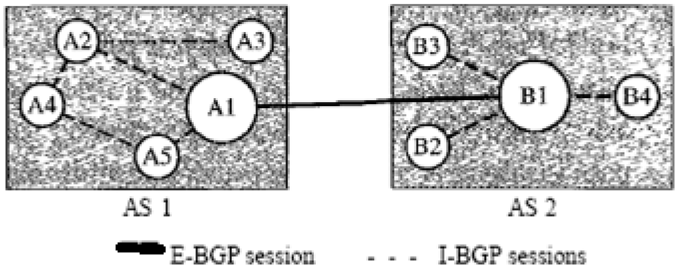
Types **of Autonomous** Systems As we said before, the Internet is divided into hierarchical domains called autonomous systems. A local ISP that provides services to local customers is an autonomous system. We can divide autonomous systems into three categories: stub, multi-homed, and transit.

Stub AS: A stub AS has only one connection to another AS. The inter-domain data traffic in a stub AS can be either created or terminated in the AS. The hosts in the AS can send data traffic to other ASs. The hosts in the AS can receive data coming from hosts in other ASs. Data traffic, however, cannot pass through a stub AS. A stub AS is either a source or a sink. A good example of a stub AS is a small corporation or a small local ISP.

Multi-homed AS: A multi-homed AS has more than one connection to other ASs, but it is still only a source or sink for data traffic. It can receive data traffic from more than one AS. It can send data traffic to more than one AS, but there is no transient traffic. E.g a large corporation that is connected to more than one national AS that does not allow transient traffic.

Transit AS: A transit AS is a multi-homed AS that also allows transient traffic. Good examples of transit ASs are national and international ISPs (Internet backbones). The list of attributes helps the receiving router make a more-informed decision when applying its policy. Attributes are divided into two broad categories: well known and optional. A well known attribute is one that every BGP router must recognize. An optional attribute is one that needs not be recognized by every router.

Internal and External BGP Sessions

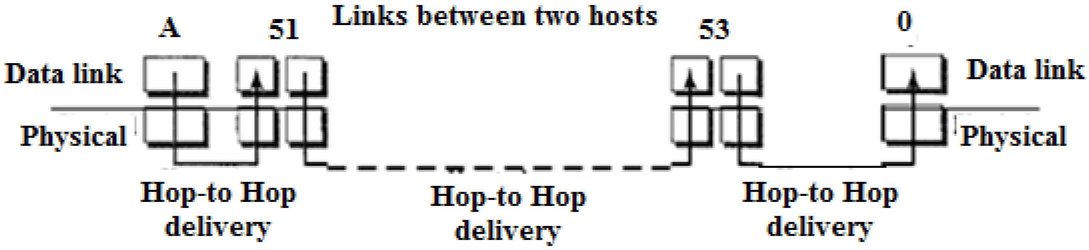


BGP Sessions The exchange of routing information between two routers using BGP takes place in a session. A session is a connection that is established between two BGP routers only for the sake of exchanging routing information. When a TCP connection is created for BGP, it can last for a long time, until something unusual happens. For this reason, BGP sessions are sometimes referred to as *semi permanent connections.*

External and Internal BGP If we want to be precise, BGP can have two types of sessions: external BGP (E-BGP) and internal BGP (I-BGP) sessions. The E-BGP session is used to exchange information between two speaker nodes belonging to two different autonomous systems. The I-BGP session, on the other hand, is used to exchange routing information between two routers inside an autonomous system.

INTERNETWORKING

The physical and data link layers of a network operate locally. These two layers are jointly responsible for data delivery on the network from one node to the next. This internetwork is made of five networks: four LANs and one WAN. If host A needs to send a data packet to host D, the packet needs to go first from A to Rl (a switch or router), then from Rl to R3, and finally from R3 to host D. We say that the data packet passes through three links. In each link, two physical and two data link layers are involved.



However, there is a big problem here. When data arrive at interface fl of Rl, how does RI know that interface f3 is the outgoing interface? There is no provision in the data link (or physical) layer to help Rl make the right decision. The frame does not carry any routing information either.

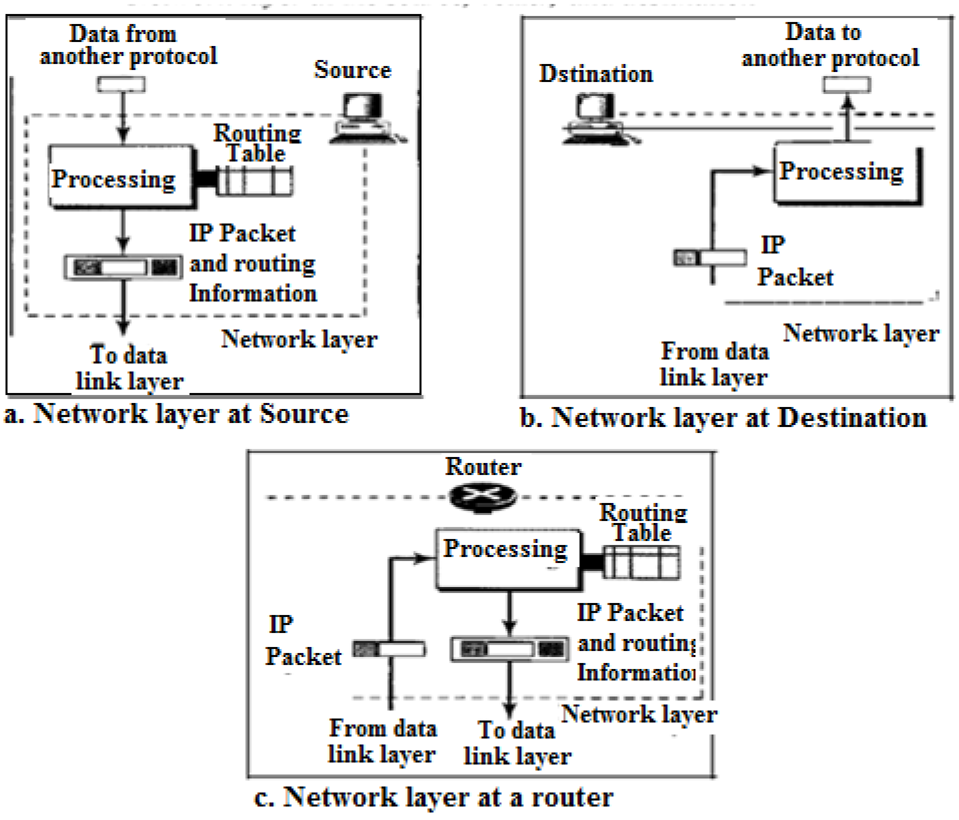
Need for Network Layer: To solve the problem of delivery through several links, the network layer (or the internetwork layer, as it is sometimes called) was designed. The network layer is responsible for host-to-host delivery and for routing the packets through the routers or switches. Figure shows the same internetwork with a network layer added.

The network layer at the source is responsible for creating a packet from the data coming from another protocol. The header of the packet contains the logical addresses of the source and destination. The network layer is responsible for checking its routing table to find the routing information.

The network layer at the switch or router is responsible for routing the packet. When a packet arrives, the router or switch consults its routing table and finds the interface from which the packet must be sent. The packet, after changes in header, with the routing information is passed to the data link layer again.

The network layer at the destination is responsible for address verification; it makes sure that the destination address on the packet is the same as the address of the host. If the packet is a fragment, the network layer waits until all fragments have arrived, and then reassembles them and delivers the reassembled packet to the transport layer.

Network layer at the source, router and



a. Network layer at Source

b. Network layer at Destination

c. Network layer at a router

Internet as a Datagram Network: The Internet, at the network layer, is a packet-switched network. Packet switching uses either the virtual circuit approach or the datagram approach. The Internet has chosen the datagram approach to switching in the network layer. It uses the universal addresses defined in the network layer to route packets from the source to the destination.

Internet as a Connectionless Network: Delivery of a packet can be accomplished by using either a connection-oriented or a connectionless network service.

In a connection-oriented service, the source first makes a connection with the destination before sending a packet. When the connection is established, a sequence of packets from the same source to the same destination can be sent one after another. They are sent on the same path in sequential order. A packet is logically connected to the packet traveling before it and to the packet traveling after it. When all packets of a message have been delivered, the connection is terminated. This type of service is used in a virtual-circuit approach to packet switching such as in Frame Relay and ATM.

In connectionless service, the network layer protocol treats each packet independently, with each packet having no relationship to any other packet. The packets in a message mayor may not travel the same path to their destination. This type of service is used in the datagram approach to packet switching. The Internet has chosen this type of service at the network layer. The reason for this decision is that the Internet is made of so many heterogeneous networks that it is almost impossible to create a connection from the source to the destination without knowing the nature of the networks in advance. Communication at the network layer in the Internet is connectionless.

Unit-IV
Transport and Congestion

The transport layer is responsible for process-to-process delivery of the entire message. A process is an application program running on a host. Whereas the network layer oversees source-to-destination delivery of individual packets, it does not recognize any relationship between those packets.

The transport layer, on the other hand, ensures that the whole message arrives intact and in order, overseeing both error control and flow control at the source-to-destination level. The transport layer is responsible for the delivery of a message from one process to another.

Elements of Transport Protocols

Client/Server Paradigm: Although there are several ways to achieve process-to-process communication, the most common one is through the client/server paradigm. A process on the local host, called a client, needs services from a process usually on the remote host, called a server.

Operating systems today support both multiuser and multiprogramming environments. A remote computer can run several server programs at the same time, just as local computers can run one or more client programs at the same time. For communication, we must define the following:

1. Local host 2. Local process 3. Remote host 4. Remote process

Addressing

Whenever we need to deliver something to one specific destination among many, we need an address. At the data link layer, we need a MAC address to choose one node among several nodes if the connection is not point-to-point. A frame in the data link layer needs a destination MAC address for delivery and a source address for the next node's reply. At the network layer, we need an IP address to choose one host among millions. A datagram in the network layer needs a destination IP address for delivery and a source IP address for the destination's reply.

Port Address: At the transport layer, we need a transport layer address, called a port number, to choose among multiple processes running on the destination host. The destination port number is needed for delivery; the source port number is needed for the reply. In the Internet model, the port numbers are 16-bit integers between 0 and 65,535. The client program defines itself with a port number, chosen randomly by the transport layer software running on the client host. This is the ephemeral port number.

IANA Ranges: The IANA (Internet Assigned Number Authority) has divided the port numbers into three ranges: well known, registered, and dynamic.

- Well-known ports. The ports ranging from 0 to 1023 are assigned and controlled by IANA. These are the well-known ports.

- Registered ports: The ports ranging from 1024 to 49,151 are not assigned or controlled by IANA. They can only be registered with IANA to prevent duplication.
- Dynamic ports: The ports ranging from 49,152 to 65,535 are neither controlled nor registered. They can be used by any process. These are the ephemeral ports.

Socket Addresses: Process-to-process delivery needs two identifiers, IP address and the port number, at each end to make a connection. The combination of an IP address and a port number is called a socket address. The client socket address defines the client process uniquely just as the server socket address defines the server process uniquely. A transport layer protocol needs a pair of socket addresses: the client socket address and the server socket address. These four pieces of information are part of the IP header and the transport layer protocol header. The IP header contains the IP addresses; the UDP or TCP header contains the port numbers.

Multiplexing

Multiplexing: At the sender site, there may be several processes that need to send packets. However, there is only one transport layer protocol at any time. This is a many-to-one relationship and requires multiplexing. ***Demultiplexing:*** At the receiver site, the relationship is one-to-many and requires demultiplexing. The transport layer receives datagrams from the network layer. After error checking and dropping of the header, the transport layer delivers each message to the appropriate process based on the port number.

Connectionless Versus Connection-Oriented Service

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

Connectionless Service: In a connectionless service, the packets are sent from one party to another with no need for connection establishment or connection release. The packets are not numbered; they may be delayed or lost or may arrive out of sequence. There is no acknowledgment either.

Connection Oriented *Service:* In a connection-oriented service, a connection is first established between the sender and the receiver. Data are transferred. At the end, the connection is released. TCP and SCTP are connection-oriented protocols.

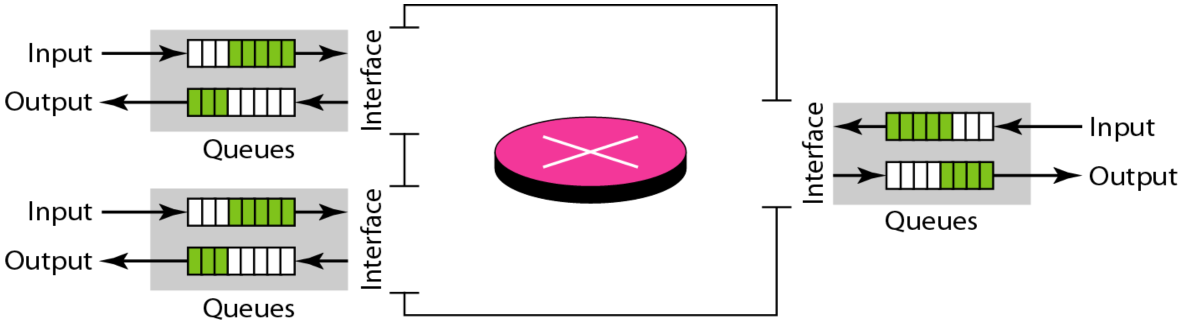
Reliable Versus Unreliable

The transport layer service can be reliable or unreliable. If the application layer program needs reliability, we use a reliable transport layer protocol by implementing flow and error control at the transport layer. This means a slower and more complex service. On the other hand, if the application program does not need reliability because it uses its own flow and error control mechanism or it needs fast service or the nature of the service does not demand flow and error control (real-time applications), then an unreliable protocol can be used.

Congestion

Queues in a router

- An important issue in a packet-switched network is congestion.
- Congestion in a network may occur if the load on the network-the number of packets sent to the network-is greater than the capacity of the network-the number of packets a network can handle.
- Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.



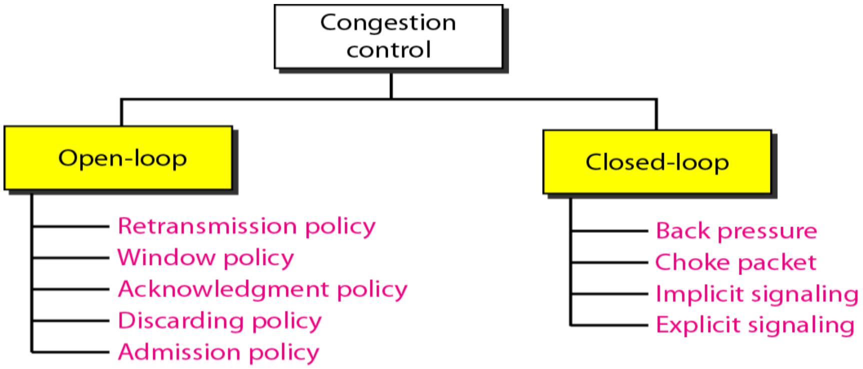
- Congestion in a network or internetwork occurs because routers and switches have queues-buffers that hold the packets before and after processing.
- A router has an input queue and an output queue for each interface. When a packet arrives at the incoming interface, it undergoes three steps before departing the packet is put at the end of the input queue while waiting to be checked.
- The processing module of the router removes the packet from the input queue once it reaches the front of the queue and uses its routing table and the destination address to find the route.
- The packet is put in the appropriate output queue and waits its turn to be sent.

We need to be aware of two issues.

- First, if the rate of packet arrival is higher than the packet processing rate, the input queues become longer and longer.
- Second, if the packet departure rate is less than the packet processing rate, the output queues become longer and longer.

Congestion Control

Congestion control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened. In general, congestion control mechanisms can be divided into two broad categories: open-loop congestion control (prevention) and closed-loop congestion control (removal) as shown in Figure.



Open-Loop Congestion Control

In open-loop congestion control, policies are applied to prevent congestion before it happens. In these mechanisms, congestion control is handled by either the source or the destination.

Retransmission Policy

- Retransmission is sometimes unavoidable. If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted.
- Retransmission in general may increase congestion in the network. However, a good retransmission policy can prevent congestion.
- The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion. For example, the retransmission policy used by TCP is designed to prevent or alleviate congestion.

Window Policy

- The type of window at the sender may also affect congestion. The Selective Repeat window is better than the Go-Back-N window for congestion control.
- In the Go-Back-N window, when the timer for a packet times out, several packets may be resent, although some may have arrived safe and sound at the receiver. This duplication may make the congestion worse.
- The Selective Repeat window, on the other hand, tries to send the specific packets that have been lost or corrupted.

Acknowledgment Policy

- The acknowledgment 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.
- Several approaches are used in this case. A receiver may send an acknowledgment only if it has a packet to be sent or a special timer expires.

- A receiver may decide to acknowledge only N packets at a time. We need to know that the acknowledgments are also part of the load in a network. Sending fewer acknowledgments means imposing fewer loads on the network.

Discarding Policy

- A good discarding policy by the routers may prevent congestion and at the same time may not harm the integrity of the transmission.
- For example, in audio transmission, if the policy is to discard less sensitive packets when congestion is likely to happen, the quality of sound is still preserved and congestion is prevented or alleviated.

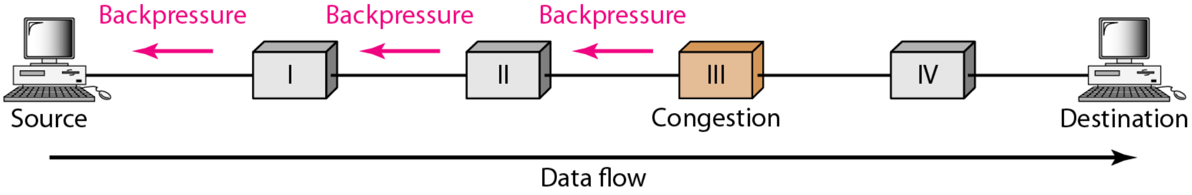
Admission Policy

- 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

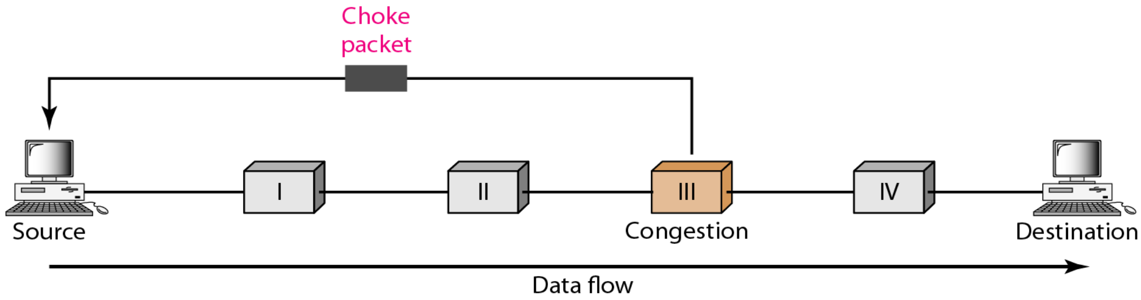
Closed-loop congestion control mechanisms try to alleviate congestion after it happens. Several mechanisms have been used by different protocols.

Backpressure method for alleviating congestion



- Figure shows the idea of backpressure. Its input buffer and informs node II to slow down. Node II, in turn, may be congested because it is slowing down the output flow of data.
- If node II is congested, it informs node I to slow down, which in turn may create congestion. If so, node I inform the source of data to slow down. This, in time, alleviates the congestion.
- Note that the pressure on node III is moved backward to the source to remove the congestion.
- None of the virtual-circuit networks we studied in this book use backpressure. It was implemented in the first virtual-circuit network, X.25.
- The technique cannot be implemented in a datagram network because in this type of network, a node (router) does not have the slightest knowledge of the upstream router.

Choke Packet



- A choke packet is a packet sent by a node to the source to inform it of congestion.
- Note the difference between the backpressure and choke packet methods.
- In backpressure, warning is from one node to its upstream node, although warning may eventually reach source.
- In the choke packet method, the warning is from the router, which has encountered congestion, to the source station directly. The intermediate nodes through which the packet has traveled are not warned.
- When a router in the Internet is overwhelmed with IP datagram's, it may discard some of them; but it informs the source host, using a source quench ICMP message.
- The warning message goes directly to the source station; the intermediate routers, and does not take any action. Figure shows the idea of a choke packet.

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 from other symptoms. For example, when a source sends several packets and there is no acknowledgment for a while, one assumption is that the network is congested.
- Delay in receiving an acknowledgment is interpreted as congestion in the network; source should slow down.

Explicit Signaling

- The node that experiences congestion can explicitly send a signal to the source or destination.
- The explicit signaling method, however, is different from the choke packet method. In the choke packet method, a separate packet is used for this purpose; in the explicit signaling method, the signal is included in the packets that carry data.
- Explicit signaling, as we will see in Frame Relay congestion control, can occur in either the forward or the backward direction.

Backward Signaling

A bit can be set in a packet moving in the direction opposite to the congestion. This bit can warn the source that there is congestion and that it needs to slow down to avoid the discarding of packets.

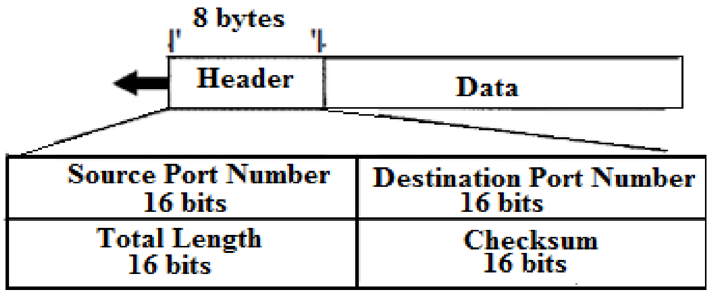
Forward Signaling

A bit can be set in a packet moving in the direction of the congestion. This bit can warn the destination that there is congestion. The receiver in this case can use policies, such as slowing down the acknowledgments, to alleviate the congestion

USER DATAGRAM PROTOCOL (UDP)

The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to process communication instead of host-to-host communication. Also, it performs very limited error checking.

User Datagram: UDP packets, called user datagrams, have a fixed-size header of 8 bytes. Figure shows the format of a user datagram.



Source Port Number 16 bits	Destination Port Number 16 bits
Total Length 16 bits	Checksum 16 bits

The fields are as follows:

Source port number: This is the port number used by the process running on the source host. It is 16 bits long, which means that the port number can range from 0 to 65,535.

Destination port number: This is the port number used by the process running on the destination host. It is also 16 bits long. If the destination host is the server (a client sending a request), the port number, in most cases, is a well-known port number.

Length: This is a 16-bit field that defines the total length of the user datagram, header plus data. The 16 bits can define a total length of 0 to 65,535 bytes. However, the total length needs to be much less because a UDP user datagram is stored in an IP datagram with a total length of 65,535 bytes. We can deduce the length of a UDP datagram that is encapsulated in an IP datagram. UDP length = IP length - IP header's length

Checksum: This field is used to detect errors over the entire user datagram (header plus data). The checksum is discussed next.

UDP Operation

Connectionless Services

As mentioned previously, UDP provides a connectionless service. This means that each user datagram sent by UDP is an independent datagram. There is no relationship between the different user datagrams even if they are coming from the same source process and going to the same destination program. The user datagrams are not numbered. Also, there is no connection establishment and no connection termination, as is the case for TCP.

This means that each user datagram can travel on a different path. One of the ramifications of being connectionless is that the process that uses UDP cannot send a stream of data to UDP and expect UDP to chop them into different related user datagrams. Instead each request must be small enough to fit into one user datagram. Only those processes sending short messages should use UDP.

Flow and Error Control: UDP is a very simple, unreliable transport protocol. There is no flow control and hence no window mechanism. The receiver may overflow with incoming messages. There is no error control mechanism in UDP except for the checksum. This means that the sender does not know if a message has been lost or duplicated.

When the receiver detects an error through the checksum, the user datagram is silently discarded. The lack of flow control and error control means that the process using UDP should provide these mechanisms.

Encapsulation and De-capsulation: To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages in an IP datagram.

Queuing: The client process can send messages to the outgoing queue by using the source port number specified in the request. UDP removes the messages one by one and, after adding the UDP header, delivers them to IP. An outgoing queue can overflow. If this happens, the operating system can ask the client process to wait before sending any more messages.

When a message arrives for a client, UDP checks to see if an incoming queue has been created for the port number specified in the destination port number field of the user datagram. If there is such a queue, UDP sends the received user datagram to the end of the queue. If there is no such queue, UDP discards the user datagram and asks the ICMP protocol to send a *port unreachable* message to the server.

At the server site, a server asks for incoming and outgoing queues, using its well-known port, when it starts running. The queues remain open as long as the server is running. When a message arrives for a server, UDP checks to see if an incoming queue has been created for the port number specified in the destination port number field of the user datagram. When a server wants to respond to a client, it sends messages to the outgoing queue, using the source port number specified in the request.

Use of UDP

- UDP is suitable for a process that requires simple request-response communication.
- UDP is suitable for a process with internal flow and error control mechanisms. For example, the Trivial File Transfer Protocol (TFTP) process includes flow and error control.

- Multicasting capability is embedded in the UDP software but not in the TCP software.
- UDP is used for management processes such as SNMP.
- UDP is used for some route updating protocols such as Routing Information Protocol

TCP

The second transport layer protocol we discuss in this chapter is called Transmission Control Protocol (TCP). TCP, like UDP, is a process-to-process (program-to-program) protocol. TCP, therefore, like UDP, uses port numbers. Unlike UDP, TCP is a connection oriented protocol; it creates a virtual connection between two TCPs to send data. In addition, TCP uses flow and error control mechanisms at the transport level. In brief, TCP is called a *connection-oriented, reliable* transport protocol. It adds connection-oriented and reliability features to the services of IP.

TCP Services

1. ***Process-to-Process Communication:*** Like UDP, TCP provides process-to-process communication using port numbers. It has some well-known port numbers.
2. ***Full-Duplex Communication:*** TCP offers full-duplex service, in which data can flow in both directions at the same time. Each TCP then has a sending and receiving buffer, and segments move in both directions.
3. ***Connection-Oriented Service:*** TCP, unlike UDP, is a connection-oriented protocol. When a process at site A wants to send and receive data from another process at site B, the following occurs: 1. The two TCPs establish a connection between them.
   2. Data are exchanged in both directions.
   3. The connection is terminated.

   Note that this is a virtual connection, not a physical connection. The TCP segment is encapsulated in an IP datagram and can be sent out of order, or lost, or corrupted, and then resent.
4. ***Reliable Service:*** TCP is a reliable transport protocol. It uses an acknowledgment mechanism to check the safe and sound arrival of data.

TCP Features

To provide the services mentioned in the previous section, TCP has several features that are briefly summarized in this section and discussed later in detail.

1. Numbering System: Although the TCP software keeps track of the segments being transmitted or received, there is no field for a segment number value in the segment header. Instead, there are two fields called the sequence number and the acknowledgment number. These two fields refer to the byte number and not the segment number.

Byte Number: TCP numbers all data bytes that are transmitted in a connection. Numbering is independent in each direction. When TCP receives bytes of data from a process, it stores them in the sending buffer and numbers them. TCP generates a random number between 0 and 2^{32} - 1 for the number of the first byte.

Sequence Number: After the bytes have been numbered, TCP assigns a sequence number to each segment that is being sent. The sequence number for each segment is the number of the first byte carried in that segment.

2. Flow Control: TCP, unlike UDP, provides *flow control.* The receiver of the data controls the amount of data that are to be sent by the sender. This is done to prevent the receiver from being overwhelmed with data. The numbering system allows TCP to use a byte-oriented flow control.

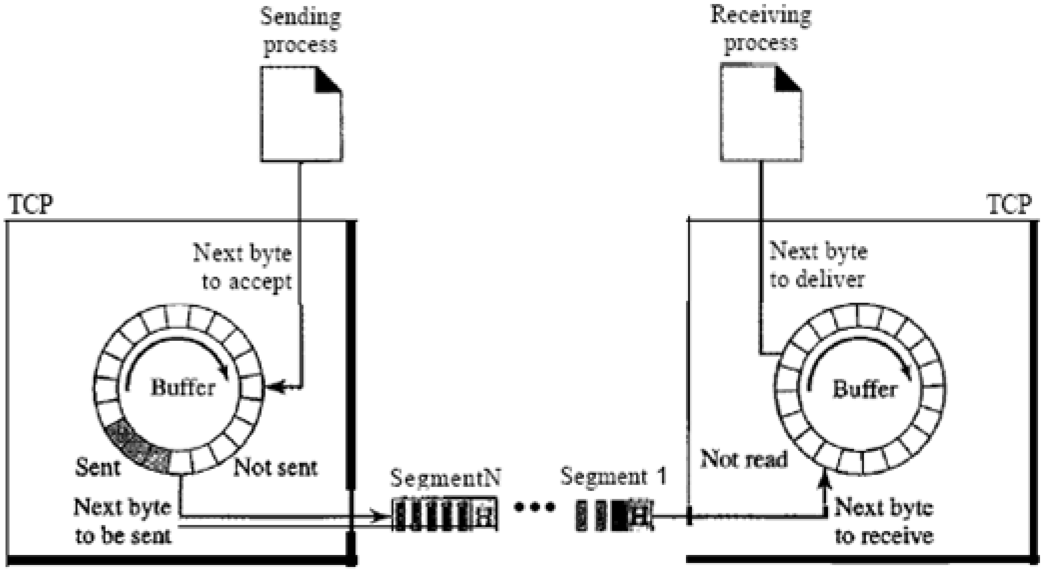
3. Error Control: To provide reliable service, TCP implements an error control mechanism. Although error control considers a segment as the unit of data for error detection (loss or corrupted segments), error control is byte-oriented, as we will see later.

4. Congestion Control: TCP, unlike UDP, takes into account congestion in the network. The amount of data sent by a sender is not only controlled by the receiver (flow control), but is also determined by the level of congestion in the network.

TCP Segment

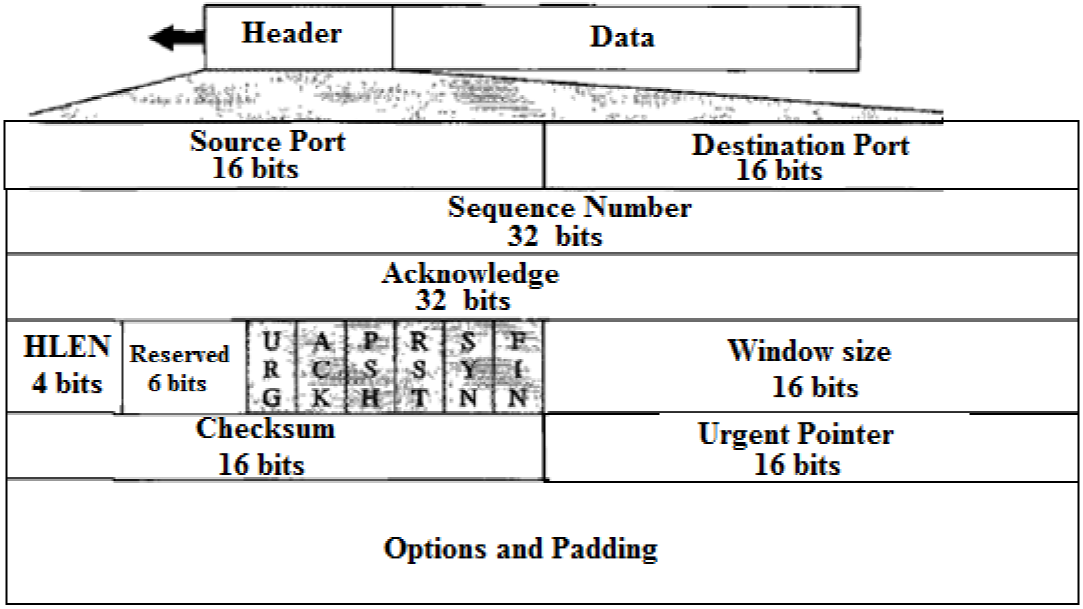
TCP adds a header to each segment (for control purposes) and delivers the segment to the IP layer for transmission. The segments are encapsulated in IP datagrams and transmitted. This entire operation is transparent to the receiving process. Later we will see that segments may be received out of order, lost, or corrupted and resent. All these are handled by TCP with the receiving process unaware of any activities. Figure shows how segments are created from the bytes in the buffers.

Figure TCP Segments



The segment consists of a 20- to 60-byte header, followed by data from the application program. The header is 20 bytes if there are no options and up to 60 bytes if it contains options. The format of a segment is shown in Figure.

TCP Segment Format



Source port address: This is a 16-bit field that defines the port number of the application program in the host that is sending the segment. This serves the same purpose as the source port address in the UDP header.

Destination port address: This is a 16-bit field that defines the port number of the application program in the host that is receiving the segment. This serves the same purpose as the destination port address in the UDP header.

Sequence number: This 32-bit field defines the number assigned to the first byte of data contained in this segment. To ensure connectivity, each byte to be transmitted is numbered. During connection establishment, each party uses a random number generator to create an initial sequence number (ISN), which is usually different in each direction.

Acknowledgment number: This 32-bit field defines the byte number that the receiver of the segment is expecting to receive from the other party. If the receiver of the segment has successfully received byte number *x* from the other party, it defines $x + 1$ as the acknowledgment number. Acknowledgment and data can be piggybacked together.

Header length: This 4-bit field indicates the number of 4-byte words in the TCP header. The length of the header can be between 20 and 60 bytes.

Reserved: This is a 6-bit field reserved for future use.

Control: This field defines 6 different control bits or flags. One or more of these bits can be set at a time.

A TCP Connection Management

TCP is connection-oriented. A connection-oriented transport protocol establishes a virtual path between the source and destination. All the segments belonging to a message are then sent over this virtual path. Using a single virtual pathway for the entire message facilitates the acknowledgment process as well as retransmission of damaged or lost frames. You may wonder how TCP, which uses the services of IP, a connectionless protocol, can be connection-oriented.

The point is that a TCP connection is virtual, not physical. TCP operates at a higher level. TCP uses the services of IP to deliver individual segments to the receiver, but it controls the connection itself. If a segment is lost or corrupted, it is retransmitted. Unlike TCP, IP is unaware of this retransmission. If a segment arrives out of order, TCP holds it until the missing segments arrive; IP is unaware of this reordering.

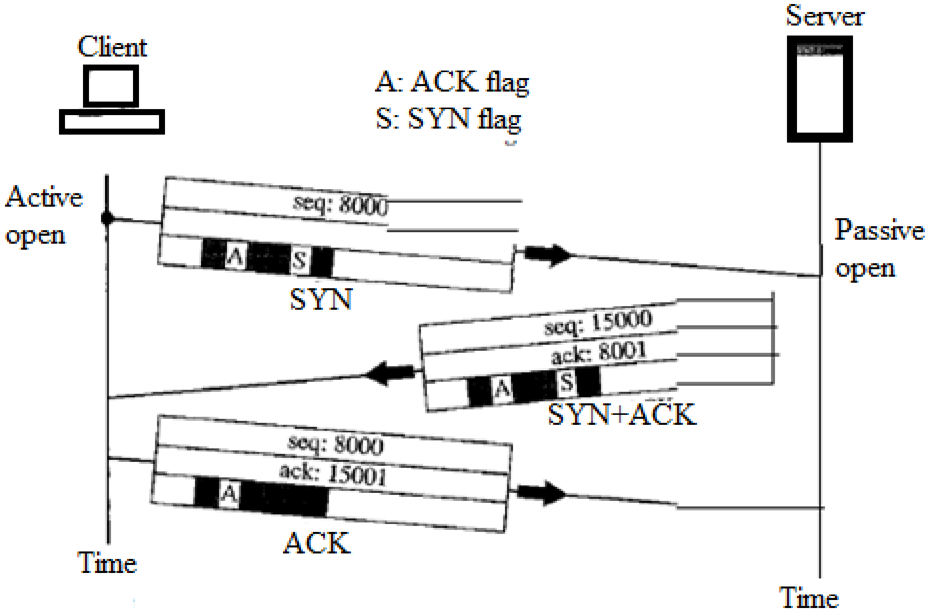
In TCP, connection-oriented transmission requires three phases: connection establishment, data transfer, and connection termination.

Connection Establishment:

TCP transmits data in full-duplex mode. When two TCPs in two machines are connected, they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data are transferred.

Three-Way Handshaking: The connection establishment in TCP is called three way handshaking. In our example, an application program, called the client, wants to make a connection with another application program, called the server, using TCP as the transport layer protocol.

Connection establishment using three way handshaking



The process starts with the server. The server program tells its TCP that it is ready to accept a connection. This is called a request for a *passive open.* Although the server TCP is ready to accept any connection from any machine in the world, it cannot make the connection itself.

The client program issues a request for an *active open.* A client that wishes to connect to an open server tells its TCP that it needs to be connected to that particular server. TCP can now start the three-way handshaking process as shown in Figure.

To show the process, we use two time lines: one at each site. Each segment has values for all its header fields and perhaps for some of its option fields, too. We show the sequence number, the acknowledgment number, the control flags (only those that are set), and the window size, if not empty. The three steps in this phase are as follows.

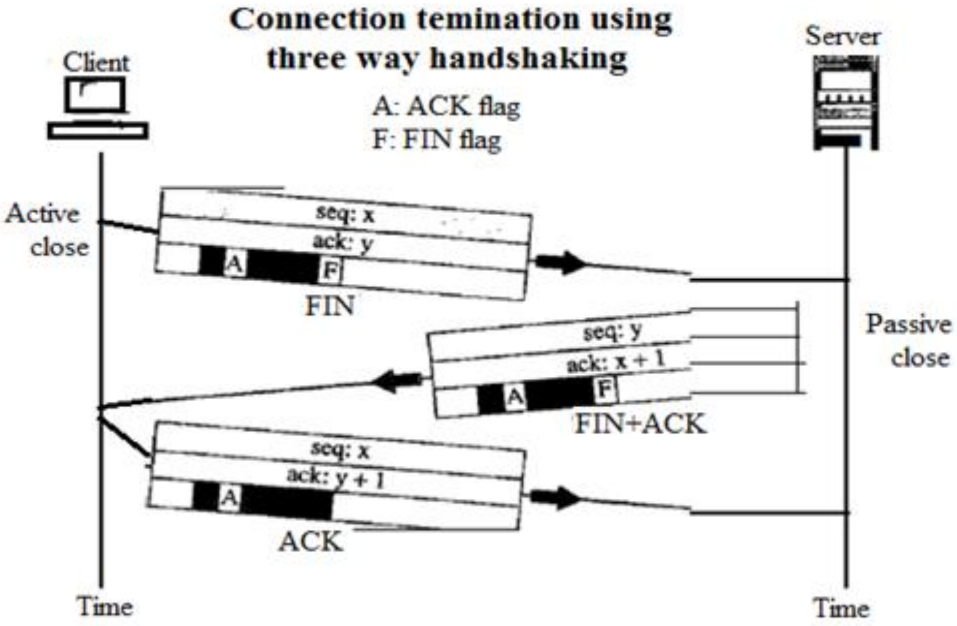
Step1: The client sends the first segment, a SYN segment, in which only the SYN flag is set. This segment is for synchronization of sequence numbers. It consumes one sequence number. When the time of data transfer starts, the sequence number is incremented by 1. We can say that the SYN segment carries no real data, but we can think of it as containing 1 imaginary byte. A SYN segment cannot carry data, but it consumes one sequence number.

Step 2: The server sends the second segment, a SYN +ACK segment, with 2 flag bits set: SYN and ACK. This segment has a dual purpose. It is a SYN segment for communication in the other direction and serves as the acknowledgment for the SYN segment. It consumes one sequence number. A SYN +ACK segment cannot carry data, but does consume one sequence number.

Step 3: The client sends the third segment. This is just an ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. Note that the sequence number in this segment is the same as the one in the SYN segment; the ACK segment does not consume any sequence numbers. An ACK segment, if carrying no data, consumes no sequence number.

Data Transfer: After connection is established, bidirectional data transfer can take place. The client and server can both send data and acknowledgments. The acknowledgment is piggybacked with the data. The data segments sent by the client have the PSH (push) flag set so that the server TCP knows to deliver data to the server process as soon as they are received. When the receiving TCP receives a segment with the URG bit set, it extracts the urgent data from the segment, and delivers them, out of order, to the receiving application program.

Connection Termination: Any of the two parties involved in exchanging data can close the connection, although it is usually initiated by the client. Most implementations today allow two options for connection termination: three-way handshaking and four-way handshaking with a half-close option.



1. In a normal situation, the client TCP, after receiving a close command from the client process, sends the first segment, a FIN segment in which the FIN flag is set. The FIN segment consumes one sequence number if it does not carry data.
2. The server TCP, after receiving the FIN segment, informs its process of the situation and sends the second segment, a FIN +ACK segment, to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the connection in the other direction.
3. The client TCP sends the last segment, an ACK segment, to confirm the receipt of the FIN segment from the TCP server. This segment contains the acknowledgment number, which is 1 plus the sequence number received in the FIN segment from the server.

Flow Control using Sliding window

TCP uses a sliding window to handle flow control. The sliding window protocol used by TCP, however, is something between the *Go-Back-N* and Selective Repeat sliding window. There are two big differences between this sliding window and the one we used at the data link layer.

1. The sliding window of TCP is byte-oriented; the one in data link layer is frame-oriented.
2. The TCP's sliding window is of variable size; the one in the data link layer was of fixed.

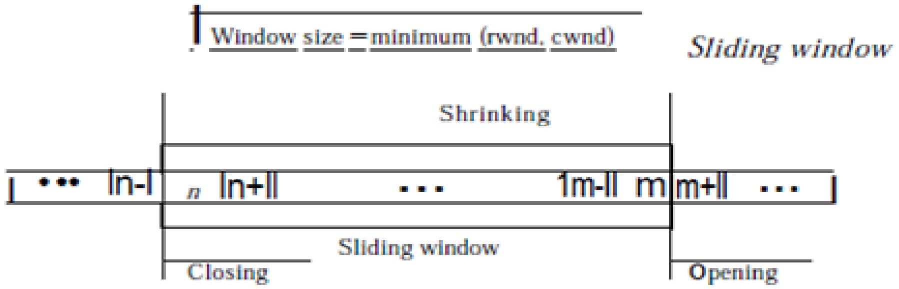
Format of Sliding Window protocol

Figure shows the sliding window in TCP. The window spans a portion of the buffer containing bytes received from the process. The bytes inside the window are the bytes that can be in transit; they can be sent without worrying about acknowledgment.

The imaginary window has two walls: one left and one right. The window is *opened, closed,* or *shrunk.* The sender must obey the commands of the receiver in this matter.

Opening a window means moving the right wall to the right. This allows more new bytes in the buffer that are eligible for sending.

Closing the window means moving the left wall to the right. This means that some bytes have been acknowledged and the sender need not worry about them anymore.



Shrinking the window means moving the right wall to the left. This is strongly discouraged and not allowed in some implementations because it means revoking the eligibility of some bytes for sending. This is a problem if the sender has already sent these bytes.

Receiver window is the value advertised by the opposite end in a segment containing acknowledgment. Note that the left wall cannot move to the left because this would revoke some of the previously sent acknowledgments.

Operation concepts of TCP sliding windows:

- TCP sliding windows are byte-oriented. The size of the window at one end is determined by the lesser of two values: *receiver window (rwnd)* or *congestion window (cwnd).*
- The source does not have to send a full window's worth of data.
- The window can be opened or closed by the receiver, but should not be shrunk.
- The destination can send an acknowledge at any time as long as it does not result in a shrinking window.
- The receiver can temporarily shut down the window; the sender, however, can always send a segment of 1 byte after the window is shut down.
- A sliding window is used to make transmission more efficient as well as to control the flow of data so that the destination does not become overwhelmed with data. It is the number of bytes the other end can accept before its buffer overflows and data are discarded. The congestion window is a value determined by the network to avoid congestion.

Error Control Mechanism

TCP is a reliable transport layer protocol. This means that an application program that delivers a stream of data to TCP relies on TCP to deliver the entire stream to the application program on the other end in order, without error, and without any part lost or duplicated.

TCP provides reliability using error control. Error control includes mechanisms for detecting corrupted segments, lost segments, out-of-order segments, and duplicated segments. Error control also includes a mechanism for correcting errors after they are detected. Error detection and correction in TCP is achieved through the use of three simple tools: checksum, acknowledgment, and time-out.

Checksum: Each segment includes a checksum field which is used to check for a corrupted segment. If the segment is corrupted, it is discarded by the destination TCP and is considered as lost. TCP uses a 16-bit checksum that is mandatory in every segment.

Acknowledgment: TCP uses acknowledgments to confirm the receipt of data segments. Control segments that carry no data but consume a sequence number are also acknowledged. ACK segments are never acknowledged. ACK segments do not consume sequence numbers and are not acknowledged.

Retransmission: The heart of the error control mechanism is the retransmission of segments. When a segment is corrupted, lost, or delayed, it is retransmitted. In modern implementations, a segment is retransmitted on two occasions: when a retransmission timer expires or when the sender receives three duplicate ACKs. In modern implementations, a retransmission occurs if the retransmission timer expires or three duplicate ACK segments have arrived.

Note that no retransmission occurs for segments that do not consume sequence numbers. In particular, there is no transmission for an ACK segment. No retransmission timer is set for an ACK segment. Retransmission after RTO A recent implementation of TCP maintains one retransmission time-out (RTO) timer for all outstanding (sent, but not acknowledged) segments.

When the timer matures, the earliest outstanding segment is retransmitted even though lack of a received ACK can be due to a delayed segment, a delayed ACK, or a lost acknowledgment. Note that no time-out timer is set for a segment that carries only an acknowledgment, which means that no such segment is resent. The value of RTO is dynamic in TCP and is updated based on the round-trip time (RTT) of segments.

An RTI is the time needed for a segment to reach a destination and for an acknowledgment to be received. It uses a back-off strategy. Sometimes, however, one segment is lost and the receiver receives so many out-of-order segments that they cannot be saved (limited buffer size). To alleviate this situation, most implementations today follow the three-duplicate-ACKs rule and retransmit the missing segment immediately.

Congestion Control in TCP

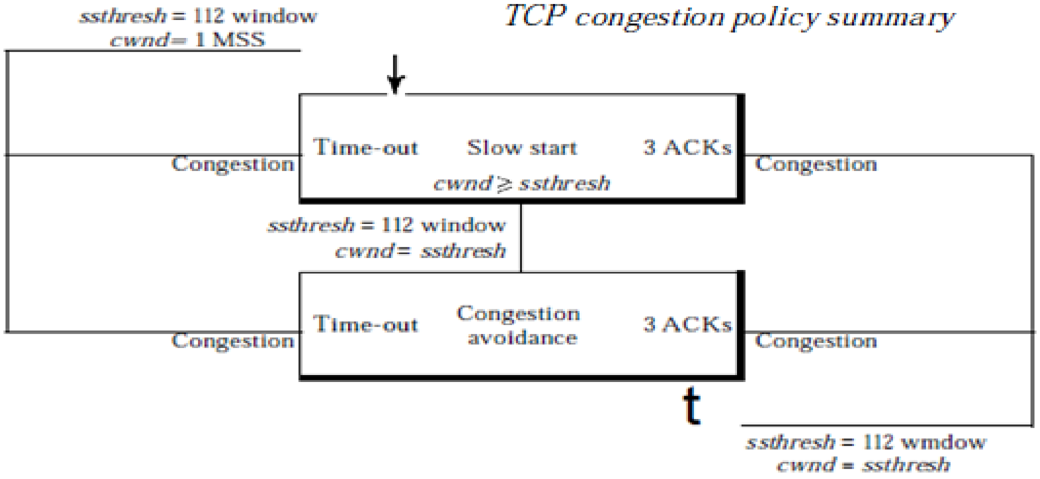
TCP uses congestion control to avoid congestion or alleviate congestion in the network.

Congestion Window

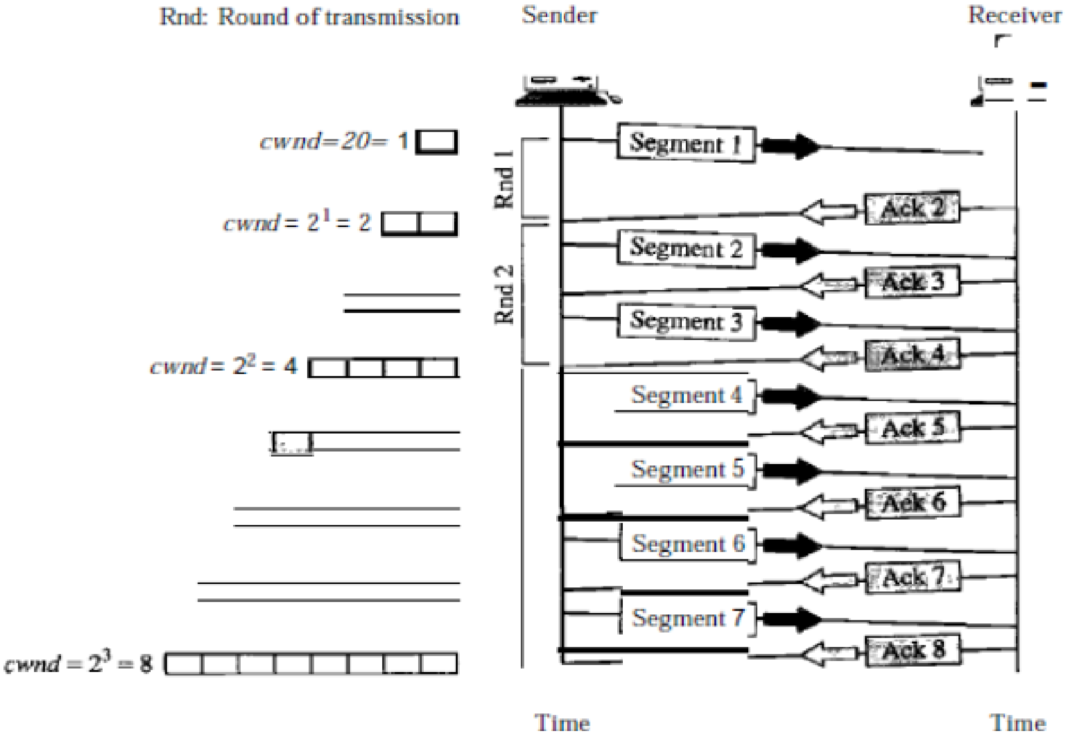
The sender window size is determined by the available buffer space in the receiver *(rwnd).* Today, the sender's window size is determined not only by the receiver but also by congestion in the network. The sender has two pieces of information: the receiver-advertised window size and the congestion window size. The actual size of the window is the minimum of these two. Actual window size = minimum (cwnd, rwnd)

Congestion Policy: TCP's general policy for handling congestion is based on three phases: Slow start, Congestion avoidance, and Congestion detection. In the slow-start phase, the sender starts with a very slow rate of transmission, but increases the rate rapidly to reach a threshold.

When the threshold is reached, the data rate is reduced to avoid congestion. Finally if congestion is detected, the sender goes back to the slow-start or congestion avoidance phase based on how the congestion is detected.


TCP congestion policy summary

Slow Start: Exponential Increase algorithm is based on the idea that the size of the congestion window *(cwnd)* starts with one maximum segment size (MSS). The MSS is determined during connection establishment by using an option of the same name. The size of the window increases one MSS each time an acknowledgment is received. As the name implies, the window starts slowly, but grows exponentially. To show the idea, let us look at figure.



After receipt of the acknowledgment for segment 1, the size of the congestion window is increased by 1, which means that *cwnd* is now 2. Now two more segments can be sent. When each acknowledgment is received, the size of the window is increased by 1 MSS. When all seven segments are acknowledged, *cwnd* = 8.

Slow start cannot continue indefinitely. There must be a threshold to stop this phase. The sender keeps track of a variable named *ssthresh* (slow-start threshold).

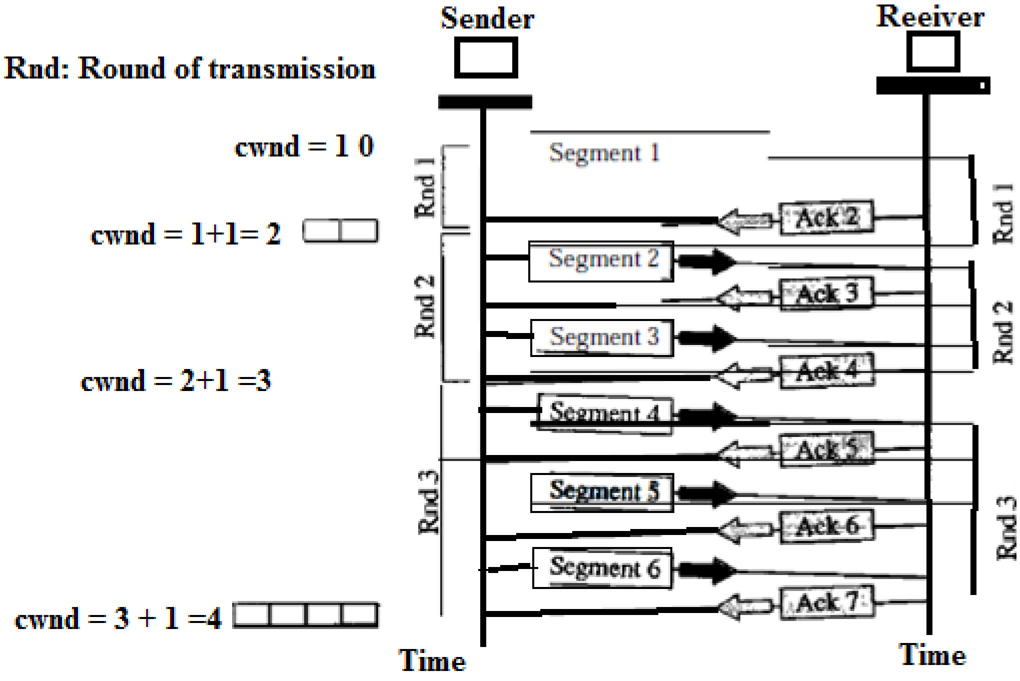
When the window size reaches the threshold Slow start stops and the next phase starts. In most implementations the value of *ssthresh* is 65,535 bytes. In the slow-start algorithm, the size of the congestion window increases exponentially until it reaches a threshold.

Congestion Avoidance using Additive Increase: If we start with the slow-start algorithm, the size of the congestion window increases exponentially. To avoid congestion before it happens, one must slow down this exponential growth. TCP defines another algorithm called congestion avoidance, which undergoes an additive increase instead of an exponential one.

When the size of the congestion window reaches the slow-start threshold, the slow-start phase stops and the additive phase begins. In this algorithm, each time the whole window of segments is acknowledged (one round), the size of the congestion window is increased by 1.

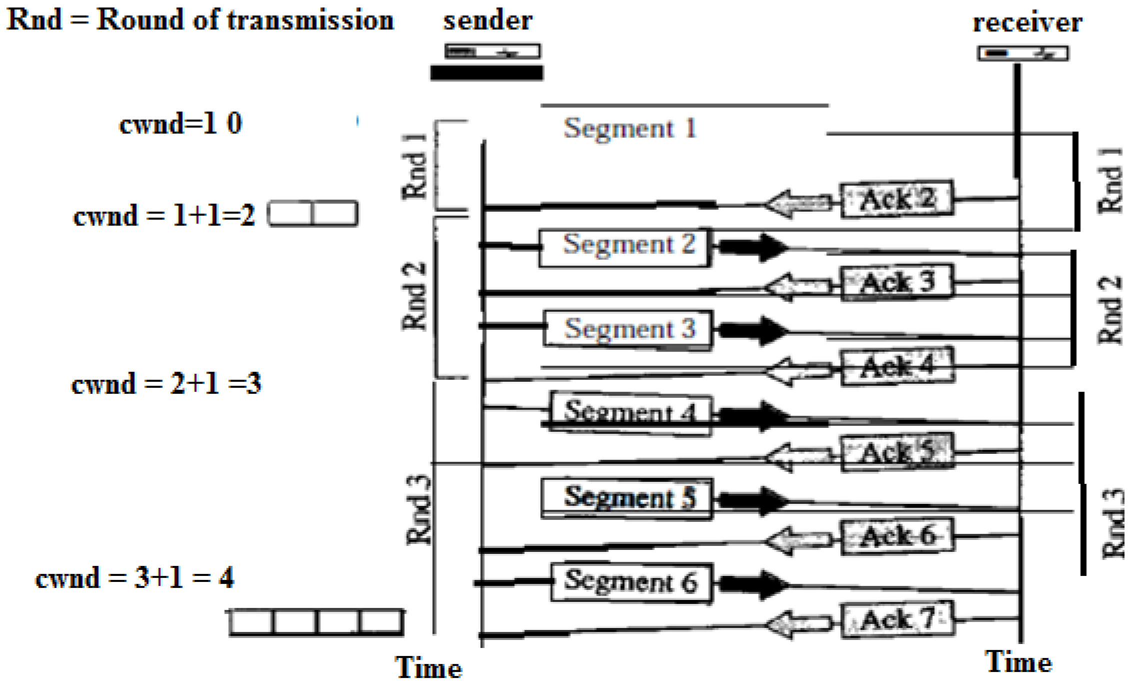
To show the idea, we apply this algorithm to the same scenario as slow start, although we will see that the congestion avoidance algorithm usually starts when the size of the window is much greater than 1. Figure shows the idea. In the congestion avoidance algorithm, the size of the congestion window increases additively until congestion is detected.

Congestion Avoidance using Additive increase



Congestion Detection using Multiplicative Decrease: If congestion occurs, the congestion window size must be decreased. The only way the sender can guess that congestion has occurred is by the need to retransmit a segment.

Congestion Detection Multiplicative Increase



However, retransmission can occur in one of two cases: when a timer times out or when three ACKs are received. In both cases, the size of the threshold is dropped to one-half, a multiplicative decrease. Most TCP implementations have two reactions:

1. If a time-out occurs, there is a stronger possibility of congestion; a segment has probably been dropped in the network, and there is no news about the sent segments. In this case TCP reacts strongly:
   a. It sets the value of the threshold to one-half of the current window size.
   b. It sets *cwnd* to the size of one segment.
   c. It starts the slow-start phase again.
2. If three ACKs are received, there is a weaker possibility of congestion; a segment may have been dropped, but some segments after that may have arrived safely since three ACKs are received. This is called fast transmission and fast recovery. In this case, TCP has a weaker reaction:a. It sets the value of the threshold to one-half of the current window size.
   b. It sets *cwnd* to the value of the threshold.
   c. It starts the congestion avoidance phase.

Implementations react to congestion detection in one of the following ways:

If detection is by time-out, a new *slow-start* phase starts.
If detection is by three ACKs, a new *congestion avoidance* phase starts.

Unit V (Security)

Introduction

No one can deny the importance of security in data communications and networking. Security in networking is based on cryptography, the science and art of transforming messages to make them secure and immune to attack. Cryptography can provide several aspects of security related to the interchange of messages through networks. These aspects are confidentiality, integrity, authentication, and non-repudiation.

Cryptography

Cryptography comes from the Greek words for "secret writing." The messages to be encrypted, known as the **plaintext**, are transformed by a function that is parameterized by a **key**.

- Cryptography can provide confidentiality, integrity, authentication, and non-repudiation of messages.
- Cryptography can also be used to authenticate the sender and receiver of the message to each other.

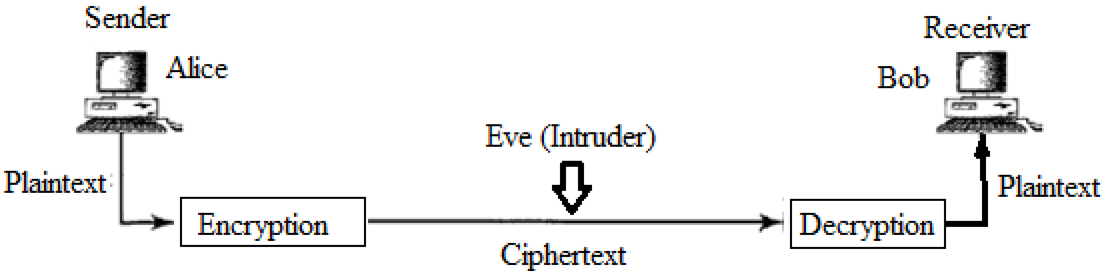


Figure 5.1. The encryption model (for a symmetric-key cipher).

Plaintext and Ciphertext: The original message, before being transformed, is called plaintext. After the message is transformed, it is called ciphertext. An encryption algorithm transforms the plaintext into ciphertext; a decryption algorithm transforms the ciphertext back into plaintext. The sender uses an encryption algorithm, and the receiver uses a decryption algorithm.

Cipher: We refer to encryption and decryption algorithms as ciphers. The term *cipher* is also used to refer to different categories of algorithms in cryptography. This is not to say that every sender-receiver pair needs their very own unique cipher for a secure communication.

Key: A key is a number (or a set of numbers) that the cipher, as an algorithm, operates on. To encrypt a message, we need an encryption algorithm, an encryption key, and the plaintext. These create the ciphertext. To decrypt a message, we need a decryption algorithm, a decryption key, and the ciphertext. These reveal the original plaintext.

Alice, Bob, and Eve: In cryptography, it is customary to use three characters in an information exchange scenario; we use Alice, Bob, and Eve. Alice is the person who needs to send secure data. Bob is the recipient of the data. Eve is the person who somehow disturbs the communication between Alice and Bob by intercepting messages to uncover the data or by sending her own disguised messages. These three names represent computers or processes that actually send or receive data, or intercept or change data.

Sometimes the intruder can not only listen to the communication channel (passive intruder) but can also record messages and play them back later, inject his own messages, or modify legitimate messages before they get to the receiver (active intruder).

The art of breaking ciphers, called **cryptanalysis**, and the art devising them (cryptography) is collectively known as **cryptology**.

Chiper Text (Encryption Method)

The output of the encryption process, known as the **cipher text**, is then transmitted, often by messenger or radio. Encryption methods have historically been divided into two categories: substitution ciphers and transposition ciphers.

Substitution Ciphers: In a substitution cipher each letter or group of letters is replaced by another letter or group of letters to disguise it. One of the oldest known ciphers is the **Caesar cipher**, attributed to Julius Caesar.

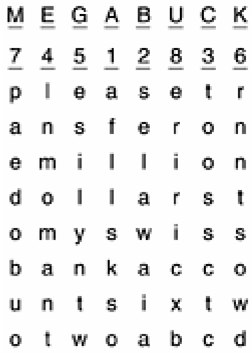
A slight generalization of the Caesar cipher allows the cipher text alphabet to be shifted by *k* letters, instead of always 3. In this case *k* becomes a key to the general method of circularly shifted alphabets. For example,

Plaintext: a b c d e f g h i j k l m n o p q r s t u v w x y z

Cipher text: Q W E R T Y U I O P A S D F G H J K L Z X C V B N M

The general system of symbol-for-symbol substitution is called a **mono alphabetic substitution**, with the key being the 26-letter string corresponding to the full alphabet. For the key above, the plaintext *attack* would be transformed into the cipher text *QZZQEA*.

Transposition Ciphers: Substitution ciphers preserve the order of the plaintext symbols but disguise them. **Transposition ciphers**, in contrast, reorder the letters but do not disguise them. The cipher is keyed by a word or phrase not containing any repeated letters. In this example, MEGABUCK is the key. The purpose of the key is to number the columns, column 1 being under the key letter closest to the start of the alphabet, and so on. The plaintext is written horizontally, in rows, padded to fill the matrix if need be. The ciphertext is read out by columns, starting with the column whose key letter is the lowest.



M	E	G	A	B	U	C	K
7	4	5	1	2	8	3	6
p	l	e	a	s	e	t	r
a	n	s	f	e	r	o	n
e	m	i	l	l	i	o	n
d	o	l	l	a	r	s	t
o	m	y	s	w	i	s	s
b	a	n	k	a	c	c	o
u	n	t	s	i	x	t	w
o	t	w	o	a	b	c	d

Figure 5- 5. A transposition cipher.

Plaintext

pleasetransferonemilliondollarsto
myswissbankaccountsixtwotwo

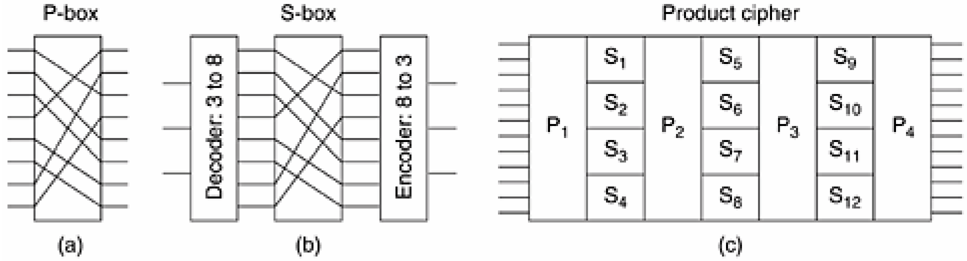
Ciphertext

AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
ESILYNTWRNNTSOWDPAEDOBUOERIRICXB

To break a transposition cipher, the cryptanalyst must first be aware that he is dealing with a transposition cipher. By looking at the frequency of *E*, *T*, *A*, *O*, *I*, *N*, etc., it is easy to see if they fit the normal pattern for plaintext. If so, the cipher is clearly a transposition cipher, because in such a cipher every letter represents itself, keeping the frequency distribution intact.

Symmetric-Key Algorithms

The first class of encryption algorithms is called **symmetric-key algorithms** because they used the same key for encryption and decryption. Fig. 5-7 illustrates the use of a symmetric-key algorithm. In particular, we will focus on **block ciphers**, which take an *n*-bit block of plaintext as input and transform it using the key into *n*-bit block of ciphertext.



Basic elements of product ciphers. (a) P-box. (b) S-box. (c) Product

Cryptographic algorithms can be implemented in either hardware (speed) or in software (flexibility). Transpositions and substitutions can be implemented with simple circuits. Figure (a) shows a device, known as a **P-box** (P-permutation), used to effect a transposition on an 8-bit input.

Substitutions are performed by **S-boxes**, as shown in Fig.(b). In this example a 3-bit plaintext is entered and a 3-bit ciphertext is output. The second stage is a P-box. The third stage encodes the selected input line in binary again.

The real power of these basic elements only becomes apparent when we cascade a whole series of boxes to form a **product cipher**, as shown in Fig.(c). Product ciphers that operate on *k*-bit inputs to produce *k*-bit outputs are very common. Typically, *k* is 64 to 256.

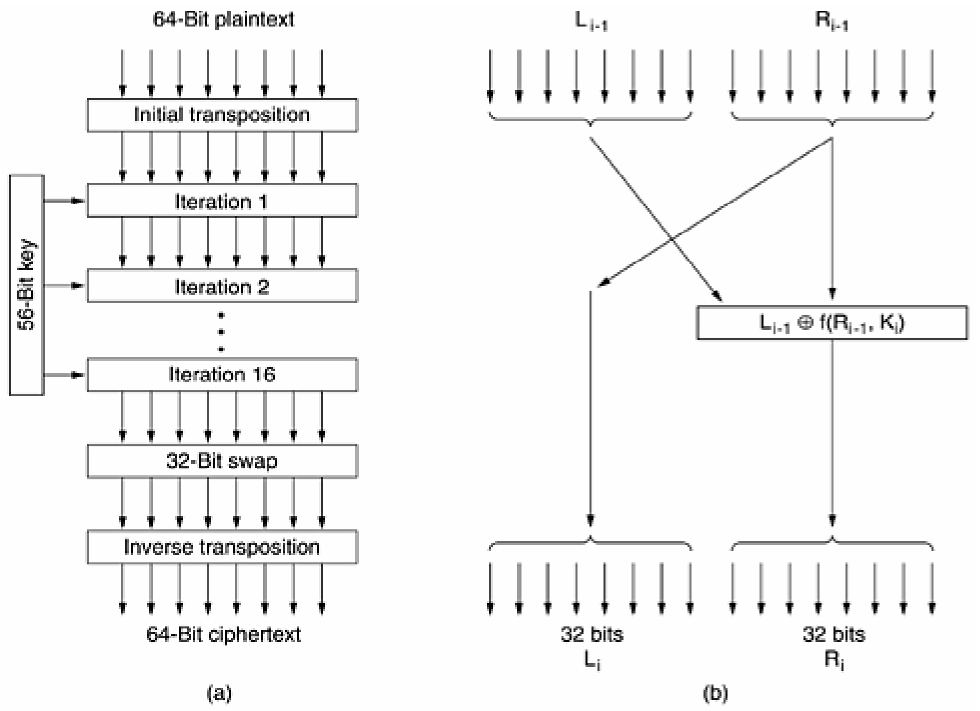
DES-Data Encryption Standard

In January 1977, the U.S. Government adopted a product cipher developed by IBM as its official standard for unclassified information. This cipher, **DES** (**Data Encryption Standard**), was widely adopted by the industry for use in security products. It is no longer secure in its original form, but in a modified form it is still useful. An outline of DES is shown in Fig.(a).

Plaintext is encrypted in blocks of 64 bits, yielding 64 bits of ciphertext. The algorithm, which is parameterized by a 56-bit key, has 19 distinct stages. The first stage is a key-independent transposition on the 64-bit plaintext. The last stage is the exact inverse of this transposition.

Fig.(b). illustrates that each stage takes two 32-bit inputs and produces two 32-bit outputs. The left output is simply a copy of the right input. The right output is the bitwise XOR of the left input and a function of the right input and the key for this stage, *Ki. The* output is then partitioned into eight groups of 6 bits each, each of which is fed into a different S-box. Each of the 64 possible inputs to an S- box is mapped onto a 4-bit output. Finally, these 8 x 4 bits are passed through a P-box.

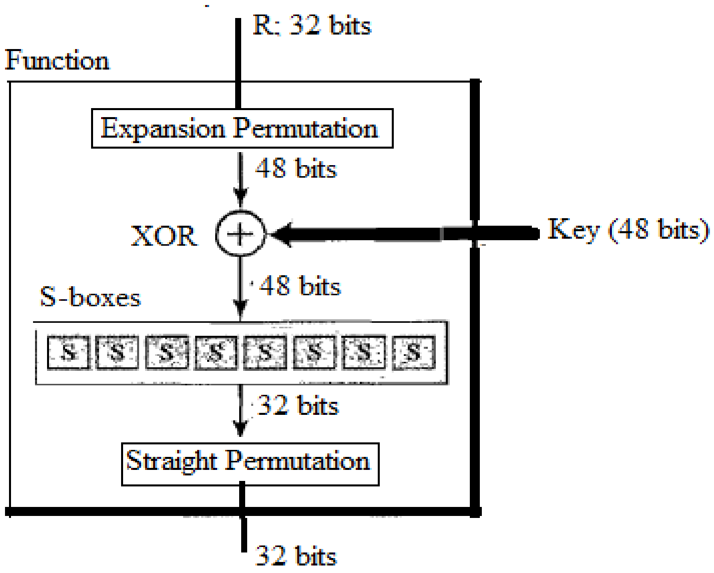
In each of the 16 iterations, a different key is used. Before the algorithm starts, a 56-bit transposition is applied to the key. Just before each iteration, the key is partitioned into two 28-bit units, each of which is rotated left by a number of bits dependent on the iteration number. *Ki* is derived from this rotated key by applying yet another 56-bit transposition to it. A different 48-bit subset of the 56 bits is extracted and permuted on each round.



The data encryption standard. (a) General outline. (b) Detail of one iteration.

A technique that is sometimes used to make DES stronger is called **whitening**. It consists of XORing a random 64-bit key with each plaintext block before feeding it into DES and then XORing a second 64-bit key with the resulting ciphertext before transmitting it.

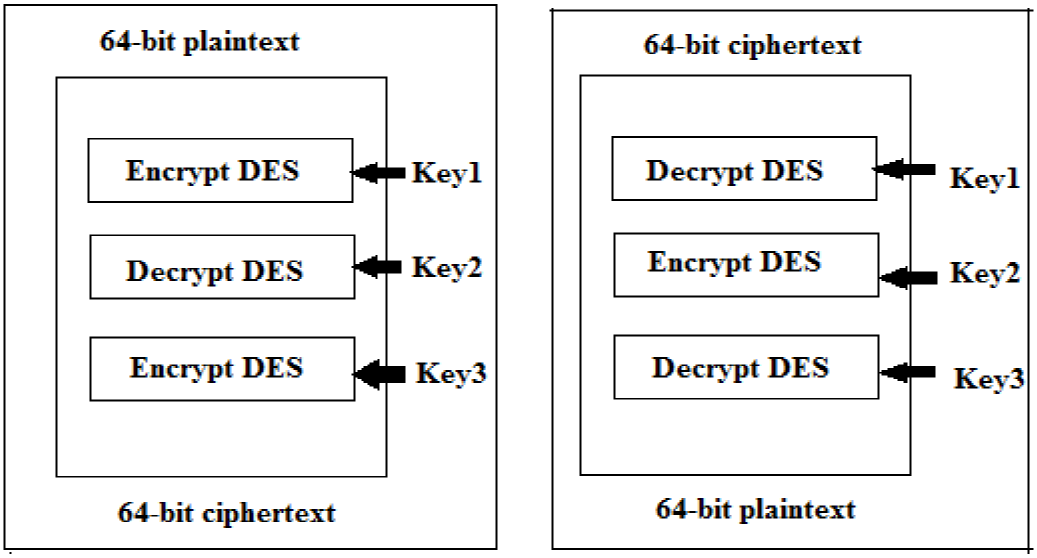
DES Function: The heart of DES is the **DES function.** The DES function applies a 48-bit key to the rightmost 32 bits *Ri* to produce a 32-bit output. This function is made up of four operations: an XOR, an expansion permutation, a group of S-boxes, and a straight permutation, as shown in Figure 30.15.



DES has two transposition blocks (P-boxes) and 16 complex round ciphers (they are repeated). Although the 16 iteration round ciphers are conceptually the same, each uses a different key derived from the original key. The initial and final permutations are keyless straight permutations that are the inverse of each other. The permutation takes a 64-bit input and permutes them according to predefined values.

Triple DES

As early as 1979, IBM realized that the DES key length was too short and devised a way to effectively increase it, using triple encryption (Tuchman, 1979). The method chosen, which has been incorporated in International Standard 8732, is illustrated in Fig. Here two keys and three stages are used. In the first stage, the plaintext is encrypted using DES in the usual way with *K*1. In the second stage, DES is run in decryption mode, using K_2 as the key. Finally, another DES encryption is done with *K*1.



a. Encryption Triple DES

b. Decryption Triple DES

Critics of DES contend that the key is too short. To lengthen the key, Triple DES or 3DES has been proposed and implemented. This uses three DES blocks, as shown in Figure. Note that the encrypting block uses an encryption-decryption-encryption combination of DESs, while the decryption block uses a decryption-encryption-decryption combination.

Two different versions of 3DES are in use: 3DES with two keys and 3DES with three keys. To make the key size 112 bits and at the same time protect DES from attacks such as the man-in-the-middle attack, 3DES with two keys was designed. In this version, the first and the third keys are the same (KeYl = KeY3)' This has the advantage in that a text encrypted by a single DES block can be decrypted by the new 3DES. We just set all keys equal to KeYl' Many algorithms use a 3DES cipher with three keys. This increases the size of the key to 168 bits.

AES-Advanced Encryption Standard

The Advanced Encryption Standard (AES) was designed because DES's key was too small. Although Triple DES ODES) increased the key size, the process was too slow. The National Institute of Standards and Technology (NIST) chose the Rijndael algorithm, named after its two Belgian inventors, Vincent Rijmen and Joan Daemen, as the basis of AES.

AES is a very complex round cipher. AES is designed with three key sizes: 128, 192, or 256 bits. In November 2001 Rijndael became a U.S. Government standard. Rijndael supports key lengths and block sizes from 128 bits to 256 bits in steps of 32 bits. The key length and block length may be chosen independently. However, AES specifies that the block size must be 128 bits and the key length must be 128, 192, or 256 bits.

Rijndael

From a mathematical perspective, Rijndael is based on Galois field theory, which gives it some provable security properties. However, it can also be viewed as C code, without getting into the mathematics. Like DES, Rijndael uses substitution and permutations, and it also uses multiple rounds. The number of rounds depends on the key size and block size, being 10 for 128-bit keys with 128- bit blocks and moving up to 14 for the largest key or the largest block. The function *rijndael* has three parameters. They are: *plaintext*, an array of 16 bytes containing the input data, *ciphertext*, an array of 16 bytes where the enciphered output will be returned, and *key*, the 16-byte key. The *state* array is initialized to the plaintext and modified by every step in the computation. In some steps, byte-for-byte substitution is performed. In others, the bytes are permuted within the array. Other transformations are also used. At the end, the contents of the *state* are returned as the ciphertext.

The code starts out by expanding the key into 11 arrays of the same size as the state. They are stored in *rk*, which is an array of structs, each containing a state array. One of these will be used at the start of the calculation and the other 10 will be used during the 10 rounds, one per round. The calculation of the round keys from the encryption key is too complicated for us to get into here. Suffice it to say that the round keys are produced by repeated rotation and XORing of various groups of key bits. For all the details, see (Daemen and Rijmen, 2002).

The next step is to copy the plaintext into the *state* array so it can be processed during the rounds. It is copied in column order, with the first four bytes going into column 0, the next four bytes going into column 1, and so on. Both the columns and the rows are numbered starting at 0, although the rounds are numbered starting at 1. This initial setup of the 12 byte arrays of size 4 x 4 is illustrated in Fig. 5- 10.

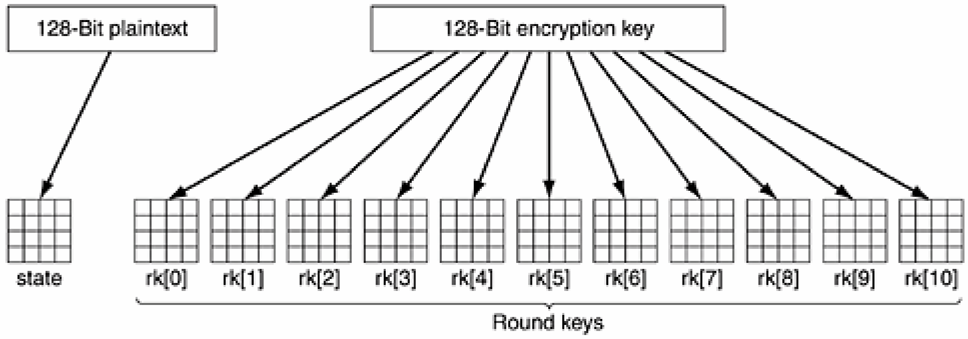


Figure 5-10. Creating of the state and rk arrays.

There is one more step before the main computation begins: *rk*[0] is XORed into *state* byte for byte. In other words each of the 16 bytes in *state* is replaced by the XOR of itself and the corresponding byte in *rk*[0]. The loop executes 10 iterations, one per round, and transforming *state* on each iteration. The contents of each round consist of four steps.

Step 1: does a byte-for-byte substitution on *state*. Each byte in turn is used as an index into an S-box to replace its value by the contents of that S-box entry.

Step 2: rotates each of the four rows to the left. Row 0 is rotated 0 bytes (i.e., not changed), row 1 is rotated 1 byte, row 2 is rotated 2 bytes, and row 3 is rotated 3 bytes.

Step 3: mixes up each column independently of the other ones.

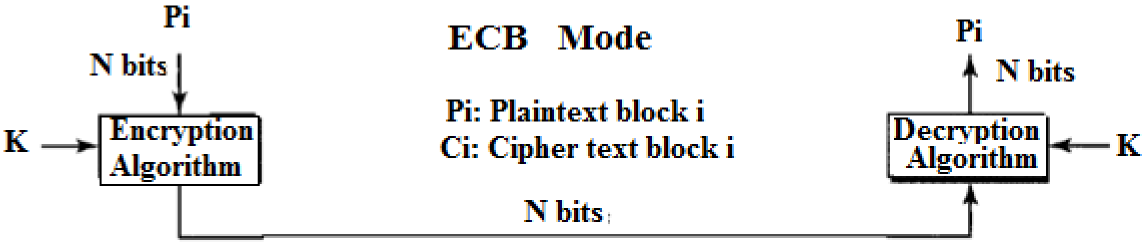
Step 4: XORs the key for this round into the *state* array.

Cipher Modes of Operation

Despite all this complexity, AES (or DES or any block cipher for that matter) is basically a mono alphabetic substitution cipher using big characters (128-bit characters for AES and 64-bit characters for DES). A mode of operation is a technique that employs the modern ciphers such as the electronic code book (ECB), cipher block chaining (CBC), cipher feedback (CFB) mode, **output** feedback (OFB) mode.

Electronic Code Book

The electronic code book (ECB) mode is a purely block cipher technique. The plaintext is divided into blocks of *N* bits. The cipher text is made of blocks of *N* bits. The value of *N* depends on the type of cipher used. Figure shows the method.



We mention four characteristics of this mode:

1. Because the key and the encryption/decryption algorithm are the same, equal blocks in the plaintext become equal blocks in the cipher text. For example, if plaintext blocks 1, 5, and 9 are the same, cipher text blocks I, 5, and 9 are also the same. This can be a security problem; the adversary can guess that the plaintext blocks are the same if the corresponding cipher text blocks are the same.

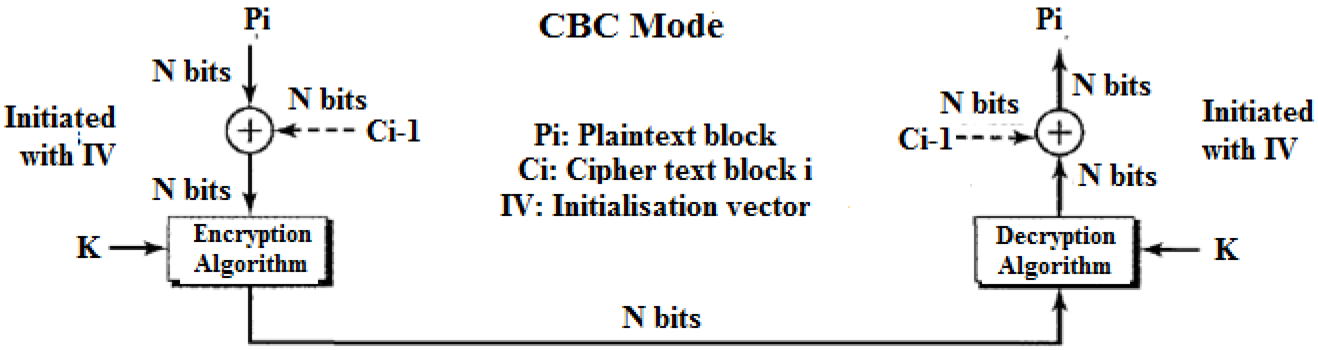
2. If we reorder the plaintext block, the cipher text is also reordered.

3. Blocks are independent of each other. Each block is encrypted or decrypted independently. A problem in encryption or decryption of a block does not affect other blocks.

4. An error in one block is not propagated to other blocks. If one or more bits are corrupted during transmission, it only affects the bits in the corresponding plaintext after decryption. Other plaintext blocks are not affected. This is a real advantage if the channel is not noise-free.

Cipher Block Chaining

The cipher block chaining (CBC) mode tries to alleviate some of the problems in ECB by including the previous cipher block in the preparation of the current block. When a block is completely enciphered, the block is sent, but a copy of it is kept in a register (a place where data can be held) to be used in the encryption of the next block. The reader may wonder about the initial block. There is no cipher text block before the first block. In this case, a phony block called the initiation vector (IV) is used. Both the sender and receiver agree upon a specific predetermined IV.



The Following are some characteristics of CBC.

1. Even though the key and the encryption/decryption algorithm are the same, equal blocks in the plaintext do not become equal blocks in the ciphertext. For example, if plaintext blocks 1, 5, and 9 are the same, cipher text blocks I, 5, and 9 will not be the same. An adversary will not be able to guess from the cipher text that two blocks are the same.

2. Blocks are dependent on each other. Each block is encrypted or decrypted based on a previous block. A problem in encryption or decryption of a block affects other blocks.

3. The error in one block is propagated to the other blocks. If one or more bits are corrupted during the transmission, it affects the bits in the next blocks of the plaintext after decryption.

Cipher Feedback

The cipher feedback (CFB) mode was created for those situations in which we need to send or receive *r* bits of data, where *r* is a number different from the underlying block size of the encryption cipher used. The value of *r* can be 1, 4, 8, or any number of bits. Since all block ciphers work on a block of data at a time, the problem is how to encrypt just *r* bits. The solution is to let the cipher encrypt a block of bits and use only the first *r* bits as a new key (stream key) to encrypt the *r* bits of user data.

The following are some characteristics of the CFB mode:

1. If we change the IV from one encryption to another using the same plaintext, the Cipher text is different.

2. The cipher text Ci depends on both *Pi* and the preceding cipher text block.

3. Errors in one or more bits of the cipher text block affect the next cipher text blocks.

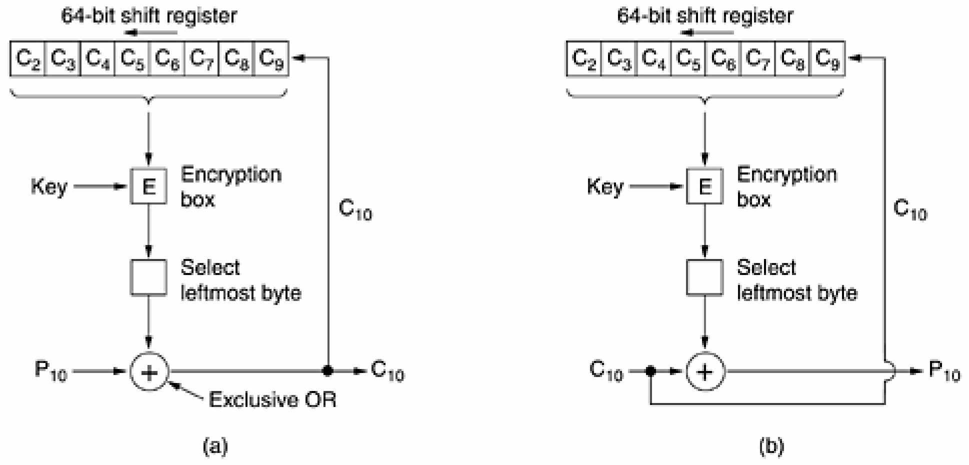
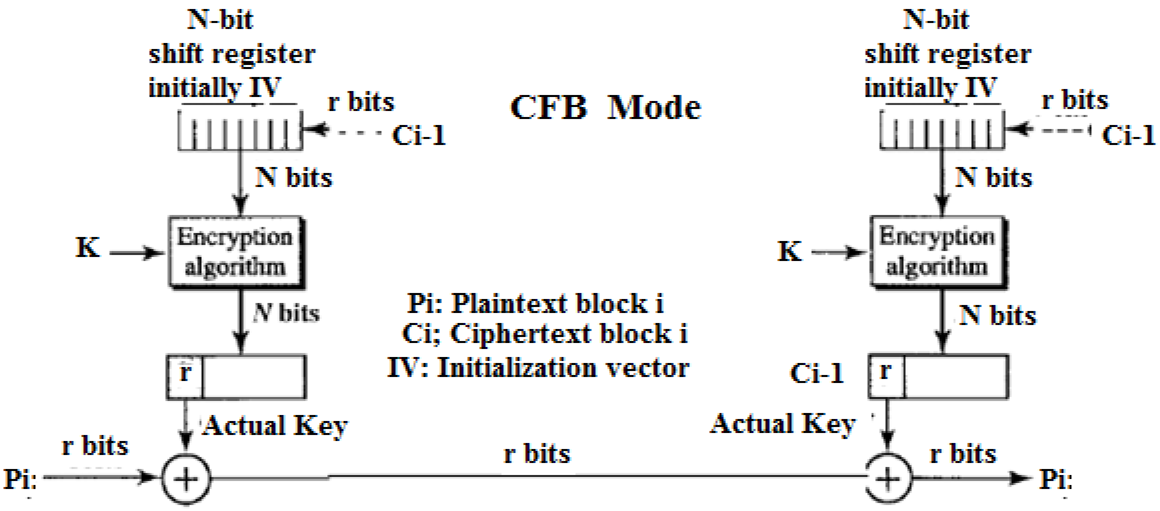


Figure 5-12. Cipher feedback mode. (a) Encryption. (b) Decryption.

Output Feedback

The **output** feedback (OFB) mode is very similar to the CFB mode with one difference. Each bit in the cipher text is independent of the previous bit or bits. This avoids error propagation. If an error occurs in transmission, it does not affect the future bits. Note that, as in CFB, both the sender and the receiver use the encryption algorithm.

Note also that in OFB, block ciphers such as DES or AES can only be used to create the key stream. The feedback for creating the next bit stream comes from the previous bits of the key stream instead of the cipher text. The cipher text does not take part in creating the key stream.

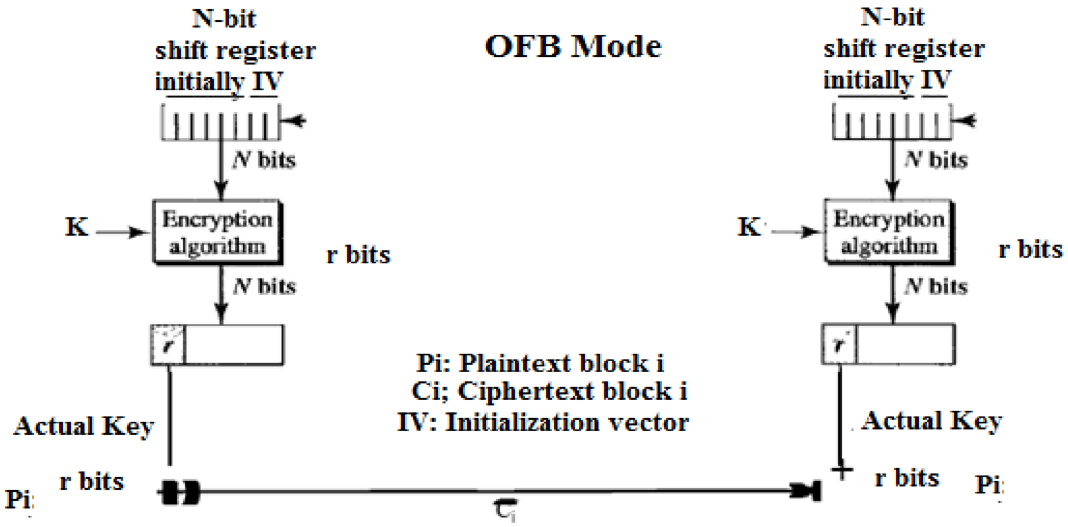


The following are some of the characteristics of the OFB mode.

1. If we change the IV from one encryption to another using the same plaintext, the Cipher text will be different.

2. The cipher text C_i depends on the plaintext *Pi'*

3. Errors in one or more bits of the cipher text do not affect future cipher text blocks.



Public-Key Algorithms

Public-key cryptography requires each user to have two keys: a public key, used by the entire world for encrypting messages to be sent to that user, and a private key, which the user needs for decrypting messages. We will consistently refer to these keys as the *public* and *private* keys, respectively, and distinguish them from the *secret* keys used for conventional symmetric-key cryptography.

RSA

The RSA method is based on some principles from number theory. We will now summarize how to use the method; for details, consult the paper.

1. Choose two large primes, p and q (typically 1024 bits).
2. Compute $n = p$ x q and $z = (p - 1)$ x $(q - 1)$.
3. Choose a number relatively prime to z and call it d.
4. Find e such that e x $d = 1$ *mod z.*

Divide the plaintext (regarded as a bit string) into blocks, so that each plaintext message, P, falls in the interval $0 \leq P < n$. Do that by grouping the plaintext into blocks of k bits, where k is the largest integer for which $2^k < n$is true.

To encrypt a message, P, compute $C = P^e$ (mod n). To decrypt C, compute $P = C^d$ (mod n). It can be proven that for all P in the specified range, the encryption and decryption functions are inverses. To perform the encryption, you need e and n. To perform the decryption, you need d and n. Therefore, the public key consists of the pair (e, n), and the private key consists of (d, n).

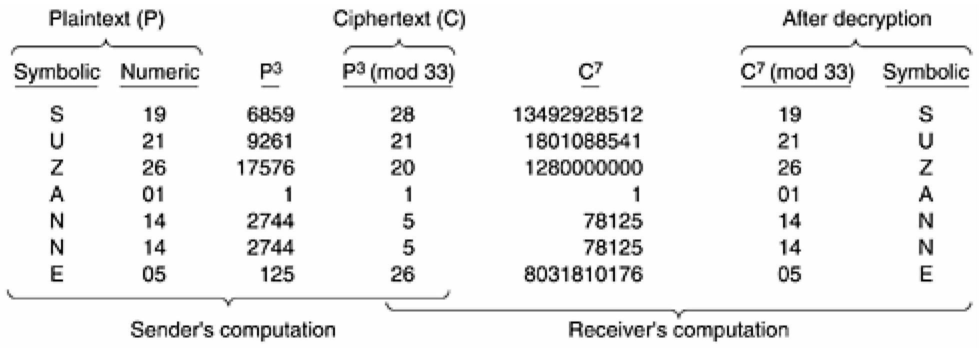
The security of the method is based on the difficulty of factoring large numbers. If the cryptanalyst could factor the (publicly known) n, he could then find p and q, and from these z. Equipped with knowledge of z and e, d can be found using Euclid's algorithm.

Fortunately, mathematicians have been trying to factor large numbers for at least 300 years, and the accumulated evidence suggests that it is an exceedingly difficult problem.

A trivial pedagogical example of how the RSA algorithm works is given in table 5-8. For this example we have chosen $p = 3$ and $q = 11$, giving $n = 33$ and $z = 20$. A suitable value for d is $d = 7$, since 7 and 20 have no common factors.

With these choices, e can be found by solving the equation $7e = 1$ (mod 20), which yields $e = 3$. The ciphertext, C, for a plaintext message, P, is given by $C = P^{3\ (}$mod 33). The ciphertext is decrypted by the receiver by making use of the rule $P = C^{7\ (}$mod 33). The figure shows the encryption of the plaintext "SUZANNE" as an example.

Table 5- 8 An example of the RSA algorithm.



Plaintext (P)			Ciphertext (C)		After decryption	
Symbolic	Numeric	P^3	P^3 (mod 33)	C^7	C^7 (mod 33)	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E

Sender's computation

Receiver's computation

Because the primes chosen for this example are so small, *P* must be less than 33, so each plaintext block can contain only a single character. The result is a monoalphabetic substitution cipher, not very impressive. If instead we had chosen *p* and $q = 2^{512}$, we would have $n = 2^{1024}$, so each block could be up to 1024 bits or 128 eight-bit characters, versus 8 characters for DES and 16 characters for AES.

Restriction

For RSA to work, the value of *P* must be less than the value of *n.* If *P* is a large number, the plaintext needs to be divided into blocks to make *P* less than *n.*

Applications

- Although RSA can be used to encrypt and decrypt actual messages, it is very slow if the message is long. RSA, therefore, is useful for short messages such as a small message or a symmetric key to be used for a symmetric-key cryptosystem.
- RSA is used in digital signatures and other cryptosystems that often need to encrypt a small message without having access to a symmetric key.
- RSA is also used for authentication.

Digital Signatures

The authenticity of many legal, financial, and other documents is determined by the presence or absence of an authorized handwritten signature. And photocopies do not count. For computerized message systems to replace the physical transport of paper and ink documents, a method must be found to allow documents to be signed in an un-forgeable way.

Need for Keys

In conventional signature a signature is like a private "key" belonging to the signer of the document. The signer uses it to sign a document; no one else has this signature. The copy of the signature is on file like a public key; anyone can use it to verify a document, to compare it to the original signature.

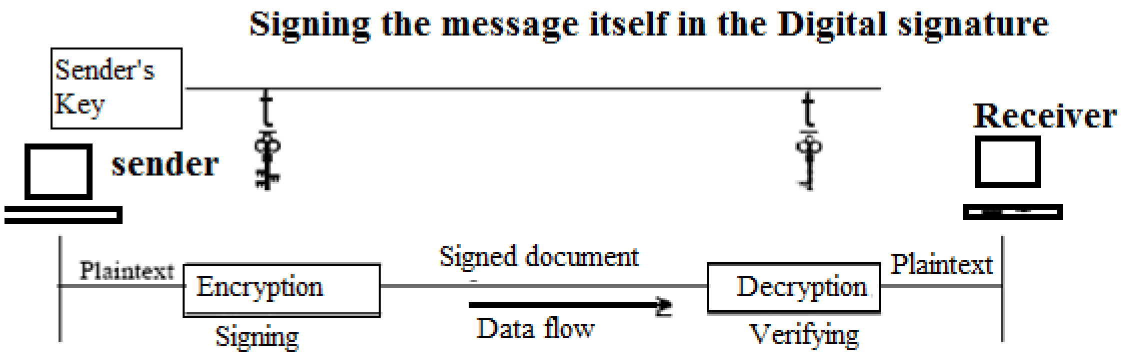
In digital signature, the signer uses her private key, applied to a signing algorithm, to sign the document. The verifier, on the other hand, uses the public key of the signer, applied to the verifying algorithm, to verify the document. A digital signature needs a public-key system.

Process

Digital signature can be achieved in two ways: signing the document or signing a digest of the document.

Signing the Document

Probably, the easier, but less efficient way is to sign the document itself. Signing a document is encrypting it with the private key of the sender; verifying the document is decrypting it with the public key of the sender. Figure 31.11 shows how signing and verifying are done.



We should make a distinction between private and public keys as used in digital signature and public and private keys as used for confidentiality. In the latter, the private and public keys of the receiver are used in the process. The sender uses the public key of the receiver to encrypt; the receiver uses his own private key to decrypt.

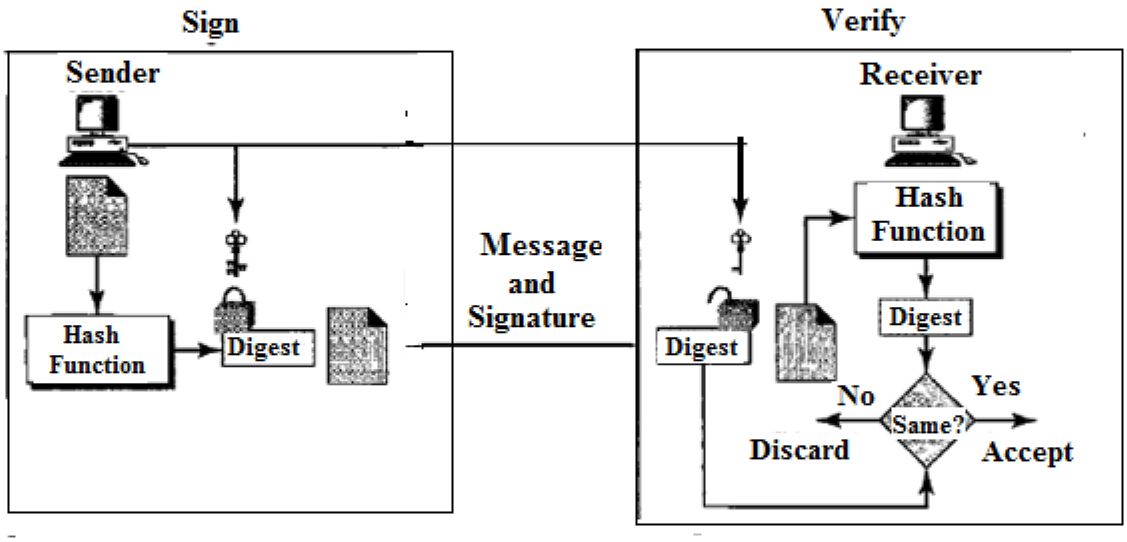
In digital signature, the private and public keys of the sender are used. The sender uses her private key; the receiver uses the public key of the sender. In a cryptosystem, we use the private and public keys of the receiver; in digital signature, we use the private and public key of the sender.

Signing the Digest

We mentioned that the public key is very inefficient in a cryptosystem if we are dealing with long messages. In a digital signature system, our messages are normally long, but we have to use public keys. The solution is not to sign the message itself; instead, we sign a digest of the message. Figure 31.12 shows signing a digest in a digital signature system.

A digest is made out of the message at sender's site. The digest then goes through the signing process using sender's private key. Sender then sends the message and the signature to Receiver.

Signing the Digest in a Digital Signature



Message Digests

This scheme is based on the idea of a one-way hash function that takes an arbitrarily long piece of plaintext and from it computes a fixed-length bit string. This hash function, *MD*, often called a **message digest**, has four important properties:

1. Given *P*, it is easy to compute *MD*(*P*).
2. Given *MD*(*P*), it is effectively impossible to find *P*.
3. Given *P* no one can find *P*' such that *MD* (*P*') = *MD*(*P*).
4. A change to the input of even 1 bit produces a very different output.

A variety of message digest functions have been proposed. The most widely used ones are MD5 (Rivest, 1992) and SHA-1 (NIST, 1993). **MD5** is the fifth in a series of message digests designed by Ronald Rivest. It operates by mangling bits in a sufficiently complicated way that every output bit is affected by every input bit.

At Receiver's site, using the same public hash function, a digest is first created out of the received message. Calculations are done on the signature and the digest. The verifying process also applies criteria on the result of the calculation to determine the authenticity of the signature. If authentic, the message is accepted; otherwise, it is rejected.

Services provided by Digital Signature

A digital signature can provide three out of the five services we mentioned for a security system: message integrity, message authentication, and non repudiation. Note that a digital signature scheme does not provide confidential communication. If confidentiality is required, the message and the signature must be encrypted using either a secret-key or public-key cryptosystem.

1. **Message Integrity**

   The integrity of the message is preserved even if we sign the whole message because we cannot get the same signature if the message is changed. The signature schemes today use a hash function in the signing and verifying algorithms that preserve the integrity of the message. A digital signature today provides message integrity.

2. **Message Authentication**

   A secure signature scheme, like a secure conventional signature (one that cannot be easily copied), can provide message authentication. Receiver can verify that the message is sent by Sender because sender's public key is used in verification. Sender's public key cannot create the same signature as intruder's private key. Digital signature provides message authentication.

3. **Message Non repudiation**

   If sender signs a message and then denies it, can receiver later prove that sender actually signed it? Receiver must keep the signature on file and later use sender's public key to create the original message to prove the message in the file and the newly created message are the same; this is not feasible because sender may have changed her private/public key during this time; sender may also claim that the file containing the signature is not authentic.

Signature Schemes

Several signature schemes have evolved during the last few decades. Some of them have been implemented such as RSA and DSS (Digital Signature Standard) schemes.

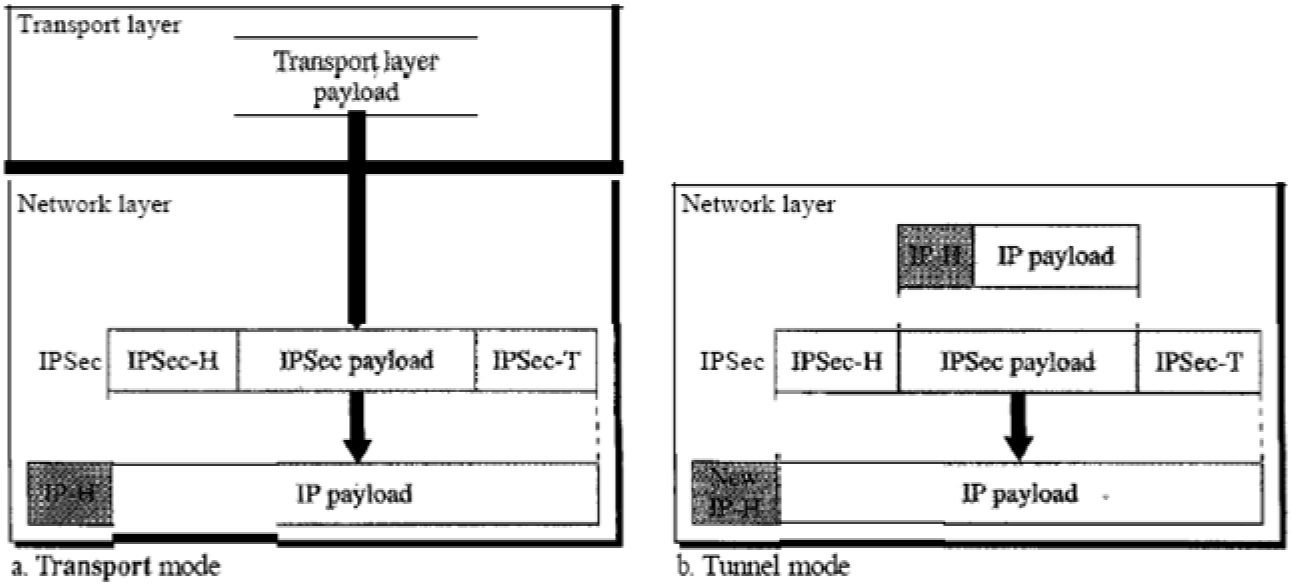
Security in the Internet: IPSec, PGP, VPN, and Firewalls

We briefly show how the IPSec protocol can add authentication and confidentiality to the IP protocol, how SSL (or TLS) can do the same for the TCP protocol, and how PGP can do it for the SMTP protocol (e-mail). In all these protocols, there are some common issues that we need to consider. First, we need to create a MAC. Then we need to enclose the message and, probably, the MAC.

IP Security (IPSec)

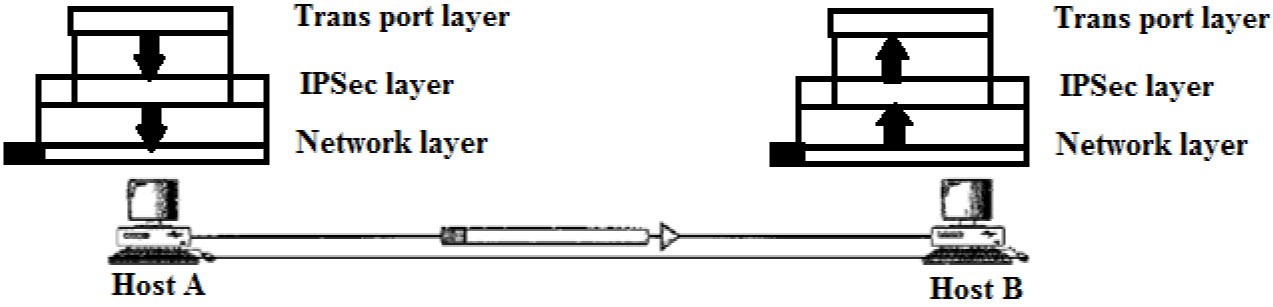
IP Security (IPSec) is a collection of protocols designed by the Internet Engineering Task Force (IETF) to provide security for a packet at the network level. IPSec helps to create authenticated and confidential packets for the IP layer.

Two Modes: IPSec operates in one of two different modes: the transport mode or the tunnel mode.



Transport Mode

In the transport mode, IPSec protects what is delivered from the transport layer to the network layer. In other words, the transport mode protects the network layer payload, the payload to be encapsulated in the network layer.



Note that the transport mode does not protect the whole IP packet; it protects only the packet from the transport layer (the IP layer payload). In this mode, the IPSec header and trailer are added to the information corning from the transport layer. The IP header is added later.

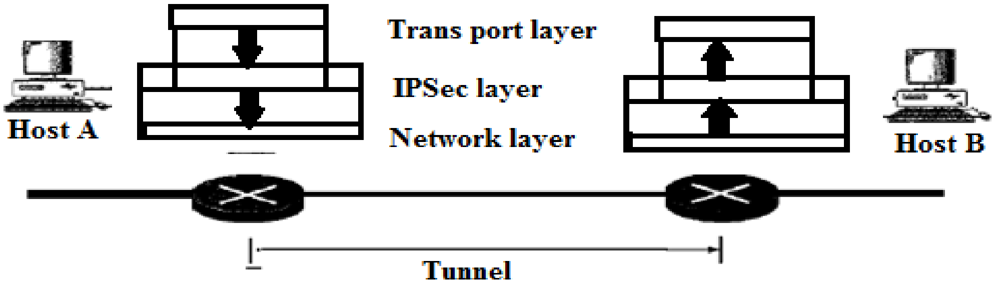
The transport mode is normally used when we need host-to-host (end-to-end) protection of data. The sending host uses IPSec to authenticate and/or encrypt the payload delivered from the transport layer.

The receiving host uses IPSec to check the authentication And lor decrypt the IP packet and deliver it to the transport layer.

Tunnel Mode

In the tunnel mode, IPSec protects the entire IP packet. It takes an IP packet, including the header, applies IPSec security methods to the entire packet, and then adds a new IP header as shown in Figure.

The new IP header, as we will see shortly, has different information than the original IF header. The tunnel mode is normally used between two routers, between a host and a router, or between a router and a host as shown in Figure.



In other words, we use the tunnel mode when either the sender or the receiver is not a host. The entire original packet is protected from intrusion between the sender and the receiver. It's as if the whole packet goes through an imaginary tunnel. IPSec in tunnel mode protects the original IP header.

Two Security Protocols

IPSec defines two protocols-the Authentication Header (AH) Protocol and the Encapsulating Security Payload (ESP) Protocol-to provide authentication and/or encryption for packets at the IP level.

Authentication Header (AH): The Authentication Header (AH) Protocol is designed to authenticate the source host and to ensure the integrity of the payload carried in the IP packet.

The protocol uses a hash function and a symmetric key to create a message digest; the digest is inserted in the authentication header. The AH is then placed in the appropriate location based on the mode (transport or tunnel). When an IP datagram carries an authentication header, the original value in the protocol field of the IP header is replaced by the value 51.

A field inside the authentication header (the next header field) holds the original value of the protocol field (the type of payload being carried by the IP datagram). The addition of an authentication header follows these steps:

1. An authentication header is added to the payload with the authentication data field set to zero.

2. Padding may be added to make the total length even for a particular hashing algorithm.

3. Hashing is based on the total packet. However, only those fields of the IP header that do not change during transmission are included in the calculation of the message digest (authentic data).

4. The authentication data are inserted in the authentication header.

5. The IP header is added after the value of the protocol field is changed to 51.

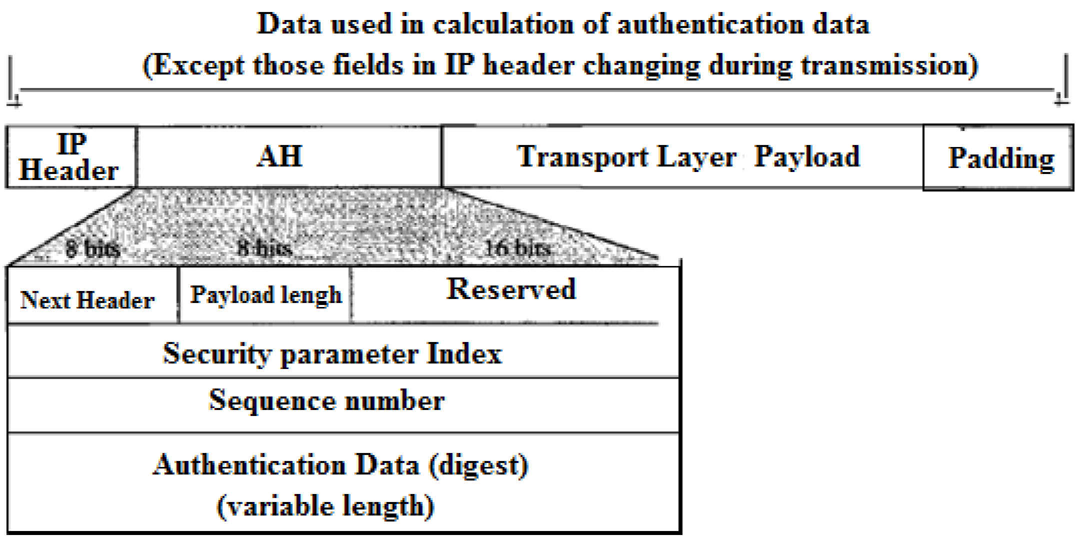
Figure 32.6 shows the fields and the position of the authentication Header in the transport mode.

A brief description of each field follows:

Next header: The 8-bit next-header field defines the type of payload carried by the IP datagram (such as TCP, UDP, ICMP, or OSPF). It has the same function as the protocol field in the IP header before encapsulation. The value of the protocol field in the new IP datagram is now set to 51.

Payload length: The name of this 8-bit field is misleading. It does not define the length of the payload; it defines the length of the authentication header in 4-byte multiples, but it does not include the first 8 bytes.

Authentication Header (AH) Protocol in transport mode



Security parameter index (32-bit): To distinguish one association from the other, each association is identified by SPI. This parameter, in conjunction with the destination addresses or source address and protocol (AR or ESP), uniquely defines an association.

Sequence number: A 32-bit sequence number provides ordering for a sequence of data grams. The sequence numbers prevent a playback and after it reaches 2^{32}; a new connection must be established.

Authentication data: Finally, the authentication data field is the result of applying a hash function to the entire IP datagram except for the fields that are changed during transit (e.g., time-to-live). The AH Protocol provides source authentication and data integrity, but not privacy.

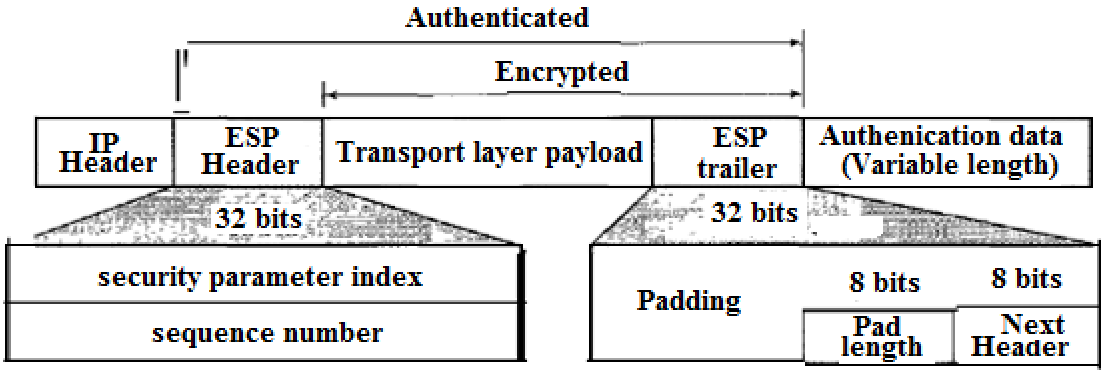
Encapsulating Security Payload (ESP)

The AH Protocol does not provide privacy, only source authentication and data integrity. IPSec later defined an alternative protocol that provides source authentication, integrity, and privacy called Encapsulating Security Payload (ESP). ESP adds a header and trailer. Note that ESP's authentication data are added at the end of the packet which makes its calculation easier. Figure shows the location of the ESP header and trailer.

When an IP datagram carries an ESP header and trailer, the value of the protocol field in the IP header is 50. A field inside the ESP trailer (the next-header field) holds the original value of the protocol field. The ESP procedure follows these steps:

1. An ESP trailer is added to the payload.
2. The payload and the trailer are encrypted.
3. The ESP header is added.
4. The ESP header, payload, and ESP trailer are used to create the authentication data.
5. The authentication data are added to the end of the ESP trailer.
6. The IP header is added after the protocol value is changed to 50

Encapsulation Security Payload (ESP) Protocol in transport mode



The fields for the header and trailer are as follows:

Security parameter index: The 32-bit security parameter index field is similar to AH Protocol.

Sequence number: 32-bit sequence number field is similar to that defined for the AH Protocol.

Padding: This variable-length field (0 to 255 bytes) of Os serves as padding.

Pad length: The 8-bit pad length field defines the number of padding bytes (0 to 255)

Next header: The 8-bitnext-header field serves the same purpose as the protocol field in the IP header before encapsulation.

Authentication data: Finally, the authentication data field is the result of applying an authentication scheme to parts of the datagram. In AH, part of the IP header is included in the of the authentication data; in ESP, it is not. IPSec supports both IPv4 and IPv6.

AH Versus ESP: The ESP Protocol was designed after the AH Protocol was already in use. AH does not provide confidentiality. ESP provides source authentication, data integrity, and privacy. If confidentiality is needed, one should use ESP instead of AH

Services Provided by IPSec

1. **Access Control**: IPSec provides access control indirectly by using a Security Association Database (SADB). When a packet arrives at a destination, and there is no security association already established for this packet, the packet is discarded.
2. **Message Authentication**: The integrity of the message is preserved in both AH and ESP by using authentication data. A digest is created and sent by sender to be checked by the receiver.
3. **Entity Authentication**: The security association and the keyed-hashed digest of the data sent by the sender authenticate the sender of the data in both AH and ESP.
4. **Confidentiality**: The encryption of the message in ESP provides confidentiality. AH does not provide confidentiality. If confidentiality is needed, one should use ESP instead of AH.
5. **Replay Attack Protection**: In both protocols, the replay attack is prevented by using sequence numbers and a sliding receiver window. Each IPSec header contains a unique sequence number when the security association is established. To prevent processing of duplicate packets, IPSec mandates the use of a fixed-size window at the receiver. The size of the window is determined by the receiver with a default value of 64.

Security Association

In IPSec, the establishment of the security parameters is done via a mechanism called security association (SA). Security association is a very important aspect of IPSec. Using security association, IPSec changes a connectionless protocol, IP, to a connection-oriented protocol. The logical connection is there and ready for sending a secure datagram. Of course, they can break the connection, or they can establish a new one after a while (which is a more secure way of communication).

Security Association Database (SADB*):* A security association can be very complex. We need a set of SAs that can be collected into a database. This database is called the security association database (SADB). The database can be thought of as a two-dimensional table with each row defining a single SA. Normally, there are two SADBs, one inbound and one outbound.

Internet Key Exchange (IKE): Now we come to the last part of the puzzle-how SADBs are created. The Internet Key Exchange (IKE) is a protocol designed to create both inbound and outbound security associations in SADBs. IKE creates SAs for IPSec. IKE is a complex protocol based on three other protocols-Oakley, SKEME, and ISAKMP-as shown in

Virtual Private Network

Virtual private network (VPN) is a technology that is gaining popularity among large organizations that use the global Internet for both intra- and inter organization communication, but require privacy in their internal communications. VPN uses the IPSec Protocol to apply security to the IP datagrams.

Although VPNs can be implemented on top of ATM (or frame relay), an increasingly popular approach is to build VPNs directly over the Internet. A common design is to equip each office with a firewall and create tunnels through the Internet between all pairs of offices, as illustrated in Fig. 5-21(b).

Once the SAs have been established, traffic can begin flowing. To a router within the Internet, a packet traveling along a VPN tunnel is just an ordinary packet. The only thing unusual about it is the presence of the IPSec header after the IP header, but since these extra headers have no effect on the forwarding process, the routers do not care about this extra header.

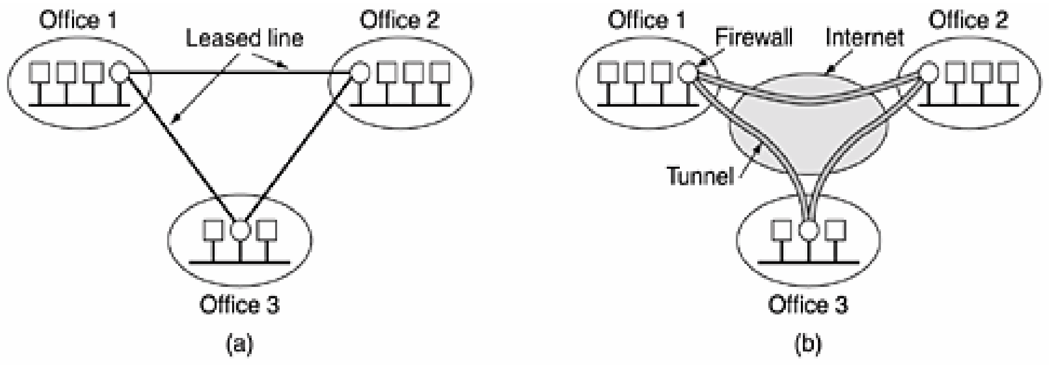
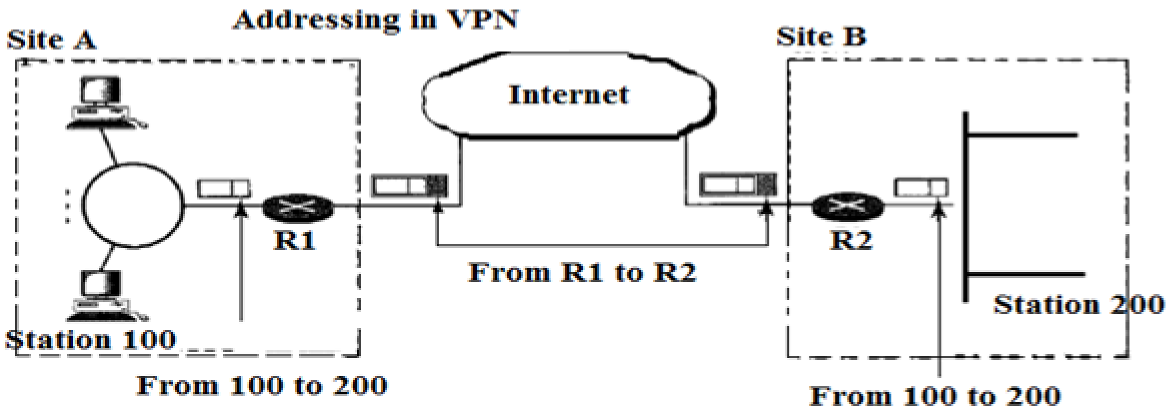


Figure 5- 15. (a) A leased-line private network. (b) A virtual private network.

A key advantage of organizing a VPN this way is that it is completely transparent to all user software. The firewalls set up and manage the SAs. The only person who is even aware of this setup is the system administrator who has to configure and manage the firewalls. To everyone else, it is like having a leased-line private network again.



Addressing in VPN Technology

VPN technology uses IPSec in the tunnel mode to provide authentication, integrity, and privacy. Tunneling To guarantee privacy and other security measures for an organization, VPN can use the IPSec in the tunnel mode. In this mode, each IP datagram destined for private use in the organization is encapsulated in another datagram. To use IPSec in tunneling, the VPNs need to use two sets of addressing, as shown in Figure. The public network (Internet) is responsible for carrying the packet from Rl to R2. Outsiders cannot decipher the contents of the packet or the source and destination addresses.

PGP

One of the protocols to provide security at the application layer is Pretty Good Privacy (PGP). PGP is designed to create authenticated and confidential e-mails. Sending an e-mail is a one-time activity. The nature of this activity is different from those we have seen in the previous two sections

Security Parameters: Phil Zimmerman, the designer and creator of PGP, has found a very elegant solution to security issues questions. The security parameters need to be sent with the message. In PGP, the sender of the message needs to include the identifiers of the algorithms used in the message as well as the values of the keys.

Services provided by PGP

PGP can provide several services based on the requirements of the user.

1. ***Plaintext:*** The simplest case is to send the e-mail message in plaintext (no service).
2. ***Message Authentication:*** Two keys are needed for this scenario.
3. ***Compression:*** PGP used to compress the message and digest to make the packet compact.
4. ***Key Confidentiality with One Time Session: C***onfidentiality in an e-mail system can be achieved by using conventional encryption with a one-time session key.
5. ***Code Conversion:*** To translate other characters not in the ASCII, PGP uses Radix 64 conversion.
6. ***Segmentation:*** PGP allows segmentation of the message after it has been converted to Radix 64 to make each transmitted unit the uniform size allowed by the underlying e-mail protocol.

A Scenario for PGP

Let us describe a scenario that combines some of these services, authentication and confidentiality. The whole idea of PGP is based on the assumption that a group of people who need to exchange e-mail messages trust one another. Everyone in the group somehow knows (with a degree of trust) the public key of any other person in the group. Based on this single assumption, Figure shows a simple scenario in which an authenticated and encrypted message is sent from Alice to Bob.

Sender Site: The following shows the steps used in this scenario at Alice's site:

1. Alice creates a session key (for symmetric encryption/decryption) and concatenates it with the identity of the algorithm which will use this key. The result is encrypted with Bob's public key. Alice adds the identification of the public-key algorithm used above to the encrypted result. This part of the message contains three pieces of information: the session key, the symmetric

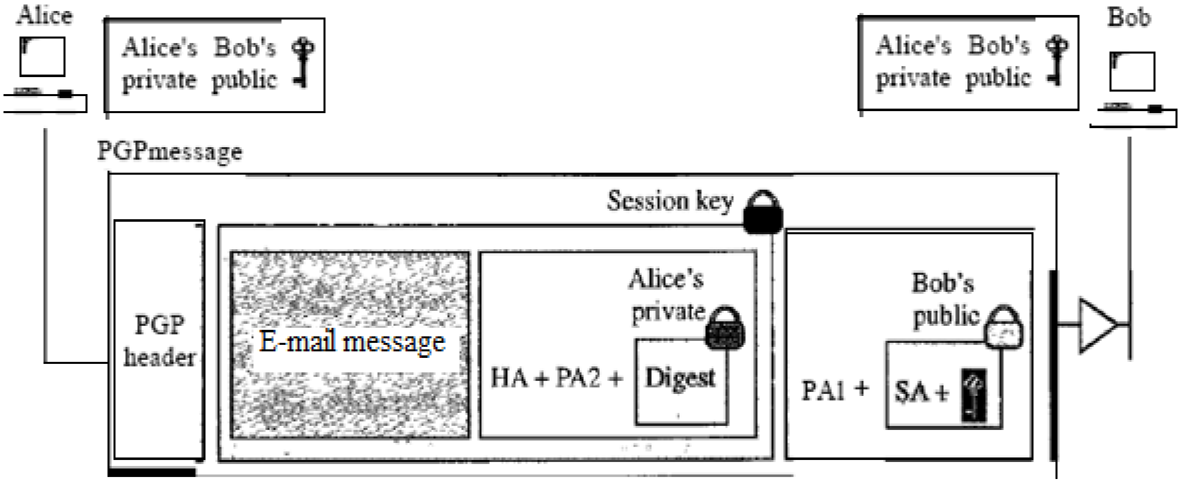
encryption/decryption algorithm to be used later, and the asymmetric encryption/decryption algorithm that was used for this part.

2.a. Alice authenticates the message (e-mail) by using a public-key signature algorithm and encrypts it with her private key. The result is called the signature. Alice appends the identification of the public key (used for encryption) as well as the identification of the hash algorithm (used for authentication) to the signature. This part of the message contains the signature and two extra pieces of information: the encryption algorithm and the hash algorithm.

b. Alice concatenates the three pieces of information created above with the message (e-mail) and encrypts the whole thing, using the session key created in step 1.

3. Alice combines the results of steps 1 and 2 and sends them to Bob (after adding tue appropriate PGP header).

A scenario in which an e-mail message is authenticated and encrypted



PA1: Public-Key algorith 1 (for encrypting session key)
PA2: Public-Key algorith 2 (for encrypting digest)
SA: Symmetric-key algorithm identification (for encrypting message and digest)
HA: Hash algorithm identification (for creating digest)

Receiver Site: The following shows the steps used in this scenario:

1. Bob uses his private key to decrypt the combination of the session key and symmetric-key algorithm identification.

2. Bob uses the session key and the algorithm obtained in step 1 to decrypt the rest of the PGP message. Bob now has the content of the message, the identification of the public algorithm used for creating and encrypting the signature, and the identification of the hash algorithm used to create the hash out of the message.

3. Bob uses Alice's public key and the algorithm defined by PA2 to decrypt the digest.

4. Bob uses hash algorithm defined by HA to create a hash out of message he obtained in step 2

5. Bob compares the hash created in step 4 and the hash he decrypted in step 3. If the two are identical, he accepts the message; otherwise, he discards the message.

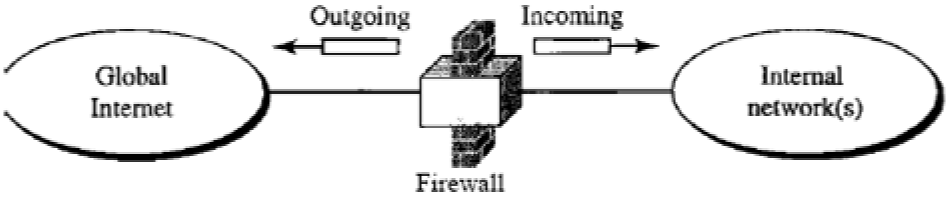
PGP Algorithms

Public key algorithms:	RSA (with ID 1, 2, 3) and DSS (with ID 17)
Hash Algorithm:	MD-5(ID 1) SHA-1 (ID 2) RIPE-MD (ID 3)
Encryption algorithms:	IDEA (ID 1) Triple DES (ID 3) AES (ID 9)

PGP Certificates: To trust the owner of the public key, each user in the PGP group needs to have, implicitly or explicitly, a copy of the certificate of the public-key owner. Although the certificate can come from a certificate authority (CA), this restriction is not required in PGP. PGP has its own certificate system.

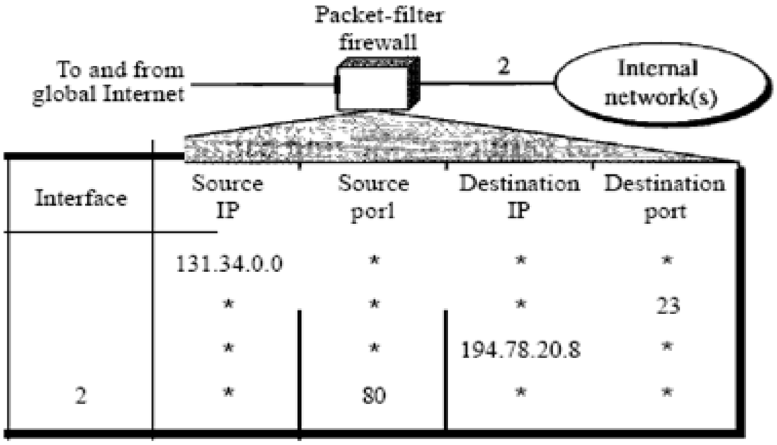
FIREWALLS

To control access to a system, we need firewalls. A **firewall** is a device (usually a router or a computer) installed between the internal network of an organization and the rest of the Internet. It is designed to forward some packets and filter (not forward) others. Figure shows a firewall.



For example, a firewall may filter all incoming packets destined for a specific host or a specific server such as HTTP. A firewall can be used to deny access to a specific host or a specific service in the organization. A firewall is usually classified as a packet-filter firewall or a proxy-based firewall.

Packet-Filter Firewall: A firewall can be used as a packet filter. It can forward or block packets based on the information in the network layer and transport layer headers: source and destination IP addresses, source and destination port addresses, and type of protocol (TCP or UDP). A packet-filter firewall is a router that uses a filtering table to decide which packets must be discarded (not forwarded). Figure below shows an example of a filtering table for this kind of a firewall.



Interface	Source IP	Source porl	Destination IP	Destination port
	131.34.0.0	*	*	*
	*	*	*	23
	*	*	194.78.20.8	*
2	*	80	*	*

According to Figure the following packets are filtered:

1. Incoming packets from network 131.34.0.0 are blocked. Note that the * (asterisk) means "any."

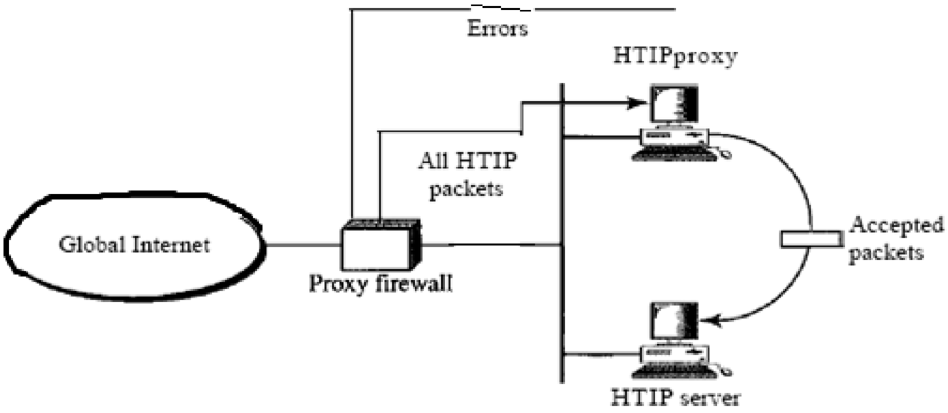
2. Incoming packets destined for any internal TELNET server (port 23) are blocked.

3. Incoming packets destined for internal host 194.78.20.8 are blocked. The organization wants this host for internal use only.

4. Outgoing packets destined for an HTfP server (port 80) are blocked. The organization does not want employees to browse the Internet. A packet filter firewall filters at the network or transport layer.

Proxy Firewall

The packet-filter firewall is based on the information available in the network layer and transport layer headers (IP and TCPIUDP). However, sometimes we need to filter a message based on the information available in the message itself (at the application layer). In this case, a packet-filter firewall is not feasible because it cannot distinguish between different packets arriving at TCP port 80 (HTTP). Testing must be done at the application level (using URLs). One solution is to install a proxy computer, which stands between the customer (user client) computer and the corporation computer shown in Figure.



When the user client process sends a message, the proxy firewall runs a server process to receive the request. The server opens the packet at the application level and finds out if the request is legitimate. If it is, the server acts as a client process and sends the message to the real server in the corporation. If it is not, the message is dropped and an error message is sent to the external user. In this way, the requests of the external users are filtered based on the contents at the application layer.

Diffie-Hellman

RSA is a public-key cryptosystem that is often used to encrypt and decrypt symmetric keys. Diffie-Hellman, on the other hand, was originally designed for key exchange. In the **Diffie-Hellman** cryptosystem, two parties create a symmetric session key to exchange data without having to remember or store the key for future use. They do not have to meet to agree on the key; it can be done through the Internet.

Let us see how the protocol works when Alice and Bob need a symmetric key to communicate. Before establishing a symmetric key, the two parties need to choose two numbers *p* and *g*. The first number, *p*, is a large prime number on the order of 300 decimal digits (1024 bits). The second number is

a random number. These two numbers need not be confidential. They can be sent throughthe Internet; they can be public.

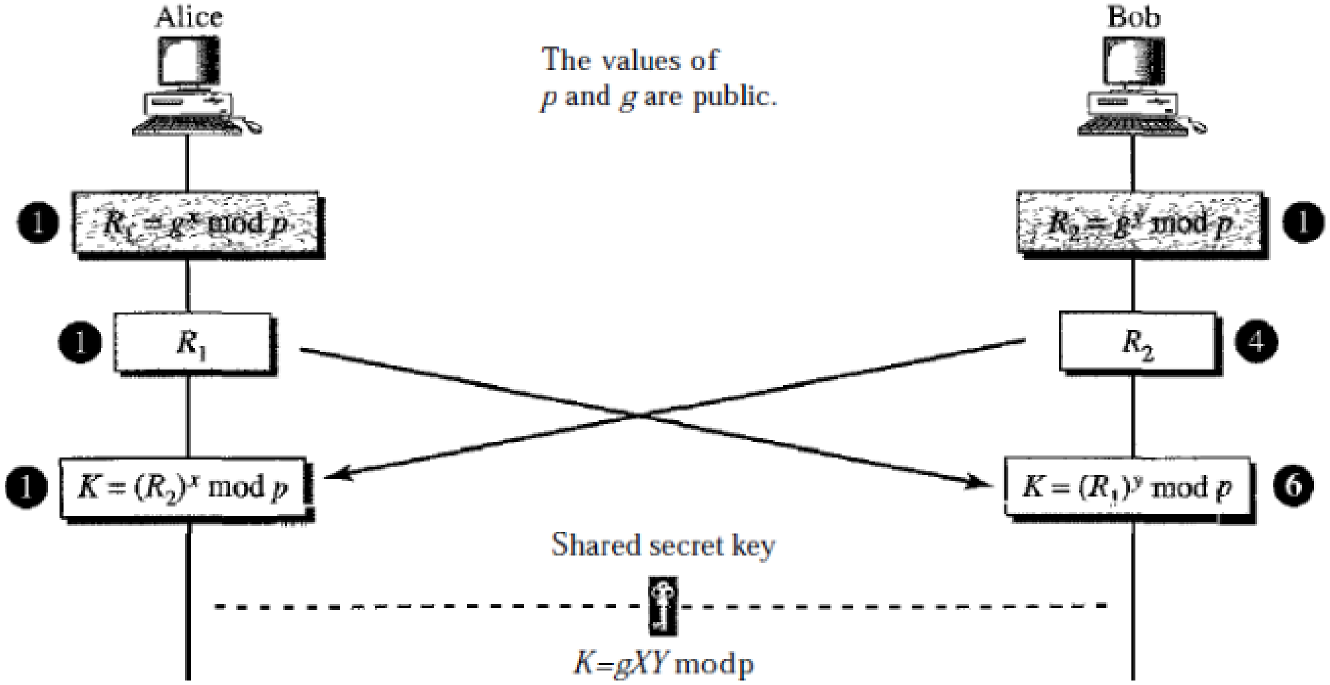


Figure shows the procedure. The steps are as follows:

Step 1: Alice chooses a large random number x and calculates $R1 = g^x \bmod p$.
Step 2: Bob chooses another large random number y and calculates $R2 = gY \bmod p$.
Step 3: Alice sends $R1$ to Bob. Note that Alice does not send the value of x; she sends only $R1$-
Step 4: Bob sends R_2 to Alice. Again, note that Bob does not send the value of y, he sends only R_2.
Step 5: Alice calculates $K = (R_2)^x \bmod p$.
Step 6: Bob also calculates $K = (R_1)^y \bmod p$.

The symmetric key for the session is K.

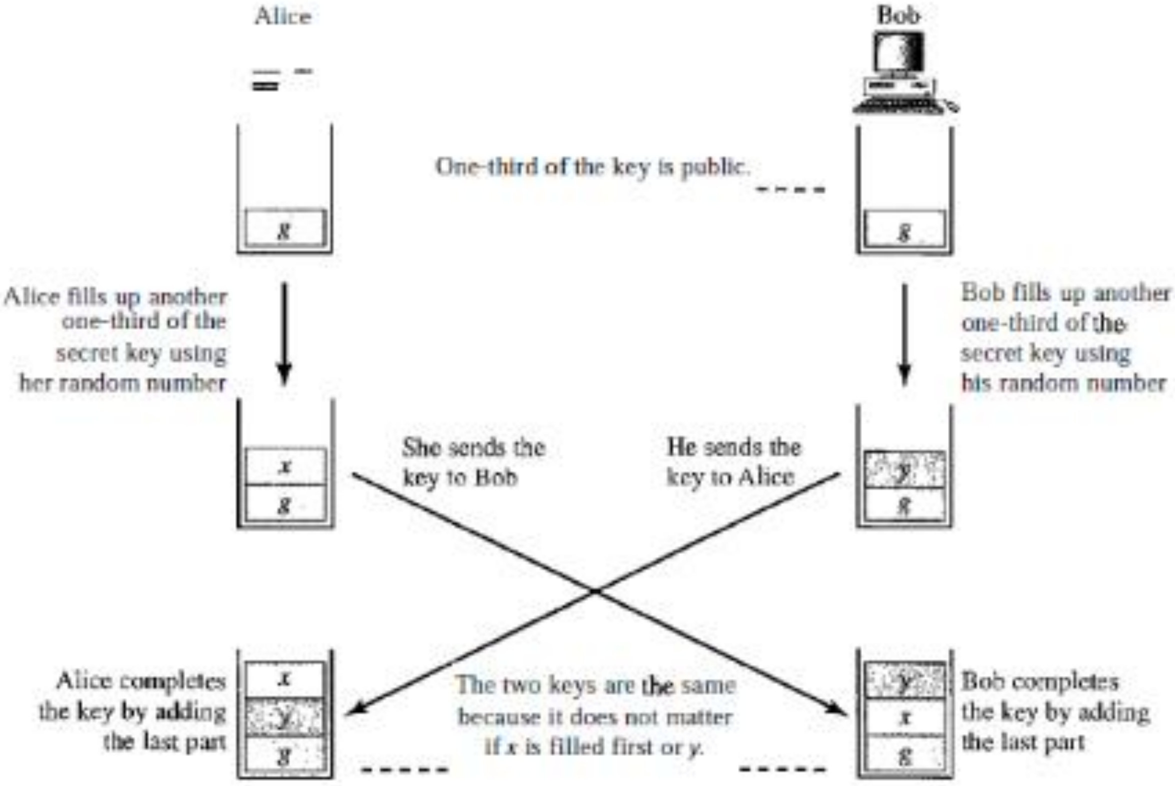
$$(g^x \bmod p)^y \bmod p = (gY \bmod p)X \bmod p = g^{xy} \bmod p$$

Bob has calculated $K = (R_1)^y \bmod p = (g^x \bmod p)^y \bmod p = g^{xy} \bmod p$. Alice has calculated $K = (R_2)X \bmod p = (gY \bmod p)X \bmod = g^{xy} \bmod p$. Both have reached the same value without Bob knowing the value of x and without Alice knowing the value of y.

Idea ofDiffie-Hellman

The Diffie-Hellman concept, shown in Figure 30.27, is simple but elegant. We can think of the secret key between Alice and Bob as made of three parts: g, *x*, and *y*. The first part is public. Everyone knows one-third of the key; *g* is a public value. The other two parts must be added by Alice and Bob. Each adds one part. Alice adds *x* as the second part for Bob; Bob adds *y* as the second part for Alice. When Alice receives the two-thirds completed key from Bob, she adds the last part, her *x*, to complete the key. When Bob receives the two-thirds completed key from Alice, he adds the last part, his *y*, to complete the key. Note that although the key in Alice's hand consists of *g-y-x* and the key in Bob's hand is *g-x-y*, these two keys are the same because g^{xy} =gYx.

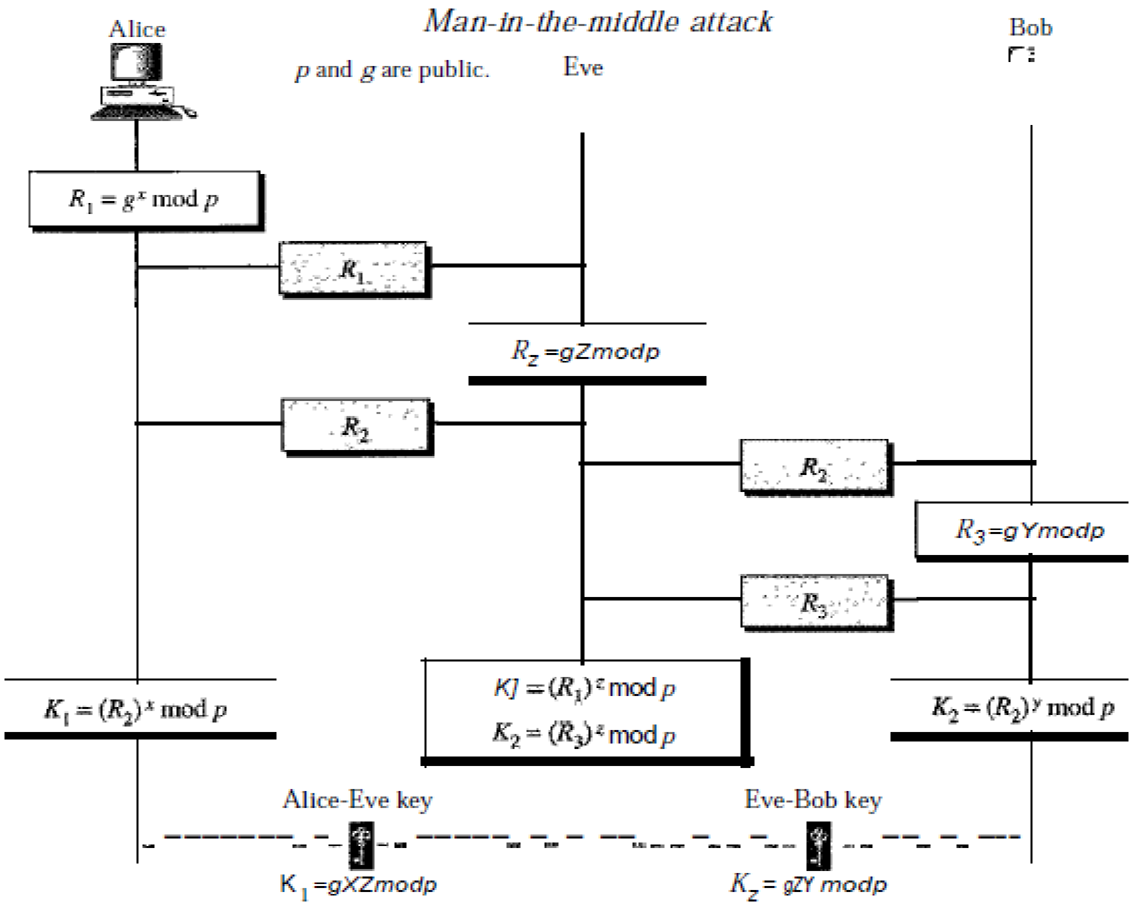
Figure 30.27 *Diffie-Hellman idea*


Alice
Bob
One-third of the key is public.
g
g
Alice fills up another one-third of the secret key using her random number
Bob fills up another one-third of the secret key using his random number
x
g
y
g
She sends the key to Bob
He sends the key to Alice
Alice completes the key by adding the last part
x
y
g
The two keys are the same because it does not matter if x is filled first or y.
y
x
g
Bob completes the key by adding the last part

Note also that although the two keys are the same, Alice cannot find the value *y* used by Bob because the calculation is done in modulo *p;* Alice receives *gY* mod *p* from Bob, not *gY*.

Man-in-the-Middle Attack

Diffie-Hellman is a very sophisticated symmetric-key creation algorithm. If *x* and *y* are very large numbers, it is extremely difficult for Eve to find the key, knowing only *p* and g. An intruder needs to determine *x* and *y* if *R1* and *R2* are intercepted. But finding *x* from *R*1 and *y* from *R2* are two difficult tasks. Eve does not have to find the value of *x* and *y* to attack the protocol. She can fool Alice and Bob by creating two keys: one between herself and Alice and another between herself and Bob. Figure shows the situation.



When Alice sends data to Bob encrypted with *K1* (shared by Alice and Eve), it can be deciphered and read by Eve. Eve can send the message to Bob encrypted by *K2* (shared key between Eve and Bob); or she can even change the message or send a totally new message. Bob is fooled into believing that the message has come from Alice. A similar scenario can happen to Alice in the other direction. This situation is called a man-in-the-middle attack because Eve comes in between and intercepts *RI*, sent by Alice to Bob, and *R3*, sent by Bob to Alice. It is also known as a bucket brigade attack because it resembles a short line of volunteers passing a bucket of water from person to person.

Authentication: The man-in-the-middle attack can be avoided if Bob and Alice first authenticate each other. In other words, the exchange key process can be combined with an authentication scheme to prevent a man-in-the-middle attack.