B. E. 2018 Scheme Seventh Semester Syllabus (EC)

Choice Based Credit System (CBCS) and Outcome Based Education (OBE)

SEMESTER – VII
COMPUTER NETWORKS

Course Code	: 18EC71	CIE Marks	: 40
Lecture Hours/Week	: 3	SEE Marks	: 60
Total Number of Lecture Hours : 40 (08 Hrs/module)		Exam Hours	: 03
CREDITS – 03			

Course Learning Objectives: This course will enable students to:

- Understand the layering architecture of OSI reference model and TCP/IP protocol suite.
- Understand the protocols associated with each layer.
- Learn the different networking architectures and their representations.
- Learn the functions and services associated with each layer.

Module-1

Introduction: Data communication: Components, Data representation, Data flow, Networks: Network criteria, Physical Structures, Network types: LAN, WAN, Switching, The Internet.
(1.1,1.2, 1.3(1.3.1to 1.3.4 of Text)
Network Models: Protocol Layering: Scenarios, Principles, Logical Connections, TCP/IP Protocol Suite: Layered Architecture, Layers in TCP/IP suite, Description of layers, Encapsulation and Decapsulation, Addressing, Multiplexing and Demultiplexing, The OSI Model: OSI Versus TCP/IP.
(2.1, 2.2, 2.3 of Text) **L1, L2**

Module-2

Data-Link Layer: Introduction: Nodes and Links, Services, Two Categories' of link, Sublayers, Link Layer addressing: Types of addresses, ARP. Data Link Control (DLC) services: Framing, Flow and Error Control, Data Link Layer Protocols: Simple Protocol, Stop and Wait protocol, Piggybacking.
(9.1, 9.2(9.2.1, 9.2.2), 11.1, 11.2of Text)
Media Access Control: Random Access: ALOHA, CSMA, CSMA/CD, CSMA/CA.**(12.1 of Text)**
Wired and Wireless LANs: Ethernet Protocol, Standard Ethernet. Introduction to wireless LAN: Architectural Comparison, Characteristics, Access Control.
(13.1, 13.2(13.2.1 to 13.2.5), 15.1 of Text) **L1,L2, L3**

Module-3

Network Layer: Introduction, Network Layer services: Packetizing, Routing and Forwarding, Other services, Packet Switching: Datagram Approach, Virtual Circuit Approach, IPV4 Addresses: Address Space, Classful Addressing, Classless Addressing, DHCP, Network Address Resolution, Forwarding of IP Packets: Based on destination Address and Label.

(18.1, 18.2, 18.4, 18.5.1, 18.5.2 of Text)

Network Layer Protocols: Internet Protocol (IP): Datagram Format, Fragmentation, Options, Security of IPv4 Datagrams. **(19.1of Text)**.

Unicast Routing: Introduction, Routing Algorithms: Distance Vector Routing, Link State Routing, Path vector routing.

(20.1, 20.2of Text) L1,L2, L3

Module-4

Transport Layer: Introduction: Transport Layer Services, Connectionless and Connection oriented Protocols, Transport Layer Protocols: Simple protocol, Stop and wait protocol, Go-Back-N Protocol, Selective repeat protocol. **(23.1, 23.2.1, 23.2.2, 23.2.3, 23.2.4 of Text)**

Transport-Layer Protocols in the Internet:

User Datagram Protocol: User Datagram, UDP Services, UDP Applications, Transmission Control Protocol: TCP Services, TCP Features, Segment, Connection, State Transition diagram, Windows in TCP, Flow control, Error control, TCP congestion control.

(24.2, 24.3.1, 24.3.2, 24.3.3, 24.3.4, 24.3.5, 24.3.6, 24.3.7, 24.3.8, 24.3.9 of Text) L1,L2, L3

Module-5

Application Layer: Introduction: providing services, Application- layer paradigms, Standard Client –Server Protocols: World wide web, Hyper Text Transfer Protocol, FTP: Two connections, Control Connection, Data Connection, Electronic Mail: Architecture, Wed Based Mail, Telnet: Local versus remote logging.Domain Name system: Name space, DNS in internet, Resolution, DNS Messages, Registrars, DDNS, security of DNS.

(25.1, 26.1, 26.2, 26.3, 26.4, 26.6 of Text) L1, L2

Course Outcomes: At the end of the course, the students will be able to:

1. Understand the concepts of networking.
2. Describe the various networking architectures.
3. Identify the protocols and services of different layers.
4. Distinguish the basic network configurations and standards associated with each network.
5. Analyze a simple network and measure its parameters.

Question paper pattern:

- Examination will be conducted for 100 marks with question paper containing 10 full questions, each of 20 marks.
- Each full question can have a maximum of 4 sub questions.
- There will be 2 full questions from each module covering all the topics of the module.
- Students will have to answer 5 full questions, selecting one full question from each module.
- The total marks will be proportionally reduced to 60 marks as SEE marks is 60.

TEXT BOOK:

- Behrouz A Forouzan, "Data Communications and Networking", 5th Edition, McGraw Hill, 2013, ISBN: 1-25-906475-3.

REFERENCE BOOKS:

1. James J Kurose, Keith W Ross, "Computer Networks", Pearson Education.
2. Wayne Tomasi, "Introduction to Data Communication and Networking", Pearson Education.
3. Andrew S Tanenbaum, "Computer Networks", Prentice Hall.
4. William Stallings, "Data and Computer Communications", Prentice Hall.

Communication Networks

MODULE1

MODULE 1

Data communications

When we communicate, we are sharing information. This sharing can be local or remote. Between individuals, local communication usually occurs face to face, while remote communication takes place over distance.

The term **telecommunication**, which includes telephony, telegraphy, and television, means communication at a distance (tele is Greek for “far”). The word data refers to information presented in whatever form is agreed upon by the parties creating and using the data.

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 effectiveness of a data communications system depends on four fundamental characteristics: delivery, accuracy, timeliness, and jitter.

1.Delivery-Thesystemmustdeliverdatato the correct destination. Data must be received by the intended device or user and only by that device or user.

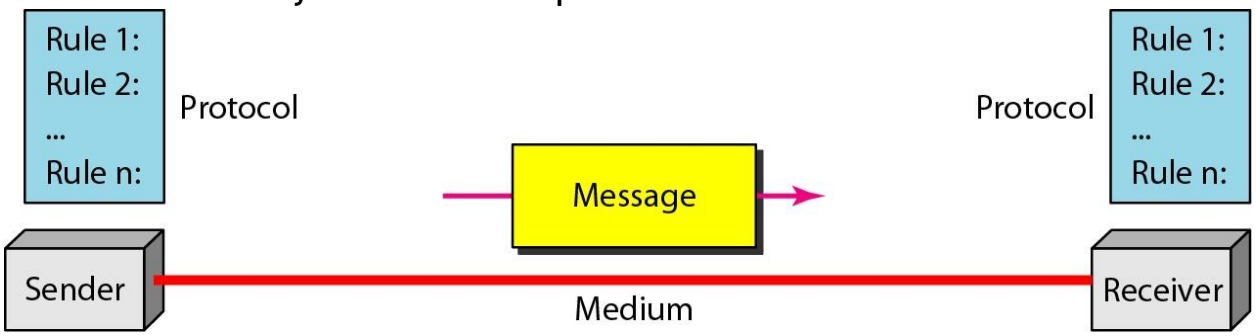
2.Accuracy- The system must deliver the data accurately. Data that have been altered in transmission and left uncorrected are unusable.

3.Timeliness.Thesystemmustdeliver 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. For example, let us assume that video packets are sent every 30ms. If some of the packets arrive with 30-msdelayandotherswith40-msdelay, an uneven quality in the video is the result.

Components

A data communications system has five components



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.Transmissionmedium-. 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, fib er 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, just as a person speaking French cannot be understood by a person who speaks only Japanese.

Data Representation

Information today comes 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. Today, the prevalent coding system is called **Unicode**, which uses 32 bits to represent a symbol or character used in any language in the world. The American Standard Code for Information Interchange (ASCII), developed some decades ago in the United States, now constitutes the first127characters in Unicode and is also referred to as Basic Latin.

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. Appendix B discusses several different numbering systems. Images **Images** - are also represented by bit patterns. In its simplest form, an image is composed of a matrix of pixels (picture elements), where each pixel is as mall dot. The size of the pixel depends on the resolution.

For example, an image can be divided into 1000pixels or10,000 pixels. In the second case, there is a better representation of the image (better resolution), but more memory is needed to store the image. After an image is divided into pixels, each pixel is assigned a bit pattern. The size and the value of the pattern depend on the image. For an image made of only black and- white dots (e.g., a chessboard), a 1-bit pattern is enough to represent a pixel. If an image is not made of pure white and pure black pixels, we can increase the size of the bit pattern to include gray scale. For example, to show four levels of gray scale, we can use 2-bit patterns. A black pixel canberepresentedby00, a dark gray pixel by01, a light graypixelby10, and a white pixel by 11. There are several methods to represent color images. One method is called RGB, so called because each color is made of a combination of three primary colors: red, green, and blue. The intensity of each color is measured, and a bit pattern is assigned toit. Another method is called YCM, in which a color is made of a combination of three other primary colors: yellow, cyan, and magenta.

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 as shown in Figure

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 a).

example-Keyboards and traditional monitors are examples of simplex devices. The keyboard can only introduce input; the monitor can only accept output.

Simplex mode can use the entire capacity of the channel to send data in one direction

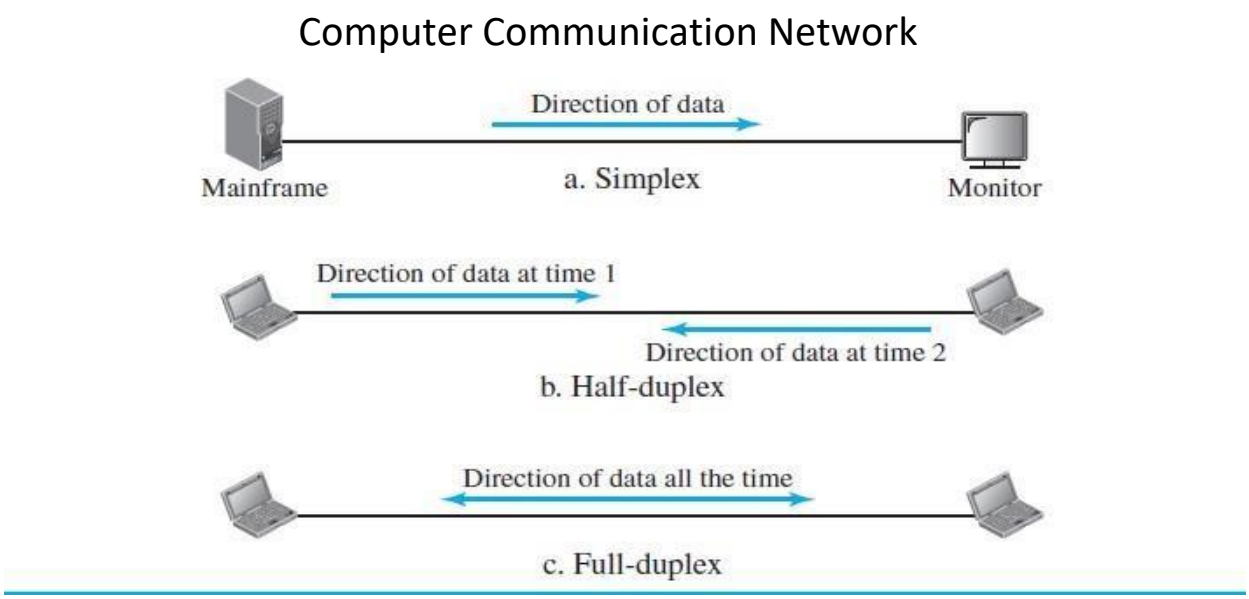


fig: data flow

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 b). The half-duplex mode is like a one-lane road with traffic allowed in both directions. When cars are traveling in one direction, cars going the other way must wait. example-Walkie-talkies and CB (citizens band) radios are both half-duplex systems.

The half-duplex mode is used in cases where there is no need for communication in both directions at the same time; the entire capacity of the channel can be utilized for each direction.

Full-Duplex- In full-duplex mode (also called duplex), both stations can transmit and receive simultaneously (see Figure c). The full-duplex mode is like a two-way street with traffic flowing in both directions at the same time. In full-duplex mode, signals going in one direction share the capacity of the link with signals going in the other direction.

This sharing can occur in two ways: Either the link must contain two physically separate transmission paths, one for sending and the other for receiving; or the capacity of the channel is divided between signals traveling in both directions.

NETWORKS

A network is the interconnection of a set of devices capable of communication. a device can be a host (or an end system as it is sometimes called) such as a large computer, desktop, laptop, workstation, cellular phone, or security system.

A device can also be a connecting device such as a router, which connects the network to other networks, a switch, which connects devices together, a modem (modulator-demodulator), which changes the form of data, and so on.

These devices in a network are connected using wired or wireless transmission media such as cable or air. When we connect two computers at home using a plug-and-play router, we have created a network, although very small.

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.

The performance of a network depends on a number of factors, including the number of users, the type of transmission medium, the capabilities of the connected hardware, and the efficiency of the software

Performance is often evaluated by two networking metrics: throughput and delay.

Reliability- 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- 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.

Physical Structures

Network attributes - Type of Connection and physical topology

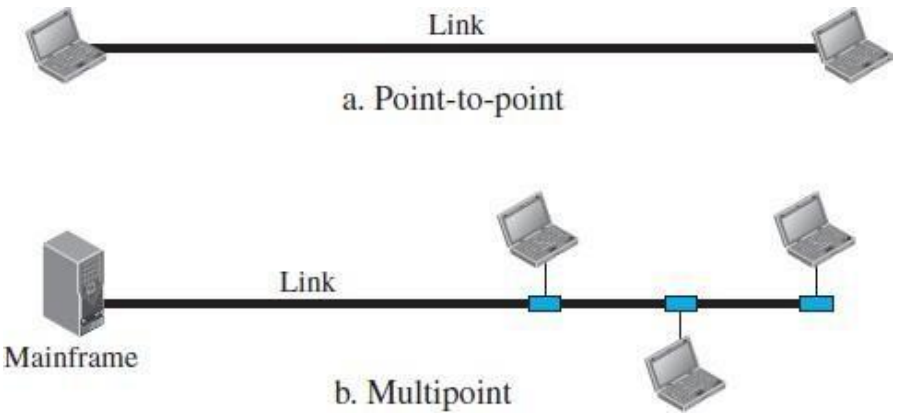
Type of Connection

A network is two or more devices connected through links. A link is a communications pathway that transfers data from one device to another.

There are two possible types of connections:

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

example-When we change television channels by infrared remote control, we are establishing a point-to point connection between the remote control and the television's control system. 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, are also possible (see Figure a).



Multipoint A multipoint (also called multidrop) connection is one in which more than two specific devices share a single link (see Figure b).

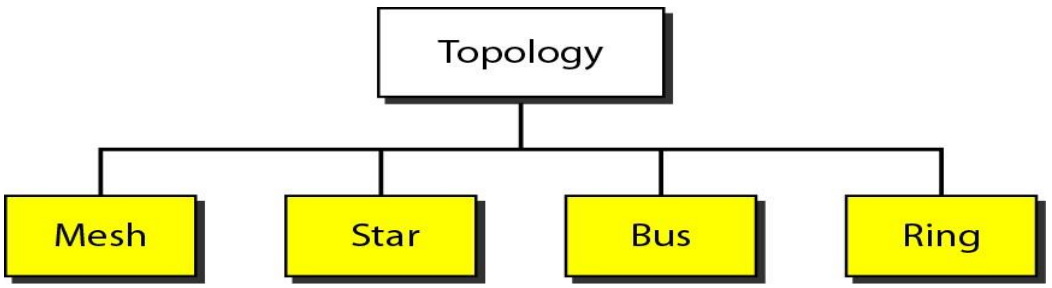
fig: Type of connection

a multipoint environment, the capacity of the channel is shared, either spatially or temporally. If several devices can use the link simultaneously, it is a spatially shared connection. If users must take turns, it is a timeshared connection.

Physical Topology

The term physical topology refers to the way in which a network is laid out physically. Two or more devices connect to a link; two or more links form a topology. The topology of a network is the geometric representation of the relationship of all the links and linking devices (usually called nodes) to one another.

There are four basic topologies possible: mesh, star, bus and ring.



Mesh Topology- In a mesh topology, every device has a dedicated point-to-point link to every other device. The term dedicated means that the link carries traffic only between the two devices it connects.

We need n (n – 1) physical links in a fully connected mesh network with n nodes. if each physical link allows communication in both directions (duplex mode), we need n (n– 1) / 2 duplex-mode links.

To accommodate that many links, every device on the network must have n – 1 input/output (I/O) ports (see Figure) to be connected to the other n– 1 station.

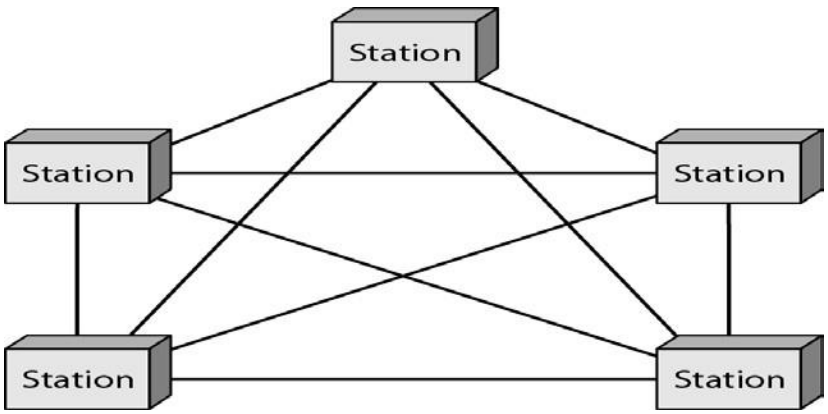
Advantages-

1.Use of dedicated links guarantees that each connection can carry its own data load, thus eliminating the trafficproblemsthatcanoccurwhenlinksmustbesharedbymultipledevices.

2.A mesh topology is robust. If one link becomes unusable, it does not in capacitate the entire system.

3.Privacyorsecurity.Wheneverymessagetravelsalongadedicatedline, only the intended recipient sees it. Physical boundaries prevent other users from gaining access to messages.

4.Point-to-point links make fault identification and fault isolation easy. Traffic can be routed to avoid links with suspected problems. This facility enables the network manager to discover the precise location of the fault and aids in finding its cause and solution.



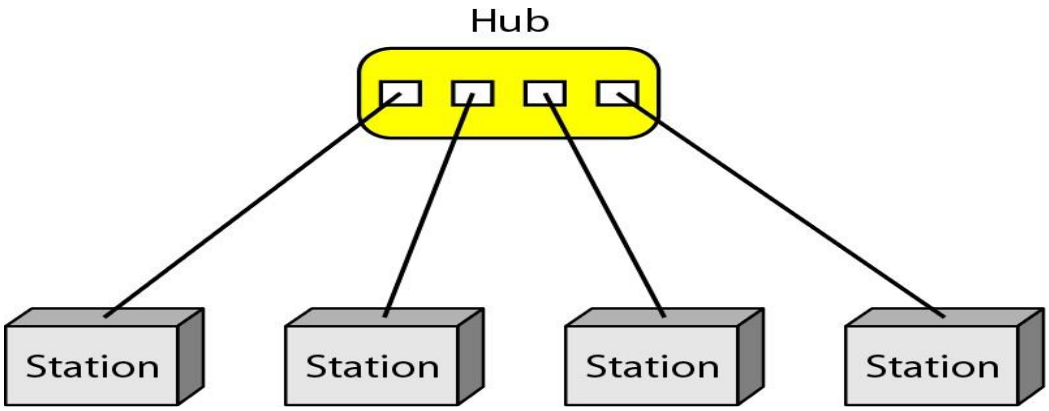
A fully connected mesh topology (fivedevices)

example-of a mesh topology is the connection of telephone regional offices in which each regional office needs to be connected to every other regional office.

Star Topology

In a star topology, each device has a dedicated point-to-point link only to a central controller, usually called a hub.

The devices are not directly linked to one another. A star topology does not allow direct traffic between devices. The controller acts as an exchange: If one device wants to send data to another, it sends the data to the controller, which then relays the data to the other connected device



A star topology connecting four stations

Advantages

1. A star topology is less expensive than a mesh topology.

2.Each device needs only one link and one I/O port to connect. This factor makes it easy to install and reconfigure. Far less cabling needs to be housed, and additions, moves, and deletions involve only one connection: between that device and the hub.

3.robust. If one link fails, only that link is affected. All other links remain active. This factor also lends itself to easy fault identification and fault isolation. As long as the hub is working, it can be used to monitor link problems and bypassed effective links.

Disadvantage

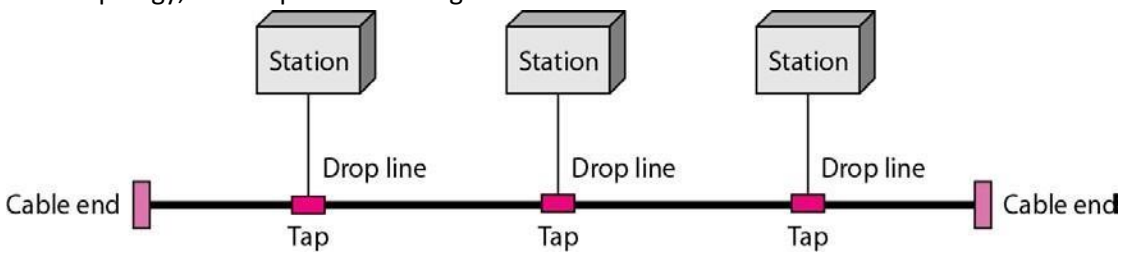
1.The dependency of the whole topology on one single point, the hub. If the hub goes down, the whole system is dead.

2. More cabling is required in a star than in some other topologies (such as ring or bus).

The star topology is used in local-area networks (LANs), High-speed LANs often use a star topology with a central hub.

Bus Topology

A bus topology, is multipoint. One long cable act as a backbone to link all the devices in a network



Nodes are connected to the bus cable by drop lines and taps. A drop line is a connection running between the device and the main cable.

A tap is a connect or that either splice into the main cable or punctures the sheathing fa cable to create a contact with the metallic core.

As a signal travels along the backbone, some of its energy is transformed into heat. Therefore, it becomes weaker and weaker as it travels farther and farther. For this reason, there is a limit on the number of taps a bus can support and, on the distance, between those taps.

Advantages

1. Easy to install.

2.bus uses less cabling than mesh or star topologies. Only the backbone cable stretches through the entire facility. Each drop line has to reach only as far as the nearest point on the backbone.

Disadvantages

1. Difficult reconnection and fault isolation.

2. Difficult to add new devices.

3.Signalreflectionat the taps can cause degradation in quality. This degradation can be controlled by limiting the number and spacing of devices connected to the given length of the cable

4. Adding new devices require modification or replacement of the backbone.

5.a fault or break in the bus cable stops all transmission. The damaged area reflects signals back in the direction of origin, creating noise in both directions.

Bus topology was the one of the first topologies used in the design of early local area networks. Traditional Ethernet LANs can use a bus topology, but they are less popular now.

Ring Topology

In a ring topology, each device has a dedicated point-to-point connection with only the two devices on either side of it. A signal is passed along the ring in one direction, from device to device, until it reaches its destination.

Each device in the ring incorporates a repeater. When a device receives a signal intended for another device, its repeater regenerates the bits and passes them along

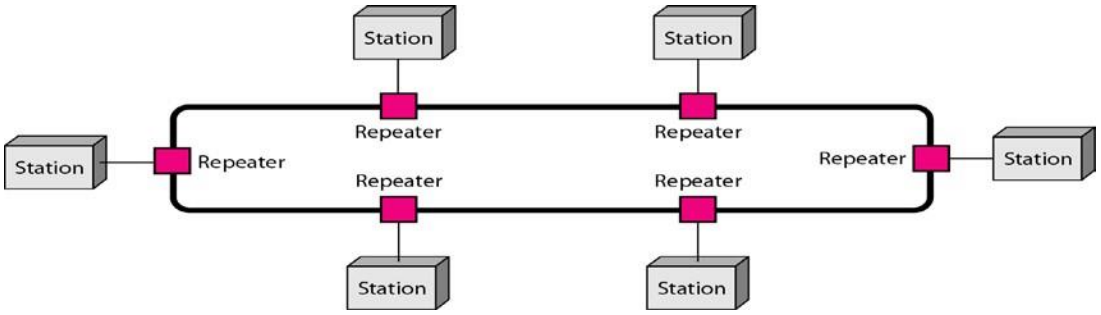


fig: A ring topology connecting six stations

Advantages

1.A ring is relatively easy to install and reconfigure. Each device is linked to only its immediate neighbours (either physically logically). To add or delete a device requires changing only connections.

2. Fault isolation is simplified.

Generally, in a ring assign a lis circulating at all times. If one device does not receive a signal within a specified period, it can issue an alarm. The alarm alerts the network operator to the problem and its location.

Disadvantage

In a simple ring, a break in the ring (such as a disabled station) can disable the entire network. This weaknesscanbesolvedbyusingadualringoraswitchcapableofclosingoffthe break.

Ring topology was prevalent when IBM introduced its local-area network, Token Ring. Today, the need for higher-speed LANs has made this topology less popular.

NETWORK TYPES

Different types of networks

Local Area Network (LAN)-

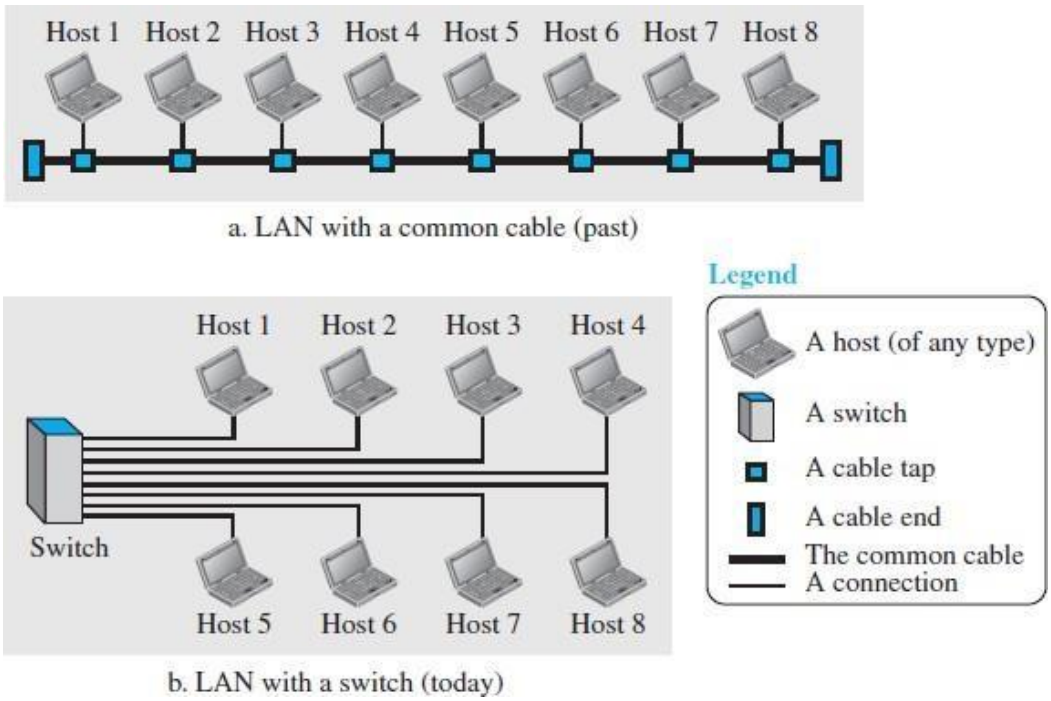


fig: An isolated LAN in the past and today

A local area network (LAN)is usually privately owned and connects some hosts in a single office, building, or campus. Depending on the needs of an organization.

A LAN can be as simple as two PCs and a printer in someone's home office, or it can extend throughout a company and include audio and video devices. Each host in a LAN has an identifier an address, that uniquely defines the host in the LAN.A packet sent by a host to anotherhostcarriesboththesourcehost'sandthedestinationhost'saddresses.

In the past, all hosts in a network were connected through a common cable, which meant that a packet sent from one host to another was received by all hosts. The intended recipient kept the packet; the others dropped the packet.

Today, most LANs use a smart **connecting switch**, which is able to recognize the destination address of the packet and guide the packet to its destination without sending it to all other hosts.

The switch alleviates the traffic in the LAN and allows more than one pair to communicate with each other at the same time if there is no common source and destination among them.

Wide Area Network

A wide area network (WAN) is also an interconnection of devices capable of communication. However, it. We see two distinct examples of WANs today: point-to-point WANs and switched WANs.

Point-to-Point WAN

A point-to-point WAN is a network that connects two communicating devices through a transmission
media (cable or air).

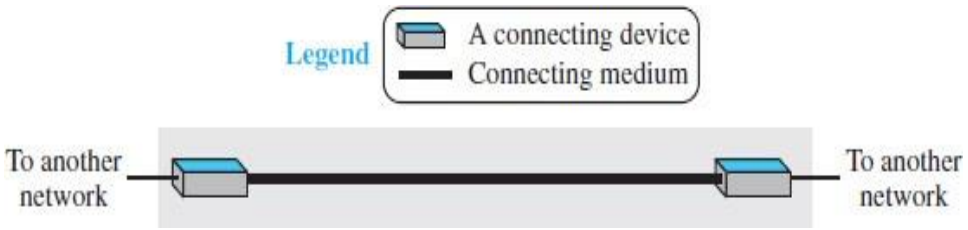


fig: Point-to-Point WAN

Switched WAN

A switched WAN is a network with more than two ends. A switched WAN, is used in the backbone of global communication today. We can say that a switched WAN is a combination of several point-to-point WANs that are connected by switches.

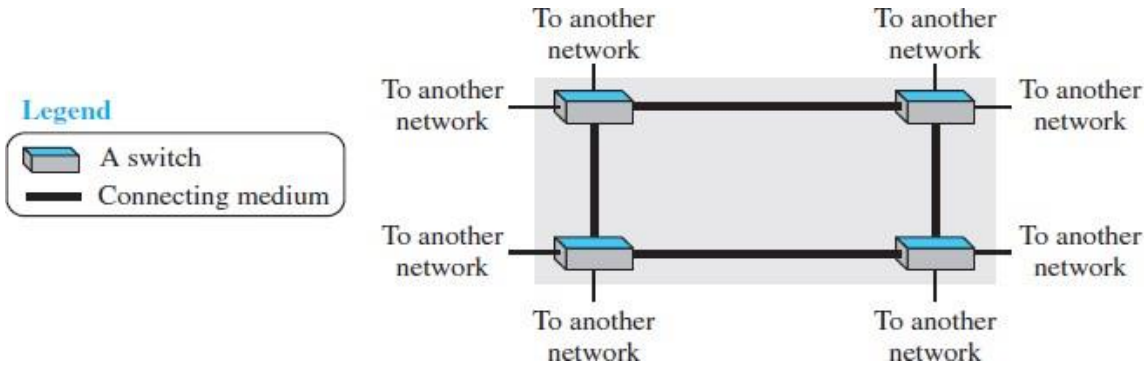


fig: A switched WAN

LAN VS WAN

1. A LAN is normally limited in size, spanning an office, a building, or a campus. 2. A LAN interconnects hosts 3.A LAN is normally privately owned by the organization that uses it	1. A WAN has a wider geographical span, spanning a town, a state, a country, or even the world 2.WAN interconnects connecting devices such as switches, routers, or modems 3.a WAN is normally created and run by communication companies and leased by an organization that uses it.

Internetwork

Today, it is very rare to see a LAN or a WAN in isolation; they are connected to one another. When two or more networks are connected, they make an internetwork, or internet.

example- Assume that an organization has two offices, one on the east coast and the other on the west coast. Each office has a LAN that allows all employees in the office to communicate with each other. To make the communication between employees at different offices possible, the management leases a point to-point dedicated WAN from a service provider, such as a telephone company and connects the two LANs. Now the company has an internetwork, or a private internet (with lowercase i). Communication between offices is now possible.

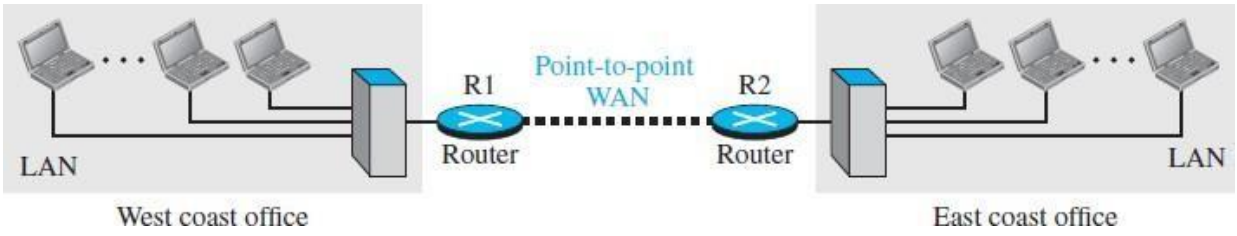


Fig: A network made of two LAN and point-to-point dedicated WAN

When a host in the west coast office sends a message to another host in the same office, the router blocks the message, but the switch directs the message to the destination. On the other hand, when a host on the west coast sends a message to a host on the east coast, router R1 routes the packet to router R2, and the packet reaches the destination. Figure shows another internet with several Ans and WANs connected. One of the WANs is a switched WAN with four switches.

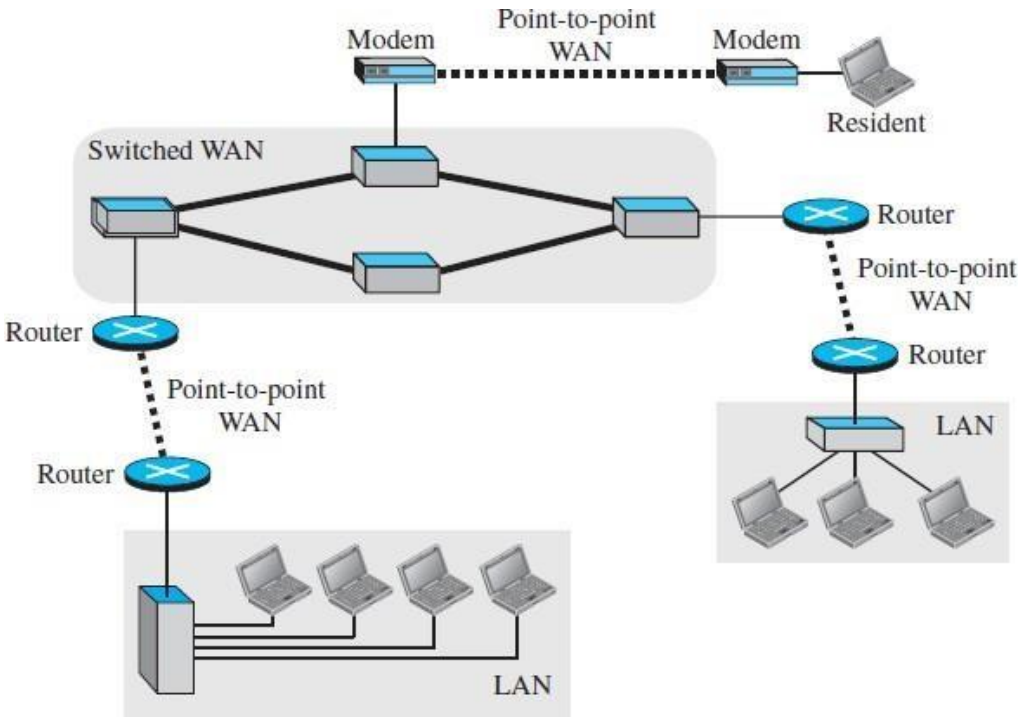


fig: A heterogeneous network made of four WANs and three LANs

Switching

An internet is a switched network in which a switch connects at least two links together. A switch needs to forward data from a network to another network when required. The two most common types of switched networks are circuit-switched and packet-switched networks.

Circuit-Switched Network

In a circuit-switched network, a dedicated connection, called a circuit, is always available between the two end systems; the switch can only make it active or in active (continuous communication between two telephone). FIG shows a very simple switched network that connects four telephones to each end. We have used telephone sets instead of computers as an end system because circuit switching was very common in telephone networks in the past,

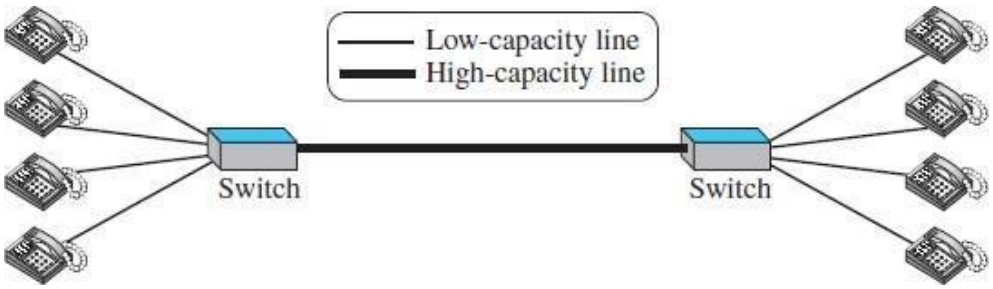


fig: Circuit-Switched Network

The thick line connecting two switches is a high-capacity communication line that can handle four voice communications at the same time; the capacity can be shared between all pairs of telephone sets. The switches used but no storing capability.

Let us look at two cases.

In the first case, all telephone sets are busy; four people at one site are talking with four people at the
Other site; the capacity of the thick line is fully used.

In the second case, only one telephone set at one side is connected to a telephone set at the other side; only one-fourth of the capacity of the thick line is used. This means that a circuit-switched network is efficient only when it is working at its full capacity; most of the time, it is in efficient because it is working at partial capacity.

There as onto make the capacity of the thick line four times the capacity of each voice line is that we do not want communication to fail when all telephone sets at one side want to be connected with all telephone sets at the other side

Packet-Switched Network

In a computer network, the communication between the two computers is done in blocks of data called packets

This allows switches to function for both storing and forwarding because a packet is an independent entity that can be stored and sent later. Fig shows a small packet-switched network that connects four computers at one site to four computers at the other site.

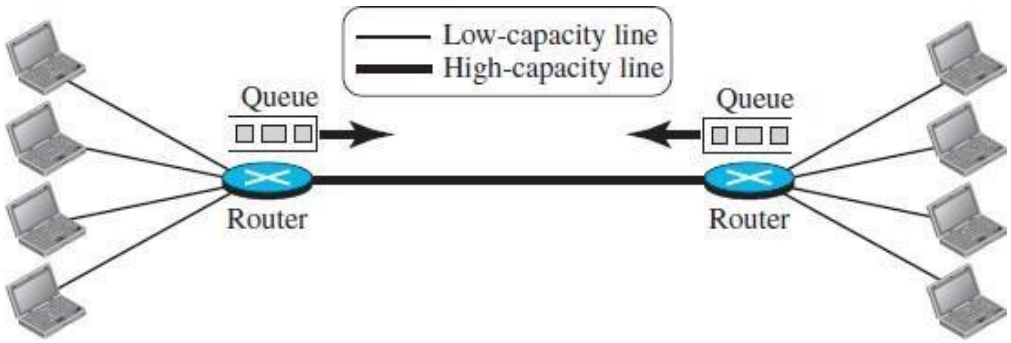


fig: **Packet-Switched Network**

A routerinapacket-switchednetworkhasaqueuethatcanstoreandforwardthe packet.

Example- Now assume that the capacity of the thick line is only twice the capacity of the data line connecting the computers to the routers.

If only two computers (one at each site) need to communicate with each other, there is no waiting for the packets. However, if packets arrive at one router when the thick line is already working at its full capacity, the packets should be stored and forwarded in the order they arrived. The two simple examples show that a packet-switched network is more efficient than a circuit switched network, but the packets may encounter some delays.

The Internet

An internet (note the lowercase i) is two or more networks that can communicate with each other. The most notable internet is called the Internet (uppercase I), and is composed of thousands of interconnected networks.

Figure 1.15 shows a conceptual (not geographical) view of the Internet. The figure shows the Internet as several backbones, provider networks, and customer networks. At the top level, the backbones are large networks owned by some communication companies such as Sprint, Verizon (MCI), AT&T, and NTT. The backbone networks are connected through some complex switching systems, called peering points. At the second level, there are smaller networks, called provider networks, that use the services of the backbones for a fee. The provider networks are connected to backbones and sometimes to other provider networks.

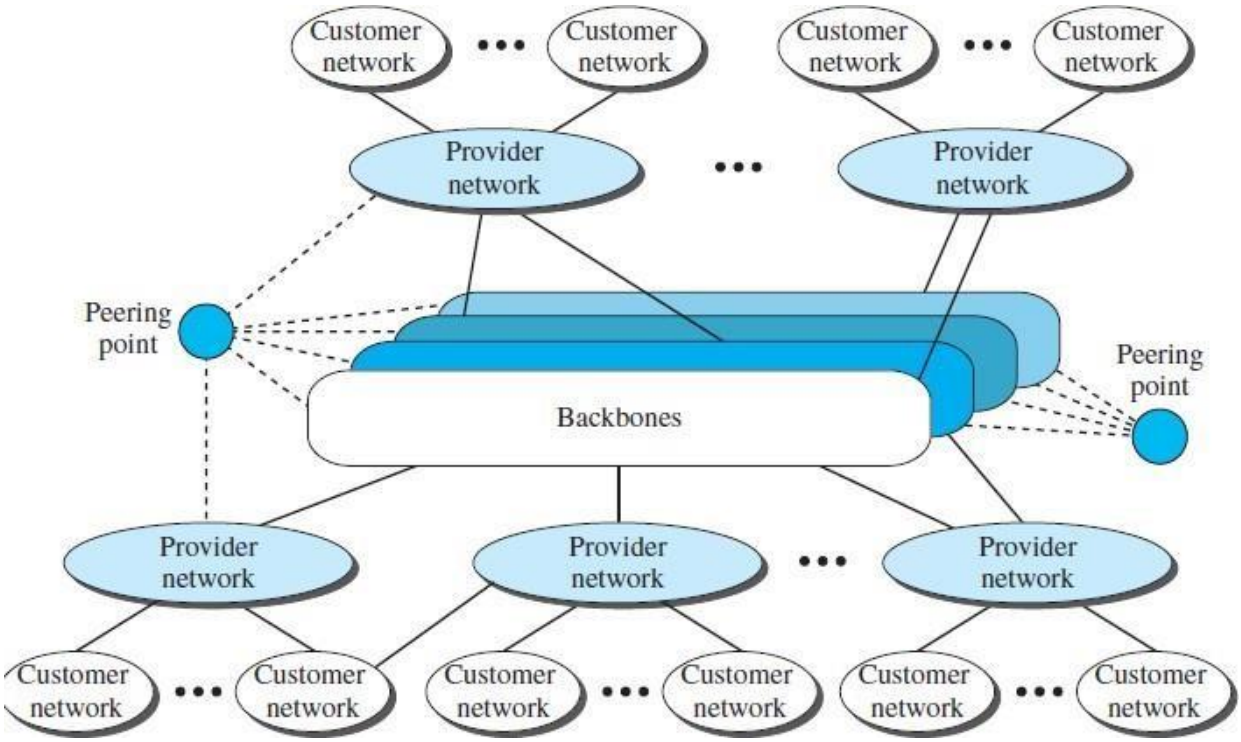


fig: The internet today

The customer networks are networks at the edge of the Internet that actually use the services provided by the Internet. They pay fees to provider networks for receiving services. Backbones and provider networks are also called Internet Service Providers (ISPs). The backbones are often referred to as international ISPs; the provider networks are often referred to as national or regional ISP

Network Models

Protocol Layering

Protocol defines the rules that both the sender and receiver and all intermediate devices need to follow to be able to communicate effectively.

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

Let us develop two simple scenarios to better understand the need for protocol layering.

Scenarios

First Scenario

In the first scenario, communication is so simple that it can occur in only one layer. Assume Maria and Ann are neighbours with a lot of common ideas. Communication between Maria and Ann takes place in one layer, face to face, in the same language, as shown in Figure

fig: single layer protocol

Second Scenario

In the second scenario, we assume that Ann is offered a higher-level position in her company, but needs to move to another branch located in a city very far from Maria. The two friends still want to continue their communication and exchange ideas because they have come up with an innovative project to start a new business when they both retire. They decide to continue their conversation using regular mail through the post office. However, they do not want their ideas to be revealed by other people if the letters are intercepted. They agree on an encryption/decryption technique. The sender of the letter encrypts it to make it unreadable by an intruder; the receiver of the to get the original letter.

Now we can say that the communication between Maria and Ann takes place in three layers, as shown in Figure. We assume that Ann and Maria each have three machines (or robots) that can perform the task at each layer.

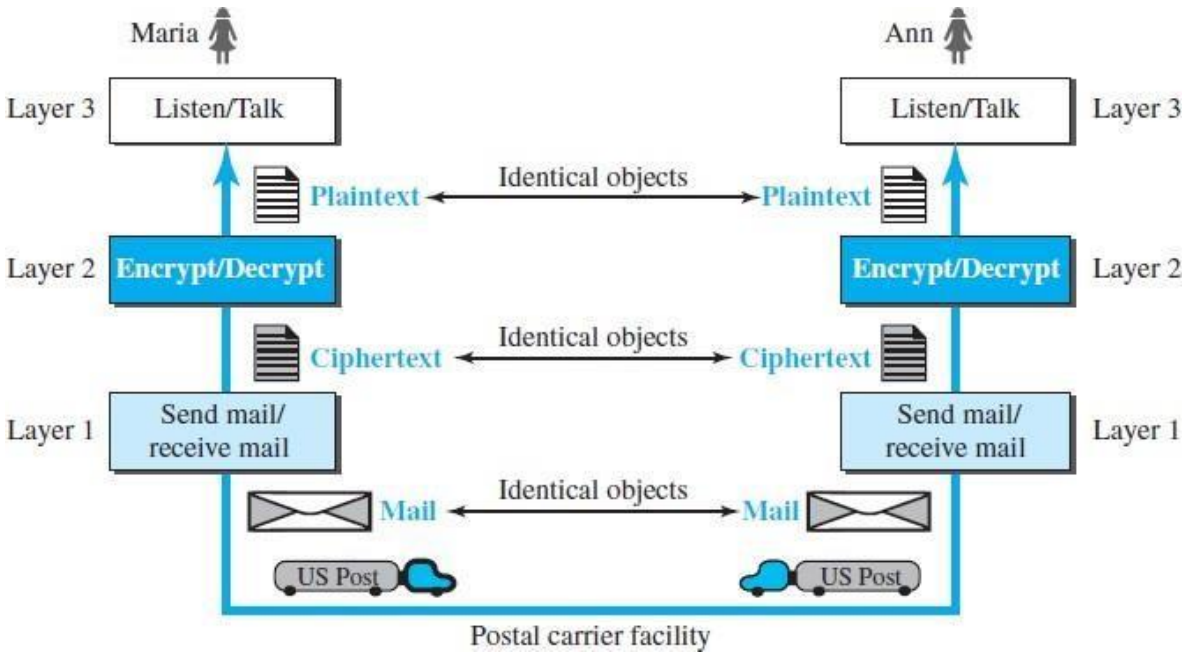


fig: A three-layer protocol

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

The second layer machine takes the plaintext, encrypts it, and creates the ciphertext, which is passed to the first layer machine. The first layer machine, presumablya robot, takes the ciphertext, puts it in an envelope, adds the sender and receiver addresses, and mails it

At Ann's side, the first layer machine picks up the letter from Ann's mailbox, recognizing the letter from Maria by the sender address. The machine takes out the ciphertext from the envelope and delivers it to the second layer machine. The second layer machine decrypts the message, creates the plaintext, and passes the plaintext to the third-layer machine. The third layer machine takes the plaintext and reads it as though Maria is speaking.

Need for protocol layering

1) Protocol layering enables us to divide a complex task into several smaller and simpler tasks.

For example, from fig, we could have used only one machine to do the job of all three machines. However, if the encryption/decryption done by the machine is not enough to protect their secrecy, they would have to change the whole machine. In the present situation, they need to change only the second layer machine; the other two can remain the same. This is referred to as modularity. Modularity in this case means independent layers.

2) A layer (module) can be defined as a black box with inputs and outputs, without concern about how inputs are changed to outputs. If two machines provide the same outputs when given the same inputs, they can replace each other.

For example, Ann and Maria can buy the second layer machine from two different manufacturers. As long as the two machines create the same ciphertext from the same plaintext and vice versa, they do the job.

advantages

1)Protocol layering allows to separate the services from the implementation. Lower layer give the services to the upper layer; we don't care about how the layer is implemented.

For example, Maria may decide not to buy the machine(robot)for the first layer; she can do the job herself. As long as Maria can do the tasks provided by the first layer, in both directions, the communication system works.

2)Protocol layering in the Internet, is that communication does not always use only two end systems; there are intermediate systems that need only some layers, but not all layers. If we did not use protocol layering, we would have to make each intermediate system as complex as the end systems, which makes the whole system more expensive.

Principles of Protocol Layering

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

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

Second Principle The second principle that we need to following protocol layering is that the **two objects** under each layer at both sites should be **identical**.

For example, the object under layer 3 at both sites should be a plaintext letter. The object under layer 2 at both sites should be a ciphertext letter. The object underlayer 1 at both sites should be a piece of mail.

Logical Connections

After following the above two principles, we can think about logical connection between each layer as shown in Figure. This means that we have layer-to-layer communication. Maria and Ann can think that there is a logical (imaginary)connection at each layer through which they can send the object created from that layer. We will see that the concept of logical connection will help us better understand the task of layering we encounter in data communication and networking.

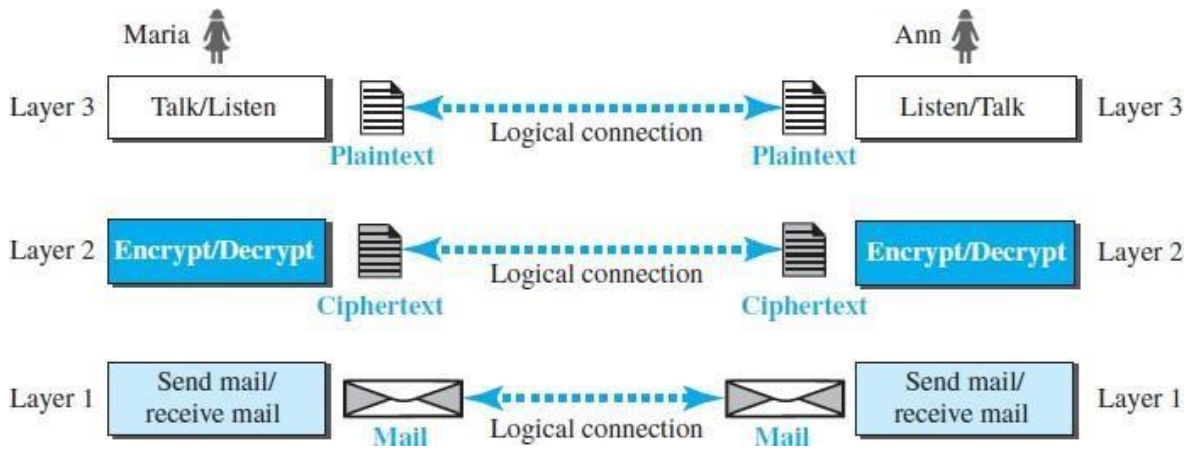


fig: Logical connection between peer layer

TCP/IP PROTOCOLSUITE

TCP/IP is a protocol suite (a set of protocols organized in different layers) used in the Internet today. It is a hierarchical protocol made up of interactive modules, each of which provides a specific functionality. The term hierarchical means that each upper-level protocol is supported by the services provided by one or more lower-level protocols. The original TCP/IP protocol suite was defined as four software layers built upon the hardware. Today, however, TCP/IP is thought of as a five-layer model. Figure shows both configurations.

Layered Architecture

To show how the layer sin the TCP/IP protocol suite are involved in communication between two hosts, we assume that we want to use the suite in a small internet made up of three LANs

(links), each with a link-layer switch. We also assume that the links are connected by one router, as shown in Figure

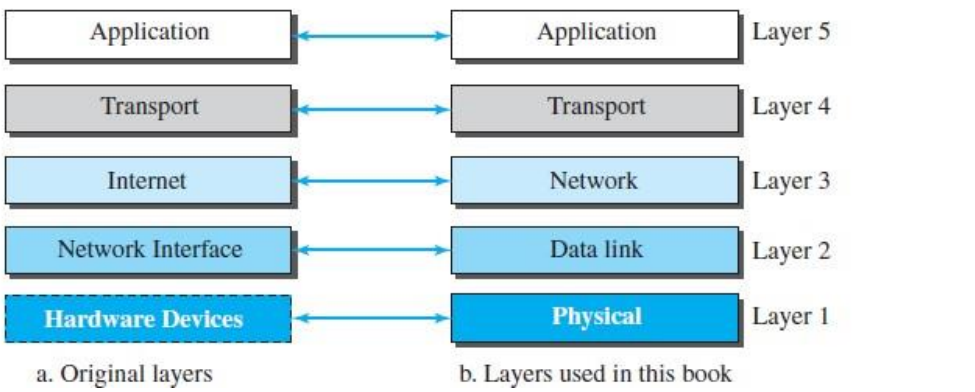


fig: layers in TCP/IP protocol suite

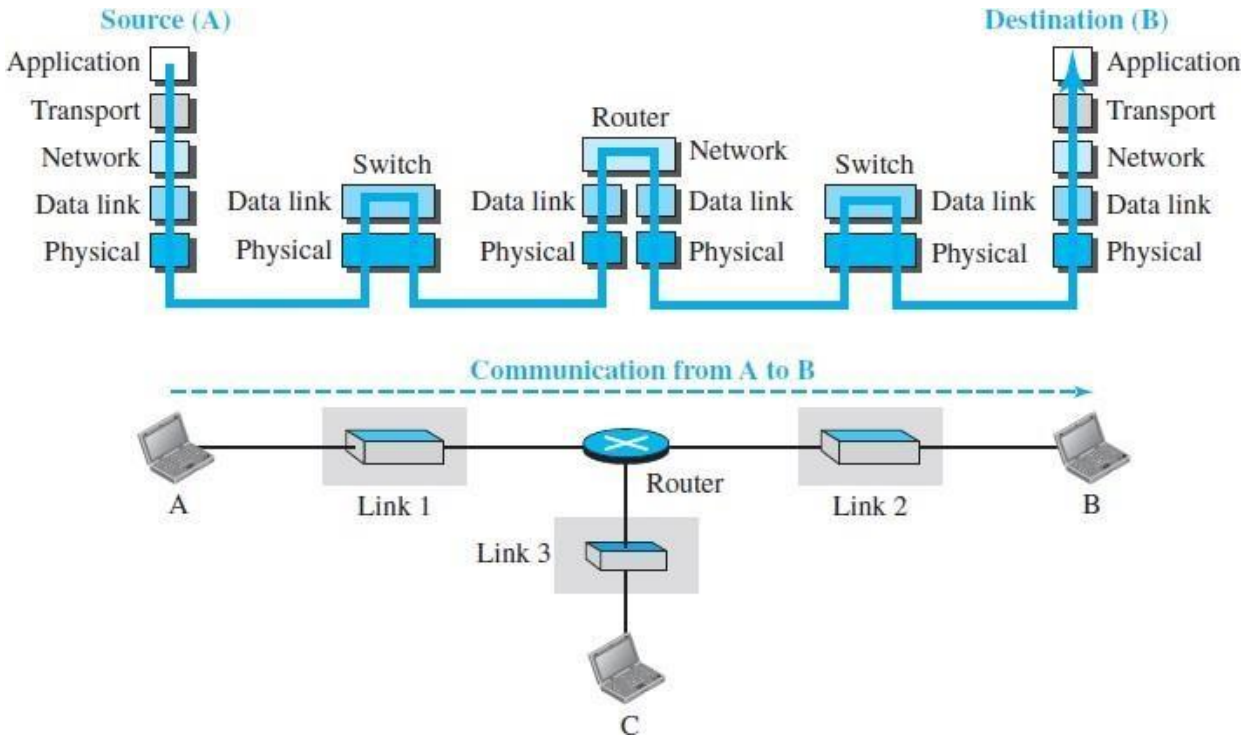


fig: Communication through an internet

Assume that computer A communicates with computer B. As the figure shows, five communicating devices in this communication: source host (computer A), the link-layer switch in link 1, the router, the link-layer switchinlink2, and the destination host (computer B).

The source host needs to create a message in the application layer and send it down the layers so that it is physically sent to the destination host. The destination host needs to receive the communication at the physical layer and then deliver it through the other layers the application layer

The router is involved in only three layers; there is no transport or application layer in a router. Although a **router is always involved in one network layer**, it is involved in n combinations of link and physical layers in which n is the number of links the router is connected to. The reason is that each link may use its own data-link or physical protocol.

For example, in the above figure, the router is involved in three links, but the message sent from source A to destination B is involved in **two links**. Each link may be using **different link- layer and physical-layer protocols; the** router needs to receive a packet from link 1based on one pair of protocols and deliver it to link2basedonanotherpairof protocols.

A link-layer switch in a link, however, is involved only in two layers, data-link and physical. Although each switch in the above figure has two different connections, the connections are in the **same link,** which uses **only one set of protocols**. This means that, unlike a router, a link-layer switch is involved only in one data-link and one physical layer.

Layers in the TCP/IP Protocol Suite

To better understand the duties of each layer, we need to think about the logical connections between layers.

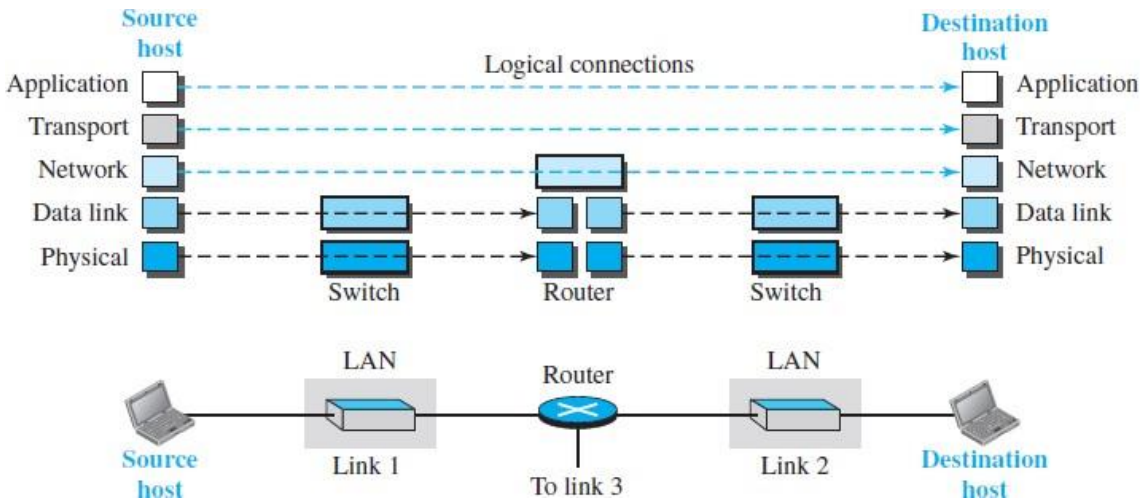


fig: Figure shows logical connections in our simple internet.

Using logical connections makes it easier to think about the duty of each layer. As the figure shows, the duty of the application, transport, and network layers is **end-to-end**. However, the duty of the data-link and physical layers is **hop-to-hop**, in which a hop is a host or router.

In other words, the domain of duty of the top three layers is the **internet**, and the domain of duty of the two lower layers is the **link**.

Another way of thinking of the logical connections is to think about the data unit created from each layer. In the top three layers, the data unit (packets) should not be changed by any router or link-layer switch. In the bottom two layers, the packet created by the host is changed only by the routers, not by the link-layer switches.

Fig shows the second principle discussed previously for protocol layering. We show the identical objects be low each layer related to each device.

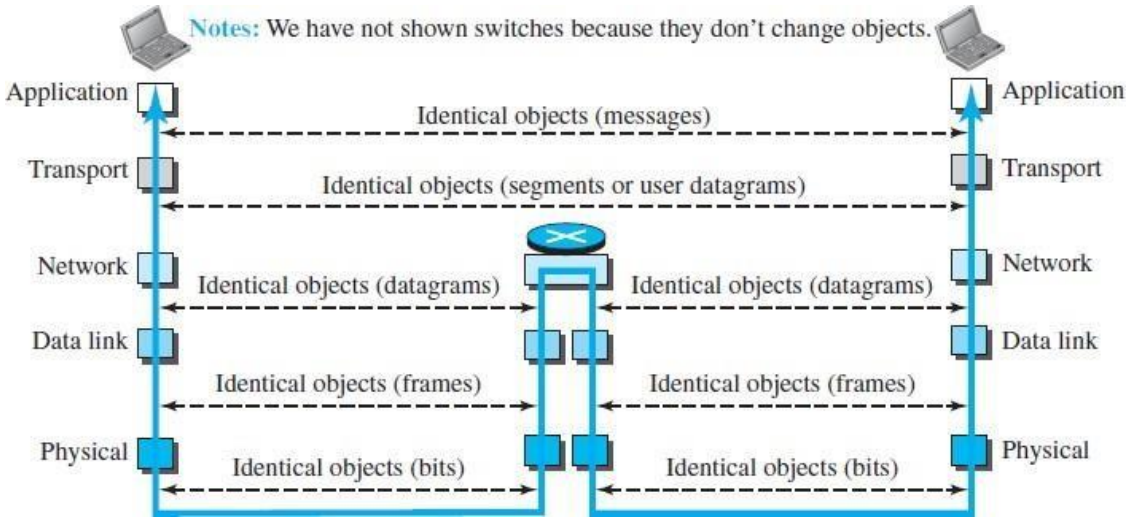


fig: identical objects in the TCP/IP protocol suite

Note that, although the logical connection at the network layer is between the two hosts, we can only say that identical objects exist between two hops in this case because a router may fragment the packet at the network layer and send more packets than received. Note that the link between two hops does not change the object.

Description of Each Layer

Physical Layer

Physical layer is responsible for **carrying individual bits** in a frame across the link. Although the physical layer is the lowest level in the TCP/IP protocol suite, the communication between two

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

Two devices are connected by a transmission medium (cable or air). Transmission medium does not carry bits, **it carries electrical or optical signals**. So, the bits received in a frame from the data-link layer are transformed and sent through the transmission media, but we can think that the logical unit between two physical layers in two devices is a bit. There are several protocols that transform a bit to a signal.

The physical layer of TCP/IP describes hardware standards such as IEEE 802.3, the specification for Ethernet network media, andRS-232, the specification for standard pin connectors.

The following are the main responsibilities of the physical layer

Definition of Hardware Specifications, Encoding and Signalling, Data Transmission and
Reception, Topology and Physical Network Design

Data-link Layer

Internet is made up of several links (LANs and WANs) connected by routers. The data-link layer is responsible for taking the datagram and moving it across the link. (Node to node communication)

The link can be a wired LAN with a link-layer switch, a wireless LAN, a wired WAN, or a wireless WAN. We can also have different protocols used with any link type.

In each case, the data-link layer is responsible for moving the packet through the link. TCP/IP does not define any specific protocol for the data-link layer. It supports all the standard and proprietary protocols. The data-link layer takes a datagram and encapsulates it in a packet called a **frame**.

Each link-layer protocol provides a different service like framing, Flow control, Error control and congestion control.

Network Layer

The network layer is responsible for creating a connection between the source computer and the destination computer. The communication at the network layer is **host-to-host**. However, since there can be several routers from the source to the destination, the routers in the path are responsible for choosing the **best route** for each packet.

The network layer is responsible packetizing and routing and forwarding the packet through possible routes. others services are error and flow control, congestion control.

The network layer in the Internet includes the main protocol, Internet Protocol (IP), that defines the format of the packet, **called a datagram** at the network layer. IP also defines the format and the structure of addresses used in this layer.

IP is also responsible for routing a packet from its source to its destination, which is achieved by each router forwarding the datagram to the next router in its path.

IP is a connectionless protocol that provides no flow control, no error control, and no congestion control services. This means that if any of these services is required for an application, the application should rely only on the transport-layer protocol.

The network layer also includes unicast (one-to-one) and multicast (one-to-many) routing protocols. A routing protocol does not take part in routing (it is the responsibility of IP), but it creates forwarding tables for routers to help them in the routing process. The network layer also has some auxiliary protocols that help IP in its delivery and routing tasks.

The Internet Control Message Protocol (ICMP) helps IP to report some problems when routing a packet. The Internet Group Management Protocol (IGMP) is another protocol that helps IP in multitasking. The Dynamic Host Configuration Protocol (DHCP) helps IP to get the network-layer address for a host. The Address Resolution Protocol (ARP) is a protocol that helps IP to find the link-layer address of a host or a router when its network-layer address is given.

Transport Layer

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

The transport layer is responsible for giving services to the application layer: to get a message from an application program running on the source host and deliver it to the corresponding application program on the destination host. (**Process to process communication**) There are more than one protocol in the transport layer, which means that each application program can use the protocol that best matches its requirement. There are a few transport- layer protocols in the Internet, each designed for some specific tasks.

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

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

Its simplicity, which means small overhead, is attractive to an application program that needs to send short messages and cannot afford the retransmission of the packets involved in TCP, when a packet is corrupted or lost.

A new protocol, Stream Control Transmission Protocol (SCTP) is designed to respond to new applications that are emerging in the multimedia.

Application Layer

As Figure shows, the logical connection between the two application layers is end to-end. The two application layers exchange messages between each other as though there were a bridge between the two layers. However, communication is done through all the layers.

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

The application layer in the Internet includes many predefined protocols.

1)The Hypertext Transfer Protocol (HTTP) is a vehicle for accessing the World Wide Web (WWW).

2)The Simple Mail Transfer Protocol (SMTP) is the main protocol used in electronic mail (e- mail) service.

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

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

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

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

7) The Internet Group Management Protocol (IGMP) is used to collect membership in a group.

Encapsulation and Decapsulation

One of the important concepts in protocol layering in the Internet is encapsulation/ decapsulation. Figure shows this concept for the small internet

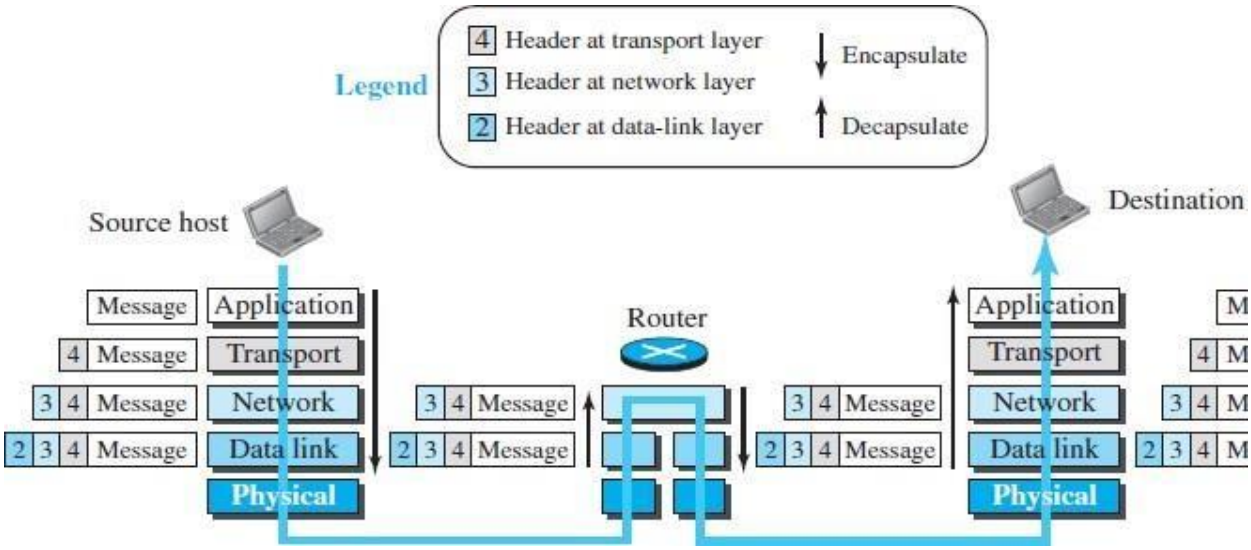


fig: **encapsulation/decapsulation**

We have not shown the layers for the link-layer switches because no encapsulation/ decapsulation occurs in this device. Figure show the encapsulation in the source host, decapsulation in the destination host, and encapsulation and decapsulation in the router.

Encapsulation at the Source Host

At the source, we have only encapsulation.

1.At the application layer, the data to be exchanged is referred to as a message. A message normally does not contain any header or trailer, but if it does, we refer to the whole as the message. The message is passed to the transport layer.

2.The transport layer takes the message as the payload, the load that the transport layer should take care of. It adds the transport layer header to the payload, which contains the identifiers of the source and destination application programs that want to communicate plus some more information that is needed for the end-to end delivery of the message, such as information

needed for flow, error control, or congestion control. The result is the transport- layer packet, which is called the segment (in TCP) and the user datagram (in UDP). The transport layer then passes the packet to the network layer.

3.The network layer takes the transport-layer packet as data or payload and adds its own header to the payload. The header contains the addresses of the source and destination hosts and some more in formation used for error checking of the header, fragmentation information, and soon. The result is the network-layer packet, called a datagram. The network layer then passes the packet to the data-link layer.

4.The data-link layer takes the network-layer packet as data or payload and adds its own header, which contains the link-layer addresses of the host or the next hop (the router). The result is the link-layer packet, which is called a frame. The frame is passed to the physical layer for transmission.

Decapsulation and Encapsulation at the Router

At the router, we have both decapsulation and encapsulation because the router is connected to two or more links.

1.After the set of bits are delivered to the data-link layer, this layer decapsulates the datagram from the frame and passes it to the network layer.

2.Thenetworklayeronlyinspectsthesourceanddestinationaddressesinthe datagram header and consults its forwarding table to find the next hop to which the datagram is to be delivered. The contents of the datagram should not be changed by the network layer in the router unless there is a need to fragment the datagram if it is too big to be passed through the next link. The datagram is then passed to the data-link layer of the next link.

3.The data-link layer of the next link encapsulates the datagram in a frame and passes it to the physical layer for transmission.

Decapsulation at the Destination Host

At the destination host, each layer only decapsulates the packet received, removes the payload, and delivers the payload to the next-higher layer protocol until the message reaches the application layer. It is necessary to say that decapsulation in the host involves error checking

Addressing

We have logical communication between pairs of layers in this model. Any communication that involves two parties needs two addresses: source address and destination address. Although it looks as if we need five pairs of addresses, one pair per layer, we normally have only four because

the physical layer does not need addresses; the unit of data exchange at the physical layer is a bit, which definitely cannot have an address.

Figure2.9 shows the addressing at each layer. At the application layer, we normally use names to define the site that provides services, such as someorg.com, or the e-mail address, such as somebody@coldmail.com.

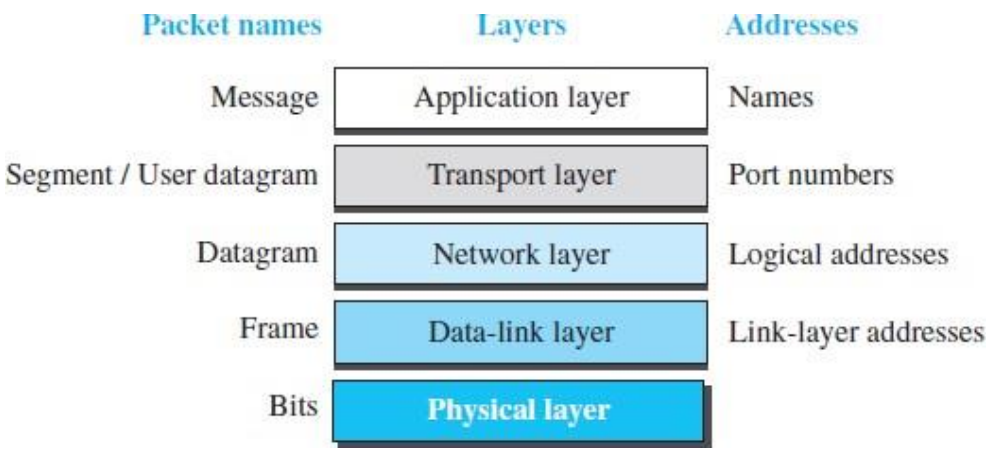


fig: Addressing in the TCP/IP Protocol suite

At the transport layer, addresses are called port numbers, and these define the application- layer programs at the source and destination. Port numbers are local addresses that distinguish between several programs running at the same time.

At the network-layer, the addresses are global, with the whole Internet as the scope. A network-layer address uniquely defines the connection of a device to the Internet.

The link-layer addresses, sometimes called MAC addresses, are locally defined addresses, each of which defines a specific host or router in a network (LAN or WAN).

Multiplexing and Demultiplexing

TCP/IP protocol suite uses several protocols at some layers, we have multiplexing at the source and Demultiplexing at the destination.

Multiplexing means that a protocol at a layer can encapsulate a packet from several next- higher layer protocols (one at a time); demultiplexing means that a protocol can decapsulate and deliver

a packet to several next-higher layer protocols (one at a time). Figure shows the concept of multiplexing and demultiplexing at the three upper layers.

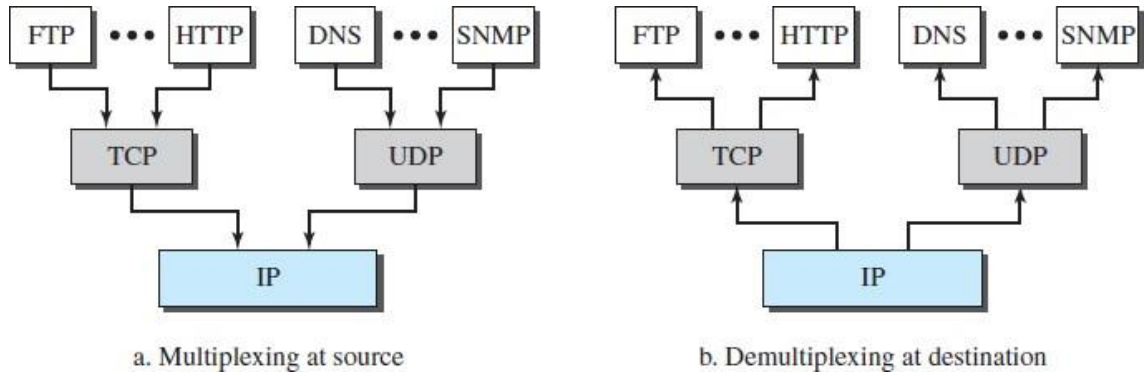


fig: multiplexing and demultiplexing

To be able to multiplex and demultiplex, a protocol needs to have a field in its header to identify to which protocol the encapsulated packets belong.

At the transport layer, either UDP or TCP can accept a message from several application-layer protocols.

At the network layer, IP can accept a segment from TCP or a user datagram from UDP. IP can also accept a packet from other protocols such as ICMP, IGMP, and so on.

At the data-link layer, a frame may carry the payload coming from IP or other protocols such as ARP.

THE OSIMODEL

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. An open system is a set of

protocols that allow any two different systems to communicate regardless of their underlying architecture.

The purpose of the OSI model is to show how to facilitate communication between different systems without requiring changes to the logic of the underlying hardware and software. The **OSI model is not a protocol**; it is a model for understanding and designing a network architecture that is flexible, robust, and interoperable. The OSI model was intended to be the basis for the creation of the protocols in the OSI stack. The OSI model is a layered framework for the design of network systems that allows communication between all types of computer systems.

It consists of seven separate but related layers, each of which defines a part of the process of moving information across a network

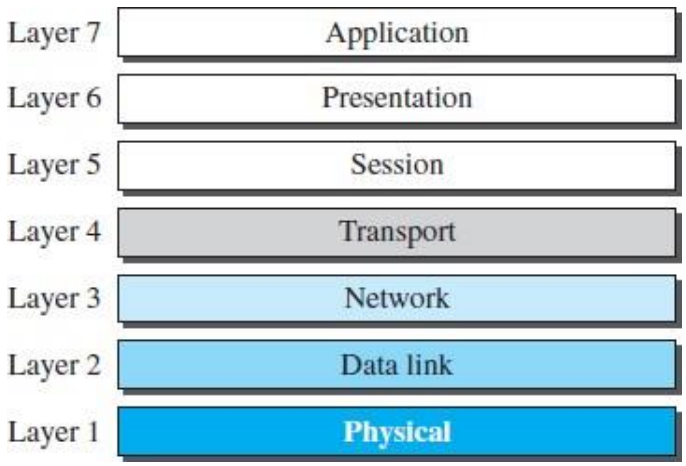


fig: OSI model

OSI versus TCP/IP

When we compare the two models, we find that two layers, session and presentation, are missing from the TCP/IP protocol suite. These two layers were not added to the TCP/IP protocol suite after the publication of the OSI model. The application layer in the suite is usually considered to be the combination of three layers in the OSI model, as shown in Figure.

Two reasons were mentioned for this decision. First, TCP/IP has more than one transport-layer protocol.
Some of the functionalities of the session layer are available in some of the transport- layer protocols.

Second, the application layer is not only one piece of software. Many applications can be developed at this layer. If some of the functionalities mentioned in the session and presentation layers are needed for a particular application, they can be included in the development to that software.

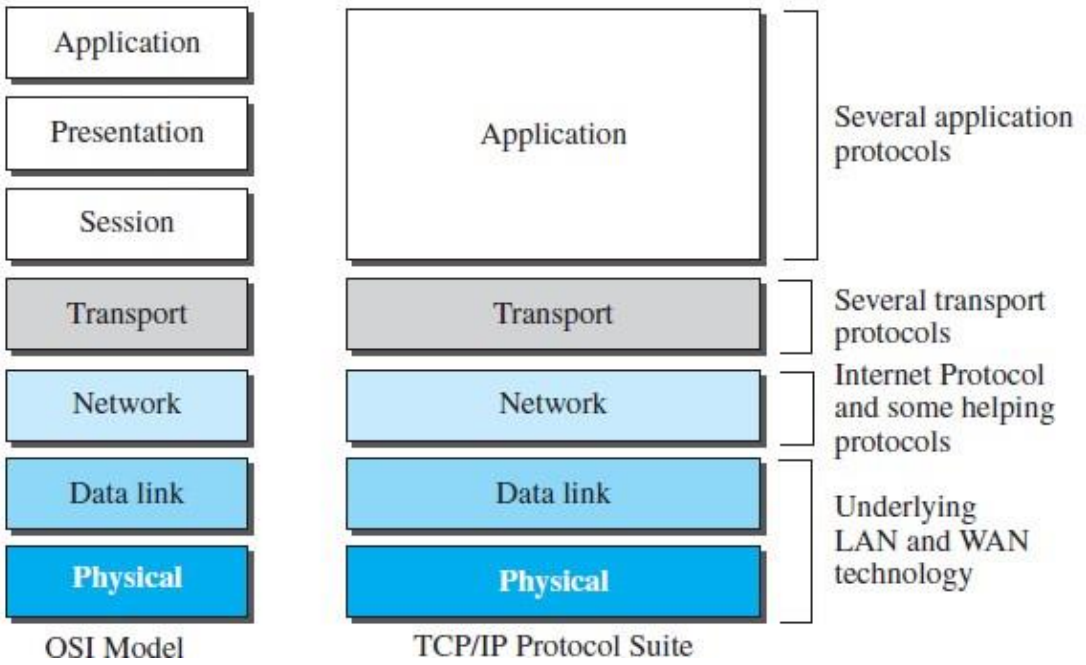


fig: TCP/IP and OSI model

Module 2

DATA-LINKLAYER

INTRODUCTION

The Internet is a combination of networks glued together by connecting devices (routers or switches). If a packet is to travel from a host to another host, it needs to pass through these networks. Communication at the data-link layer is made up of five separate logical connections between the data-link layers in the path.

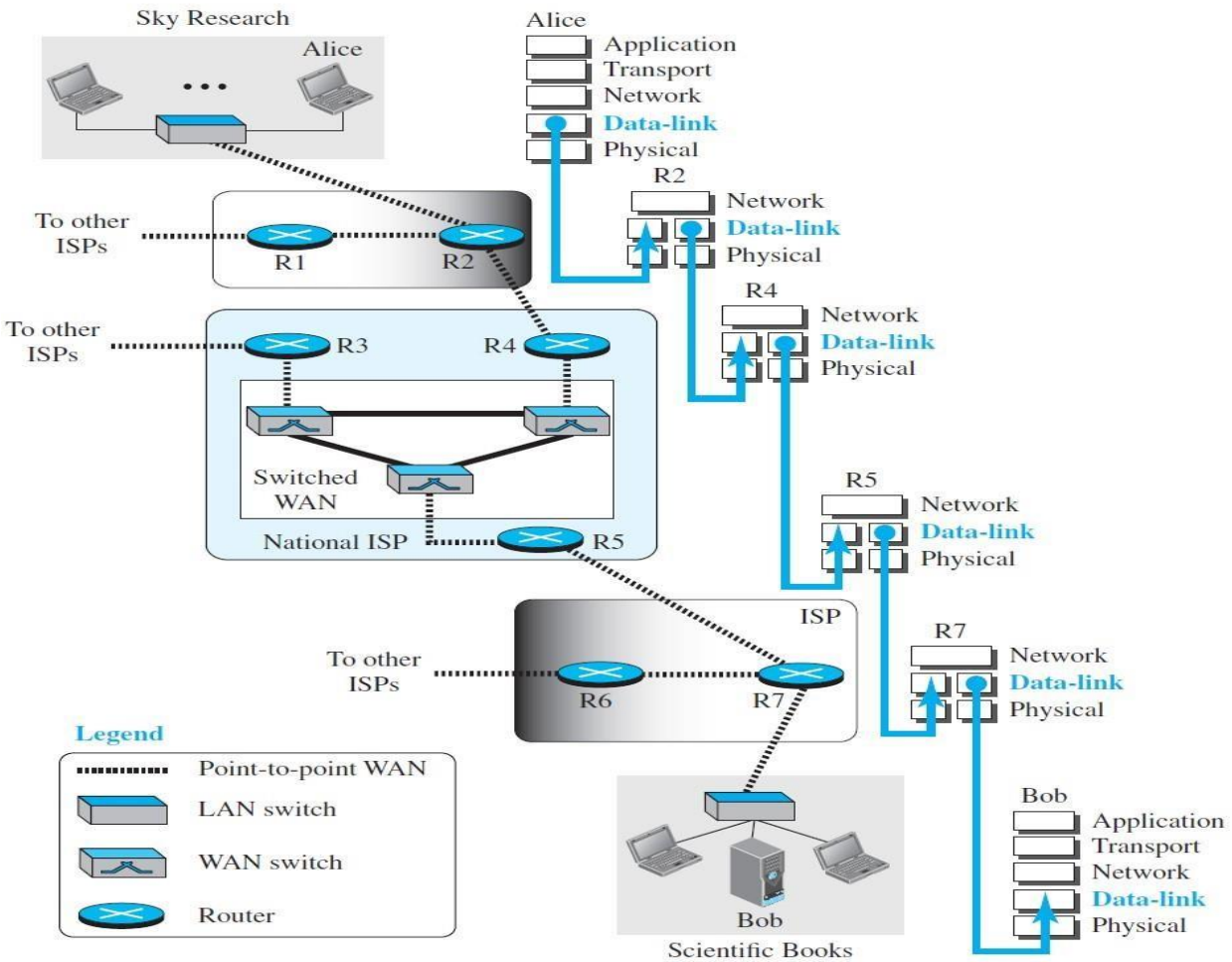


fig: communication at the data-link layer

The data-link layer at Alice's computer communicates with the data-link layer at routerR2.The data-link layer at routerR2communicateswiththedata-linklayerat router R4,

and so on. Finally, the data-link layer at router R7 communicates with the data-link layer at Bob's computer. Only one data-link layer is involved at the source or the destination, but two data-link layers are involved at each router. The reason is that Alice's and Bob's computers are

each connected to a single network, but each router takes input from one network and sends output to another network. Note that although switches are also involved in the data-link-layer communication, for simplicity we have not shown them in the figure.

Nodes and Links

Communication at the data-link layer is **node-to-node**. A data unit from one point in the Internet needs to pass through many networks (LANs and WANs) to reach another point. Theses LANs and WANs are connected by routers. It is customary to refer to the two end hosts and the routers as nodes and the networks in between as links. Figure shows representation of links and nodes when the path of the data unit is only six nodes.

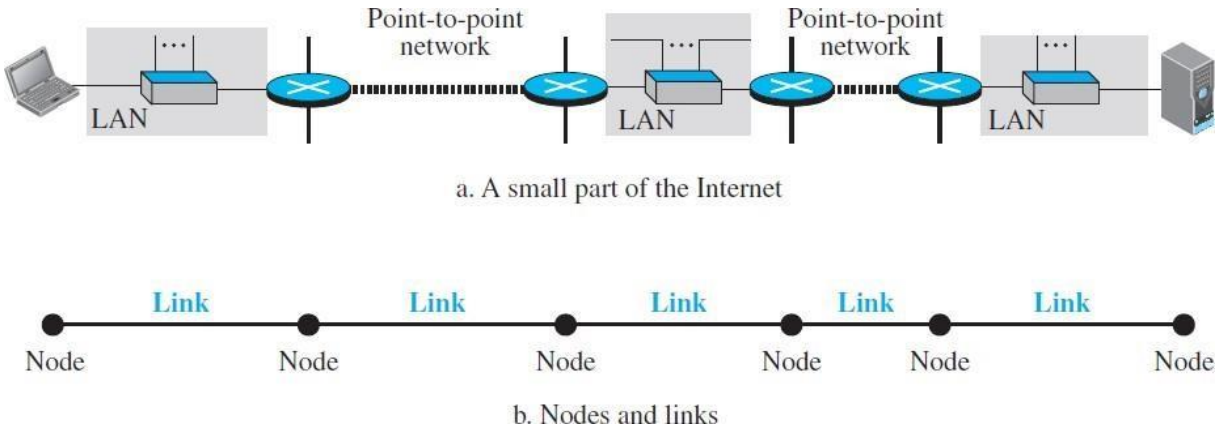


fig: links and nodes

The first node is the source host; the last node is the destination host. The other four nodes are four routers. The first, the third, and the fifth links represent the three LANs; the second and the fourth links represent the two WANs.

Services

The data-link layer is located between the physical and the network layers. The data link layer provides services to the network layer; it receives services from the physical layer.

Services provided by the data-link layer.

The duty scope of the data-link layer is node-to-node. When a packet is travelling in the Internet, the datalink layer of anode (host or router) is responsible for delivering a datagram to then ext node in the path. For this purpose, the data-link layer of the sending node needs to encapsulate the datagram received from the network in a frame, and the data-link layer of the receiving node needs to decapsulate the datagram from the frame. In other words, the data-

link layer of the source host needs only to encapsulate, the data-link layer of the destination host needs to decapsulate, but each intermediate node needs to both encapsulate and decapsulate.

One may ask why we need encapsulation and decapsulation at each intermediate node. The reason is that each link maybe using a different protocol with a different frame format. Even if one link and the next are using the same protocol, encapsulation and decapsulation are needed because the link-layer addresses are normally different.

Analogy: -may help in this case. Assume a person needs to travel from her home to her friend's home in another city. The traveller can use three transportation tools. She can take a taxi to go to the train station in her own city, then travel on the train from her own city to the city where her friend lives, and finally reach her friend's home using another taxi. Here we have a source node, a destination node, and two intermediate nodes. The traveller needs to get into the taxi at the source node, get out of the taxi and get into the train at the first intermediate node (train station in the city where she lives), get out of the train and get into another taxi at these con intermediate node (train station in the city where her friend lives), and finally get out of the taxi when she arrives at her destination. A kind of encapsulation occurs at the source node, encapsulation and decapsulation occur at the intermediate nodes, and decapsulation occurs at the destination node. Our traveller is the same, but she uses three transporting tools to reach the destination. Figure shows the encapsulation and decapsulation at the data-link layer.

For simplicity, we have assumed that we have only one router between the source and destination. The datagram received by the data-link layer of the source host is encapsulated in a frame. The frame is logically transported from the source host to the router. The frame is decapsulated at the data-link layer of the router and encapsulated at another frame. The new frame is logically transported from the router to the destination host. Note that, although we have shown only two data-link layers at the router, the router actually has three data-link layers because it is connected to three physical links.

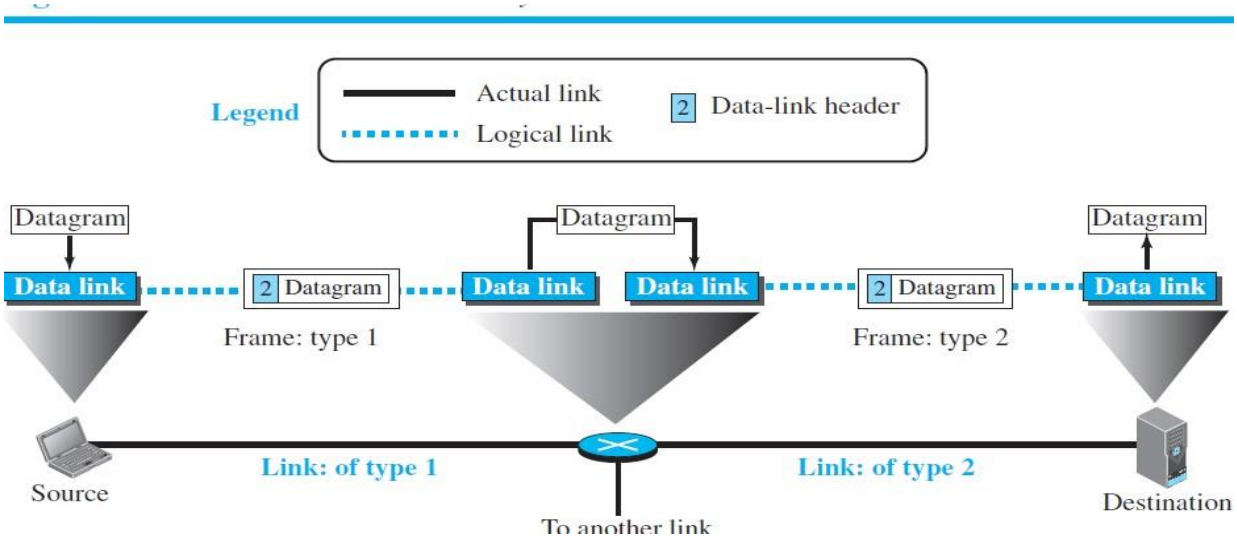


fig: communication with only three nodes

Framing

The first service provided by the data-link layer is framing. The data-link layer at each node needs to encapsulate the datagram (packet received from the network layer)in a frame before sending it to the next node. The node also needs to decapsulate the datagram from the frame received on the logical channel. Although we have shown only a header for a frame, frame may have both a header and a trailer. Different data-link layers have different formats for framing.

Flow Control

If the rate of produced frames is higher than the rate of consumed frames, frame sat the receiving end need to be buffered while waiting to be consumed (processed). Definitely, we cannot have an unlimited buffer size at the receiving side. We have two choices.

The first choice is to let the receiving data-link layer drop the frames if its buffer is full. The second choice is to let the receiving data-link layer send feedback to the sending data-link layer to ask it to stop or slow down.

Different data-link-layer protocols use different strategies for flow control. Since flow control also occurs at the transport layer, with a higher degree of importance.

Error Control

At the sending node, a frame in a data-link layer needs to be changed to bits, transformed to electromagnetic signals, and transmitted through the transmission media. At the receiving node,

electromagnetic signals are received, transformed to bits, and put together to create a frame. Since electromagnetic signals are susceptible to error, a frame is susceptible to error.

The error needs first to be detected. After detection, it needs to be either corrected at the receiver node or discarded and retransmitted by the sending node. Since error detection and correction is an issue in every layer (node-to node or host-to-host).

Congestion Control

Although a link may be congested with frames, which may result in frame loss, most data-link-layer protocols do not directly use a congestion control to alleviate congestion, although some wide-area networks do. In general, congestion control is considered an issue in the network layer or the transport layer because of its end-to-end nature.

Two Categories of Links

In a **point-to-point link**, the link is dedicated to the two devices; in a **broadcast link**, the link is shared between several pairs of devices. For example, when two friends use the traditional home phones to chat, they are using a point-to-point link; when the same two friends use their cellular phones, they are using a broadcast link (the air is shared among many cell phone users).

Two Sub layers

The data-link layer is divided into two sublayers: data link control (DLC) and media access control (MAC). LAN protocols actually use the same strategy. **The data link control sublayer** deals with all issues common to both point-to-point and broadcast links; **the media access control sublayer** deals only with issues specific to broadcast links. In other words, we separate these two types of links at the data-link layer, as shown in fig

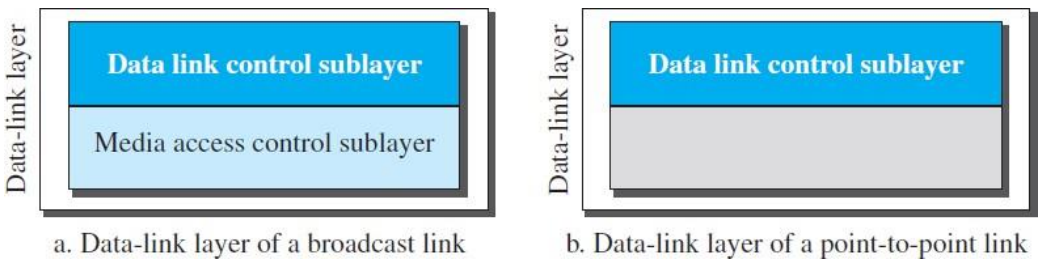


fig: dividing the data link layer in to two sublayers

LINK-LAYERADDRESSING

In a connectionless internetwork such as the Internet we cannot make a datagram reach its destination using only IP addresses. The reason is that each datagram in the Internet, from the same source host to the same destination host, may take a different path. The source and destination IP addresses define the two ends but cannot define which links the datagram should pass through. We need to remember that the IP addresses in a datagram should not be changed. If the destination IP address in a datagram changes, the packet never reaches its destination; if the source IP address in a datagram changes, the destination host or a router can never communicate with the source if a response needs to be sent back or an error needs to be reported back to the source (ICMP).

The above discussion shows that we need another addressing mechanism in a connection less internetwork: the link-layer addresses of the two nodes. A link-layer address is sometimes called a **link address, sometimes a physical address, and sometimes a MAC address**. Since a link is controlled at the data-link layer, the addresses need to belong to the data-link layer. When a datagram passes from the network layer to the data-link layer, the datagram will be encapsulated in a frame and two data-link addresses are added to the frame header. These two addresses are changed every time the frame moves from one link to another. Figure demonstrates the concept in a small internet.

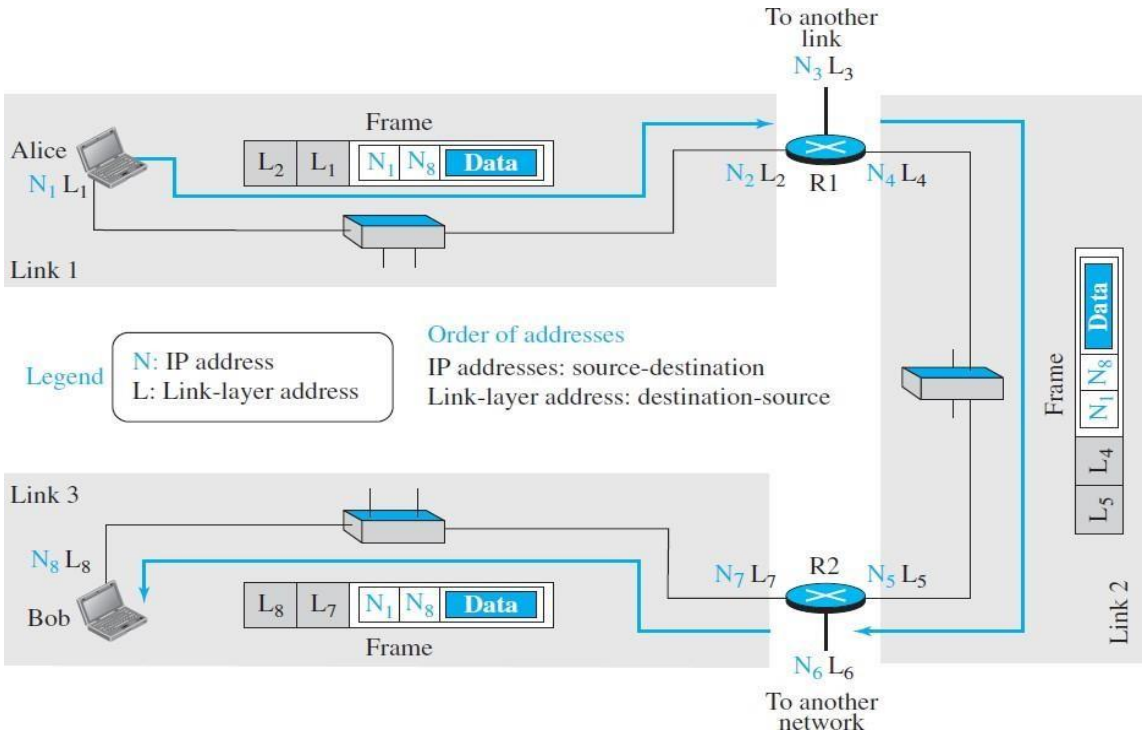


fig: IP address and link layer addresses in a small internet

above Figure each host, two addresses, the IP addresses(N) and the link-layer addresses shows three links and two routers and also have only two hosts: Alice (source) and Bob (destination). For (L) are shown. Note that a router has as many pairs of addresses as the number of links the router is connected to. We have shown three frames, one in each link. Each frame

carries the same datagram with the same source and destination addresses (N1 and N8), but the link-layer addresses of the frame change from link to link.

In link 1, the link-layer addresses are L1 and L2. In link 2, they are L4 and L5. In link 3, they are L7 and L8. Note that the IP addresses and the link-layer addresses are not in the same order. For IP addresses, the source address comes before the destination address; for link-layer addresses, the destination address comes before the source. The datagrams and frames are designed in this way, and we follow the design. We may raise several questions:

❑The IP address of a router does not appear in any datagram sent from a source to a destination, why do we need to assign IP addresses to routers? The answer is that in some protocols a router may act as a send error receiver of a datagram. For example, in routing protocols a router is a sender or a receiver of a message. The communications in these protocols are between routers.

❑Why do we need more than one IP address in a router, one for each interface? The answer is that an interface is a connection of a router to a link. We will see that an IP address defines a point in the Internet at which a device is connected. A router with n interfaces is connected to the Internet at n points. This is the situation of a house at the corner of a street with two gates; each gate has the address related to the corresponding street.

❑How are the source and destination IP addresses in a packet determined? The answer is that the host should know its own IP address, which becomes the source IP address in the packet. the application layer uses the services of DNS to find the destination address of the packet and passes it to the network layer to be inserted in the packet.

❑How are the source and destination link-layer addresses determined for each link? Again, each hop (router or host) should know its own link-layer address, The destination link-layer address is determined by using the Address Resolution Protocol.

❑What is the size of link-layer addresses? The answer is that it depends on the protocol used by the link. Although we have only one IP protocol for the whole Internet, we may be using different data-link protocols in different links.

Different link-layer protocols.

Three Types of addresses

Some link-layer protocols define three types of addresses: unicast, multicast and broadcast.

Unicast Address: Each host or each interface of a router is assigned a unicast address. Unicasting means one-to-one communication. A frame with a unicast address destination is destined only for one entity in the link.

Example: The unicast link-layer addresses in the most common LAN, Ethernet, are 48 bits (six bytes) that are presented as12 hexadecimal digits separated by colons; for example, the following is a link-layer address of a computer.

A2:34:45: 11:92: F1

Multicast Address: Some link-layer protocols define multicast addresses. Multicasting means one-to many communications. However, the jurisdiction is local (inside the link).

Example: the multicast link-layer addresses in the most common LAN, Ethernet, are 48 bits (six bytes) that arepresentedas12hexadecimaldigitsseparatedbycolons.Theseconddigit, however, needs to be an even number in hexadecimal. The following shows a multicast address:

A3:34:45: 11:92: F1

Broadcast Address: Some link-layer protocols define a broadcast address. Broadcasting means one-to-all communication. A frame with a destination broadcast address is sent to all entities in the link.

Example: the broadcast link-layer addresses in the most common LAN, Ethernet, are 48bits, all 1s, that are presented as 12 hexadecimal digits separated by colons. The following shows a broadcast address:

FF: FF: FF: FF: FF: FF

Address Resolution Protocol (ARP)

Anytime a node has an IP datagram to send to another node in a link, it has the IP address of the receiving node. The source host knows the IP address of the default router. Each router except the last one in the path gets the IP address of the next router by using its forwarding table. The last router knows the IP address of the destination host. However, the IP address of the next node is not helpful in moving a frame through a link; we need the link-layer address of the next node. This is the time when the Address Resolution Protocol (ARP)becomes helpful.

The ARP protocol is one of the auxiliary protocols defined in the network layer, as shown in Figure. It belongs to the network layer; it maps an IP address to a logical-link address. ARP accepts an IP address from the IP protocol, maps the address to the corresponding link-layer address, and passes it to the data-link layer.

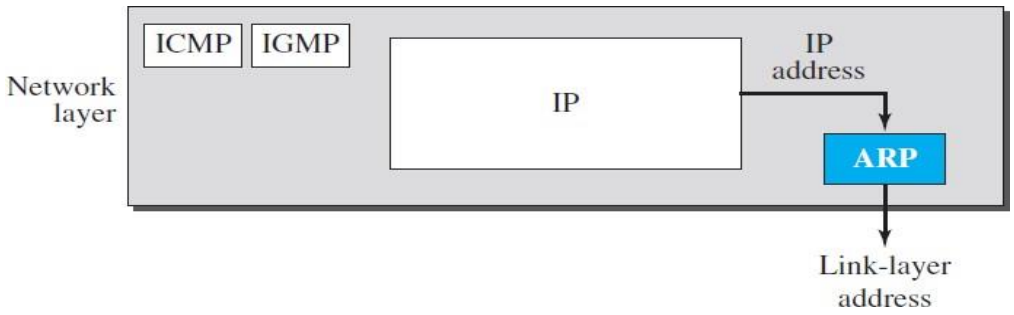
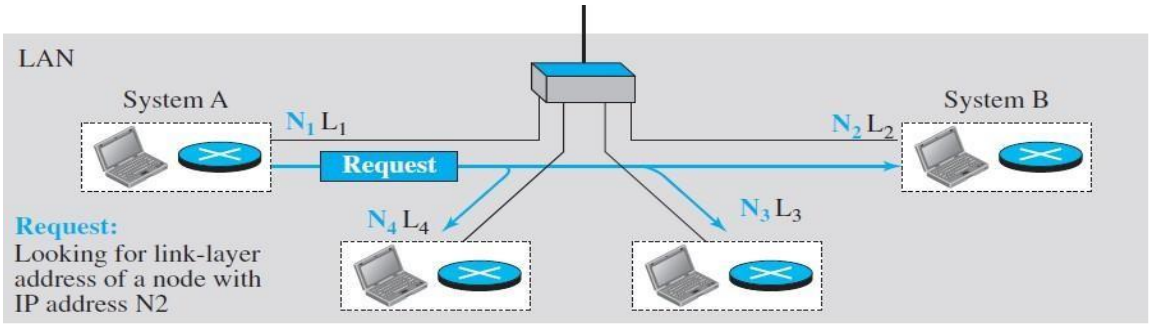
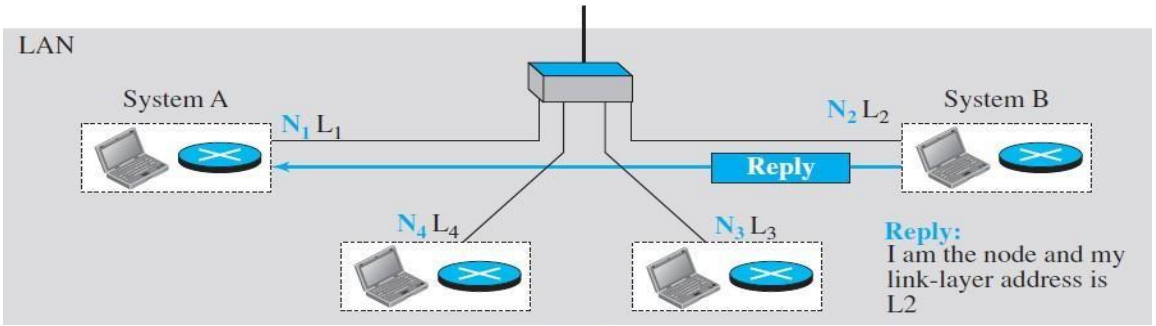


fig: Position of ARP in TCP/IP protocol suite

Anytime a host or a router needs to find the link-layer address of another host or router in its network, it sends an ARP request packet. The packet includes the link-layer and IP addresses of the sender and the IP address of the receiver. Because the sender does not know the link-layer address of the receiver, the query is broadcast over the link using the link-layer broadcast address.



a. ARP request is broadcast



b. ARP reply is unicast

fig: ARP operation

Every host or router on the network receives and processes the ARP request packet, but only the intended recipientrecognizesitsIPaddressandsendsbackanARPresponsepacket.The response packet contains the recipient's IP and link-layer addresses.

The packet is unicast directly to the node that sent the request packet. In Figure (a) the system on the left (A) has a packet that needs to be delivered to another system (B) with IP address N2. 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 IP address of N2. This packet is received by every system on the physical network, but only system B will answer it, as shown in Figure (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 using the physical address it received.

Caching

Let us assume that there are 20systems connected to the network (link): system A, system B, and18 other systems. We also assume that system A has10datagramsto send to system B in one second.

a. Without using ARP, system A needs to send 10 broadcast frames. Each of the 18 other systems need to receive the frames, decapsulate the frames, remove the datagram and pass it to their network-layer to find out the datagrams do not belong to them. This means processing anddiscarding180broadcast frames.

b. Using ARP, system A needs to send only one broadcast frame. Each of the 18 other systems need to receive the frames, decapsulate the frames, remove the ARP message and pass the message to their ARP protocol to find that the frame must be discarded. This means processing and discarding only18 (instead of 180) broadcast frames. After system B responds with its own data-link address, system A can store the link layer address in its cache memory. The rest of the nine frames are only unicast. Since processing broadcast frames is expensive (time consuming), the first method is preferable.

Packet Format

Figure shows the format of an ARP packet.

The hardware type field - defines the type of the link-layer protocol; Ethernet given the type 1. The

protocol type field- defines the network-layer protocol: IPv4protocol is (0x0800).

The source hardware and source protocol addresses are variable-length fields defining the link-layer and network-layer addresses of the sender.

The destination hardware address and destination protocol address fields define the receiver link-layer and network-layer addresses.

An ARP packet is encapsulated directly into a data-link frame. The frame needs to have a field to show that the payload belongs to the ARP and not to the network-layer datagram

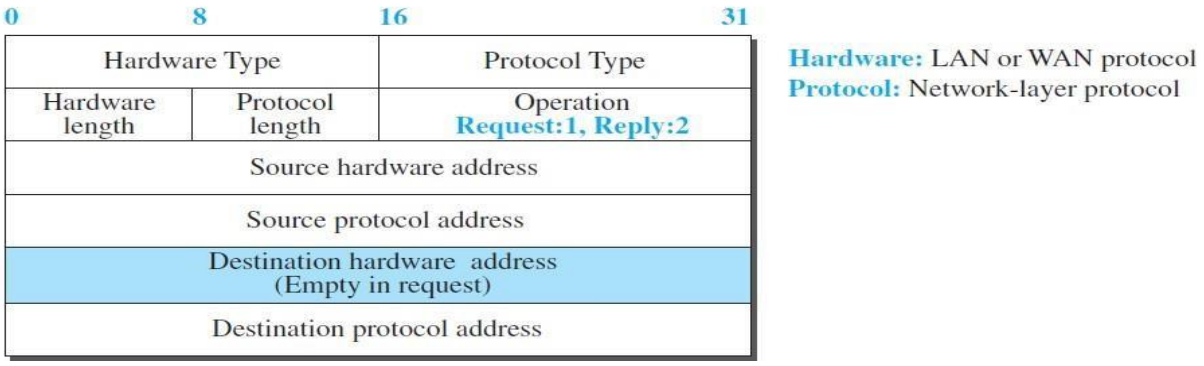
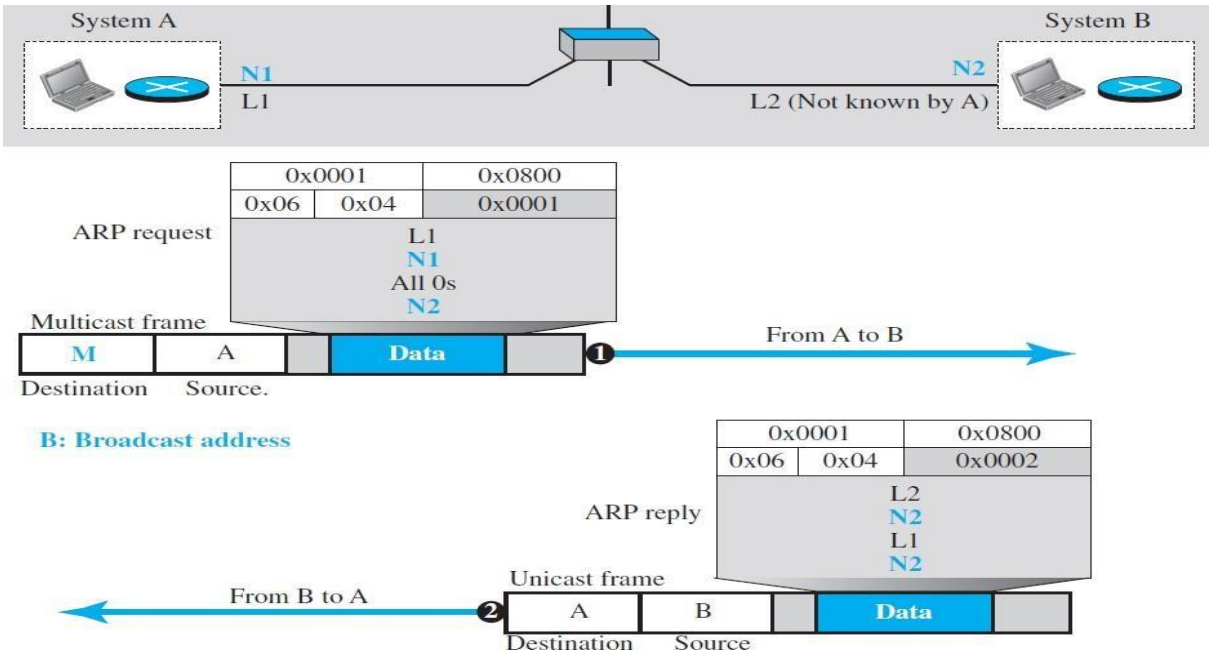


fig: ARP packet

Example: A host with IP address N1 and MAC address L1 has a packet to send to another host with IP address N2 and physical address L2 (which is unknown to the first host). The two hosts are on the same network. Figure shows the ARP request and response messages



DATA LINK CONTROL

DLC SERVICES

The data link control (DLC) deals with procedures for communication between two adjacent nodes—node to-node communication—no matter whether the link is dedicated or broadcast. Data link control functions include framing and flow and error control.

Framing Data

Transmission in the physical layer means moving bits in the form of a signal from the source to the destination. The physical layer provides bit synchronization to ensure that the sender and receiver use the same bit durations and timing.

The data-link layer, on the other hand, needs to pack bits into frames, so that each frame is distinguishable from another.

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

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 frame. When a message is divided in to smaller frames, a single-bit error affects only that small frame. **Frame Size**

Frames can be of fixed or variable size.

Fixed-size framing: there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. An example of this type of framing is the (Asynchronous transfer mode) ATM WAN, which uses frames of fixed size called cells.

Variable-size framing: prevalent in local-area networks. Invariable- size framing, we need a way to define the end of one frame and the beginning of the next.

Two approaches were used for this purpose: a character-oriented approach and a bit-oriented approach.

Character-Oriented Framing

In character-oriented (or byte-oriented) framing, 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 redundant bits, are also multiples of 8 bits.

To separate one frame from the next, an 8-bit (1-byte) flag is added at the beginning and the end of a frame. The flag, composed of **protocol-dependent special characters**, signals the start or end of a frame. Figure shows the format of a frame in a character-oriented protocol.

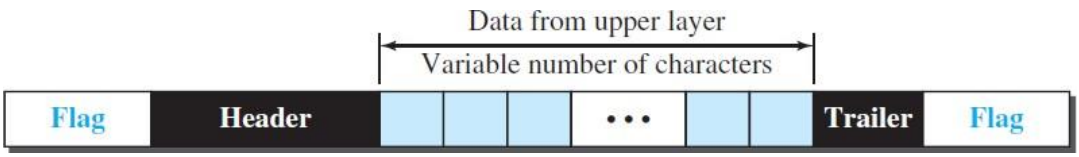


fig: A frame in a character-oriented protocol

Character-oriented framing was popular when only **text was exchanged** by the data-link layers. The flag could be selected to be any character not used for text communication. Now, however, we send other types of information such as graphs, audio and video;

Any character used for the flag could also be part of the information. If this happens, the receiver, when it encounters this pattern in the middle of the data, thinks it has reached the end of frame.

To fix this problem**, a byte-stuffing** strategy was added to character-oriented framing. In byte stuffing (or character stuffing), a special byte is added to the data section of the frame when there is a character with the same pattern as the flag. The data section is stuffed with an extra byte. This byte is usually called the escape character (ESC) and has a predefined bit pattern. Whenever the receiver encounters the ESC character, it removes it from the data section and

treats the next character as data, not as a delimiting flag. Figure shows the situation.

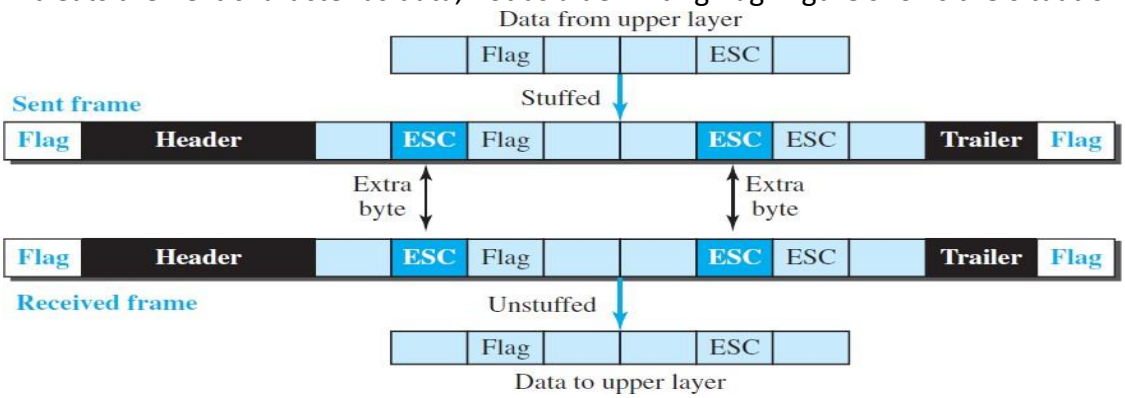


fig: Byte stuffing and unstuffing

Byte stuffing by the escape character allows the presence of the flag in the data section of the frame, but it creates another problem. What happens if the text contains one or more escape characters followed by a byte with the same pattern as the flag?

The receiver removes the escape character, but keeps the next byte, which is incorrectly interpreted as the end of the frame. To solve this problem, the escape characters that are part of the text must also be marked by another escape character. In other words, if the escape character is part of the text, an extra one is added to show that these can done is part of the text.

Character-oriented protocols present another problem in data communications. The universal coding systems in use today, such as Unicode, have 16-bit and32-bit characters that conflict with 8-bit characters. We can say that, in general, the tendency is moving toward the bit- oriented protocols that we discuss next.

Bit-Oriented Framing

In bit-oriented framing, 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 fig**

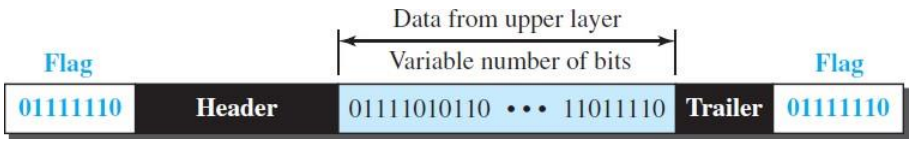


fig: A frame in bit-oriented protocol

If the flag pattern appears in the data, 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. In bit stuffing, if 0 and five consecutive 1 bits are encountered, an extra0 is added. This extra stuffed bit is eventually removed from the data by the receiver. Note that the extra bit is added after one 0 followed by five 1s regardless of the value of the next bit. This guarantees that the flag field sequence does not in advertently appear in the frame.

Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that thereceiverdoesnotmistakethepattern0111110 fora flag.

Figure 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 for a flag by the receiver. There al flag 01111110 is not stuffed by the sender and is recognized by the receiver.

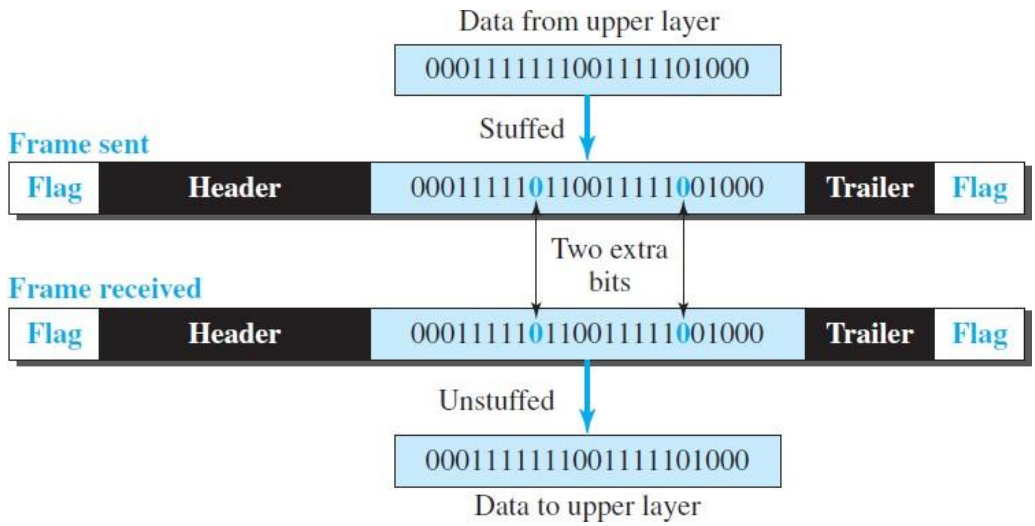


fig: Bit stuffing and unstuffing

Flow and Error Control

One of the responsibilities of the data-link control sublayer is flow and error control at the data-link layer.

Flow Control Whenever an entity produces items and another entity consumes them, there should be a balance between production and consumption rates. If the items are produced faster than they can be consumed, the consumer can be overwhelmed and may need to discard some items. If the items are produced more slowly than they can be consumed, the consumer must wait, and the system becomes less efficient. Flow control is related to the first issue. We need to prevent losing the data items at the consumer site.

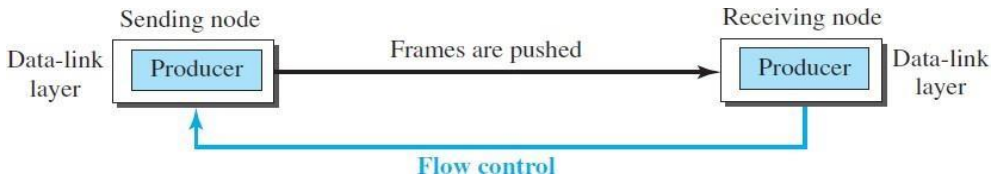


fig: Flow control at the data link layer

The figure shows that the data-link layer at the sending node tries to push frames toward the data-link layer at the receiving node. If the receiving node cannot process and deliver the packet to its network at the same rate that the frames arrive, it becomes overwhelmed with frames. Flow control in this case can be feedback from the receiving node to the sending node to stop or slow down pushing frames.

Buffers

Although flow control can be implemented in several ways, one of the solutions is normally to use two buffers; one at the sending data-link layer and the other at the receiving data-link layer. A buffer is a set of memory locations that can hold packets at the sender and receiver. The flow control communication can occur by sending signals from the consumer to the producer. When the buffer of the receiving data-link layer is full, it informs the sending data- link layer to stop pushing frames.

Error Control

Since the underlying technology at the physical layer is not fully reliable, we need to implement error control at the data-link layer to prevent the receiving node from delivering corrupted packets to its network layer. Error control at the data-link layer is normally very simple and implemented using one of the following two methods. In both methods, a CRC is added to the frame header by the sender and checked by the receiver.

❑In the first method, if the frame is corrupted, it is silently discarded; if it is not corrupted, the packet is delivered to the network layer. This method is used mostly in wired LANs such as Ethernet.

❑In the second method, if the frame is corrupted, it is silently discarded; if it is not corrupted, an acknowledgment is sent (for the purpose of both flow and error control) to the sender.

Combination of Flow and Error Control

Flow and error control can be combined. In a simple situation, the acknowledgment that is sent for flow control can also be used for error control to tell the sender the packet has arrived uncorrupted. The lack of acknowledgment means that there is a problem in the sent frame. We show this situation when we discuss some simple protocols in the next section. A frame that carries an acknowledgment is normally called an ACK to distinguish it from the data frame.

Connectionless and Connection-Oriented

A DLC protocol can be either connection less or connection-oriented.

Connection less Protocol

Ina connectionless protocol, frames are sent from one node to the next without any relationship between the frames; each frame is independent. Note that the term connectionless here does not mean that there is no physical connection (transmission medium) between the nodes; it means that there is no connection between frames. The frames are not numbered and there is no sense of ordering. Most of the data-link protocols for LANs are connectionless protocols.

Connection-Oriented Protocol

Ina connection-oriented protocol, a logical connection should first be established between the two nodes (setup phase). After all frames that are somehow related to each other are transmitted (transfer phase), the logical connection is terminated (teardown phase). In this type of communication, the frames are numbered and sent in order. If they are not received in order, the receiver needs to wait until all frames belonging to the same set are received and then deliver them in order to the network layer. Connection oriented protocols are rare in wired LANs, but we can see them in some point-to-point protocols, some wireless LANs and some WANs.

DATA-LINKLAYERPROTOCOLS

Traditionally four protocols have been defined for the data-link layer to deal with flow and error control: Simple, Stop-and-Wait, Go-Back-N, and Selective-Repeat.

The behaviour of a data-link-layer protocol can be better shown as a finite state machine (FSM). An FSM is thought of as a machine with a finite number of states. The figure shows a machine with three states. There are only three possible events and three possible actions. The machine starts in state I. If event 1 occurs, the machineperformsactions1and2andmovesto state II. When them a chine is in state II, two events may occur. If event 1 occurs, the machine performs action 3 and remains in the same state, state II. I fevent3occurs, them a chine performs no action, but move to state I.

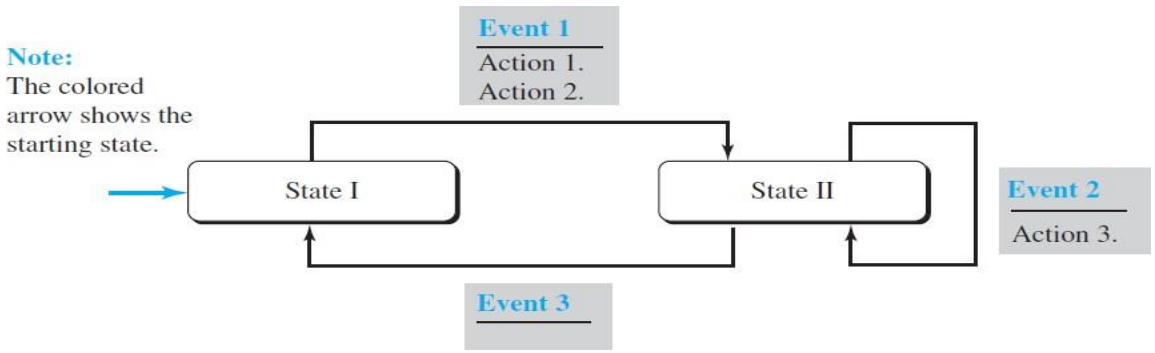


fig: Connectionless and connection-oriented service represented as FSMs

Simple Protocol

First protocol is a simple protocol with neither flow nor error control. We assume that the receiver can immediately handle any frame it receives. In other words, the receiver can never be overwhelmed with incoming frames. **Figure** shows the layout for this protocol.

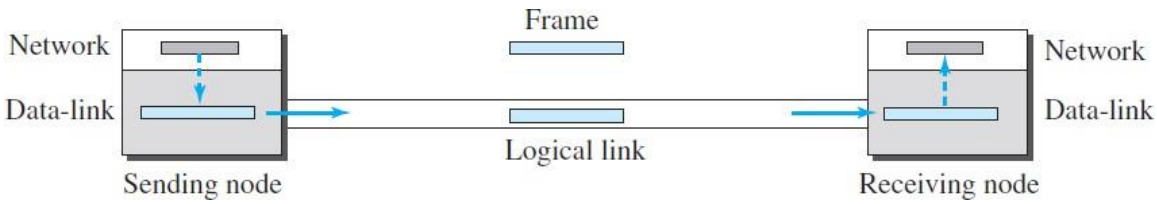


fig: Simple Protocol

The data-link layer at the sender gets a packet from its network layer, makes a frame out of it, and sends the frame. The data-link layer at the receiver receives a frame from the link, extracts

the packet from the frame, and delivers the packet to its network layer. The data-link layers of the sender and receiver provide transmission services for their network layers.

FSMs The sender site should not send a frame until its network layer has a message to send. The receiver site cannot deliver a message to its network layer until a frame arrives. We can show these requirements using two FSMs. Each FSM has only one state, the ready state.

sending machine remains in the ready state until a request comes from the process in the network layer. When this event occurs, the sending machine encapsulates the message in a frame and sends it to the receiving machine.

The receiving machine remains in the ready state until a Frame arrives from the sending machine. When this event occurs, the receiving machine decapsulates the message out of the frame and delivers it to the process at the network layer**. Figure** shows the FSMs for the simple protocol.

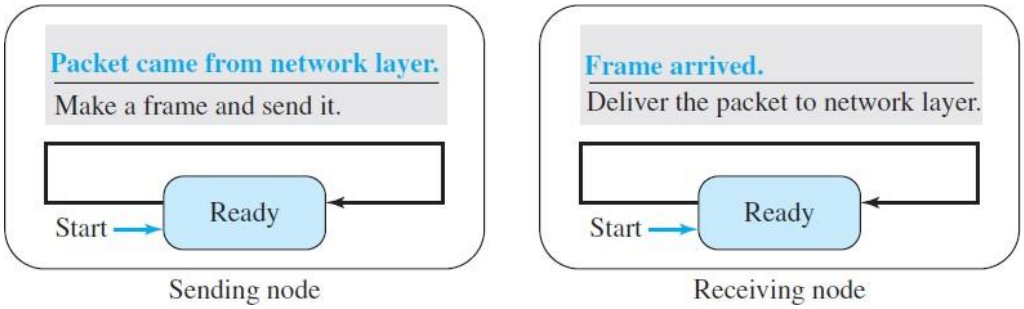


fig: FSMs for the simple protocol

Example

Flow diagram shows an example of communication using this protocol. It is very simple. The sender sends frames one after another without even thinking about the receiver.

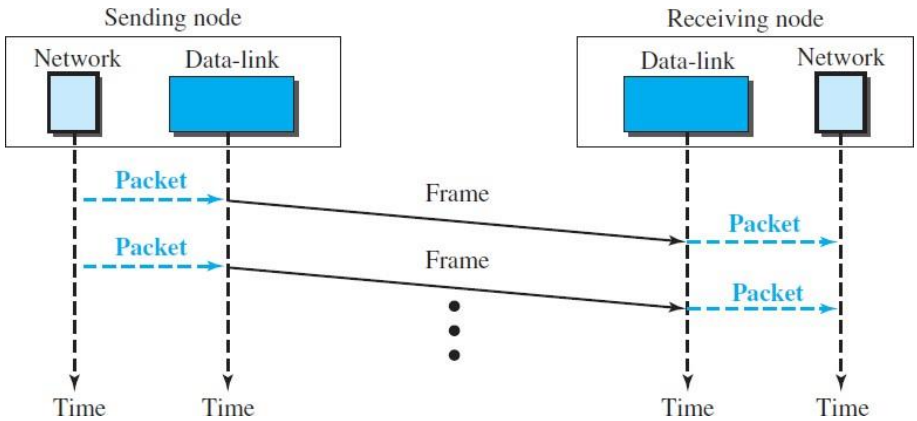


fig: Flow diagram for above example

Stop-and-Wait Protocol

Our second protocol is called the Stop-and-Wait protocol, which **uses both flow and error control**.

In this protocol, the sender sends one frame at a time and waits for an acknowledgment before sending the next one. To detect corrupted frames, we need to add a CRC to each data frame. When a frame arrives at the receiver site, it is checked. If its CRC is incorrect, the frame is corrupted and silently discarded. The silence of the receiver is assigning al for the sender that a frame was either corrupted or lost.

Every time the sender sends a frame, it starts a timer. If an acknowledgment arrives before the timer expires, the timer is stopped and the sender sends the next frame (if it has one to send). If the timer expires, the sender resends the previous frame, assuming that the frame was either lost or corrupted. This means that the sender needs to keep a copy of the frame until its acknowledgment arrives.

When the corresponding acknowledgment arrives, the sender discards the copy and sends the next frame if it is ready. Figure shows the outline for the Stop-and-Wait protocol. Note that only one frame and one acknowledgment can be in the channels at any time.

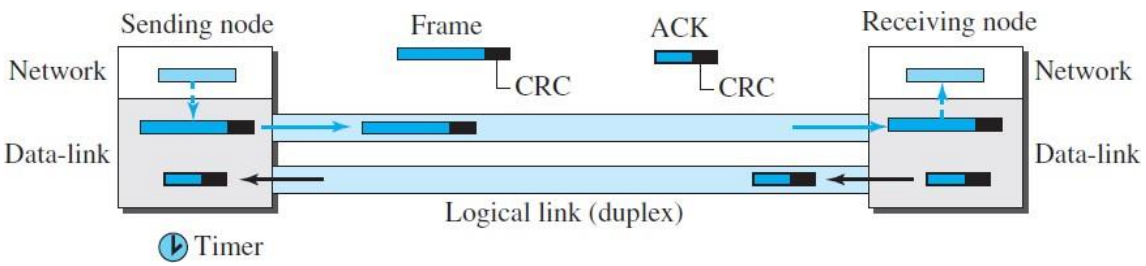


fig: Stop and Wait protocol

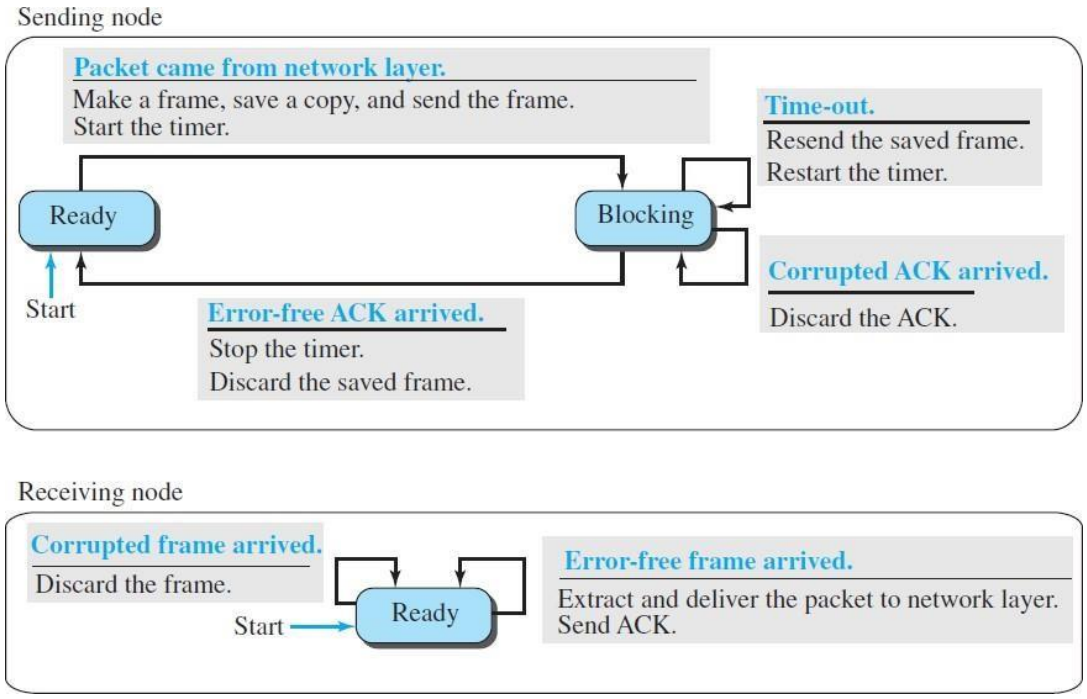


fig: FSMs for Stop-and-Wait protocol.

FSMs Figure shows the FSMs for primitive Stop-and-Wait protocol. We

describe the sender and receiver states below.

Ready State. When the sender is in this state, it is only waiting for a packet from the network layer. If a packet comes from the network layer, the sender creates a frame, saves a copy of the frame, starts the only timer and sends the frame. The sender then moves to the blocking state.

❑ **Blocking State.** When the sender is in this state, three events can occur:

a. If a time-out occurs, the sender resends the saved copy of the frame and restarts the timer.

b. If a corrupted ACK arrives, it is discarded.

c. If an error-free ACK arrives, the sender stops the timer and discards the saved copy of the frame. It then moves to the ready state.

Receiver

The receiver is always in the ready state. Two events may occur:

a. If an error-free frame arrives, the message in the frame is delivered to the network layer and an ACK is sent.

b. If a corrupted frame arrives, the frame is discarded

Example

Flow diagram for this example is shown below.

The first frame is sent and acknowledged. The second frame is sent, but lost. After time-out, it is resent. The third frame is sent and acknowledged, but the acknowledgment is lost. The frame is resent. However, there is a problem with this scheme. The network layer at the receiver site receives two copies of the third packet, which is not right. In the next section, we will see how we can correct this problem using sequence numbers and acknowledgment numbers.

Sequence and Acknowledgment Numbers

Problem in above Example needs to be addressed and corrected. Duplicate packets, as much as corrupted packets, need to be avoided. we need to add sequence numbers to the data frames and acknowledgment numbers to the ACK frames. However, numbering in this case is very simple. Sequence numbers are 0, 1, 0,
1, 0, 1, . . .; theacknowledgmentnumberscanalsobe1,
0, 1, 0, 1, 0, …In other words, the sequence numbers start with 0, the acknowledgment numbers start with 1. **An acknowledgment number always defines the sequence number of the next frame to receive.**

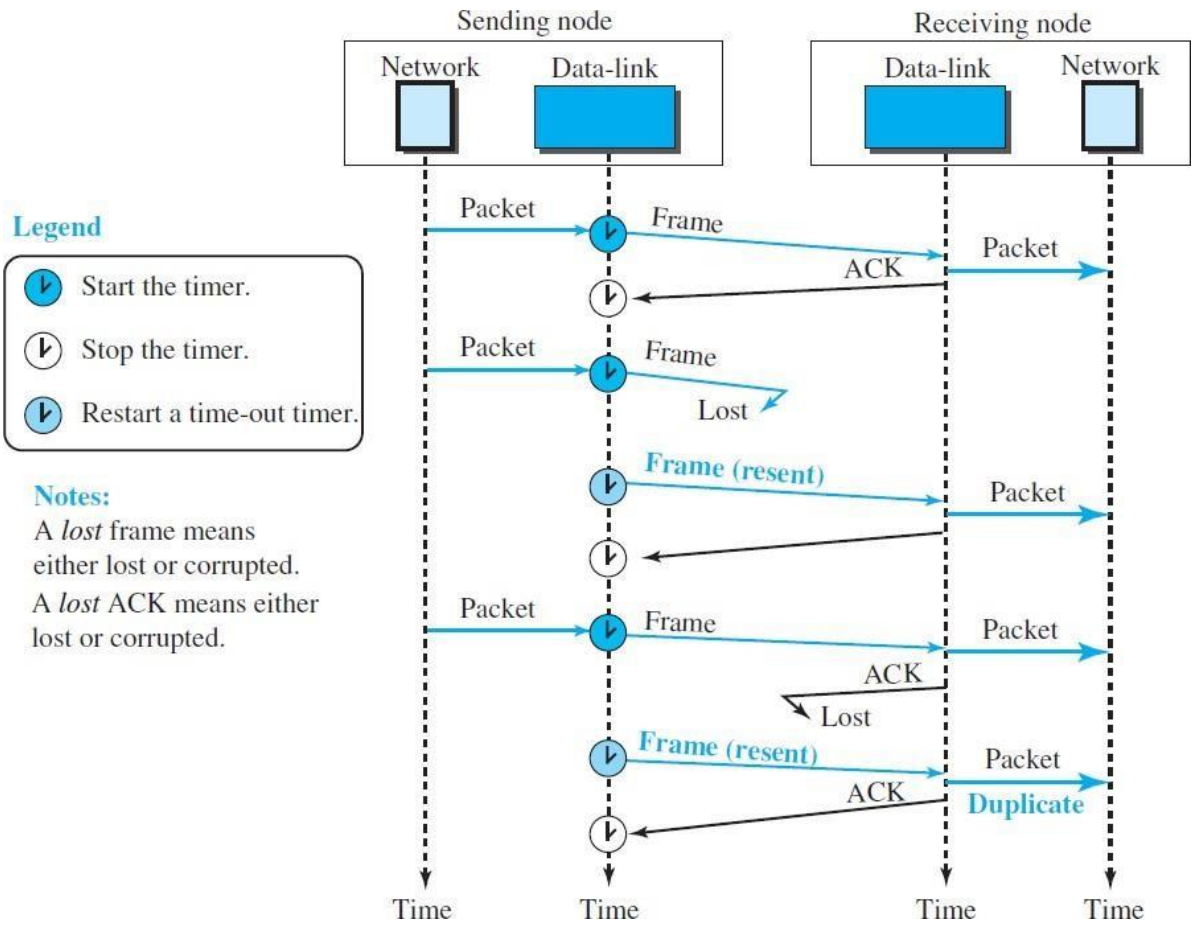


fig: flow diagram for example

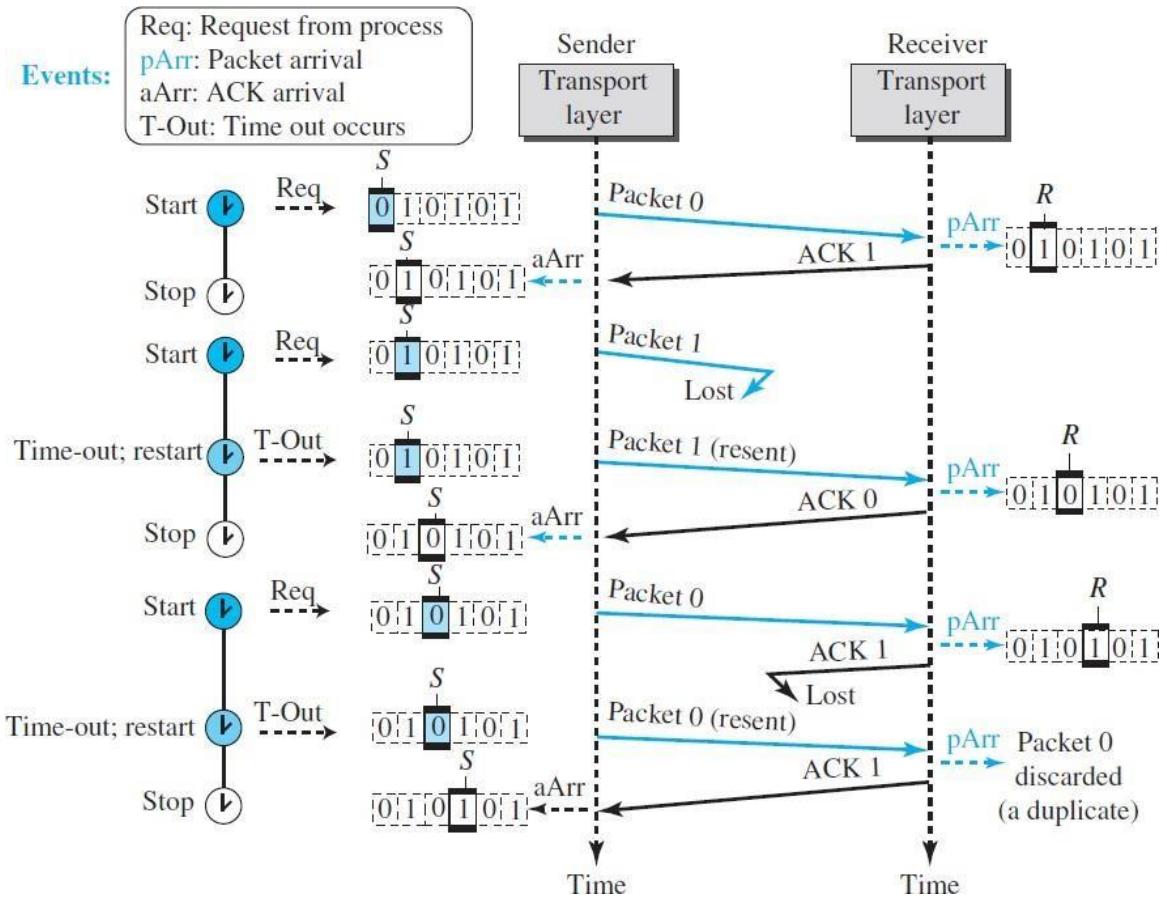
Figure below shows how adding sequence numbers and acknowledgment numbers can prevent duplicates.

The first frame is sent and acknowledged. The second frame is sent, but lost. After time-out, it is resent.

The third frame is sent and acknowledged, but the acknowledgment is lost. The frame is resent

FSMs with Sequence and Acknowledgment Numbers

We can change the FSM in Figure (FSM for stop and wait protocol) to include the sequence and acknowledgment numbers.

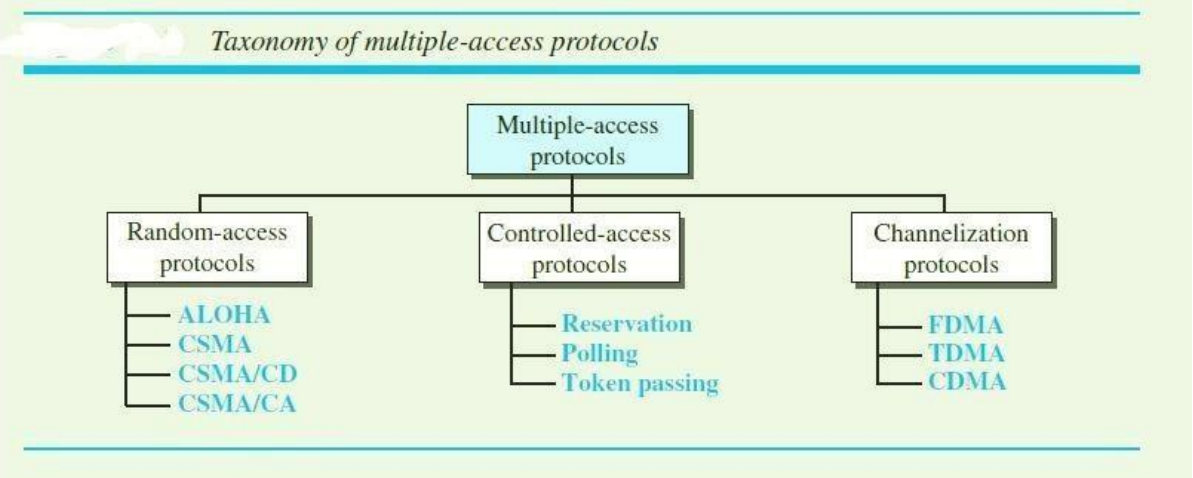


Piggybacking

The two protocols discussed in this section are designed for unidirectional communication, in which data is flowing only in one direction although the acknowledgment may travel in the other direction. Protocols have been designed in the past to allow data to flow in both directions. However, to make the communication more efficient, the data in one direction is piggybacked with the acknowledgment in the other direction. In other words, when node A is sending data to node B, Node A also acknowledges the data received from node B. Because piggybacking makes communication at the data link layer more complicated, it is not a common practice.

Media Access Control (MAC)

- When nodes or stations are connected and use a common link, called a multipoint or broadcast link, we need a multiple-access protocol to coordinate access to the link.
- Many protocols have been devised to handle access to a shared link. All of these protocols belong to a sub layer in the data-link layer called media access control (MAC).



RANDOM ACCESS

- In **random-access** or **contention** methods, no station is superior to another station and none is assigned control over another.
- At each instance, a station that has data to send uses a procedure defined by the protocol to make a decision on whether or not to send.
- This decision depends on the state of the medium (idle or busy). In other words, each station can transmit when it desires on the condition that it follows the predefined procedure, including testing the state of the medium.
- Two features give this method its name. First, there is no scheduled time for a station to transmit. Transmission is random among the stations. That is why these methods are called **random access**. Second, no rules specify which station should send next. Stations compete with one another to access the medium. That is why these methods are also called **contention methods.**

- In a random-access method, each station has the right to the medium without being controlled by any other station. However, if more than one station tries to send, there is an access conflict "**collision**" and the frames will be either destroyed or modified.
- The random-access methods have evolved from a very interesting protocol known as ALOHA, which used a very simple procedure called **multiple access (MA).**
- The method was improved with the addition of a procedure that forces the station to sense the medium before transmitting. This was called **carrier sense multiple access (CSMA).**
- CSMA method later evolved into two parallel methods: **carrier sense multiple access with collision detection (CSMA/CD),** which tells the station what to do when a collision is detected, and **carrier sense multiple access with collision avoidance (CSMA/CA)**, which tries to avoid the collision.

ALOHA

- ALOHA**,** the earliest random-access method, was developed at the University of Hawaii in early 1970. It was designed for a radio (wireless) LAN, but it can be used on any shared medium.
- The medium is shared between the stations. When a station sends data, another station may attempt to do so at the same time. The data from the two stations collide and become garbled.

Pure ALOHA

- The original ALOHA protocol is called **pure ALOHA.** This is a simple but elegant protocol.
- The idea is that each station sends a frame whenever it has a frame to send (multiple access) there is only one channel to share, there is the possibility of collision between frames from different stations.

Figure 1 below shows an example of frame collisions in pure ALOHA

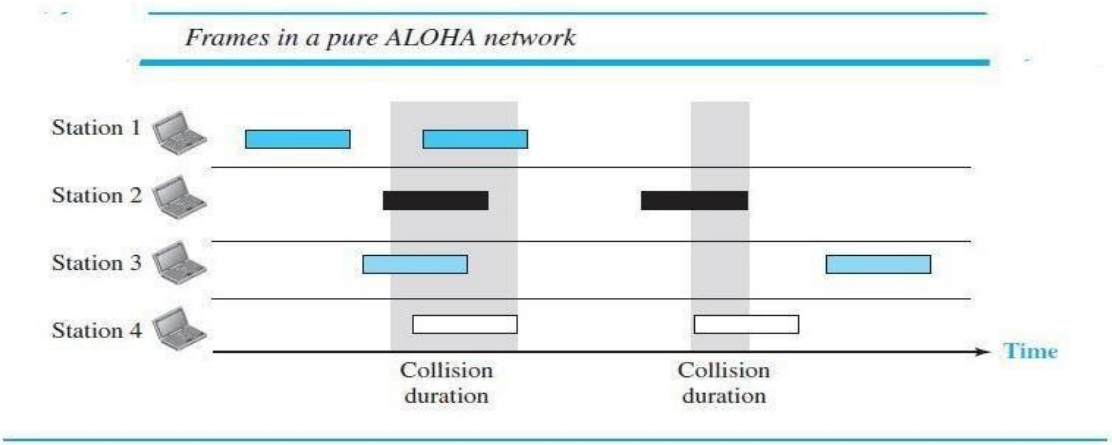


Figure 1: Frames in pure ALOHA network

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

- If one bit of a frame coexists on the channel with one bit from another frame, there is a collision and both will be destroyed. It is obvious that the frames have to be resend 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 (or the acknowledgment) has been destroyed and resends the frame.
- A collision involves two or more stations. If all these stations try to resend their frames after the time-out, the frames will collide again.
- 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. This time is called as the back off time T_B.
- Pure ALOHA has a second method to prevent congesting the channel with retransmitted frames. After a maximum number of re transmission attempts K_{max}, a station must give up and try later. Figure 1 shows the procedure for pure ALOHA based on the above strategy.

- 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 ($2 \times T_p$).
- The backoff time T_B is a random value that normally depends on K (the number of attempted unsuccessful transmissions).
- In this method, for each retransmission, a multiplier R = 0 to 2^K is randomly chosen and multiplied by T_p (maximum propagation time) or Tfr (the average time required to send out a frame) to find T_B.

Note: The range of the random numbers increases after each collision. The value of K_{max} is usually chosen as 15.

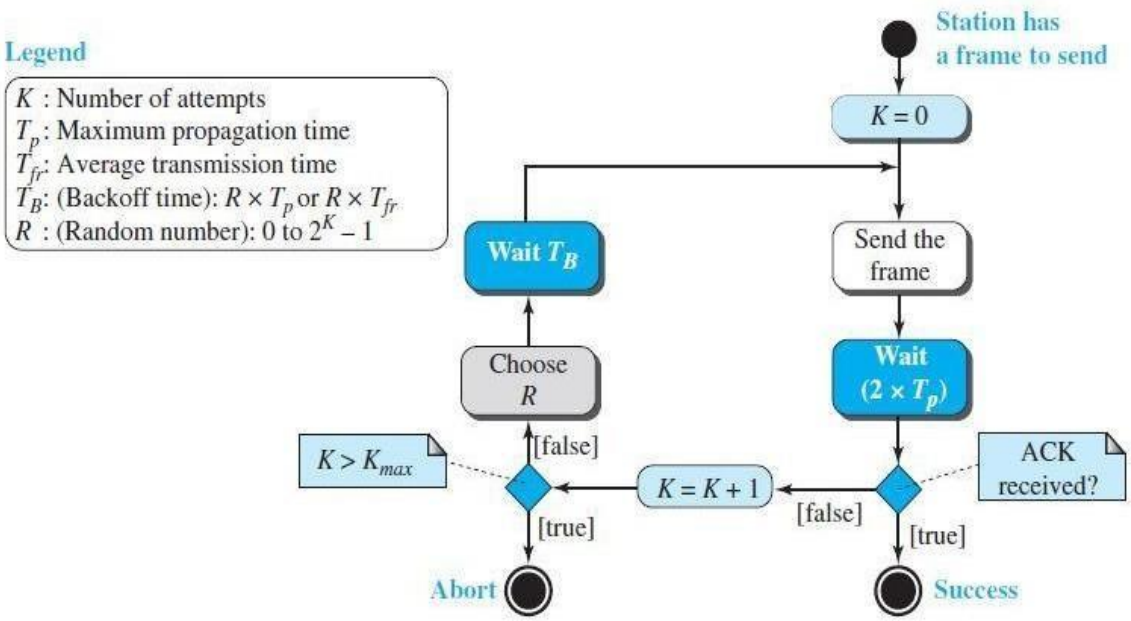


Figure 3: Procedure for pure ALOHA protocol

PROBLEM 1

The stations on a wireless ALOHA network are a maximum of 600 km apart. If we assume that signals propagate at 3×10^8 m/s, find Tp . Assume K = 2, Find the range of R.

Solution: Tp = $(600 \times 10^3) / (3 \times 10^8)$ = 2 ms.

The range of R is, R= (0 to 2^k) = {0, 1, 2, 3}.

This means that T_B can be 0, 2, 4, or 6 ms, based on the outcome of the random variable R.

Vulnerable time

The length of time in which there is a possibility of collision. The stations send fixed-length frames with each frame taking T_{fr} seconds to send. Figure 4 shows the vulnerable time for station B.

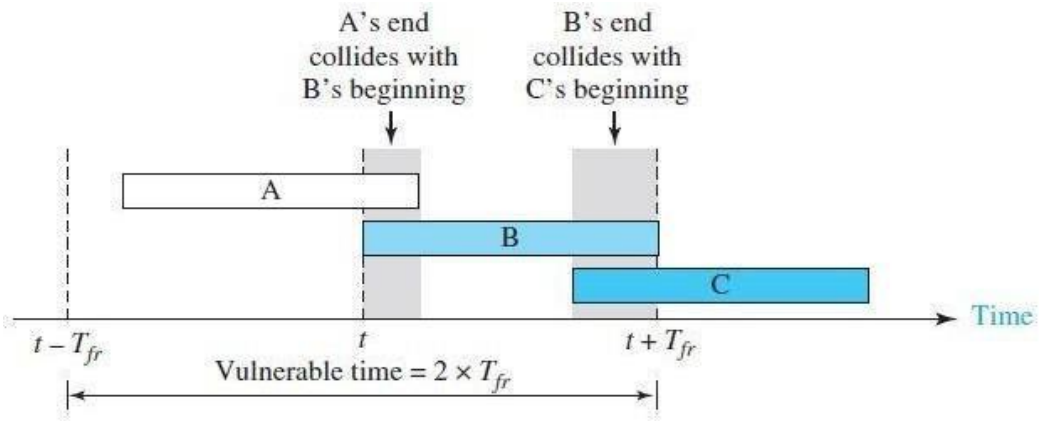


Figure 4: Vulnerable time for pure ALOHA protocol

Station B starts to send a frame at time t. Imagine station A has started to send its frame after t − T_{fr} . This leads to a collision between the frames from station B and station A. On the other hand, suppose that station C starts to send a frame before time t + T_{fr}. There is also a collision between frames from station B and station C.

From the Figure 4, it can be seen that the vulnerable time during which a collision may occur in pure ALOHA is 2 times the frame transmission time.

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

PROBLEM 2

A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the requirement to make this frame collision-free?

Solution: Average frame transmission time T_{fr} is 200 bits/200 kbps or 1 ms.

The vulnerable time is 2 × 1 ms = 2 ms.

This means no station should send later than 1 ms before this station starts transmission and no station should start sending during the period (1 ms) that this station is sending.

Throughput

G = the average number of frames generated by the system during one frame transmission time (T_{fr})

S= the average number of successfully transmitted frames for pure ALOHA.

And is given by, $S = G \times e^{-2G}$. ----------(1)

Differentiate equation (1) with respect to G and equate it to 0, we get G = 1/2.

Substitute G=1/2 in equation (1) to get Smax.

The maximum throughput Smax = 0.184.

- If one-half a frame is generated during one frame transmission time (one frame during two frame transmission times), then 18.4 percent of these frames reach their destination successfully.
- G is set to G = 1/2 to produce the maximum throughput because the vulnerable time is 2 times the frame transmission time. Therefore, if a station generates only one frame in this vulnerable time (and no other stations generate a frame during this time), the frame will reach its destination successfully.

NOTE: The throughput for pure ALOHA is $S = G \times e^{-2G}$.

The maximum throughput Smax = 1/(2e) = 0.184 when G = (1/2).

PROBLEM 3

A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces a. 1000 frames per second?

b. 500 frames per second?

c. 250 frames per second?

Solution: The frame transmission time T_{fr} is 200/200 kbps or 1ms.

(a) If the system creates 1000 frames per second, or 1 frame per millisecond (1s = 1000ms) then G = 1 (because G= number of frames generated for one T_{fr}).

$S = G \times e^{-2G} = 0.135$ (13.5 percent). This means that the throughput is 1000 ×0.135 = 135 frames. Only 135 frames out of 1000 will probably survive.

(b) If the system creates 500 frames per second, or 1/2 frame per millisecond (1s = 1000ms) then G = 1/2 (because G= number of frames generated for one T_{fr}).

$S = G \times e^{-2G} = 0.184$ (18.4 percent). This means that the throughput is 500 ×0.184 = 92 frames. Only 92 frames out of 500 will probably survive.

(c) If the system creates 250 frames per second, or 1/4 frame per millisecond (1s = 1000ms) then G = 1/4 (because G= number of frames generated for one T_{fr}).

$S = G \times e^{-2G} = 0.152$ (15.2 percent). This means that the throughput is

250 ×0.152 = 135 frames. Only 38 frames out of 250 will probably survive.

Slotted ALOHA

- Pure ALOHA has a vulnerable time of 2 × Tfr. 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 just 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 Tfr seconds and force the station to send only at the beginning of the time slot. Figure 5 shows an example of frame collisions in slotted ALOHA.

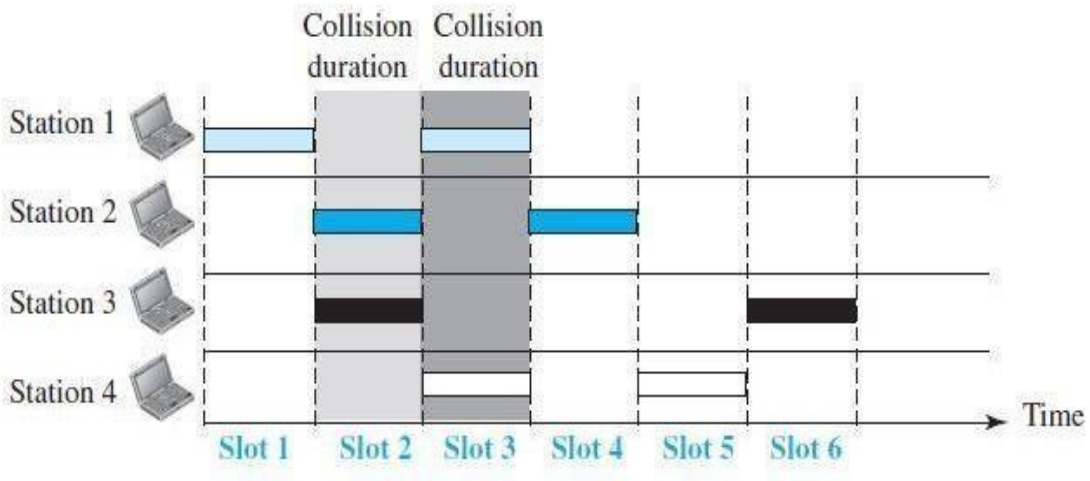


Figure 5: Frames in Slotted ALOHA network

- 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.
- There is still the possibility of collision if two stations try to send at the beginning of the same time slot. However, the vulnerable time is now reduced to one-half, equal to Tfr. Figure 6 shows the situation.

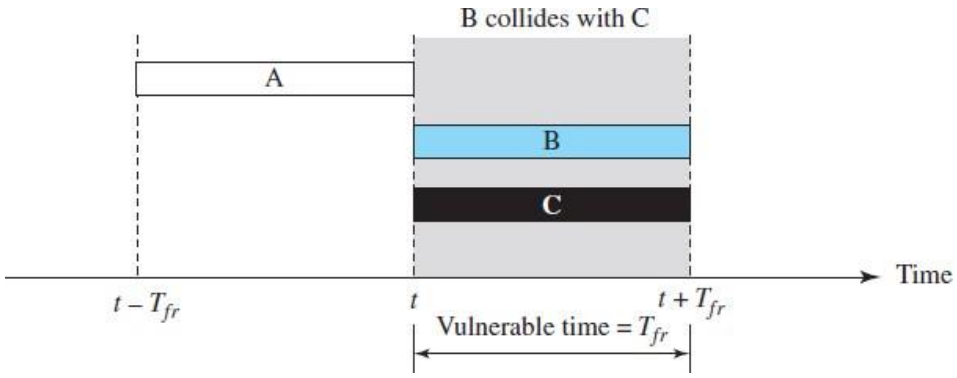


Figure 6: Vulnerable time for slotted ALOHA protocol

Slotted ALOHA vulnerable time = T_{fr}

Throughput

G = the average number of frames generated by the system during one frame transmission time (T_{fr})

S= the average number of successfully transmitted frames for Slotted ALOHA.

And is given by, $S = G \times e^{-G}$. ----------(1)

Differentiate equation (1) with respect to G and equate it to 0, we get G = 1. Substitute G=1 in equation (1) to get Smax.

The maximum throughput Smax = 0.368.

- If one frame is generated during one frame transmission time then 36.8 percent of these frames reach their destination successfully.
- G is set to G = 1 to produce the maximum throughput because the vulnerable time is equal to the frame transmission time. Therefore, if a station generates only one frame in this vulnerable time (and no other stations generate a frame during this time), the frame will reach its destination successfully.

NOTE: The throughput for Slotted ALOHA is $S = G \times e^{-G}$.

The maximum throughput Smax = 1/(e) = 0.368 when G = 1.

PROBLEM 3

A Slotted ALOHA network transmits 200-bit frames on a shared channel of 200 kbps.

What is the throughput if the system (all stations together) produces a. 1000 frames per second?

b. 500 frames per second?

c. 250 frames per second?

Solution: The frame transmission time T_{fr} is 200/200 kbps or 1ms.

(a) If the system creates 1000 frames per second, or 1 frame per millisecond (1s = 1000ms) then G = 1 (because G= number of frames generated for one T_{fr}).

$S = G \times e^{-G} = 0.368$ (36.8 percent). This means that the throughput is

1000 ×0.368 = 368 frames. Only 368 frames out of 1000 will probably survive.

(b) If the system creates 500 frames per second, or 1/2 frame per millisecond (1s = 1000ms) then G = 1/2 (because G= number of frames generated for one T_{fr}).

$S = G \times e^{-G} = 0.303$ (30.3 percent). This means that the throughputis

500 ×0.303 = 151 frames. Only 151 frames out of 500 will probably survive.

(c) If the system creates 250 frames per second, or 1/4 frame per millisecond (1s = 1000ms) then G = 1/4 (because G= number of frames generated for one T_{fr}).

$S = G \times e^{-G} = 0.195$ (19.5 percent). This means that the throughput is

250 ×0.195 = 49 frames. Only 49 frames out of 250 will probably survive.

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.
- CSMAis based on the principle “sense before transmit” or “listen before talk.”
- CSMA can reduce the possibility of collision, but it cannot eliminate it. The reason for this is shown in Figure 7, a space and time model of a CSMA network. Stations are connected to a shared channel.
- The possibility of collision still exists because of propagation delay; when a station sends a frame, it still takes time (although very short) for the first bit to reach every station and for every station to sense it.
- A station may sense the medium and find it idle, only because the first bit sent by another station has not yet been received.

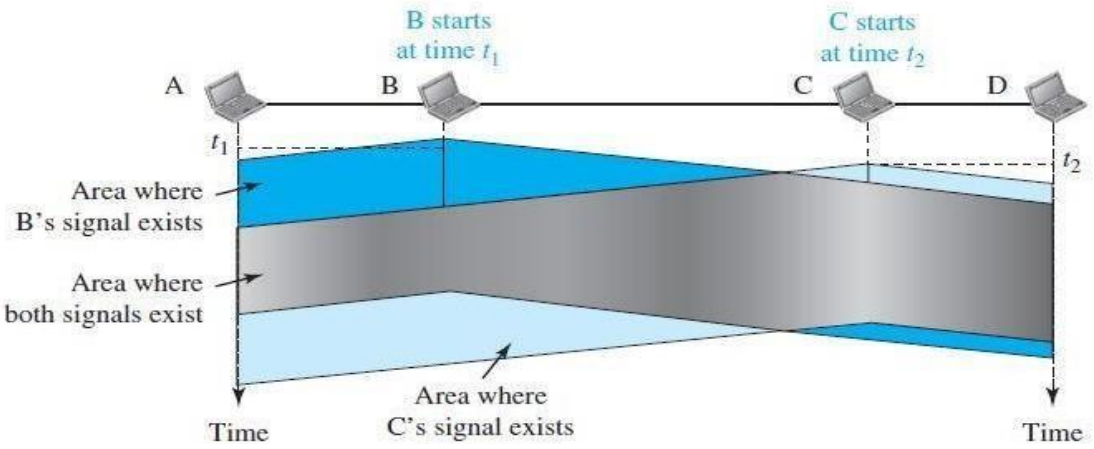


Figure 7: Space/time model of a collision in CSMA

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

Vulnerable Time

- 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.
- Figure 8 below shows the worst case. The leftmost station, A, sends a frame at time t1, which reaches the rightmost station, D, at time t1 + Tp. The gray area shows the vulnerable area in time and space.

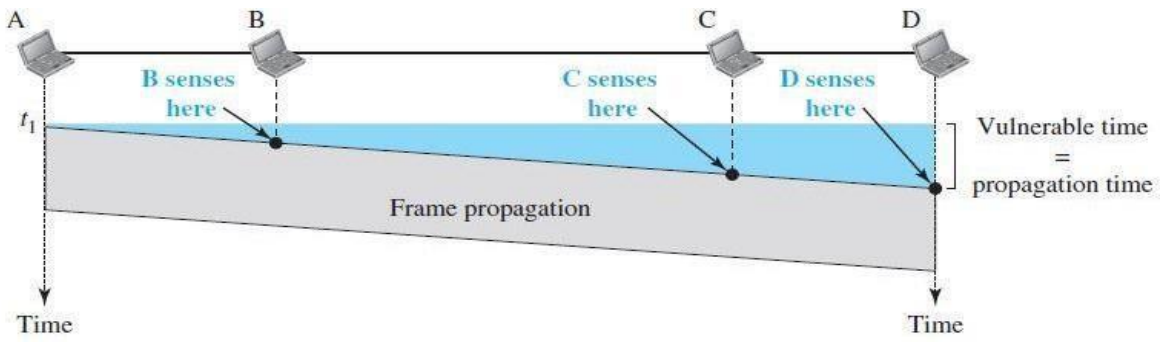


Figure 8: Vulnerable time in CSMA

Persistence Methods

Persistence method is developed to determine what the station has to do whenever it encounters the channel is idle or busy. There are 3 persistent methods

1. 1-persistent method
2. Non persistent method, and
3. p-Persistent method.

Figure 9 shows the behaviour of three persistence methods when a station finds a channel busy.

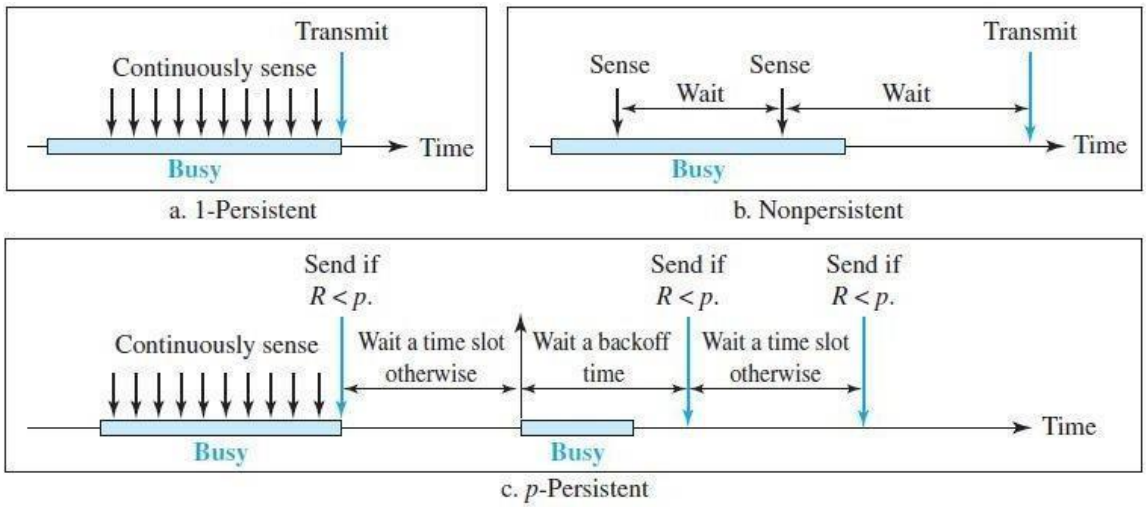


Figure 9: Behaviour of three persistence methods

1-Persistent

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

Non persistent

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

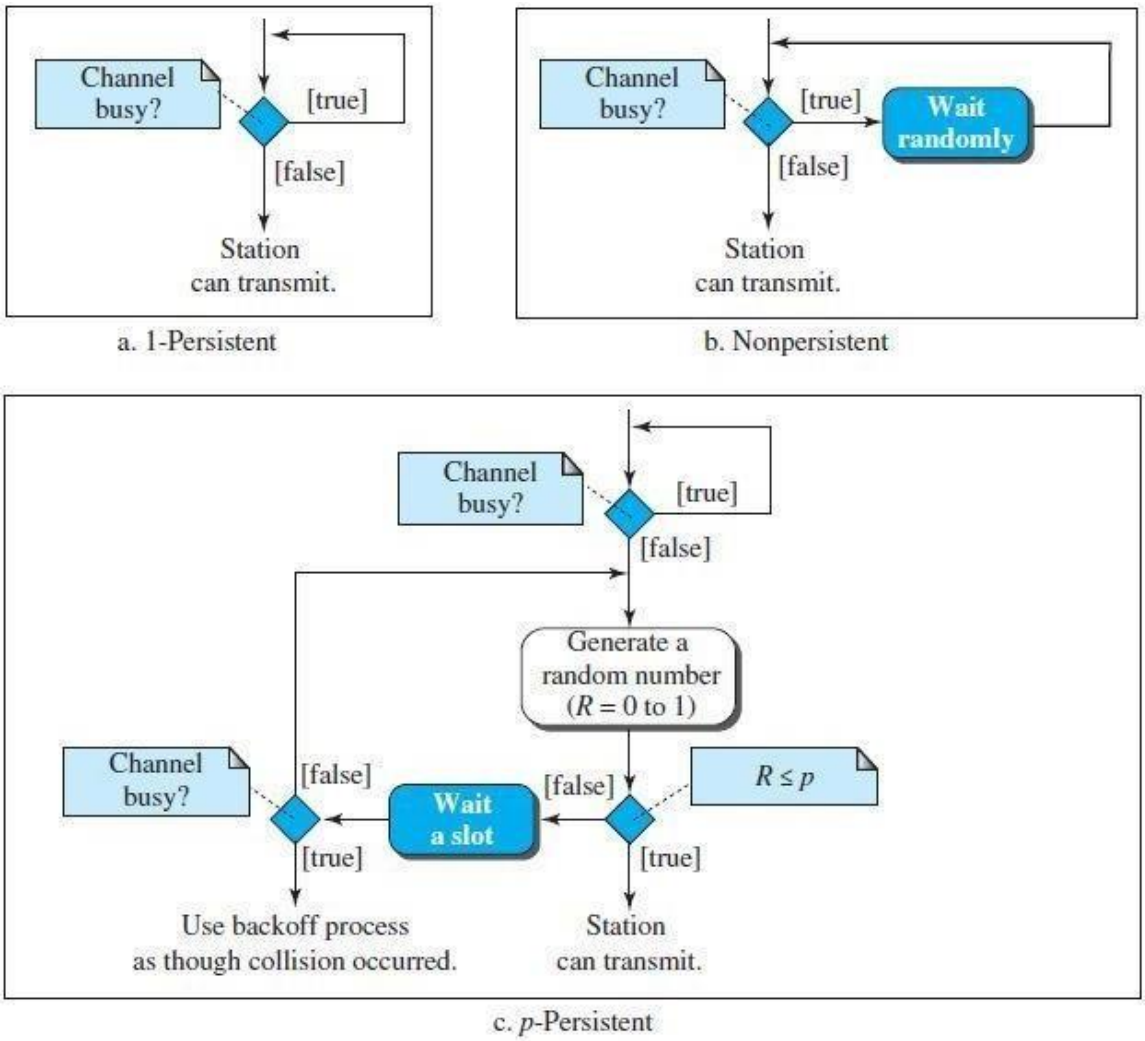


Figure 10: Flow diagram for three persistence methods

p-Persistent

- The p-persistent method is used if the channel has time slots with a slot duration equal to or greater than the maximum propagation time.
- The p-persistent approach combines the advantages of the other two strategies. It reduces the chance of collision and improves efficiency. In this method, after the station finds the line idle it follows these steps:

1. With probability p, the station sends its frame.
2. With probability q = 1 − p, the station waits for the beginning of the next time slot and checks the line again.

(a) If the line is idle, it goes to step 1.

(b) If the line is busy, it acts as though a collision has occurred and uses the backoff procedure.

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.
- Station monitors the medium after it sends a frame to see if the transmission was successful.
- 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 In Figure 11, stations A and C are involved in the collision.

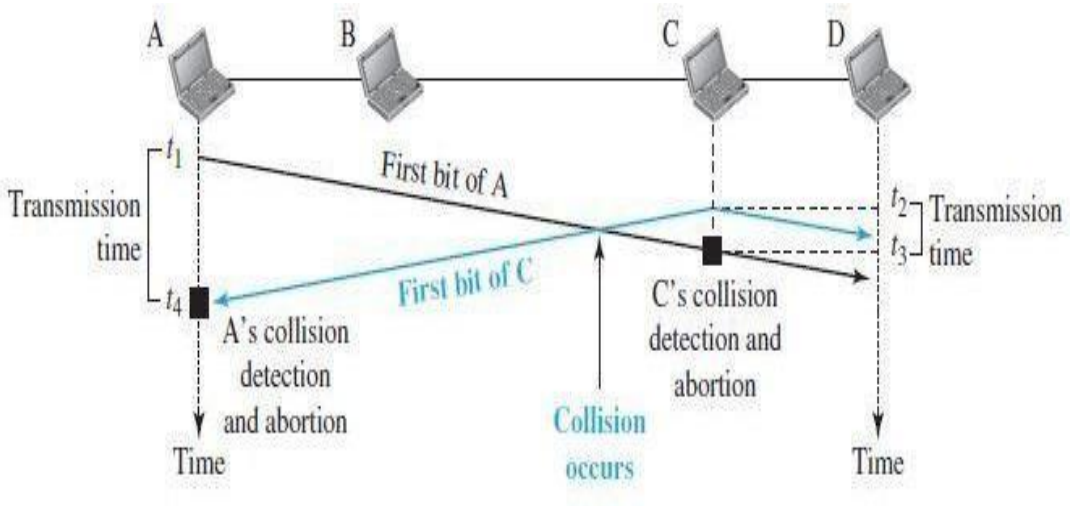


Figure 11: Collision of the first bits in CSMA/CD

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

From the Figure 11, A transmits for the duration t4 -t1, C transmits for the duration t3 -t2.

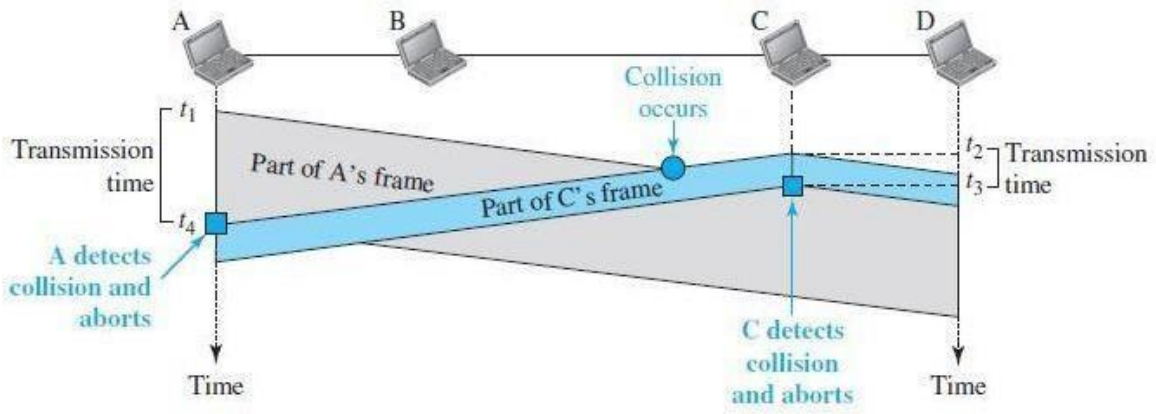


Figure 12: Collision and abortion in CSMA/CD

Minimum Frame Size

- Before sending the last bit of the frame, the sending station must detect a collision, if any, and abort the transmission.
- Once the entire frame is sent, station does not keep a copy of the frame and does not monitor the line for collision detection. Therefore, the frame transmission time Tfr must be at least two times the maximum propagation time Tp.
- If the two stations involved in a collision are the maximum distance apart, the signal from the first takes time Tp to reach the second, and the effect of the collision takes another time Tp to reach the first. So the requirement is that the first station must still be transmitting after 2Tp.

PROBLEM:

A network using CSMA/CD has a bandwidth of 10 Mbps. If the maximum propagation time (including the delays in the devices and ignoring the time needed to send a jamming signal) is 25.6 μs, what is the minimum size of the frame?

Solution:

The minimum frame transmission time is Tfr = 2 × Tp = 51.2 μs. This means, in the worst case, a station needs to transmit for a period of 51.2 μs to detect the collision.

The minimum size of the frame is, Band width × Tfr = 10 Mbps × 51.2 μs = 512 bits or 64 bytes. This is actually the minimum size of the frame for Standard Ethernet.

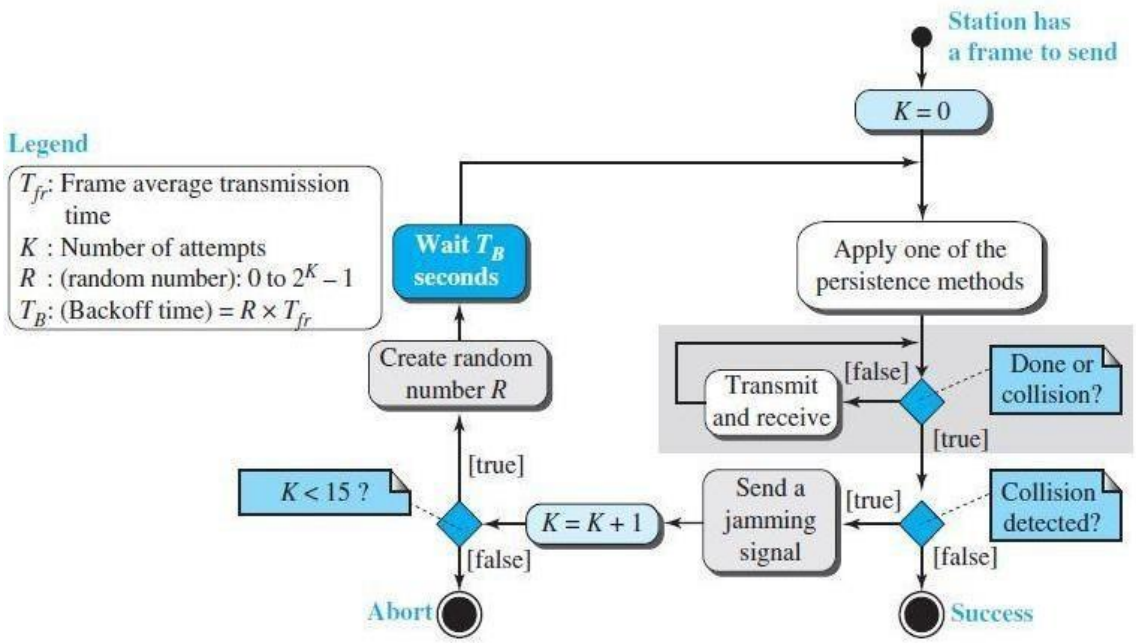


Figure 13: Flow diagram for the CSMA/CD

The flow diagram for CSMA/CD is as shown in Figure 13. It is similar to the one for the ALOHA protocol, but there are differences.

1. The first difference is the addition of the persistence process. It is required to sense the channel before sending the frame by using one of the persistence processes (non persistent, 1 persistent, or p-persistent).
2. The second difference is the frame transmission. In ALOHA, there is transmission of the entire frame and then wait for an acknowledgment. In CSMA/CD, transmission and collision detection are continuous processes.
    - It is not like the entire frame is sent and then look for a collision. The station transmits and receives continuously and simultaneously (using two different ports or a bidirectional port).
    - Loop is used to show that transmission is a continuous process. It is constantly monitored in order to detect one of two conditions: either transmission is finished or a collision is detected.
    - Either event stops transmission. When it comes 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.
3. The third difference is the sending of a short **jamming signal** to make sure that all other stations become aware of the collision.

Energy Level

The level of energy in a channel can have three values:

1) Zero level : The channel is idle
2) Normal level: A station has successfully captured the channel and is sending its frame.
3) Abnormal level: There is a collision and the level of the energy is twice the normal level.

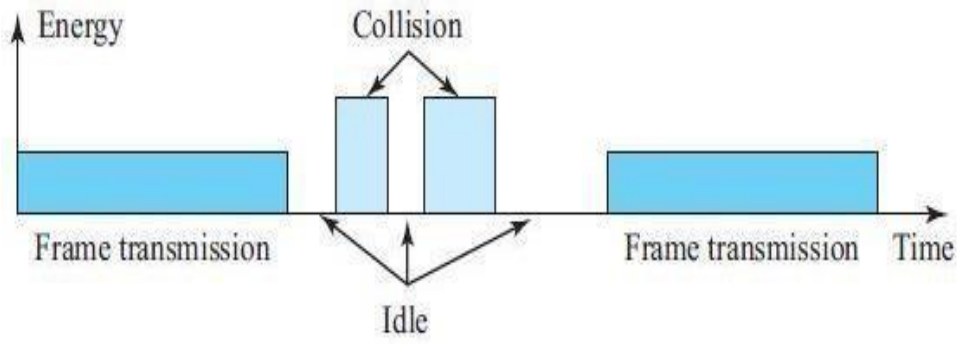


Figure 14: Energy level during transmission, idleness, or collision

NOTE: A station that has a frame to send or is sending a frame needs to monitor the energy level to determine if the channel is idle, busy, or in collision mode.

Throughput

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

CSMA/CA

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

Inter frame Space (IFS):

- 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 distant station's signal has not yet reached this station. The IFS time allows the front of the transmitted signal by the distant station to reach this station.
- After waiting an IFS time, if the channel is still idle, the station can send, but it still needs to wait a time equal to the contention window.
- The IFS variable can also be used to prioritize stations or frame types. For example, a station that is assigned a shorter IFS has a higher priority.

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. This means that it is set to one slot the first time and then doubles each time the station cannot detect an idle channel after the IFS time.
- This is very similar to the p-persistent method except that a random outcome defines the number of slots taken by the waiting station.
- One interesting point about the contention window is that the station needs to sense the channel after each time slot. However, if the station finds the channel busy, it does not restart the process; it just stops the timer and restarts it when the channel is sensed as idle. This gives priority to the station with the longest waiting time.

Refer Figure 16 of Contention window

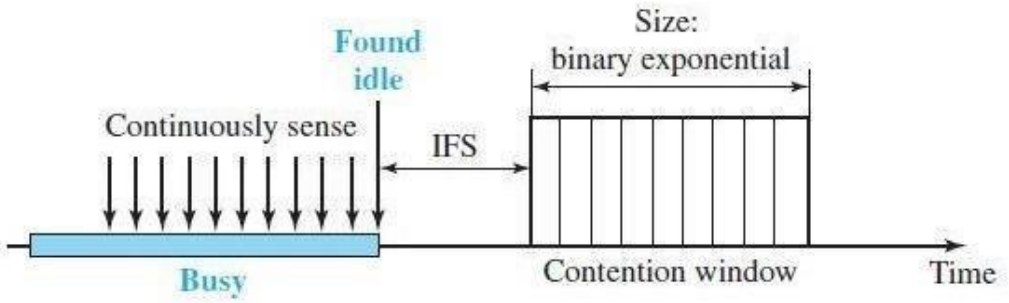


Figure 16: Contention window

Acknowledgement

Even with all the precautions considered, 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.

Frame Exchange Time Line

Figure 17 shows the exchange of data and control frames in time.

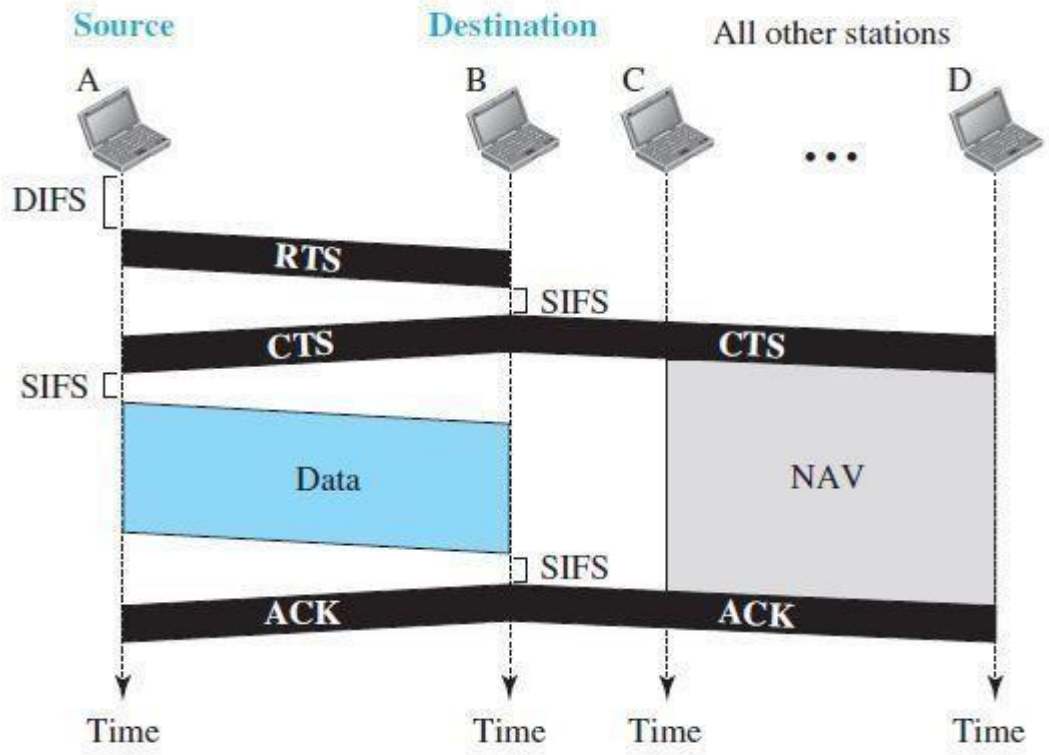


Figure 17: CSMA/CA and NAV

1.Before sending a frame, the source station senses the medium by checking the energy level at the carrier frequency.

a. The channel uses a persistence strategy with back off until the channel is idle.

b. After the station is found to be idle, the station waits for a period of time called the DCF inter frame space (DIFS), then the station sends a control frame called the request to send (RTS).

2.After receiving the RTS and waiting a period of time called the short inter frame space (SIFS), the destination station sends a control frame, called the clear to send (CTS), to the source station. This control frame indicates that the destination station is ready to receive data.

3. The source station sends data after waiting an amount of time equal to SIFS.

4. The destination station, after waiting an amount of time equal to SIFS, sends an acknowledgment to show that the frame has been received. Acknowledgment is needed in this protocol because the station does not have any means to check for the successful arrival of its data at the destination. On the other hand, the lack of collision in CSMA/CD is a kind of indication to the source that data have arrived.

NOTE: DIFS=DCF Inter frame space or Distributed Coordination Function Inter frame Space time.

Network Allocation Vector

- When a station sends an RTS frame, it includes the duration of time that it needs to occupy the channel.
- The stations that are affected by this transmission create a timer called a **Network Allocation Vector (NAV)** that shows how much time must pass before these stations are allowed to check the channel for idleness.
- Each time a station accesses the system and sends an RTS frame, other stations start their NAV. In other words, each station, before sensing the physical medium to see if it is idle, first checks its NAV to see if it has expired. Figure 17 shows the idea of NAV.

Collision during Handshaking

- If there is a collision during the time when RTS or CTS control frames are in transition, often called the handshaking period.
- Two or more stations may try to send RTS frames at the same time. These control frames may collide. However, because there is no mechanism for collision detection, the sender assumes there has been a collision if it has not received a CTS frame from the receiver. The back off strategy is employed, and the sender tries again.

Hidden-Station Problem

- The solution to the hidden station problem is the use of the handshake frames (RTS and CTS). Figure 17 shows that the RTS message from A reaches B, but not C.

- Both A and C are within the range of B, the CTS message, which contains the duration of data transmission from B to A, reaches C.
- Station C knows that some hidden station is using the channel and refrains from transmitting until that duration is over.

CONTROLLEDACCESS

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.

There are three controlled access methods,

1. Reservation.
2. Polling.
3. Token passing.

1. Reservation

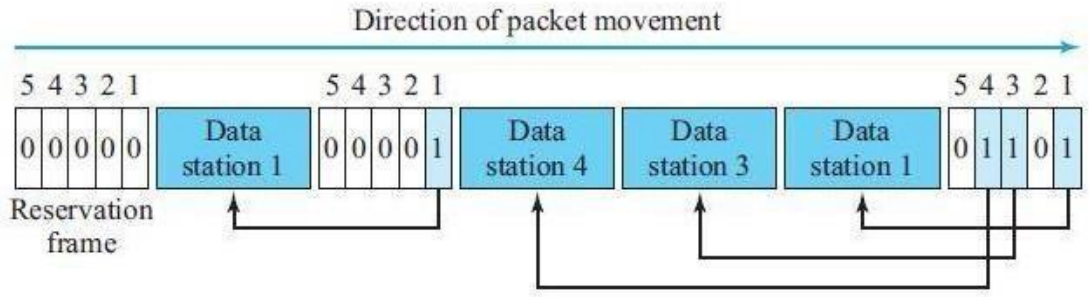


Figure 18: Reservation access method

- 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. 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 18 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.

2. 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.
- The primary device determines which device is allowed to use the channel at a given time. The primary device, therefore, is always the initiator of a session
- This method uses poll and select functions to prevent collisions. However, the drawback is if the primary station fails, the system goes down.

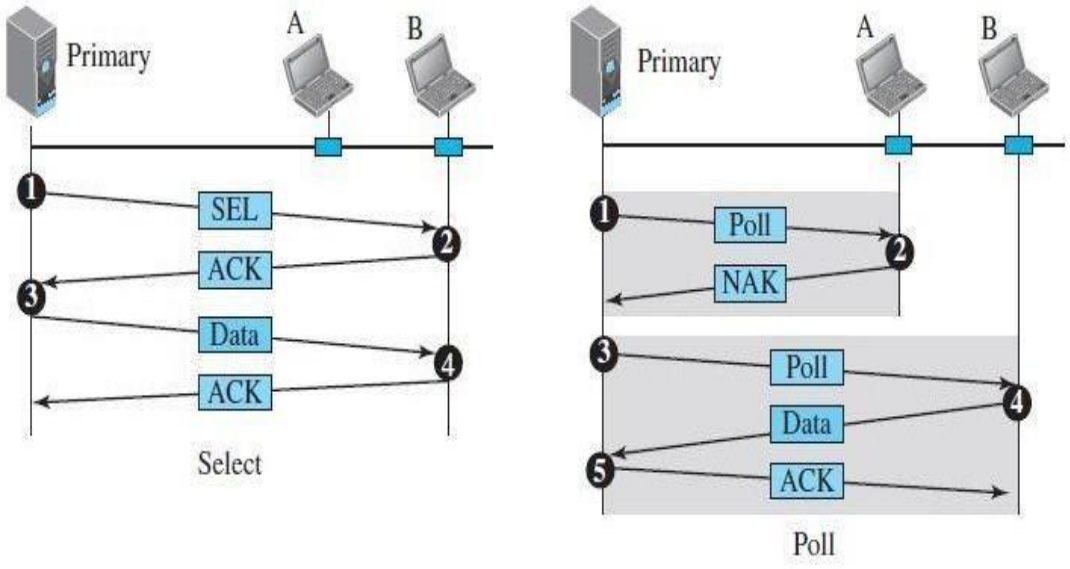


Figure 19: Select and poll functions in polling-access method

Select

- The select function is used whenever the primary device has something to send. Since the primary controls the link. If it is neither sending nor receiving data, it knows the link is available.
- If it has something to send, the primary device sends it. The primary station has to confirm whether the target device is prepared to receive.
- The primary must alert the secondary to the upcoming transmission and wait for an acknowledgment of the secondary's ready status. 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. When the first secondary is approached, it responds either with a NAK frame if it has nothing to send or with data (in the form of a data frame) if it does.
- If the response is negative (a NAK frame), then the primary polls the next secondary in the same manner until it finds one with data to send.
- When the response is positive (a data frame), the primary reads the frame and returns an acknowledgment (ACK frame), verifying its receipt.

3. Token Passing

- In the **token-passing** method, the stations in a network are organized in a logical ring. 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. The current station is the one that is accessing the channel now.
- The right to this access has been passed from the predecessor to the current station. The right will be passed to the successor when the current station has no more data to send.
- 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. When a station has some data to send, it waits until it receives the token from its predecessor. It then holds the token and sends its data.
- When the station has no more data to send, it releases the token, passing it to the next logical station in the ring. The station cannot send data until it receives the token again in the next round. In this process, when a station receives the token and has no data to send, it just passes the data to the next station.
- Token management is needed for this access method. Stations must be limited in the time they can have possession of the token. The token must be monitored to ensure it has not been lost or destroyed. For example, if a station that is holding the token fails, the token will disappear from the network.

- Another function of token management is to assign priorities to the stations and to the types of data being transmitted. And finally, token management is needed to make lowpriority stations release the token to high-priority stations.

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 20 shows four different physical topologies that can create a logical ring.

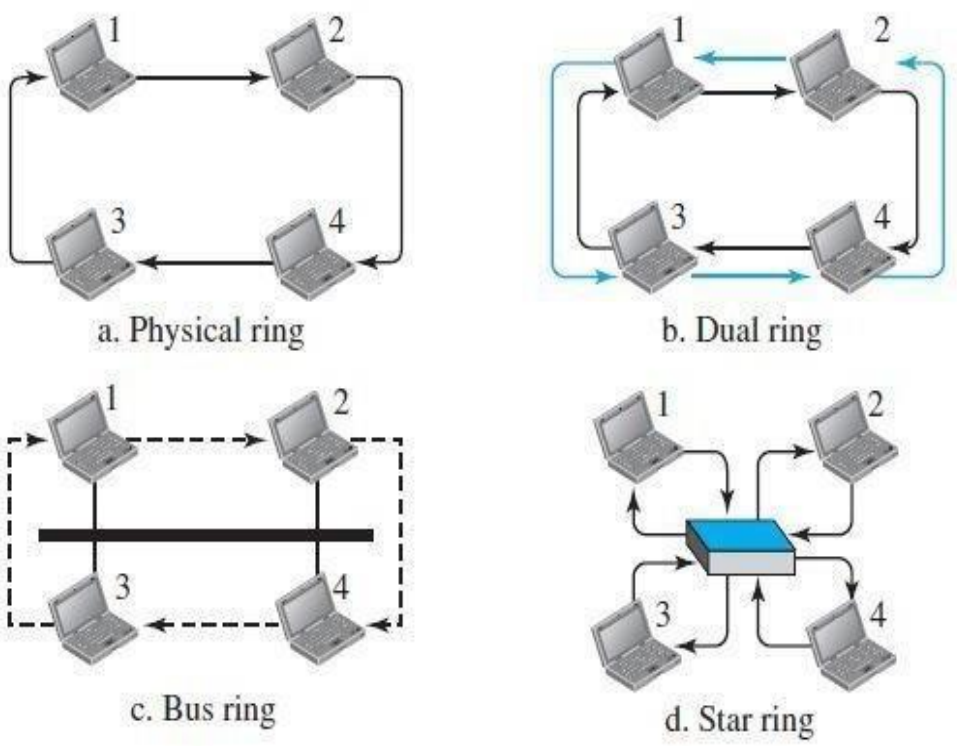


Figure 20: Logical ring and physical topology in token-passing access method

(a) Physical 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 problem with this topology is that if one of the links, the medium between two adjacent stations fails, the whole system fails.

(b) Dual ring:

- The dual ring topology uses a second (auxiliary) ring which operates in the reverse direction compared with the main ring. The second ring is for emergencies only (such as a spare tire for a car).

- If one of the links in the main ring fails, the system automatically combines the two rings to form a temporary ring. After the failed link is restored, the auxiliary ring becomes idle again.
- Each station needs to have two transmitter ports and two receiver ports. The highspeed Token Ring networks called FDDI (Fiber Distributed Data Interface) and CDDI (Copper Distributed Data Interface) use this topology.

(c) Bus Ring:

- In the bus ring topology, also called a token bus, the stations are connected to a single cable called a bus. They, make a logical ring, because each station knows the address of its successor (and also predecessor for token management purposes).

- When a station has finished sending its data, it releases the token and inserts the address of its successor in the token. Only the station with the address matching the destination address of the token gets the token to access the shared media. The Token Bus LAN, standardized by IEEE, uses this topology.

(d) Star Ring:

- In a star ring topology, the physical topology is a star. There is a hub, however, that acts as the connector.
- The wiring inside the hub makes the ring; the stations are connected to this ring through the two wire connections.
- This topology makes the network less prone to failure because if a link goes down, it will be bypassed by the hub and the rest of the stations can operate.
- Adding and removing stations from the ring is easier. This topology is still used in the Token Ring LAN designed by IBM.

Wired LANs: Ethernet

ETHERNET PROTOCOL

A local area network (LAN) is a computer network that is designed for a limited geographic area such as a building or a campus. Although a LAN can be used as an isolated network to connect computers in an organization for the sole purpose of sharing resources, most LANs today are also linked to a wide area network (WAN) or the Internet. Almost every LAN except Ethernet has disappeared from the marketplace because Ethernet was able to update itself to meet the needs of the time

IEEE Project 802

- In 1985, the Computer Society of the IEEE started a project, called ***Project 802,*** to set standards to enable intercommunication among equipment from a variety of manufacturers.
- Project 802 does not seek to replace any part of the OSI model or TCP/IP protocol suite. Instead, it is a way of specifying functions of the physical layer and the data-link layer of major LAN protocols.
- The relationship of the 802 Standard to the TCP/IP protocol suite is shown in Figure 13.1. The IEEE has subdivided the data-link layer into two sub layers:
    - Logical link control (LLC)
    - Media access control (MAC)

  IEEE has also created several physical-layer standards for different LAN protocols.

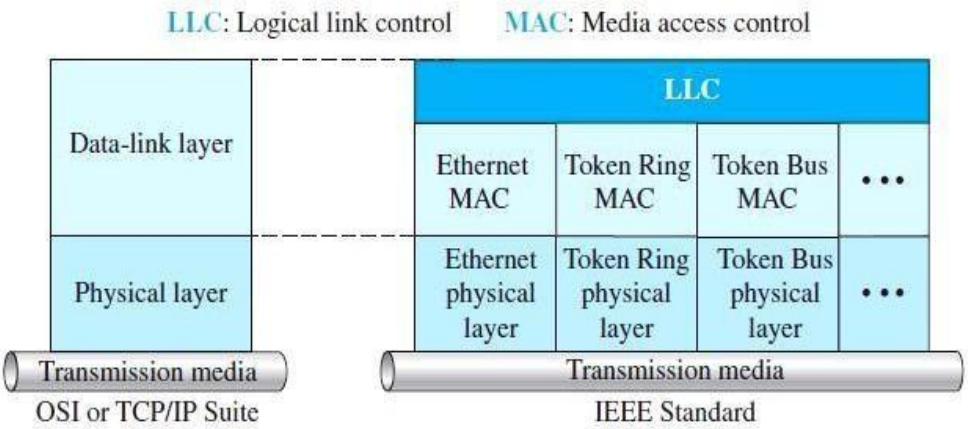


Figure 1: IEEE standard for LANs

Logical Link Control (LLC)

- In IEEE Project 802, flow control, error control, and part of the framing duties are collected into one sub layer called the logical link control (LLC). Framing is handled in both the LLC sublayer and the MAC sublayer.
- The LLC provides a single link-layer control protocol for all IEEE LANs. This means LLC protocol can provide interconnectivity between different LANs because it makes the MAC sub layer transparent.

Media Access Control (MAC)

- IEEE Project 802 has created a sublayer called media access control that defines the specific access method for each LAN. For example, it defines CSMA/CD as the media access method for Ethernet LANs and defines the token-passing method for Token Ring and Token Bus LANs.
- Part of the framing function is also handled by the MAC layer.

Ethernet Evolution

The Ethernet LAN was developed in the 1970s by Robert Metcalfe and David Boggs.

The four generations of Ethernet are :

1. Standard Ethernet (10 Mbps)
2. Fast Ethernet (100 Mbps)
3. Gigabit Ethernet (1 Gbps) and
4. 10 Gigabit Ethernet (10 Gbps)

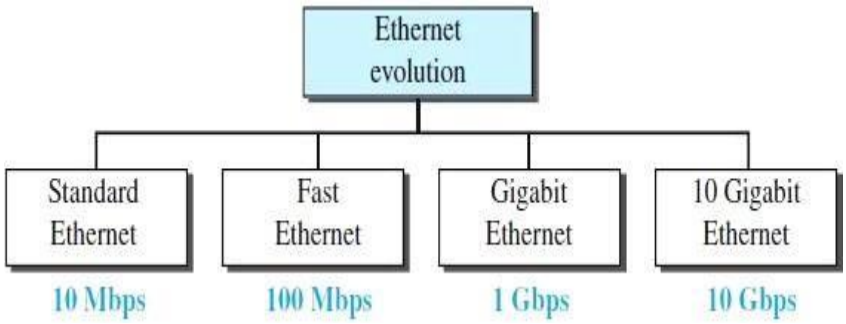


Figure 2: Ethernet evolution through four generations

STANDARD ETHERNET

Characteristics

1. Connectionless and Unreliable Service

- Ethernet provides a connectionless service, which means each frame sent is independent of the previous or next frame. Ethernet has no connection establishment or connection termination phases.
- The sender sends a frame whenever it has, the receiver may or may not be ready for it. The sender may overwhelm the receiver with frames, which may result in dropping frames. If a frame drops, the sender will not know about it. Since IP, which is using the service of Ethernet, is also connectionless, it will not know about it either. If the transport layer is also a connectionless protocol, such as UDP, the frame is lost and salvation may only come from the application layer. However, if the transport layer is TCP, the sender TCP does not receive acknowledgment for its segment and sends it again.
- Ethernet is also unreliable like IP and UDP. If a frame is corrupted during transmission and the receiver finds out about the corruption, which has a high level of probability of happening because of the CRC-32, the receiver drops the frame silently.

  It is the duty of high-level protocols to find out about it.

2. Frame Format

The Ethernet frame contains seven fields, as shown in Figure 3

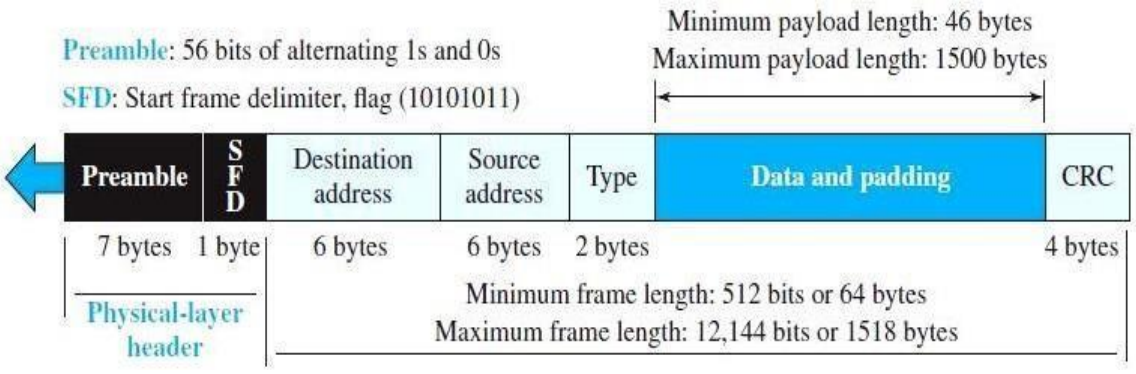


Figure 3: Ethernet frame

- **Preamble.** This field contains 7 bytes (56 bits) of alternating 0s and 1s that alert the receiving system to the coming frame and enable it to synchronize its clock if it's out of synchronization. The pattern provides only an alert and a timing pulse. The 56-bit pattern

allows the stations to miss some bits at the beginning of the frame. The preamble is actually added at the physical layer and is not part of the frame.

- **Start frame delimiter (SFD).** This field (1 byte: 10101011) signals the beginning of the frame. The SFD warns the station or stations that this is the last chance for synchronization. The last 2 bits are $(11)_2$ and alert the receiver that the next field is the destination address. This field is actually a flag that defines the beginning of the frame, an Ethernet frame is a variable-length frame. It needs a flag to define the beginning of the frame. The SFD field is also added at the physical layer.

- **Destination address (DA).** This field is six bytes (48 bits) and contains the link layer address of the destination station or stations to receive the packet. When the receiver sees its own link-layer address, or a multicast address for a group that the receiver is a member of, or a broadcast address, it decapsulates the data from the frame and passes the data to the upper layer protocol defined by the value of the type field.

- **Source address (SA).** This field is also six bytes and contains the link-layer address of the sender of the packet.

- **Type.** This field defines the upper-layer protocol whose packet is encapsulated in the frame. This protocol can be IP, ARP, OSPF, and so on. In other words, it serves the same purpose as the protocol field in a datagram and the port number in a segment or user datagram. It is used for multiplexing and demultiplexing.

- **Data.** This field carries data encapsulated from the upper-layer protocols. It is a minimum of 46 and a maximum of 1500 bytes. If the data coming from the upper layer is more than 1500 bytes, it should be fragmented and encapsulated in more than one frame. If it is less than 46 bytes, it needs to be padded with extra 0s. A padded data frame is delivered to the upper-layer protocol as it is (without removing the padding), which means that it is the responsibility of the upper layer to remove or, in the case of the sender, to add the padding. The upper-layer protocol needs to know the length of its data. For example, a datagram has a field that defines the length of the data.

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

3. Frame Length

- Ethernet has imposed restrictions on both the minimum and maximum lengths of a frame. The minimum length restriction is required for the correct operation of CSMA/CD.
- An Ethernet frame needs to have a minimum length of 512 bits or 64 bytes. Part of this length is the header and the trailer. If we count 18 bytes of header and trailer (6 bytes of source address, 6 bytes of destination address, 2 bytes of length or type, and 4 bytes of CRC), then the minimum length of data from the upper layer is 64 − 18 = 46 bytes. If the upper-layer packet is less than 46 bytes, padding is added to make up the difference.
- The standard defines the maximum length of a frame (without preamble and SFD field) as 1518 bytes. If we subtract the 18 bytes of header and trailer, the maximum length of the payload is 1500 bytes.
- The maximum length restriction has two historical reasons. First, memory was very expensive when Ethernet was designed; a maximum length restriction helped to reduce the size of the buffer. Second, the maximum length restriction prevents one station from monopolizing the shared medium, blocking other stations that have data to send.

NOTE:

Minimum frame length: 64 bytes	Minimum data length: 46 bytes
Maximum frame length: 1518 bytes	Maximum data length: 1500 bytes

Addressing

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

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

Transmission of Address Bits

The way the addresses are sent out online is different from the way they are written in hexadecimal notation. The transmission is left to right, byte by byte; however, for each byte, the least significant bit is sent first and the most significant bit is sent last. This means that the bit that defines an address as unicast or multicast arrives first at the receiver. This helps the receiver to immediately know if the packet is unicast or multicast.

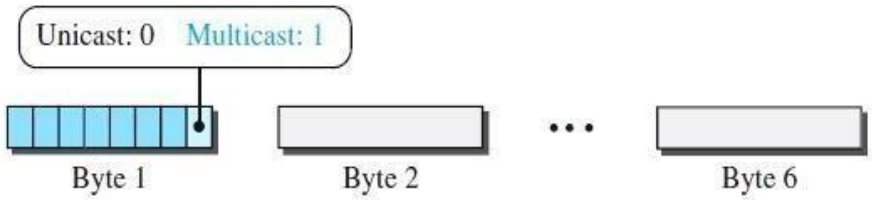
Example

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

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

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

Unicast, Multicast, and Broadcast Addresses



A source address is always a unicast address, the frame comes from only one station. The destination address, however, can be unicast, multicast, or broadcast. Figure 4 shows how to distinguish a unicast address from a multicast address. If the least significant bit of the first byte in a destination address is 0, the address is unicast; otherwise, it is multicast. With the way the bits are transmitted, the unicast/multicast bit is the first bit which is transmitted or received. The broadcast address is a special case of the multicast address: the recipients are all the stations on the LAN. A broadcast destination address is forty-eight 1s.

Example

Define the type of the following destination addresses a.

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

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

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

Solution: To find the type of the address, we need to look at the second hexadecimal digit from the left. If it is even, the address is unicast. If it is odd, the address is multicast. If all digits are Fs, the address is broadcast. Therefore, we have the following: a. This is a unicast address because A in binary is 1010 (even).

b. This is a multicast address because 7 in binary is 0111 (odd).

c. This is a broadcast address because all digits are Fs in hexadecimal.

Access Method

Since the network that uses the standard Ethernet protocol is a broadcast network, The standard Ethernet chose CSMA/CD with 1-persistent method, Let us use a scenario to see how this method works for the Ethernet protocol.

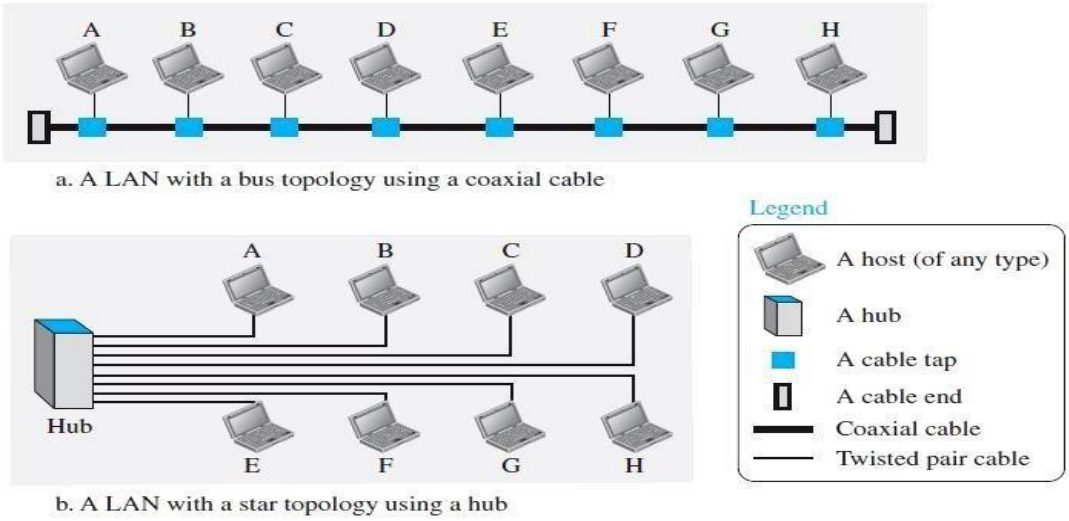


Figure 5: Implementation of standard Ethernet

- Assume station A in Figure.5 has a frame to send to station D. Station A first should check whether any other station is sending (carrier sense). Station A measures the level of energy on the medium (for a short period of time, normally less than 100μs). If there is no signal energy on the medium, it means that no station is sending (or the signal has not reached station A). Station A interprets this situation as idle medium. It starts sending its frame. On the other hand, if the signal energy level is not zero, it means that the medium is being used by another station. Station A continuously monitors the medium until it becomes idle for 100μs. It then starts sending the frame. However, station A needs to keep a copy of the frame in its buffer until it is sure that there is no collision.

- The medium sensing does not stop after station A has started sending the frame. Station A needs to send and receive continuously. Two cases may occur:
  (a) Station A has sent 512 bits and no collision is sensed (the energy level did not go above the regular energy level), the station then is sure that the frame will go through and stops sensing the medium. Where does the number 512 bits come from? If we consider the transmission rate of the Ethernet as 10 Mbps, this means that it takes the station 512/ (10 Mbps) = 51.2 μs to send out 512 bits. With the speed of propagation in a cable (2 × 108 meters), the first bit could have gone 10,240 meters (one way) or only 5120 meters (round trip), have collided with a bit from the last station on the cable, and have gone back. In other words, if a collision were to occur, it should occur by the time the sender has sent out 512 bits (worst case) and the first bit has made a round trip of 5120 meters, if the collision happens in the middle of the cable, not at the end, station A hears the collision earlier and aborts the transmission. The above assumption is that the length of the cable is 5120 meters. The designer of the standard Ethernet actually put a restriction of 2500 meters because we need to consider the delays encountered throughout the journey. It means that they considered the worst case. The whole idea is that if station A does not sense the collision before sending 512 bits, there must have been no collision, because during this time, the first bit has reached the end of the line and all other stations know that a station is sending and refrain from sending. In other words, the problem occurs when another station (for example, the last station) starts sending before the first bit of station A has reached it. The other station mistakenly thinks that the line is free because the first bit has not yet reached it. The restriction of 512 bits actually helps the sending station: The sending

     station is certain that no collision will occur if it is not heard during the first 512 bits, so it can discard the copy of the frame in its buffer.
  (b) Station A has sensed a collision before sending 512 bits. This means that one of the previous bits has collided with a bit sent by another station. In this case both stations should refrain from sending and keep the frame in their buffer for resending when the line becomes available. However, to inform other stations that there is a collision in the network, the station sends a 48-bit jam signal. The jam signal is to create enough signal (even if the collision happens after a few bits) to alert other stations about the collision. After sending the jam signal, the stations need to increment the value of K (number of attempts). If after increment $K = 15$, the experience has shown that the network is too

busy, the station needs to abort its effort and try again. If $K < 15$, the station can wait a backoff time (T_B) and restart the process. The station creates a random number between 0 and $2^K - 1$, which means each time the collision occurs, the range of the random number increases exponentially. After the first collision ($K = 1$) the random number is in the range (0, 1). After the second collision ($K = 2$) it is in the range (0, 1, 2, 3). After the third collision ($K = 3$) it is in the range (0, 1, 2, 3, 4, 5, 6, 7). So, after each collision, the probability increases that the backoff time becomes longer. This is due to the fact that if the collision happens even after the third or fourth attempt, it means that the network is really busy; a longer backoff time is needed.

Efficiency of Standard Ethernet

The efficiency of the Ethernet is defined as the ratio of the time used by a station to send data to the time the medium is occupied by this station. The practical efficiency of standard Ethernet has been measured to be,

$$\text{Efficiency} = \frac{\quad}{\quad}$$

Where, = the number of frames that can fit on the medium.

$$= \frac{\quad}{\quad}$$

The transmission delay is the time it takes a frame of average size to be sent out and the propagation delay is the time it takes to reach the end of the medium. As the value of parameter decreases, the efficiency increases. This means that if the length of the media is shorter or the frame size longer, the efficiency increases. In the ideal case, = 0 and the efficiency is 1.

Example13.3

In the Standard Ethernet with the transmission rate of 10 Mbps, we assume that the length of the medium is 2500 m and the size of the frame is 512 bits. The propagation speed of a signal in a cable is normally 2×10^8 m/s.

Propagation delay = $2500/(2 \times 10^8) = 12.5\ \mu s$ **Transmission delay = $512/(10^7) = 51.2\ \mu s$**

$a = 12.5/51.2 = 0.24$ **Efficiency = 39%**

The example shows that a = 0.24, which means only 0.24 of a frame occupies the whole medium in this case. The efficiency is 39 percent, which is considered moderate; it means that only 61 percent of the time the medium is occupied but not used by a station.

Implementation

The Standard Ethernet defined several implementations, but only four of them became popular during the 1980s. Table below shows a summary of Standard Ethernet implementations.

Implementation	*Medium*	*Medium Length*	*Encoding*
10Base5	Thick coax	500 m	Manchester
10Base2	Thin coax	185 m	Manchester
10Base-T	2 UTP	100 m	Manchester
10Base-F	2 Fiber	2000 m	Manchester

In the nomenclature 10BaseX, the number defines the data rate (10 Mbps), the term *Base* means baseband (digital) signal, and X approximately defines either the maximum size of the cable in 100 meters (for example 5 for 500 or 2 for 185 meters) or the type of cable, T for unshielded twisted pair cable (UTP) and F for fiber-optic. The standard Ethernet uses a baseband signal, which means that the bits are changed to a digital signal and directly sent on the line.

Encoding and Decoding

All standard implementations use digital signalling (baseband) at 10 Mbps. At the sender, data are converted to a digital signal using the Manchester scheme; at the receiver, the received signal is interpreted as Manchester and decoded into data. Manchester encoding is self-synchronous, providing a transition at each bit interval. Figure 6 shows the encoding scheme for Standard Ethernet.

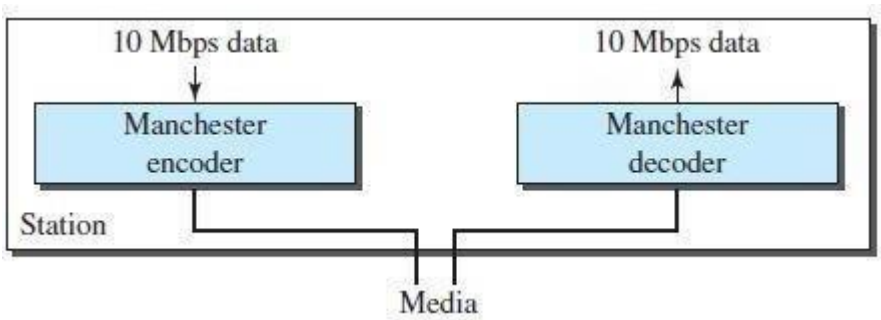


Figure 6: Encoding in a Standard Ethernet implementation

10Base5: Thick Ethernet

The first implementation is called 10Base5, thick Ethernet, or Thicknet. The nickname derives from the size of the cable, which is roughly the size of a garden hose and too stiff to bend with your hands. 10Base5 was the first Ethernet specification to use a bus topology with an external transceiver (transmitter/receiver) connected via a tap to a thick coaxial cable.

Figure 7 shows a schematic diagram of a 10Base5 implementation.

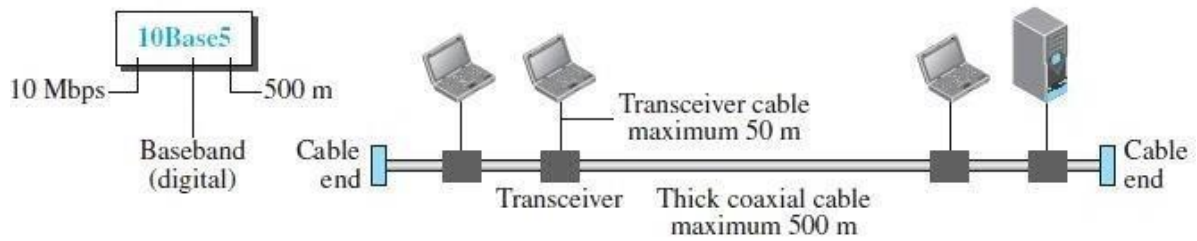


Figure 7: 10Base5 implementation

The transceiver is responsible for transmitting, receiving, and detecting collisions. The transceiver is connected to the station via a transceiver cable that provides separate paths for sending and receiving. This means that collision can only happen in the coaxial cable. The maximum length of the coaxial cable must not exceed 500 m, otherwise, there is excessive degradation of the signal. If a length of more than 500 m is needed, up to five segments, each a maximum of 500 meters, can be connected using repeaters.

10Base2: Thin Ethernet

The second implementation is called 10Base2, thin Ethernet, or Cheaper net. 10Base2 also uses a bus topology, but the cable is much thinner and more flexible. The cable can be bent to pass very close to the stations. In this case, the transceiver is normally part of the network interface card (NIC), which is installed inside the station. Figure 8 shows the schematic diagram of a 10Base2 implementation.

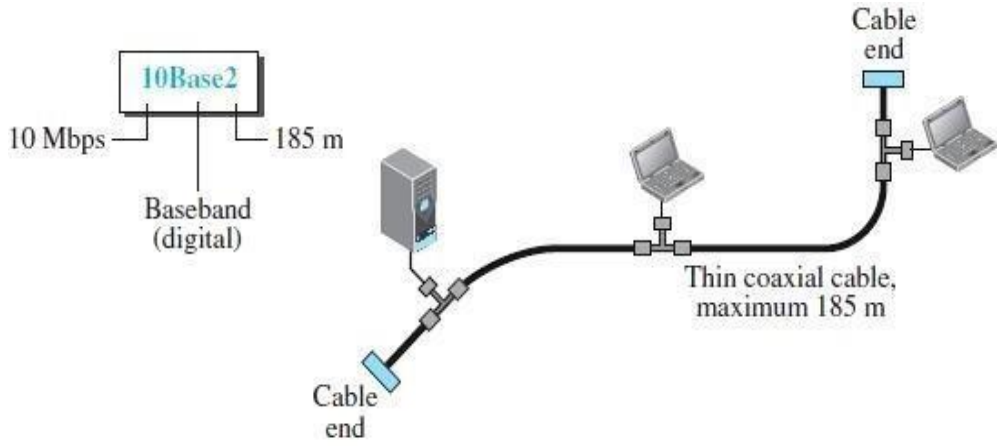


Figure 8: 10Base2 implementation

The collision here occurs in the thin coaxial cable. This implementation is more cost effective than 10Base5 because thin coaxial cable is less expensive than thick coaxial and the tee connections are much cheaper than taps. Installation is simpler because the thin coaxial cable is very flexible. However, the length of each segment cannot exceed 185 m (close to 200 m) due to the high level of attenuation in thin coaxial cable.

10Base-T: Twisted-Pair Ethernet

The third implementation is called **10Base-T** or **twisted-pair Ethernet.** 10Base-T uses a physical star topology. The stations are connected to a hub via two pairs of twisted cable, as shown in Figure 9.

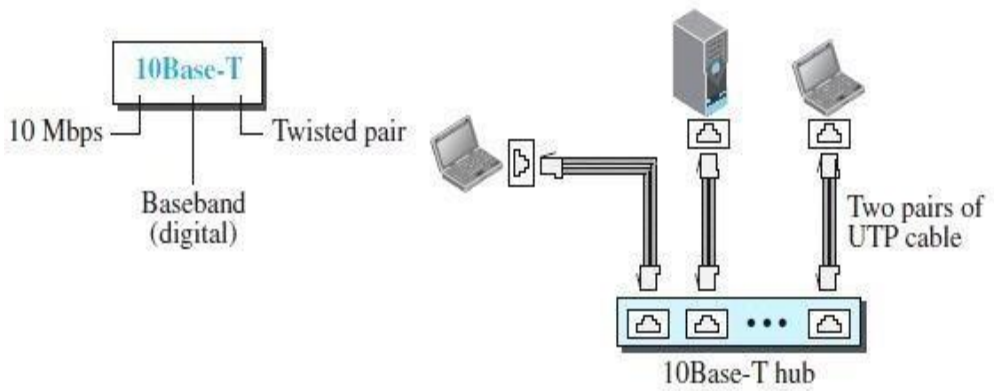


Figure 9: 10Base-T implementation

Two pairs of twisted cable create two paths (one for sending and one for receiving) between the station and the hub. Any collision here happens in the hub. Compared to 10Base5 or 10Base2, we can see that the hub actually replaces the coaxial cable as far as a collision is concerned. The maximum length of the twisted cable here is defined as 100 m, to minimize the effect of attenuation in the twisted cable.

10Base-F: Fiber Ethernet

Although there are several types of optical fiber 10-Mbps Ethernet, the most common is called **10Base-F.** 10Base-F uses a star topology to connect stations to a hub. The stations are connected to the hub using two fiber-optic cables, as shown in Figure 10.

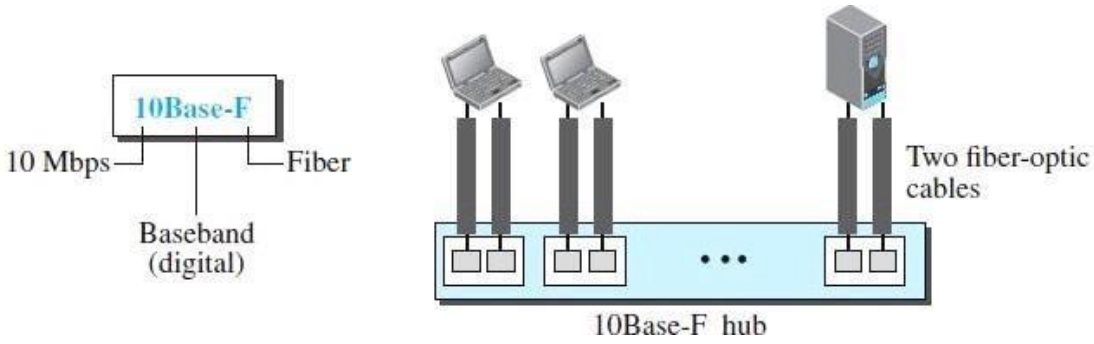


Figure 10: 10Base-F implementation

FAST ETHERNET (100MBPS)

In the 1990s, some LAN technologies with transmission rates higher than 10 Mbps, such as FDDI and Fiber Channel, appeared on the market. If the Standard Ethernet wanted to survive, it had to compete with these technologies. Ethernet made a big jump by increasing the transmission rate to 100 Mbps, and the new generation was called the Fast Ethernet. The designers of the Fast Ethernet needed to make it compatible with the Standard Ethernet. The MAC sublayer was left unchanged, which meant the frame format and the maximum and minimum size could also remain unchanged. By increasing the transmission rate, features of the Standard Ethernet that depend on the transmission rate, access method, and implementation had to be reconsidered.

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.

Access Method

The proper operation of the CSMA/CD depends on the transmission rate, the minimum size of the frame, and the maximum network length. If we want to keep the minimum size of the frame, the maximum length of the network should be changed. In other words, if the minimum frame size is still 512 bits, and it is transmitted 10 times faster, the collision needs to be detected 10 times sooner, which means the maximum length of the network should be 10 times shorter (the propagation speed does not change). So the Fast Ethernet came with two solutions (it can work with either choice):

1. The first solution was to totally drop the bus topology and use a passive hub and star topology but make the maximum size of the network 250 meters instead of 2500 meters as in the Standard Ethernet. This approach is kept for compatibility with the Standard Ethernet.
2. The second solution is to use a link-layer switch with a buffer to store frames and a full-duplex connection to each host to make the transmission medium private for each host. In this case, there is no need for CSMA/CD because the hosts are not competing with each other. The link-layer switch receives a frame from a source host and stores it in the buffer (queue) waiting for processing. It then checks the destination address and sends the frame out of the corresponding interface. Since the connection to the switch is full-duplex, the destination address can even send a frame to another station at the same time that it is receiving a frame. In other words, the shared medium is changed to many point-to- point media, and there is no need for contention.

Auto negotiation

A new feature added to Fast Ethernet is called auto negotiation. It allows a station or a hub a range of capabilities. Auto negotiation allows two devices to negotiate the mode or data rate of operation. It was designed particularly to allow incompatible devices to connect to one another.

It was designed particularly for these purposes:

- To allow incompatible devices to connect to one another. For example, a device with a maximum capacity of 10 Mbps can communicate with a device with a 100 Mbps capacity (but which can work at a lower rate).
- To allow one device to have multiple capabilities.
- To allow a station to check a hub’s capabilities.

Physical Layer Topology

Fast Ethernet is designed to connect two or more stations. 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.

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. However, it was found that one scheme would not perform equally well for all three implementations.

Therefore, three different encoding schemes were chosen.

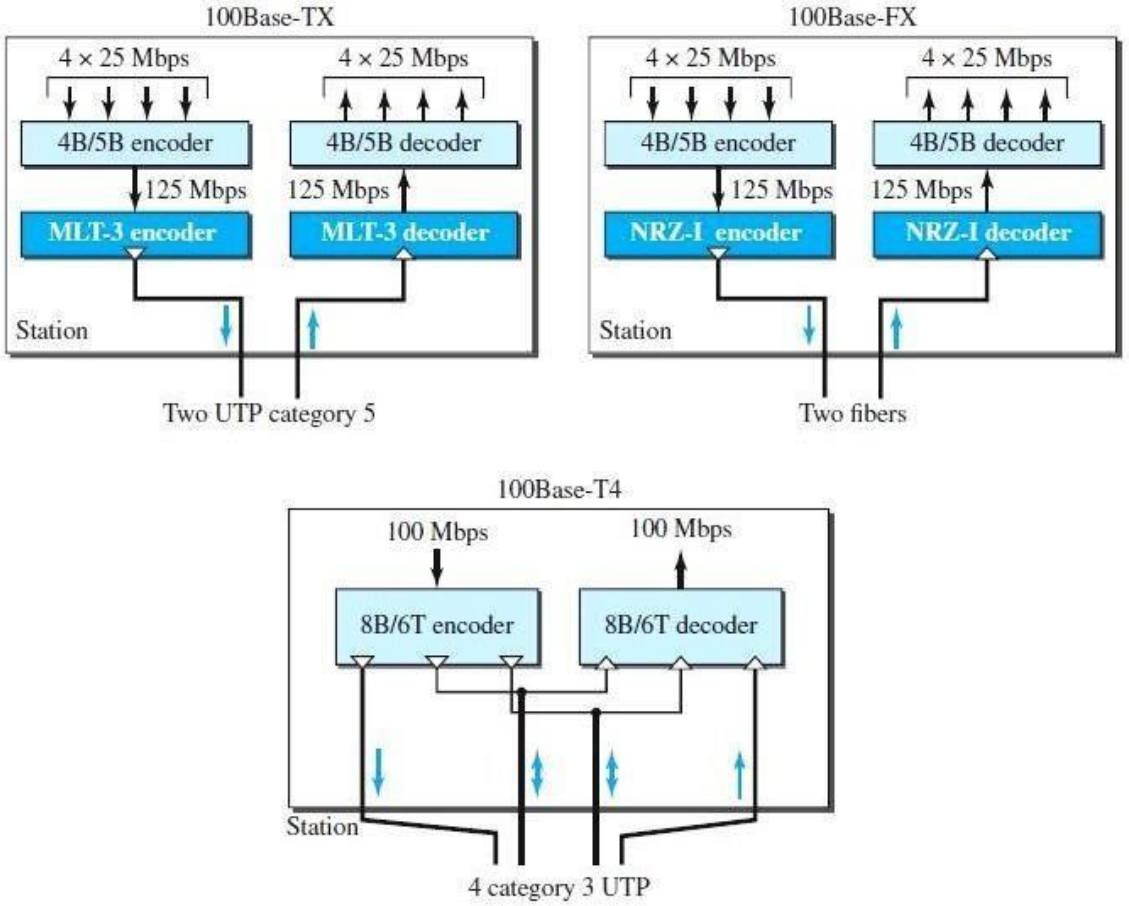


Figure 11: Encoding for fast Ethernet implementations

1. **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. 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.

2. **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. However, NRZ-I has a bit synchronization problem for long sequences of 0s (or 1s, based on the encoding). To overcome this problem, the designers used 4B/5B block encoding, as we described for 100Base-TX. 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 (category 3).

3. **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 M baud. In this design, one pair switch between sending and receiving. Three pairs of UTP category 3, however, can handle only 75 M baud (25 M baud) each. We need to use an encoding scheme that converts 100 Mbps to a 75 M baud signal. 8B/6T satisfies this requirement. In 8B/6T, eight data elements are encoded as six signal elements. This means that 100 Mbps uses only (6/8) × 100 Mbps, or 75 M baud.

Implementation	*Medium*	*Medium Length*	*Wires*	*Encoding*
100Base-TX	UTP or STP	100 m	2	4B5B + MLT-3
100Base-FX	Fiber	185 m	2	4B5B + NRZ-I
100Base-T4	UTP	100 m	4	Two 8B/6T

GIGABITETHERNET

The need for an even higher data rate resulted in the design of the Gigabit Ethernet Protocol (1000 Mbps). The IEEE committee calls it the Standard 802.3z. The goals of the Gigabit Ethernet were to upgrade the data rate to 1 Gbps, but keep the address length, the frame format, and the maximum and minimum frame length the same. 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. Support auto negotiation as defined in Fast Ethernet.

MAC Sublayer

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

Full-Duplex Mode

In full-duplex mode, there is a central switch connected to all computers or other switches. In this mode, for each input port, each switch has buffers in which data are stored until they are transmitted. Since the switch uses the destination address of the frame and sends a frame out of the port connected to that particular destination, there is no collision. This means that CSMA/CD is not used. Lack of collision implies that the maximum length of the cable is determined by the signal attenuation in the cable, not by the collision detection process.

NOTE: In the full-duplex mode of Gigabit Ethernet, there is no collision; the maximum length of the cable is determined by the signal attenuation in the cable.

Half-Duplex Mode

The half-duplex approach uses CSMA/CD. 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 1/100 shorter in Gigabit Ethernet than in 10-Mbps Ethernet, the slot time for Gigabit Ethernet is 512 bits × 1/1000 μs, which is equal to 0.512 μs. The reduced slot time means that collision is detected 100 times earlier. This means that the maximum length of the network is 25 m. This length may be suitable if all the stations are in one room, but it may not even be long enough to connect the computers in one single office.

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). This means that the minimum length is 8 times longer. This method forces a station to add extension bits (padding) to any frame that is less than 4096 bits. In this way, the maximum length of the network can be increased 8 times to a length of 200 m. This allows a length of 100m from the hub to the station.

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. However, to make these multiple frames look like one frame, padding is added between the frames (the same as that used for the carrier extension method) so that the channel is not idle. In other words, the method deceives other stations into thinking that a very large frame has been transmitted.

Physical Layer

The physical layer in Gigabit Ethernet is more complicated than that in Standard or Fast Ethernet.

Topology

Gigabit Ethernet is designed to connect two or more stations. 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.

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 (**1000Base-T**).

Implementation	*Medium*	*Medium Length*	*Wires*	*Encoding*
1000Base-SX	Fiber S-W	550 m	2	8B/10B + NRZ
1000Base-LX	Fiber L-W	5000 m	2	8B/10B + NRZ
1000Base-CX	STP	25 m	2	8B/10B + NRZ
1000Base-T4	UTP	100 m	4	4D-PAM5

Encoding

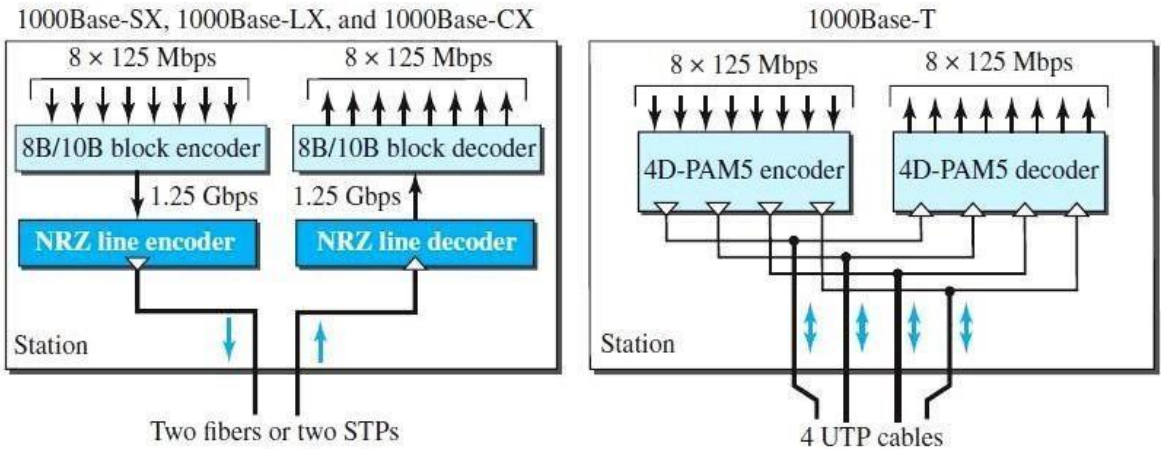


Figure 12: Encoding in Gigabit Ethernet implementations Figure 12 shows the encoding/decoding schemes for the four implementations.

- 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. To synchronize bits, particularly at this high data rate, 8B/10B block encoding, is used. This block encoding prevents long sequences of 0s or 1s in the stream, but the resulting stream is 1.25 Gbps. In this implementation, one wire (fiber or STP) is used for sending and one for receiving.
- 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.

10 GIGABIT ETHERNET

- The IEEE committee created 10 Gigabit Ethernet and called it Standard 802.3ae.
- The goals of the 10 Gigabit Ethernet design can be summarized as upgrading the data rate to 10 Gbps, keeping the same frame size and format, and allowing the interconnection of LANs, MANs, and WAN possible.
- This data rate is possible only with fiber-optic technology at this time. The standard defines two types of physical layers: LAN PHY and WAN PHY. The first is designed to support existing LANs; the second actually defines a WAN with links connected through SONET OC-192.

Implementation

10 Gigabit Ethernet operates only in full-duplex mode, which means there is no need for contention; CSMA/CD is not used in 10 Gigabit Ethernet.

Four implementations are the most common:

1. 10GBase-SR
2. 10GBase-LR
3. 10GBase-EW and
4. 10GBase-X4.

Implementation	*Medium*	*Medium Length*	*Number of wires*	*Encoding*
10GBase-SR	Fiber 850 nm	300 m	2	64B66B
10GBase-LR	Fiber 1310 nm	10 Km	2	64B66B
10GBase-EW	Fiber 1350 nm	40 Km	2	SONET
10GBase-X4	Fiber 1310 nm	300 m to 10 Km	2	8B10B

Table: Summary of 10 Gigabit Ethernet implementations

Wireless LANs

INTRODUCTION

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.,

Architectural Comparison

1. Medium

In a wireless LAN, the medium is air, the signal is generally broadcast. When hosts in a wireless LAN communicate with each other, they are sharing the same medium (multiple access). In a very rare situation, we may be able to create a point-to-point communication between two wireless hosts by using a very limited bandwidth and two-directional antennas.

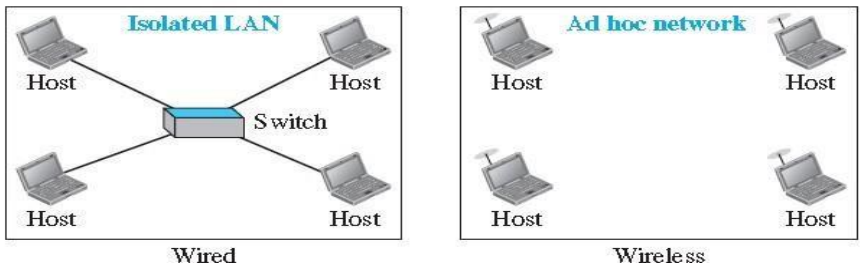
2.Hosts

In a wired LAN, a host is always connected to its network at a point with a fixed link layer address related to its network interface card (NIC). Of course, a host can move from one point in the Internet to another point. In this case, its link-layer address remains the same, but its network-layer address will change. However, before the host can use the services of the Internet, it needs to be physically connected to the Internet. In a wireless LAN, a host is not physically connected to the network; it can move freely and can use the services provided by the network

3.IsolatedLANs

A wired isolated LAN is a set of hosts connected via a link-layer switch. A wireless isolated LAN, called an ***ad hoc network in wireless*** LAN terminology, is a set of hosts that communicate freely with each other. The concept of a link-layer switch does not exist in wireless LANs.

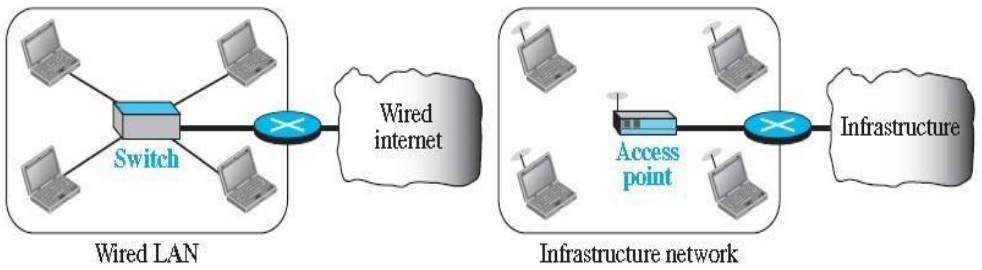
15.1 *Isolated LANs: wired versus wireless*



4. Connection to Other Networks

A wired LAN can be connected to another network or an internetwork such as the Internet using a router. A wireless LAN may be connected to a wired infrastructure network, to a wireless infrastructure network, or to another wireless LAN.

Figure 15.2 *Connection of a wired LAN and a wireless LAN to other networks*



5.Moving between Environments

In order to move from the wired environment to a wireless environment we need to change the network interface cards designed for wired environments to the ones designed for wireless environments. We replace the link-layer switch with an access point. In this change, the link-layer addresses will change but the network-layer addresses (IP addresses) will remain the same; we are moving from wired links to wireless links.

Access Control

The most important issue we need to discuss in a wireless LAN is access control. The CSMA/CD algorithm does not work in wireless LANs for three reasons:

1. To detect a collision, a host needs to send and receive at the same time (sending the frame and receiving the collision signal), which means the host needs to work in a duplex mode. Wireless hosts do not have enough power to do so (the power is supplied by batteries). They can only send or receive at one time.

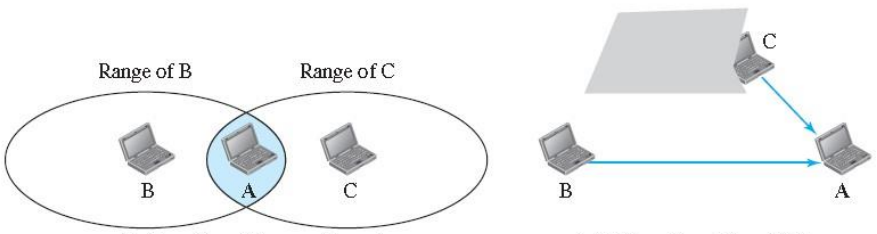
2. Hidden station problem,

In this a station may not be aware of another station's transmission due to some obstacles or range problems, collision may occur but not be detected. Hidden stations can reduce the capacity of the network because of the possibility of collision.

3.Since the distance between stations can be great. Signal fading could prevent a station at one end from hearing a collision at the other end.

To overcome the above three problems, Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) was invented for wireless LANs

Figure 15.3 *Hidden station problem*



IEEE 802.11PROJECT

IEEE has defined the specifications for a wireless LAN, called IEEE 802.11, It covers the physical and data-link layers. It is sometimes called *wireless Ethernet*. In some countries, including the United States, the public uses the term *Wi Fi (short for* wireless fidelity) as a synonym for *wireless LAN. However*, *W iFi is a wireless LAN* that is certified by the Wi Fi Alliance, a global, non-profit industry association

Architecture

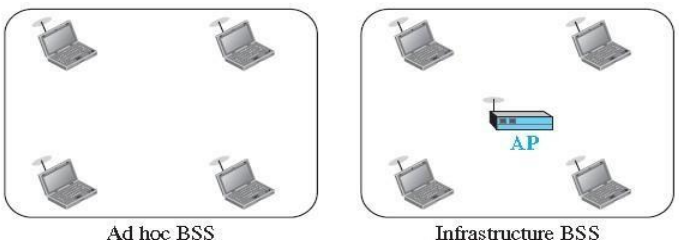
The standard defines two kinds of services:

- The basic service set (BSS)and
- The extended service set (ESS).

Basic Service Set

IEEE 802.11 defines the **basic service set (BSS) as the building blocks of a wireless** LAN.A basic service set 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 forma network without* the need of an AP; A BSS with an AP is sometimes referred toas an *infrastructure BSS.*

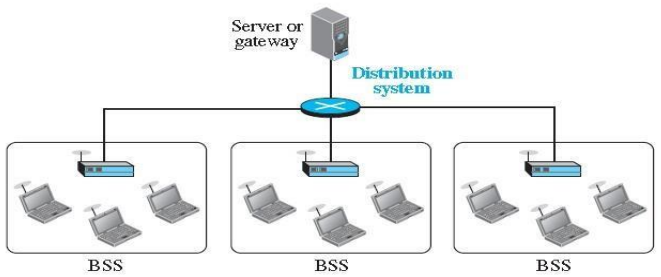
Figure 15.4 *Basic service sets (BSSs)*



Extended ServiceSet

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 a wired or a* wireless network. The distribution system connects the Aps in the BSSs. 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 15.5 *Extended service set (ESS)*



Here the stations within reach of one another can communicate without the use of an AP. However, communication between a station in a BSS and the outside BSS occurs via the AP. The idea is similar to communication in acellular network.

Station Types

IEEE802.11 defines three types of stations based on their mobility in a wireless LAN:

1. No-transition-is either stationary (not moving) or moving only inside a BSS.

2. BSS-transition-can move from one BSS to another, but the movement is confined in side one ESS.

3. ESS-transition mobility-can move from one ESS to another.

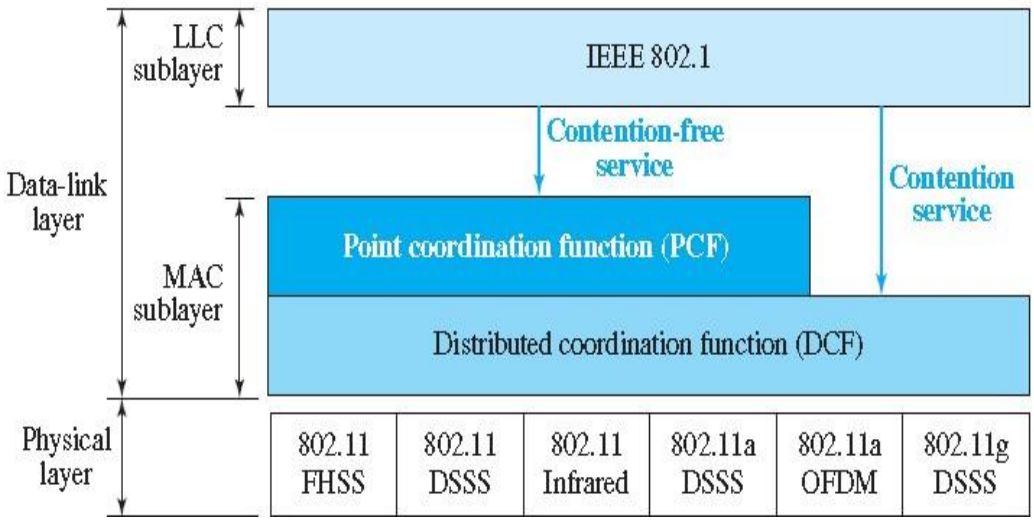
MAC Sublayer

IEEE802.11 defines two MAC sublayers:

- Distributed Co-ordination Function (DCF)
- Point Co-ordination Function (PCF).

Figure 15.6 shows the relationship between the two MAC sublayers, the LLC sublayer, and the physical layer.

Figure 15.6 *MAC layers in IEEE 802.11 standard*



Distributed Coordination Function

DCF uses CSMA/CA as the access method.

How do other stations defer sending their data if one station acquires access? In other words, how is the *collision avoidance aspect of this protocol accomplished?*

The key is a feature called NAV.

The stations create a timer called a ***network allocation vector (NAV)that shows how much time must pass before*** these stations are allowed to check the channel for idleness.

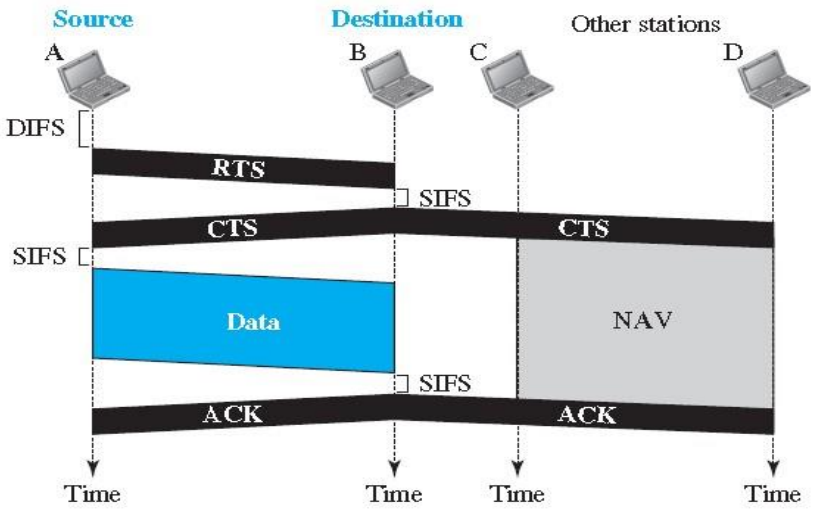
Network Allocation Vector

How is the collision avoidance aspect of this protocol accomplished?

The key is a feature called NAV.

The stations create a timer called a network allocation vector (NAV) that shows how much time must pass before these stations are allowed to check the channel for idleness. Each time a station accesses the system and sends an CTS frame, other stations start their NAV.

Figure 15.7 *CSMA/CA and NAV*



Collision During Handshaking

What happens if there is a collision during the time when RTS or CTS control frames are in transition, often called the *handshaking period? i.e. Two or more stations may try to* send RTS frames at the same time.

These control frames may collide.

However, because there is no mechanism for collision detection, the sender assumes there has been a collision if it has not received a CTS frame from the receiver. The backoff strategy is employed, and the sender tries again.

Hidden-Station Problem

The solution to the hidden station problem is the use of the handshake frames (RTS and CTS)

Figure 15.7 also shows that the RTS message from A reaches B, but not C. However, because both B and Care within the range of A, the CTS message, which contains the duration of data transmission from A to B, reaches C. Station C knows that some hidden station is using the channel and refrains from transmitting until that duration is over.

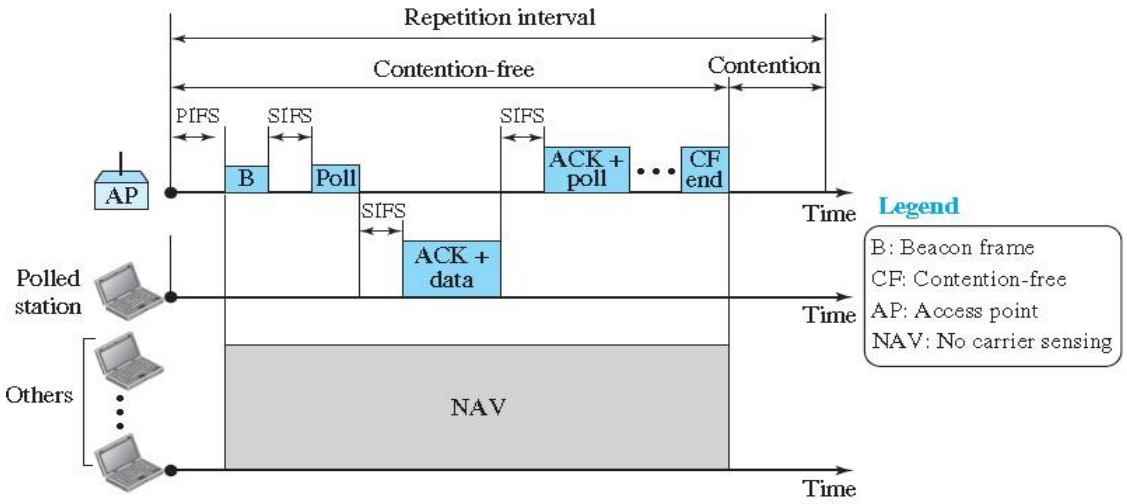
Point Coordination Function (PCF)

The **point coordination function (PCF)is an optional access method that can be** implemented in an infrastructure network (not in an ad hoc network). It is implemented on top of the DCF and is used mostly for time-sensitive transmission. PCF has a centralized, contention-free polling access method.

The AP performs polling for stations that are capable of being polled. The stations are polled one after another, sending any data they have to the AP.

To give priority to PCF over DCF, another interframe space, PIFS, has been defined. PIFS (PCF IFS) is shorter than DIFS. This means that if, at the same time, a station wants to use only DCF and an AP wants to use PCF, the AP has priority. Due to the priority of PCF over DCF, stations that only use DCF may not gain access to the medium. To prevent this, a repetition interval has been designed to cover both contention-free PCF and contention-based DCF traffic.

Figure 15.8 *Example of repetition interval*



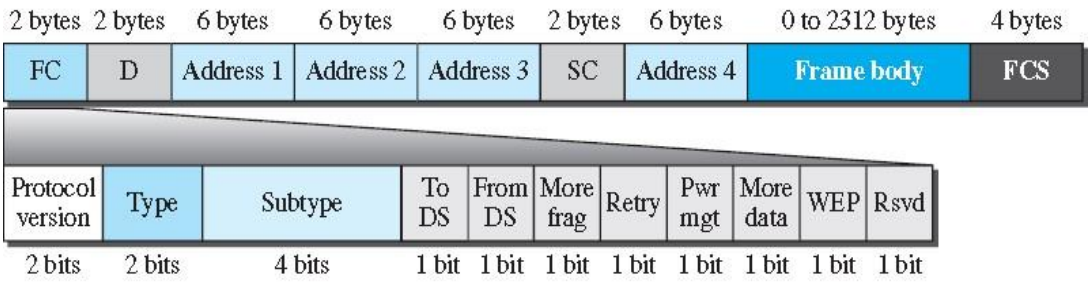
The *repetition interval*, which is repeated continuously, starts with a special control frame, called a beacon ***frame. When the stations hear the beacon frame, they start their NAV for the duration*** of the contention-free period of the repetition interval. During the repetition interval, the PC (point controller) can send a poll frame, receive data, send an ACK, receive an ACK, or do any combination of these (802.11 uses piggybacking). At the end of the contention-free period, the PC sends a CF end (contention-free end)frame to allow the contention-based stations to use the medium.

Fragmentation

The wireless environment is very noisy, so frames are often corrupted. A corrupt frame has to be retransmitted. The protocol, therefore, recommends fragmentation—the division of a large frame into smaller ones. It is more efficient to resend a small frame than a large one.

Frame Format

Figure 15.9 *Frame format*



Frame control (FC). The FC field is 2 bytes long and defines the type of frame and some control information.

Table 15.1 *Subfields in FC field*

Field	*Explanation*
Version	Current version is 0
Type	Type of information: management (00), control (01), or data (10)
Subtype	Subtype of each type (see Table 15.2)
To DS	Defined later
From DS	Defined later
More frag	When set to 1, means more fragments
Retry	When set to 1, means retransmitted frame
Pwr mgt	When set to 1, means station is in power management mode
More data	When set to 1, means station has more data to send
WEP	Wired equivalent privacy (encryption implemented)
Rsvd	Reserved

- ***D.*** *This field defines the duration of the transmission that is used to set the value* of NAV.
- ***Addresses.*** *There are four address fields, each 6 bytes long. The meaning of each* address field depends on the value of the *To DS and From DS subfields and will be* discussed later.
- ***Sequence control.*** *This field, often called the SC field, defines a 16-bit value. The* first four bits define the fragment number; the last 12 bits define the sequence number, which is the same in all fragments.
- ***Frame body.*** *This field, which can bebetween0 and 2312bytes, contains in formation* based on the type and the sub type defined in the FC field.

- ***FCS.****TheFCSfieldis4 bytes long and contains a CRC-32 error-detection* sequence.

Frame Types

A wireless LAN defined by IEEE 802.11 has three categories of frames:

1.Management Frames

Management frames are used for the initial communication between stations and access points.

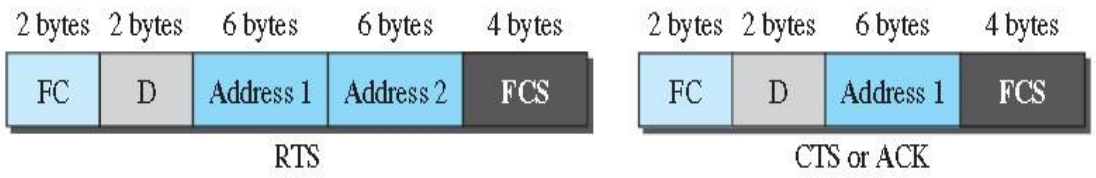
2.Control Frames

Control frames are used for accessing the channel and acknowledging frames.

3.DataFrames

Data frames are used for carrying data and control information.

Figure 15.10 *Control frames*



For control frames the value of the type field is 01; the values of the subtype fields for frames we have discussed are shown in Table 15.2.

Table 15.2 *Values of subtype fields in control frames*

Subtype	*Meaning*
1011	Request to send (RTS)
1100	Clear to send (CTS)
1101	Acknowledgment (ACK)

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 or1, resulting in* four different situations.

The interpretation of the four addresses (address 1to address4) in the MAC frame depends on the value of these flags,

- Address1isalwaysthe address of the next device that the frame will visit.
- Address2isalwaysthe address of the previous device that the frame has left.
- Address3isthe address of the final destination station
- Address 4 is the original source when the distribution system is also wireless.

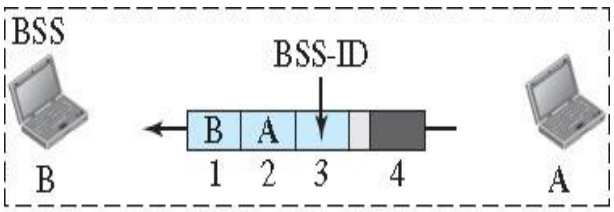
Table 15.3 *Addresses*

To DS	*From DS*	*Address 1*	*Address 2*	*Address 3*	*Address 4*
0	0	Destination	Source	BSS ID	N/A
0	1	Destination	Sending AP	Source	N/A
1	0	Receiving AP	Source	Destination	N/A
1	1	Receiving AP	Sending AP	Destination	Source

Case 1:00

- In this case, *To DS = 0 and From DS = 0. This means that the frame is* not going to a distribution system (*To DS = 0) and is not coming from a distribution* system (*From DS = 0)*.

- *The frame is going from one station in a BSS to* another without passing through the distribution system.



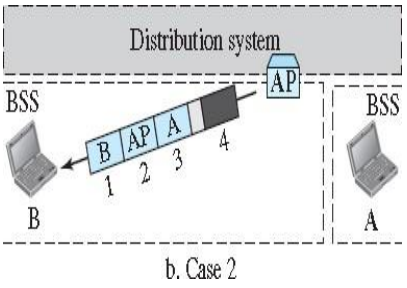
a. Case 1

Case 2:01

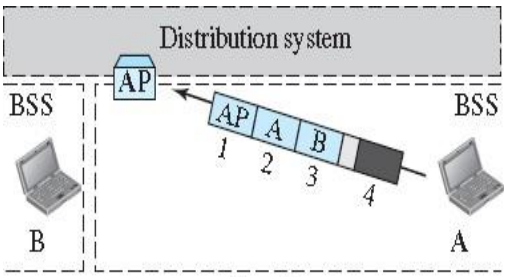
In this case, *To DS = 0 and From DS = 1. This means that the frame is* coming from a distribution system (*From DS = 1). The frame is coming from an* AP and going to a station.

Case 3:10

In this case, *To DS = 1 and From DS = 0. This means that the frame* is going to a distribution system (*To DS = 1). The frame is going from a station* to an AP. The ACK is sent to the original station.



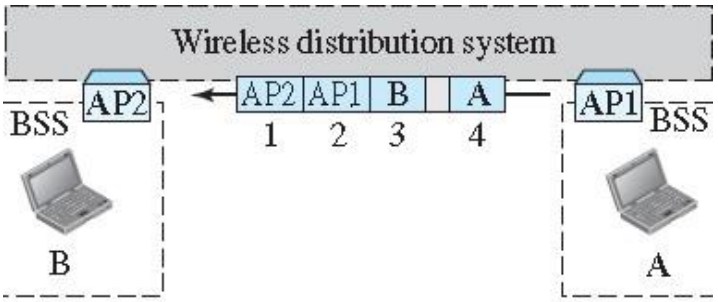
b. Case 2



c. Case 3

Case 4:11

In this case, *To DS = 1 and From DS = 1. This is the case in which the* distribution system is also wireless. The frame is going from one AP to another AP in a wireless distribution system.

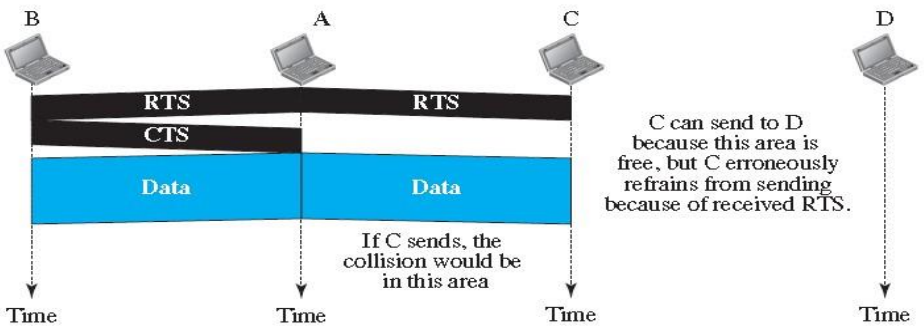


d. Case 4

Exposed Station Problem

In this problem a station refrains from using a channel when it is, in fact, available.

Figure 15.12 *Exposed station problem*



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.

Table 15.4 *Specifications*

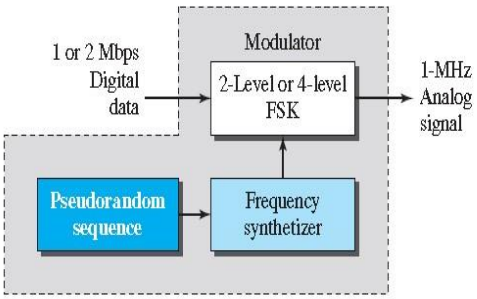
IEEE	*Technique*	*Band*	*Modulation*	*Rate (Mbps)*
802.11	FHSS	2.400–4.835 GHz	FSK	1 and 2
	DSSS	2.400–4.835 GHz	PSK	1 and 2
	None	Infrared	PPM	1 and 2
802.11a	OFDM	5.725–5.850 GHz	PSK or QAM	6 to 54
802.11b	DSSS	2.400–4.835 GHz	PSK	5.5 and 11
802.11g	OFDM	2.400–4.835 GHz	Different	22 and 54
802.11n	OFDM	5.725–5.850 GHz	Different	600

IEEE 802.11 FHSS

IEEE 802.11 FHSS uses the **frequency-hopping spread spectrum (FHSS) method**. FHSS uses the 2.400–4.835 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.

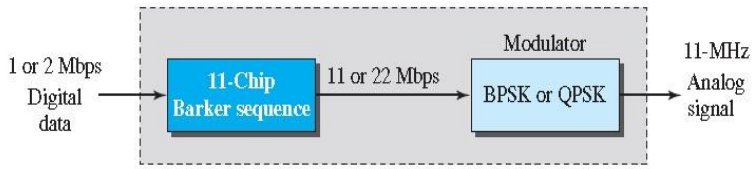
Figure 15.13 *Physical layer of IEEE 802.11 FHSS*



IEEE 802.11DSSS

IEEE 802.11 DSSS uses the **direct-sequence spread spectrum (DSSS) method.** DSSS uses the 2.400– 4.835 GHz ISM band. The modulation technique in this specification is PSK at 1 M baud/s. The system allows1or 2 bits/baud (BPSK or QPSK), which results in a data rate of 1or 2Mbps.

Figure 15.14 *Physical layer of IEEE 802.11 DSSS*

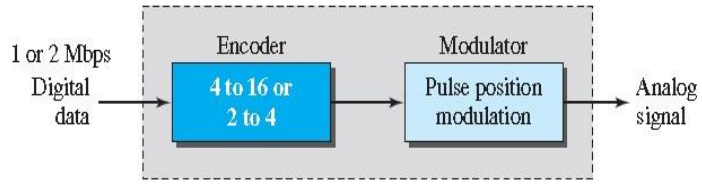


IEEE 802.11 Infrared

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. For a 2-Mbps data rate, a 2-bit sequence is first mapped into a4-bitsequence

The mapped sequences are then converted to optical signals; the presence of light specifies 1, the absence of light specifies0.

Figure 15.15 *Physical layer of IEEE 802.11 infrared*



IEEE 802.11aOFDM

IEEE 802.11a OFDM describes the **orthogonal frequency-division multiplexing (OFDM) method for signal generation in a 5.725-5.850 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.

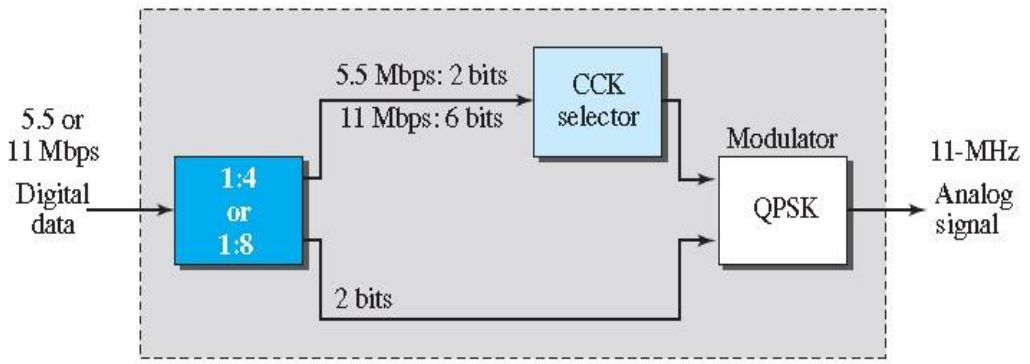
The band is divided into 52 sub bands, with 48subbandsfor sending 48groupsof bits at a time and4 sub bands for control information. Dividing the band into sub bands diminishes the effects of interference. If the sub bands are used randomly, security can also be increased. OFDM uses PSK and QAM for modulation. The common data rates are 18 Mbps (PSK) and 54 M bps (QAM).

IEEE 802.11bDSSS

IEEE 802.11b DSSS describes the **high-rate direct-sequence spread spectrum (HRDSSS)** method for signal generation in the 2.400–4.835 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. The 5.5-Mbps version uses BPSK and transmits at 1.375 M baud/s with 4-bit CCK encoding. The 11-Mbps version uses QPSK and transmits at 1.375 Mbps with 8-bit CCK encoding.

Figure 15.16 *Physical layer of IEEE 802.11b*



IEEE 802.11g

This new specification defines forward error correction and OFDM using the 2.400– 4.835 GHz ISM band. The modulation technique achieves a 22- or 54-Mbps data rate. It is backward-compatible with 802.11b, but the modulation technique is OFDM.

IEEE 802.11n

An upgrade to the 802.11 project is called 802.11n (the next generation of wireless LAN). The goal is to increase the throughput of 802.11 wireless LANs. The new standard emphasizes not only the higher bit rate but also eliminating some unnecessary overhead.

The standard uses what is called ***MIMO (multiple-input multiple-output antenna) to overcome the noise problem in wireless LANs. The idea is that if we can*** send multiple output signals and receive multiple input signals, we are in a better position to eliminate noise. Some implementations of this project have reached up to 600 Mbps data rate.

BLUETOOTH

Bluetooth isa wireless LAN technology designed to connect devices of different functions such as telephones, notebooks, computers (desktop and laptop), cameras, printers, and even coffee makers when they are at a short distance from each other.

A Bluetooth LAN is an ad hoc network, which means that the network is formed spontaneously. A Bluetooth LAN can even be connected to the Internet

History

Bluetooth was originally started as a project by the Ericsson Company. It is named for Harald Blaat and, the king of Denmark (940-981) who united Denmark and Nor way. *Blaat and 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.

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 healthcare 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.

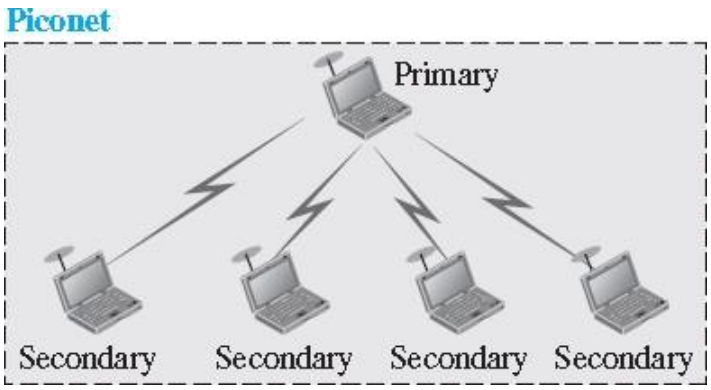
Architecture

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; there stare called secondaries.*

All the secondary stations synchronize their clocks and hopping sequence with the primary. The communication between the primary and secondary stations can be one-to-one or one-to-many. Although a piconet can have a maximum of seven secondaries, additional secondaries can be in the *parked state.*

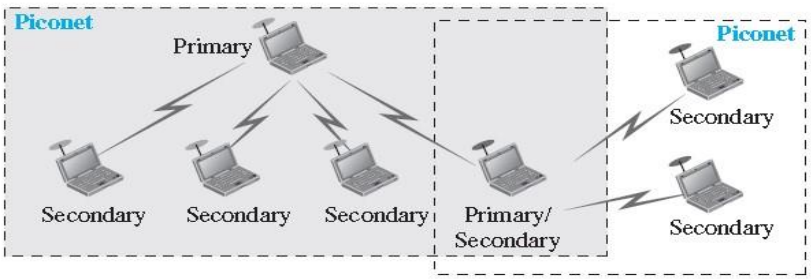
A secondary in a parked state is synchronized with the primary, but cannot take part in communication until it is moved from the parked state to the active state.



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.

Figure 15.18 *Scatternet*

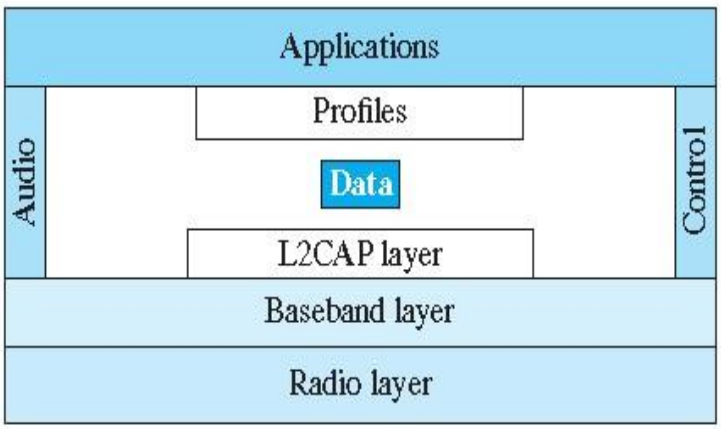


Bluetooth Devices

A Bluetooth device has a built-in short-range radio transmitter. The current data rate is 1 M bps 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 Layers

Figure 15.19 *Bluetooth layers*



L2CAP

The **Logical Link Control and Adaptation Protocol, or L2CAP (L2 here means LL),** is roughly equivalent to the LLC sublayer in LANs. It is used for data exchange on an ACL link; SCO channels do not useL2CAP.

Figure 15.20 *L2CAP data packet format*

The 16-bit length field defines the size of the data, in bytes, coming from the upper layers. Data can be up to 65,535 bytes. The channel ID (CID) defines a unique identifier for the virtual channel created at this level

The L2CAP has specific duties: multiplexing, segmentation and reassembly, Quality of service (QoS), and group management.

Base band Layer

The baseband layer is roughly equivalent to the MAC sublayer in LANs. The access method is TDMA. The primary and secondary stations communicate with each other using timeslots.

Note: communication is only between the primary and a secondary; Secondaries cannot communicate directly with one another.

TDMA

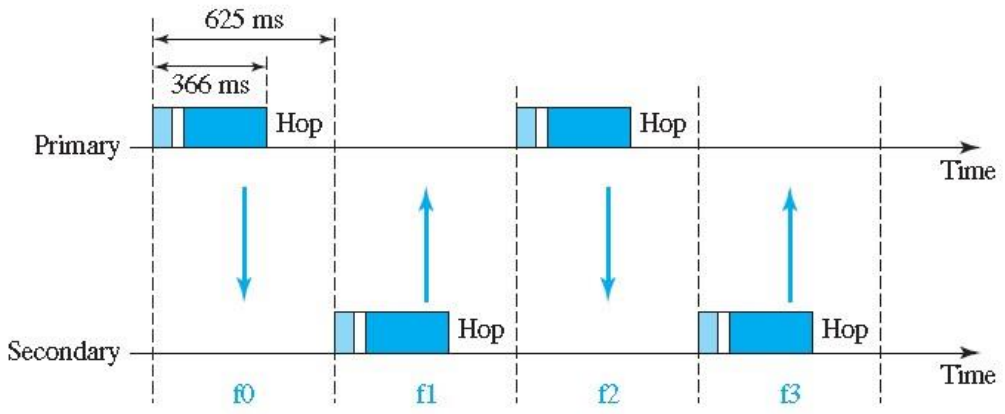
Bluetooth uses a form of TDMA that is called ***TDD-TDMA (time-division duplex TDMA). TDD- TDMA is a kind of half-duplex communication in which the sender and*** receiver send and receive data, but not at the same time (half-duplex);

Single-Secondary Communication

If the piconet has only one secondary, the TDMA operation is very simple. The time is divided into slots of 625 µs. The primary uses even-numbered slots (0, 2, 4, . . .); the secondary uses odd- numbered slots (1, 3, 5, . ..).

TDD-TDMA allows the primary and the secondary to communicate in half-duplex mode. In slot 0, the primary sends and the secondary receives; in slot 1, the secondary sends and the primary receives. The cycle is repeated.

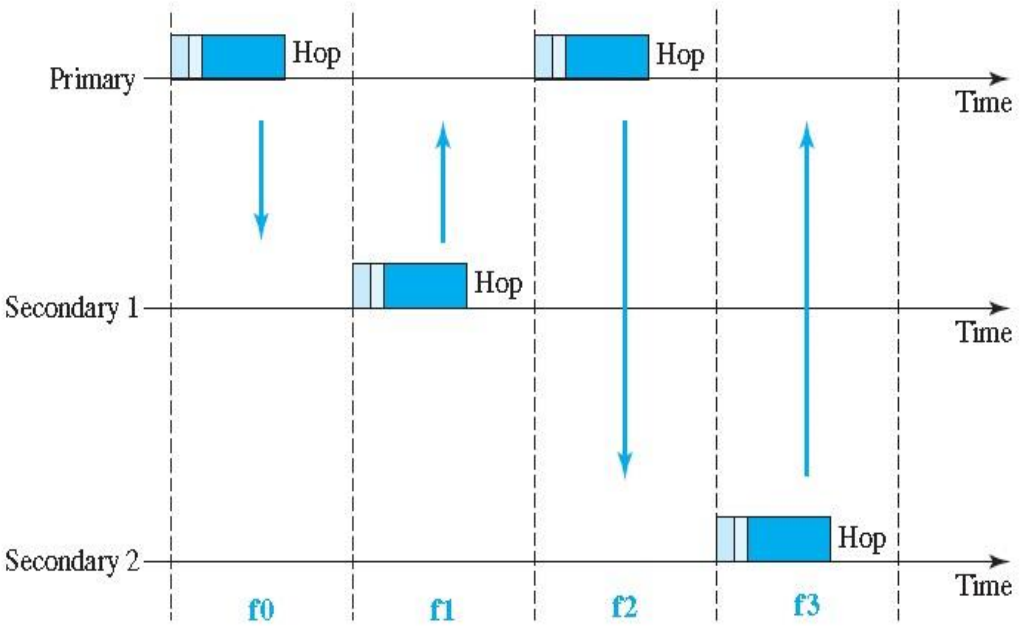
Figure 15.21 *Single-secondary communication*



Multiple-Secondary Communication

The process is a little more involved if there is more than one secondary in the piconet. Again, the primary uses the even-numbered slots, but a secondary send in the next odd-numbered slot if the packet in the previous slot was addressed to it. All secondaries listen on even-numbered slots, but only one secondary sends in any odd-numbered slot.

Figure 15.22 *Multiple-secondary communication*



Links

Two types of links can be created between a primary and a secondary:

- SCO (synchronous connection-oriented) Links

  A SCO link is used when avoiding latency (delay in data delivery) is more important than integrity (error-free delivery). In an SCO link, a physical link is created between the primary and a secondary by reserving specific slots at regular intervals. The basic unit of connection is two slots, one for each direction. If a packet is damaged, it is never retransmitted. SCO is used for real-time audio where avoiding delay is all-important. A secondary can create up to three SCO links with the primary, sending digitized audio (PCM)at 64kbpsin each link.

- ACL (**asynchronous connectionless link**) links.

  An ACL link is used when data integrity is more important than avoiding latency. An asynchronous **connectionless link (ACL) is used when data integrity is** more important than avoiding latency.

In this type of link, if a payload encapsulated in the frame is corrupted, it is retransmitted. A secondary return an ACL frame in the available odd-numbered slot if the previous slot has been addressed to it. ACL can use one, three, or more slots and can achieve a maximum data rate of 721kbps.

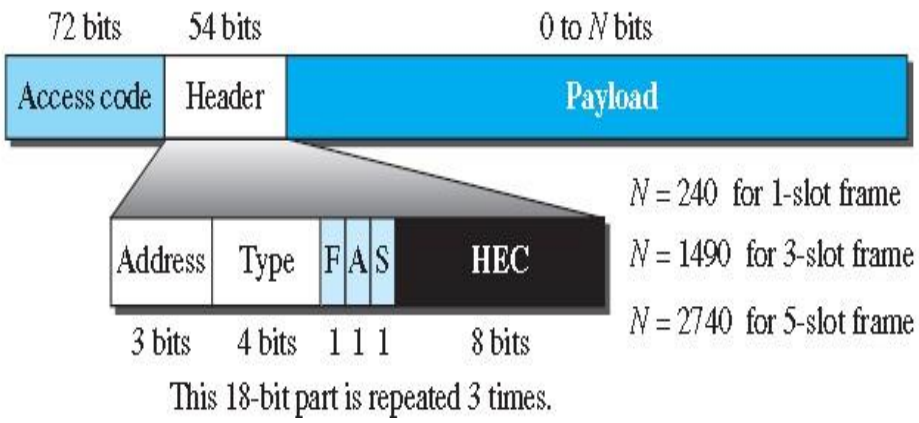
Frame Format

A frame in the baseband layer can be one of three types: one-slot, three-slot, or five slot. A slot is 625 µs in length. However, in a one-slot frame exchange, 259 µs is needed for hopping and control mechanisms. This means that a one-slot frame can last only625 −259, or 366 µs. With a 1-MHz bandwidthand1bit/Hz, the size of a one slot frame is366 bits.

A three-slot frame occupies three slots. However, since 259 µs is used for hopping, the length of the frame is 3 ×625 − 259 = 1616 µs or 1616 bits. A device that uses a three-slot frame remains at the same hop (at the same carrier frequency) for three slots. A five-slot frame also uses 259 bits for hopping, which means that the length of the frame is5×625 −259 =2866bits.

Frame format types

Figure 15.23 *Frame format types*



- *Access code. This72-bit field normally contains synchronization bits and the* identifier of the primary to distinguish the frame of on piconet from that of another.

Header.This54-bitfield isa repeated 18-bitpattern. Each pattern has the following subfields:

Address. The 3-bit address subfield can define up to seven secondaries (1 to7).

If the address is zero, it is used for broadcast communication from the primary to all secondaries.

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

F. This 1-bit subfield is for flow control. When set (1), it indicates that that device is unable to receive more frames (buffer is full).

A. This 1-bit subfield is for acknowledgment. Bluetooth uses Stop-and-Wait ARQ; 1 bit is sufficient for acknowledgment.

S. This 1-bit subfield holds a sequence number. Bluetooth uses Stop-and-Wait ARQ; 1 bit is sufficient for sequence numbering.

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*** coming from the upper 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 10m.

Modulation

To transform bits to a signal, Bluetooth uses a sophisticated version of FSK, called GFSK (FSK with Gaussian bandwidth filtering;)GFSK has a carrier frequency.

- Bit1 is represented by a frequency deviation above the carrier;
- bit0 is represented by a frequency deviation below the carrier.

The frequencies, are defined according to the following formula for each channel.

$$Fc = 2402 + n \text{ MHz} \qquad n = 0, 1, 2, 3, \ldots, 78$$

Frequency-Hopping Spread Spectrum (FHSS)method

Bluetooth uses the **frequency-hopping spread spectrum (FHSS) method in the physical layer** to avoid interference from other devices or other networks. Bluetoothhops1600 times per second, which means that each device changes its modulation frequency 1600 times per second. A device uses a frequency for only 625 μs (1/1600 s) before it hops to another frequency; the dwell time is 625 μs.

Module 3

Network Layer

Introduction

The network layer in the TCP/IP protocol suite is responsible for the host-to-host delivery of datagrams. It provides services to the transport layer and receives services from the data-link layer. In this chapter, we introduce the general concepts and issues in the network layer.

NETWORK-LAYER SERVICES

- **Packetizing**
- **Routing and Forwarding**
- **Other Services**

  i)Error Control

  ii)Flow Control

  iii)Congestion

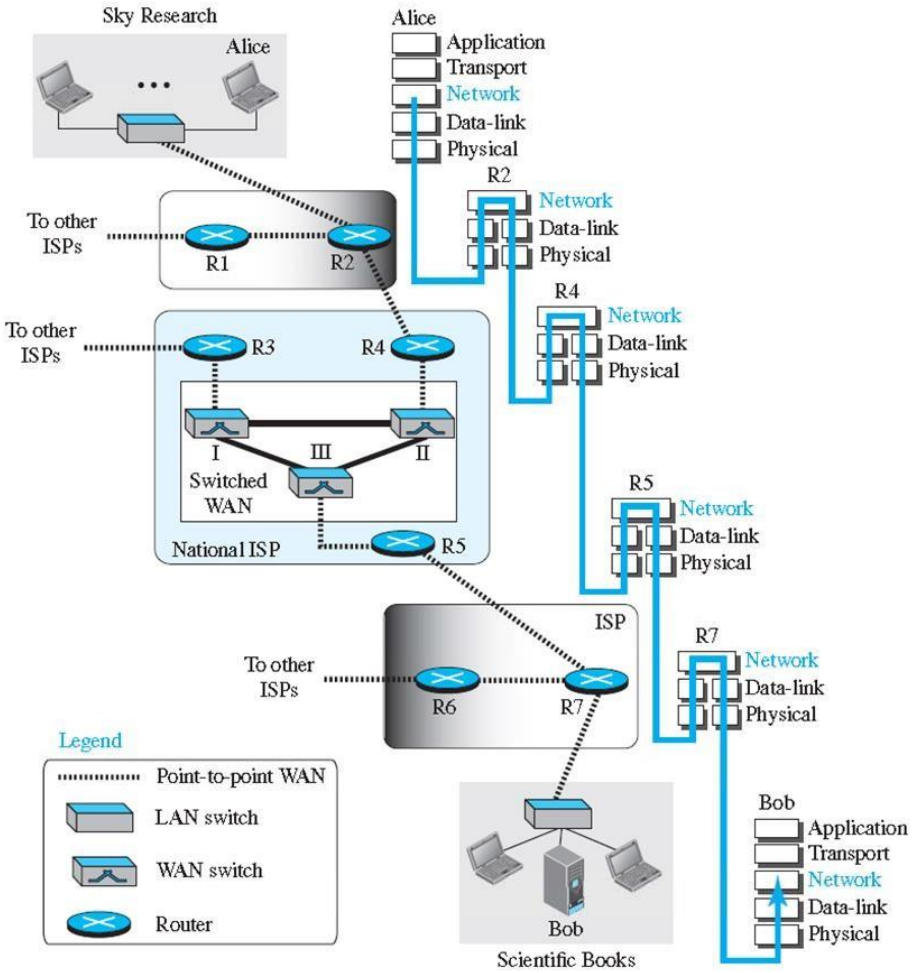
  Control

  iv)Quality f Service

  v)Security

As the figure shows, the network layer is involved at the source host, destination host, and all routers in the path (R2, R4, R5, and R7). At the source host (Alice), the network layer accepts a packet from a transport layer, encapsulates the packet in a datagram, and delivers the packet to the data-link layer. At the destination host (Bob), the datagram is decapsulated, and the packet is extracted and delivered to the corresponding transport layer. Although the source and destination hosts are involved in all five layers of the TCP/IP suite, the routers use three layers if they are routing packets only;

Figure 18.1 *Communication at the network layer*



Packetizing

The first duty of the network layer is definitely **packetizing: encapsulating the payload** in a packet at the source and decapsulating the payload from the packet at the destination. In other words, network layer is to carry a pay load from the source to the destination without changing it or using it.

The source is not allowed to change the content of the payload unless it is too large for delivery and needs to be fragmented. If the packet is fragmented at the source or at routers along the path, the network layer is responsible for waiting until all fragments arrive, reassembling them, and delivering them to the upper-layer protocol. The routers are not allowed to change source and destination addresses either.

Routing and Forwarding

Routing

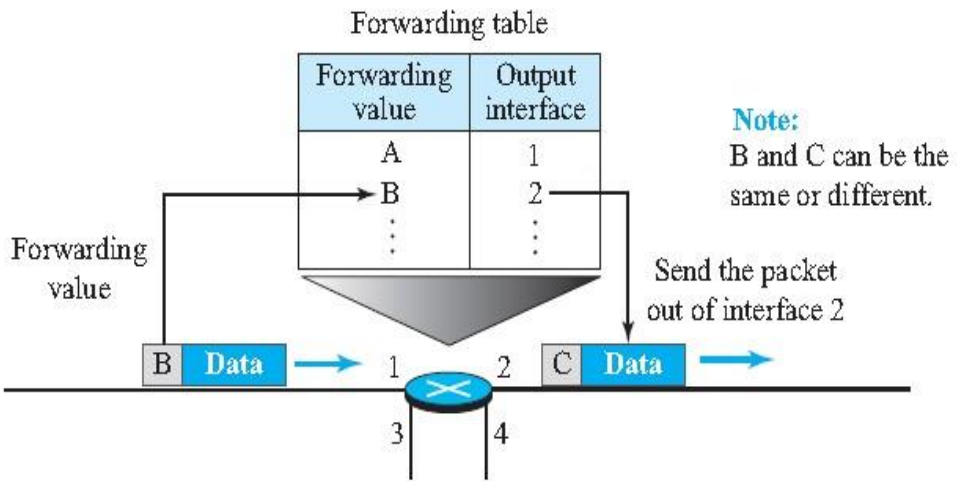
The network layer is responsible for routing the packet from its source to the destination. Generally, there is more than one route from the source to the destination. The network layer is responsible for finding the best one among these possible routes. The network layer needs to have some specific

strategies for defining the best route. The routing protocols, should be run before any communication occurs.

Forwarding

Forwarding can be defined as the action applied by each router when a packet arrives at one of its interfaces. A router normally uses *forwarding table* for applying this action is sometimes called the routing *table*. To make decision, the router uses a piece of information in the packet header, which can be the destination address or a label, to find the corresponding output interface number in the forwarding table.

Figure 18.2 *Forwarding process*



Other Services

Error Control

Although error control also can be implemented in the network layer, the designers of the network layer ignore this issue. One reason is the fact that the packet in the network layer may be fragmented at each router, which makes error checking at this layer in efficient. Although the network layer in the Internet does not directly provide error control, the Internet uses an auxiliary protocol, ICMP, that provides some kind of error control .

Flow Control

Flow control regulates the amount of data a source can send without overwhelming the receiver. To control the flow of data, the receiver needs to send some feedback to the sender to inform the latter that it is overwhelmed with data. The network layer, however, does not directly provide any flow control. The datagrams are sent by the sender when they are ready, without any attention to the readiness of the receiver.

Congestion Control

Congestion in the network layer is a situation in which too many datagrams are present in an area of the

Internet. Congestion may occur if the number of datagrams sent by source computers is beyond the capacity of the network or routers. In this situation, some routers may drop some of the datagrams.

However, as more datagrams are dropped, the situation may become worse because, due to the error control mechanism at the upper layers, the sender may send duplicates of the lost packets. If the congestion continues, sometimes a situation may reach a point where the system collapses and no datagrams are delivered.

Quality of Service

As the Internet has allowed new applications such as multimedia communication the quality of service (QoS)of the communication has become more and more important. However, to keep the network layer untouched, these provisions are mostly implemented in the upper layer.

Security

Security was not a concern when the Internet was originally designed because it was used by a small number of users at universities for research activities; other people had no access to the Internet. The network layer was designed with no security provision. Today, however, security is a big concern. To provide security for a connectionless network layer, we need to have another virtual level that changes the connectionless service toa connection-oriented service.

PACKETSWITCHING

A router, in fact, is a switch that creates a connection between an input port and an output port (or a set of output ports), Just as an electrical switch connects the input to the output to let electricity flow. Switching techniques are divided into two broad categories, circuit switching and packet switching,

Only packet switching is used at the network layer because the unit of data at this layer is a packet. Circuit switching is mostly used at the physical layer;

A packet-switched network can use two different approaches to route the packets:

1. Datagram Approach: Connectionless Service

When the network layer provides a connection less service, each packet traveling in the Internet is an independent entity; There is no relationship between packets belonging to the same message.

The switches in this type of network are called *routers. A packet* belonging to a message may be followed by a packet belonging to the same message or to a different message. Each packet is routed based on the information contained in its header: source and destination addresses. The destination address defines where it should go; the source address defines where it comes from.

Figure 18.3 *A connectionless packet-switched network*

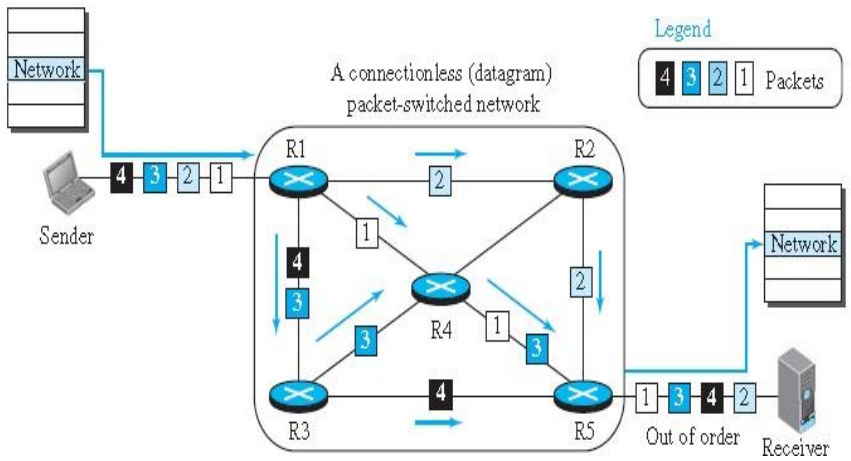
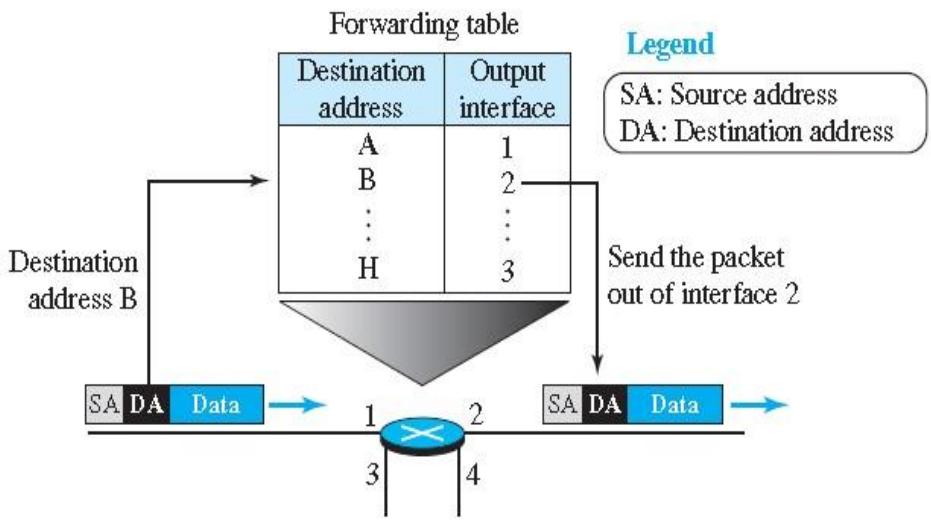


Figure 18.4 *Forwarding process in a router when used in a connectionless network*



2. Virtual-Circuit Approach: Connection-Oriented Service

In a connection-oriented service (also called *virtual-circuit approach*), *there is a relationship between* all packets belonging to a message. Before all datagrams in a message can be sent, a virtual connection should be set up to define the path for the datagrams. After connection setup, the datagrams can all follow the same path. In this type of service, not only must the packet contain the source and destination addresses, it must also contain a flow label. A flow label is a virtual circuit identifier that defines the virtual path the packet should follow.

Figure 18.5 *A virtual-circuit packet-switched network*

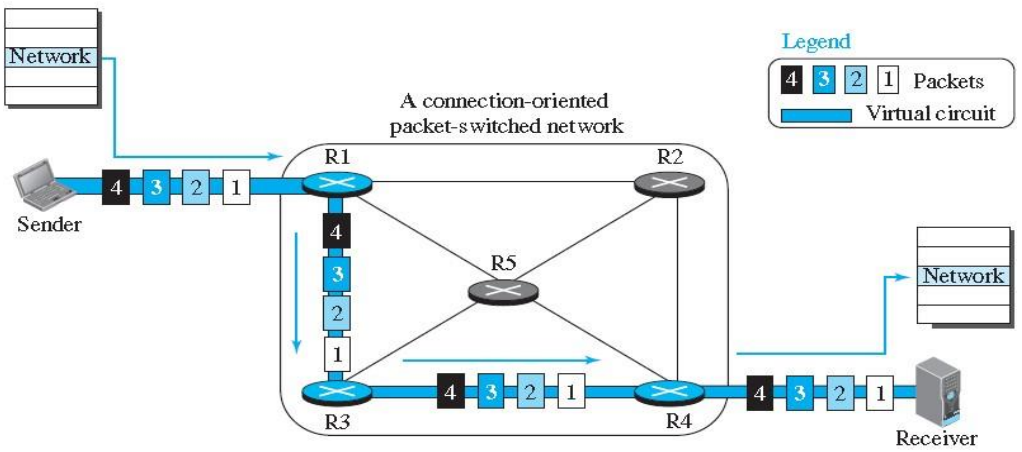
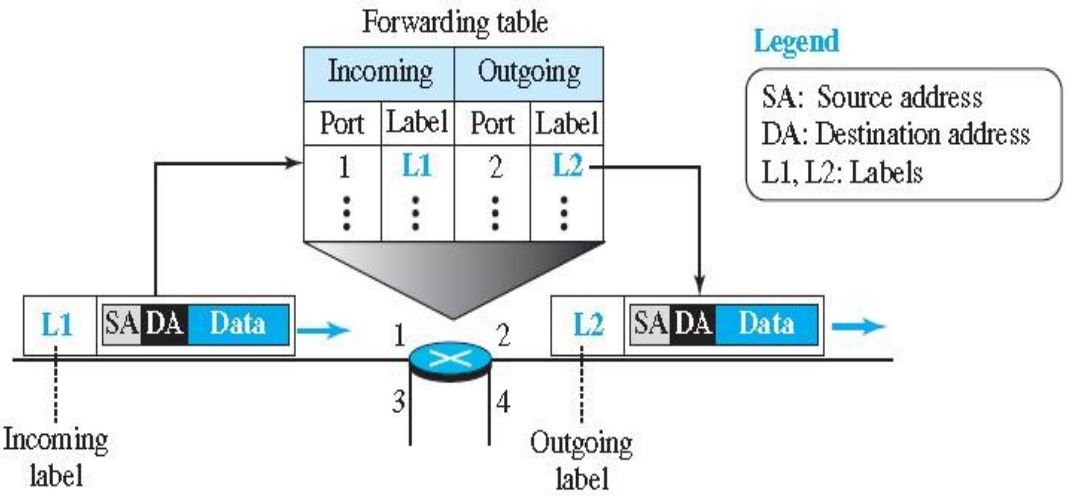


Figure 18.6 *Forwarding process in a router when used in a virtual-circuit network*



IPV4ADDRESSES

The identifier used in the IP layer of the TCP/IP protocol suite to identify the connection of each device to the Internet is called the Internet address or IP address. An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a host or a router to the Internet.

The IP address is the address of the connection, not the host or the router, because if the device is moved to another network, the IP address may be changed. IPv4 addresses are unique in the sense that each address defines one, and only one, connection to the Internet.

Address Space

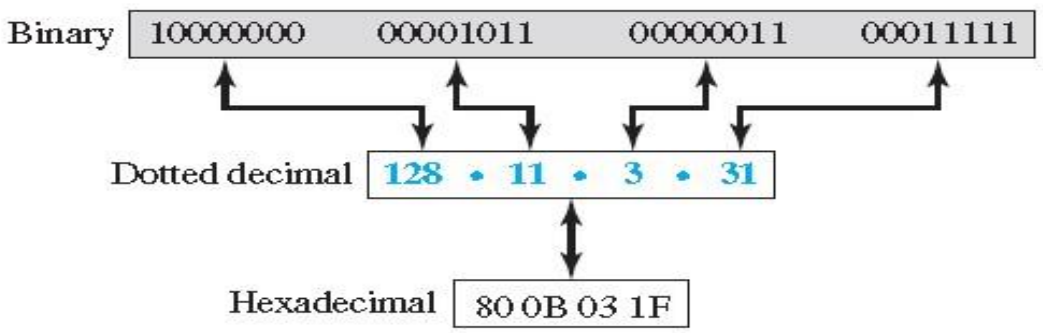
An **address space is** the total number of addresses used by the protocol. If a protocol uses *b bits to define an* address, the address space is 2^b *because each bit can have two different values (0 or 1).* IPv4 uses 32-bit addresses, which means that the address space is 2^{32} or 4,294,967,296 (more than four billion).If there were no restrictions, more than 4billion devices could be connected to the Internet.

Notation

There are three common notations to show an IPv4 address:

- binary notation (base2),
- dotted-decimal notation (base 256), and
- hexadecimal notation (base16).

Three different notations in IPv4 addressing



Hierarchy in Addressing

In any communication network that involves delivery, such as a telephone network or a postal network, the addressing system is hierarchical. In a postal network, the postal address (mailing address) includes the country, state, city, street, house number, and the name of the mail

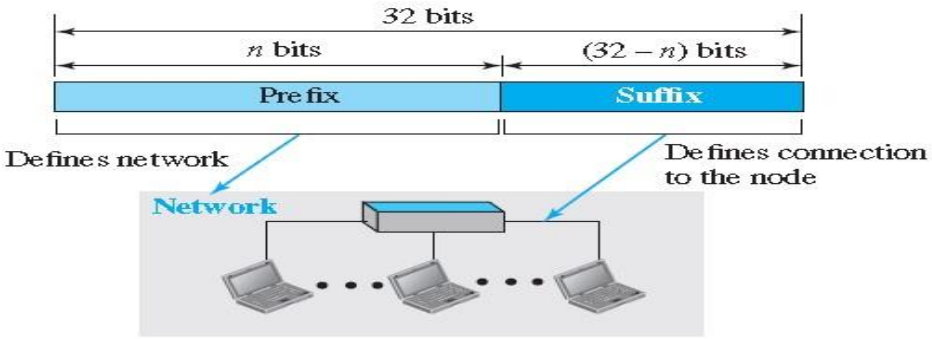
recipient. Similarly, a telephone number is divided into the country code, area code, local exchange, and the connection.

A 32-bit IPv4 address is also hierarchical, but divided only into two parts. The first part of the address, called the *prefix, defines the network; the second part of the* address, called the *suffix, defines the node* (connection of a device to the Internet).

The prefix length is *n bits and the suffix length is (32 - n) bits.*

A prefix can be fixed length or variable length. The network identifier in the IPv4 was first designed as a fixed-length prefix. This scheme, which is now obsolete, is referred to as classful addressing. The new scheme, which is referred to as classless addressing, uses a variable-length network prefix.

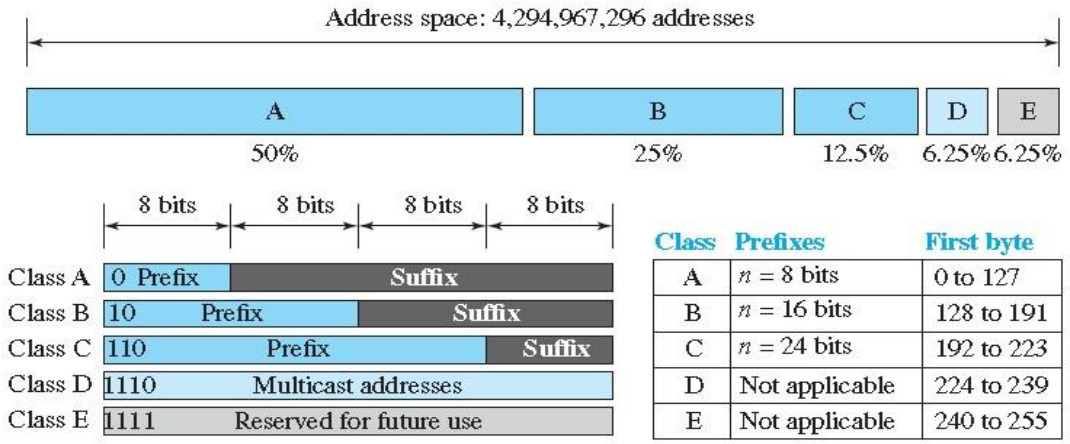
Hierarchy in addressing



Classful Addressing

When the Internet started, an IPv4 address was designed with a fixed-length prefix, but to accommodate both small and large networks, three fixed-length prefixes were designed instead of one (*n = 8*, *n = 16*, *and n = 24)*. The whole address space was divided into five classes (class A, B, C, D, and E), as shown in Figure 18.18. This scheme is referred to as **classful addressing.**

Figure 18.18 *Occupation of the address space in classful addressing*



Class	Prefixes	First byte
A	*n* = 8 bits	0 to 127
B	*n* = 16 bits	128 to 191
C	*n* = 24 bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

Address Depletion

The reason that classful addressing has become obsolete is address depletion. Since the addresses were not distributed properly, the Internet was faced with the problem of the addresses being rapidly used up. This resulted in no more addresses available for organizations and individuals that needed to be connected to the Internet.

To understand the problem, let us think about class A. This class can be assigned to only 128 organizations in the world, but each organization needs to have a single network (seen by the rest of the world) with 16,777,216 nodes (computers in this single network). Since there may be only a few organizations that are this large, most of the addresses in this class were wasted (unused).

Class B addresses were designed for midsize organizations, but many of the addresses in this class also remained unused.

Class C addresses have a completely different flaw in design. The number of addresses that can be used in each network (256) was so small that most companies were not comfortable using a block in this address class. Class E addresses were almost never used, wasting the whole class.

In class A, the network length is8bits, but since the first bit, which is0, defines the class, we can have only seven bits as the network identifier. This means there are only 27 = 128 networks in the world that can have a class A address.

Subnetting and Super netting

To alleviate address depletion, two strategies were proposed and, to some extent, implemented: subnetting and super netting.

In subnetting, a class A or class B block is divided into several subnets. Each subnet has a larger prefix length than the original network. For example, if a network in class A is divided into four subnets, each subnet has a prefix of n sub=10.

At the same time, if all of the addresses in a network are not used, subnetting allows the addresses to be divided among several organizations. This idea did not work because most large organizations were not happy about dividing the block and giving some of the unused addresses to smaller organizations.

While subnetting was devised to divide a large block into smaller ones, super netting was devised to combine several class C blocks into a larger block to be attractive to organizations that need more than the 256 addresses available in a class C block. This idea did not work either because it makes the routing of packets more difficult.

Advantage of Classful Addressing

Given an address, we can easily find the class of the address and, since the prefix length for each classis fixed, we can find the prefix length immediately. In other words, the prefix length in classful addressing is inherent in the address; no extra information is needed to extract the prefix and the suffix.

Network Layer Protocols and Unicast Routing

Network Layer Protocols

In this chapter, we show how the network layer is implemented in the TCP/IP protocol suite. The protocols in the network layer have gone through a few versions; in this chapter, we concentrate on the current versionv4.

Communication at the network layer is host-to-host (computer-to-computer); a computer somewhere in the world needs to communicate with another computer somewhere else in the world through the Internet.

The packet transmitted by the sending computer may pass through several LANs or WANs before reaching the destination computer. A global addressing scheme called logical addressing in required for this communication. The term IP address refers to the logical address in the network layer of the TCP/IP protocol suite.

Communication at the network layer in the Internet is connectionless. If reliability is important, IPv4 must be paired with a reliable transport-layer protocol such as TCP.

1. Position of IPv4 and other network protocols in TCP/IP protocol suite

The network layer in version 4 can be thought of as one main protocol and three auxiliary ones as shown in Figure 4.1.

- The main protocol, Internet Protocol version 4 (IPv4), is responsible for packetizing, forwarding, and delivery of a packet at the network layer.
- The Internet Control Message Protocol version 4 (ICMPv4) helps IPv4 to handle some errors that may occur in the network-layer delivery.
- The Internet Group Management Protocol (IGMP) is used to help IPv4 in multicasting.
- The Address Resolution Protocol (ARP) is used to glue the network and data-link layers in mapping network-layer addresses to link-layer addresses

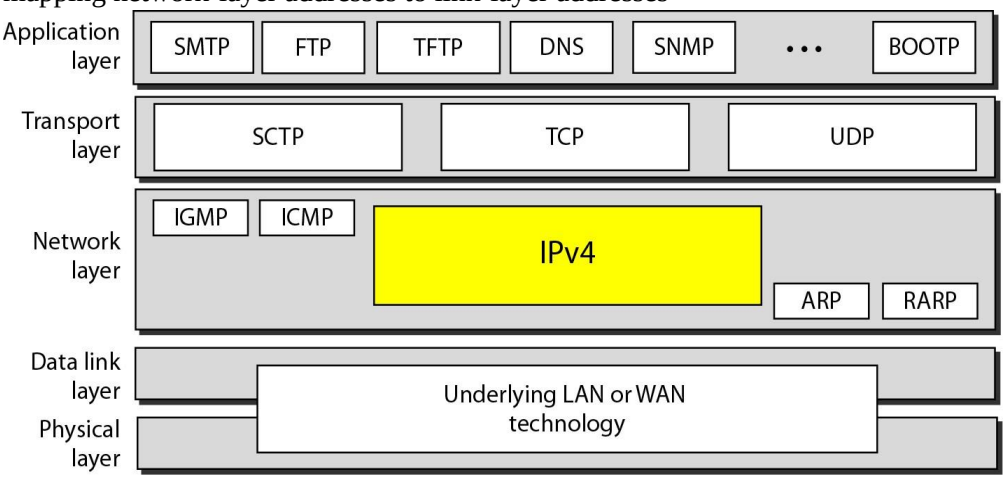


Figure 4.1: Position of IP and other network-layer protocols in TCP/IP protocol suite

Datagram Format

The Internet Protocol version4 (IPv4) is the delivery mechanism used by the TCP/IP protocols. Packets used by the IP are called datagrams. A datagram is a variable-length packet consisting of two parts: header and payload (data). The header is 20 to 60 bytes in length and contains information essential to routing and delivery. It is customary in TCP/IP to show the header in 4-byte sections.

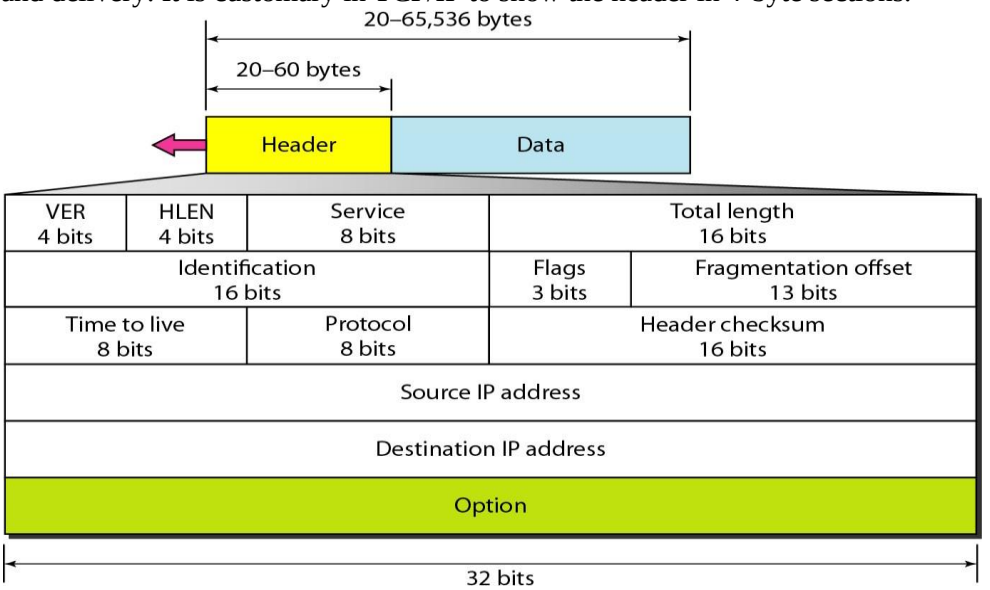


Figure 4.2: IPv4 Datagram

A brief description of each field is in order:

1. **Version Number:** The 4-bit version number (VER) field defines the version of the IPv4 protocol, which, obviously, has the value of 4.

2. **Header Length:** The 4-bit header length (HLEN) field defines the total length of the datagram header in 4-byte words. The IPv4 datagram has a variable-length header. When a device receives a datagram, it needs to know when the header stops and the data, which is encapsulated in the packet, starts. The total length is divided by 4 and the value is inserted in the field. The receiver needs to multiply the value of this field by 4 to find the total length.

3. **Service Type:** In the original design of the IP header shown in Fig 4.3, this field was referred to as type of service (TOS), which defined how the datagram should be handled. In the late 1990s, IETF redefined the field to provide differentiated services (DiffServb).

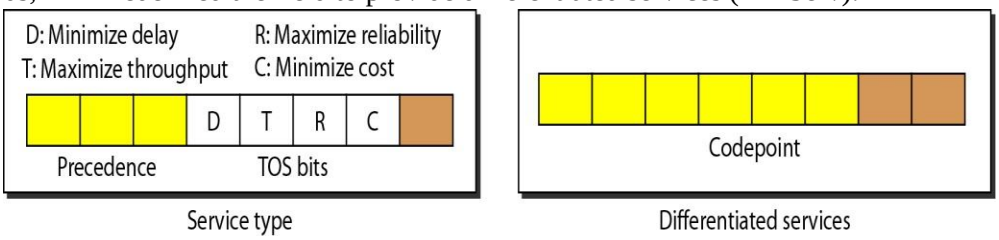


Figure 4.3: Service Type

Note: **The precedence subfield was part of version 4, but never used**

4. **Total Length:** This 16-bit field defines the total length (header plus data) of the IP datagram in bytes. A 16-bit number can define a total length of up to 65,535 (when all bits are 1s). However, the size of the datagram is normally much less than this. This field helps the receiving device to know when the packet has completely arrived. To find the length of the data coming from the upper layer, subtract the header length from the total length. The header length can be found by multiplying the value in the HLEN field by 4.

Note: The total length field defines the total length of the datagram including the header.

5. **Identification, Flags, and Fragmentation Offset:** These three fields are related to the fragmentation of the IP datagram when the size of the datagram is larger than the underlying network can carry.
6. **Time-to-live:** Due to some malfunctioning of routing protocols (discussed later) a datagram may be circulating in the Internet, visiting some networks over and over without reaching the destination. This may create extra traffic in the Internet. The time-to-live (TTL) field is used to control the maximum number of hops (routers) visited by the datagram. When a source host sends the datagram, it stores a number in this field. This value is approximately two times the maximum number of routers between any two hosts. Each router that processes the datagram decrements this number by one. If this value, after being decremented, is zero, the router discards the datagram.
7. **Protocol:** In TCP/IP, the data section of a packet, called the payload, carries the whole packet from another protocol. A datagram, for example, can carry a packet belonging to any transport ayer protocol such as UDP or TCP. A datagram can also carry a packet from other protocols that directly use the service of the IP, such as some routing protocols or some auxiliary protocols. The Internet authority has given any protocol that uses the service of IP a unique 8bit number which is inserted in the protocol field. When the payload is encapsulated in a datagram at the source IP, the corresponding protocol number is inserted in this field; when the datagram arrives at the destination, the value of this field helps to define to which protocol the payload should be delivered. In other words, this field provides multiplexing at the source and demultiplexing at the destination

Value	Protocol
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF

Table *Protocol values*

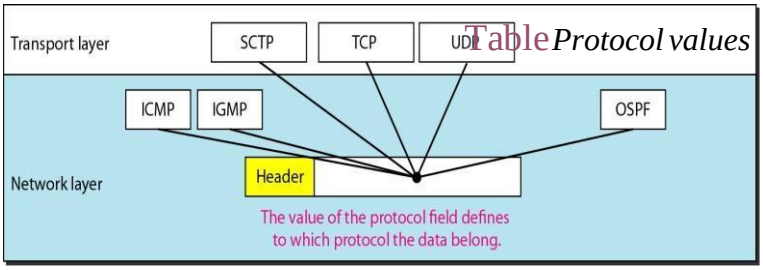


Figure 4.4: Multiplexing and demultiplexing using the value of the protocol field

8. **Header checksum**: IP is not a reliable protocol; it does not check whether the payload carried by a datagram is corrupted during the transmission. IP puts the burden of error checking of the payload on the protocol that owns the payload, such as UDP or TCP.
   The datagram header, however, is added by IP, and its error-checking is the responsibility of IP. Errors in the IP header can be a disaster. For example, if the destination IP address is corrupted, the packet can be delivered to the wrong host. If the protocol field is corrupted, the payload may be delivered to the wrong protocol. If the fields related to the fragmentation are corrupted, the datagram cannot be reassembled correctly at the destination, and so on. For these reasons, IP adds a header checksum field to check the header, but not the payload. We need to remember that, since the value of some fields, such as TTL, which are related to fragmentation and options, may change from router to router, the checksum needs to be recalculated at each router. Checksum in the Internet normally uses a 16-bit field, which is the complement of the sum of other fields calculated using 1s complement arithmetic.

9. **Source and Destination Addresses:** These 32-bit source and destination address fields define the IP address of the source and destination respectively. The source host should know its IP address. The destination IP address is either known by the protocol that uses the service of IP or is provided by the DNS. Note that the value of these fields must remain unchanged during the time the IP datagram travels from the source host to the destination host.
10. **Options:** A datagram header can have up to 40 bytes of options. Options can be used for network testing and debugging. Although options are not a required part of the IP header, option processing is required of the IP software. This means that all implementations must be able to handle options if they are present in the header. The existence of options in a header creates some burden on the datagram handling; some options can be changed by routers, which forces each router to recalculate the header checksum. There are one-byte and multi-byte options
11. **Payload:** Payload, or data, is the main reason for creating a datagram. Payload is the packet coming from other protocols that use the service of IP. Comparing a datagram to a postal package, payload is the content of the package; the header is only the information written on the package ***Example 1:***

An IPv4 packet has arrived with the first 8 bits as shown: 01000010
The receiver discards the packet. Why?
Solution
There is an error in this packet. The 4 leftmost bits (0100) show the version, which is correct. The next 4 bits (0010) show an invalid header length (2 × 4 = 8). The minimum number of bytes in the header must be 20. The packet has been corrupted in transmission. Example 2

In an IPv4 packet, the value of HLEN is 1000 in binary. How many bytes of options are being carried by this packet?

Solution
The HLEN value is 8, which means the total number of bytes in the header is 8 × 4, or 32 bytes. The first 20 bytes are the base header, the next 12 bytes are the options. Example 3

Solution
The HLEN value is 5, which means the total number of bytes in the header is 5 × 4, or 20 bytes (no options). The total length is 40 bytes, which means the packet is carrying 20 bytes of data (40 − 20). Example 4

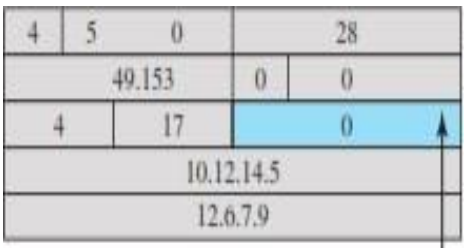
An IPv4 packet has arrived with the first few hexadecimal digits as shown.

0x45000028000100000102 . . .

How many hops can this packet travel before being dropped? The data belong to what upper-layer protocol?

Solution
To find the time-to-live field, we skip 8 bytes. The time-to-live field is the ninth byte, which is 01. This means the packet can travel only one hop. The protocol field is the next byte (02), which means that the upper-layer protocol is IGMP. Example 6

An IPv4 packet has arrived with the header decimal digits as shown below. Calculate the checksum for this header



Fragmentation

A datagram can travel through different networks. Each router decapsulates the IP datagram from the frame it receives, processes it, and then encapsulates it in another frame. The format and size of the received(or sent) frames depend on the protocol used by the physical network through which the frame has just travelled(or going to travel). For example, if a router connects a LAN to a WAN, it receives a frame in the LAN format and sends a frame in the WAN format.

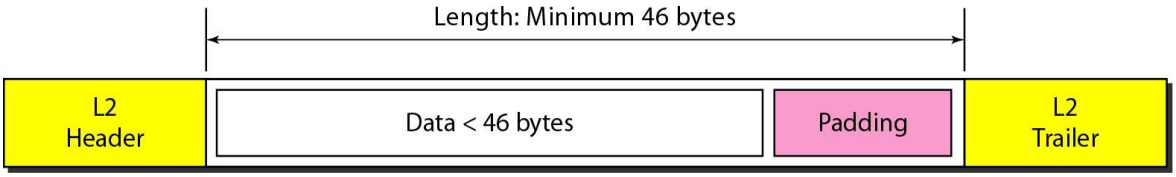
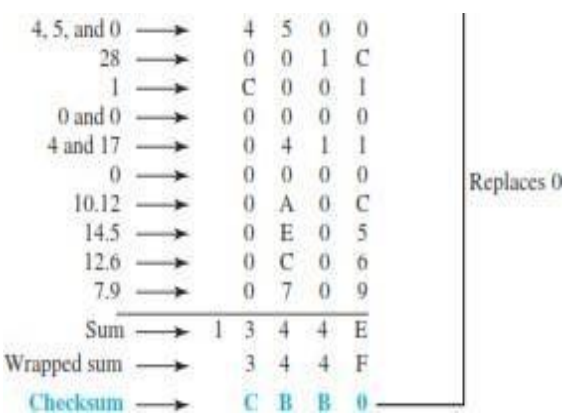


Figure 4.5: Encapsulation of a small datagram in an Ethernet frame



Maximum transfer unit (MTU)

Each link-layer protocol has its own frame format. One of the features of each format is the maximum size of the payload that can be encapsulated. In other words, when a datagram is encapsulated in a frame, the total size of the datagram must be less than this maximum size, which is defined by the restrictions imposed by the hardware and software used in the network

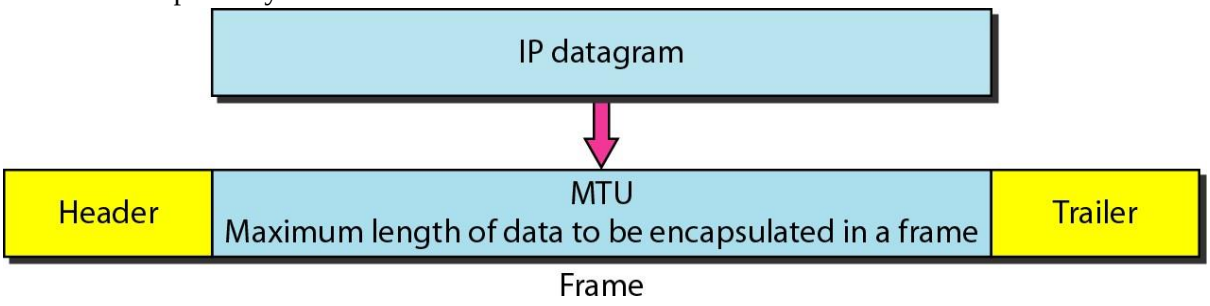


Figure 4.6: Maximum transfer unit (MTU)

In order to make the IP protocol independent of the physical network, the designers decided to make the maximum length of the IP datagram equal to 65,535 bytes. This makes transmission more efficient if one day we use a link-layer protocol with an MTU of this size. However, for other physical networks, we must divide the datagram to make it possible for it to pass through these networks. This is called fragmentation. When a datagram is fragmented, each fragment has its own header with most of the fields repeated, but some have been changed.

A fragmented datagram may itself be fragmented if it encounters a network with an even smaller MTU. In other words, a datagram may be fragmented several times before it reaches the final destination. A datagram can be fragmented by the source host or any router in the path. The reassembly of the datagram, however, is done only by the destination host, because each fragment becomes an independent datagram

Table 4.1 MTUs for some networks

Protocol	*MTU*
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296

Fields Related to Fragmentation Three fields in an IP datagram are related to fragmentation: identification, flags, and fragmentation offset

When a datagram is fragmented, the value in the identification field is copied into all fragments. In other words, all fragments have the same identification number, which is also the same as the original datagram. The identification number helps the destination in reassembling the datagram. It knows that all fragments having the same identification value should be assembled into one datagram

Flags used in fragmentation

When the payload of the IP datagram is fragmented, most parts of the header, with the exception of some options, must be copied by all fragments. The host or router that fragments a datagram must change the values of three fields: flags, fragmentation offset, and total length. The value of the checksum must be recalculated regardless of fragmentation.

Figure 4.7 Flags in fragmentation

The leftmost bit is reserved (not used). The second bit (D bit) is called the do not fragment bit. If its value is 1, the machine must not fragment the datagram. If it cannot pass the datagram through any available physical network, it discards the datagram and sends an ICMP error message to the source host (discussed later). If its value is 0, the datagram can be fragmented if necessary. The third bit (M bit) is called the more fragment bit. If its value is 1, it means the datagram is not the last fragment; there are more fragments after this one. If its value is 0, it means this is the last or only fragment.

Fragmentation offset

The 13-bit fragmentation offset field shows the relative position of this fragment with respect to the whole datagram. It is the offset of the data in the original datagram measured in units of 8 bytes.

Fragmentation example

Figure 4.8 shows a datagram with a data size of 4000 bytes fragmented into three fragments. The bytes in the original datagram are numbered 0 to 3999. The first fragment carries bytes 0 to 1399. The offset for this datagram is 0/8 = 0. The second fragment carries bytes 1400 to 2799; the offset value for this fragment is 1400/8 = 175. Finally, the third fragment carries bytes 2800 to 3999. The offset value for this fragment is 2800/8 = 350.

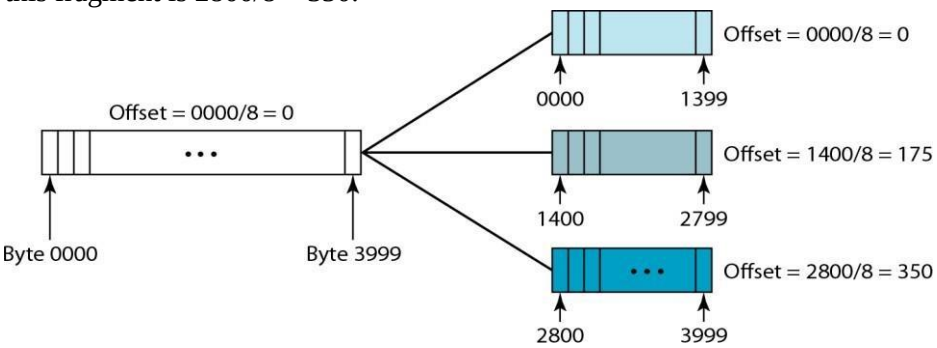


Figure 4.8 A fragmentation example

Detailed fragmentation example

The value of the offset is measured in units of 8 bytes. This is done because the length of the offset field is only 13 bits long and cannot represent a sequence of bytes greater than 8191. This force hosts or routers that fragment datagrams to choose the size of each fragment so that the first byte number is divisible by 8. Figure 4.9. shows an expanded view of the fragments in the previous figure. The original packet starts at the client; the fragments are reassembled at the server. The value of the identification field is the same in all fragments, as is the value of the flags field with the more bit set for all fragments except the last. Also, the value of the offset field for each fragment is shown. Note that although the fragments arrived out of order at the destination, they can be correctly reassembled.

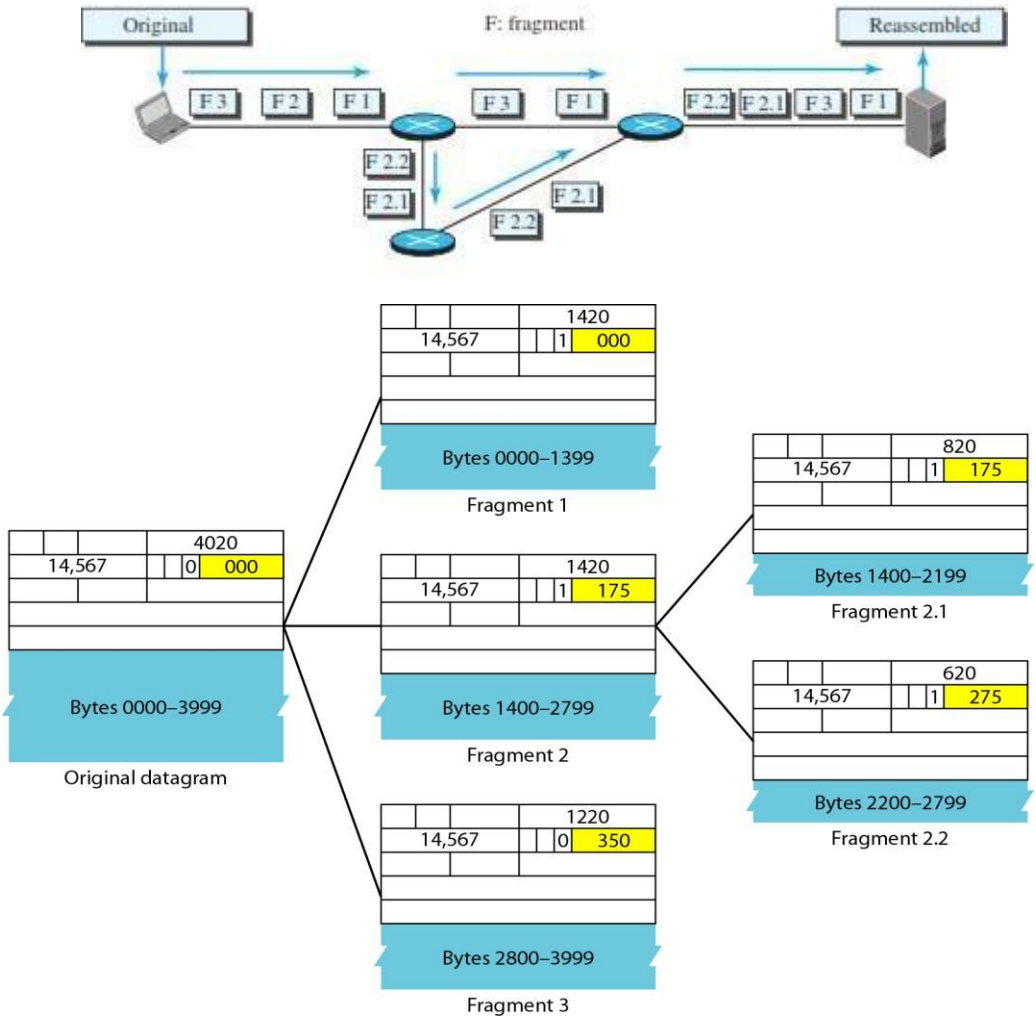


Figure 4.9 Detailed fragmentation example - Expanded view

Example

A packet has arrived with an M bit value of 0. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Solution: If the M bit is 0, it means that there are no more fragments; the fragment is the last one. However, we cannot say if the original packet was fragmented or not. A non-fragmented packet is considered the last fragment.

Example: A packet has arrived with an M bit value of 1. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Solution - If the M bit is 1, it means that there is at least one more fragment. This fragment can be the first one or a middle one, but not the last one. We don't know if it is the first one or a middle one; we need more information (the value of the fragmentation offset).

Example: A packet has arrived with an M bit value of 1 and a fragmentation offset value of 0. Is this the first fragment, the last fragment, or a middle fragment?

Solution - Because the M bit is 1, it is either the first fragment or a middle one. Because the offset value is 0, it is the first fragment.

Example A packet has arrived in which the offset value is 100. What is the number of the first byte? Do we know the number of the last byte?

Solution - To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 800. We cannot determine the number of the last byte unless we know the length of the data.

Example - A packet has arrived in which the offset value is 100, the value of HLEN is 5, and the value of the total length field is 100. What are the numbers of the first byte and the last byte?

Solution The first byte number is 100 × 8 = 800. The total length is 100 bytes, and the header length is 20 bytes (5 × 4), which means that there are 80 bytes in this datagram. If the first byte number is 800, the last byte number must be 879.

Options

The header of the IPv4 datagram is made of two parts: a fixed part and a variable part. The fixed part is 20 bytes long and was discussed in the previous section. The variable part comprises the options that can be a maximum of 40 bytes (in multiples of 4-bytes) to preserve the boundary of the header. Options, as the name implies, are not required for a datagram. They can be used for network testing and debugging.

Taxonomy of options in IPv4

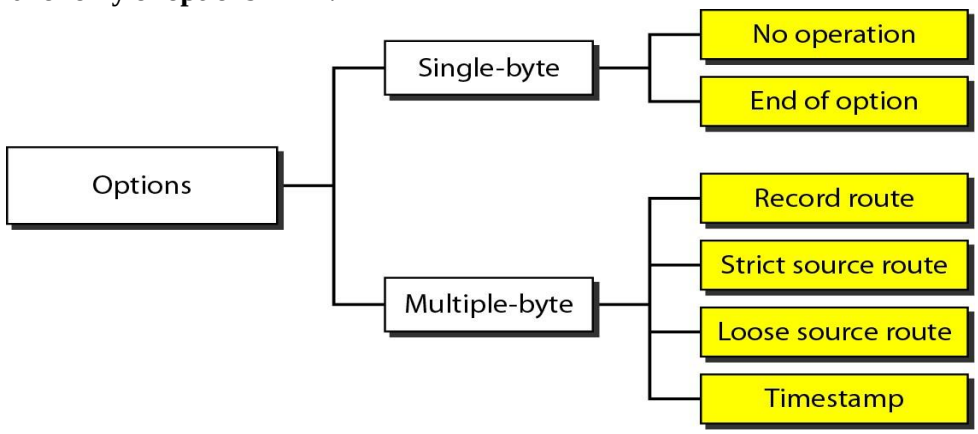


Figure 4.10: Classification of Options

I.Single-Byte Options: There are two single-byte options. No Operation A no-operation option is a 1byte option used as a filler between options. End of Option An end-of-option option is a 1-byte option used for padding at the end of the option field. It, however, can only be used as the last option.

II.Multliple-Byte Options: There are four multiple-byte options.

a) **Record Route** A record route option is used to record the Internet routers that handle the datagram. It can list up to nine router addresses. It can be used for debugging and management purposes.
b) **Strict Source Route** A strict source route option is used by the source to predetermine a route for the datagram as it travels through the Internet. Here, the sender can choose a route with a specific type of service, such as minimum delay or maximum throughput. Alternatively, it may choose a route that is safer or more reliable for the sender's purpose. For example, a sender can choose a route so that its datagram does not travel through a competitor's network. If a datagram specifies a strict source route, all the routers defined in the option must be visited by the datagram
c) **Loose Source Route**: A loose source route option is similar to the strict source route, but it is less rigid. Each router in the list must be visited, but the datagram can visit other routers as well.
d) **Timestamp**: A timestamp option is used to record the time of datagram processing by a router. The time is expressed in milliseconds from midnight, Universal time or Greenwich mean time.

Knowing the time a datagram is processed can help users and managers track the behaviour of the routers in the Internet

Security of IPv4 Datagrams

The IPv4 protocol, as well as the whole Internet, was started when the Internet users trusted each other. No security was provided for the IPv4 protocol. Today, however, the situation is different; the Internet is not secure anymore. There are three security issues that are particularly applicable to the IP protocol: packet sniffing, packet modification, and IP spoofing.

Packet Sniffing: An intruder may intercept an IP packet and make a copy of it. Packet sniffing is a passive attack, in which the attacker does not change the contents of the packet. This type of attack is very difficult to detect because the sender and the receiver may never know that the packet has been copied. Although packet sniffing cannot be stopped, encryption of the packet can make the attacker's effort useless. The attacker may still sniff the packet, but the content is not detectable.

Packet Modification: The second type of attack is to modify the packet. The attacker intercepts the packet, changes its contents, and sends the new packet to the receiver. The receiver believes that the packet is coming from the original sender. This type of attack can be detected using a data integrity mechanism. The receiver, before opening and using the contents of the message, can use this mechanism to make sure that the packet has not been changed during the transmission

IP Spoofing: An attacker can masquerade as somebody else and create an IP packet that carries the source address of another computer. An attacker can send an IP packet to a bank pretending that it is coming from one of the customers

IPsec The IP packets today can be protected from the previously mentioned attacks using a protocol called IPsec (IP Security).

ICMPv4

The IPv4 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 manager needs information from another host or router.

The Internet Control Message Protocol version 4 (ICMPv4) has been designed to compensate for the above two deficiencies. It is a companion to the IP protocol. ICMP itself is a network-layer protocol. However, its messages are not passed directly to the data-link layer as would be expected. Instead, the messages are first encapsulated inside IP datagrams before going to the lower layer. When an IP datagram encapsulates an ICMP message, the value of the protocol field in the IP datagram is set to 1 to indicate that the IP payroll is an ICMP message.

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. For example, nodes can discover their neighbours

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 4. 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 (to be discussed later in the chapter). The rest of the header is specific for each message type.

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 query.

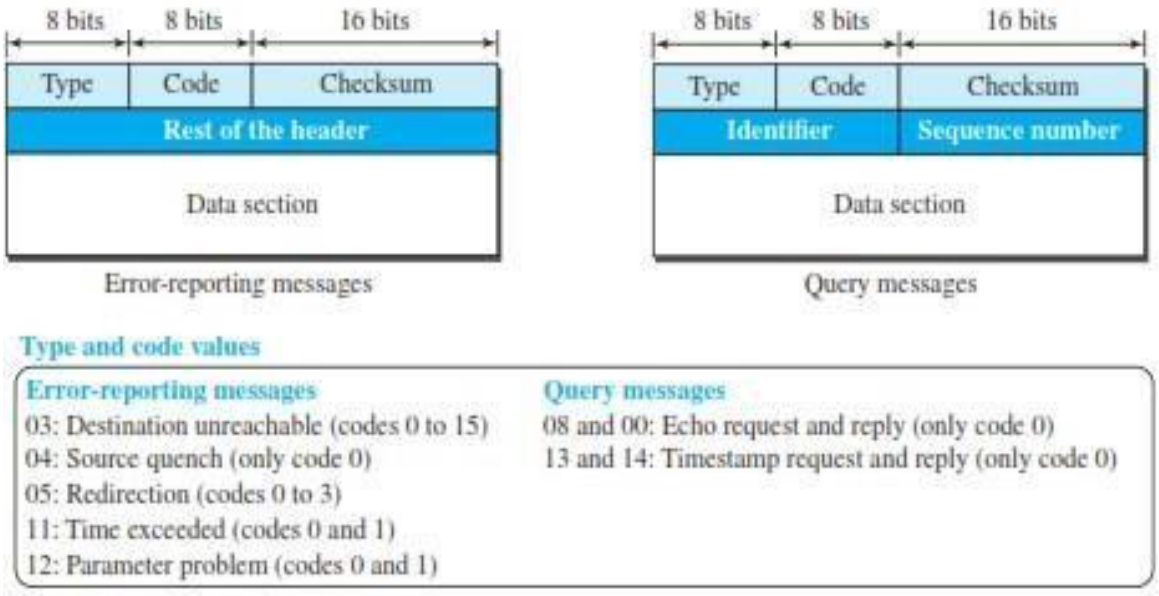


Figure 4.11 - General format of ICMP messages

Error Reporting Messages

Since IP is an unreliable protocol, one of the main responsibilities of ICMP is to report some errors that may occur during the processing of the IP datagram. ICMP does not correct errors, it simply reports them. Error correction is left to the higher-level protocols. Error messages are always sent to the original source because the only information available in the datagram about the route is the source and destination IP addresses. ICMP uses the source IP address to send the error message to the source (originator) of the datagram. To make the error-reporting process simple, ICMP follows some rules in reporting messages.

1. No error message will be generated for a datagram having a multicast address or special address (such as this host or loopback).
2. No ICMP error message will be generated in response to a datagram carrying an ICMP error message.
3. No ICMP error message will be generated for a fragmented datagram that is not the first fragment.

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. The 8 bytes of data are included because the first 8 bytes provide information about the port numbers (UDP and TCP) and sequence number (TCP). This information is needed so the source can inform the protocols (TCP or UDP) about the error.

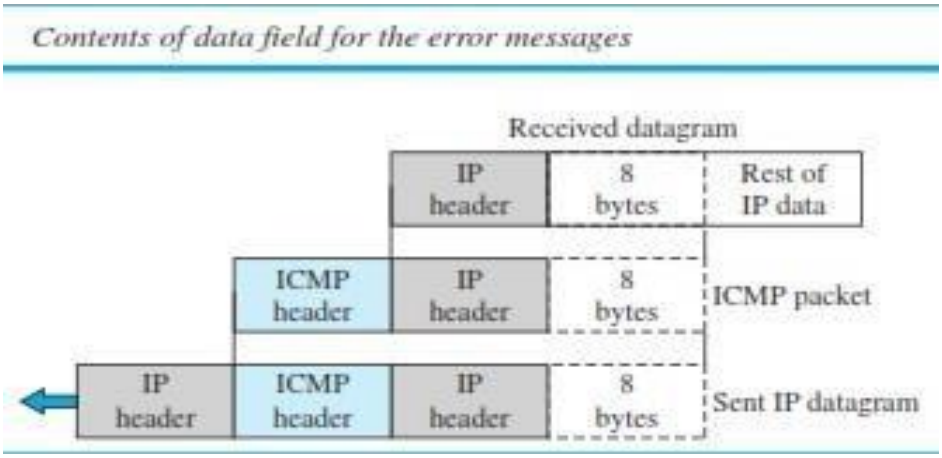
The following are important points about ICMP error messages:

❑ 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 ICMP 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.



Error Message Types

- **Destination Unreachable** - The most widely used error message is the destination unreachable (type 3). This message uses different codes (0 to 15) to define the type of error message and the reason why a datagram has not reached its final destination
- **Source Quench** - Another error message is called the source quench (type 4) message, which informs the sender that the network has encountered congestion and the datagram has been dropped; the source needs to slow down sending more datagrams. In other words, ICMP adds a kind of congestion control mechanism to the IP protocol by using this type of message.
- **Redirection Message** - The redirection message (type 5) is used when the source uses a wrong router to send out its message. The router redirects the message to the appropriate router, but informs the source that it needs to change its default router in the future. The IP address of the default router is sent in the message.
- **Parameter Problem** - A parameter problem message (type 12) can be sent when either there is a problem in the header of a datagram (code 0) or some options are missing or cannot be interpreted (code 1).
- **Query Messages** - Query messages are used to probe or test the liveliness of hosts or routers in the Internet, find the one-way or the round-trip time for an IP datagram between two devices, or even find out whether the clocks in two devices are synchronized. Query messages in ICMP can be used independently without relation to an IP datagram. But a query message needs to be encapsulated in a datagram, as a carrier. Naturally, query messages come in pairs: request and reply. The echo request (type 8) and the echo reply (type 0) pair of messages are used by a host or a router to test the liveliness of another host or router

Deprecated Messages

Three pairs of messages are declared obsolete by IETF:

1.Information request and replay messages are not used today because their duties are done by the Address Resolution Protocol (ARP).

2.Address mask request and reply messages are not used today because their duties are done by the Dynamic Host Configuration Protocol (DHCP)

3.Router solicitation and advertisement messages are not used today because their duties are done by the Dynamic Host Configuration Protocol (DHCP)

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.

Ping - We can use the ping program to find if a host is alive and responding. We use ping here to see how it uses ICMP packets. The source host sends ICMP echo-request messages; the destination, if alive, responds with ICMP echo-reply messages. The ping program sets the identifier field in the echo-request and echo-reply message and starts the sequence number from 0; this number is incremented by 1 each time a new message is sent. 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).

Example: The following shows how we send a ping message to the auniversity.edu site. We set the identifier field in the echo request and reply message and start the sequence number from 0; this number is incremented by one each time a new message is sent. 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 auniversity.edu PING auniversity.edu (152.181.8.3) 56 (84) bytes of data.
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=0 ttl=62 time=1.91 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=1 ttl=62 time=2.04 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=2 ttl=62 time=1.90 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=3 ttl=62 time=1.97 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=4 ttl=62 time=1.93 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=5 ttl=62 time=2.00ms
--- auniversity.edu statistics -- 6 packets transmitted, 6 received, 0% packet loss rtt min/avg/max = 1.90/1.95/2.04 ms

Traceroute or Tracert - The traceroute program in UNIX or tracert in Windows can be used to trace the path of a packet from a source to the destination. It can find the IP addresses of all the routers that are visited along the path. The program is usually set to check for the maximum of 30 hops (routers) to be visited. The number of hops in the Internet is normally less than this.

Traceroute - The traceroute program is different from the ping program. The ping program gets help from two query messages; the traceroute program gets help from two error-reporting messages: time exceeded and destination-unreachable. The traceroute is an application layer program, but only the client program is needed, because, as we can see, the client program never reaches the application layer in the destination host. In other words, there is no traceroute server program. The traceroute application program is encapsulated in a UDP user datagram, but traceroute intentionally uses a port number that is not available at the destination. If there are n routers in the path, the traceroute program sends (n + 1) messages. The first n messages are discarded by the n routers, one by each router; the last message is discarded by the destination host. The traceroute client program uses the (n + 1) ICMP error-reporting messages received to find the path between the routers.

The first traceroute message is sent with time-to-live (TTL) value set to 1; the message is discarded at the first router and a time-exceeded ICMP error message is sent, from which the traceroute program can find the IP address of the first router (the source IP address of the error message) and the router name (in the data section of the message). The second traceroute message is sent with TTL set to 2, which can find the IP address and the name of the second router. Similarly, the third message can find the information about router 3. The fourth message, however, reaches the destination host.

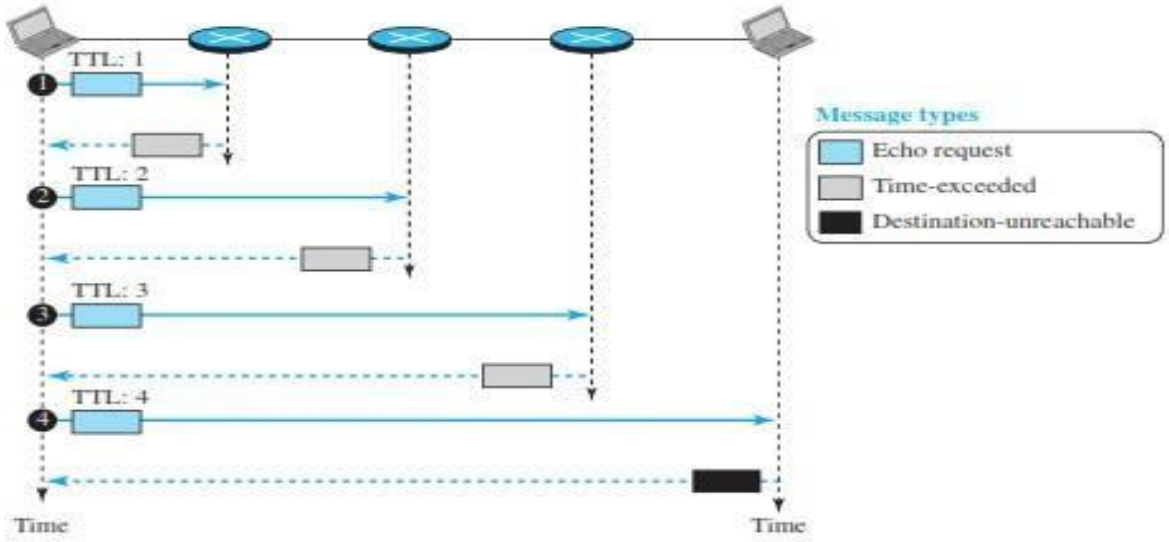


Figure 4.12: Use of ICMPv4 in traceroute

The traceroute program also sets a timer to find the round-trip time for each router and the destination. Most traceroute programs send three messages to each device, with the same TTL value, to be able to find a better estimate for the round-trip time. The following shows an example of a traceroute program, which uses three probes for each device and gets three RTTs.

$ traceroute printers.com traceroute to printers.com (13.1.69.93), 30 hops max, 38-byte packets
1 route.front.edu (153.18.31.254) 0.622ms 0.891ms 0.875ms

2 ceneric.net (137.164.32.140) 3.069ms 2.875ms 2.930ms

3 satire.net (132.16.132.20) 3.071ms 2.876ms 2.929ms

4 alpha.printers.com (13.1.69.93) 5.922ms 5.048ms 4.922ms

Tracert - The tracert program in windows behaves differently. The tracert messages are encapsulated directly in IP datagrams. The tracert, like traceroute, sends echo-request messages. However, when the last echo request reaches the destination host, an echo replay message is issued

ICMP Checksum

In ICMP the checksum is calculated over the entire message (header and data).

Example : Figure 4.13 below shows an example of checksum calculation for a simple echo-request message. We randomly chose the identifier to be 1 and the sequence number to be 9. The message is divided into 16-bit (2-byte) words. The words are added and the sum is complemented. Now the sender can put this value in the checksum field.

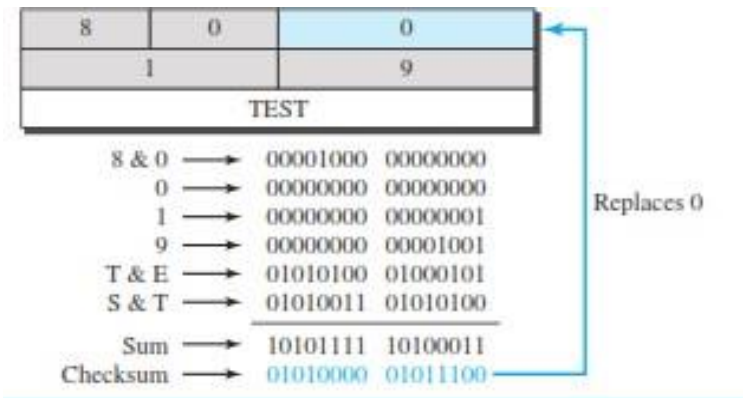


Figure 4.13 – Calculation of ICMP checksum

MOBILE IP

As mobile and personal computers such as notebooks become increasingly popular, we need to think about mobile IP, the extension of IP protocol that allows mobile computers to be connected to the Internet at any location where the connection is possible.

Addressing

The main problem that must be solved in providing mobile communication using the IP protocol is addressing

Stationary Hosts - The original IP addressing was based on the assumption that a host is stationary, attached to one specific network. A router uses an IP address to route an IP datagram. An IP address has two parts: a prefix and a suffix. The prefix associates a host with a network. For example, the IP address 10.3.4.24/8 defines a host attached to the network 10.0.0.0/8.

The IP addresses are designed to work with stationary hosts because part of the address defines the network to which the host is attached.

Mobile Hosts - When a host moves from one network to another, the IP addressing structure needs to be modified. Several solutions have been proposed.

1. **Changing the Address** - One simple solution is to let the mobile host change its address as it goes to the new network. The host can use DHCP to obtain a new address to associate it with the new network. This approach has several drawbacks. First, the configuration files would need to be changed. Second, each time the computer moves from one network to another, it must be rebooted. Third, the DNS tables need to be revised so that every other host in the Internet is aware of the change. Fourth, if the host roams from one network to another during a transmission, the data exchange will be interrupted. This is because the ports and IP addresses of the client and the server must remain constant for the duration of the connection.
2. **Two Addresses** - The approach that is more feasible is the use of two addresses. The host has its original address, called the home address, and a temporary address, called the care-of address. The home address is permanent; it associates the host with its home network, the network that is the permanent home of the host. The care-of address is temporary. When a host moves from one network to another, the care-of address changes; it is associated with the foreign network, the network to which the host moves. When a mobile host visits a foreign network, it receives its care-of address during the agent discovery and registration phase

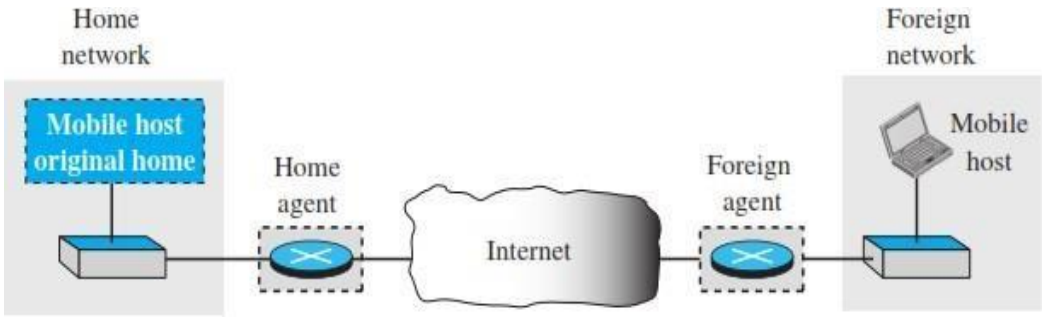


Figure 4.14: Home address and care-of address

Agents

To make the change of address transparent to the rest of the Internet requires a home agent and a foreign agent. Figure 4.14 shows the position of a home agent relative to the home network and a foreign agent relative to the foreign network. We have shown the home and the foreign agents as routers, but we need to emphasize that their specific function as an agent is performed in the application layer. In other words, they are both routers and hosts.

Home Agent - The home agent is usually a router attached to the home network of the mobile host. The home agent acts on behalf of the mobile host when a remote host sends a packet to the mobile host. The home agent receives the packet and sends it to the foreign agent.

Foreign Agent - The foreign agent is usually a router attached to the foreign network. The foreign agent receives and delivers packets sent by the home agent to the mobile host. The mobile host can also act as a foreign agent. In other words, the mobile host and the foreign agent can be the same. However, to do this, a mobile host must be able to receive a care-of address by itself, which can be done through the use of DHCP. In addition, the mobile host needs the necessary software to allow it to communicate with the home agent and to have two addresses: its home address and its care-of address. This dual addressing must be transparent to the application programs. When the mobile host acts as a foreign agent, the careof address is called a collocated care-of address.

The advantage of using a collocated care-of address is that the mobile host can move to any network without worrying about the availability of a foreign agent. The disadvantage is that the mobile host needs extra software to act as its own foreign agent

Three Phases - To communicate with a remote host, a mobile host goes through three phases: agent discovery, registration, and data transfer, as shown in Figure 4.15.

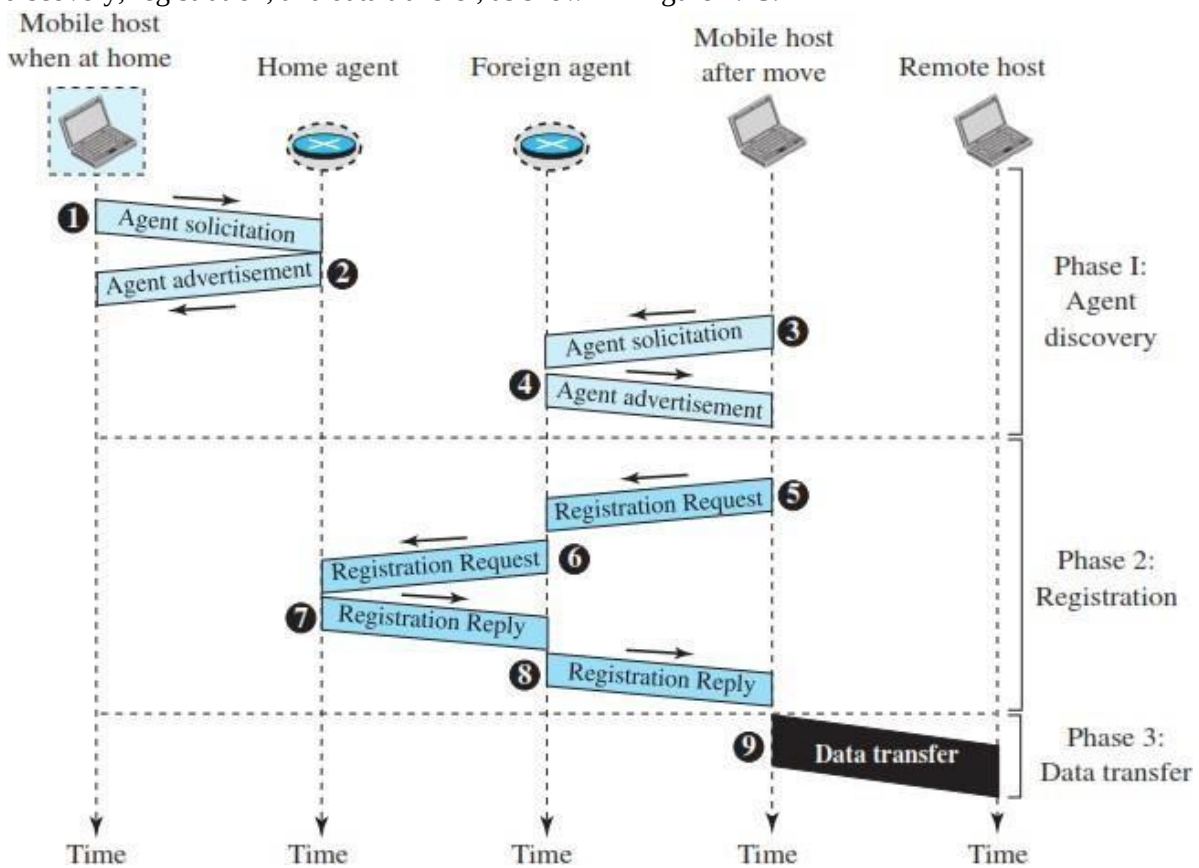


Figure 4.15 : Remote host and mobile host communication

The first phase, agent discovery, involves the mobile host, the foreign agent, and the home agent. The second phase, registration, also involves the mobile host and the two agents. Finally, in the third phase, the remote host is also involved. We discuss each phase separately.

Agent Discovery - The first phase in mobile communication, agent discovery, consists of two subphases. A mobile host must discover (learn the address of) a home agent before it leaves its home network. A mobile host must also discover a foreign agent after it has moved to a foreign network. This

discovery consists of learning the care-of address as well as the foreign agent's address. The discovery involves two types of messages: advertisement and solicitation

Agent Advertisement - When a router advertises its presence on a network using an ICMP router advertisement, it can append an agent advertisement to the packet if it acts as an agent. Mobile IP does not use a new packet type for agent advertisement; it uses the router advertisement packet of ICMP, and appends an agent advertisement message

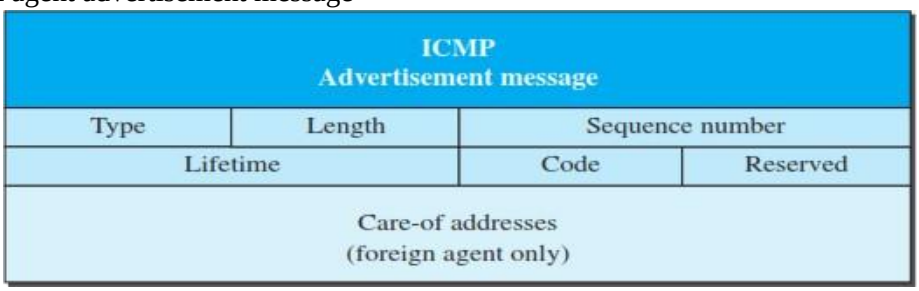


Figure 4.16: Agent advertisement

The field descriptions are as follows:

❑ **Type**. The 8-bit type field is set to 16.

❑**Length**. The 8-bit length field total length of the extension message (not the length of the ICMP advertisement message).

❑**Sequence number**. The 16-bit sequence number field holds the message number. The recipient can use the sequence number to determine if a message is lost.

❑**Lifetime.** The lifetime field defines the number of seconds that the agent will accept requests. If the value is a string of 1s, the lifetime is infinite.

❑**Code.** The code field is an 8-bit flag in which each bit is set (1) or unset (0). The meanings of the bits are shown in Table 4.2

Table 4.2 :Code bits

Bit	*Meaning*
0	Registration required. No collocated care-of address.
1	Agent is busy and does not accept registration at this moment.
2	Agent acts as a home agent.
3	Agent acts as a foreign agent.
4	Agent uses minimal encapsulation.
5	Agent uses generic routing encapsulation (GRE).
6	Agent supports header compression.
7	Unused (0).

❑**Care-of Addresses**. This field contains a list of addresses available for use as careof addresses. The mobile host can choose one of these addresses. The selection of this care-of address is announced in the registration request. Note that this field is used only by a foreign agent.

Agent Solicitation

When a mobile host has moved to a new network and has not received agent advertisements, it can initiate an agent solicitation. It can use the ICMP solicitation message to inform an agent that it needs assistance. Mobile IP does not use a new packet type for agent solicitation; it uses the router solicitation packet of ICMP.

Registration - The second phase in mobile communication is registration. After a mobile host has moved to a foreign network and discovered the foreign agent, it must register. There are four aspects of registration:

1. The mobile host must register itself with the foreign agent.

2.The mobile host must register itself with its home agent. This is normally done by the foreign agent on behalf of the mobile host.

3. The mobile host must renew registration if it has expired.

4.The mobile host must cancel its registration (deregistration) when it returns home. Request and Reply To register with the foreign agent and the home agent, the mobile host uses a registration request and a registration reply as shown in Figure 4.17.

Registration Request - A registration request is sent from the mobile host to the foreign agent to register its care-of address and also to announce its home address and home agent address. The foreign agent, after receiving and registering the request, relays the message to the home agent.

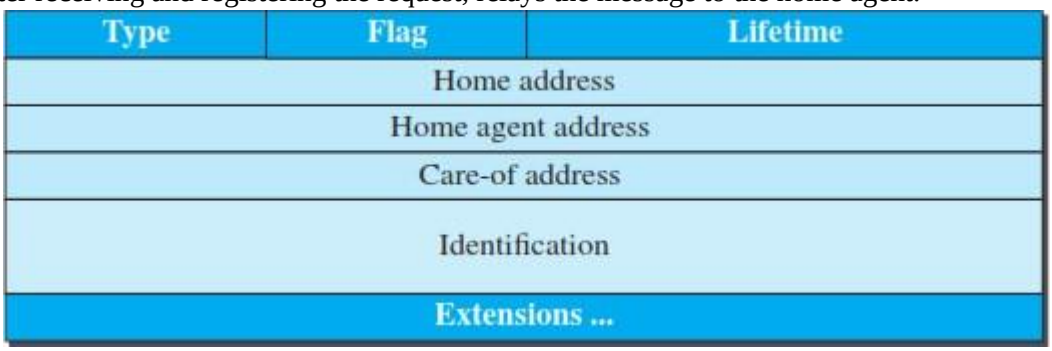


Figure 4.17: Registration request format The

field descriptions are as follows:

❑**Type.** The 8-bit type field defines the type of message. For a request message the value of this field is 1.

❑**Flag**. The 8-bit flag field defines forwarding information. The value of each bit can be set or unset. The meaning of each bit is given in Table 4.3.

Table 4.3: Registration request flag field bits

Bit	*Meaning*
0	Mobile host requests that home agent retain its prior care-of address.
1	Mobile host requests that home agent tunnel any broadcast message.
2	Mobile host is using collocated care-of address.
3	Mobile host requests that home agent use minimal encapsulation.
4	Mobile host requests generic routing encapsulation (GRE).
5	Mobile host requests header compression.
6–7	Reserved bits.

❑**Lifetime**. This field defines the number of seconds the registration is valid. If the field is a string of 0s, the request message is asking for deregistration. If the field is a string of 1s, the lifetime is infinite.

❑ **Home address**. This field contains the permanent (first) address of the mobile host.

❑ **Home agent address.** This field contains the address of the home agent.

❑**Care-of address**. This field is the temporary (second) address of the mobile host. ❑ Identification. This field contains a 64-bit number that is inserted into the request by the mobile host and repeated in the reply message. It matches a request with a reply.

❑**Extensions.** Variable length extensions are used for authentication. They allow a home agent to authenticate the mobile agent

Registration Reply - A registration reply is sent from the home agent to the foreign agent and then relayed to the mobile host. The reply confirms or denies the registration request. Figure 4.18 shows the format of the registration reply. The fields are similar to those of the registration request with the following exceptions. The value of the type field is 3. The code field replaces the flag field and shows the result of the registration request (acceptance or denial). The care-of address field is not needed.

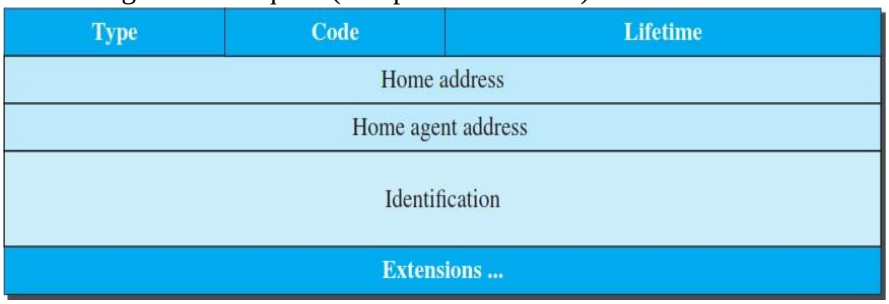


Figure 4.18: Registration reply format

Encapsulation - Registration messages are encapsulated in a UDP user datagram. An agent uses the well-known port 434; a mobile host uses an ephemeral port.

Data Transfer - After agent discovery and registration, a mobile host can communicate with a remote host. Figure 4.19 shows the idea.

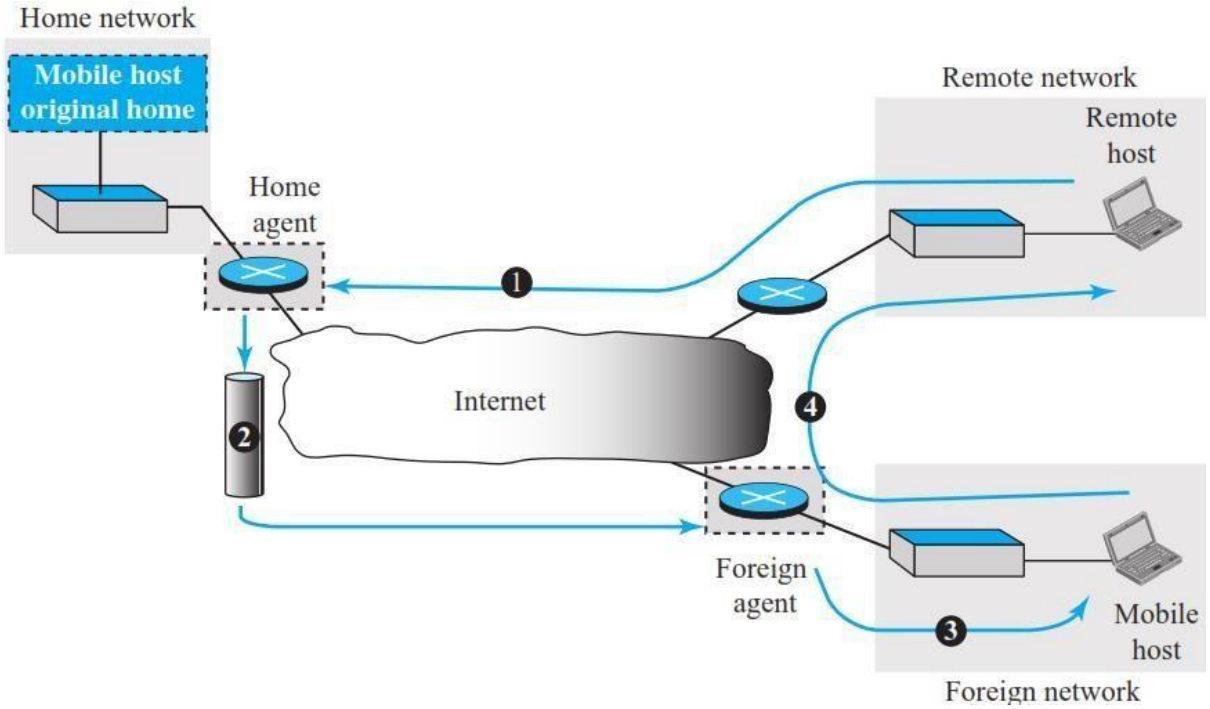


Figure 4.19: Data transfer

From Remote Host to Home Agent - When a remote host wants to send a packet to the mobile host, it uses its address as the source address and the home address of the mobile host as the destination address. In other words, the remote host sends a packet as though the mobile host is at its home network. The packet, however, is intercepted by the home agent, which pretends it is the mobile host. This is done using the proxy ARP technique. Path 1 of Figure 4.19 shows this step.

From Home Agent to Foreign Agent - After receiving the packet, the home agent sends the packet to the foreign agent, using the tunnelling concept. The home agent encapsulates the whole IP packet inside another IP packet using its address as the source and the foreign agent's address as the destination. Path 2 of Figure 4.19 shows this step.

From Foreign Agent to Mobile Host When the foreign agent receives the packet, it removes the original packet. However, since the destination address is the home address of the mobile host, the foreign agent consults a registry table to find the care-of address of the mobile host. (Otherwise, the packet would just be sent back to the home network.) The packet is then sent to the care-of address. Path 3 of Figure 4.19. shows this step.

From Mobile Host to Remote Host When a mobile host wants to send a packet to a remote host (for example, a response to the packet it has received), it sends as it does normally. The mobile host prepares a packet with its home address as the source, and the address of the remote host as the destination. Although the packet comes from the foreign network, it has the home address of the mobile host. Path 4 of Figure 4.19. shows this step.

Transparency - In this data transfer process, the remote host is unaware of any movement by the mobile host. The remote host sends packets using the home address of the mobile host as the destination address; it receives packets that have the home address of the mobile host as the source address. The movement is totally transparent. The rest of the Internet is not aware of the movement of the mobile host.

Inefficiency in Mobile IP

Communication involving mobile IP can be inefficient. The inefficiency can be severe or moderate. The severe case is called double crossing or 2X. The moderate case is called triangle routing or dog-leg routing.

Double Crossing - Double crossing occurs when a remote host communicates with a mobile host that has moved to the same network (or site) as the remote host (see Figure 4.20). When the mobile host sends a packet to the remote host, there is no inefficiency; the communication is local. However, when the remote host sends a packet to the mobile host, the packet crosses the Internet twice. Since a computer usually communicates with other local computers (principle of locality), the inefficiency from double crossing is significant.

Triangle Routing - Triangle routing, the less severe case, occurs when the remote host communicates with a mobile host that is not attached to the same network (or site) as the mobile host. When the mobile host sends a packet to the remote host, there is no inefficiency

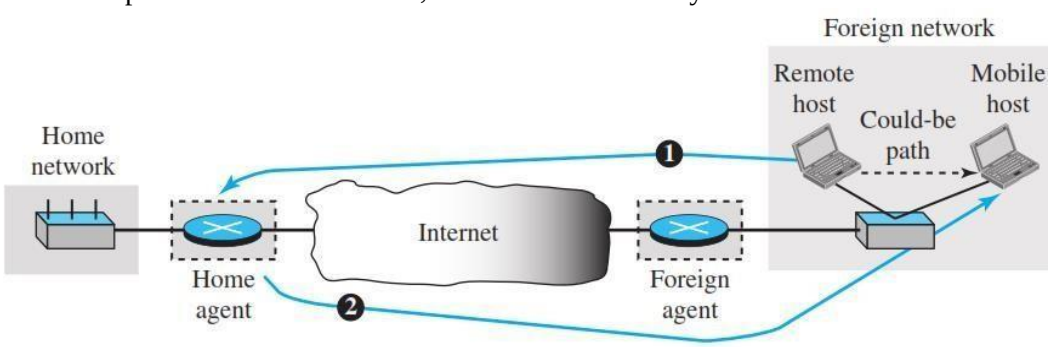


Figure 4. 20: Double crossing

However, when the remote host sends a packet to the mobile host, the packet goes from the remote host to the home agent and then to the mobile host. The packet travels the two sides of a triangle, instead of just one side

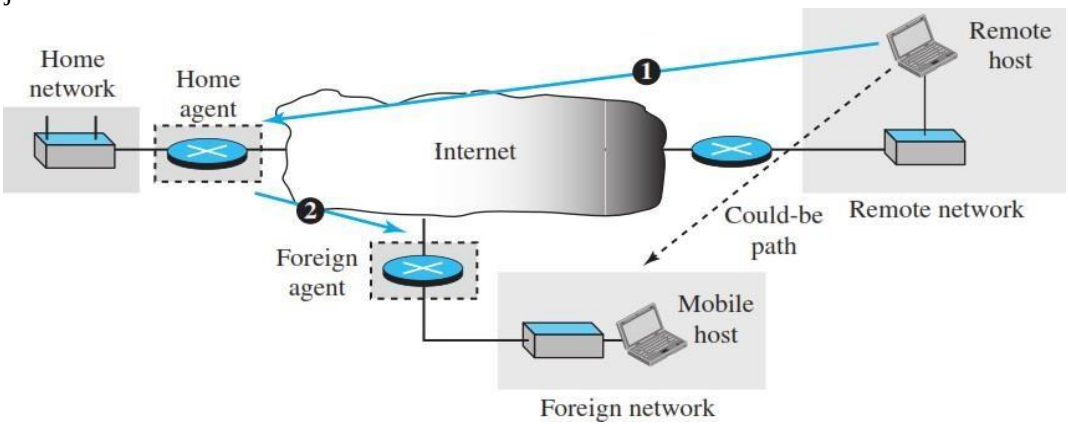


Figure 4.21: Triangle routing

Solution - One solution to inefficiency is for the remote host to bind the care-of address to the home address of a mobile host. For example, when a home agent receives the first packet for a mobile host, it forwards the packet to the foreign agent; it could also send an update binding packet to the remote host so that future packets to this host could be sent to the care-of address. The remote host can keep this information in a cache. The problem with this strategy is that the cache entry becomes outdated once the mobile host moves. In this case the home agent needs to send a warning packet to the remote host to inform it of the change.

Network Layer Protocols and Unicast Routing

Unicast routing

Unicast routing in the Internet, with a large number of routers and a huge number of hosts, can be done only by using hierarchical routing: routing in several steps using different routing algorithms.

In this section, we first discuss the general concept of unicast routing in an internet. After the routing concepts and algorithms are understood, we show how we can apply them to the Internet.

General Idea

In unicast routing, a packet is routed, hop by hop, from its source to its destination by the help of forwarding tables. The source host needs no forwarding table because it delivers its packet to the default router in its local network. The destination host needs no forwarding table either because it receives the packet from its default router in its local network. This means that only the routers that glue together the networks in the internet need forwarding tables. So, routing a packet from its source to its destination means routing the packetfrom a sourcerouter to a destinationrouter

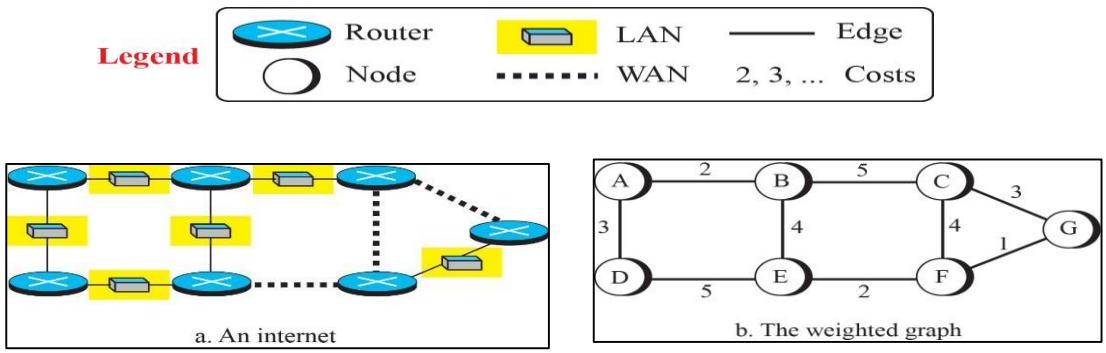


Figure4.1: Aninternet anditsgraphicalrepresentation

An Internet as a Graph

To find the best route, an internet can be modelled as a graph. A graph in computer science is a set of nodes and edges(lines)that connect the nodes. To model an internet as a graph, we can think of each router as a node and each network between a pair of routers as an edge. An internet is, in fact, modelled as a weighted graph, in which each edge is associated with a cost. If a weighted graph is used to represent a geographical area, the nodes can be cities and the edges can be roads connecting the cities; the weights, in this case, are distances between cities. In routing, however, the cost of an edge has a different interpretation in different routing protocols, which we discuss in a later section. For the moment, we assume that there is a cost associated with each edge. If there is no edge between the nodes, the cost is infinity. Figure4.1showshowaninternet can be modelled as a graph.

Least-Cost Routing

When an internet is modelled as a weighted graph, one of the ways to interpret the best route from the source router to the destination router is to find the least cost between the two. In other words, the source router chooses a route to the destination router in such a way that the total cost for the route is the least cost among all possible routes. In Figure 4.1, the best route between A and E is A-B-E, with the cost of 6. This means that each router needs to find the least-cost route between itself and all the other routers to be able to route a packet using this criterion.

Least-Cost Trees

If there are N routers in an internet, there are (N −1) least-cost paths from each router to any other router. This means we need N × (N − 1) least-cost paths for the whole internet. If we have only 10 routers in an internet, we need90least-costpaths. A better way to see all of these paths is to combine them in a least-cost tree. A least-cost tree is a tree with the source router as the root that spans the whole graph (visits all other nodes) and in which the path between the root and any other node is the shortest. In this way, we can have only one shortest-path tree for each node; we have N least-cost trees for the whole internet.

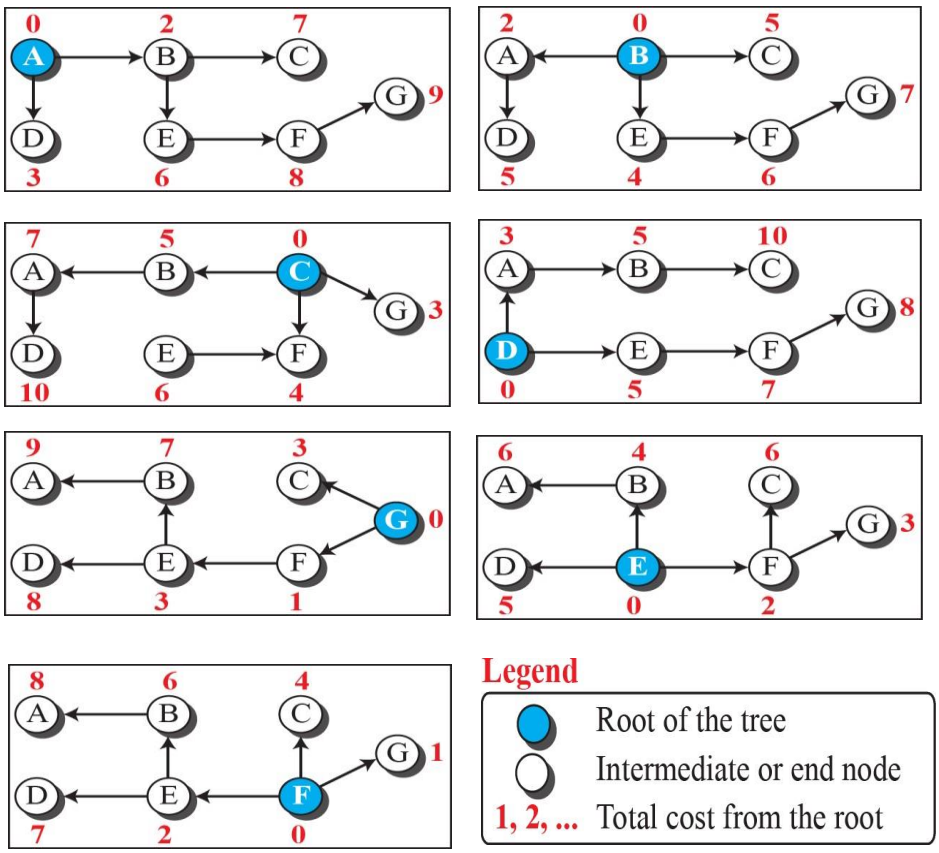


Figure4.2: Least-cost trees for nodes in the internetofFigure4.1

The least-cost trees for a weighted graph can have several properties if they are created using consistent criteria1.

1. The least-cost route from X to Yin X's tree is the inverse of the least-cost route from Y to Xin Y's tree; the cost in both directions is the same. For example, in Figure20.2, the route from A to Fin A's tree is (A→B→E→F), but the route from F to Ain F's tree is(F→E→B→A), which is the inverse of the first route. The cost is8 in each case.2.
2. Instead of travelling from X to Z using X's tree, we can travel from X to Y using X's tree and continue from Y to Z using Y's tree. For example, in Figure20.2, we can go from A to Gin A's tree using the route (A→B→E→F→G). We can also go from A to Ein A's tree (A→B→E) and then continue in E's tree using the route (E→F→G). The combination of the two routes

In the second case is the same route as in the first case. The cost in the first case is 9; the cost in the second case is also 9 (6 +3).

ROUTINGALGORITHMS

Several routing algorithms have been designed in the past. The differences between these methods are in the way they interpret the least cost and the way they create the least-cost tree for each node. In this section, we discuss the common algorithms; later we show how a routing protocol in the Internet implements one of these algorithms.

Distance-Vector Routing

The distance-vector (DV) routing uses the goal we discussed in the introduction, to find the best route. In distance-vector routing, the first thing each node creates is its own least-cost tree with the rudimentary information it has about its immediate neighbours. The incomplete trees are exchanged between immediate neighbours to make the trees more and more complete and to represent the whole internet. In distance-vector routing, a router continuously tells all of its neighbours what it knows about the whole internet (although the knowledge can be in complete).

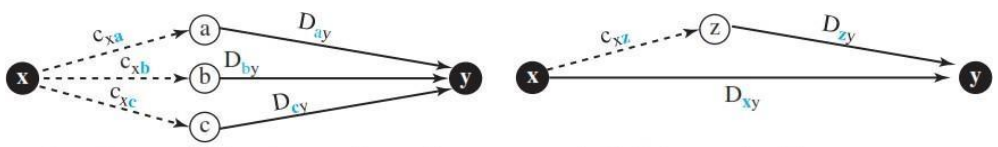
Bellman-Ford Equation

The heart of distance-vector routing is the famous Bellman-Ford equation. This equation is used to find the least cost (shortest distance) between a source node, x, and a destination node, y, through some intermediary nodes (a, b, c,) when the costs between the source and the intermediary nodes and the least costs between the intermediary nodes and the destination are given. The following shows the general case in which Dij is the shortest distance and cij is the cost between node si and j. Dxy=min{(cxa+Day),(cxb+Dby),(cxc+Dcy), …}

In distance-vector routing, normally we want to update an existing least cost with a least cost through an intermediary node, such as z, if the latter is shorter. In this case, the equation becomes simpler, as shown below:

Dxy = min {Dxy, (cxz +Dzy)}

Figure4.3showstheideagraphicallyfor both cases.



a. General case with three intermediate nodes b. Updating a path with a new route

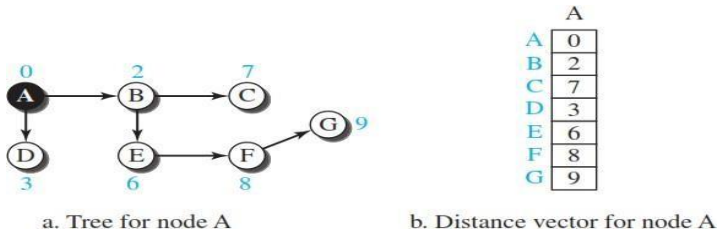
Figure 4.3 Graphical idea behind Bellman-Ford equation

We can say that the Bellman-Ford equation enables us to build a new least-cost path from previously established least-cost paths. In Figure 4.3, we can think of (a→ y), (b→ y), and (c→ y) as previously established least-cost paths and (x →y) as the new least-cost path

Distance Vectors

The concept of a distance vector is the rationale for the name distance-vector routing. A least-cost tree is a combination of least-cost paths from the root of the tree to all destinations. These paths are graphically glued together to form the tree. Distance-vector routing unglues these paths and creates a distance vector, a one-dimensional array to represent the tree. Figure 4.4shows the tree for node A

in the internet in Figure4.1andthe corresponding distance vector. Note that the name of the distance vector defines the root, the indexes define the destinations, and the value of each cell defines the least cost from the root to the destination. A distance vector does not give the path to the destinations as the least-cost tree does; it gives only the least costs to the destinations



a. Tree for node A b. Distance vector for node A

Figure 4.4: The distance vector corresponding to a tree

Each node in an internet, when it is booted, creates a very rudimentary distance vector with the minimum information the node can obtain from its neighbourhood. The node sends some greeting messages out of its interfaces and discovers the identity of the immediate neighbours and the distance between itself and each neighbour. It then makes a simple distance vector by inserting the discovered distances in the corresponding cells and leaves the value of other cells as infinity

Figure 4.5 shows all distance vectors for our internet. However, we need to mention that these vectors are made a synchronously, when the corresponding node has been booted; the existence of all of them in a figure does not mean synchronous creation of them

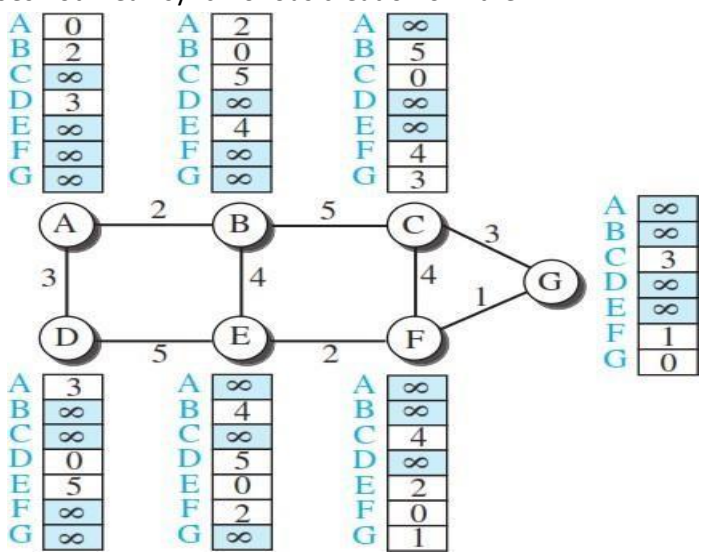
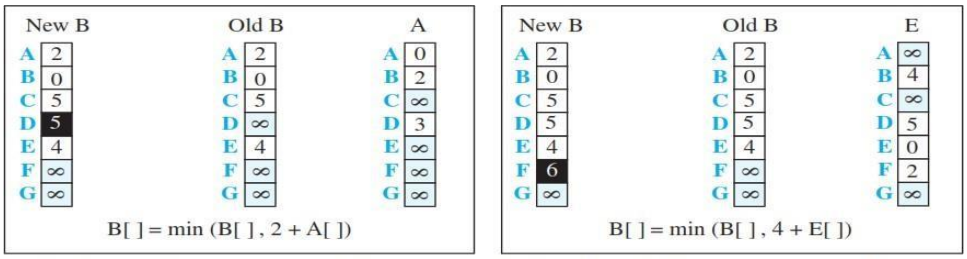


Figure 4.5: The first distance vector for an internet

The sedimentary vectors cannot help the internet to effectively forward a packet. For example, node A thinks that it is not connected to node G because the corresponding cell shows the least cost of infinity. To improve these vectors, the nodes in the internet need to help each other by exchanging information. After each node has created its vector, it sends a copy of the vector to all its immediate neighbours. After a node receives a distance vector from a neighbour, it updates its distance vector using the Bellman-Ford equation (second case). However, we need to understand that we need to update, not only one least cost, but N of them in which N is the number of the nodes in the internet. If we are using a program, we can do this using a loop; if we are showing the concept on paper, we can show

the whole vector instead of the N separate equations. We show the whole vector instead of seven equations for each update in Figure4.6.



a. First event: B receives a copy of A's vector.

b. Second event: B receives a copy of E's vector.

Note:
X[ ]: the whole vector

Figure 4.6: Updating distance vectors

In the first event, node A has sent its vector to node B. Node B updates its vector using the cost c BA= 2. In the second event, node E has sent its vector to node B. Node B updates its vector using the cost c EA=4. After the first event, node B has one improvement in its vector: its least cost to node D has changed from infinity to 5(via node A). After the second event, node B has one more improvement in its vector; its least cost to node F has changed from infinity to 6 (via node E). We hope that we have convinced the reader that exchanging vectors eventually stabilizes the system and allows all nodes to find the ultimate least cost between themselves and any other node. We need to remember that after updating a node, it immediately sends its updated vector to all neighbours. Even if its neighbours have received the previous vector, the update done may help more

Distance-Vector Routing Algorithm:

Now we can give a simplified pseudocode for the distance-vector routing algorithm, as shown in Figure 4.7. The algorithm is run by its node independently and a synchronously.

Lines 4 to 11 initialize the vector for the node. Lines 14 to 23 show how the vector can be updated after receiving a vector from the immediate neighbour. The for loop in lines 17 to 20 allows all entries (cells) in the vector to be updated after receiving anew vector. Note that the node sends its vector in line 12, after being initialized, and in line22, after it is updated.

Figure 4.7: Distance-Vector Routing Algorithm for a Node

```
1  Distance_Vector_Routing ( )
2  {
3      // Initialize (create initial vectors for the node)
4      D[myself] = 0
```

```
5     for (y = 1 to N)
6     {
7         if (y is a neighbor)
8             D[y] = c[myself][y]
9         else
10            D[y] = ∞
11    }
12    send vector {D[1], D[2], ..., D[N]} to all neighbors
13    // Update (improve the vector with the vector received from a neighbor)
14    repeat (forever)
15    {
16        wait (for a vector D_w from a neighbor w or any change in the link)
17        for (y = 1 to N)
18        {
19            D[y] = min [D[y], (c[myself][w] + D_w[y])]     // Bellman-Ford equation
20        }
21        if (any change in the vector)
22            send vector {D[1], D[2], ..., D[N]} to all neighbors
23    }
24  } // End of Distance Vector
```

Figure 4.7: Distance-Vector Routing Algorithm for a Node(continued)

Count to Infinity A problem with distance-vector routing is that any decrease in cost (good news) propagates quickly, but any increase in cost (bad news) will propagate slowly. For a routing protocol to work properly, if a link is broken (cost becomes infinity), every other router should be aware of it immediately, but in distance-vector routing, this takes some time. The problem is referred to as count to infinity. It sometimes takes several updates before the cost for a broken link is recorded as infinity by all routers.

Two-Node Loop One example of count to infinity is the two-node loop problem. To understand the problem, let us look at the scenario depicted in Figure 4.8. The figure shows a system with three nodes. We have shown only the portions of the forwarding table needed for our discussion. At the beginning, both nodes A and B. know how to reach node X. But suddenly, the link between A and X fails. Node A changes its table. If A can send its table to B immediately, everything is fine. However, the system becomes unstable if B sends its forwarding table to A before receiving A's forwarding table. Node A receives the update and, assuming that B has found a way to reach X, immediately updates its forwarding table. Now A sends its new update to B. Now B thinks that something has been changed around A and updates its forwarding table. The cost of reaching X increases gradually until it reaches infinity. At this moment, both A and B know that X cannot be reached. However, during this time the system is not stable. Node A thinks that the route to X is via B; node B thinks that the route to X is via A. If A receives a packet destined for X, the packet goes to Band then comes back to A. Similarly, if B receives a packet destined for X, it goes to A and comes back to B. Packets bounce between A and B, creating a two-node loop problem

Split Horizon One solution to instability is called split horizon. In this strategy, instead of flooding the table through each interface, each node sends only part of its table through each interface. If, according to its table, node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A; the information has come from A (A already knows). In our scenario, node B eliminates the last line of its forwarding table before it sends it to A. In this case, node A keeps the value of infinity as the distance to X. Later, when node A sends its forwarding table to B, node B also corrects its forwarding table. The system becomes stable after the first update: both node A and node B know that X is not reachable.

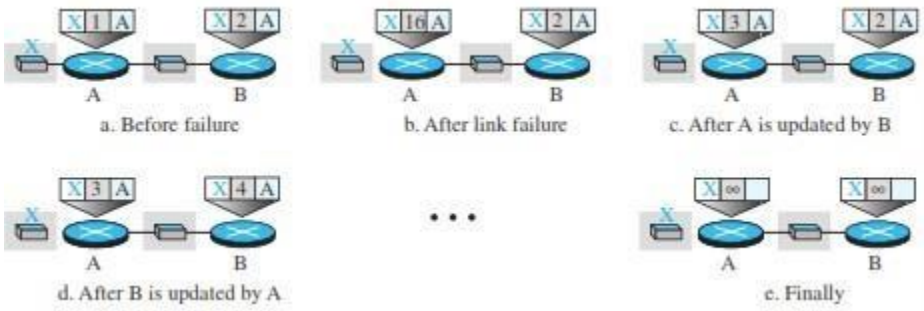


Figure 4.8 Two-

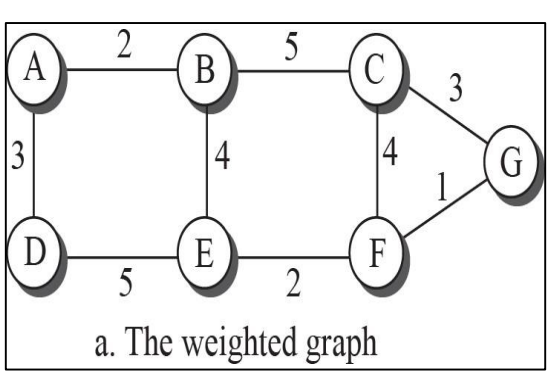
Poison Reverse Using the split-horizon strategy has one drawback. Normally, the corresponding protocol uses a timer, and if there is no news about a route, the node deletes the route from its table. When node B in the previous scenario eliminates the route to X from its advertisement to A, node A cannot guess whether this is due to the split-horizon strategy (the source of information was A) or because B has not received any news about X recently. In the poison reverse strategy B can still advertise the value for X, but if the source of information is A, it can replace the distance with infinity as a warning: "Do not use this value; what I know about this route comes from you."

Three-Node Instability The two-node instability can be avoided using split horizon combined with poison reverse. However, if the instability is between three nodes, stability cannot be guaranteed

Link-State Routing

This method uses the term link-state to define the characteristic of a link (an edge) that represents a network in the internet. In this algorithm the cost associated with an edge defines the state of the link. Links with lower costs are preferred to links with higher costs; if the cost of a link is infinity, it means that the link does not exist or has been broken.

Link-State Database (LSDB) - To create a least-cost tree with this method, each node needs to have a complete map of the network, which means it needs to know the state of each link. The collection of states for all links is called the link-state database (LSDB). There is only one LSDB for the whole internet; each node needs to have a duplicate of it to be able to create the least-cost tree. Figure 4.9 shows an example of an LSDB for the graph in Figure4.1. The LSDB can be represented as a two- dimensional array (matrix) in which the value of each cell defines the cost of the corresponding link.



	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

b. Link state database

Figure4.9:Exampleof a link-statedatabase

How each node can create this LSDB?

By flooding: Each node send *LS packet* (LSP) to all its immediate neighbours (each interface) to collect two pieces of information for each neighbouring node:

1. The identity of the node.

2. The cost of the link.

When a node receives an LSP, it compares the LSP with the copy it may already have. The node checks the sequence number in both LSP to know which one is old and discards it, and keep the new one. Then the node sends a copy of it out of each interface except the one from which the packet arrived. This guarantees that floodingstopssomewhereinthenetwork(whereanodehasonlyoneinterface).

Each node creates the comprehensive LSDB. This LSDB is the same for each node and shows the whole map of the internet. In other words, a node can make the whole map if it needs to, using this LSDB.

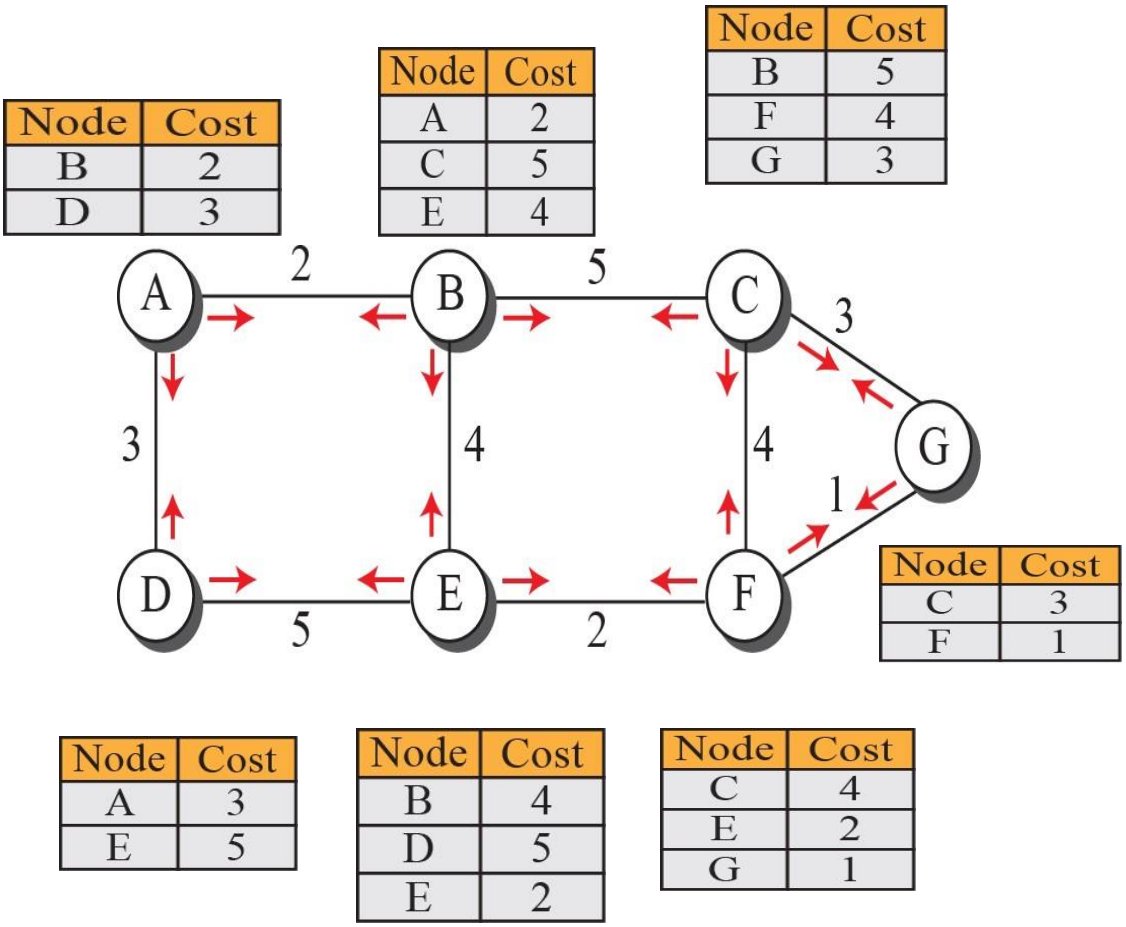


Figure 4.10: LSPs created and sent out by each node to build LSDB

Formation of Least-Cost Trees -To create a least-cost tree for itself, using the shared LSDB, each node needs to runthefamousDijkstraAlgorithm.Thisiterativealgorithmusesthefollowingsteps:

1.Thenode chooses itself as the root of the tree, creating a tree with a single node, and sets the total cost of each node based on the information in the LSDB.

2.The nodes elects one node, among all nodes not in the tree, which is closest to the root, and adds this to the tree. After this node is added to the tree, the cost of all other nodes not in the tree needs to be updated because the paths may have been changed.

3. The node repeats step 2 until all nodes are added to the tree. We need to convince ourselves that the above three steps finally create the least-cost tree. Table 4.11 shows a simplified version of Dijkstra's algorithm

```
1  Dijkstra's Algorithm ( )
2  {
3      // Initialization
4      Tree = {root}                    // Tree is made only of the root

5      for (y = 1 to N)                 // N is the number of nodes
6      {
7          if (y is the root)
8              D[y] = 0                 // D[y] is shortest distance from root to node y
9          else if (y is a neighbor)
10             D[y] = c[root][y]        // c[x][y] is cost between nodes x and y in LSDB
11         else
12             D[y] = ∞
13     }
14     // Calculation
15     repeat
16     {
17         find a node w, with D[w] minimum among all nodes not in the Tree
18         Tree = Tree ∪ {w}            // Add w to tree
19         // Update distances for all neighbors of w
20         for (every node x, which is a neighbor of w and not in the Tree)
21         {
22             D[x] = min{D[x], (D[w] + c[w][x])}
23         }
24     } until (all nodes included in the Tree)
25 } // End of Dijkstra
```

Table 4.11 Dijkstra's Algorithm

Lines 4 to 13 implement step 1 in the algorithm. Lines 16 to 23 implement step 2 in the algorithm. Step 2 is repeated until all nodes are added to the tree. Figure4.12showsthe formation of the least- cost tree for the graphinFigure4.10usingDijkstra'salgorithm.

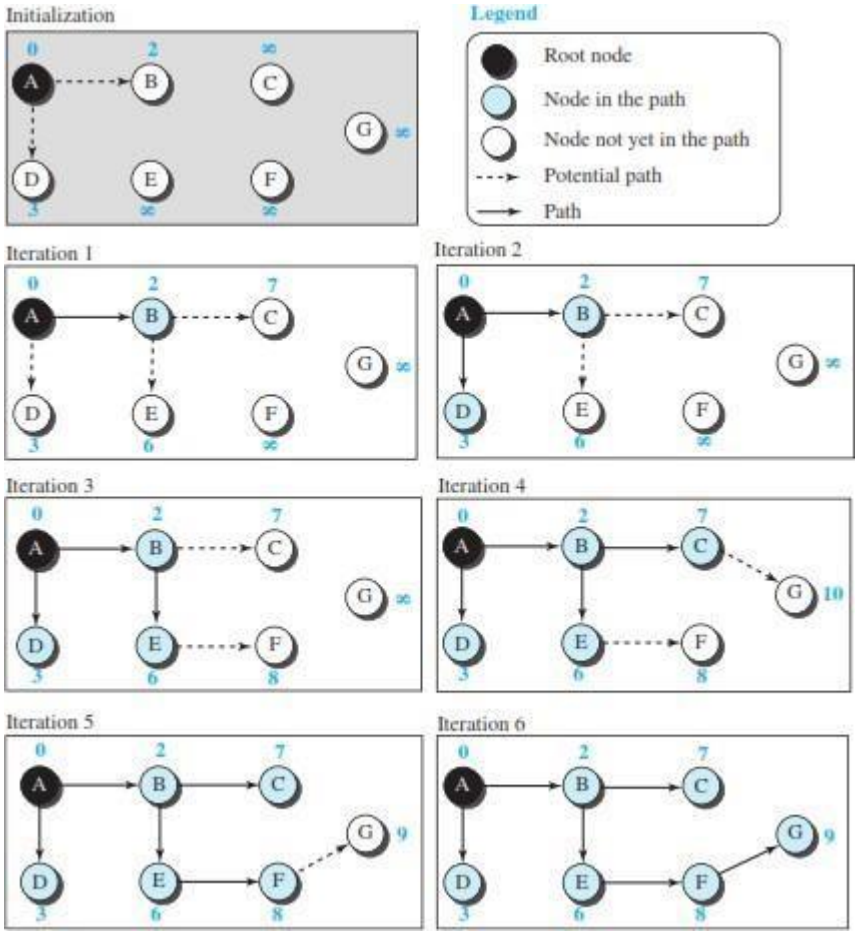


Figure4.12: Least-cost tree

Path-Vector Routing

Both link-state and distance-vector routing are based on the least-cost goal. However, there are instances where this goal is not the priority. For example, assume that there are some routers in the internet that a sender wants to prevent its packets from going through. For example, a router may belong to an organization that does not provide enough security or it may belong to a commercial rival of the sender which might inspect the packets for obtaining information. Least-cost routing does not prevent a packet from passing through an area when that area is in the least-cost path. In other words, the least-cost goal, applied by LS or DV routing, does not allow a sender to apply specific policies to the route a packet may take. A side from safety and security, there are occasions, in which the goal of routing is merely reachability: to allow the packet to reach its destination more efficiently without assigning costs to the route.

To respond to these demands, a third routing algorithm, called path-vector (PV) routing has been devised. Path-vector routing does not have the drawbacks of LS or DV routing as described above because it is not based on least-cost routing. The best route is determined by the source using the policy it imposes on the route. In other words, the source can control the path. Although path-vector routing is not actually used in an internet, and is mostly designed to route a packet between ISPs.

Spanning Trees

In path-vector routing, the path from a source to all destinations is also determined by the best spanning tree. The best spanning tree, however, is not the least- cost tree; it is the tree determined by

the source when it imposes its own policy. If there is more than one route to a destination, the source can choose the route that meets its policy best. A source may apply several policies at the same time. Figure 4.13 shows a small internet with only five nodes. Each source has created its own spanning tree that meets its policy. The policy imposed by all sources is to use the minimum number of nodes to reach a destination. The spanning tree selected by A and E is such that the communication does not passthrough D as a middle node. Similarly, the spanning tree selected by B is such that the communication does not pass-through C as a middle node.

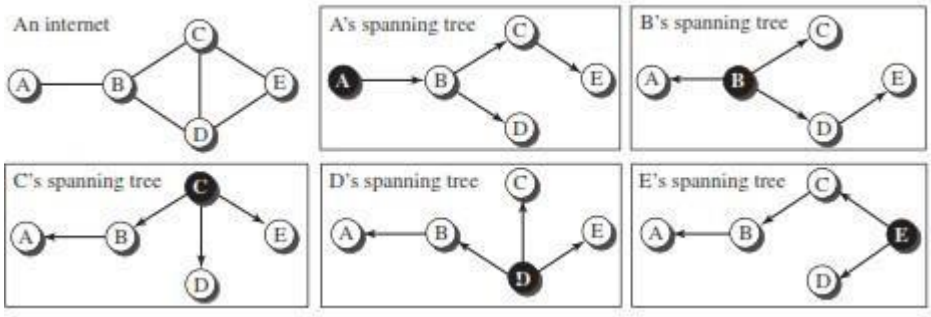


Figure 4.13: Spanning trees in path-vector routing

Creation of Spanning Trees

Path-vector routing, like distance-vector routing, is an asynchronous and distributed routing algorithm. The spanning trees are made, gradually and a synchronously, by each node. When a node is booted, it creates a path vector based on the information it can obtain about its immediate neighbour. Anode sends greeting messages to its immediate neighbours to collect these pieces of information. Figure 4.14 shows all of these path vectors for our internet in Figure 4.13. All of these tables are not created simultaneously; they are created when each node is booted. The figure also shows how these path vectors are sent to immediate neighbours after they have been created (arrows). Each node, after the creation of the initial path vector, sends it to all its immediate neighbours. Each node, when it receives a path vector from a neighbour, updates its path vector using an equation similar to the Bellman-Ford, but applying its own policy instead of looking for the least cost.

Path(x,y)=best{Path(x,y),[(x +Path(v,y)]} for all v'sin the internet.

In this equation, the operator (+) means to add x to the beginning of the path. We also need to be cautious to avoid adding a node to an empty path because an empty path means one that does not exist.

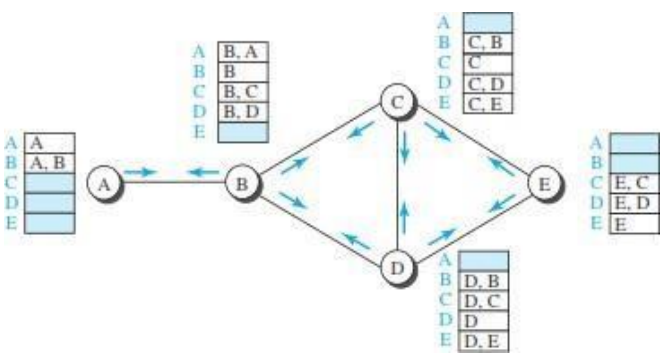


Figure 4.14: Path vectors made at bootingtime

Path-vector routing also imposes one more condition on this equation: If Path (v, y) includes x, that path is discarded to avoid a loop in the path. In other words, x does not want to visit itself when it selects a path to y. Figure4.15showsthe path vector of node C after two events. In the first event, node C receives a copy of B's vector, which improves its vector: now it knows how to reach node A. In the second event, node C receives a copy of D's vector, which does not change its vector. The vector for node C after the first event is stabilized and serves as its forwarding table.

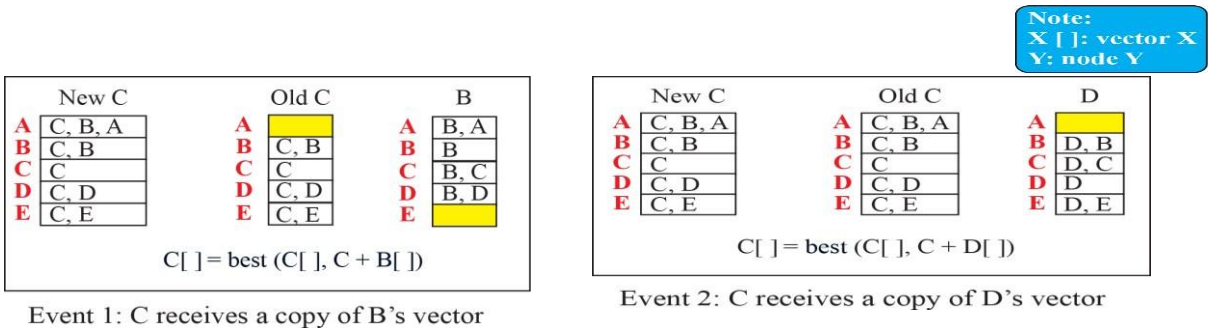


Figure 4.15: Updating path vectors

Path-Vector Algorithm

Based on the initialization process and the equation used in updating each forwarding table after receiving path vectors from neighbours, we can write a simplified version of the path vector algorithm as shown in Table
4.2.

```
1  Path_Vector_Routing ( )
2  {
3     // Initialization
4     for (y = 1 to N)
5     {
6        if (y is myself)
7           Path[y] = myself
8        else if (y is a neighbor)
9           Path[y] = myself + neighbor node
10       else
11          Path[y] = empty
12    }
13    Send vector {Path[1], Path[2], ..., Path[y]} to all neighbors
14    // Update
15    repeat (forever)
16    {
17       wait (for a vector Path_w from a neighbor w)
18       for (y = 1 to N)
19       {
20          if (Path_w includes myself)
21             discard the path                    // Avoid any loop
22          else
23             Path[y] = best {Path[y], (myself + Path_w[y])}
24       }
25       If (there is a change in the vector)
26          Send vector {Path[1], Path[2], ..., Path[y]} to all neighbors
27    }
28 } // End of Path Vector
```

Table 4.2 Path-vector algorithm for anode

Lines 4 to 12 show the initialization for the node. Lines 17 to 24 show how the node updates its vector after receiving a vector from the neighbour. The update process is repeated forever. We can see the similarities between this algorithm and the DV algorithm.

UNICAST ROUTING PROTOCOLS

A protocol is more than an algorithm. A protocol needs to define its domain of operation, the messages exchanged, communication between routers, and interaction with protocols in other domains.

We discuss three common protocols used in the Internet:

1. Routing Information Protocol (RIP), based on the distance-vector algorithm.
2. Open Shortest Path First (OSPF), based on the link-state algorithm.
3. Border Gateway Protocol (BGP), based on the path-vector algorithm.

Internet Structure

Today, the Internet has changed from a tree-like structure, with a single backbone, to a multi-backbone structure run by different private corporations. The Internet has a structure similar to what is shown in Figure 4.16. The Internet has changed from a tree-like structure, with a single backbone, to a multi-backbone structure run by different private corporations today.

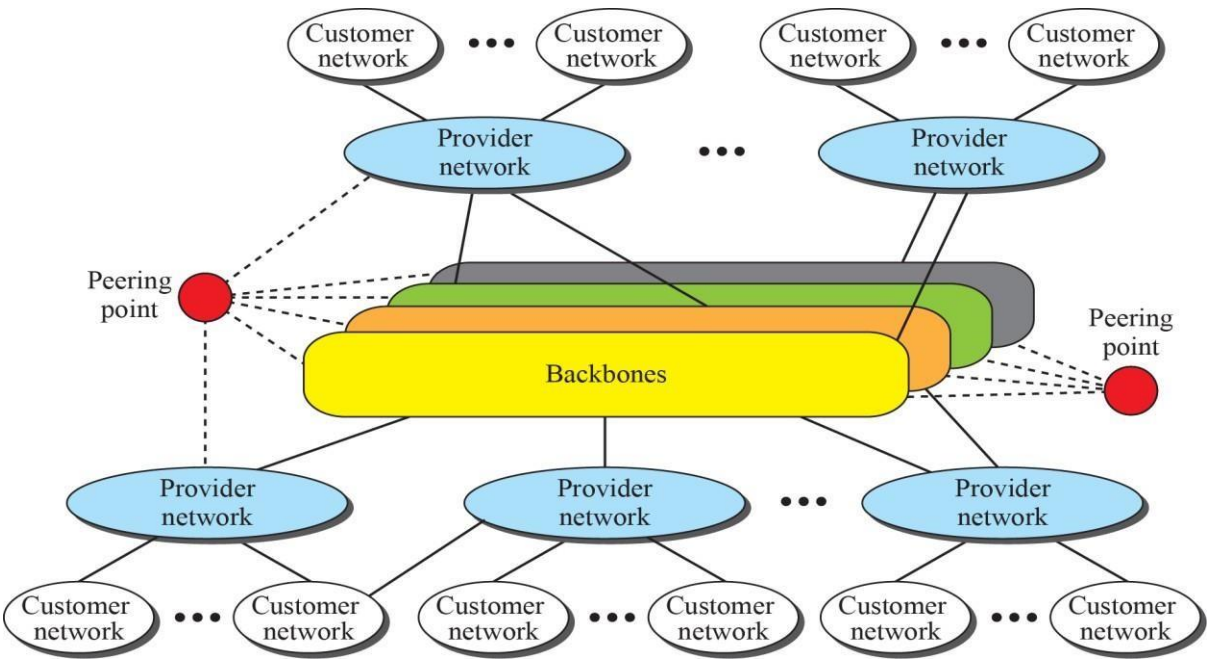


Figure 4.16: Internet structure

There are several backbones run by private communication companies that provide global connectivity. These backbones are connected by some peering points that allow connectivity between backbones. At a lower level, there are some provider networks that use the backbones for global connectivity but provide services to Internet customers. Finally, there are some customer networks that use the services provided by the provider networks. Any of these three entities (backbone, provider network, or customer network) can be called an Internet Service Provider or ISP. They provide services, but at different levels.

Hierarchical Routing

The Internet today is made of a huge number of networks and routers that connect them. Routing in the Internet cannot be done using a single protocol for two reasons: as capability problem and an administrative issue. Scalability problem means that the size of the forwarding tables becomes huge, searching for a destination in a forwarding table becomes time-consuming, and updating creates a huge amount of traffic.
TheadministrativeissueisrelatedtotheInternetstructuredescribedinFigure
4.16. As the figure shows, each ISP is run by an administrative authority. The administrator needs to have control in its system. The organization must be able to use as many subnets and routers as it needs, may desire that the routers be from a particular manufacturer, may wish to run a specific routing algorithm to meet the needs of the organization, and may want to impose some policy on the traffic passing through its ISP. Hierarchical routing means considering each ISP as an autonomous system (AS). Each A Scan run a routing protocol that meets its needs, but the global Internet runs a global protocol to glue all Ass together.

The **routing protocol run in each AS** is referred to as **intra-AS routing protocol, intradomain routing protocol, or interior gateway protocol (IGP)**; the **global routing protocol is referred to as inter-AS routing protocol, interdomain routing protocol, or exterior gateway protocol (EGP).** There may be several intradomain routing protocols, and each AS is free to choose one, but it should be clear that there should be only one interdomain protocol that handles routing between these entities. Presently, the two common intradomain routing protocols are RIP and OSPF; the only interdomain routing protocol is BGP. The situationmaychangewhenwemovetoIPv6.

Autonomous Systems

Each ISP is an autonomous system when it comes to managing networks and routers under its control. There are small, medium-size, and large Ass, and each AS is given an autonomous number (ASN)by the ICANN. Each ASN is a 16-bit unsigned integer that uniquely defines an AS. The autonomous systems, however, are not categorized according to their size; they are categorized according to the way they are connected to another ASs. We have stub ASs, multihomed ASs, and transient ASs. The type, affects the operation of the interdomain routing protocol in relation to that AS.

❑**Stub AS.** A stub AS has only one connection to another AS. The data traffic can be either initiated or terminated in a stub AS; the data cannot pass through it. A good example of a stub AS is the customer network, which is either the source or the sink of data.

❑**Multihomed AS**. A multihomed AS can have more than one connection to other ASs, but it does not allow data traffic to pass through it. A good example of such an AS is some of the customer ASs that may use the services of more than one provider network, but their policy does not allow data to be passed through them.

❑**Transient AS**. A transient AS is connected to more than one other AS and also allows the traffic to pass through. The provider networks and the backbone are good examples of transient ASs.

Routing Information Protocol (RIP)

The Routing Information Protocol (RIP)is one of the most widely used intradomain routing protocols based on the distance-vector routing algorithm. RIP was started as part of the Xerox Network System (XNS), but it was the Berkeley Software Distribution (BSD)version of UNIX that helped make the use of RIP widespread.

Hop Count

A router in this protocol basically implements the distance-vector routing algorithm shown in Table 1. However, the algorithm has been modified as described below.

1. SincearouterinanASneedstoknowhowtoforwardapackettodifferentnetworks(subnets) in an AS, RIP routers advertise the cost of reaching different networks instead of reaching other nodes in a theoretical graph. In other words, the cost is defined between a router and the network in which the destination host is located.
2. To make the implementation of the cost simpler (independent from performance factors of the routers and links, such as delay, bandwidth, and so on), the cost is defined as the number of hops, which means the number of networks (subnets) a packet needs to travel through from the source router to the final destination host. Note that the network in which the source host is connected is not counted in this calculation because the source host does not use a forwarding table; the packet is delivered to the default router. Figure 4.17 shows the concept of hop count advertised by three routers from a source host to a destination host. In RIP, the maximum cost of a path canbe15, which means 16 is considered as infinity (no connection). For this reason, **RIP can be used only in autonomous systems in which the diameter of the AS is not more than 15 hops.**

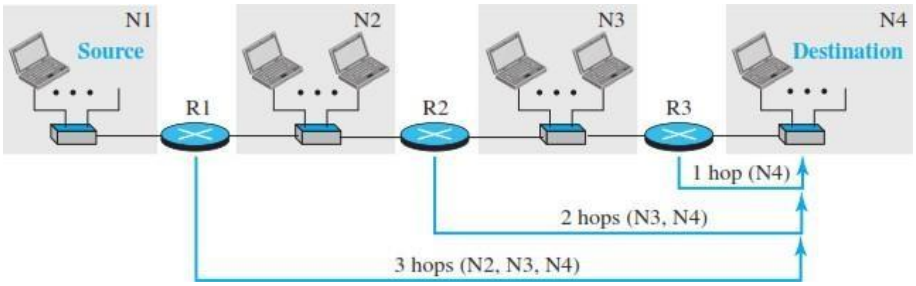


Figure 4.17: Hop counts in RIP

Forwarding Tables

The routers in an autonomous system need to keep forwarding tables to forward packets to their destination networks. **A forwarding table in RIP is a three-column table in which the first column is the address of the destination network, the second column is the address of the next router to which the packet should be forwarded, and the third column is the cost (the number of hops) to reach the destination network.** Figure 4.18 shows the three forwarding tables for the routers in Figure 4.17. Note that the first and the third columns together convey the same information as does a distance vector, but the cost shows the number of hops to the destination networks.

Forwarding table for R1

Destination network	Next router	Cost in hops
N1	——	1
N2	——	1
N3	R2	2
N4	R2	3

Forwarding table for R2

Destination network	Next router	Cost in hops
N1	R1	2
N2	——	1
N3	——	1
N4	R3	2

Forwarding table for R3

Destination network	Next router	Cost in hops
N1	R2	3
N2	R2	2
N3	——	1
N4	——	1

Figure 4.18 Forwarding tables

Although a forwarding table in RIP defines only the next router in the second column, it gives the information about the whole least-cost tree. For example, R1definesthatthe next router for the path

to N4isR2; R2definest hat the next router to N4 is R3; R3 defines that there is no next router for this path. The tree is then R1→R2→R3→ N4.

RIP is implemented as **a process that uses the service of UDP** on the well-known **port number 520**. In BSD, RIP is a daemon process (a process running in the background), named "routed" (abbreviation for route daemon and pronounced route-dee). This means that, although RIP is a routing protocol to help IP route its datagrams through the AS, the RIP messages are encapsulated inside UDP user datagrams, which in turn are encapsulated inside IP datagrams. In other words, RIP runs at the application layer, but creates forwarding tables for IP at the network later

RIP Messages

Two RIP processes, a client and a server, like any other processes, need to exchange messages. RIP-2 defines the format of the message, as shown in Figure4.19. Part of the message, called **entry**, can be repeated as needed in a message. Each entry carries the information related to one line in the forwarding table of the router that sends the message.

RIP

RIP has two types of messages: **request and response**. A request message is sent by a router that has just come up or by a router that has some time-out entries. A request message can ask about specific entries or all entries. A response (or update) message can be either solicited or unsolicited. A solicited response message is sent only in answer to a request message. It contains information about the destination specified in the corresponding request message. An unsolicited response message, on the other hand, is sent periodically, every 30secondsorwhenthereisachangeinthe forwarding table

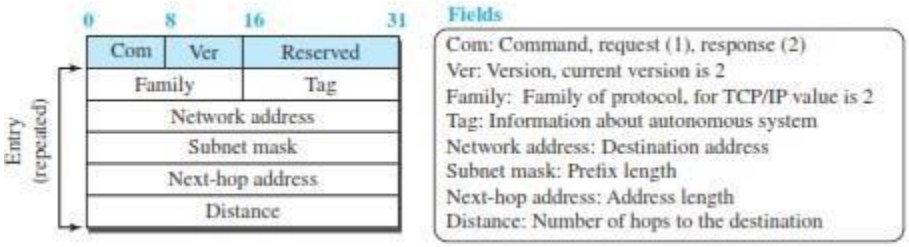


Figure 4.19 RIP message format

RIP Algorithm

RIP implements the same algorithm as the distance-vector routing algorithm. Some changes need to be made to the algorithm to enable a router to update its forwarding table:

❑Instead of sending only distance vectors, a router needs to send the whole contents of its forwarding table in a response message.

❑The receiver adds one hop to each cost and changes the next router field to the address of the sending router. Each route in the modified forwarding table is the received route and each route in the old forwarding table the old route. The received router selects the old routes as the new ones except in the following three cases:

1. If the received route does not exist in the old forwarding table, it should be added to the route.

2.If the cost of the received route is lower than the cost of the old one, the received route should be selected as the new one.

3. If the cost of the received route is higher than the cost of the old one, but the value of the next router is the same in both routes, the received route should be selected as the new one. This is the case where the route was actually advertised by the same router in the past, but now the situation has been changed. For example, suppose a neighbour has previously advertised a route to a destination with cost3, but now there is no path between this neighbour and that destination. The neighbour advertises this destination with cost value infinity (16in RIP). The receiving router must not ignore this value even though its old route has a lower cost to the same destination.

❑The new forwarding table needs to be sorted according to the destination route (mostly using the longest prefix first)

Example 4.2 Figure 4.20 shows a more realistic example of the operation of RIP in an autonomous system. First, the figure shows all forwarding tables after all routers have been booted. Then there are changes in some tables when some update messages have been exchanged. Finally, there are the stabilized forwarding tables when there is no more change

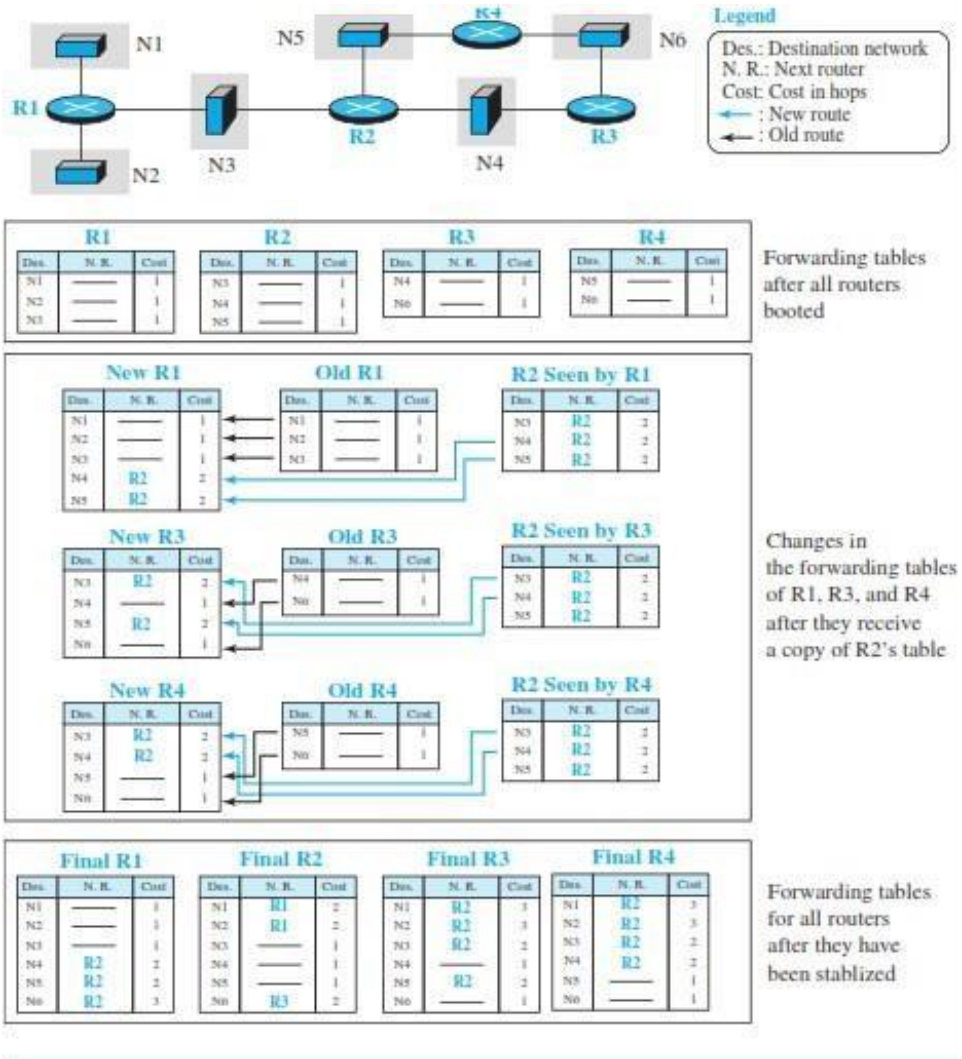


Figure 4.20: Example of an autonomous system using RIP

Timers in RIP

RIP uses three timers to support its operation.

1. The **periodic timer** controls the advertising of regular update messages. Each router has one periodic timer that is randomly set to a number between 25 and 35 seconds (to prevent all routers sending their messages at the same time and creating excess traffic). The timer counts down; when zero is reached, the update message is sent, and the timer is randomly set once again.

2. The **expiration timer** governs the validity of a route. When a router receives update information for a route, the expiration timer is set to 180 seconds for that particular route. Every time a new update for the route is received, the timer is reset. If there is a problem on an internet and no update is received within the allotted 180 seconds, the route is considered expired and the hop count of the route is set to 16, which means the destination is unreachable. Every route has its own expiration timer.

3. The **garbage collection timer** is used to purge a route from the forwarding table. When the information about a route becomes invalid, the router does not immediately purge that route from its table. Instead, it continues to advertise the route with a metric value of 16. At the same time, a garbage collection timer is set to 120 seconds for that route. When the count reaches zero, the route is purged from the table. This timer allows neighbours to become aware of the invalidity of a route prior to purging

   Performance

   Before ending this section, let us briefly discuss the performance of RIP:

   ❑**Update Messages** - The update messages in RIP have a very simple format and are sent only to neighbours; they are local. They do not normally create traffic because the routers try to avoid sending them at the same time.

   ❑**Convergence of Forwarding Tables** – RIP uses the distance-vector algorithm, which can converge slowly if the domain is large, but, since RIP allows only 15 hops in a domain (16 is considered as infinity), there is normally no problem in convergence. The only problems that may slow down convergence are count-to-infinity and loops created in the domain; use of poison-reverse and split horizon strategies added to the RIP extension may alleviate the situation.

   ❑**Robustness** - As distance-vector routing is based on the concept that each router sends what it knows about the whole domain to its neighbours. This means that the calculation of the forwarding table depends on information received from immediate neighbours, which in turn receive their information from their own neighbours. If there is a failure or corruption in one router, the problem will be propagated to all router sand the forwarding in each router will be affected.

Open Shortest Path First

Open Shortest Path First (OSPF)is also an intradomain routing protocol like RIP, but it is based on the link-state routing protocol. OSPF is an open protocol, which means that the specification is a public document.

Metric

In OSPF, like RIP, the cost of reaching a destination from the host is calculated from the source router to the destination network. Each link (network) can be assigned a weight based on the throughput, round-trip time, reliability, and soon. An administration can also decide to use the hop count as the cost. An interesting point about the cost in OSPF is that different service types (TOSs)can have different weights as the cost. Figure 4.21 shows the idea of the cost from a router to the destination host network. We can compare the figure with Figure4.17 for the RIP.

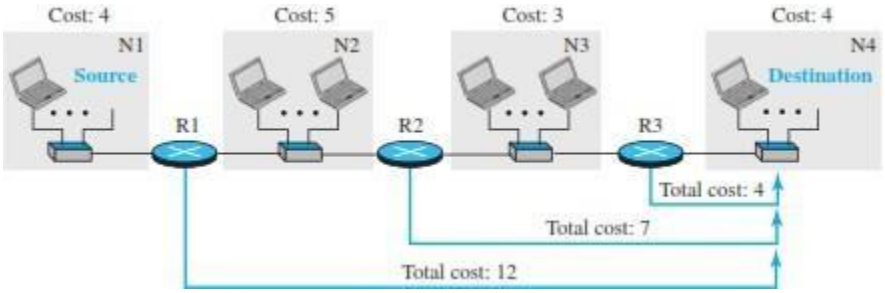


Figure4.21 Metric in OSPF

Forwarding Tables

Each OSPF router can create a forwarding table after finding the shortest-path tree between itself and the destination using Dijkstra's algorithm. Figure4.22showsthe forwarding tables for the simple AS in Figure4.21. Comparing the forwarding tables for the OSPF and RIP in the same AS, we find that the only difference is the cost values. The reason for this consistency is that both protocols use the shortest-path trees to define the best route from a source to a destination.

Areas

Compared with RIP, which is normally used in small ASs, OSPF was designed to be able to handle routing in a small or large autonomous system. However, the formation of shortest-path trees in OSPF requires that all routers flood the whole AS with their LSPs to create the global LSDB. Although this may not create a problem in a small AS, it may have created a huge volume of traffic in a large AS. To prevent this, the AS needs to be divided into small sections called areas. Each area acts as a small independent domain for flooding LSPs. In other words, OSPF uses another level of hierarchy in routing: the first level is the autonomous system, the second is the area.

Forwarding table for R1

Destination network	Next router	Cost
N1	——	4
N2	——	5
N3	R2	8
N4	R2	12

Forwarding table for R2

Destination network	Next router	Cost
N1	R1	9
N2	——	5
N3	——	3
N4	R3	7

Forwarding table for R3

Destination network	Next router	Cost
N1	R2	12
N2	R2	8
N3	——	3
N4	——	4

Figure 4.22: Forwarding tables in OSPF

However, each router in an area needs to know the information about the link states not only in its area but also in other areas. For this reason, one of the areas in the AS is designated as the backbone area, responsible for gluing the areas together. The routers in the backbone area are responsible for

Passing the information collected by each area to all other areas. In this way, a router in an area can receive all LSPs generated in other areas. For the purpose of communication, each area has an area identification. The area identification of the backbone is zero. Figure 20.23 shows an autonomous system and its areas.

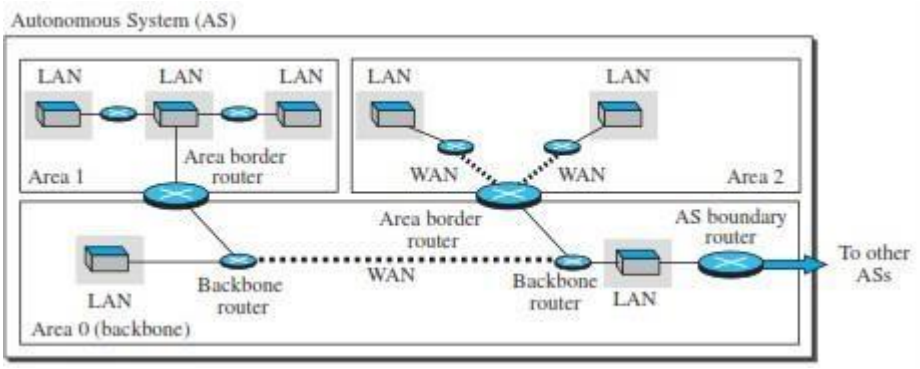


Figure 20.23: Areas in an autonomous system

Link-State Advertisement

OSPF is based on the link-state routing algorithm, which requires that a router advertise the state of each link to all neighbours for the formation of the LSDB. When we discussed the link-state algorithm, we used the graph theory and assumed that each router is a node and each network between two routers is an edge. The situation is different in the real world, in which we need to advertise the existence of different entities as nodes, the different types of links that connect each node to its neighbours, and the different types of cost associated with each link. This means we need different types of advertisements, each capable of advertising different situations. We can have five types of link-state advertisements: router link, network link, summary link to network, summary link to AS border router, and external link. Figure 20.24 shows these five advertisements and their uses.

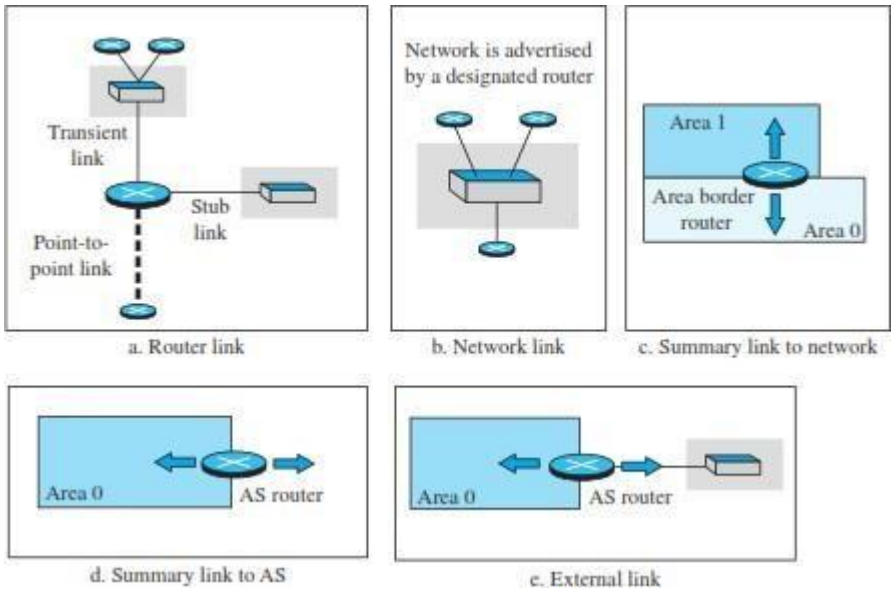


Figure 4.24: Five different LSPs

Router link. A router link advertises the existence of a router as a node. In addition to giving the address of the announcing router, this type of advertisement can define one or more types of links

that connect the advertising router to other entities. A **transient link** announces a link to a transient network, a network that is connected to the rest of the networks by one or more routers. This type of advertisement should define the address of the transient network and the cost of the link. A **stub link** advertises a link to a stub network, a network that is not a through network. Again, the advertisement should define the address of the network and the cost. A **point-to-point link** should define the address of the router at the end of the point-to-point line and the cost to get there.

❑**Network link**. A network link advertises the network as a node. However, since a network cannot do announcements itself (it is a passive entity), one of the routers is assigned as the designated router and does the advertising. In addition to the address of the designated router, this type of LSP announces the IP address of all routers (including the designated router as a router and not as speaker of the network), but no cost is advertised because each router announces the cost to the network when it sends a router link advertisement.

❑**Summary link to network**. This is done by an area border router; it advertises the summary of links collected by the backbone to an area or the summary of links collected by the area to the backbone. This type of information exchange is needed to glue the areas together.

❑**Summary link to AS**. This is done by an AS router that advertises the summary links from other ASs to the backbone area of the current AS, information which later can be disseminated to the areas so that they will know about the networks in another ASs. The need for this type of information exchange is better understood in inter-AS routing (BGP).

❑**External link**. This is also done by an AS router to announce the existence of a single network outside the AS to the back bone area to be disseminated into the areas.

OSPF Implementation

OSPF is implemented as a program in the network layer, using the service of the IP for propagation. An IP datagram that carries a message from OSPF sets the value of the protocol field to 89. This means that, although OSPF is a routing protocol to help IP to route its datagrams inside an AS, the OSPF messages are encapsulated inside datagrams. OSPF has gone through two versions:version1and version2.Most implementations useversion2

OSPF Messages

OSPF is a very complex protocol; it uses five different types of messages. In Figure20.25, the format of the OSPF common header (which is used in all messages) and the link-state general header (which is used in some messages) are given. Then, the outlines of five message types used in OSPF are given. The **hello message** (type1) is used by a router to introduce itself to the neighbours and announce all neighbours that it already knows. The **database description message** (type 2) is normally sent in response to the hello message to allow a newly joined router to acquire the full LSDB. The **link state request message** (type 3) is sent by a router that needs information about a specific LS. The **link-state update message** (type 4) is the main OSPF message used for building the LSDB. This message, in fact, has five different versions (router link, network link, summary link to network, summary link to AS

border router, and external link). The **link-state acknowledgment message** (type 5) is used to create reliability in OSPF; each router that receives a link-state update message need to acknowledge it.

Authentication - As Figure20.25shows, the OSPF common header has the provision for authentication of the Message sender. This prevents a malicious entity from sending OSPF messages
to a router and causing the router to become part of the routing system to which it actually does not belong.

OSPF Algorithm

OSPF implements the link-state routing algorithm we discussed in the previous section. However, some changes and augmentations need to be added to the algorithm:

❑After each router has created the shortest-path tree, the algorithm needs to use it to create the corresponding routing algorithm.

❑ The algorithm needs to be augmented to handle sending and receiving all five types of messages.

Performance

The performance of OSPF includes:

❑**Update Messages**. The link-state messages in OSPF have a somewhat complex format. They also are flooded to the whole area. If the area is large, these messages may create heavy traffic and use a lot of bandwidth.

❑**Convergence of Forwarding Tables**. When the flooding of LSPs is completed, each router can create its own shortest-path tree and forwarding table; convergence is fairly quick. However, each router needs to run Dijkstra's algorithm, which may take some time.

❑**Robustness**. The OSPF protocol is more robust than RIP because, after receiving the completed LSDB, each router is independent and does not depend on other routers in the area. Corruption or failure in one router does not affect other routers as seriously as in RIP

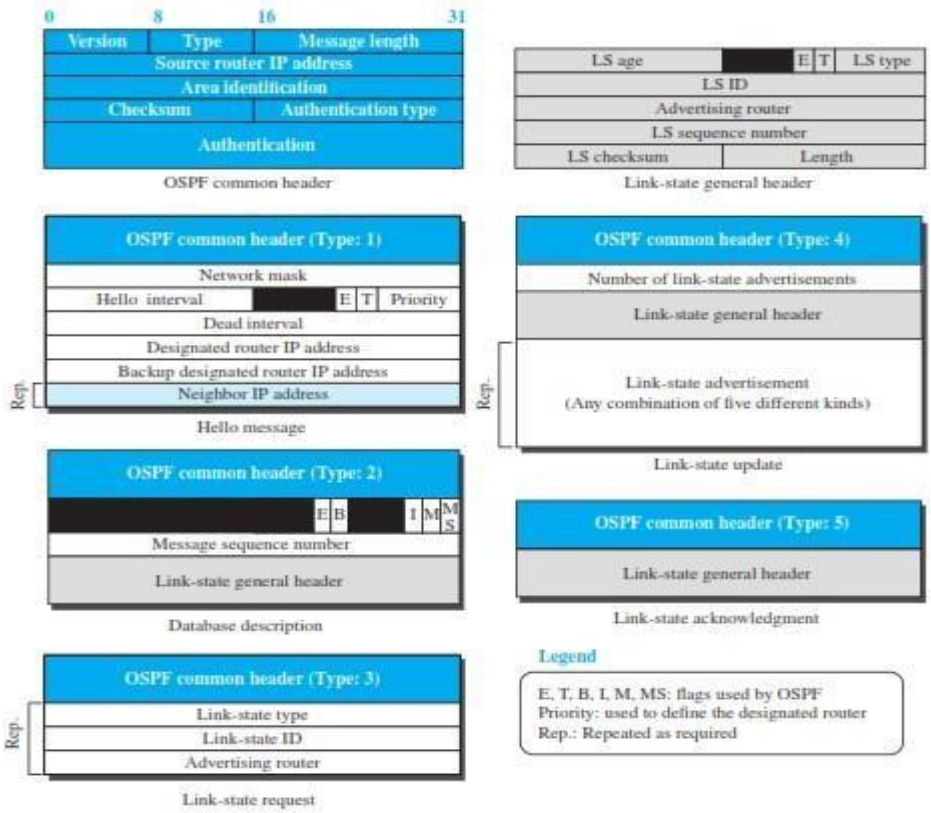


Figure 4.25: OSPF message formats

Border Gateway Protocol Version 4 (BGP4)

The Border Gateway Protocol version 4 (BGP4) is the only interdomain routing protocol used in the Internet today. BGP4 is based on the path-vector algorithm we described before, but it is tailored to provide information about the reachability of networks in the Internet.

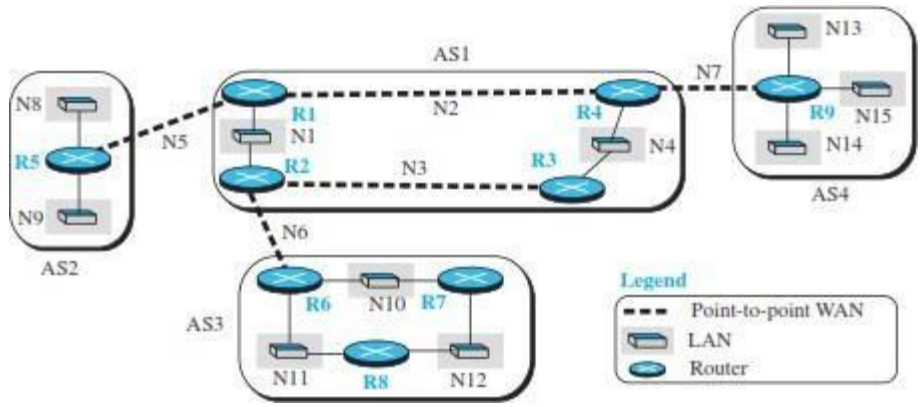


Figure 4.26 A sample internet with four ASs

Figure 4.26 shows an example of an internet with four autonomous systems. AS2, AS3, and AS4 are stub autonomous systems; AS1is a transient one. In our example, data exchange between AS2, AS3, and AS4should passthrough AS1.Eachautonomous systemin Figure4.26usesoneof the two common intradomain protocols, RIP or OSPF. Each router in each AS knows how to reach a network that is in its own AS, but it does not know how to reach a network in another AS. To enable each router to route a packet to any network in the internet, a variation of BGP4, called external BGP (eBGP)is installed on each border router (the one at the edge of each AS which is connected to a router at another AS). Then

install the second variation of BGP, called internal BGP (iBGP), on all routers. This means that the border routers will be running three routing protocols (intradomain, eBGP, and iBGP), but other routers are running two protocols (intradomain and iBGP). In this section, we introduce the basics of BGP and its relationship with intradomain routing protocols (RIP or OSPF).

Operation of External BGP (eBGP)

BGP is a kind of point-to-point protocol. When the software is installed on two routers, they try to create a TCP connection using the well-known port 179. Here, a pair of client and server processes continuously communicate with each other to exchange messages. The two routers that run the BGP processes are called BGP peers or BGP speakers. The eBGP variation of BGP allows two physically connected border routers in two different Ass to form pairs of eBGP speakers and exchange messages. The routers that are eligible in our example in Figure 20.26form three pairs: R1-R5, R2-R6, and R4-R9. The connection between the se pairs is established over three physical WANs (N5, N6, and N7). However, there is a need for a logical TCP connection to be created over the physical connection to make the exchange of information possible. Each logical connection in BGP parlance is referred to as a session. This means that we need three sessions in our example, as shown in Figure4.27.

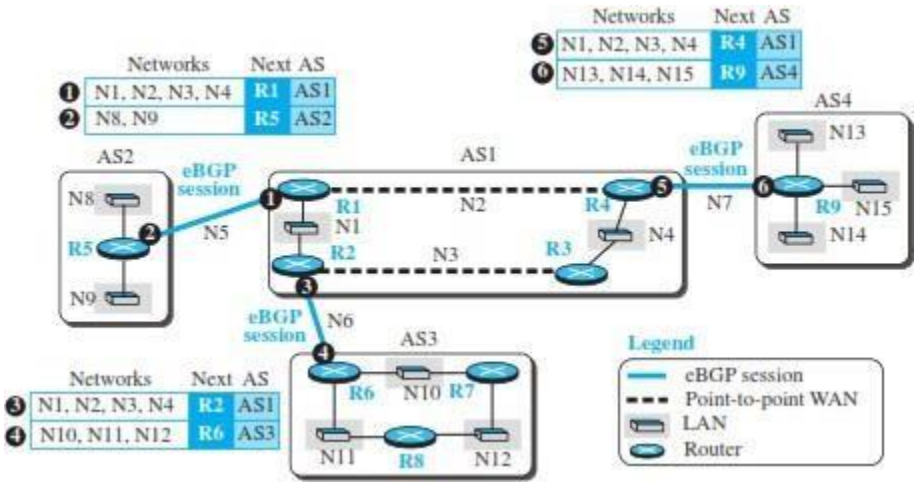


Figure 4.27 eBGP operation

The figure 4.27 also shows the simplified update messages sent by routers involved in the eBGP sessions. The circled number defines the sending router in each case. For example, message number 1 is sent by router R1 and tells router R5that N1, N2, N3, and N4 can be reached through router R1 (R1gets this information from the corresponding intradomain forwarding table). Router R5cannow add these pieces of information at the end of its forwarding table. When R5 receives any packet destined for these four networks, it can use its forwarding table and find that the next router isR1.

The messages exchanged during three eBGP sessions help some routers know how to route packets to some networks in the internet, but the reachability information is not complete. There are two problems that need to be addressed:

1.Someborder routers do not know how to route a packet destined for non-neighbour ASs. For example, R5does not know how to route packets destined for networks in AS3 and AS4. Routers R6 and R9 are in the same situation asR5:R6doesnot know about networks in AS2andAS4;R9doesnot know about networks in AS2and
AS3.

2.None of the non-border routers know how to route a packet destined for any networks in other ASs. To address the above two problems, all pairs of routers (border or non-border) will have to run the second variation of the BGP protocol, iBGP.

Operation of Internal BGP (iBGP)

The iBGP protocol is similar to the eBGP protocol in that it uses the service of TCP on the well-known port 179, but it creates a session between any possible pair of routers inside an autonomous system. However, some points should be made clear. First, if an AS has only one router, there cannot be an iBGP session. For example, we cannotcreateaniBGPsessioninsideAS2or AS4in our internet. Second, if there are n routers in an autonomous system, there should be [n ×(n −1) / 2] iBGP sessions in that autonomous system (a fully connected mesh) to prevent loops in the system. In other words, each router needs to advertise its own reachability to the peer in the session instead of flooding what it receives from another peer in another session. Figure 4.28 shows the combination of eBGP and iBGP sessions in our internet.

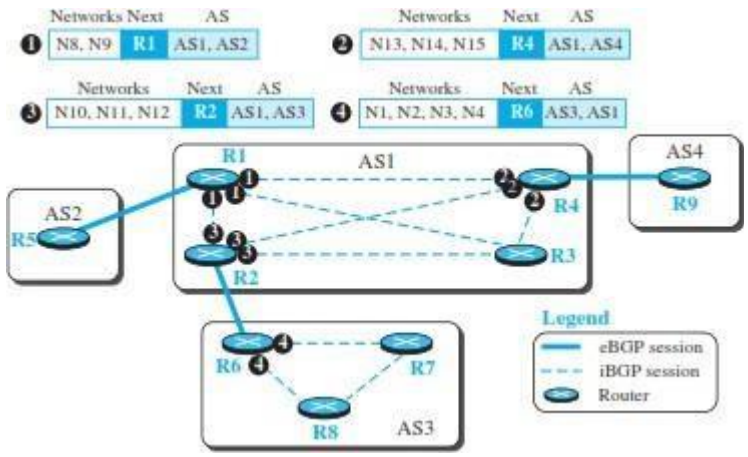
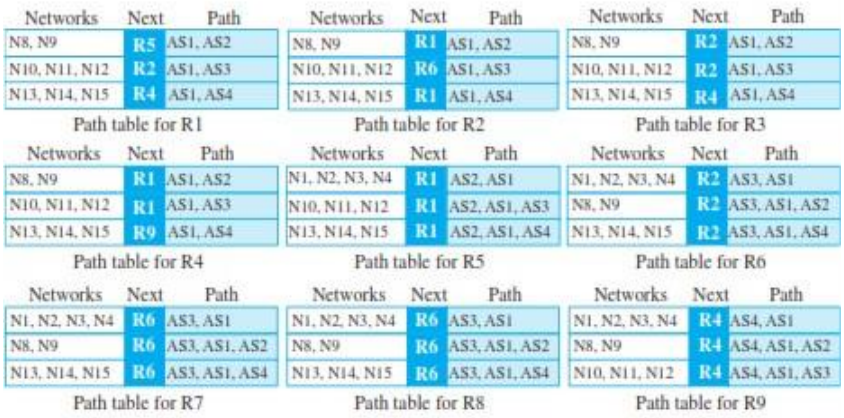


Figure4.28Combinationof eBGP and iBGP sessions in our internet

Note that the physical networks inside Ass are not shown because a session is made on an overlay network (TCP connection), possibly spanning more than one physical network as determined by the route dictated by intradomain routing protocol. Also note that in this stage only four messages are exchanged. The first message (numbered 1) is sent by R1announcing that networks N8and N9are reachable through the path AS1-AS2, but the next router is R1. This message is sent, through separate sessions, to R2, R3, andR4.RoutersR2, R4, and R6 do the same thing but send different messages to different destinations. The interesting point is that, at this stage, R3, R7, and R8create sessions with their peers, but they actually have no message to send. The updating process does not stop here. For example, after R1 receives the update message from R2, it combines the reachability information about AS3with the reachability information it already knows about AS1and sends a new update message to R5. Now R5knows how to reach networks in AS1and AS3.Theprocesscontinues when R1receives the update message from R4. The point is that we need to make certain that at a point in time there are no changes in the previous updates and that all information is propagated through all ASs. At this time, each router combines the information received from eBGP and iBGP and creates what we may call a path table after applying the criteria for finding the best path. To demonstrate, we show the path tables in Figure 4.29 for the routers in Figure 4.26. For example, router R1now knows that any packet destined for networks N8 or N9 should go through AS1and AS2and the next router to deliver the packet to is router R5. Similarly, routerR4knowsthatanypacketdestinedfor networks

N10,N11,or N12shouldgothroughAS1andAS3andthe next router to deliver this packet to is router R1,andsoon.



Path table for R1

Networks	Next	Path
N8, N9	R5	AS1, AS2
N10, N11, N12	R2	AS1, AS3
N13, N14, N15	R4	AS1, AS4

Path table for R2

Networks	Next	Path
N8, N9	R1	AS1, AS2
N10, N11, N12	R6	AS1, AS3
N13, N14, N15	R1	AS1, AS4

Path table for R3

Networks	Next	Path
N8, N9	R2	AS1, AS2
N10, N11, N12	R2	AS1, AS3
N13, N14, N15	R4	AS1, AS4

Path table for R4

Networks	Next	Path
N8, N9	R1	AS1, AS2
N10, N11, N12	R1	AS1, AS3
N13, N14, N15	R9	AS1, AS4

Path table for R5

Networks	Next	Path
N1, N2, N3, N4	R1	AS2, AS1
N10, N11, N12	R1	AS2, AS1, AS3
N13, N14, N15	R1	AS2, AS1, AS4

Path table for R6

Networks	Next	Path
N1, N2, N3, N4	R2	AS3, AS1
N8, N9	R2	AS3, AS1, AS2
N13, N14, N15	R2	AS3, AS1, AS4

Path table for R7

Networks	Next	Path
N1, N2, N3, N4	R6	AS3, AS1
N8, N9	R6	AS3, AS1, AS2
N13, N14, N15	R6	AS3, AS1, AS4

Path table for R8

Networks	Next	Path
N1, N2, N3, N4	R6	AS3, AS1
N8, N9	R6	AS3, AS1, AS2
N13, N14, N15	R6	AS3, AS1, AS4

Path table for R9

Networks	Next	Path
N1, N2, N3, N4	R4	AS4, AS1
N8, N9	R4	AS4, AS1, AS2
N10, N11, N12	R4	AS4, AS1, AS3

Figure 4.29: Finalized BGP path tables

Injection of Information into Intra domain Routing

The role of an interdomain routing protocol such as BGP is to help the routers inside the AS to augment their routing information. In other words, the path tables collected and organized by not used for routing packets; they are injected into intradomain forwarding tables (RIP or OSPF) for routing packets. This can be done in several depending on the type of AS. In the case of a stub AS, the only area border router adds a default entry at the end of its forwarding table and defines the next router to be the speaker router at the end of the eBGP connection. In Figure 4.26, R5in AS2defines R1asthe default router for all networks other thanN8andN9.Thesituationisthe same for router R9 in AS4with the default router to beR4.InAS3, R6setits default router to be R2, but R7and R8set their default router to be R6. These settings are in accordance with the path tables described in Figure 4.29 for these routers. In other words, the path tables are injected into intradomain forwarding tables by adding only one default entry. In the case of a transient AS, the situation is more complicated. R1 in AS1 needs to inject the whole contents of the path table for R1 in Figure 20.29 into its intradomain forwarding table. The situation is the same for R2, R3, and R4. One issue to be resolved is the cost value. We know that RIP and OSP Fuse different metrics. One solution, which is very common, is to set the cost to the foreign networks at the same cost value as to reach the first AS in the path. For example, the cost for R5to reach all networks in other Ass is the cost to reachN5.Thecostfor R1to reachnetworksN10to N12isthe cost to reach N6, and so on. The cost is taken from the intradomain forwarding tables (RIP or OSPF). Figure 20.30 shows the interdomain forwarding tables. For simplicity, we assume that all Ass are using RIP as the intradomain routing protocol. The shaded areas are the augmentation injected by the BGP protocol; the default destinations are indicated as zero

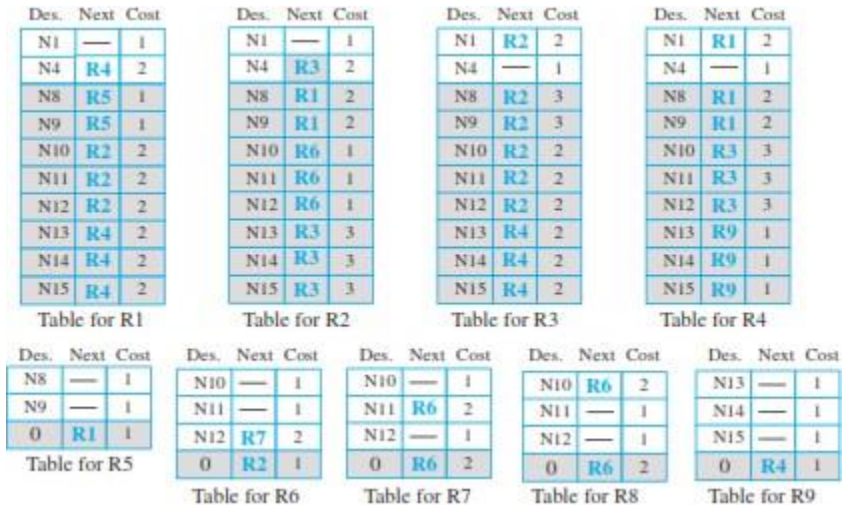


Figure 4.30 Forwarding tables after injection from BGP

Address Aggregation

The intradomain forwarding tables obtained with the help of the BGP4protocols may become huge in the case of the global Internet because many destination networks may be included in a forwarding table. Fortunately, BGP4usesthe prefixes as destination identifiers and allows the aggregation of these prefixes. For example, subnets can be reached through one path. Even if one or two of the aggregated prefixes need a separate path, the longest prefix principle allows us to do so. Prefixes 14.18.20.0/26, 14.18.20.64/26,
14.18.20.128/26, and14.18.20.192/26, can be combined into
14.18.20.0/24andall subnets can be reached through one path.

Path Attributes

In both intradomain routing protocols (RIP or OSPF), a destination is normally associated with two pieces of information: next hop and cost. The first one shows the address of the next router to deliver the packet; the second defines the cost to the final destination. Interdomain routing is more involved and naturally needs more information about how to reach the final destination. In BGP these pieces are called path attributes. BGP allows a destination to be associated with up to seven path attributes. Path attributes are divided into two broad categories: well-known and optional. A well-known attribute must be recognized by all routers; an optional attribute need not be. A well-known attribute can be mandatory, which means that it must be present in any BGP update message, or discretionary, which means it does not have to be. An optional attribute can be either transitive, which means it can pass to the next AS, or intransitive, which means it cannot. All attributes are inserted after the corresponding destination prefix in an update message (discussed later). The format for an attribute is shown in Figure4.31.

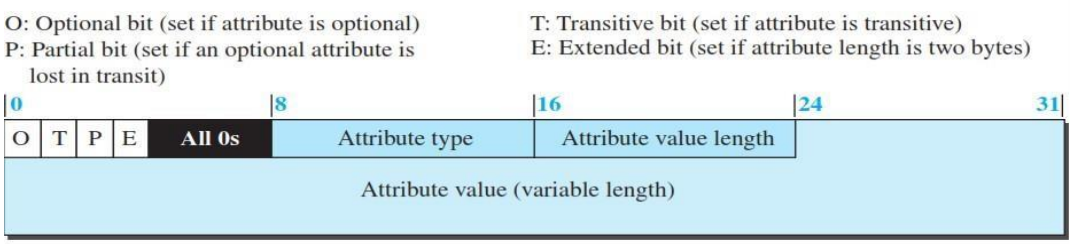


Figure 4. 31: Format of path attribute

The first byte in each attribute defines the four attribute flags (as shown in the figure). The next byte defines the type of attributes assigned by ICANN (only seven types have been assigned, as explained next). The attribute value length defines the length of the attribute value field (not the length of the whole attributes section). The following gives a brief description of each attribute.

❑**ORIGIN (type 1**). This is a well-known mandatory attribute, which defines the source of the routing information. This attribute can be defined by one of the three values: 1, 2, and 3. Value 1 means that the information about the path has been taken from an intradomain protocol (RIP or OSPF). Value 2 means that the informationcomesfromBGP.Value3meansthatit comes from an unknown source.

❑**AS-PATH (type 2).** This is a well-known mandatory attribute, which defines the list of autonomous systems through which the destination can be reached. We have used this attribute in our examples. The AS PATH attribute, as we discussed in path-vector routing in the last section, helps prevent a loop. Whenever an update message arrives at a router that lists the current AS as the path, the router drops that path. The AS-PATH can also be used in route selection.

❑**NEXT-HOP (type 3).** This is a well-known mandatory attribute, which defines the next router to which the data packet should be forwarded. This attribute helps to inject path information collected through the operations of eBGP and iBGP into the intradomain routing protocols such as RIP or OSPF.

❑**MULT-EXIT-DISC (type 4)**. The multiple-exit discriminator is an optional intransitive attribute, which discriminates among multiple exit paths to a destination. The value of this attribute is normally defined by the metric in the corresponding intradomain protocol (an attribute value of 4-byte unsigned integer). For example, if a router has multiple paths to the destination with different values related to these attributes, the one with the lowest value is selected. Note that this attribute is intransitive, which means that it is not propagated from one AS to another.

❑**LOCAL-PREF (type 5).** The local preference attribute is a well-known discretionary attribute. It is normally set by the administrator, based on the organization policy. The routes the administrator prefers are given a higher local preference value (an attribute value of 4-byte unsigned integer). For example, in an internet with five ASs, the administrator of AS1can set the local preference value of 400 to the path AS1→ AS2→AS5, the value of 300to AS1→AS3→AS5, and the value of 50toAS1 → AS4 → AS5. This means that the administrator prefers the first path to the second one and prefers the second one to the third one. This may be a case where AS2 is the mostsecuredandAS4isthe least secured AS for the administration of AS1.Thelastroute should be selected if the other two are not available.

❑**ATOMIC-AGGREGATE (type 6).** This is a well-known discretionary attribute, which defines the destination prefix as not aggregate; it only defines a single destination network. This attribute has no value field, which means the value of the length field is zero.

❑**AGGREGATOR (type 7).** This is an optional transitive attribute, which emphasizes that the destination prefix is an aggregate. The attribute value gives the number of the last AS that did the aggregation followed by the IP address of the router that did so.

Route Selection

In the case where multiple routes are received to a destination, BGP needs to select one among them. The route selection process in BGP is not as easy as the ones in the intradomain routing protocol that is based on the shortest-path tree. A route in BGP has some attributes attached to it and it may come

from an eBGP session or an iBGP session. Figure 4.32 shows the flow diagram as used by common implementations. The router extracts the routes which meet the criteria in each step. If only one route is extracted, it is selected and the process stops; otherwise, the process continues with the next step. Note that the first choice is related to the LOCALPREF attribute, which reflects the policy imposed by the administration on the route.

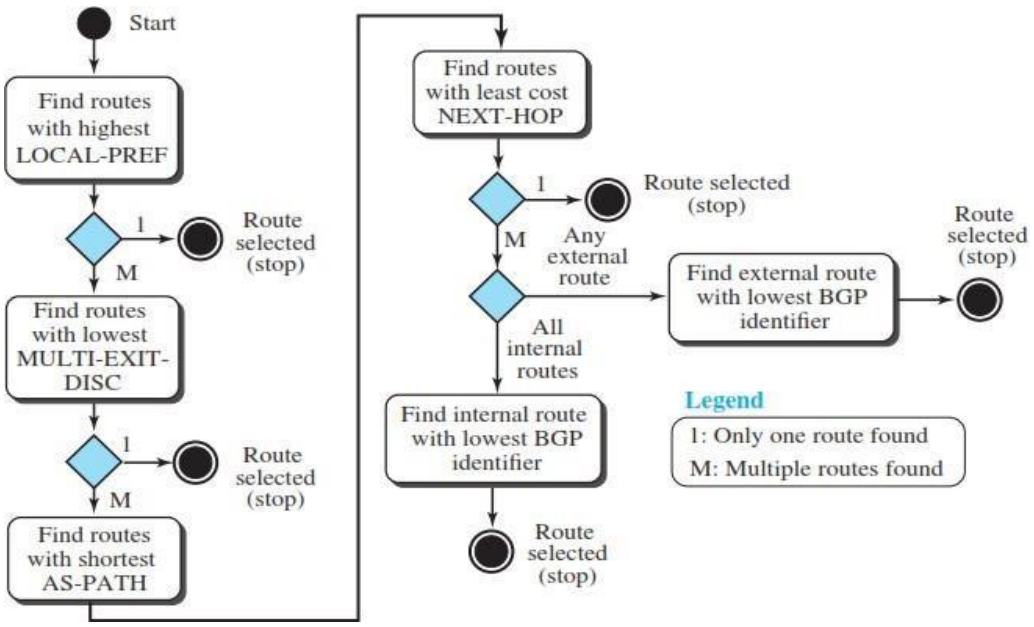


Figure 4.32: Flow diagram for route selection

Messages

BGP uses four types of messages for communication between the BGP speakers across the Ass and inside an AS: open, update, keep a live, and notification(seeFigure4.33).All BGP packets share the same common header.

❑**Open Message.** To create a neighbourhood relationship, a router running BGP opens a TCP connection with a neighbour and sends an open message.

❑**Update Message**. The update message is the heart of the BGP protocol. It is used by a router to withdraw destinations that have been advertised previously, to announce a route to a new destination, or both. Note that BGP can withdraw several destinations that were advertised before, but it can only advertise one new destination (or multiple destinations with the same path attributes) in a single update message.

❑**Keepalive Message**.TheBGPpeersthatarerunningexchangekeepalivemessagesregularly(before their hold time expires) to tell each other that they are alive.

❑**Notification**. A notification message is sent by a router whenever an error condition is detected or a router wants to close the session.

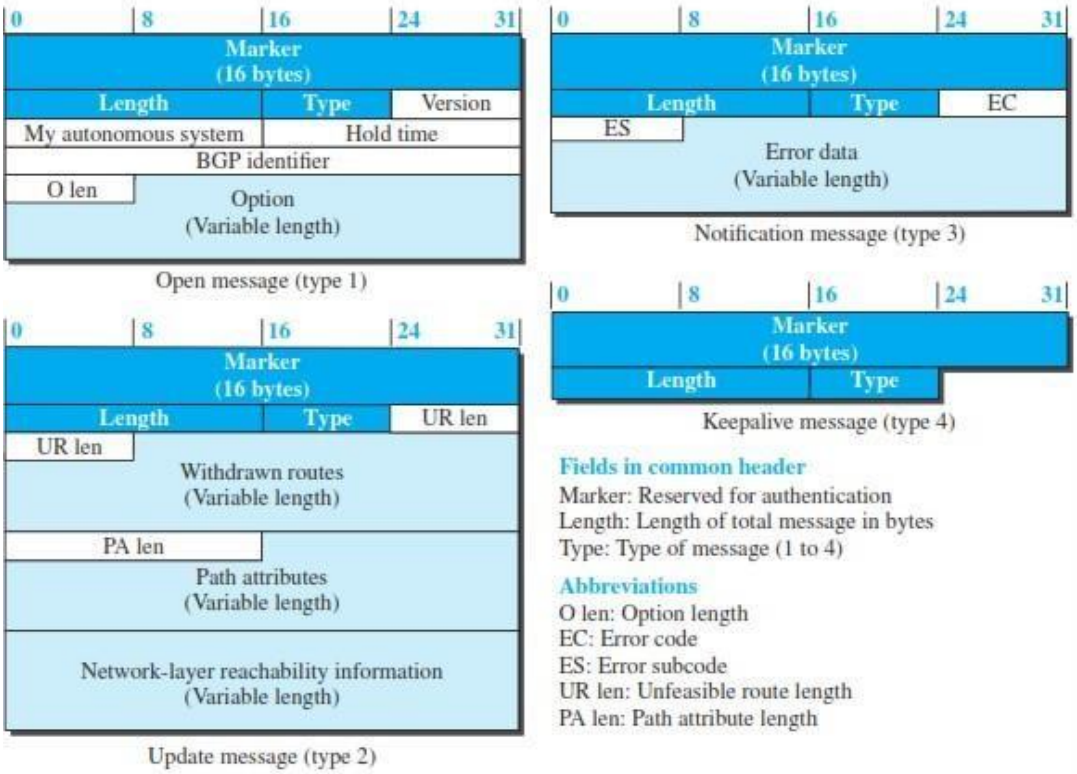


Figure 4.33: BGP messages

Performance

BGP performance can be compared with RIP. BGP speakers exchange a lot of messages to create forwarding tables, but BGP is free from loops and count-to-infinity. The same weakness we mention for RIP about propagation of failure and corruption also exists in BGP.

MODULE 4

1) What are the different services provided by the transport layer?

1. Process-to-Process Communication

The first duty of a transport-layer protocol is to provide **process-to-process communication.** A process is an application-layer entity (running program) that uses the services of the transport layer. The network layer is responsible for communication at the computer level (host-to-host communication). A network-layer protocol can deliver the message only to the destination computer. However, this is an incomplete delivery. The message still needs to be handed to the correct process. This is where a transport-layer protocol takes over. A transport-layer protocol is responsible for delivery of the message to the appropriate process. Figure shows the domains of a network layer and a transport layer.

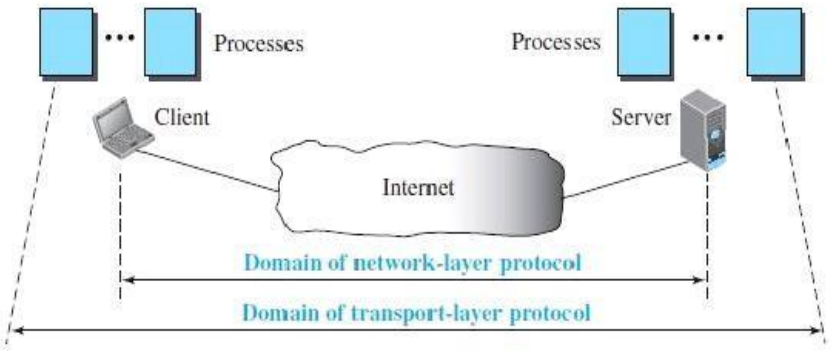


Figure 1: Network layer versus transport layer

2. Encapsulation and Decapsulation

To send a message from one process to another, the transport-layer protocol encapsulates and decapsulates messages (Figure 2). Encapsulation happens at the sender site. When a process has a message to send, it passes the message to the transport layer along with a pair of socket addresses and some other pieces of information, which depend on the transport-layer protocol. The transport layer receives the data and adds the transport-layer header. The packets at the transport layer in the Internet are called *user datagrams, segments,* or *packets,* depending on what transport-layer protocol we use. In general,transport-layer payloads are called as *packets.*

Decapsulation happens at the receiver site. When the message arrives at the destination transport layer, the header is dropped and the transport layer delivers the CCN Module 5: Transport

message to the process running at the application layer. The sender socket address is passed to the process in case it needs to respond to the message received.

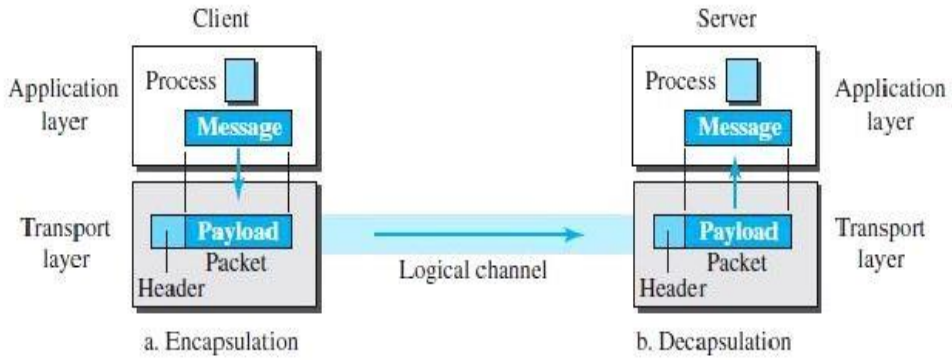


Figure 2: Encapsulation and decapsulation

3. Multiplexing and Demultiplexing

Whenever an entity accepts items from more than one source, this is referred to as multiplexing (many to one); whenever an entity delivers items to more than one source, this is referred to as demultiplexing (one to many). The transport layer at the source performs multiplexing; the transport layer at the destination performs demultiplexing

Figure 3 shows communication between a client and two servers. Three client processes are running at the client site, P1, P2, and P3. The processes P1 and P3 need to send requests to the corresponding server process running in a server. The client process P2 needs to send a request to the corresponding server process running at another server. The transport layer at the client site accepts three messages from the three processes and creates three packets. It acts as a multiplexer. The packets 1 and 3 use the same logical channel to reach the transport layer of the first server. When they arrive at the server, the transport layer does the job of a demultiplexer and distributes the messages to two different processes. The transport layer at the second server receives packet 2 and delivers it to the corresponding process. Note that we still have demultiplexing although there is only one message.

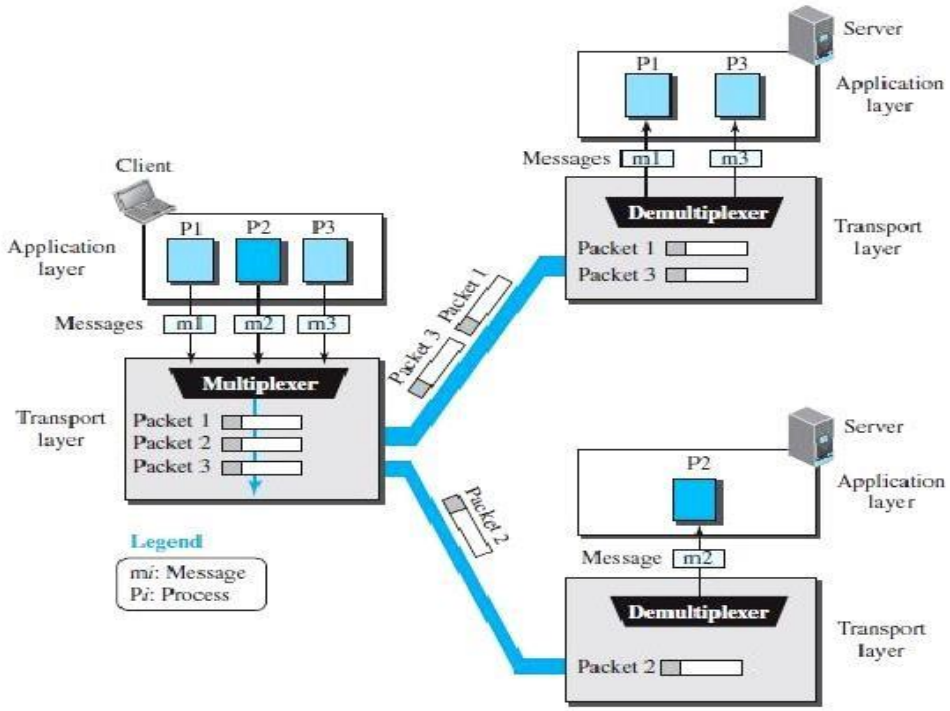


Figure 3: Multiplexing and demultiplexing

4. Flow Control

Whenever an entity produces items and another entity consumes them, there should be a balance between production and consumption rates. If the items are produced faster than they can be consumed, the consumer can be overwhelmed and may need to discard some items. If the items are produced more slowly than they can be consumed, the consumer must wait, and the system becomes less efficient. Flow control is related to the first issue. We need to prevent losing the data items at the consumer site.

Pushing or Pulling

Delivery of items from a producer to a consumer can occur in one of two ways: *pushing* or *pulling*. If the sender delivers items whenever they are produced without a prior request from the consumer the delivery is referred to as *pushing*. If the producer delivers the items after the consumer has requested them, the delivery is referred to as *pulling*. Figure 4 shows these two types of delivery.

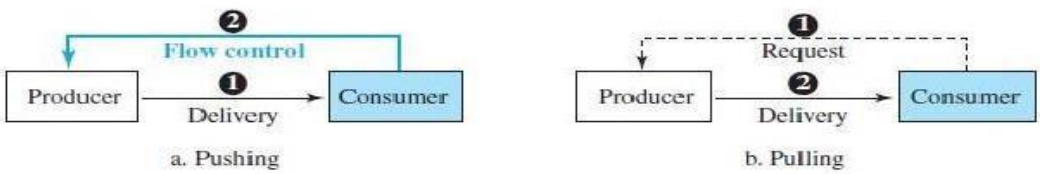


Figure 4: Pushing or pulling

Flow Control at Transport Layer

In communication at the transport layer, we are dealing with four entities: sender process, sender transport layer, receiver transport layer, and receiver process. The sending process at the application layer is only a producer. It produces message chunks and pushes them to the transport layer.

The sending transport layer has a double role: It is both a consumer and a producer. It consumes the messages pushed by the producer. It encapsulates the messages in packets and pushes them to the receiving transport layer. The receiving transport layer also has a double role: it is the consumer for the packets received from the sender and the producer that decapsulates the messages and delivers them to the application layer. The last delivery, however, is normally a pulling delivery; the transport layer waits until the application-layer process asks for messages.

Figure 5 shows that we need at least two cases of flow control: from the sending transport layer to the sending application layer and from the receiving transport layer to the sending transport layer.

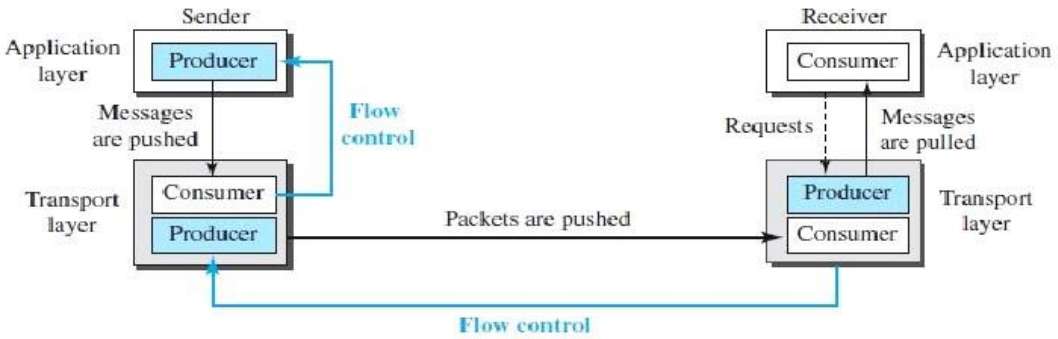


Figure 5: Flow control at the transport layer

5. Error Control

In the Internet, since the underlying network layer (IP) is unreliable, we need to make the transport layer reliable if the application requires reliability. Reliability can be achieved to add error control services to the transport layer. Error control at the transport layer is responsible for

1. Detecting and discarding corrupted packets.
2. Keeping track of lost and discarded packets and resending them.
3. Recognizing duplicate packets and discarding them.
4. Buffering out-of-order packets until the missing packets arrive.

Error control, unlike flow control, involves only the sending and receiving transport layers. Assume that the message chunks exchanged between the application and transport layers are error free. Figure 6 shows the error control between the sending and receiving transport layers. As with the case of flow control, the receiving transport layer manages error control, most of the time, by informing the sending transport layer about the problems.

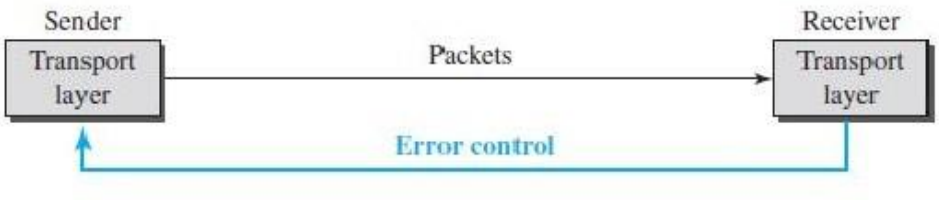


Figure 6: Error control at the transport layer

Sequence Numbers

Error control requires that the sending transport layer knows which packet is to be resent and the receiving transport layer knows which packet is a duplicate, or which packet has arrived out of order. This can be done if the packets are numbered. We can add a field to the transport-layer packet to hold the **sequence number** of the packet. When a packet is corrupted or lost, the receiving transport layer can somehow inform the sending transport layer to resend that packet using the sequence number. The receiving transport layer can also detect duplicate packets if two received packets have the same sequence number. The out-oforder packets can be recognized by observing gaps in the sequence numbers. Packets are numbered sequentially. However, because we need to include the sequence number of each packet in the header, we need to set a limit. If the header of the packet allows *m* bits for the sequence number, the sequence numbers range from 0 to 2^m-1.
For example, if *m* is 4, the only sequence numbers are 0 through 15, inclusive. However, we can wrap around the sequence. So the sequence numbers in this case are

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, ...

In other words, the sequence numbers are modulo 2^m.

Acknowledgment

The receiver side can send an acknowledgment (ACK) for each of a collection of packets that have arrived safe and sound. The receiver can simply discard the corrupted packets. The sender can detect lost packets if it uses a timer. When a packet is sent, the sender starts a timer. If an ACK does not arrive before the timer expires, the sender resends

the packet. Duplicate packets can be silently discarded by the receiver. Out-of-order packets can be either discarded (to be treated as lost packets by the sender), or stored until the missing one arrives.

6. Congestion Control

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 that control the congestion and keep the load below the capacity.

Congestion happens in any system that involves waiting. For example, congestion happens on a freeway because any abnormality in the flow, such as an accident during rush hour, creates blockage. Congestion in a network or internetwork occurs because routers and switches have queues—buffers that hold the packets before and after processing. A router, for example, has an input queue and an output queue for each interface. If a router cannot process the packets at the same rate at which they arrive, the queues become overloaded and congestion occurs. Congestion at the transport layer is actually the result of congestion at the network layer, which manifests itself at the transport layer.

2) Explain the concept of sliding window with a neatdiagram. ***Sliding Window***

Since the sequence numbers use modulo 2^m, a circle can represent the sequence numbers from 0 to $2^m - 1$ (Figure 7). The buffer is represented as a set of slices, called the ***sliding window,*** that occupies part of the circle at any time. At the sender site, when a packet is sent, the corresponding slice is marked. When all the slices are marked, it means that the buffer is full and no further messages can be accepted from the application layer.

When an acknowledgment arrives, the corresponding slice is unmarked. If some consecutive slices from the beginning of the window are unmarked, the window slides over the range of the corresponding sequence numbers to allow more free slices at the end of the window. Figure 7 shows the sliding window at the sender. The sequence numbers are in modulo 16 ($m = 4$) and the size of the window is 7. The sliding window is just an abstraction: the actual situation uses computer variables to hold the sequence numbers of the next packet to be sent and the last packet sent.

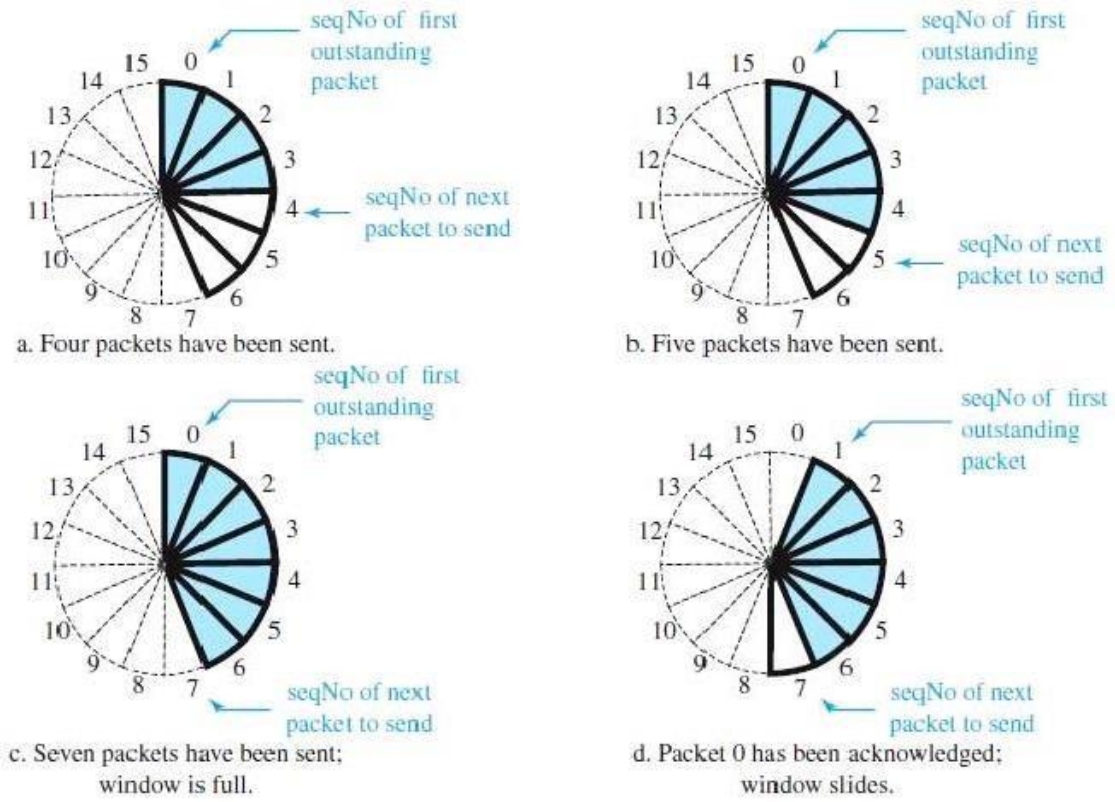


Figure 7: Sliding window in circular format

Most protocols show the sliding window using linear representation. The idea is the same, but it normally takes less space on paper. Figure 8 shows this representation.

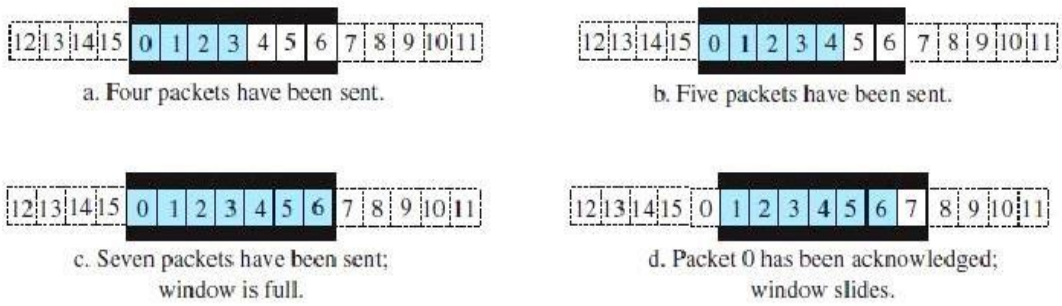


Figure 8: Sliding window in linear format

3. Write outline and explain send window and receive window for Go back N protocol/ selective repeat protocol.

Go-Back N protocol

To improve the efficiency of transmission (to fill the pipe), multiple packets must be in transition while the sender is waiting for acknowledgment. In other words, we need to let

more than one packet be outstanding to keep the channel busy while the sender is waiting for acknowledgment. One of the protocols is called ***Go-Back-N* (GBN)**. The key to Go-back-*N* is that we can send several packets before receiving acknowledgments, but the receiver can only buffer one packet. We keep a copy of the sent packets until the acknowledgments arrive. Figure 9 shows the outline of the protocol.

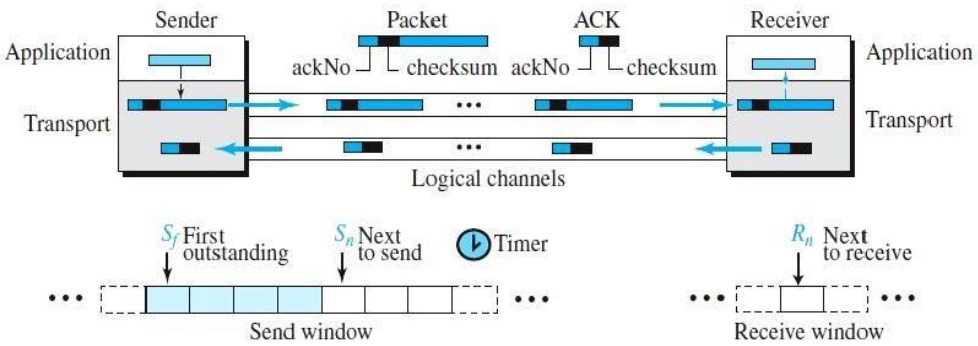


Figure 9: Go-Back-N protocol

Send Window

The send window is an imaginary box covering the sequence numbers of the data packets that can be in transit or can be sent. In each window position, some of the sequence numbers define the packets that have been sent; others define those that can be sent. The maximum size of the window is $2^m - 1$, we let the size be fixed and set to the maximum value, Figure 10 shows a sliding window of size 7 (m = 3) for the Go-Back-N protocol.

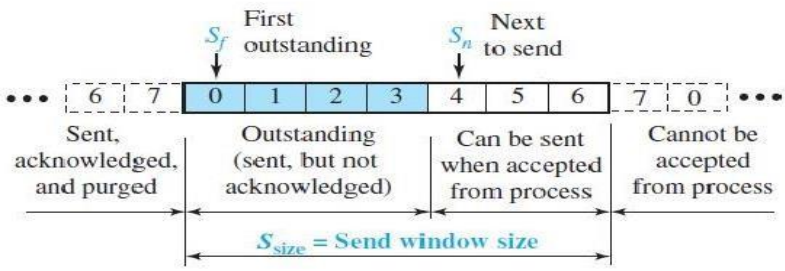


Figure 10: Send window for Go-Back-N

The send window at any time divides the possible sequence numbers into four regions. The first region, left of the window, defines the sequence numbers belonging to packets that are already acknowledged. The sender does not worry about these packets and keeps no copies of them. The second region, colored, defines the range of sequence numbers belonging to the packets that have been sent, but have an unknown status. The sender needs to wait to find out if these

packets have been received or were lost. These are called as *outstanding* packets. The third range, white in the figure, defines the range of sequence numbers for packets that can be sent; however, the corresponding data have not yet been received from the application layer. Finally, the fourth region, right of the window, defines sequence numbers that cannot be used until the window slides.

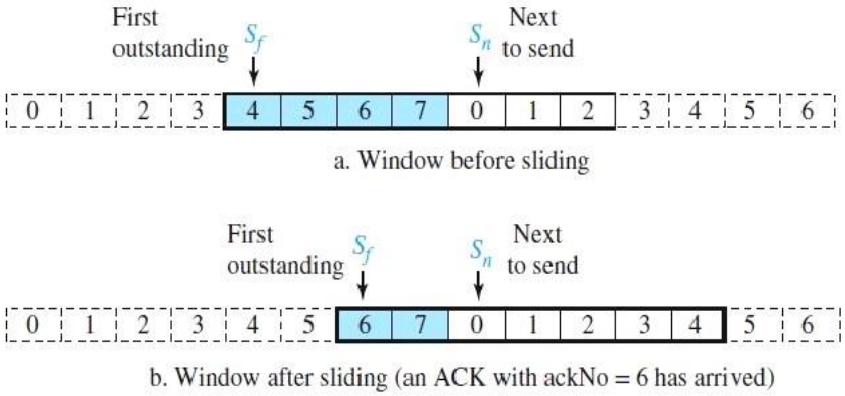


Figure 11: Sliding the send window

Figure 11 shows how a send window can slide one or more slots to the right when an acknowledgment arrives from the other end. In the figure, an acknowledgment with ack No = 6 has arrived. This means that the receiver is waiting for packets with sequence no 6.

Receive Window

The receive window makes sure that the correct data packets are received and that the correct acknowledgments are sent. In Go-Back-N, the size of the receive window is always 1. The receiver is always looking for the arrival of a specific packet. Any packet arriving out of order is discarded and needs to be resent. Figure 12 shows the receive window. It needs only one variable, Rn (receive window, next packet expected), to define this abstraction. The sequence numbers to the left of the window belong to the packets already received and acknowledged; the sequence numbers to the right of this window define the packets that cannot be received. Any received packet with a sequence number in these two regions is discarded. Only a packet with a sequence number matching the value of Rn is accepted and acknowledged. The receive window also slides, but only one slot at a time. When a correct packet is received, the window slides, Rn = (Rn + 1) modulo 2^m.

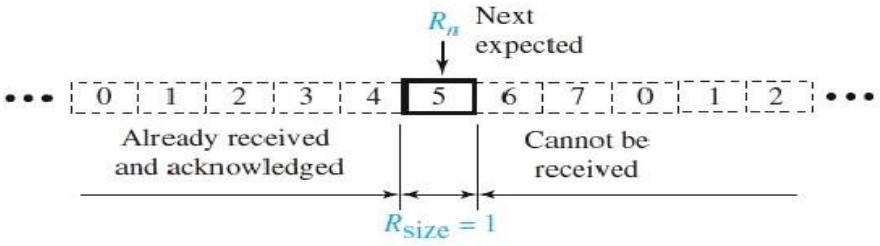


Figure 12: Receive window for Go-Back-N

FSMs

Figure 13 shows the FSMs for the GBN protocol.

Sender

The sender starts in the ready state, but thereafter it can be in one of the two states: ready or blocking. The two variables are normally initialized to 0 (Sf = Sn = 0).

- **Ready state.** Four events may occur when the sender is in ready state.

a. If a request comes from the application layer, the sender creates a packet with the sequence number set to Sn. A copy of the packet is stored, and the packet is sent. The sender also starts the only timer if it is not running. The value of Sn is now incremented, (Sn = Sn + 1) modulo 2^m. If the window is full, Sn = (Sf + Ssize) modulo 2^m, the sender goes to the blocking state.

b. If an error-free ACK arrives with ackNo related to one of the outstanding packets,the sender slides the window (set Sf = ackNo), and if all outstanding packets are acknowledged (ackNo = Sn), then the timer is stopped. If all outstanding packets are not acknowledged, the timer is restarted.

c. If a corrupted ACK or an error-free ACK with ack number not related to the outstanding packet arrives, it is discarded.

d. If a time-out occurs, the sender resends all outstanding packets and restarts the timer.

- **Blocking state.** Three events may occur in this case:

a. If an error-free ACK arrives with ackNo related to one of the outstanding packets, the sender slides the window (set Sf = ackNo) and if all outstanding **p**ackets are acknowledged (ackNo = Sn), then the timer is stopped. If all outstanding packets are not acknowledged, the timer is restarted. The sender then moves to the ready state.

b. If a corrupted ACK or an error-free ACK with the ackNo not related to the outstanding packets arrives, the ACK is discarded.

c. If a time-out occurs, the sender sends all outstanding packets and restarts the timer.

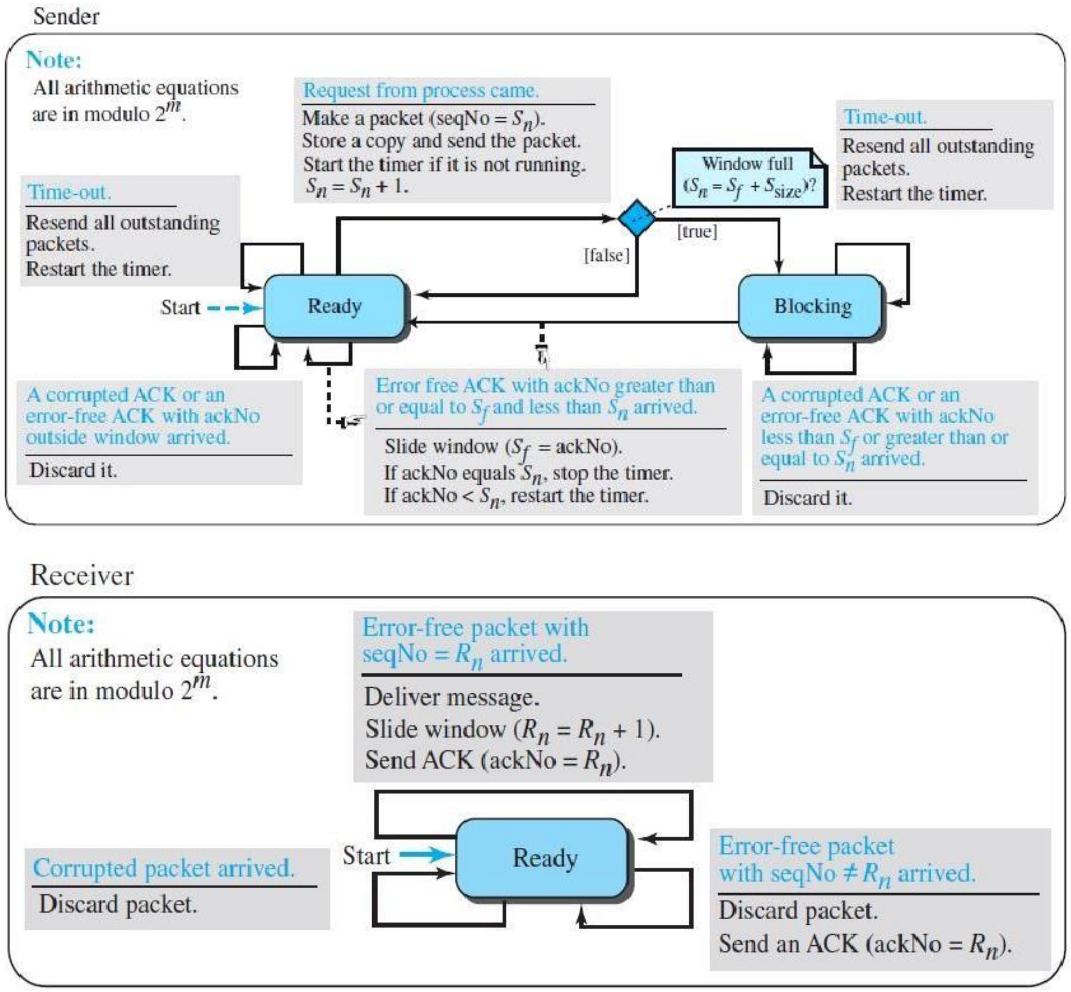


Figure 13 FSMs for the Go-Back-N protocol

Receiver

The receiver is always in the ready state. The only variable, Rn, is initialized to 0. Three events may occur:

a. If an error-free packet with seq No = Rn arrives, the message in the packet is delivered to the application layer. The window then slides, Rn = (Rn + 1) modulo 2m.

Finally an ACK is sent with ack No = Rn.

b. If an error-free packet with seq No outside the window arrives, the packet is discarded, but an ACK with ack No = Rn is sent.

c. If a corrupted packet arrives, it is discarded.

Send Window Size

The size of the send window must be less than 2^m is because for example, choose m = 2, which means the size of the window can be 2^m - 1, or 3. Figure 14 compares a window sizeof 3 against a window size of 4. If the size of the window is 3 (less than 2^m) and all three acknowledgments are lost, the only timer expires and all three packets are resent. The receiver is now expecting packet 3, not packet 0, so the duplicate packet is correctly discarded. On the other hand, if the size of the window is 4 (equal to 22) and all acknowledgments are lost, the sender will send a duplicate of packet 0. However, this time the window of the receiver expects to receive packet 0 (in the next cycle), so it accepts packet 0, not as a duplicate, but as the first packet in the next cycle. This is an error. This shows that the size of the send window must be less than 2^m.

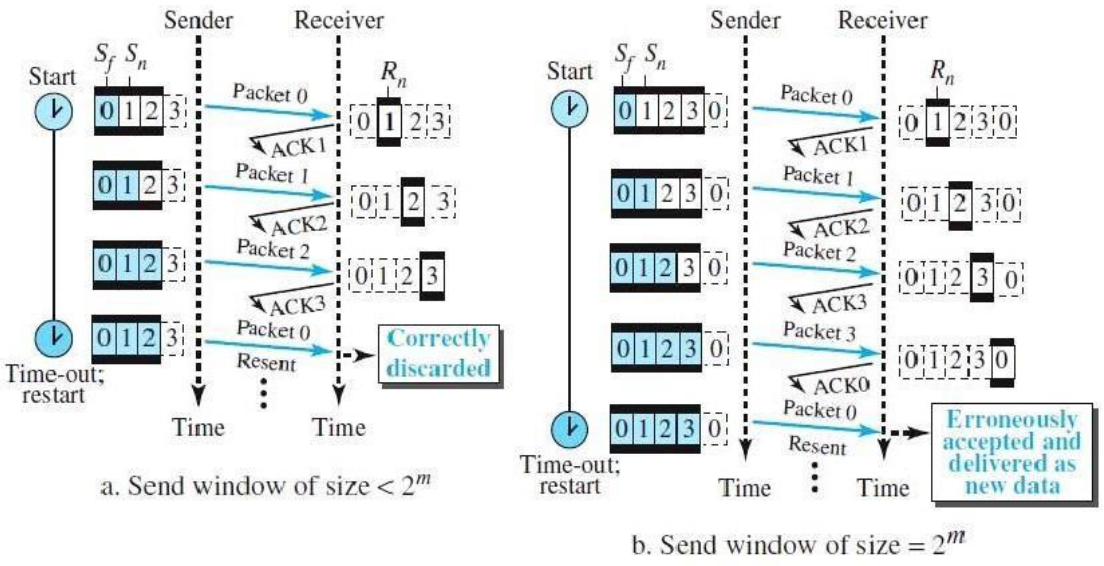


Figure 14 Send window size for Go-Back-N

Example 1

Figure 15 shows an example of Go-Back-*N*. This is an example of a case where the forward channel is reliable, but the reverse is not. No data packets are lost, but some ACKs are delayed and one is lost. The example also shows how cumulative acknowledgments can help if acknowledgments are delayed or lost.

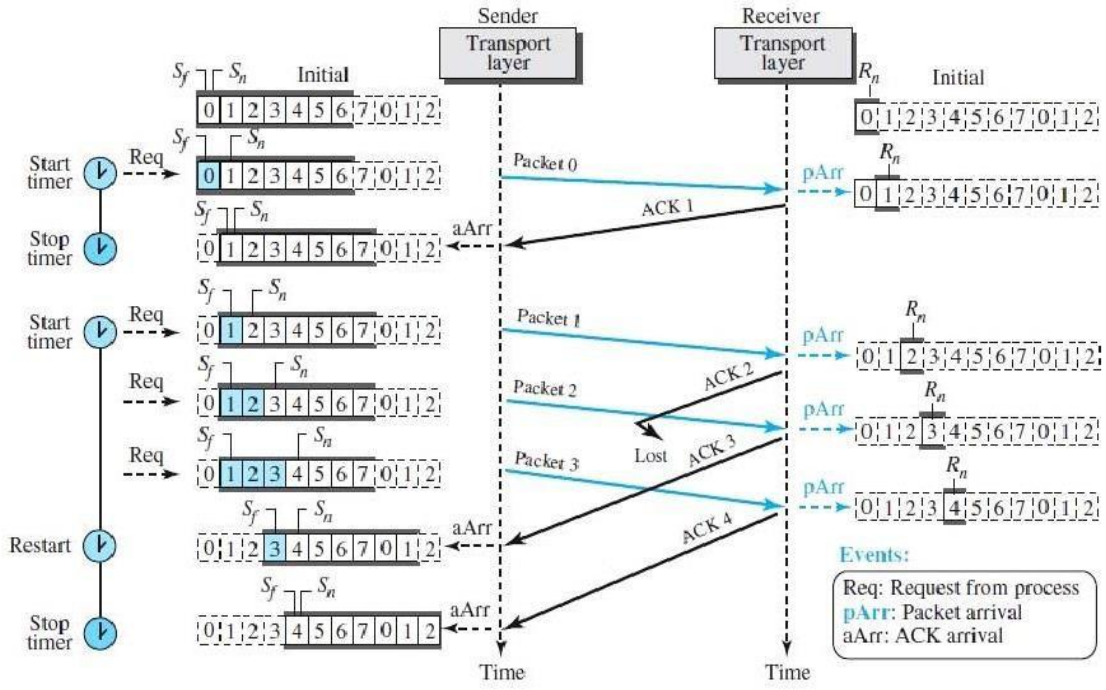


Figure15: Flow diagram for Example 1

After initialization, there are some sender events. Request events are triggered by message chunks from the application layer; arrival events are triggered by ACKs received from the network layer. There is no time-out event here because all outstanding packets are acknowledged before the timer expires. Although ACK 2 is lost, ACK 3 is cumulative and serves as both ACK 2 and ACK 3. There are four events at the receiver site.

Example 2

Figure 16 shows what happens when a packet is lost. Packets 0, 1, 2, and 3 are sent.
However, packet 1 is lost. The receiver receives packets 2 and 3, but they are discarded because they are received out of order (packet 1 is expected). When the receiver receives packets 2 and 3, it sends ACK1 to show that it expects to receive packet 1. However, these ACKs are not useful for the sender because the ackNo is equal to *Sf* , not greater than *Sf*. So the sender discards them. When the time-out occurs, the sender resends packets 1, 2, and 3, which are acknowledged.

Go-Back-N versus Stop-and-Wait

The Stop-and-Wait protocol is actually a Go-Back-N protocol in which there are only two sequence numbers and the send window size is 1. In other words, m = 1 and $2^m - 1 = 1$. In Go-

Back-N, we said that the arithmetic is modulo 2^m; in Stop-and-Wait it is modulo 2, which is the same as 2^m when m =1.

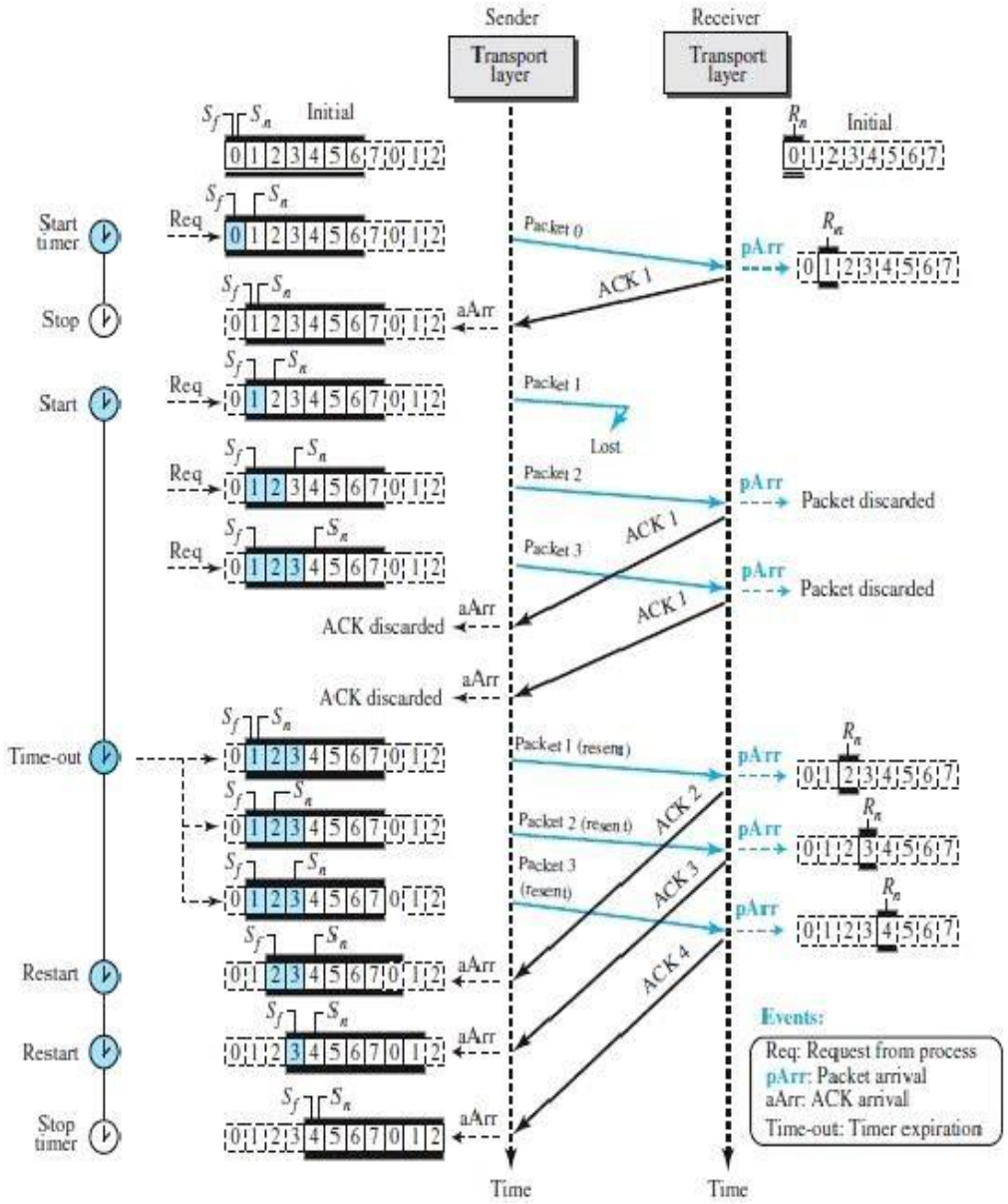


Figure 16: Flow diagram for Example 2

Selective-Repeat Protocol

The Go-Back-*N* protocol simplifies the process at the receiver. The receiver keeps track of only one variable, and there is no need to buffer out-of-order packets; they are simply discarded. However, this protocol is inefficient if the underlying network protocol loses a lot of packets. Each time a single packet is lost or corrupted, the sender resends all outstanding packets, even though some of these packets may have been received safe and sound but out of order. If the network layer is losing many packets because of congestion in the network, the resending of all of these outstanding packets makes the congestion worse, and eventually more packets are lost. This has an avalanche effect that may result in the total collapse of the network.

Another protocol, called the **Selective-Repeat (SR) protocol,** has been devised, which, as the name implies, resends only selective packets, those that are actually lost.

The outline of this protocol is shown in Figure 17.

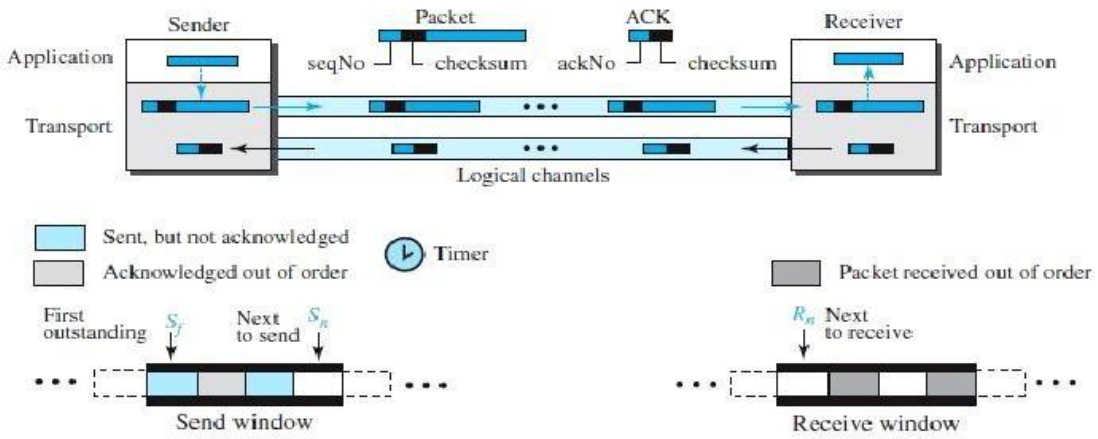


Figure 17: Outline of Selective-Repeat

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 maximum size of the send window is much smaller; it is 2^m-1. The reason for this will be discussed later. Second, the receive window is the same size as the send window.

The send window maximum size can be 2^m-1. For example, if m = 4, the sequence numbers go from 0 to 15, but the maximum size of the window is just 8 (it is 15 in the GoBack-N Protocol). The Selective-Repeat send window in Figure 18.1 to emphasize the size.

The receive window in Selective-Repeat is totally different from the one in Go-Back-N. The size of the receive window is the same as the size of the send window (max 2^m-1). The Selective-Repeat protocol allows as many packets as the size of the receive window to arrive out

of order and be kept until there is a set of consecutive packets to be delivered to the application layer. Because the sizes of the send window and receive window are the same, all the packets in the send packet can arrive out of order and be stored until they can be delivered. To emphasize that in a reliable protocol the receiver never delivers packets out of order to the application layer.

Figure 18.2 shows the receive window in Selective-Repeat. Those slots inside the window that are shaded define packets that have arrived out of order and are waiting for the earlier transmitted packet to arrive before delivery to the application layer.

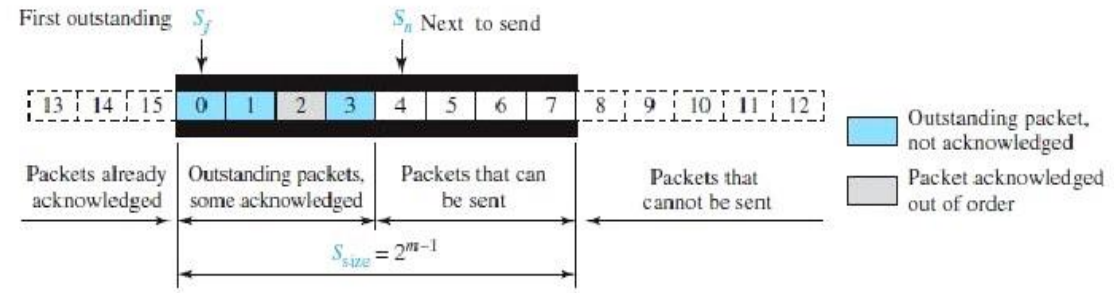


Figure 18.1: Send window for Selective-Repeat protocol

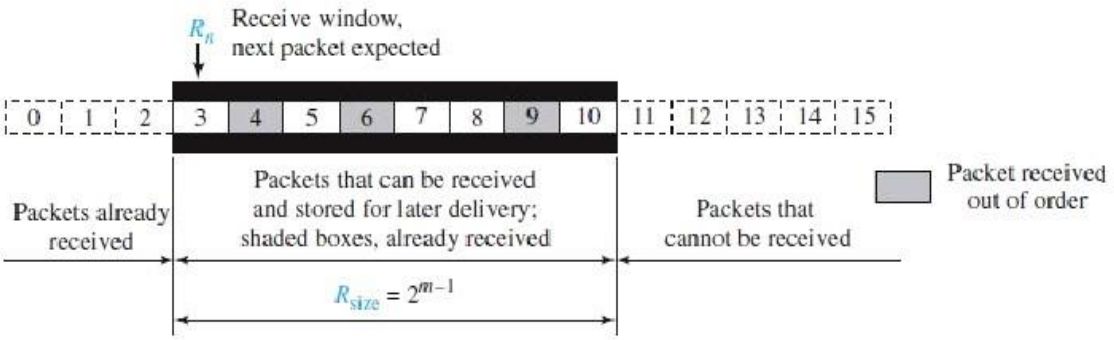


Figure 18.2: Receive window for Selective-Repeat protocol

Example 3

Assume a sender sends 6 packets: packets 0, 1, 2, 3, 4, and 5. The sender receives an ACK with ackNo = 3. What is the interpretation if the system is using GBN or SR?

Solution

If the system is using GBN, it means that packets 0, 1, and 2 have been received uncorrupted and the receiver is expecting packet 3. If the system is using SR, it means that packet 3 has been received uncorrupted; the ACK does not say anything about other packets.

FSMs

Figure 19 shows the FSMs for the Selective-Repeat protocol. It is similar to the ones for the GBN, but there are some differences.

Sender

The sender starts in the ready state, but later it can be in one of the two states: ready or blocking. The following shows the events and the corresponding actions in each state.

- **Ready state.** Four events may occur in this case:

a. If a request comes from the application layer, the sender creates a packet with the sequence number set to *Sn*. A copy of the packet is stored, and the packet is sent. If the timer is not running, the sender starts the timer. The value of *Sn* is now incremented, *Sn* = (*Sn* + 1) modulo 2*m*. If the window is full, *Sn* = (*Sf* + *S*size) modulo 2*m*, the sender goes to the blocking state.

b. If an error-free ACK arrives with ackNo related to one of the outstanding packets, that packet is marked as acknowledged. If the ackNo = *Sf*, the window slides to the right until the *Sf* points to the first unacknowledged packet (all consecutive acknowledged packets are now outside the window). If there are outstanding packets, the timer is restarted; otherwise, the timer is stopped.

c. If a corrupted ACK or an error-free ACK with ackNo not related to an outstanding packet arrives, it is discarded.

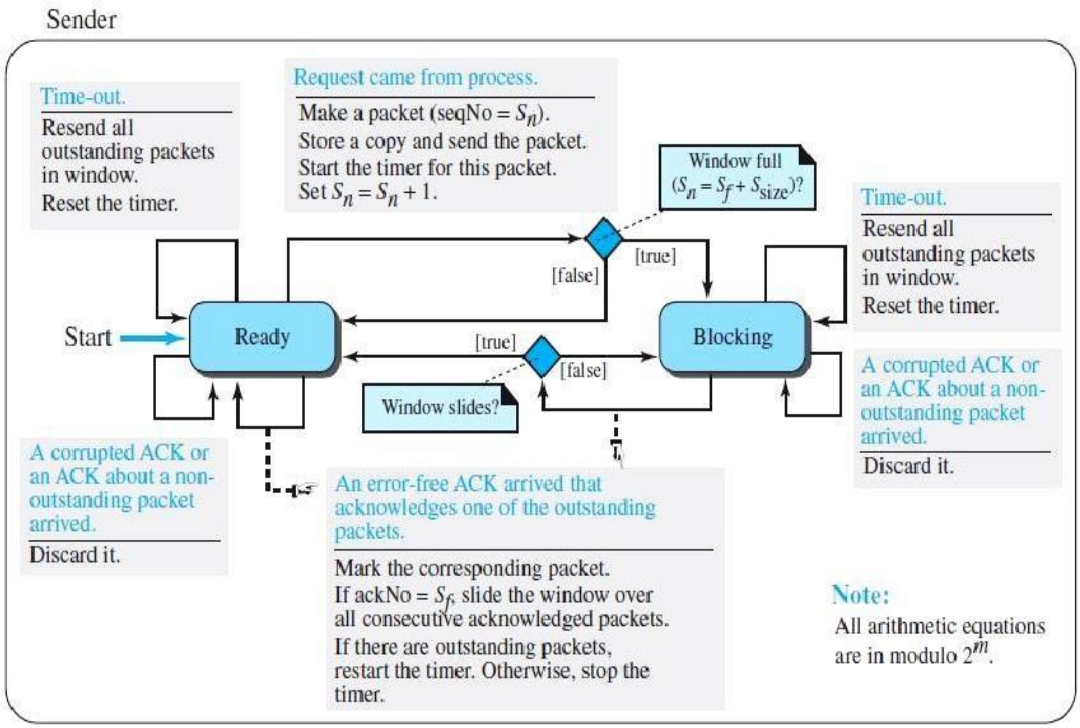
d. If a time-out occurs, the sender resends all unacknowledged packets in the window and restarts the timer.

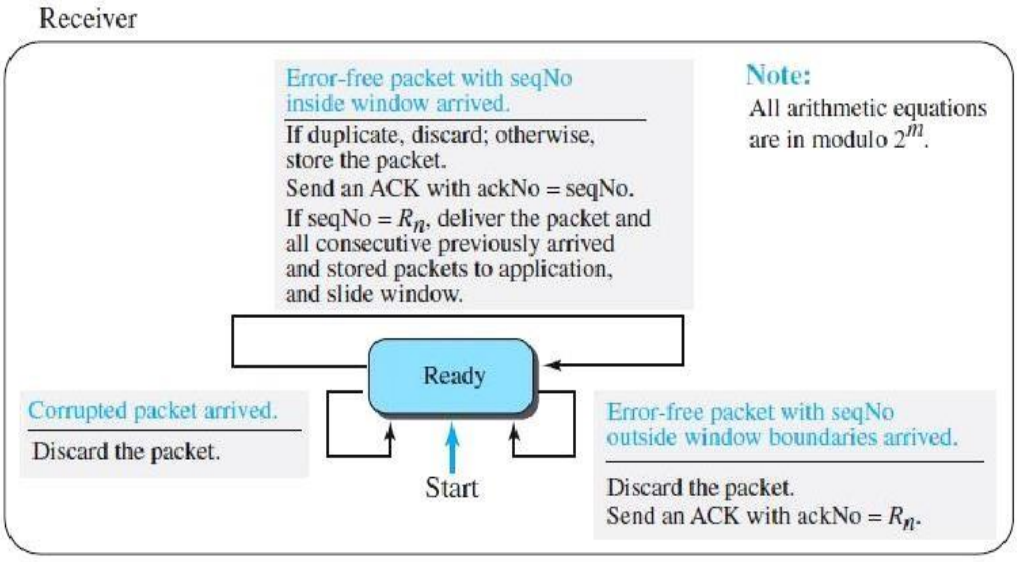
- **Blocking state.** Three events may occur in this case:

a. If an error-free ACK arrives with ackNo related to one of the outstanding packets, that packet is marked as acknowledged. In addition, if the ackNo = *Sf*, the window is slid to the right until the *Sf* points to the first unacknowledged packet (all consecutive acknowledged packets are now outside the window). If the window has slid, the sender moves to the ready state.

b. If a corrupted ACK or an error-free ACK with the ackNo not related to outstanding packets arrives, the ACK is discarded.

c. If a time-out occurs, the sender resends all unacknowledged packets in the window and restarts the timer.





Receiver

The receiver is always in the ready state. Three events may occur:

a. If an error-free packet with seqNo in the window arrives, the packet is stored and an ACK with ackNo = seqNo is sent. In addition, if the seqNo = *Rn*, then the packet and all previously arrived consecutive packets are delivered to the application layer and the window slides so that the *Rn* points to the first empty slot.

b. If an error-free packet with seqNo outside the window arrives, the packet is discarded, but an ACK with ackNo = *Rn* is returned to the sender. This is needed to let the sender slide its window if some ACKs related to packets with seqNo < *Rn* were lost.

c. If a corrupted packet arrives, the packet is discarded.

Example 4

This example is sim lar to Example 2 (Figure 16) in which packet 1 is lost. Selective-Repeat behaviour is shown in this case. Figure 19 shows the situation.

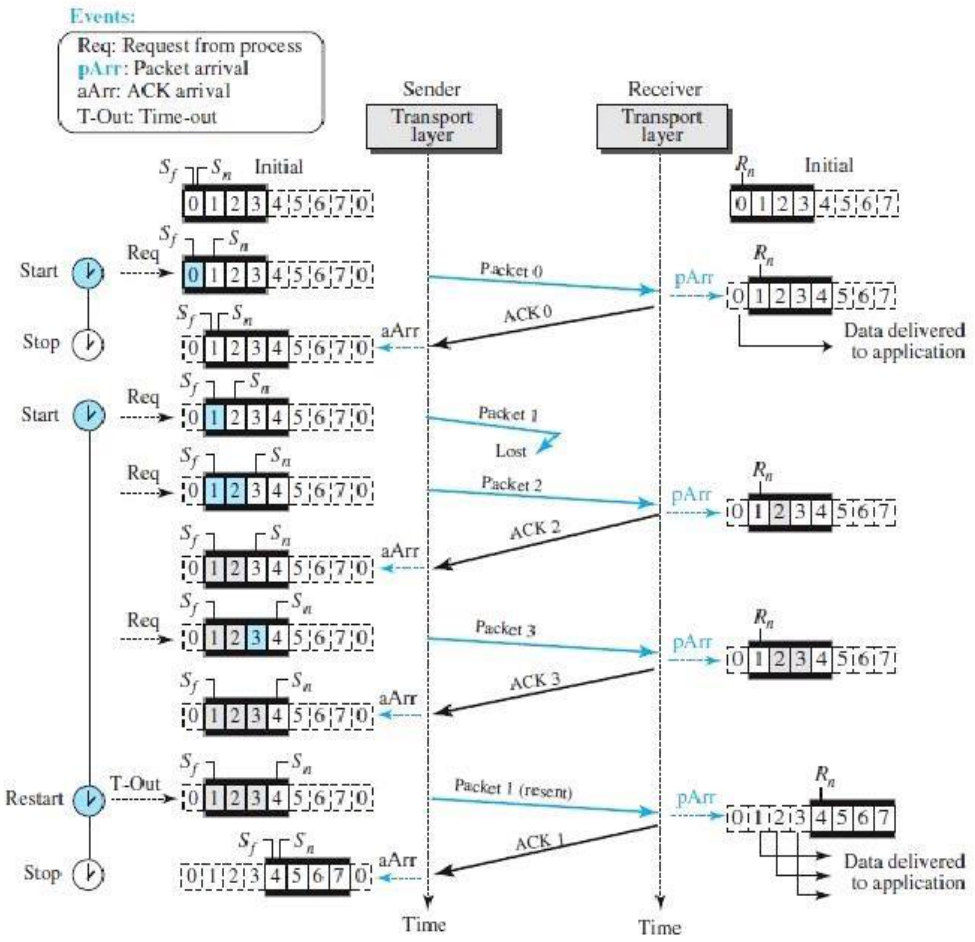


Figure 19: Flow diagram for Example 4

At the sender, packet 0 is transmitted and acknowledged. Packet 1 is lost. Packets 2 and 3 arrive out of order and are acknowledged. When the timer times out, packet 1 (the only unacknowledged packet) is resent and is acknowledged. The send window then slides.

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

beginning of the window. The key is that a reliable transport layer promises to deliver packets in order.

Window Sizes

We can now show why the size of the sender and receiver windows can be at most one-half of 2^m. For an example, we choose $m = 2$, which means the size of the window is $2^m/2$ or $2^{(m-1)} = 2$. Figure 23.36 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 packet 0 expires and packet 0 is resent. However, the window of the receiver is now expecting packet 2, not packet 0, so this duplicate packet is correctly discarded (the sequence number 0 is not in the window). When the size of the window is 3 and all acknowledgments are lost, the sender sends a duplicate of packet 0. However, this time, the window of the receiver expects to receive packet 0 (0 is part of the window), so it accepts packet 0, not as a duplicate, but as a packet in the next cycle.
This is clearly an error.

4. What are the services provided by UDP? Mention any four typical applications of UDP.

Process-to-Process Communication

UDP provides process-to-process communication using **socket addresses,** a combination of IP addresses and port numbers.

Connectionless Services

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, unlike TCP, there is no connection establishment and no connection termination. 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, messages less than 65,507 bytes (65,535 minus 8 bytes for the UDP header and minus 20 bytes for the IP header), can use UDP.

Flow Control

UDP is a very simple protocol. There is no flow control, and hence no window mechanism. The receiver may overflow with incoming messages. The lack of flow control means that the process using UDP should provide for this service, if needed.

Error Control

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 error control means that the process using UDP should provide for this service, if needed.

Checksum

UDP checksum calculation includes three sections: a pseudo header, the UDP header, and the data coming from the application layer. The pseudo header is the part of the header of the IP packet in which the user datagram is to be encapsulated with some fields filled with 0s (see Figure 20).

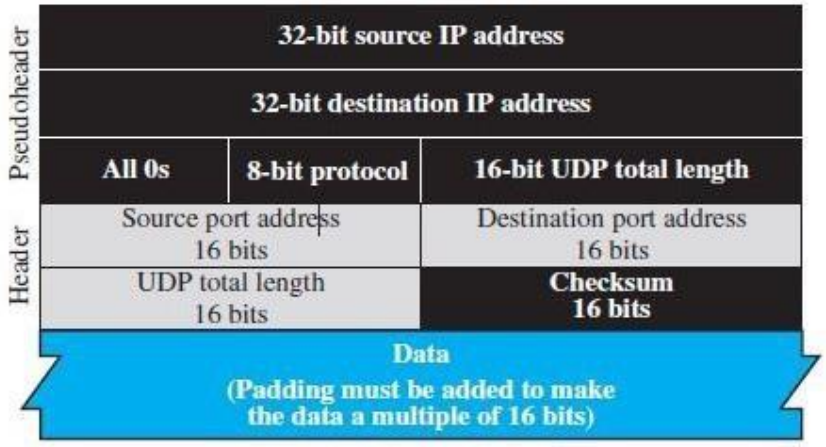


Figure 20: Pseudo header for checksum calculation

If the checksum does not include the pseudo header, a user datagram may arrive safe and sound. However, if the IP header is corrupted, it may be delivered to the wrong host. The protocol field is added to ensure that the packet belongs to UDP, and not to TCP. We will see later that if a process can use either UDP or TCP, the destination port number can be the same.

The value of the protocol field for UDP is 17. If this value is changed during

transmission, the checksum calculation at the receiver will detect it and UDP drops the packet. It is not delivered to the wrong protocol.

Congestion Control

Since UDP is a connectionless protocol, it does not provide congestion control. UDP assumes that the packets sent are small and sporadic and cannot create congestion in the network. This assumption may or may not be true today, when UDP is used for interactive real-time transfer of audio and video.

Encapsulation and Decapsulation

To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages.

Queuing

In UDP, queues are associated with ports. At the client site, when a process starts, it requests a port number from the operating system. Some implementations create both an incoming and an outgoing queue. associated with each process. Other implementations create only an incoming queue associated with each process.

Multiplexing and Demultiplexing

In a host running a TCP/IP protocol suite, there is only one UDP but possibly several processes that may want to use the services of UDP. To handle this situation, UDP multiplexes and demultiplexes.

Typical Applications

The following shows some typical applications that can benefit more from the services of UDP than from those of TCP.

< UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control. It is not usually used for a process such as FTP that needs to send bulk data.

< 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. It can easily use UDP.

< UDP is a suitable transport protocol for multicasting. 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 (RIP).

< UDP is normally used for interactive real-time applications that cannot tolerate uneven delay between sections of a received message.

5. What are the different TCP services and features? Explain them

- **Process-to-Process Communication**

TCP provides process-to-process communication using port numbers.

- **Stream Delivery Service**

TCP, unlike UDP, is a stream-oriented protocol. In UDP, a process sends messages with predefined boundaries to UDP for delivery. UDP adds its own header to each of these messages and delivers it to IP for transmission. Each message from the process is called a user datagram, and becomes, eventually, one IP datagram. Neither IP nor UDP recognizes any relationship between the datagrams.

TCP, on the other hand, allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes. TCP creates an environment in which the two processes seem to be connected by an imaginary "tube" that carries their bytes across the Internet. This imaginary environment is depicted in Figure 21. The sending process produces (writes to) the stream and the receiving process consumes (reads from) it.

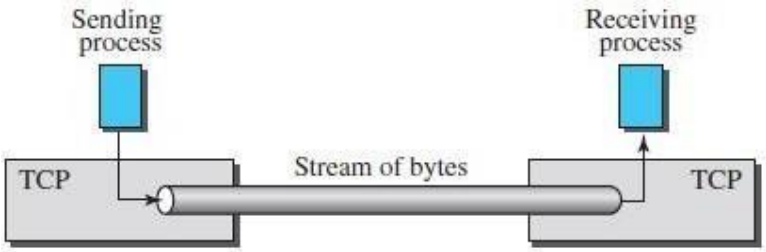


Figure 21: Stream delivery

- **Sending and Receiving Buffers**

One way to implement a buffer is to use a circular array of 1-byte locations as shown in Figure 22. For simplicity, it is shown as two buffers of 20 bytes each; normally the buffers are hundreds or thousands of bytes, depending on the implementation. We also show the buffers as the same size, which is not always the case. The figure shows the movement of the data in one direction. At the sender, the buffer has three types of chambers. The white section contains empty chambers that can be filled by the sending process (producer). The colored area holds bytes that have been sent but not yet acknowledged. The TCP sender keeps these CCN Module 5:

bytes in the buffer until it receives an acknowledgment. The shaded area contains bytes to be sent by the sending TCP. However, as we will see later in this chapter, TCP may be able to send only part of this shaded section. This could be due to the slowness of the receiving process or to congestion in the network. Also note that, after the bytes in the colored chambers are acknowledged, the chambers are recycled and available for use by the sending process.

The operation of the buffer at the receiver is simpler. The circular buffer is divided into two areas (shown as white and colored). The white area contains empty chambers to be filled by bytes received from the network. The colored sections contain received bytes that can be read by the receiving process. When a byte is read by the receiving process, the chamber is recycled and added to the pool of empty chambers.

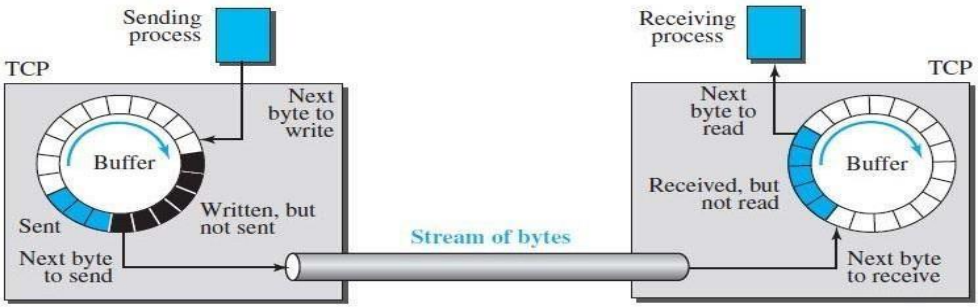


Figure 22: Sending and receiving buffers

- **Segments**

Although buffering handles the disparity between the speed of the producing and consuming processes, we need one more step before we can send data. The network layer, as a service provider for TCP, needs to send data in packets, not as a stream of bytes. At the transport layer, TCP groups a number of bytes together into a packet called a segment.

TCP adds a header to each segment (for control purposes) and delivers the segment to the network layer for transmission. The segments are encapsulated in an IP datagram and transmitted. This entire operation is transparent to the receiving process. Segments may be received out of order, lost or corrupted, and resent. All of these are handled by the TCP receiver with the receiving application process unaware of TCP's activities. Figure 23 shows how segments are created from the bytes in the buffers.

Segments are not necessarily all the same size. In the figure, for simplicity, it is shown one segment carrying 3 bytes and the other carrying 5 bytes. In reality, segments carry hundreds, if not thousands, of bytes.

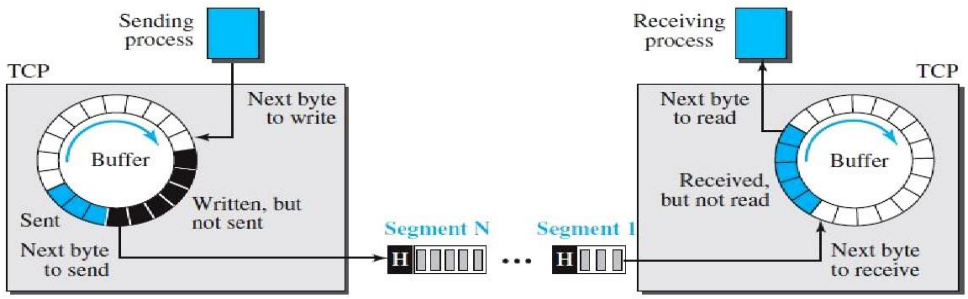


Figure 23: TCP segments

Full-Duplex Communication

TCP offers full-duplex service, where data can flow in both directions at the same time. Each TCP endpoint then has its own sending and receiving buffer, and segments move in both directions.

- **Multiplexing and Demultiplexing**

Like UDP, TCP performs multiplexing at the sender and demultiplexing at the receiver. However, since TCP is a connection-oriented protocol, a connection needs to be established for each pair of processes.

- **Connection-Oriented Service**

TCP, unlike UDP, is a connection-oriented protocol. When a process at site A wants to send to and receive data from another process at site B, the following three phases occur:

1. The two TCPs establish a logical connection between them.
2. Data are exchanged in both directions.
3. The connection is terminated.

This is a logical 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. Each may be routed over a different path to reach the destination. There is no physical connection. TCP creates a stream-oriented environment in which it accepts the responsibility of delivering the bytes in order to the other site.

- **Reliable Service**

TCP is a reliable transport protocol. It uses an acknowledgment mechanism to check the safe and sound arrival of data.

TCP Features

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 a byte number and not a segment number.

Byte Number

TCP numbers all data bytes (octets) that are transmitted in a connection. Numbering is independent in each direction. When TCP receives bytes of data from a process, TCP stores them in the sending buffer and numbers them. The numbering does not necessarily start from 1.Instead, TCP chooses an arbitrary number between 0 and 2^{32} - 1 for the number of the first byte. For example, if the number happens to be 1057 and the total data to be sent is 6000 bytes, the bytes are numbered from 1057 to 7056. We will see that byte numbering is used for flow and error control.

Sequence Number

After the bytes have been numbered, TCP assigns a sequence number to each segment that is being sent. The sequence number, in each direction, is defined as follows:

1. The sequence number of the first segment is the ISN (initial sequence number), which is a random number.
2. The sequence number of any other segment is the sequence number of the previous segment plus the number of bytes (real or imaginary) carried by the previous segment.

Acknowledgment Number

Communication in TCP is full duplex; when a connection is established, both parties can send and receive data at the same time. Each party numbers the bytes, usually with a different starting byte number. The sequence number in each direction shows the number of the first byte carried by the segment. Each party also uses an acknowledgment number to confirm the bytes it has received. However, the acknowledgment number defines the number of the next byte that the party expects to receive. In addition, the acknowledgment number is cumulative, which means

that the party takes the number of the last byte that it has received, safe and sound, adds 1 to it, and announces this sum as the acknowledgment number. The term *cumulative* here means that if a party uses 5643 as an acknowledgment number, it has received all bytes from the beginning up to 5642. Note that this does not mean that the party has received 5642 bytes, because the first byte number does not have to be 0.

6. With a neat diagram explain TCP segment format

Segment

A packet in TCP is called a segment.

Format

The format of a segment is shown in Figure 23.1. The segment consists of a header of 20 to 60 bytes, 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.

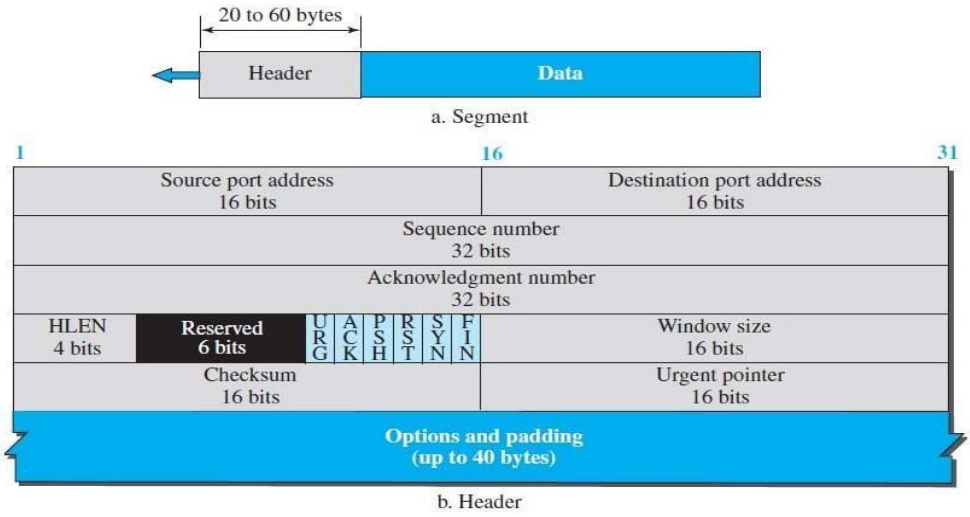


Figure 23.1: TCP segment format

< 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.

< 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.

< Sequence number. This 32-bit field defines the number assigned to the first byte of data contained in this segment. As we said before, TCP is a stream transport protocol. To ensure connectivity, each byte to be transmitted is numbered. The sequence number tells the destination which byte in this sequence is the first byte in the segment. During connection

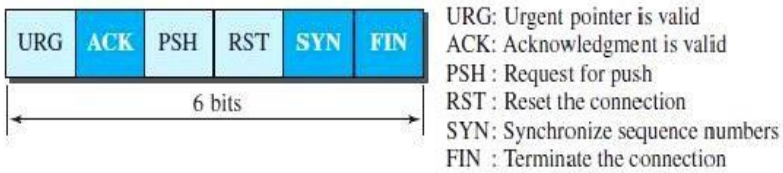
establishment, each party uses a random number generator to create an initial sequence number (ISN), which is usually different in each direction.

< 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 returns 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. Therefore, the value of this field is always between 5 (5 × 4 = 20) and 15 (15 × 4 = 60).

< Control. This field defines 6 different control bits or flags, as shown in Figure 24.8. One or more of these b ts can be set at a time. These bits enable flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP. A brief description of each bit is shown in the figure. 24

Figure 24. *Control field*



< **Window size.** This field defines the window size of the sending TCP in bytes. Note that the length of this field is 16 bits, which means that the maximum size of the window is 65,535 bytes. This value is normally referred to as the receiving window (rwnd) and is determined by the receiver. The sender must obey the dictation of the receiver in this case.

< **Checksum.** This 16-bit field contains the checksum. The calculation of the checksum for TCP follows the same procedure as the one described for UDP. However, the use of the checksum in the UDP datagram is optional, whereas the use of the checksum for TCP is mandatory.

< **Urgent pointer.** This 16-bit field, which is valid only if the urgent flag is set, is used when the segment contains urgent data. It defines a value that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment.

< **Options.** There can be up to 40 bytes of optional information in the TCP header.

Encapsulation

A TCP segment encapsulates the data received from the application layer. The TCP segments is encapsulated in an IP datagram, which in turn is encapsulated in a frame at the data-link layer.

TEXT BOOK: Data Communications and Networking, B Forouzan**,** 5th Ed, McGrawHill Education, 2016 ISBN: 1-25-906475-3