Unit 1

Introduction

Network

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

"Computer network" to mean a collection of autonomous computers interconnected by a single technology. Two computers are said to be interconnected if they are able to exchange information. The connection need not be via a copper wire; fiber optics, microwaves, infrared, and communication satellites can also be used. Networks come in many sizes, shapes and forms. They are usually connected together to make larger networks, with the Internet being the most well-known example of a network of networks. There is considerable confusion in the literature between a computer network and a distributed system. The key distinction is that in a distributed system, a collection of independent computers appears to its users as a single coherent system. Usually, it has a single model or paradigm that it presents to the users. Often a layer of software on top of the operating system, called middleware, is responsible for implementing this model. A well-known example of a distributed system is the World Wide Web. It runs on top of the Internet and presents a model in which everything looks like a document (Web page).

Applications:

1. Financial Services: Electronic Funds Transfer (EFT)
2. Electronic Messaging
3. Video conferencing
4. On-line reservation services
5. **Marketing and sales:** – Computer networks are widely used in both marketing sales firms. These are used by marketing professionals to collect, exchange, and analyzes data relating to customer requirements and product development cycles.

Characteristics:

1. Resource Sharing
2. Communication speed
3. Backup
4. Scalability
5. Reliability
6. S/W & H/W sharing
7. Remote Access
8. Security

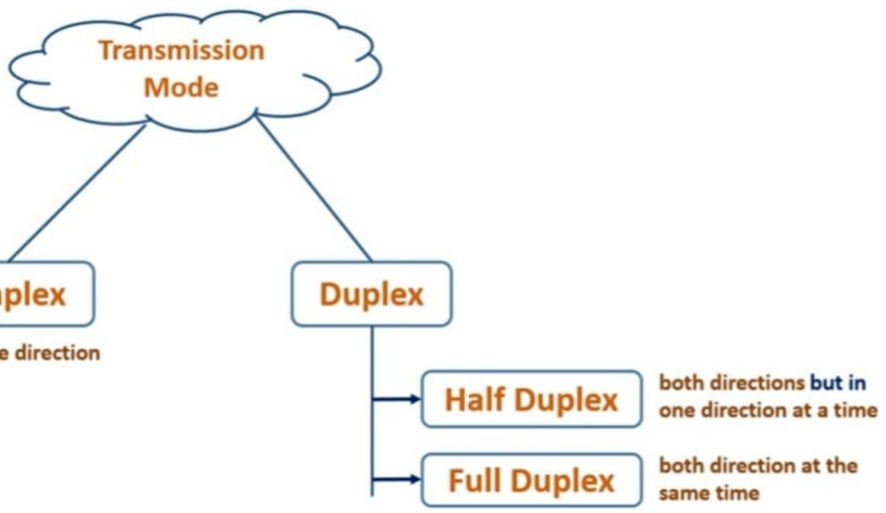
Types of network devices

1. Hub.
2. Switch.
3. Router.
4. Bridge.
5. Gateway.
6. Modem.
7. Repeater.
8. Access Point

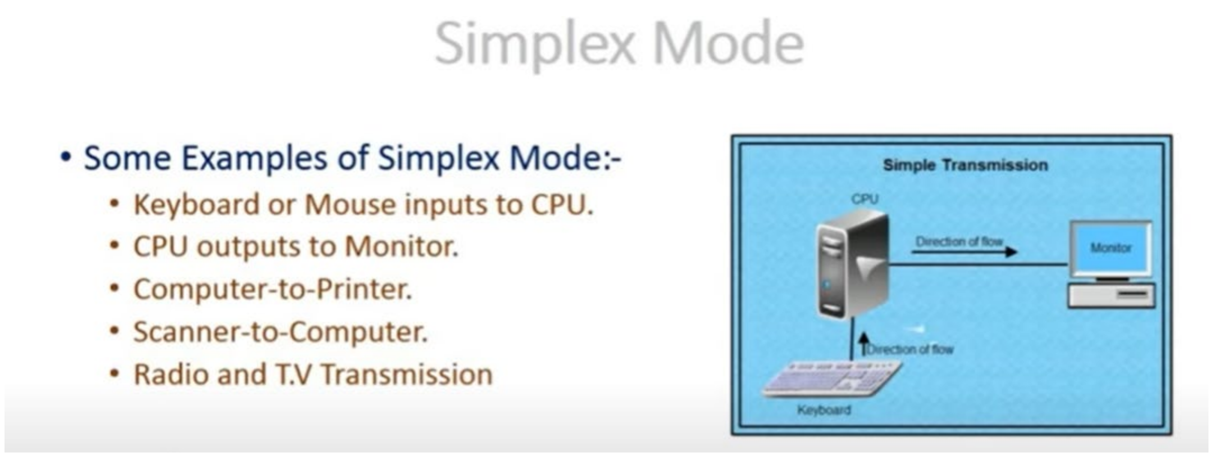
Data Transmission modes :

Simplex, hall duplex, full duplex

Data Transmission Modes | Simplex, Half Duplex & Full Duplex - YouTube

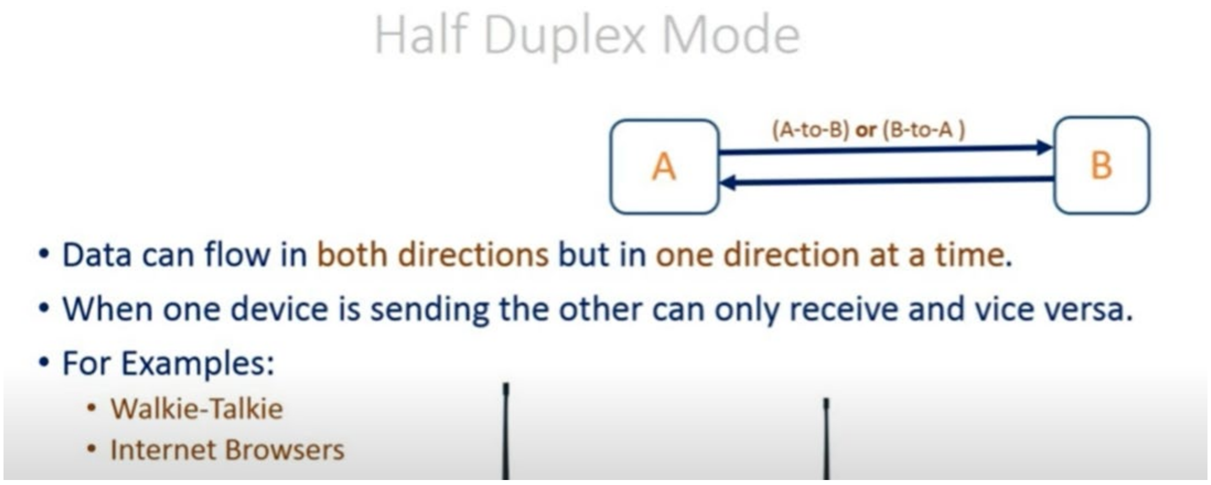

Transmission Mode
Simplex
only one direction
Duplex
Half Duplex
both directions but in one direction at a time
Full Duplex
both direction at the same time

-



Simplex Mode

- Some Examples of Simplex Mode:-
  - Keyboard or Mouse inputs to CPU.
  - CPU outputs to Monitor.
  - Computer-to-Printer.
  - Scanner-to-Computer.
  - Radio and T.V Transmission



Half Duplex Mode

- Data can flow in both directions but in one direction at a time.
- When one device is sending the other can only receive and vice versa.
- For Examples:
  - Walkie-Talkie
  - Internet Browsers

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

Half-Duplex

In half-duplex mode, each station can both transmit and receive, but not at the same time. When one device is sending, the other can only receive, and vice versa (Figure b). Walkie-talkies and CB (citizens band) radios are both half-duplex systems.

Full-Duplex

In full-duplex, both stations can transmit and receive simultaneously (Figure c). One common example of full-duplex communication is the telephone network. When two people are communicating by a telephone line, both can talk and listen at the same time. The full-duplex mode is used when communication in both directions is required all the time.

Network Types: Network Types: LAN, WAN, PAN, CAN, MAN, SAN, WLAN - YouTube

- **LAN:** Local area networks are mainly used to connect personal devices within a few kilometers of a limited area. These networks are used in offices, companies, and factories to exchange data and Information.
- **MAN:** Metropolitan area networks are used to connect the devices over an entire city under the range of up to 50 km. These networks are used in the telephone company network and cable TV network.

- **WAN:** Wide Area Networks are used in the wide geological range over a country and continent. These networks are used in military services, mobile operators, railways, and airlines reservations.
- **PAN:** Personal area networks appropriate to personal or separate workspace under the range of 10 meters. These networks are mostly used to connect tablets, smartphones, and laptops.
- **CAN:** Campus area networks are used to connect limited geographic areas. CAN interconnect multiple local area networks (LAN) within Colleges, Universities, Corporates buildings, etc.

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.

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.

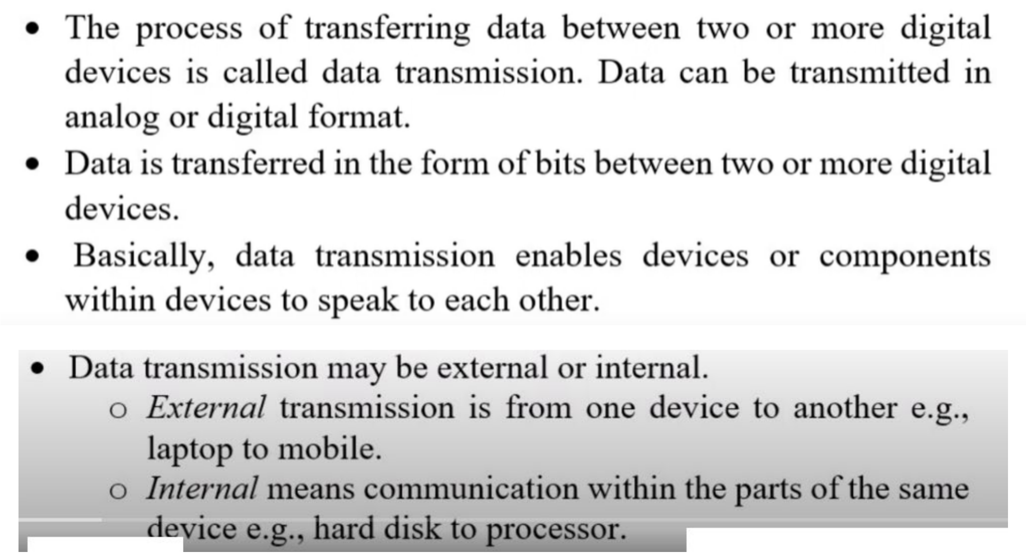
1. Delivery. The system must deliver data to the correct destination. Data must be received by the intended device or user and only by that device or user.
2. Accuracy. The system must deliver the data accurately. Data that have been altered in transmission and left uncorrected are unusable.
3. Timeliness.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

[Network Types: LAN, WAN, PAN, CAN, MAN, SAN, WLAN - YouTube]

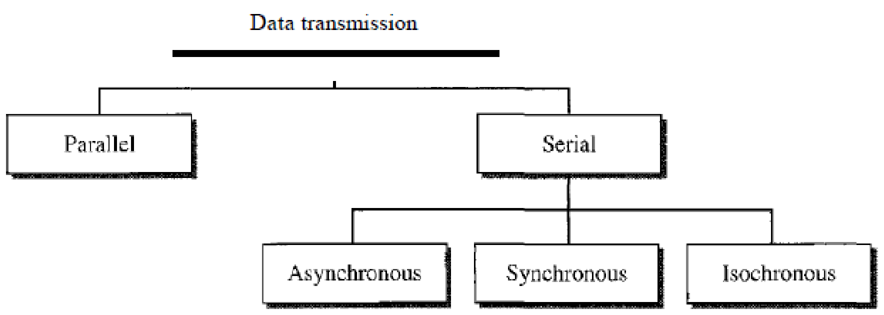
1. What types of technologies are used in PAN?
2. What types of files are transferred in PAN?
3. Most common type of LAN?
4. What does a WLAN typically have?
5. Example for CAN
6. MANs are connected using ___________.
7. _______ is used for data storage. They are not affected by _________.

Transmission Modes

The transmission of binary data across a link can be accomplished in either parallel or serial mode. In parallel mode, multiple bits are sent with each clock tick. In serial mode, 1 bit is sent with each clock tick. While there is only one way to send parallel data, there are three subclasses of serial transmission: asynchronous, synchronous, and isochronous.

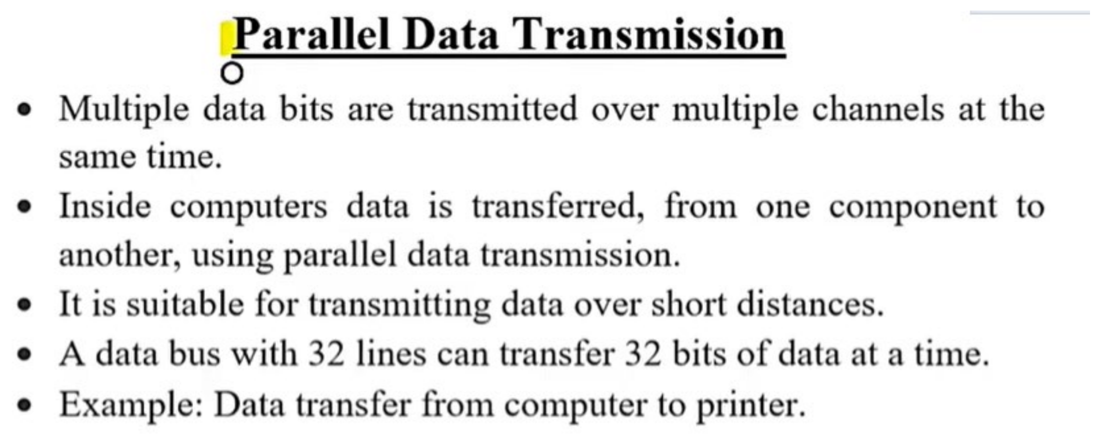


- The process of transferring data between two or more digital devices is called data transmission. Data can be transmitted in analog or digital format.
- Data is transferred in the form of bits between two or more digital devices.
- Basically, data transmission enables devices or components within devices to speak to each other.
- Data transmission may be external or internal.
  - *External* transmission is from one device to another e.g., laptop to mobile.
  - *Internal* means communication within the parts of the same device e.g., hard disk to processor.



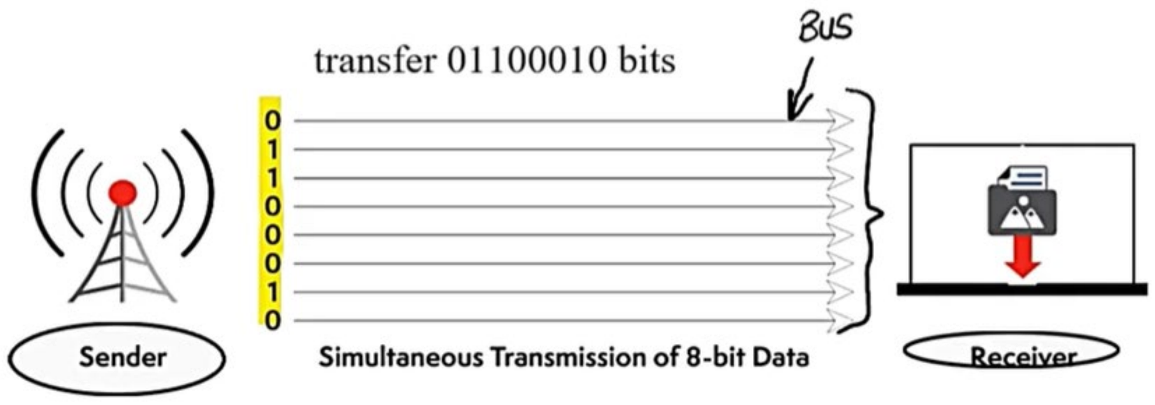
Parallel transmission :

Sending data *n* bits at a time instead of 1 is called parallel transmission. The mechanism for parallel transmission is a conceptually simple one: Use *n* wires to send *n* bits at one time. That way each bit has its own wire, and all *n* bits of one group can be transmitted with each clock tick from one device to another. The advantage of parallel transmission is speed. But there is a significant disadvantage: cost. Parallel transmission requires *n* communication lines (wires in the example) just to transmit the data stream. Because this is expensive, parallel transmission is usually limited to short distances.



Parallel Data Transmission

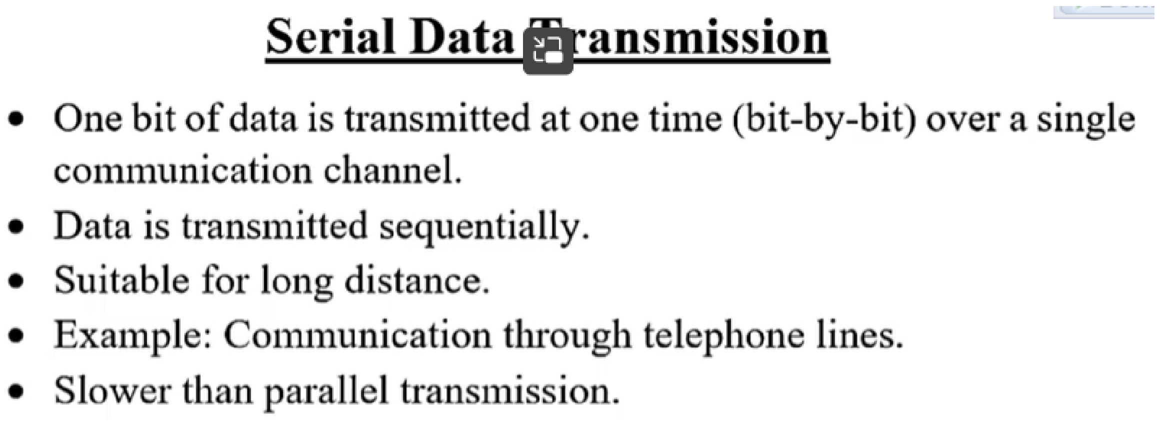
- Multiple data bits are transmitted over multiple channels at the same time.
- Inside computers data is transferred, from one component to another, using parallel data transmission.
- It is suitable for transmitting data over short distances.
- A data bus with 32 lines can transfer 32 bits of data at a time.
- Example: Data transfer from computer to printer.



Parallel and Serial Transmission| Synchronous and Asynchronous Data Transmission - YouTube

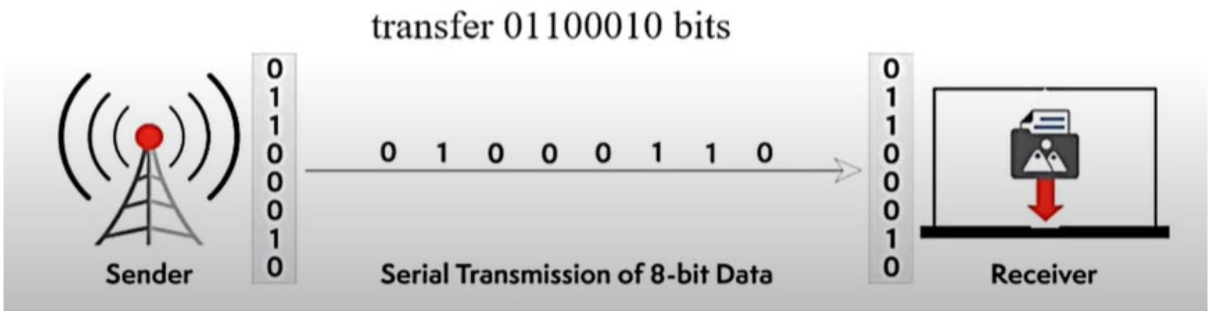
Serial Transmission

In serial transmission one bit follows another, so we need only one communication channel rather than n to transmit data between two communicating devices.



Serial Data Transmission

- One bit of data is transmitted at one time (bit-by-bit) over a single communication channel.
- Data is transmitted sequentially.
- Suitable for long distance.
- Example: Communication through telephone lines.
- Slower than parallel transmission.



The advantage of serial over parallel transmission is that with only one communication channel, serial transmission reduces the cost of transmission over parallel by roughly a factor of n. Since communication within devices is parallel, conversion devices are required at the interface between the sender and the line (parallel-to-serial) and between the line and the receiver (serial-to-parallel). Serial transmission occurs in one of three ways: asynchronous, synchronous, and isochronous.

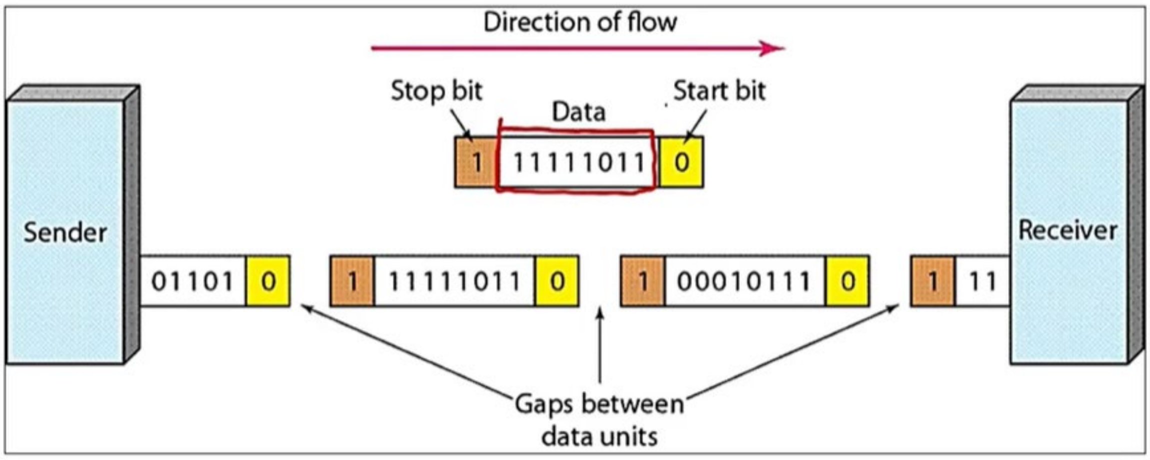
Asynchronous Transmission

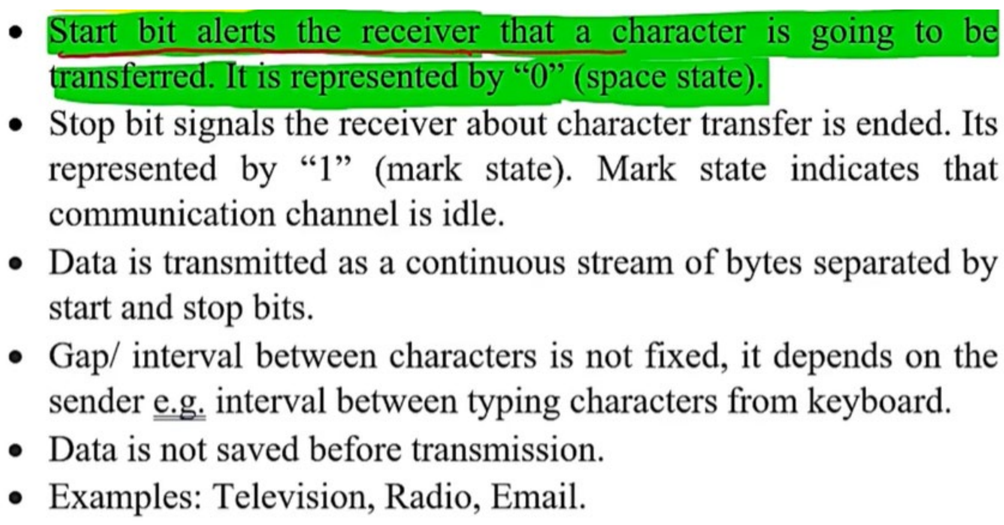
Asynchronous transmission is so named because the timing of a signal is unimportant. Instead, information is received and translated by agreed upon patterns. As long as those patterns are followed, the receiving device can retrieve the information without regard to the rhythm in which it is sent. Patterns are based on grouping the bit stream into bytes. Each group, usually 8 bits, is sent along the link as a unit. The sending system handles each group independently, relaying it to the link whenever ready, without regard to a timer. Without synchronization, the receiver cannot use timing to predict when the next group will arrive. To alert the receiver to the arrival of a new group, therefore, an extra bit is added to the beginning of each byte. This bit, usually a 0, is called the start bit. To let the receiver know that the byte is finished, 1 or more additional bits are appended to the end of the byte. These bits, usually 1 s, are called stop bits. By this method, each byte is increased in size to at least 10 bits, of which 8 bits is information and 2 bits or more are signals to the receiver. In addition, the transmission of each byte may then be followed by a gap of varying duration. This gap can be represented either by an idle channel or by a stream of additional stop bits.

The start and stop bits and the gap alert the receiver to the beginning and end of each byte and allow it to synchronize with the data stream. This mechanism is called asynchronous because, at the byte level, the sender and receiver do not have to be synchronized. But within each byte, the receiver must still be synchronized with the incoming bit stream. That is, some synchronization is required, but only for the duration of a single byte. The receiving device resynchronizes at the onset of each new byte.

When the receiver detects a start bit, it sets a timer and begins counting bits as they come in. After n bits, the receiver looks for a stop bit. As soon as it detects the stop bit, it waits until it detects the next start bit.

In asynchronous transmission, we send 1 start bit (0) at the beginning and 1 or more stop bits (1s) at the end of each byte. There may be a gap between each byte. Asynchronous here means "asynchronous at the byte level; but the bits are still synchronized; their durations are the same.




- Start bit alerts the receiver that a character is going to be transferred. It is represented by "0" (space state).
- Stop bit signals the receiver about character transfer is ended. Its represented by "1" (mark state). Mark state indicates that communication channel is idle.
- Data is transmitted as a continuous stream of bytes separated by start and stop bits.
- Gap/ interval between characters is not fixed, it depends on the sender e.g. interval between typing characters from keyboard.
- Data is not saved before transmission.
- Examples: Television, Radio, Email.

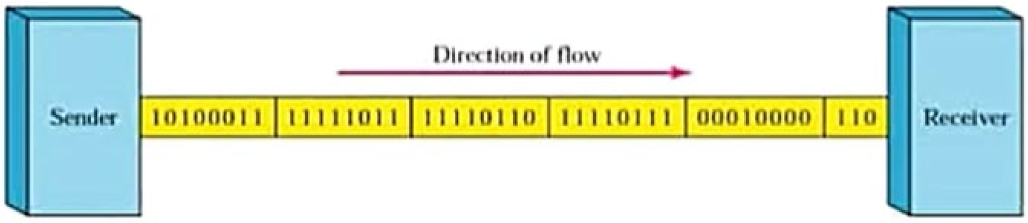
The addition of stop and start bits and the insertion of gaps into the bit stream make asynchronous transmission slower than forms of transmission that can operate without the addition of control information. But it is cheap and effective, two advantages that make it an attractive choice for situations such as low-speed communication.

Synchronous Transmission

In synchronous transmission, the bit stream is combined into longer "frames," which may contain multiple bytes. Each byte, however, is introduced onto the transmission link without a gap between it and the next one. It is left to the receiver to separate the bit stream into bytes for decoding purposes. In other words, data are transmitted as an unbroken string of 1s and 0s, and the receiver separates that string into the bytes, or characters, it needs to reconstruct the information. In synchronous transmission, we send bits one after another without start or stop bits or gaps. It is the responsibility of the receiver to group the bits.

- Data is transmitted block-by-block or word-by-word.
- Each block may contain several bytes of data.
- Data is saved before transmission.
- Synchronized clock (device) is used.
- Clock signals are used to control the transmission of data.
- Large volume of data can be transmitted at a time.
- No gap between data blocks.
- Examples: Video conferencing, Chatting, Telephone.

If the sender wishes to send data in separate bursts, the gaps between bursts must be filled with a special sequence of 0s and 1s that means *idle*. The receiver counts the bits as they arrive and groups them in 8-bit units.

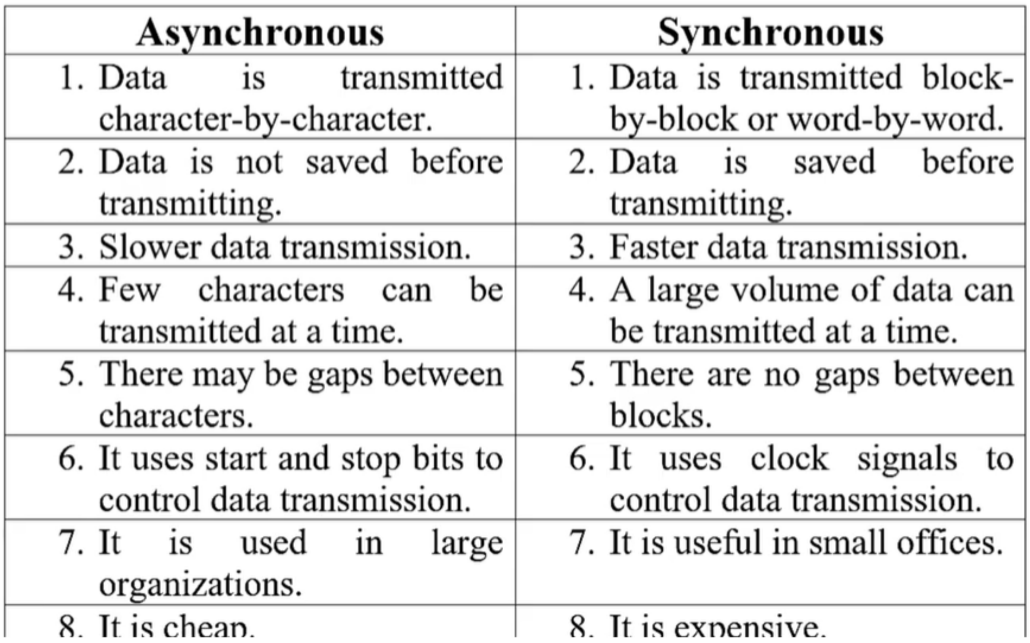


Without gaps and start and stop bits, there is no built-in mechanism to help the receiving device adjust its bit synchronization midstream. Timing becomes very important, therefore, because the accuracy of the received information is completely dependent on the ability of the receiving device to keep an accurate count of the bits as they come in. The advantage of synchronous transmission is speed. With no extra bits or gaps to introduce at the sending end and remove at the receiving end, and, by extension, with fewer bits to move across the link, synchronous transmission is faster than asynchronous transmission. For this reason, it is more useful for high-speed applications such as the transmission of data from one computer to another. Byte synchronization is accomplished in the data link layer.

Isochronous

In real-time audio and video, in which uneven delays between frames are not acceptable, synchronous transmission fails. For example, TV images are broadcast at the rate of 30 images per second; they must be viewed at the same rate. If each image is sent by using one or more frames, there should be no delays between frames. For this type of application, synchronization between characters is not enough; the entire stream of bits must be synchronized. The isochronous transmission guarantees that the data arrive at a fixed rate.

Eg: Multimedia streams require an isochronous transport mechanism to ensure that data is delivered as fast as it is displayed and to ensure that the audio is synchronized with the video.



Asynchronous	Synchronous
1. Data is transmitted character-by-character.	1. Data is transmitted block-by-block or word-by-word.
2. Data is not saved before transmitting.	2. Data is saved before transmitting.
3. Slower data transmission.	3. Faster data transmission.
4. Few characters can be transmitted at a time.	4. A large volume of data can be transmitted at a time.
5. There may be gaps between characters.	5. There are no gaps between blocks.
6. It uses start and stop bits to control data transmission.	6. It uses clock signals to control data transmission.
7. It is used in large organizations.	7. It is useful in small offices.
8. It is cheap.	8. It is expensive.

Network Models: TCP / IP protocol suite

The TCP/IP suite is a set of protocols used on computer networks today (most notably on the Internet). It provides an end-to-end connectivity by specifying how data should be packetized, addressed, transmitted, routed and received on a TCP/IP network. This functionality is organized into four abstraction layers and each protocol in the suite resides in a particular layer.

The main work of TCP/IP is to transfer the data of a computer from one device to another. The main condition of this process is to make data reliable and accurate so that the receiver will receive the same information which is sent by the sender. To ensure that, each message reaches its final destination accurately, the TCP/IP model divides its data into packets and combines them at the other end, which helps in maintaining the accuracy of the data while transferring from one end to another end.

TCP/IP uses the client-server model of communication in which a user or machine (a client) is provided a service, like sending a webpage, by another computer (a server) in the network. Collectively, the TCP/IP suite of protocols is classified as stateless, which means each client request is considered new because it is unrelated to previous requests. Being stateless frees up network paths so they can be used continuously.

Why is TCP/IP important?

TCP/IP is non-proprietary and, as a result, is not controlled by any single company. Therefore, the IP suite can be modified easily. It is compatible with all operating systems (OSes), so it can communicate with any other system. The IP suite is also compatible with all types of computer hardware and networks. TCP/IP is highly scalable and, as a routable protocol, can determine the most efficient path through the network. It is widely used in current internet architecture.

Uses of TCP/IP

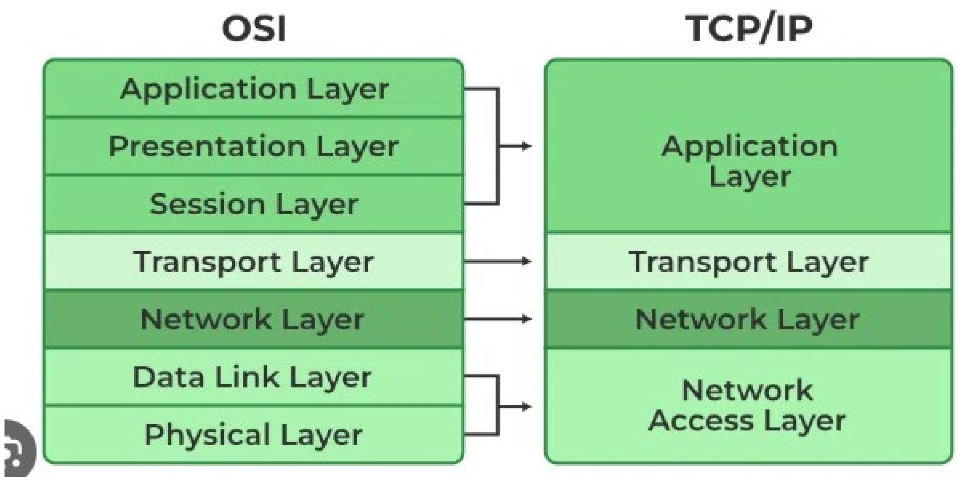
TCP/IP can be used to provide remote login over the network for interactive file transfer to deliver email, to deliver webpages over the network and to remotely access a server host's file system. Most broadly, it is used to represent how information changes form as it travels over a network from the concrete physical layer to the abstract application layer. It details the basic protocols, or methods of communication, at each layer as information passes through.

The advantages of using the TCP/IP model include the following:

- helps establish a connection between different types of computers;
- works independently of the OS; supports many routing protocols;
- uses client-server architecture that is highly scalable;
- is lightweight and doesn't place unnecessary strain on a network or computer.

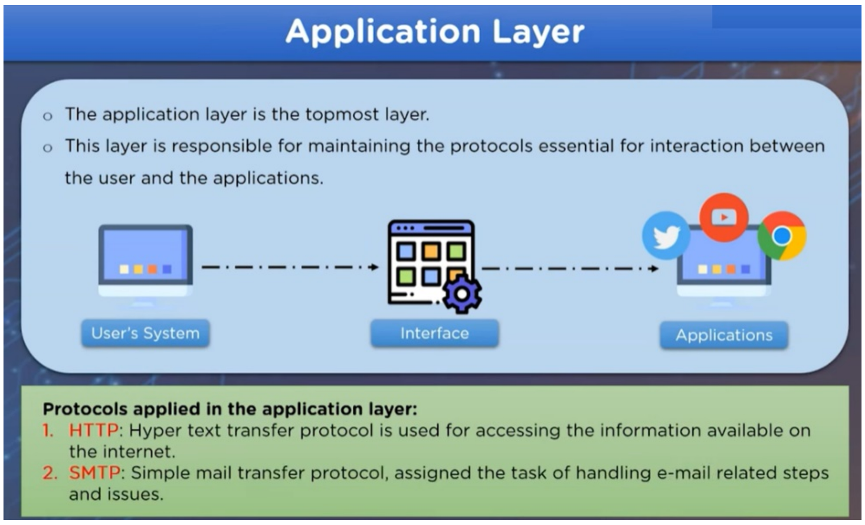
The disadvantages of TCP/IP include the following:

- is complicated to set up and manage;
- transport layer does not guarantee delivery of packets;
- is not easy to replace protocols in TCP/IP;
- does not clearly separate the concepts of services, interfaces and protocols, so it is not suitable for describing new technologies in new networks; and
- is especially vulnerable to a synchronization attack, which is a type of denial-of-service attack in which a bad actor uses TCP/IP.



Layers of TCP/IP Model

1. Application Layer
2. Transport Layer(TCP/UDP)
3. Network/Internet Layer(IP)
4. Data Link Layer (MAC)
5. Physical Layer



Application Layer

- The application layer is the topmost layer.
- This layer is responsible for maintaining the protocols essential for interaction between the user and the applications.

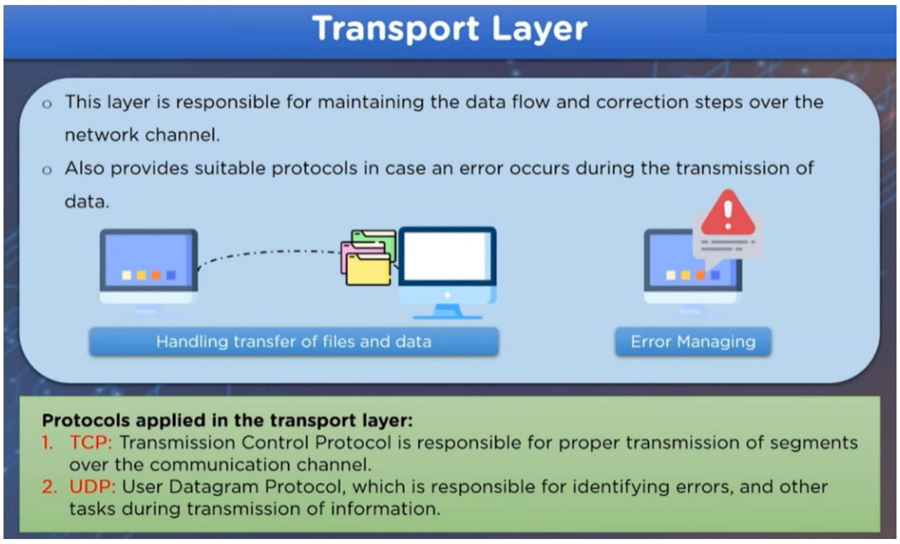
Protocols applied in the application layer:

1. HTTP: Hyper text transfer protocol is used for accessing the information available on the internet.
2. SMTP: Simple mail transfer protocol, assigned the task of handling e-mail related steps and issues.

Following are the main protocols used in the application layer:

- **HTTP:** HTTP stands for Hypertext transfer protocol. This protocol allows us to access the data over the world wide web. It transfers the data in the form of plain text, audio, video. It is known as a Hypertext transfer protocol as it has the efficiency to use in a hypertext environment where there are rapid jumps from one document to another.
- **SNMP:** SNMP stands for Simple Network Management Protocol. It is a framework used for managing the devices on the internet by using the TCP/IP protocol suite.

- **SMTP:** SMTP stands for Simple mail transfer protocol. The TCP/IP protocol that supports the e-mail is known as a Simple mail transfer protocol. This protocol is used to send the data to another e-mail address.
- **DNS:** DNS stands for Domain Name System. An IP address is used to identify the connection of a host to the internet uniquely. But, people prefer to use the names instead of addresses. Therefore, the system that maps the name to the address is known as Domain Name System.
- **TELNET:** It is an abbreviation for Terminal Network. It establishes the connection between the local computer and remote computer in such a way that the local terminal appears to be a terminal at the remote system.
- **FTP:** FTP stands for File Transfer Protocol. FTP is a standard internet protocol used for transmitting the files from one computer to another computer.



Transport Layer

- This layer is responsible for maintaining the data flow and correction steps over the network channel.
- Also provides suitable protocols in case an error occurs during the transmission of data.

Handling transfer of files and data
Error Managing

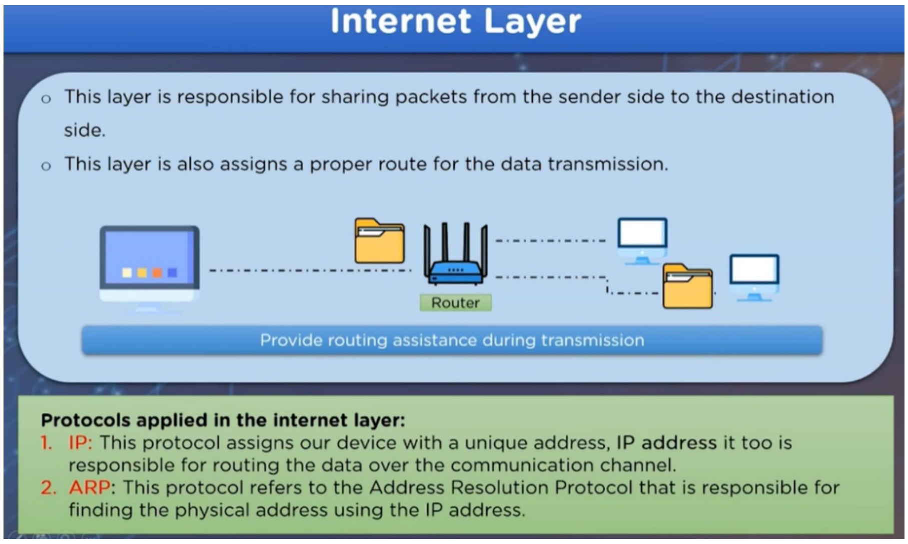
Protocols applied in the transport layer:
1. TCP: Transmission Control Protocol is responsible for proper transmission of segments over the communication channel.
2. UDP: User Datagram Protocol, which is responsible for identifying errors, and other tasks during transmission of information.

The transport layer is responsible for the reliability, flow control, and correction of data which is being sent over the network. The two protocols used in the transport layer are **User Datagram protocol and Transmission control protocol**.

	TCP	UDP
Structure	Segments	Datagrams
Connection Model	Connected, one-to-one	Connectionless, one-to-many
Data Transfer	Ordered	Unordered
Reliability	Reliable	Unreliable
Overhead	Resource intensive	Lightweight

Difference between TCP and UDP

	TCP	UDP
Connection	Connection-oriented protocol	Connection-less protocol
Speed	The speed for TCP is slower	The speed for UDP is higher
Error Detection	Yes	No
Data Packets	Data packets are arranged in the correct order	Data packets are independent of each other, and therefore does not follow a sequence.
Acknowledgement	Acknowledgement Segments	No Acknowledgement Segments
Handshake Protocol	Uses protocols such as: - SYN, ACK, SYN-ACK	No Handshake, and therefore connectionless protocol.
Reliability	Successful deliverance of data to destination router, and therefore reliable.	Deliverance of data to the destination router is not reliable.



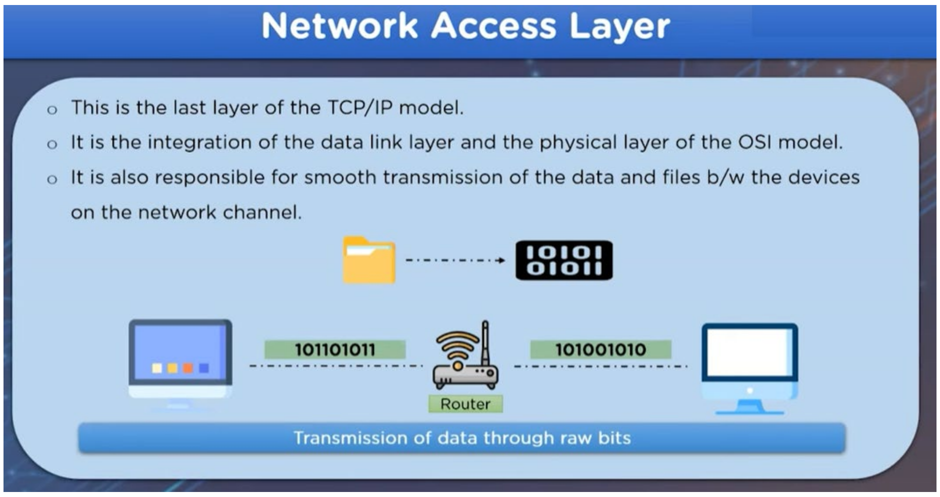
Internet Layer

- This layer is responsible for sharing packets from the sender side to the destination side.
- This layer is also assigns a proper route for the data transmission.

Router
Provide routing assistance during transmission

Protocols applied in the internet layer:
1. IP: This protocol assigns our device with a unique address, IP address it too is responsible for routing the data over the communication channel.
2. ARP: This protocol refers to the Address Resolution Protocol that is responsible for finding the physical address using the IP address.

- **ICMP** stands for Internet Control Message Protocol.It is a mechanism used by the hosts or routers to send notifications regarding datagram problems back to the sender.



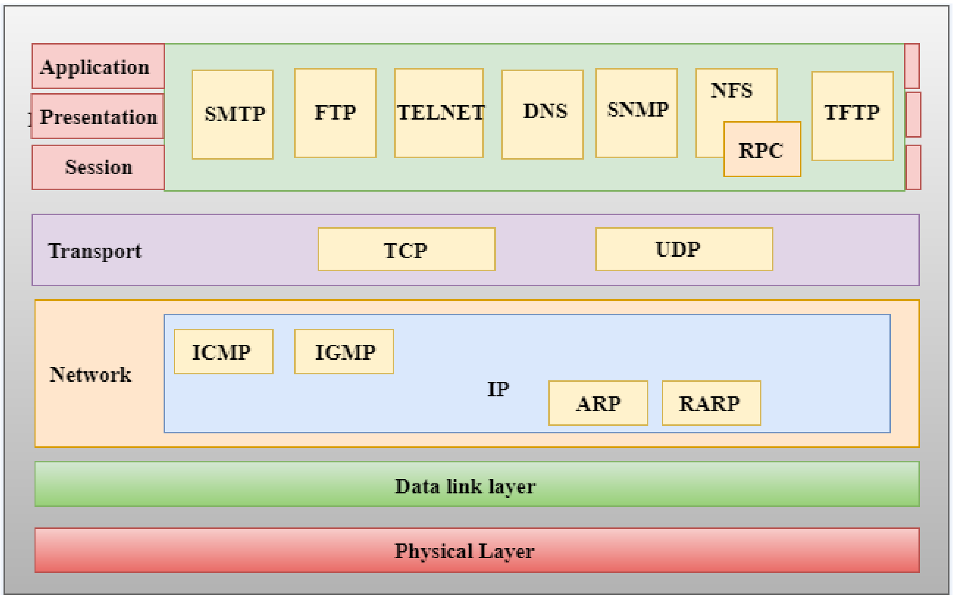
Network Access Layer

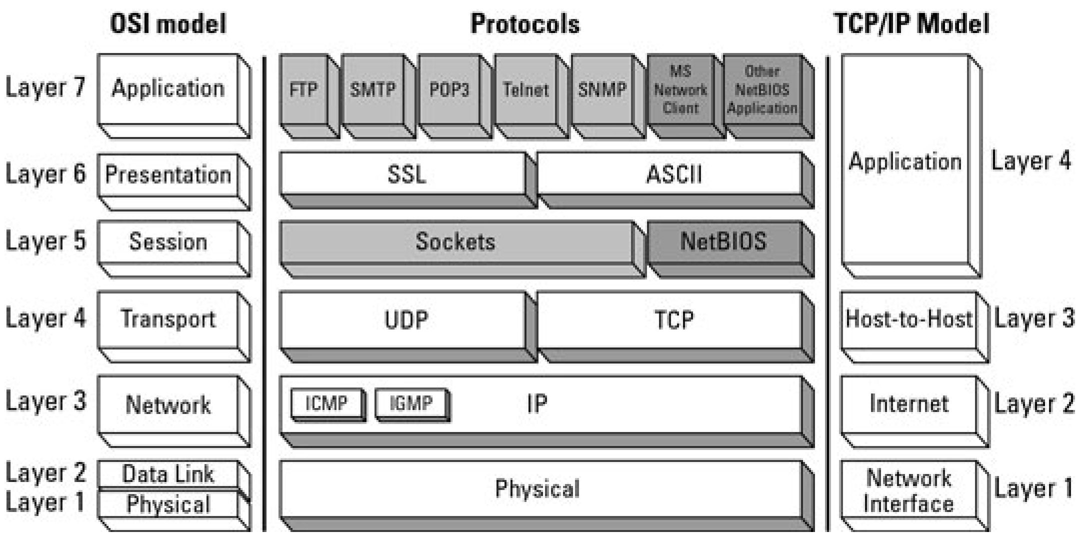
- This is the last layer of the TCP/IP model.
- It is the integration of the data link layer and the physical layer of the OSI model.
- It is also responsible for smooth transmission of the data and files b/w the devices on the network channel.

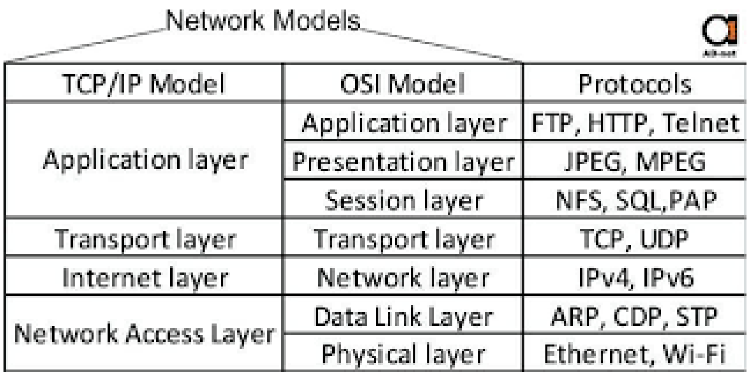
10101
01011
101101011
101001010
Router
Transmission of data through raw bits

- A network layer is the lowest layer of the TCP/IP model.

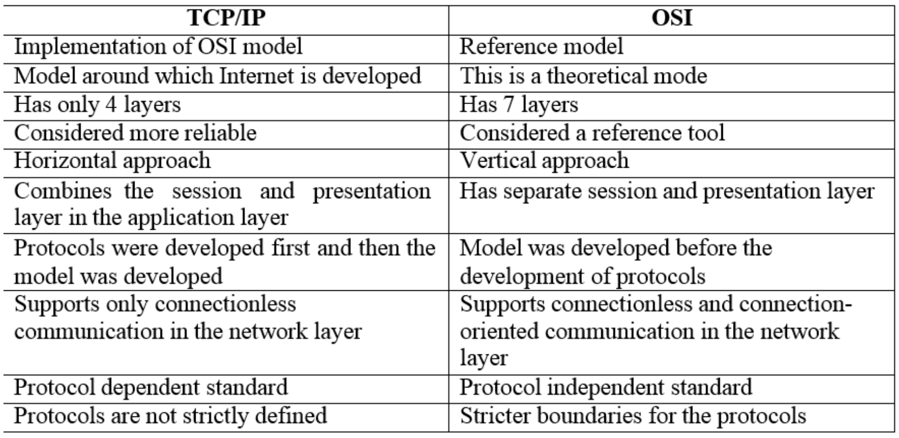
- A network layer is the combination of the Physical layer and Data Link layer defined in the OSI reference model.
- It defines how the data should be sent physically through the network.
- This layer is mainly responsible for the transmission of the data between two devices on the same network.
- The functions carried out by this layer are encapsulating the IP datagram into frames transmitted by the network and mapping of IP addresses into physical addresses.



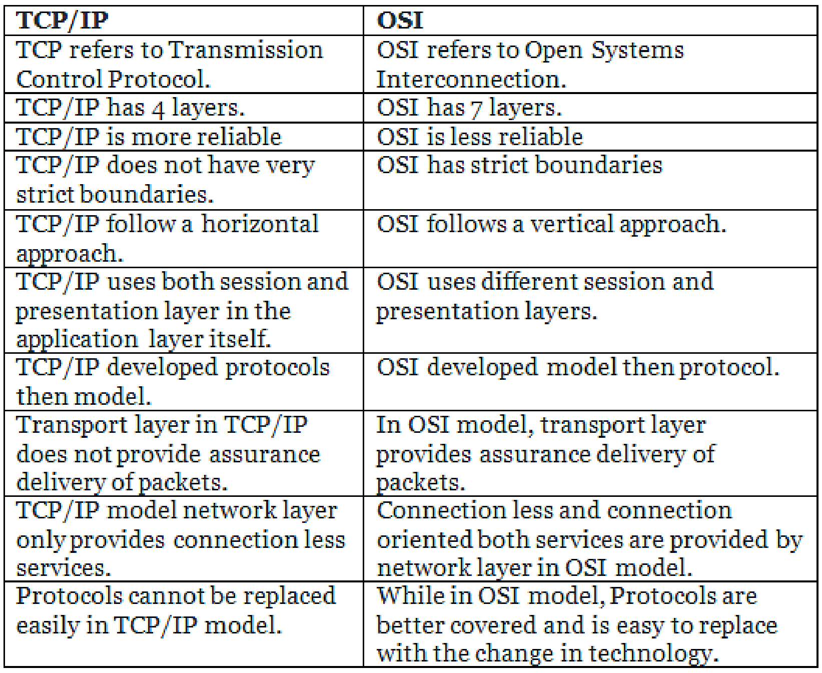



Network Models

TCP/IP Model	OSI Model	Protocols
Application layer	Application layer	FTP, HTTP, Telnet
	Presentation layer	JPEG, MPEG
	Session layer	NFS, SQL,PAP
Transport layer	Transport layer	TCP, UDP
Internet layer	Network layer	IPv4, IPv6
Network Access Layer	Data Link Layer	ARP, CDP, STP
	Physical layer	Ethernet, Wi-Fi



TCP/IP	OSI
Implementation of OSI model	Reference model
Model around which Internet is developed	This is a theoretical mode
Has only 4 layers	Has 7 layers
Considered more reliable	Considered a reference tool
Horizontal approach	Vertical approach
Combines the session and presentation layer in the application layer	Has separate session and presentation layer
Protocols were developed first and then the model was developed	Model was developed before the development of protocols
Supports only connectionless communication in the network layer	Supports connectionless and connection-oriented communication in the network layer
Protocol dependent standard	Protocol independent standard
Protocols are not strictly defined	Stricter boundaries for the protocols



TCP/IP	OSI
TCP refers to Transmission Control Protocol.	OSI refers to Open Systems Interconnection.
TCP/IP has 4 layers.	OSI has 7 layers.
TCP/IP is more reliable	OSI is less reliable
TCP/IP does not have very strict boundaries.	OSI has strict boundaries
TCP/IP follow a horizontal approach.	OSI follows a vertical approach.
TCP/IP uses both session and presentation layer in the application layer itself.	OSI uses different session and presentation layers.
TCP/IP developed protocols then model.	OSI developed model then protocol.
Transport layer in TCP/IP does not provide assurance delivery of packets.	In OSI model, transport layer provides assurance delivery of packets.
TCP/IP model network layer only provides connection less services.	Connection less and connection oriented both services are provided by network layer in OSI model.
Protocols cannot be replaced easily in TCP/IP model.	While in OSI model, Protocols are better covered and is easy to replace with the change in technology.

OSI Model

The OSI Model (Open Systems Interconnection Model) is a conceptual framework used to describe the functions of a networking system. OSI is **a generic, protocol-independent model intended to describe all forms of network communication.** OSI model was developed by the International Organization for Standardization (ISO) in 1984, and it is now considered as an architectural model for the inter-computer communications.

OSI Model (Part 1) - Application, Presentation, and Session Layer | TechTerms - YouTube

OSI Model (Part 2) - Transport layer and Network Layer | TechTerms - YouTube

OSI Model (Part 3) - Data Link Layer, and Physical Layer| TechTerms - YouTube

The 7 Layers of the OSI Model

Physical Layer

The lowest layer of the OSI Model is concerned with electrically or optically transmitting raw unstructured data bits across the network from the physical layer of the sending device to the physical layer of the receiving device. It can include specifications such as voltages, pin layout, cabling, and radio frequencies. At the physical layer, one might find "physical" resources such as network hubs, cabling, repeaters, network adapters or modems.

Data Link Layer

At the data link layer, directly connected nodes are used to perform node-to-node data transfer where data is packaged into frames. The data link layer also corrects errors that may have occurred at the physical layer.

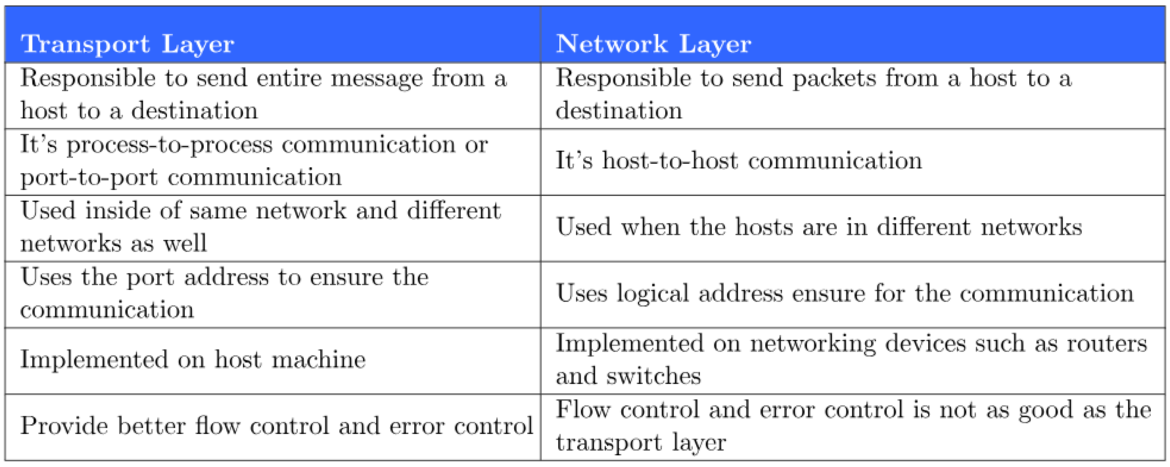
The data link layer encompasses two sub-layers of its own. The first, media access control (MAC), provides flow control and multiplexing for device transmissions over a network. The second, the logical link control (LLC), provides flow and error control over the physical medium as well as identifies line protocols.

Network Layer

The network layer is responsible for receiving frames from the data link layer, and delivering them to their intended destinations among based on the addresses contained inside the frame. The network layer finds the destination by using logical addresses, such as IP (internet protocol). At this layer, routers are a crucial component used to quite literally route information where it needs to go between networks.

Transport Layer

The transport layer manages the delivery and error checking of data packets. It regulates the size, sequencing, and ultimately the transfer of data between systems and hosts. One of the most common examples of the transport layer is TCP or the Transmission Control Protocol.



Transport Layer	Network Layer
Responsible to send entire message from a host to a destination	Responsible to send packets from a host to a destination
It's process-to-process communication or port-to-port communication	It's host-to-host communication
Used inside of same network and different networks as well	Used when the hosts are in different networks
Uses the port address to ensure the communication	Uses logical address ensure for the communication
Implemented on host machine	Implemented on networking devices such as routers and switches
Provide better flow control and error control	Flow control and error control is not as good as the transport layer

Session Layer

The session layer controls the conversations between different computers. A session or connection between machines is set up, managed, and terminated at layer 5. Session layer services also include authentication and reconnections.

Presentation Layer

The presentation layer formats or translates data for the application layer based on the syntax or semantics that the application accepts. Because of this, it at times also called the syntax layer. This layer can also handle the encryption and decryption required by the application layer.

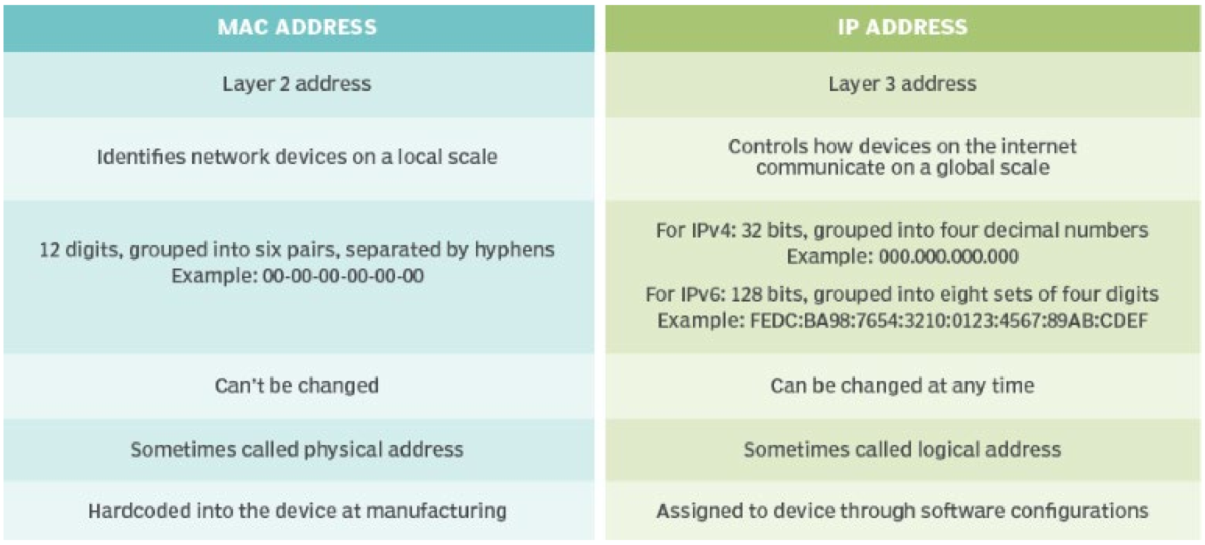
Application Layer

At this layer, both the end user and the application layer interact directly with the software application. This layer sees network services provided to end-user applications such as a web browser or Office 365. The application layer identifies communication partners, resource availability, and synchronizes communication.

Addressing

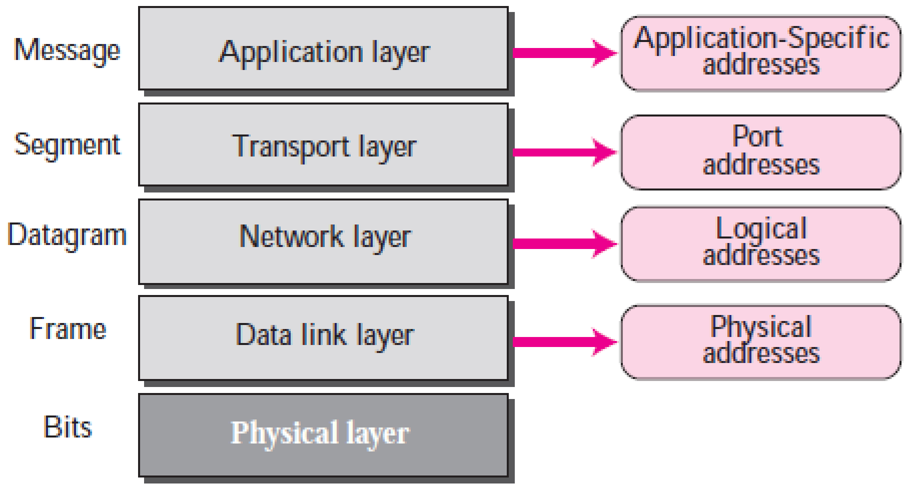
Four levels of addresses are used in an internet employing the *TCP/IP* protocols:
physical (link) addresses, logical (IP) addresses, port addresses, and specific addresses

The physical address -- which is also called a media access control, or MAC, address -- identifies a device to other devices on the same local network. The internet address -- or IP address -- identifies the device globally. A network packet needs both addresses to get to its destination.



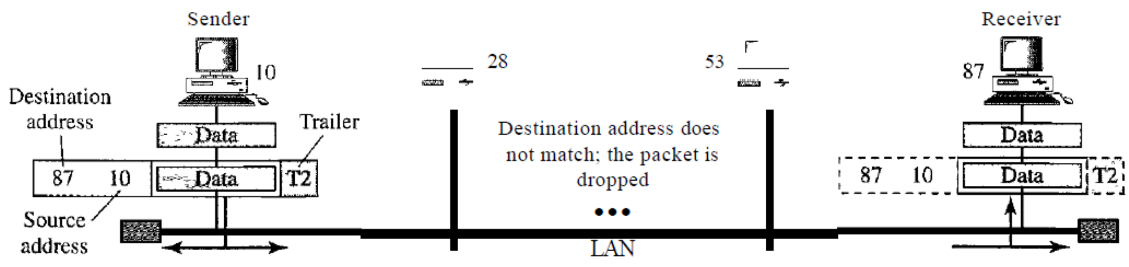
MAC ADDRESS	IP ADDRESS
Layer 2 address	Layer 3 address
Identifies network devices on a local scale	Controls how devices on the internet communicate on a global scale
12 digits, grouped into six pairs, separated by hyphens Example: 00-00-00-00-00-00	For IPv4: 32 bits, grouped into four decimal numbers Example: 000.000.000.000 For IPv6: 128 bits, grouped into eight sets of four digits Example: FEDC:BA98:7654:3210:0123:4567:89AB:CDEF
Can't be changed	Can be changed at any time
Sometimes called physical address	Sometimes called logical address
Hardcoded into the device at manufacturing	Assigned to device through software configurations

The physical address, also known as the link address, is the address of a node as defined by its LAN or WAN. It is included in the frame used by the data link layer. It is the lowest-level address. The physical addresses have authority over the network (LAN or WAN). The size and format of these addresses vary depending on the network. For example, Ethernet uses a 6-byte (48-bit) physical address that is imprinted on the network interface card (NIC).



In the below figure, a node with physical address 10 sends a frame to a node with physical address 87. The two nodes are connected by a link (bus topology LAN). At the data link layer, this frame contains physical (link) addresses in the header. These are the only addresses needed. The rest of the header contains other information needed at this level. The trailer usually contains extra bits needed for error detection. As the figure shows, the computer with physical address lOis the sender, and the computer with physical address 87 is the receiver. The data link layer at the

sender receives data from an upper layer. It encapsulates the data in a frame, adding a header and a trailer. The header, among other pieces of information, carries the receiver and the sender physical (link) addresses. Note that in most data link protocols, the destination address, 87 in this case, comes before the source address (10 in this case).

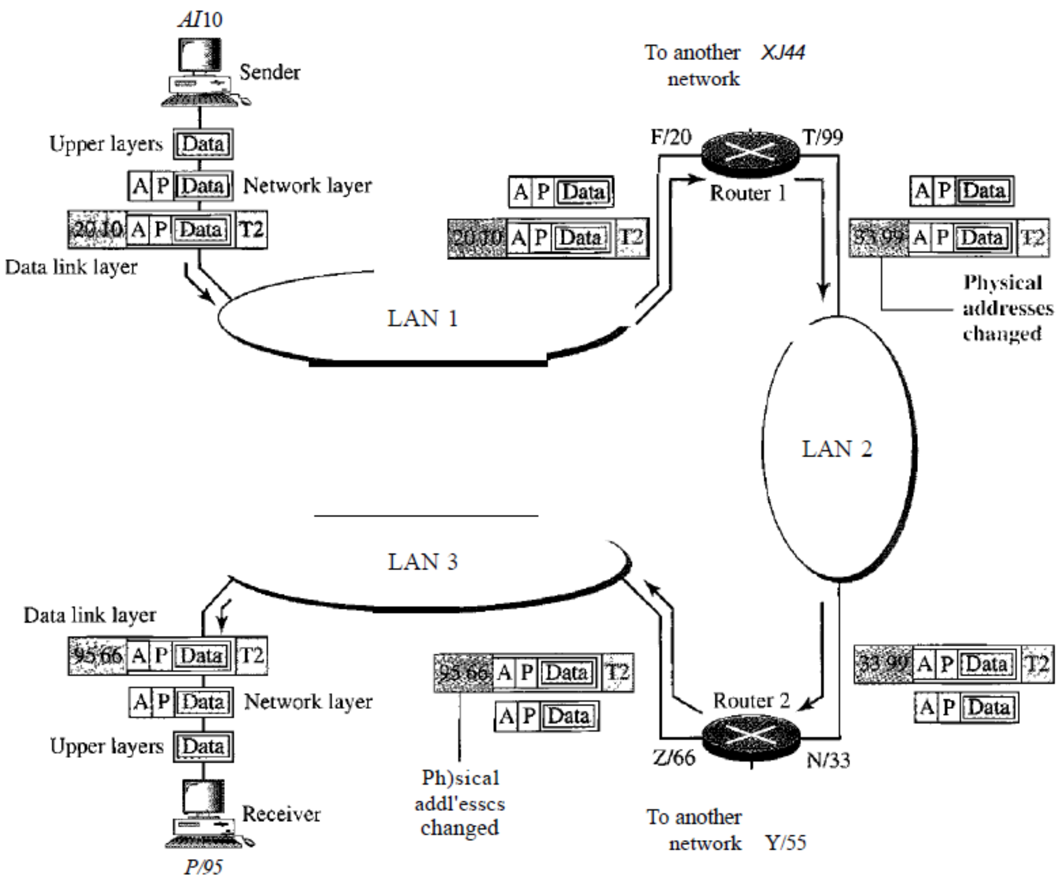


In a bus topology, the frame is propagated in both directions (left and right). The frame propagated to the left dies when it reaches the end of the cable if the cable end is terminated appropriately. The frame propagated to the right is sent to every station on the network. Each station with a physical addresses other than 87 drops the frame because the destination address in the frame does not match its own physical address. The intended destination computer, however, finds a match between the destination address in the frame and its own physical address. The frame is checked, the header and trailer are dropped, and the data part is decapsulated and delivered to the upper layer. most local-area networks use a 48-bit (6-byte) physical address written as 12 hexadecimal digits; 07:01:02:01 :2C:4B
A 6-byte (12 hexadecimal digits) physical address.

Logical Addresses

Logical addresses are necessary for universal communications that are independent of underlying physical networks. Physical addresses are not adequate in an internetwork environment where different networks can have different address formats. A universal addressing system is needed in which each host can be identified uniquely, regardless of the underlying physical network.
The logical addresses are designed for this purpose. A logical address in the Internet is currently a 32-bit address that can uniquely define a host connected to the Internet. No two publicly addressed and visible hosts on the Internet can have the same IP address. The below figure shows a part of an internet with two routers connecting three LANs. Each device (computer or router) has a pair of addresses (logical and physical) for each connection. In this case, each computer is connected to only one link and therefore has only one pair of addresses. Each router, however, is connected to three networks (only two are shown in the figure). So each router has three pairs of addresses, one for each connection.

The computer with logical address A and physical address 10 needs to send a packet to the computer with logical address P and physical address 95. We use letters to show the logical addresses and numbers for physical addresses. The sender encapsulates its data in a packet at the network layer and adds two logical addresses (A and P). The network layer, however, needs to find the physical address of the next hop before the packet can be delivered. The network layer consults its routing table and finds the logical address of the next hop (router I) to be F. The ARP discussed previously finds the physical address of router 1 that corresponds to the logical address of 20. Now the network layer passes this address to the data link layer, which in turn, encapsulates the packet with physical destination address 20 and physical source address 10. The frame is received by every device on LAN 1, but is discarded by all except router 1, which finds that the destination physical address in the frame matches with its own physical address.



The router decapsulates the packet from the frame to read the logical destination address P. Since the logical destination address does not match the router's logical address, the router knows that the packet needs to be forwarded. The router consults its routing table and ARP to find the physical destination address of the next hop (router 2), creates a new frame, encapsulates the packet, and sends it to router 2. Note the physical addresses in the frame. The source physical address changes from 10 to 99. The destination physical address changes from 20 (router 1 physical address) to 33 (router 2 physical address). The logical source and destination addresses must remain the same; otherwise the packet will be lost. At router 2 we have a similar scenario. The physical addresses are changed, and a new frame is sent to the destination computer. When the frame reaches the destination, the packet is decapsulated. The destination logical address P matches the logical address of the computer. The data are decapsulated from the packet and delivered to the upper layer. Note that although physical addresses will change from hop to hop, logical addresses remain the same from the source to destination.

Port Addresses

A port or port number is a number assigned to uniquely identify a connection endpoint and to direct data to a specific service. The IP address and the physical address are necessary for a quantity of data to travel from a source to the destination host. However, arrival at the destination host is not the final objective of data communications on the Internet. A system that sends nothing but data from one computer to another is not complete. Today, computers are devices that can run multiple processes at the same time. The end objective of Internet communication is a process communicating with another process. For example, computer A can communicate with computer C by using TELNET. At the same time, computer A communicates with computer B by using the File Transfer Protocol (FTP). For these processes to receive data simultaneously, we need a method to label the different processes. In other words, they need addresses. In the TCPIIP architecture, the label assigned to a process is called a port address. A port address in TCPIIP is 16 bits in length.

Specific Addresses: Some applications have user-friendly addresses that are designed for that specific address. Examples include the e-mail address and the Universal Resource Locator (URL). The first defines the recipient of an e-mail; the second is used to find a document on the World Wide Web.

Link Layer: Data Link Control (DLC):

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.

1. **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. Our postal system practices a type of framing. The simple act of inserting a letter into an envelope separates one piece of information from another; the envelope serves as the delimiter. In addition, each envelope defines the sender and receiver addresses, which is necessary since the postal system is a manyto- many carrier facility.
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 into smaller frames, a single-bit error affects only that small frame.

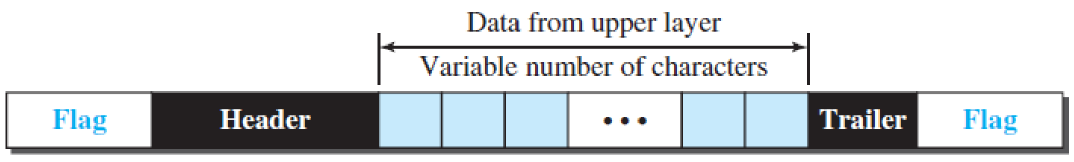
Frame Size

Frames can be of fixed or variable size. In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. An example of this type of framing is the ATM WAN. Variable-size framing, prevalent in local-area networks. In variable-size framing, we need a way to define the end of one frame and the beginning of the next. Historically, two approaches were used for this purpose: a character-oriented approach and a bit-oriented approach.

Character-Oriented Framing

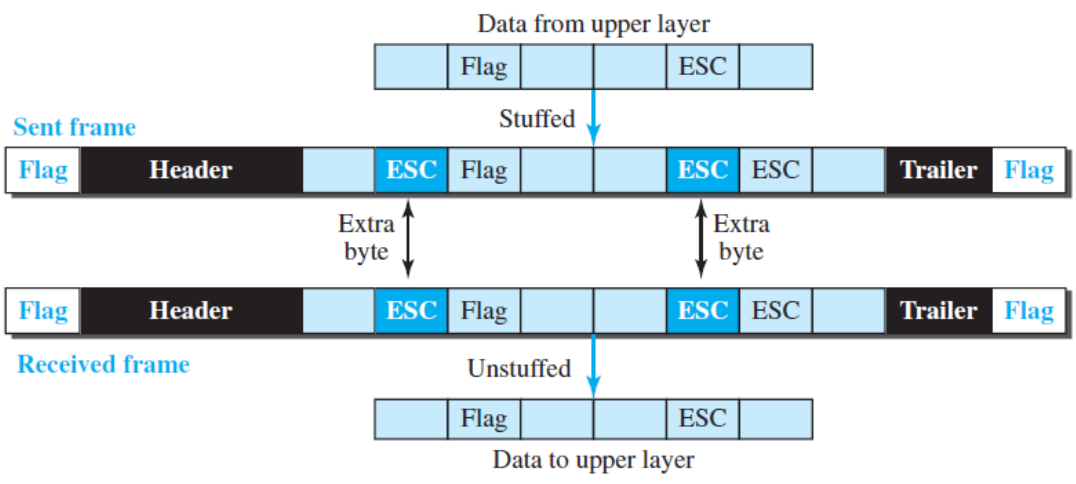
In character-oriented (or byte-oriented) framing, data to be carried are 8-bit characters from a coding system such as ASCII. The header, which normally carries the source and destination addresses and other control information, and the trailer, which carries error detection redundant bits, are also 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 11.1 shows the format of a frame in a character-oriented protocol.

Figure 11.1 *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 the 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 11.2 shows the situation.

Figure 11.2 *Byte stuffing and unstuffing*



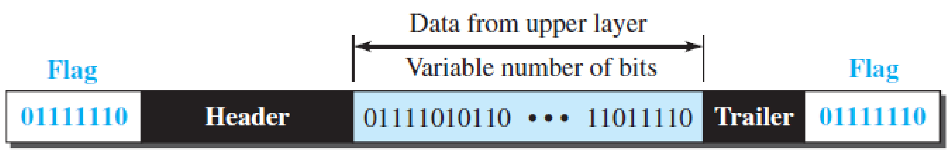
Byte stuffing is the process of adding one extra byte whenever there is a flag or escape character in the text.

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 the second one 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 and 32-bit characters that conflict with 8-bit characters.

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 Figure 11.3.

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

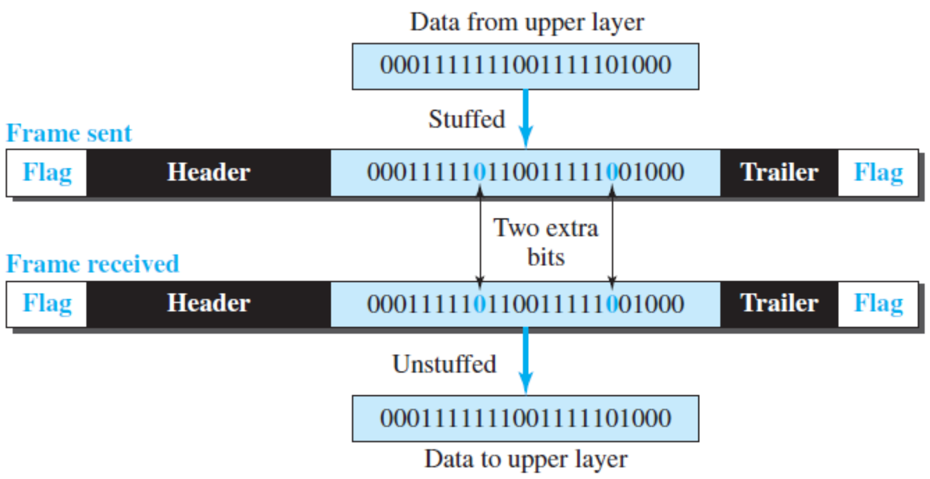


Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag.

This flag can create the same type of problem we saw in the character-oriented protocols. That is, if the flag pattern appears in the data, we need to somehow inform the receiver that this is not the end of the frame. We do this by stuffing 1 single bit (instead of 1 byte) to prevent the pattern from looking like a flag. The strategy is called bit stuffing. In bit stuffing, if a 0 and five consecutive 1 bits are encountered, an extra 0 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 inadvertently appear in the frame.

Figure 11.4 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 flaglike pattern 01111110 appears in the data, it will change to 011111010 (stuffed) and is not mistaken for a flag by the receiver. The real flag 01111110 is not stuffed by the sender and is recognized by the receiver.

Figure 11.4 *Bit stuffing and unstuffing*



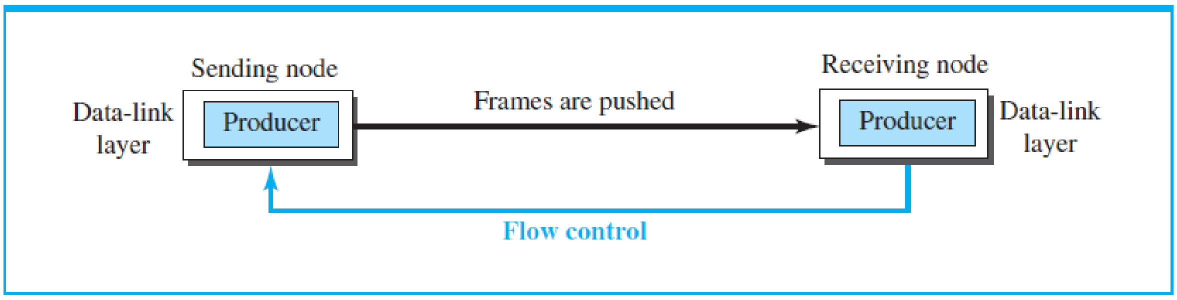
2. Flow and Error Control

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.

Figure 11.5 *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.

Error Control

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. A frame that carries an acknowledgment is normally called an ACK to distinguish it from the data frame.

3. Connectionless and Connection-Oriented

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

Connectionless Protocol

In a 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

In a 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.

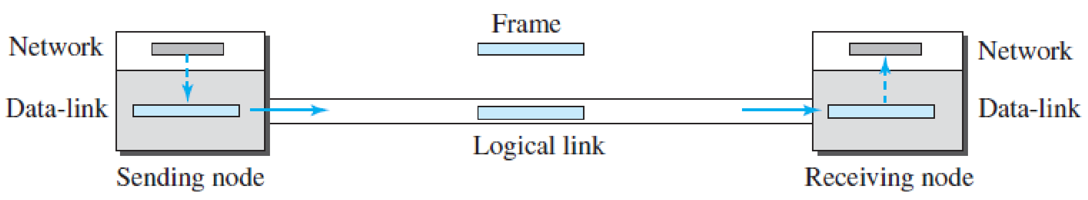
Data Link Layer Protocols

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 machine is always in one of the states until an event occurs. Each event is associate with two reactions: defining the list (possibly empty) of actions to be performed and determining the next state (which can be the same as the current state). One of the states must be defined as the initial state, the state in which the machine starts when it turns on.

1. Simple Protocol

Simple protocol is 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 11.7 shows the layout for this protocol.

Figure 11.7 *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. The 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 11.8 shows the FSMs for the simple protocol.

Figure 11.8 *FSMs for the simple protocol*

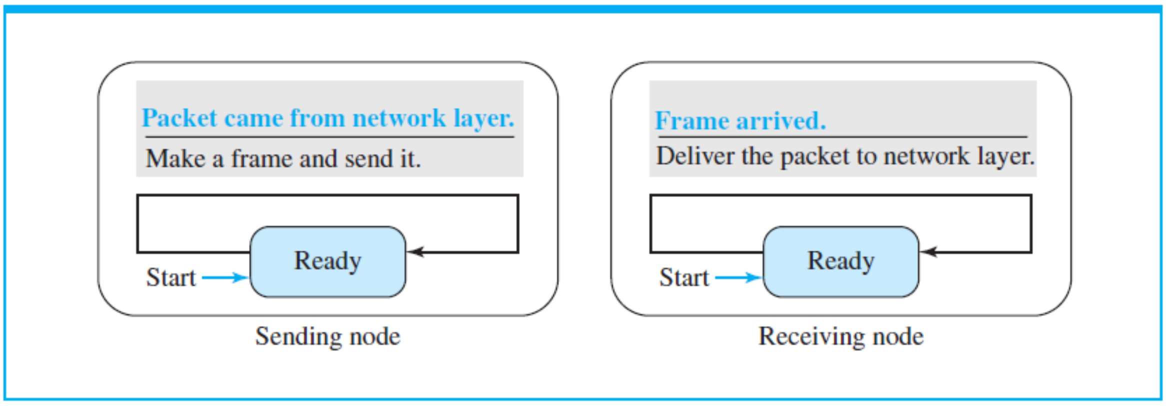
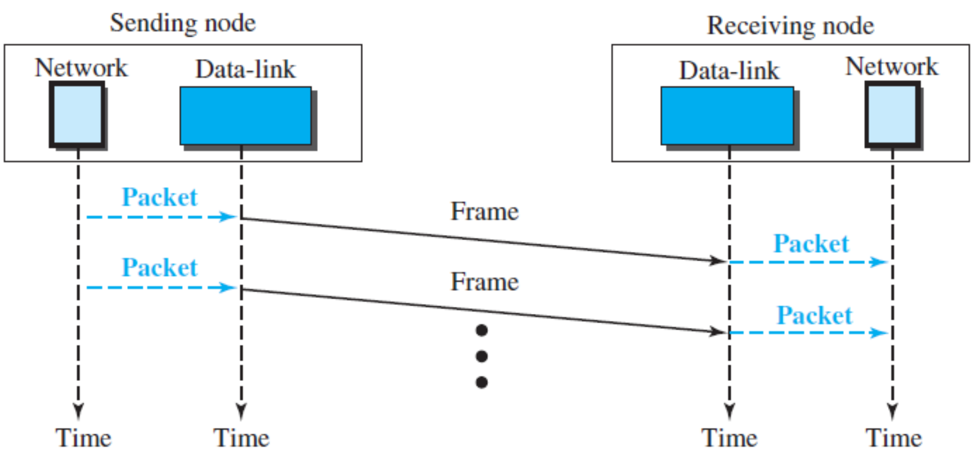

Packet came from network layer.
Make a frame and send it.
Start
Ready
Sending node
Frame arrived.
Deliver the packet to network layer.
Start
Ready
Receiving node

Figure 11.9 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.

Figure 11.9 *Flow diagram for Example 11.2*



2. Stop-and-Wait Protocol

This uses both flow and error cont00.
rol. 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 a signal 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 11.10 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.

Figure 11.10 *Stop-and-Wait protocol*

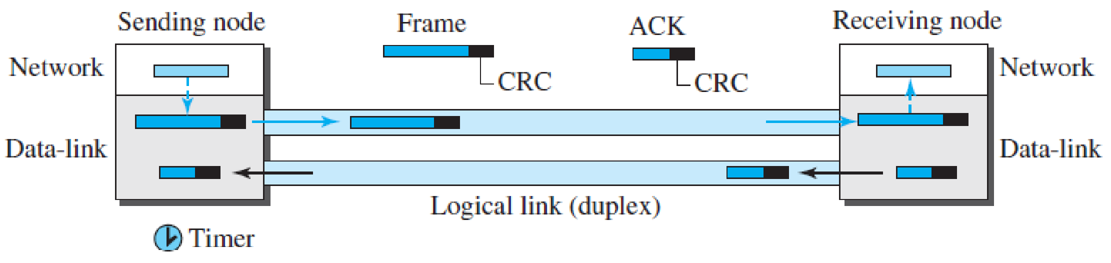
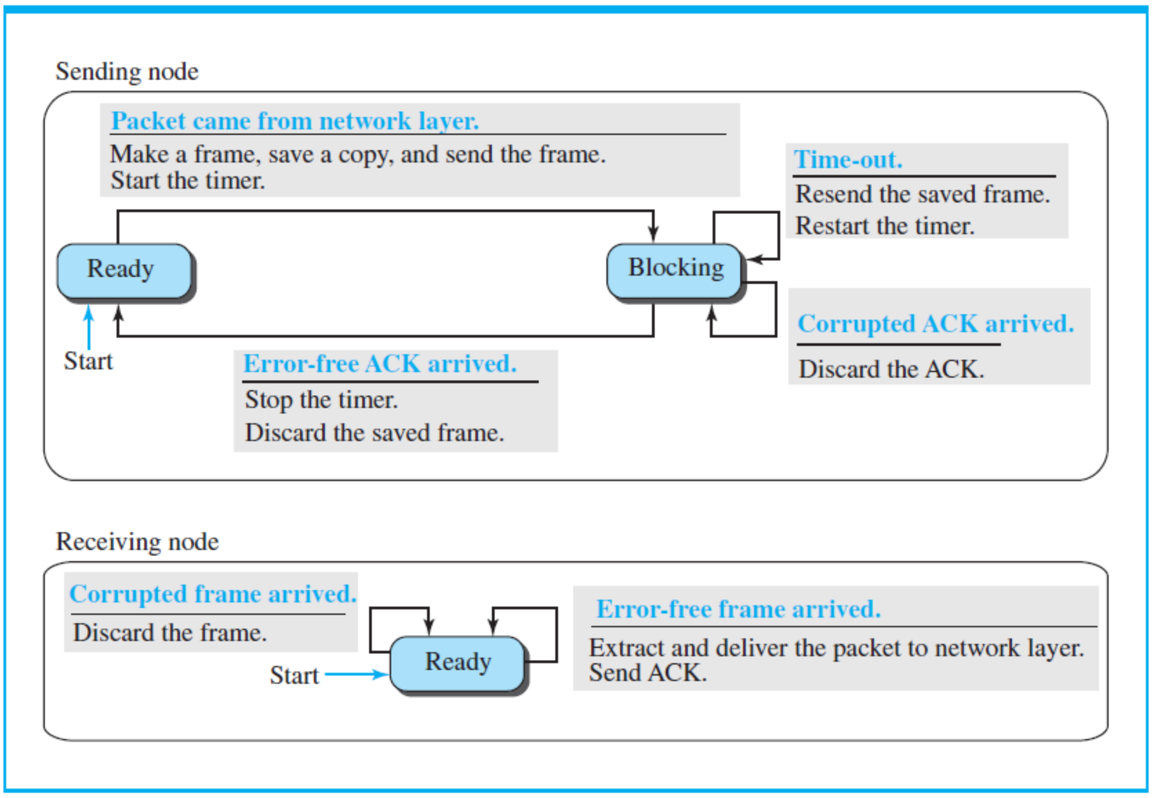


Figure 11.11 shows the FSMs for our primitive Stop-and-Wait protocol.

Figure 11.11 *FSM for the Stop-and-Wait protocol*



We describe the sender and receiver states below.

Sender States

The sender is initially in the ready state, but it can move between the ready and blocking state.

- **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:

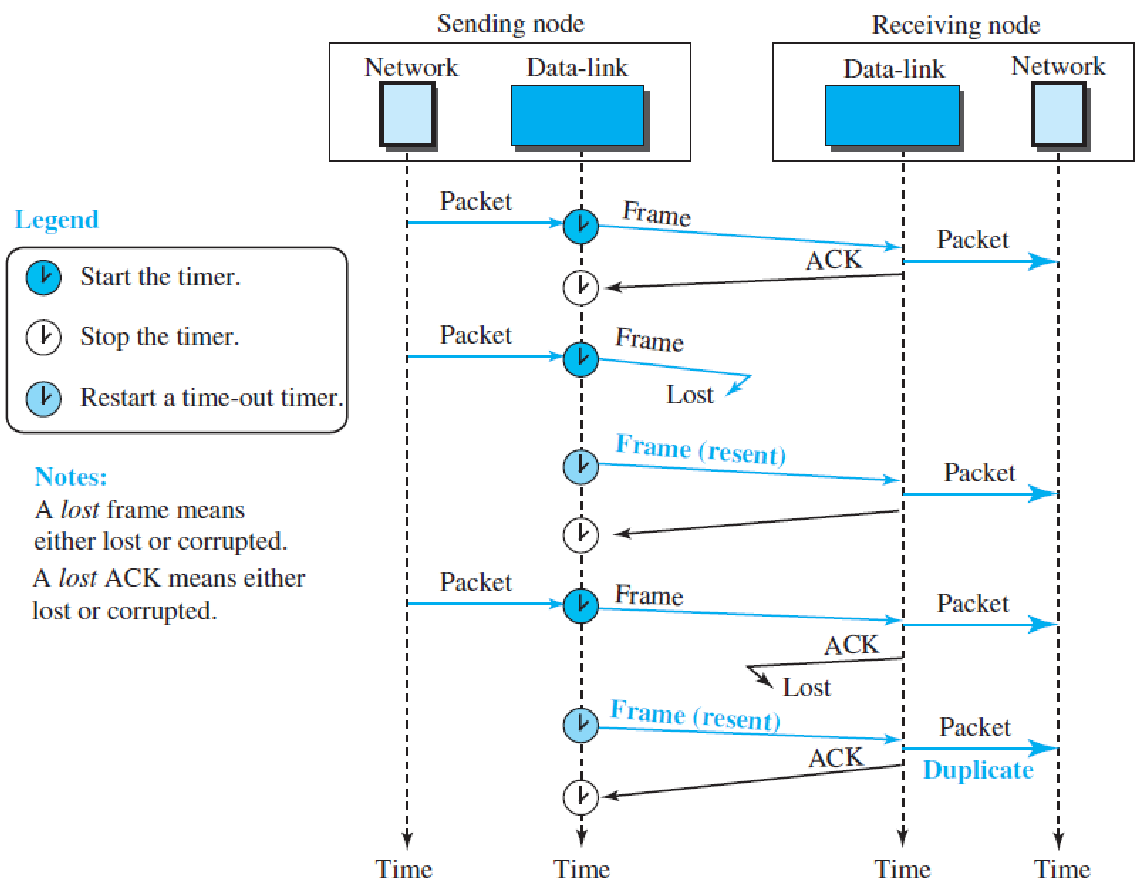
a. If an error-free frame arrives, the message in the frame is delivered to the network layer and an ACK is sent.

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

Example 11.3

Figure 11.12 shows an example. 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.

Figure 11.12 *Flow diagram for Example 11.3*



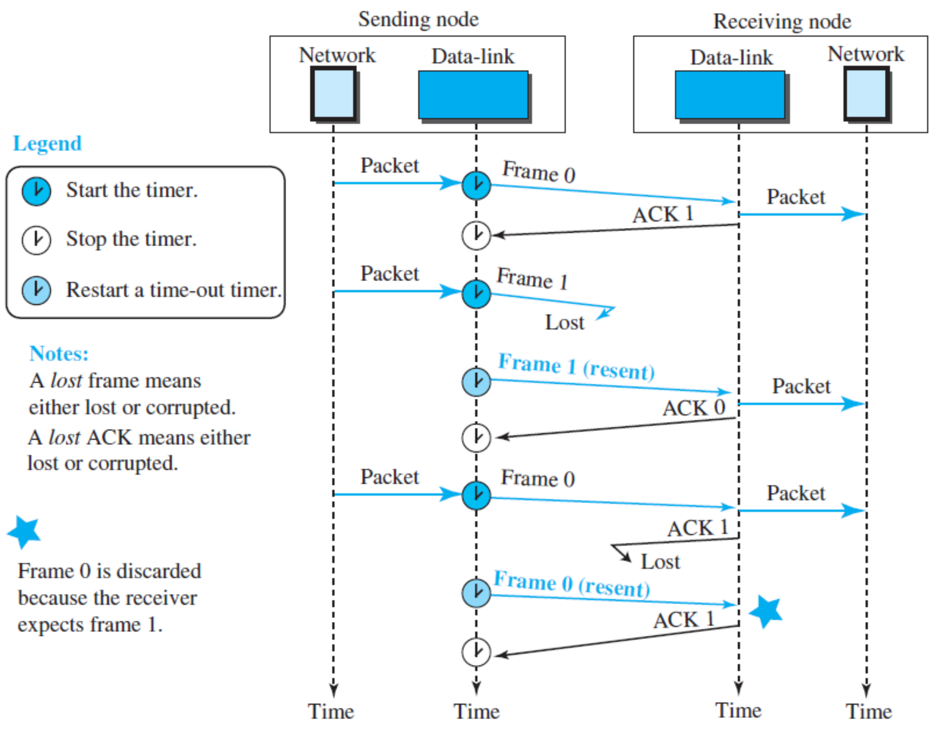
Sequence and Acknowledgment Numbers

We saw a problem in Example 11.3 that needs to be addressed and corrected. Duplicate packets, as much as corrupted packets, need to be avoided. As an example, assume we are ordering some item online. If each packet defines the specification of an item to be ordered, duplicate packets mean ordering an item more than once. To correct the problem in Example 11.3, 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, . . . ; the acknowledgment numbers can also be 1, 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.

Example 11.4

Figure 11.13 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.

Figure 11.13 *Flow diagram for Example 11.4*



3. Piggybacking

The two protocols 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 datalink layer more complicated, it is not a common practice.

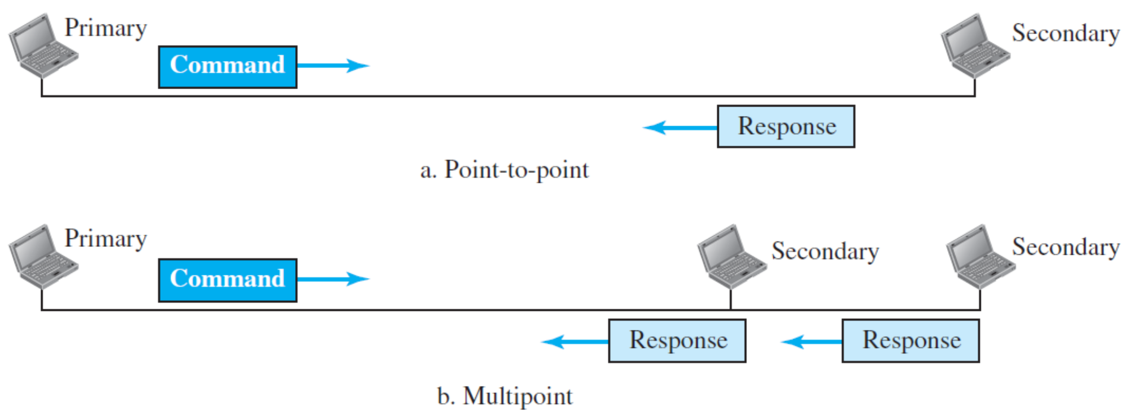
High Level Data Link Control (HDLC)

HDLC is a bit-oriented protocol for communication over point-to-point and multipoint links. It implements the Stop-and-Wait protocol

1. Configurations and Transfer Modes

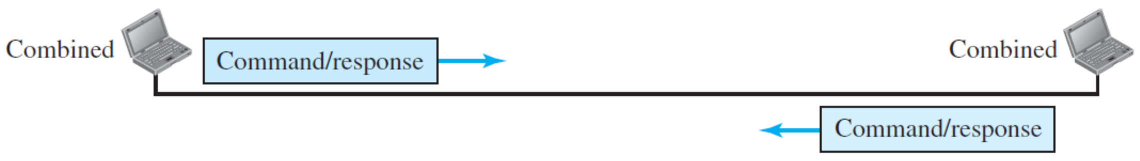
HDLC provides two common transfer modes that can be used in different configurations: normal response mode (NRM) and asynchronous balanced mode (ABM). In normal response mode (NRM), the station configuration is unbalanced. We have one primary station and multiple secondary stations. A primary station can send commands; a secondary station can only respond. The NRM is used for both point-to-point and multipoint links, as shown in Figure 11.14.

Figure 11.14 *Normal response mode*



In ABM, the configuration is balanced. The link is point-to-point, and each station can function as a primary and a secondary (acting as peers), as shown in Figure 11.15. This is the common mode today.

Figure 11.15 *Asynchronous balanced mode*



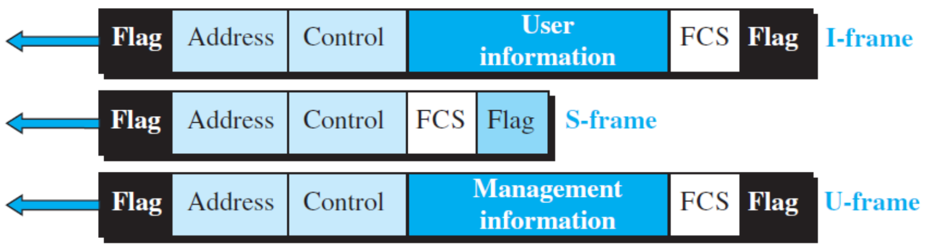
2. Framing

HDLC defines three types of frames: Each type of frame serves as an envelope for the transmission of a different type of message.

1. information frames (I-frames): are used to data-link user data and control information relating to user data (piggybacking).
2. supervisory frames (S-frames): are used only to transport control information
3. unnumbered frames (U-frames): are reserved for system management. Information carried by U-frames is intended for managing the link itself.

Each frame in HDLC may contain up to six fields, as shown in Figure 11.16: a beginning flag field, an address field, a control field, an information field, a frame check sequence (FCS) field, and an ending flag field. In multiple-frame transmissions, the ending flag of one frame can serve as the beginning flag of the next frame.

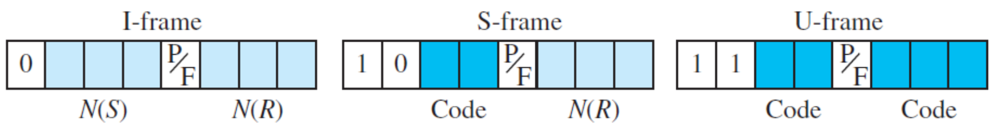
Figure 11.16 *HDLC frames*



- ***Flag field.*** This field contains synchronization pattern 01111110, which identifies both the beginning and the end of a frame.
- ***Address field.*** This field contains the address of the secondary station. If a primary station created the frame, it contains a *to* address. If a secondary station creates the frame, it contains a *from* address. The address field can be one byte or several bytes long, depending on the needs of the network.
- ***Control field.*** The control field is one or two bytes used for flow and error control.
- ***Information field.*** The information field contains the user's data from the network layer or management information. Its length can vary from one network to another.
- ***FCS field.*** The frame check sequence (FCS) is the HDLC error detection field. It can contain either a 2- or 4-byte CRC.

The control field determines the type of frame and defines its functionality. The format is specific for the type of frame, as shown in Figure 11.17.

Figure 11.17 *Control field format for the different frame types*



Control Field for I-Frames

I-frames are designed to carry user data from the network layer. In addition, they can include flow- and error-control information (piggybacking). The subfields in the control field are used to define these functions. The first bit defines the type. If the first bit of the control field is 0, this means the frame is an I-frame. The next 3 bits, called *N*(*S*), define the sequence number of the frame. Note that with 3 bits, we can define a sequence number between 0 and 7. The last 3 bits, called *N*(*R*), correspond to the acknowledgment number when piggybacking is used. The single bit between *N*(*S*) and *N*(*R*) is called the P/F bit. The P/F field is a single bit with a dual purpose. It has meaning only when it is set (bit = 1) and can mean poll or final. It means *poll* when the frame is sent by a primary station to a secondary (when the address field contains the address of the receiver). It means *final* when the frame is sent by a secondary to a primary (when the address field contains the address of the sender).

Control Field for S-Frames

Supervisory frames are used for flow and error control whenever piggybacking is either impossible or inappropriate. S-frames do not have information fields. If the first 2 bits of the control field are 10, this means the frame is an S-frame. The last 3 bits, called *N*(*R*), correspond to the acknowledgment number (ACK) or negative acknowledgment number (NAK), depending on the type of S-frame. The 2 bits called *code* are used to define the type of S-frame itself. With 2 bits, we can have four types of S-frames, as described below:

❖ ***Receive ready (RR).*** If the value of the code subfield is 00, it is an RR S-frame. This kind of frame acknowledges the receipt of a safe and sound frame or group of frames. In this case, the value of the *N*(*R*) field defines the acknowledgment number.

❖ ***Receive not ready (RNR).*** If the value of the code subfield is 10, it is an RNR Sframe. This kind of frame is an RR frame with additional functions. It acknowledges the receipt of a frame or group of frames, and it announces that the receiver is busy and cannot receive more frames. It acts as a kind of congestion-control mechanism by asking the sender to slow down. The value of *N*(*R*) is the acknowledgment number.

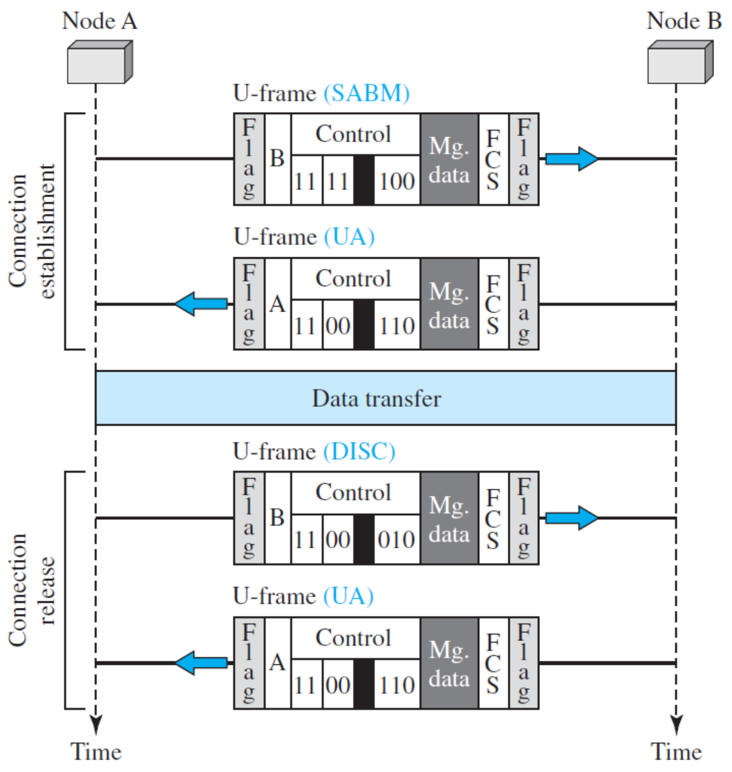
❖ ***Reject (REJ).*** If the value of the code subfield is 01, it is an REJ S-frame. This is a NAK frame, but not like the one used for Selective Repeat ARQ. It is a NAK that can be used in Go-Back-*N* ARQ to improve the efficiency of the process by informing the sender, before the sender timer expires, that the last frame is lost or damaged. The value of *N*(*R*) is the negative acknowledgment number.

❖ ***Selective reject (SREJ).*** If the value of the code subfield is 11, it is an SREJ Sframe. This is a NAK frame used in Selective Repeat ARQ. Note that the HDLC Protocol uses the term *selective reject* instead of *selective repeat*. The value of *N*(*R*) is the negative acknowledgment number.

Control Field for U-Frames

Unnumbered frames are used to exchange session management and control information between connected devices. Unlike S-frames, U-frames contain an information field, but one used for system management information, not user data. As with S-frames, however, much of the information carried by U-frames is contained in codes included in the control field. U-frame codes are divided into two sections: a 2-bit prefix before the P/ F bit and a 3-bit suffix after the P/F bit. Together, these two segments (5 bits) can be used to create up to 32 different types of U-frames. Figure 11.18 shows how U-frames can be used for connection establishment and connection release. Node A asks for a connection with a set asynchronous balanced mode (SABM) frame; node B gives a positive response with an unnumbered acknowledgment (UA) frame. After these two exchanges, data can be transferred between the two nodes (not shown in the figure). After data transfer, node A sends a DISC (disconnect) frame to release the connection; it is confirmed by node B responding with a UA (unnumbered acknowledgment).

Figure 11.18 *Example of connection and disconnection*



Point-to-Point Protocol (PPP):

One of the most common protocols for point-to-point access is the Point-to-Point Protocol (PPP). Today, millions of Internet users who need to connect their home computers to the server of an Internet service provider use PPP. The majority of these users have a traditional modem; they are connected to the Internet through a telephone line, which provides the services of the physical layer. But to control and manage the transfer of data, there is a need for a point-to-point protocol at the data-link layer. PPP is by far the most common.

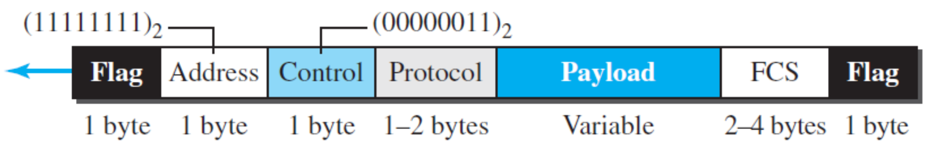
1. Framing

PPP uses a character-oriented (or byte-oriented) frame. Figure 11.20 shows the format of a PPP frame. The description of each field follows:

❑ **Flag.** A PPP frame starts and ends with a 1-byte flag with the bit pattern 01111110.

❑**Address.** The address field in this protocol is a constant value and set to 11111111 (broadcast address).

Figure 11.20 *PPP frame format*



❑ **Control.** This field is set to the constant value 00000011 (imitating unnumbered frames in HDLC). PPP does not provide any flow control. Error control is also limited to error detection.

❑ **Protocol.** The protocol field defines what is being carried in the data field: either user data or other information. This field is by default 2 bytes long, but the two parties can agree to use only 1 byte.

❑ **Payload field.** This field carries either the user data or other information. The data field is a sequence of bytes with the default of a maximum of 1500 bytes; but this can be changed during negotiation. The data field is byte-stuffed if the flag byte pattern appears in this field. Because there is no field defining the size of the data field, padding is needed if the size is less than the maximum default value or the maximum negotiated value.

❑ **FCS.** The frame check sequence (FCS) is simply a 2-byte or 4-byte standard CRC.

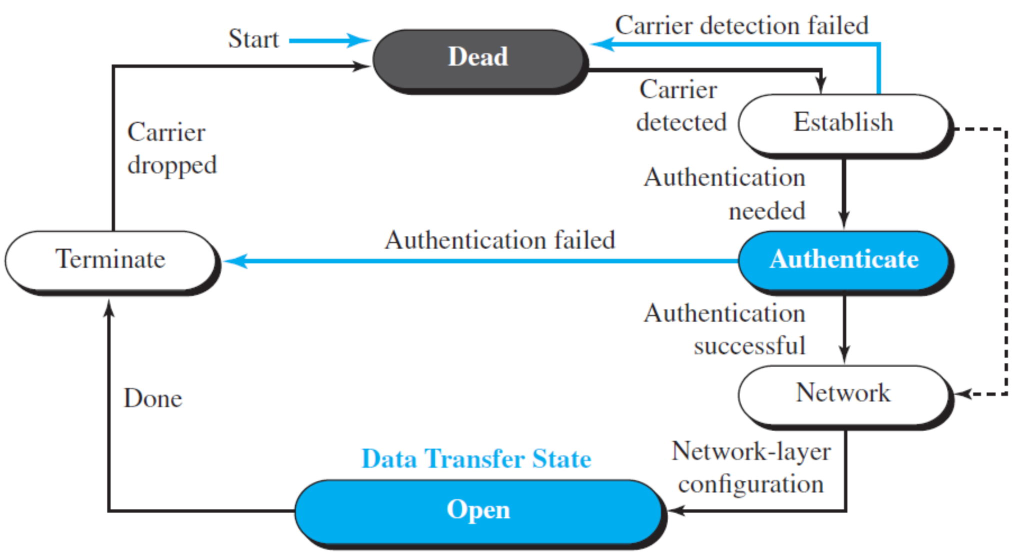
Byte Stuffing

Since PPP is a byte-oriented protocol, the flag in PPP is a byte that needs to be escaped whenever it appears in the data section of the frame. The escape byte is 01111101, which means that every time the flaglike pattern appears in the data, this extra byte is stuffed to tell the receiver that the next byte is not a flag. Obviously, the escape byte itself should be stuffed with another escape byte.

2. Transition Phases

A PPP connection goes through phases which can be shown in a transition phase diagram (see Figure 11.21). The transition diagram, which is an FSM, starts with the dead state. In this state, there is no active carrier (at the physical layer) and the line is quiet. When one of the two nodes starts the communication, the connection goes into the establish state. In this state, options are negotiated between the two parties. If the two parties agree that they need authentication (for example, if they do not know each other), then the system needs to do authentication (an extra step); otherwise, the parties can simply start communication. The link-control protocol packets, are used for this purpose. Several packets may be exchanged here. Data transfer takes place in the open state. When a connection reaches this state, the exchange of data packets can be started. The connection remains in this state until one of the endpoints wants to terminate the connection. In this case, the system goes to the terminate state. The system remains in this state until the carrier (physical-layer signal) is dropped, which moves the system to the dead state again.

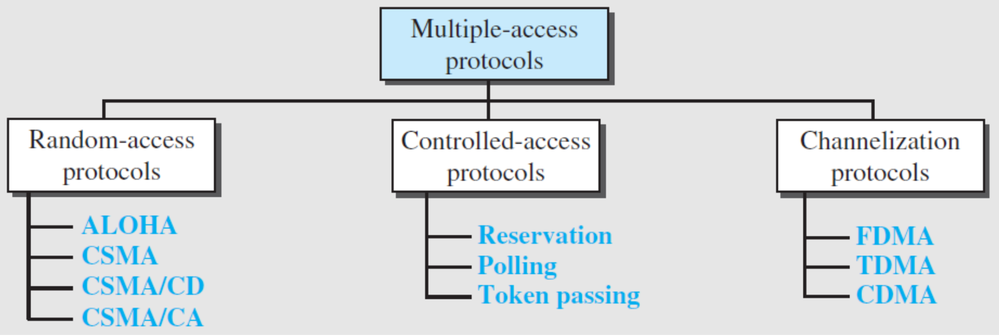
Figure 11.21 *Transition phases*



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. The problem of controlling the access to the medium is similar to the rules of speaking in an assembly. The procedures guarantee that the right to speak is upheld and ensure that two people do not speak at the same time, do not interrupt each other, do not monopolize the discussion, and so on. Many protocols have been devised to handle access to a shared link. All of these protocols belong to a sublayer in the data-link layer called *media access control (MAC).* We categorize them into three groups, as shown in Figure 12.1.

Figure 12.1 *Taxonomy of multiple-access protocols*



ALOHA (Advocates of Linux Open-source Hawaii Association) is a multiple access protocol for transmission of data via a shared network channel. It operates in the medium access control sublayer (MAC sublayer) of the open systems interconnection (OSI) model.

Carrier Sense Multiple Access (CSMA)-here every node on the network must monitor the bus for a period of no activity before trying to send a message on that bus (carrier sense).

Frequency Division Multiple Access (FDMA) is a type of channelization protocol. In this bandwidth is divided into various frequency bands. Each station is allocated with band to send data and that band is reserved for particular station for all the time.(FM radio)

Time Division Multiple Access (TDMA) is the channelization protocol in which bandwidth of channel is divided into various stations on the time basis. There is a time slot given to each station, the station can transmit data during that time slot only.(writing test)

Code Division Multiple Access (CDMA) : In CDMA, all the stations can transmit data simultaneously. It allows each station to transmit data over the entire frequency all the time. Multiple simultaneous transmissions are separated by unique code sequence. Each user is assigned with a unique code sequence.(amount withdrawal)

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. To avoid access conflict or to resolve it when it happens, each station follows a procedure that answers the following questions:

- ❑ When can the station access the medium?
- ❑ What can the station do if the medium is busy?
- ❑ How can the station determine the success or failure of the transmission?
- ❑ What can the station do if there is an access conflict?

The random-access methods have evolved from a very interesting protocol known as *ALOHA*, which used a very simple procedure called ***multiple access (MA).*** The method was improved with the addition of a procedure that forces the station to sense the medium before transmitting. This was called *carrier sense multiple access* (*CSMA*). This method later evolved into two parallel methods: *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.

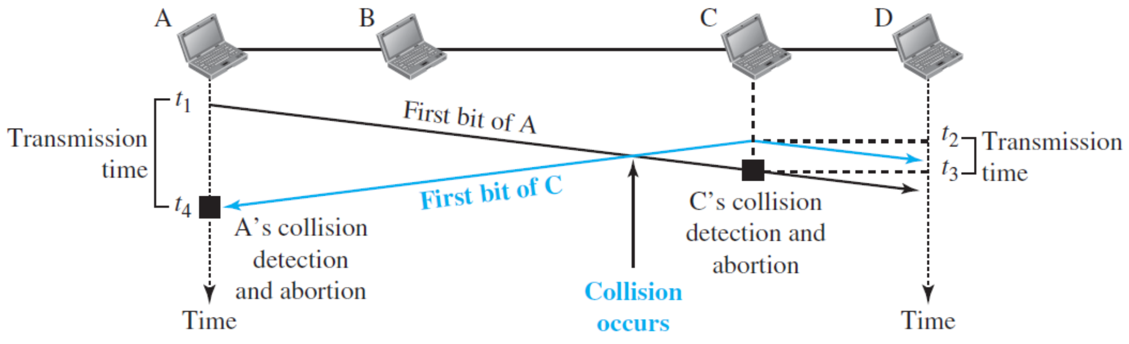
CSMA

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

CSMA/CD

The CSMA method does not specify the procedure following a collision. Carrier sense multiple access with collision detection (CSMA/CD) augments the algorithm to handle the collision.
In this method, a station monitors the medium after it sends a frame to see if the transmission was successful. If so, the station is finished. If, however, there is a collision, the frame is sent again.
To better understand CSMA/CD, let us look at the first bits transmitted by the two stations involved in the collision. Although each station continues to send bits in the frame until it detects the collision, we show what happens as the first bits collide. In Figure 12.11, stations A and C are involved in the collision.

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



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

Minimum Frame Size

For CSMA/CD to work, we need a restriction on the frame size. Before sending the last bit of the frame, the sending station must detect a collision, if any, and abort the transmission. This is so because the station, once the entire frame is sent, does not keep a copy of the frame and does not monitor the line for collision detection. Therefore, the frame transmission time T_{fr} must be at least two times the maximum propagation time T_p. To understand the reason, let us think about the worst-case scenario. If the two stations involved in a collision are the maximum distance apart, the signal from the first takes time T_p to reach the second, and the effect of the collision takes another time TP to reach the first. So the requirement is that the first station must still be transmitting after $2T_p$.

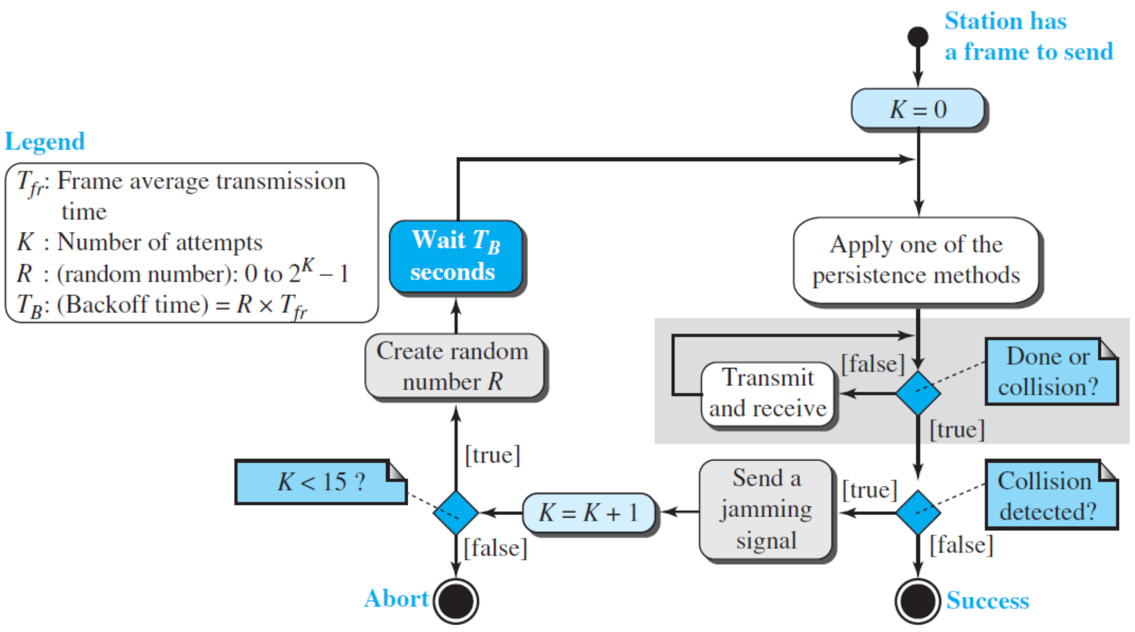
Example 12.5

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 $T_{fr} = 2 \times T_p = 51.2$ 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 10 Mbps × 51.2 μs = 512 bits or 64 bytes.

Figure 12.13 *Flow diagram for the CSMA/CD*



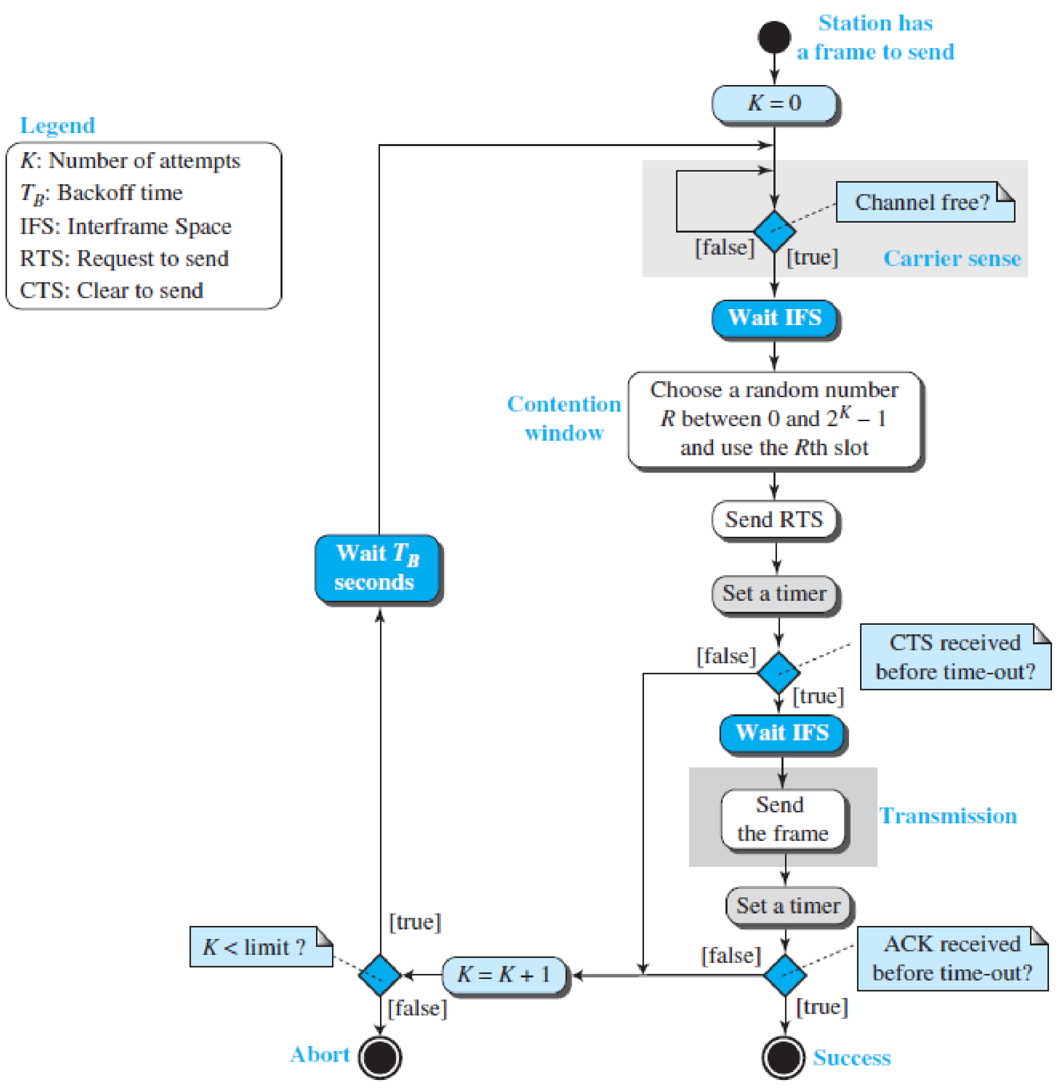
In CSMA/CD, transmission and collision detection are continuous processes. We do not send the entire frame and then look for a collision. The station transmits and receives continuously and simultaneously (using two different ports or a bidirectional port). We use a loop to show that transmission is a continuous process. We constantly monitor in order to detect one of two conditions: either transmission is finished or a collision is detected. Either event stops transmission. When we come out of the loop, if a collision has not been detected, it means that transmission is complete; the entire frame is transmitted. Otherwise, a collision has occurred. CSMA/CD sends a short **jamming signal** to make sure that all other stations become aware of the collision.

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 interframe space, the contention window, and acknowledgments, as shown in Figure 12.15.

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

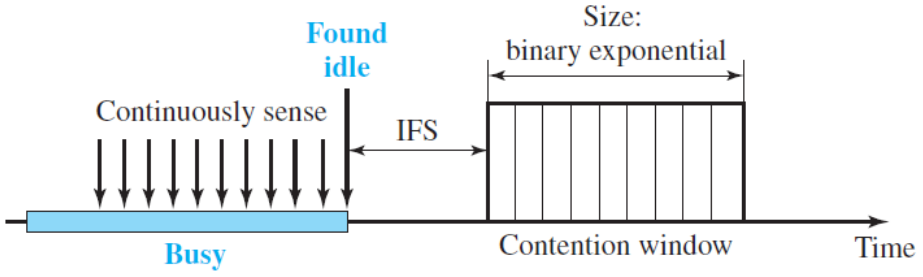
Figure 12.15 *Flow diagram of CSMA/CA*



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 backoff 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. See Figure 12.16.

Figure 12.16 *Contention window*



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

Frame Exchange Time Line

Figure 12.17 shows the exchange of data and control frames in time.

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

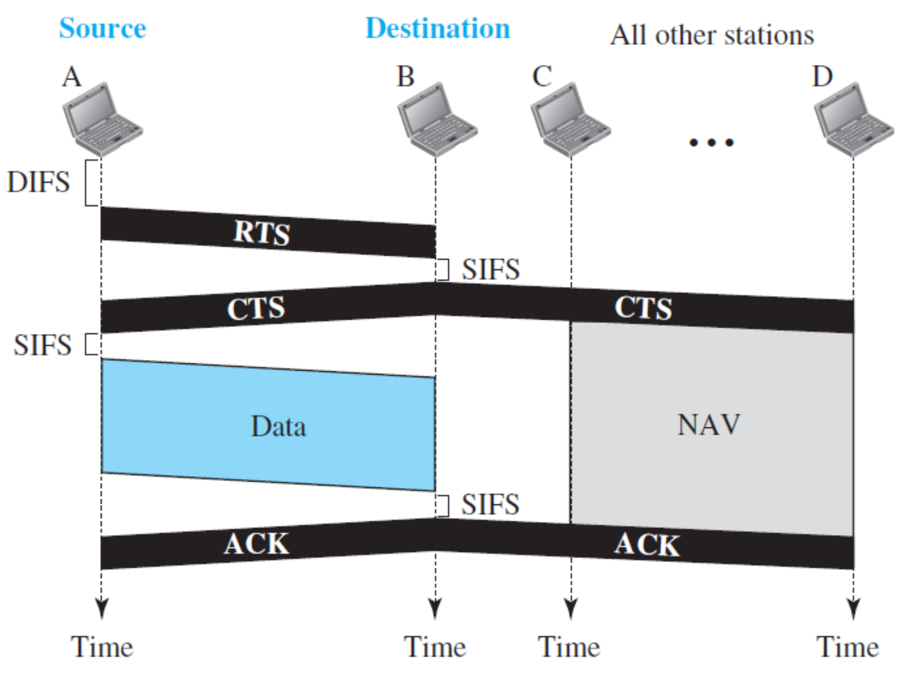
a. The channel uses a persistence strategy with backoff until the channel is idle.

b. After the station is found to be idle, the station waits for a period of time called the ***DCF*** Distributed coordination function ***interframe space (DIFS);*** then the station sends a control frame called the *request to send (RTS).* DIFS is calculated as the sum of SIFS and twice the slot time.

2. After receiving the RTS and waiting a period of time called the ***short interframe 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. The source station sends data after waiting an amount of time equal to SIFS. The duration of SIFS is usually 10 µs.

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.

Figure 12.17 *CSMA/CA and NAV*



Network Allocation Vector

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*.

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 12.17 shows the idea of 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?* 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 12.17 also shows that the RTS message from B reaches A, but not C. However, because both B and C are within the range of A, 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.

CSMA/CA and Wireless Networks

CSMA/CA was mostly intended for use in wireless networks. The procedure described above, however, is not sophisticated enough to handle some particular issues related to wireless networks, such as hidden terminals or exposed terminals.