UNIT 1

Fundamentals in communication

1

FUNDAMENTALS OF DATA COMMUNICATION

Unit Structure

1.1 Introduction to data communication
1.2 Characteristics of a data communication system
1.3 Data & Signal
    1.3.1 Analog & Digital data
    1.3.2 Analog & Digital Signal
    1.3.3 Periodic & Non-Periodic Signal
    1.3.4 Composite Signal
1.4 Signal Encoding
1.5 Synchronization
1.6 Digital Data to Digital Signal
    1.6.1 Coding methods
    1.6.2 Line Encoding
    1.6.3 Block Coding
Further Reading & References

1.1. INTRODUCTION TO DATA COMMUNICATION

- The main objective of data communication and networking is to enable seamless exchange of data between any two points in the world.
- This exchange of data takes place over a computer network.
- A computer network is a collection of interconnected nodes. Each node is a device capable of sending or receiving data.

Data & Information

- **Data** refers to the raw facts that are collected while **information** refers to processed data that enables us to take decisions.

- Ex. When result of a particular test is declared it contains data of all students, when you find the marks you have scored you have the information that lets you know whether you have passed or failed.
- The word ***data*** refers to any information which is presented in a form that is agreed and accepted upon by is creators and users.

1.2. CHARACTERISTICS OF A DATA COMMUNICATION SYSTEM

The effectiveness of any data communications system depends upon the following four fundamental characteristics:

1. **Delivery**: The data should be delivered to the correct destination and correct user.
2. **Accuracy**: The communication system should deliver the data accurately, without introducing any errors. The data may get corrupted during transmission affecting the accuracy of the delivered data.
3. **Timeliness**: Audio and Video data has to be delivered in a timely manner without any delay, such a data delivery is called real time transmission of data.
4. **Jitter** : It is the variation in the packet arrival time. Unven Jitter may affect the timeliness of data being transmitted.

1.3. DATA & SIGNAL

To be transmitted, data must be transformed to electromagnetic signals.

1.3.1 Data can be Analog or Digital.

1. **Analog data** refers to information that is continuous; ex. sounds made by a human voice

2. **Digital data** refers to information that has discrete states. Digital data take on discrete values.

3. For example, data are stored in computer memory in the form of Os and 1s

1.3.2. Signals can be of two types:

1. **Analog Signal:** They have infinite values in a range.

2. **Digital Signal:** They have limited number of defined values

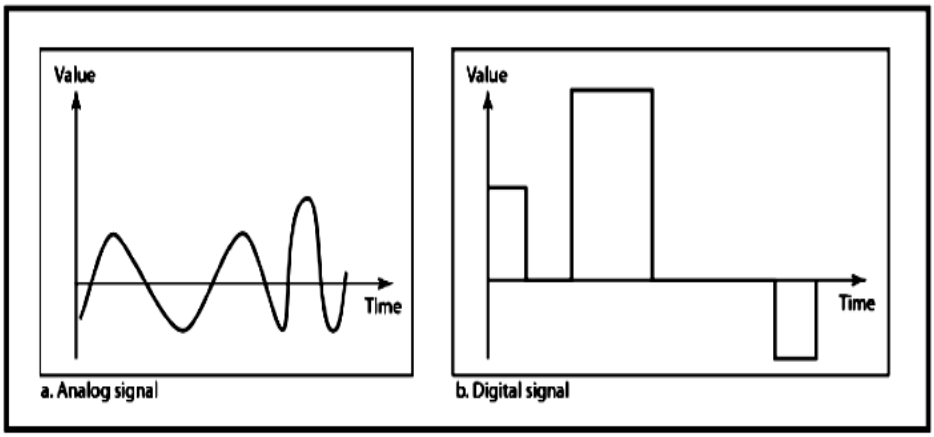


Figure: a. Analog Signal b. Digital Signal

1.3.3. Periodic & Non Periodic Signals

- Signals which repeat itself after a fixed time period are called Periodic Signals.
- Signals which do not repeat itself after a fixed time period are called Non-Periodic Signals.

1.3.4. Composite Signal

- A composite signal is a combination of two or more simple sine waves with different frequency, phase and amplitude.
- If the composite signal is periodic, the decomposition gives a series of signals with discrete frequencies; if the composite signal is non-periodic, the decomposition gives a combination of sine waves with continuous frequencies.

1.4. SIGNAL ENCODING

- Data can be analog or digital, so can be the signal that represents it.
- **Signal encoding** is the conversion from analog/digital data to analog / digital signal.

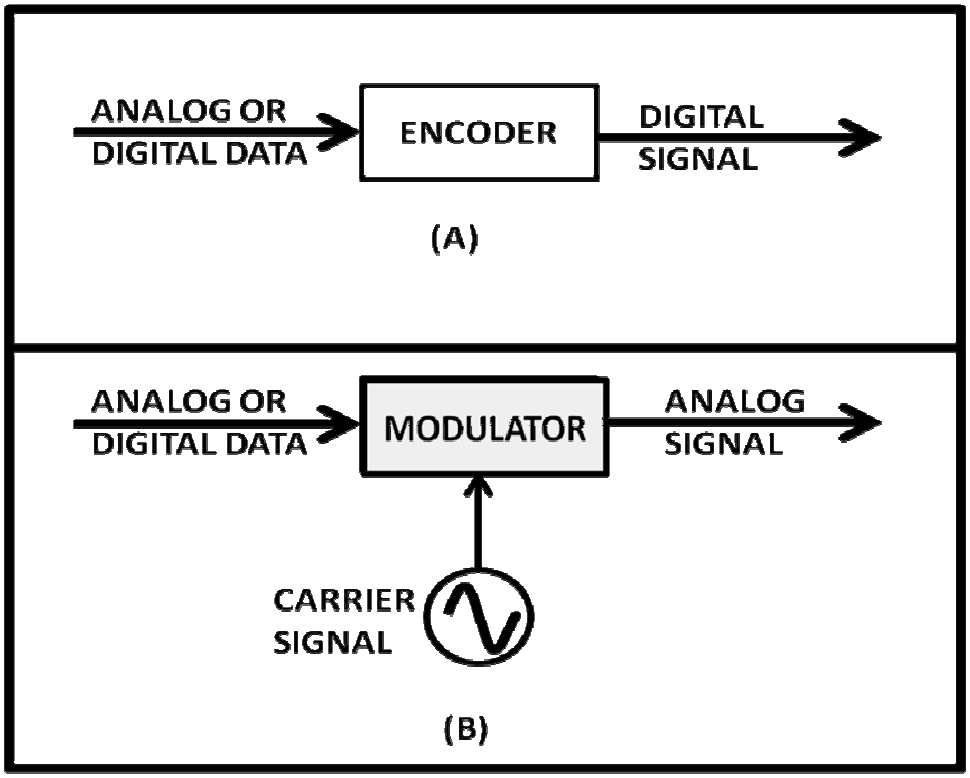


Figure: Signal Encoding

In the Figure above,

A) Demonstrates Digital Signaling where data from an analog/digital source is encoded into Digital Signal.

B) Demonstrates Analog signaling in which the analog/digital source modulates a continuous carrier signal to produce an analog signal.

The possible encodings are:

1. Digital data to Digital Signal
2. Digital data to Analog Signal
3. Analog data to Digital Signal
4. Analog data to Analog Signal

1.5. SYNCHRONIZATION

- In order to receive the signals correctly, the receivers bit intervals must correspond exactly to the senders bit intervals.
- The clock frequency of the transmitter and receiver should be the same.

- If the clock frequency at the receiver is slower or faster than the bit intervals are not matched and the received signal is different than the transmitted one.

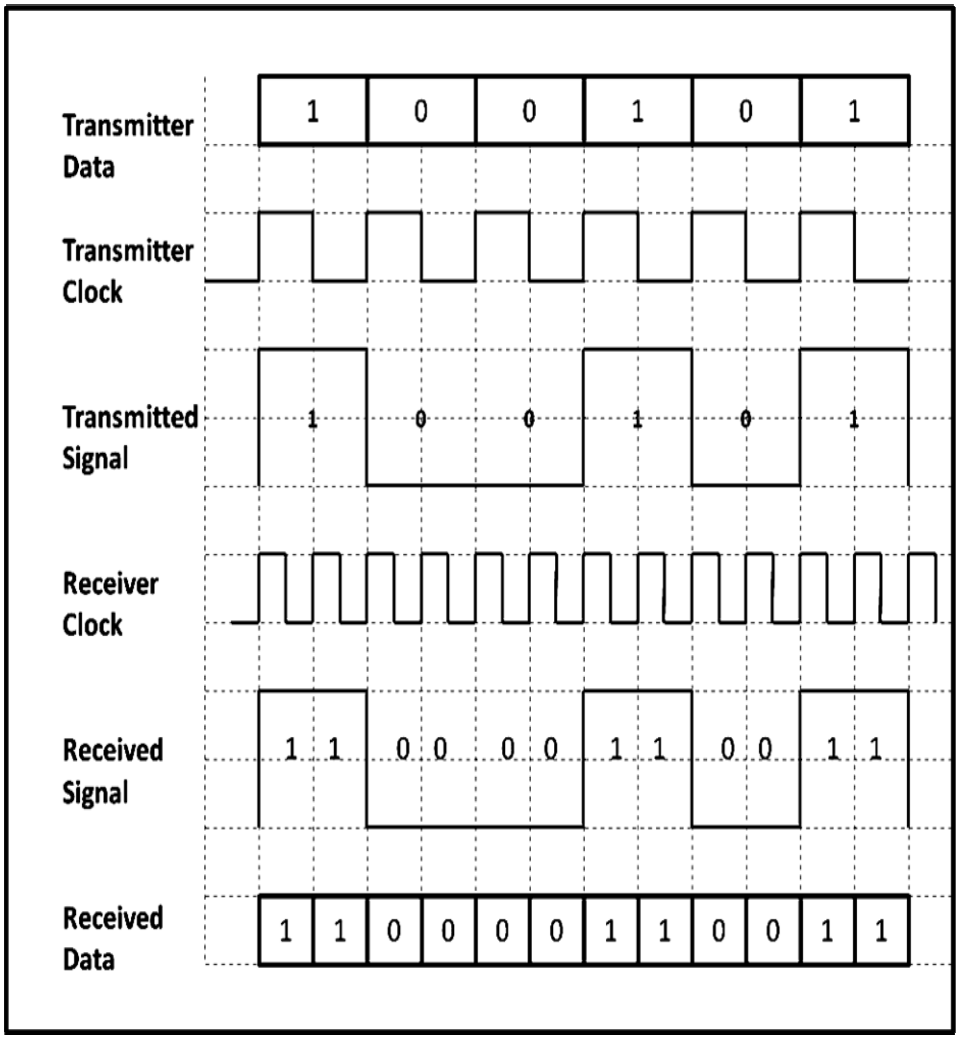


Figure : Synchronization

- In the above figure, the receiver clock frequency is twice that of the transmitter frequency. Hence the received data is totally different than the transmitted one
- To avoid this, receiver and transmitter clocks have to be **synchronized.**
- To achieve this the transmitted digital signal should include timing information which forces synchronization

1.6. DIGITAL DATA TO DIGITAL SIGNAL

1.6.1. Coding methods

Coding methods are used to convert digital data into digital signals.

There are two types of coding methods:
1 Line Coding
2 Block Coding

Scrambling is also one of the ways to convert digital data to digital signals but is not used.

1.6.2. Line Encoding

It is the process of converting Digital data into digital signal.

In other words, it is converting of binary data(i.e. A sequence of bits) into digital signal (i.e. a sequence of discrete, discontinuous voltage pulses)

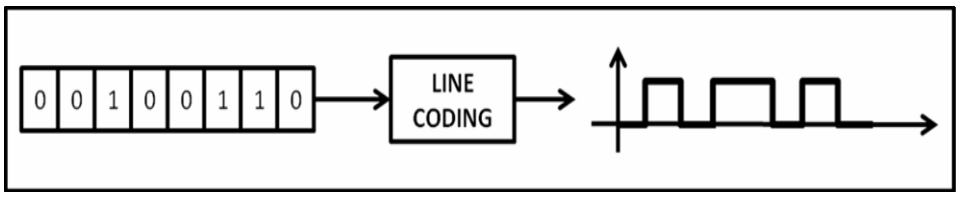


Figure: Line Coding

Classification of Line Codes
The following figure shows the classification of Line coding schemes:

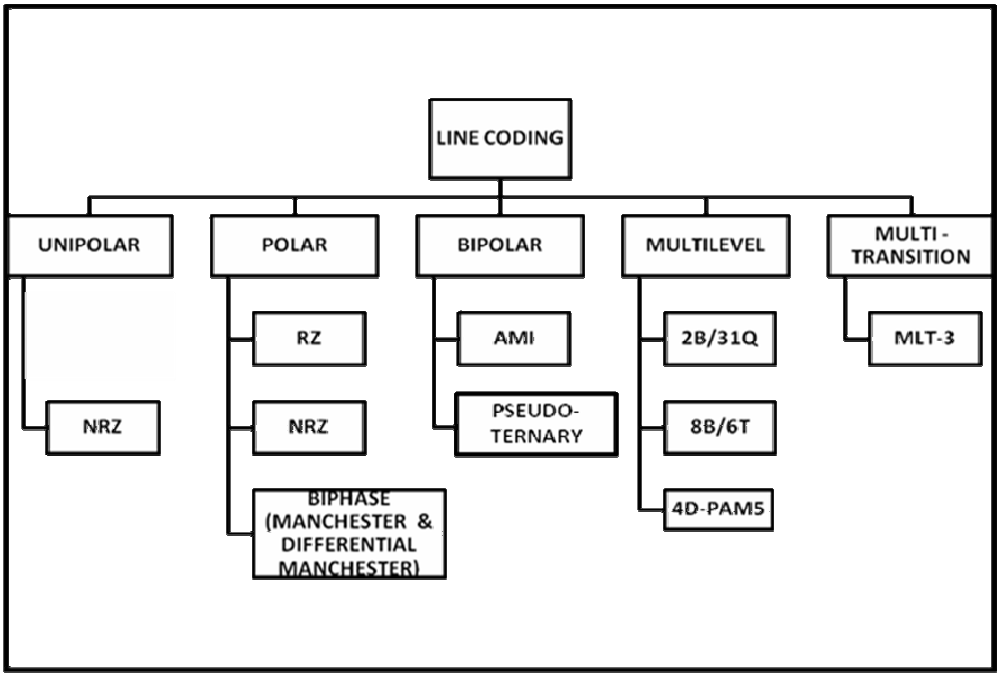


Figure : Classification of line coding schemes

A. Unipolar

- All signal levels are either above or below the time axis.
- NRZ - Non Return to Zero scheme is an example of this code. The signal level does not return to zero during a symbol transmission.

B. Polar

- **NRZ-voltages** are on both sides of the time axis.
- Polar NRZ scheme can be implemented with two voltages. E.g. +V for 1 and -V for 0.
- There are two variations:
    - **NZR - Level (NRZ-L)** - positive voltage for one symbol and negative for the other
    - **NRZ - Inversion (NRZ**-I) - the change or lack of change in polarity determines the value of a symbol. E.g. a “1” symbol inverts the polarity a “0” does not.

- **Polar – RZ**
    - The Return to Zero (RZ) scheme uses three voltage values. +, 0, -.
    - Each symbol has a transition in the middle. Either from high to zero or from low to zero

- More complex as it uses three voltage level. It has no error detection capability

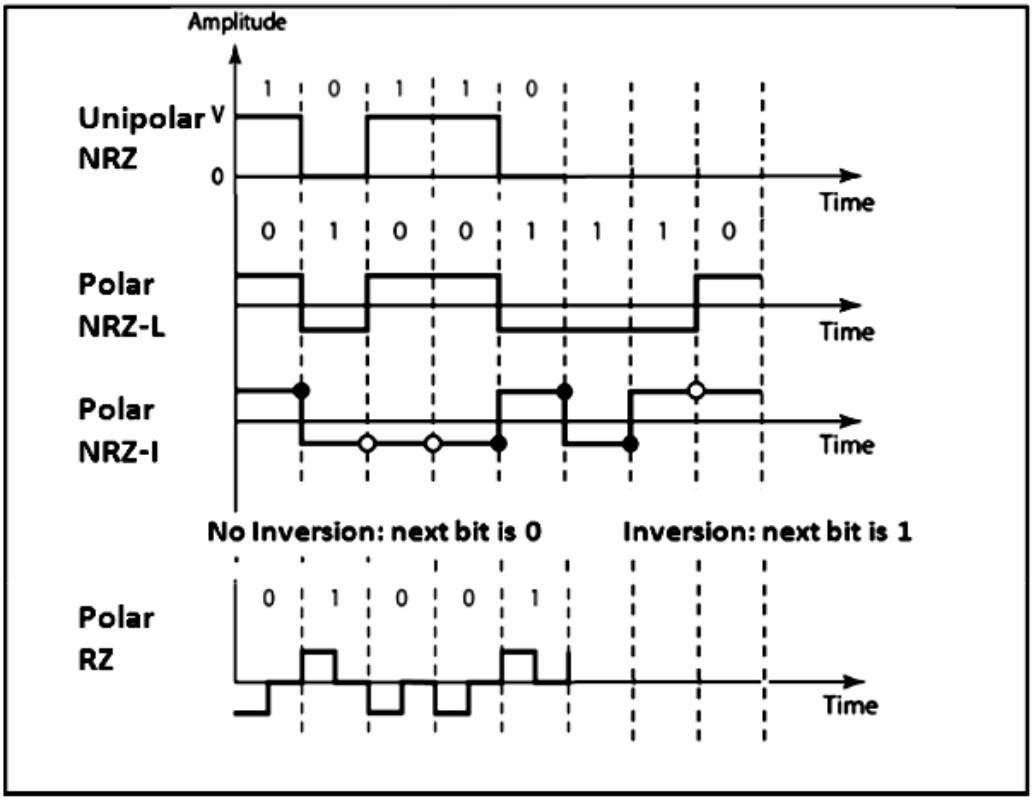


Figure : Unipolar(NRZ) & Polar(RZ & NRZ) Encoding

- **Polar - Biphase: Manchester and Differential Manchester**
  - **Manchester coding** is a combination of NRZ-L and RZ schemes.
    - Every symbol has a level transition in the middle: from high to low or low to high.
    - It uses only two voltage levels.
  - **Differential Manchester coding** consists of combining the NRZ-I and RZ schemes.
    - Every symbol has a level transition in the middle. But the level at the beginning of the symbol is determined by the symbol value. One symbol causes a level change the other does not.

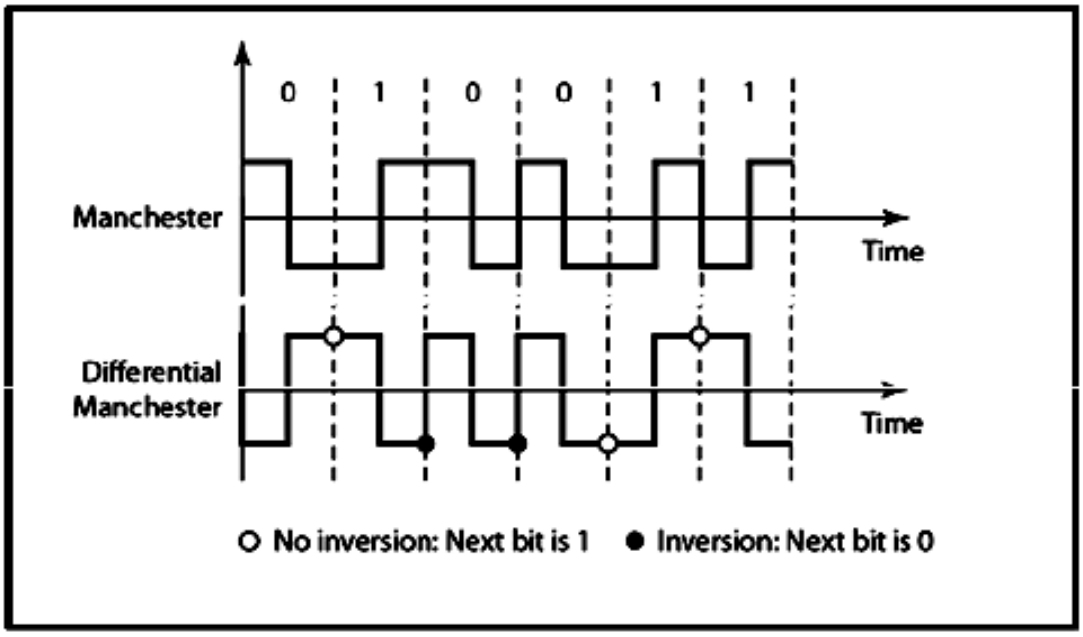


Figure : Polar biphase: Manchester and differential Manchester coding schemes

C. Bipolar - AMI and Pseudoternary

- This coding scheme uses 3 voltage levels: - +, 0, -, to represent the symbols
- Voltage level for one symbol is at “0” and the other alternates between + & -.
- **Bipolar Alternate Mark Inversion (AMI)** - the “0” symbol is represented by zero voltage and the “1” symbol alternates between +V and -V.
- **Pseudoternary** is the reverse of AMI

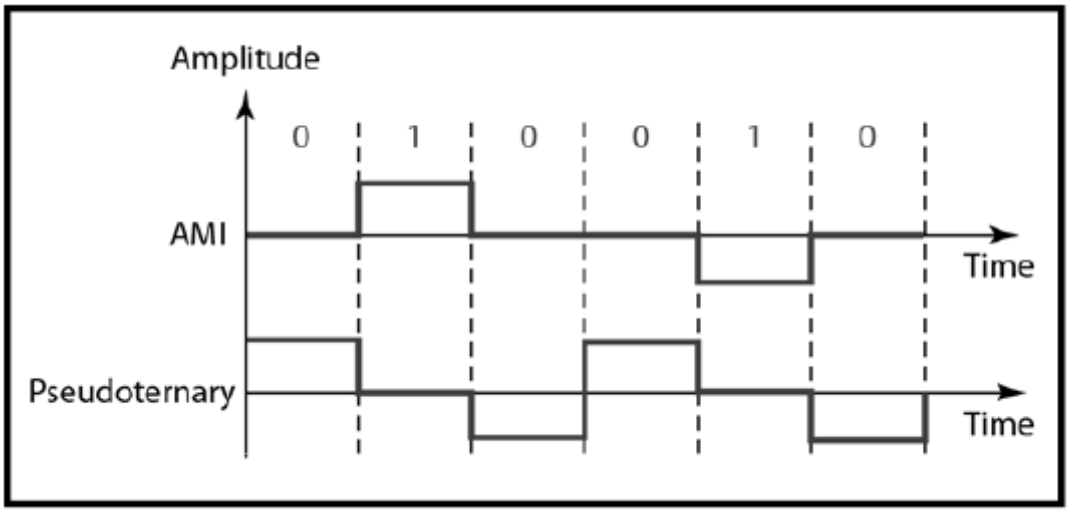


Figure: Bipolar coding scheme - AMI and Pseudoternary

D. Multilevel

- Here the number of data bits is increased per symbol to increase the bit rate.
- 2 types of data element a 1 or a 0 are available, it can be combined into a pattern of ***n*** elements to create $\boldsymbol{2^m}$ symbols.
- Using L signal levels we can have ***n*** signal elements to create $\boldsymbol{L^n}$ signal elements. The following possibilities can occur:
- With 2^m symbols and L^n signals:
  - If $2^m > L^n$ then we cannot represent the data elements, we don't have enough signals.
  - If $2^m = L^n$ then we have an exact mapping of one symbol on one signal.
  - If $2^m < L^n$ then we have more signals than symbols and we can choose the signals that are more distinct to represent the symbols and therefore have better noise immunity and error detection as some signals are not valid
- These types of codings are classified as **mBnL** schemes. In ***mBnL*** schemes, a pattern of *m* data elements is encoded as a pattern of *n* signal elements in which $\boldsymbol{2^m \leq L^n}$.
- **2B1Q (**two binary, one quaternary**)**
- Here m = 2; n = 1 ; Q = 4. It uses data patterns of size 2 and encodes the 2-bit patterns as one signal element belonging to a four-level signal.

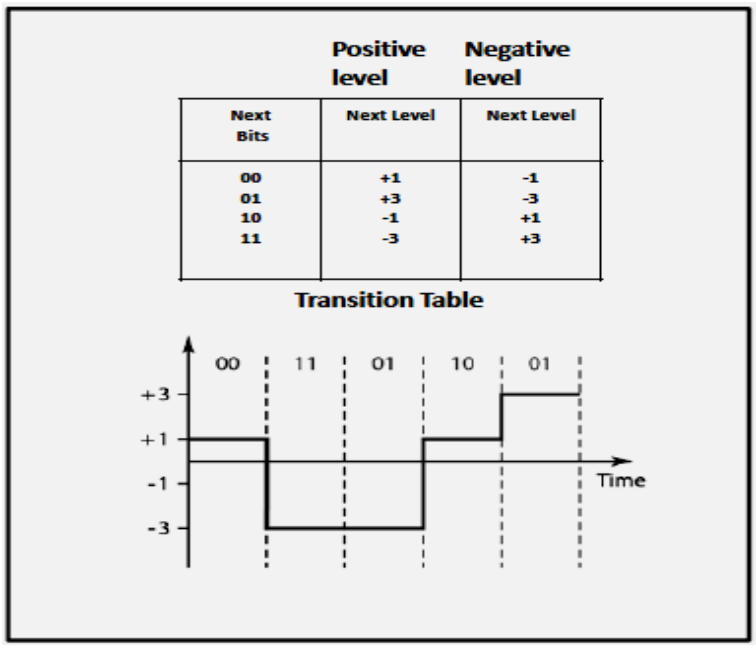


Figure: Multilevel coding scheme : 2B1Q

8B6T(eight binary, six ternary**)**

- Here a pattern of 8 bits is encoded a pattern of 6 signal elements, where the signal has three levels
- Here m = 8; n = 6 ; T = 3
- So we can have 2^8 =256 different data patterns and 3^6 =478 different signal patterns.

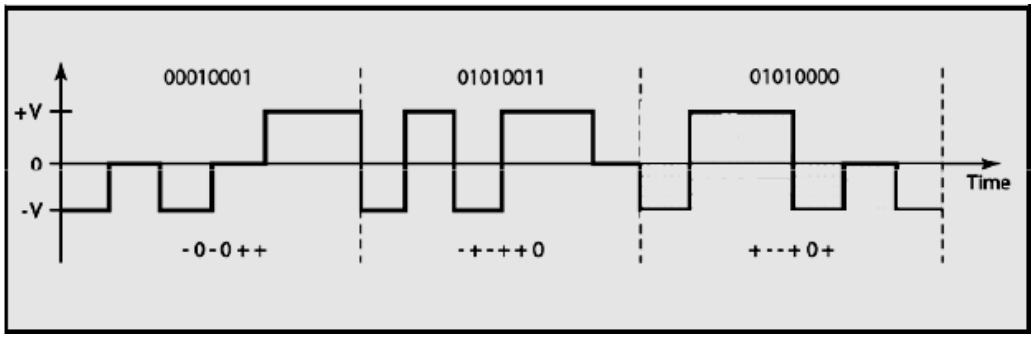


Figure : Multilevel coding scheme : 8B6T

4D-PAM5 (Four Dimensional Five-Level Pulse Amplitude Modulation)

- **4D -**means that data is sent over four channels at the same time.
- It uses five voltage levels, such as -2, -1, 0, 1, and 2.

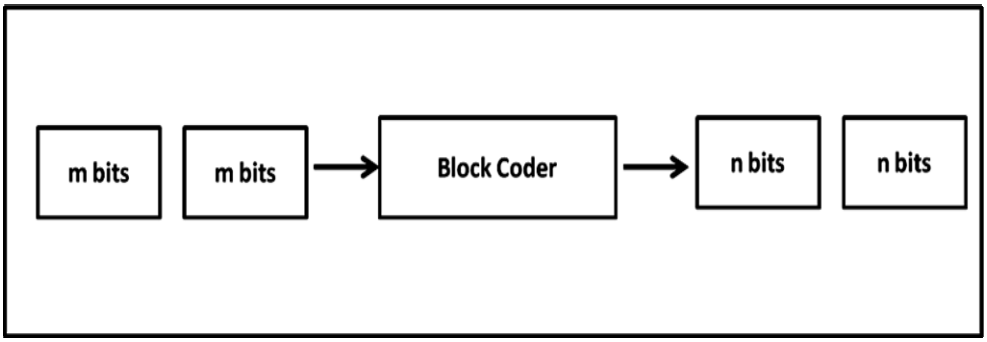
E. Multitransition

- Because of synchronization requirements we force transitions. This can result in very high bandwidth requirements -> more transitions than are bits (e.g. mid bit transition with inversion).
- Codes can be created that are differential at the bit level forcing transitions at bit boundaries. This results in a bandwidth requirement that is equivalent to the bit rate.
- In some instances, the bandwidth requirement may even be lower, due to repetitive patterns resulting in a periodic signal.
- **MLT-3**
    - Signal rate is same as NRZ-I
    - Uses three levels (+*v,* 0, and - *V)* and three transition rules to move between the levels.
        - If the next bit is 0, there is no transition.

- If the next bit is 1 and the current level is not 0, the next level is 0.
- If the next bit is 1 and the current level is 0, the next level is the opposite of the last nonzero level.

1.6.3. Block Coding

- Block coding adds redundancy to line coding so that error detection can be implemented.
- Block coding changes a block of *m* bits into a block of *n* bits, where *n* is larger than *m.*
- **Block coding is referred to as an *mB/nB* encoding technique.**
- The additional bits added to the original “m bits” are called parity **bits** or **check bits**



m : message bits

Figure : Block Coding

Example: 4B/5B encoding

Here a 4 bit code is converted into a 5 bit code

Further Reading & References

- Data Communication & Networking – Behrouz Forouzan.

❖❖❖❖

2

MODULATION

Unit Structure

2.1. Introduction
2.2. Modulation (Analog to digital signal conversion)
2.2.1. PAM
2.2.2. PCM
2.2.3. PWM
2.3. Modulation (Analog data to analog signal conversion)
2.3.1. AM
2.3.2. FM
2.3.3. PM
2.4. Digital Modulation (Digital Data to Digital Signal conversion)
2.4.1. ASK
2.4.2. FSK
2.4.3. PSK
2.4.4. QAM
Further Reading & References

2.1. INTRODUCTION

The previous chapter described encoding techniques to convert digital data to digital signal, now we look at the modulation techniques to convert

- Analog data to digital signals.
- Analog data to analog signal.
- Digital Data to Analog Signal

2.2. MODULATION (ANALOG DATA TO DIGITAL SIGNALS)

The definition of the term modulation is described in the next section. Here we discuss 3 modulation techniques:

1. PAM
2. PCM
3. PWM

2.2.1. PAM (Pulse Amplitude Modulation)

- Pulse Amplitude Modulation refers to a method of carrying information on a train of pulses, the information being encoded in the amplitude of the pulses.

2.2.2. PCM (Pulse Code Modulation)

- PCM is a general scheme for transmitting analog data in a digital and binary way, independent of the complexity of the analog waveform. With PCM all forms of analog data like video, voice, music and telemetry can be transferred.
- To obtain PCM from an analog waveform at the source (transmitter), the analog signal amplitude is sampled at regular time intervals. The sampling rate (number of samples per second), is several times the maximum frequency of the analog waveform. The amplitude of the analog signal at each sample is rounded off to the nearest binary level (quantization).
- The number of levels is always a power of 2 (4, 8, 16, 32, 64,). These numbers can be represented by two, three, four, five, six or more binary digits (bits) respectively.
- At the destination (receiver), a pulse code demodulator converts the binary numbers back into pulses having the same quantum levels as those in the modulator. These pulses are further processed to restore the original analog waveform.

2.2.3. PWM (Pulse Width Modulation)

- Pulse Width Modulation refers to a method of carrying information on a train of pulses, the information being encoded in the width of the pulses. In applications to motion control, it is not exactly information we are encoding, but a method of controlling power in motors without (significant) loss.
- There are several schemes to accomplish this technique. One is to switch voltage on and off, and let the current recirculate through diodes when the transistors have switched off. Another technique is to switch voltage polarity back and forth with a full-bridge switch arrangement, with 4 transistors.
- This technique may have better linearity, since it can go right down to an cycles, and may jitter between minimum duty cycles of positive and negative polarity.

- In battery systems PWM is the most effective way to achieve a constant voltage for battery charging by switching the system controller's power devices on and off. The generation of exact working PWM circuitry is complicated, but it is extremely conceptually important since there is good reason to believe that neurons transmit information using PWM spike trains.

2.3. ANALOG DATA TO ANALOG SIGNAL

Modulation

- The Process of converting analog data to analog signal is called Modulation.
- Modulation is used to send an information bearing signal over long distances.
- Modulation is the process of varying some characteristic of a periodic wave with an external signal called carrier signal.
- These carrier signals are high frequency signals and can be transmitted over the air easily and are capable of traveling long distances.
- The characteristics (amplitude, frequency, or phase) of the carrier signal are varied in accordance with the information bearing signal(analog data).
- The information bearing signal is also known as the modulating signal.
- The modulating signal is a slowly varying – as opposed to the rapidly varying carrier frequency.
- **Types of Modulation:**

  Signal modulation can be divided into two broad categories:

  - Analog modulation and
  - Digital modulation.

  - **Analog or digital** refers to how the data is modulated onto a sine wave.
  - If analog audio data is modulated onto a carrier sine wave, then this is referred to as **analog modulation**.
  - **Digital modulation** is used to convert digital data to analog signal. Ex ASK, FSK, PSK.

- Analog Modulation can be accomplished in three ways:
  1. Amplitude modulation (AM)
  2. Frequency modulation (FM)
  3. Phase modulation (PM).

2.3.1. Amplitude modulation (AM)

- Amplitude modulation is a type of modulation where the amplitude of the carrier signal is varied in accordance with modulating signal.
- The envelope, or boundary, of the amplitude modulated signal embeds modulating signal.
- Amplitude Modulation is abbreviated *AM*.

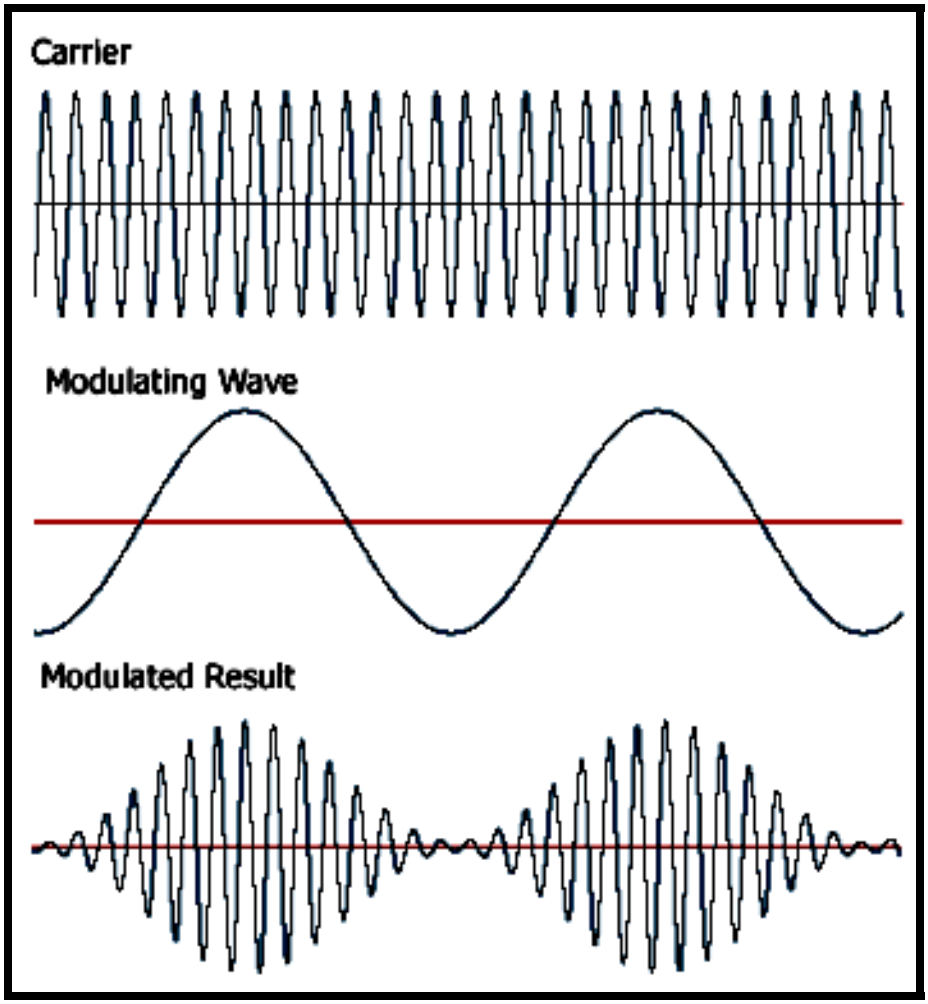


Figure : Amplitude modulation (AM)

2.3.2. Frequency modulation (FM)

- Frequency modulation is a type of modulation where the frequency of the carrier is varied in accordance with the modulating signal. The amplitude of the carrier remains constant.
- The information-bearing signal (the modulating signal) changes the instantaneous frequency of the carrier. Since the amplitude is kept constant, FM modulation is a low-noise process and provides a high quality modulation technique which is used for music and speech in hi-fidelity broadcasts.
- Frequency Modulation is abbreviated *FM*.

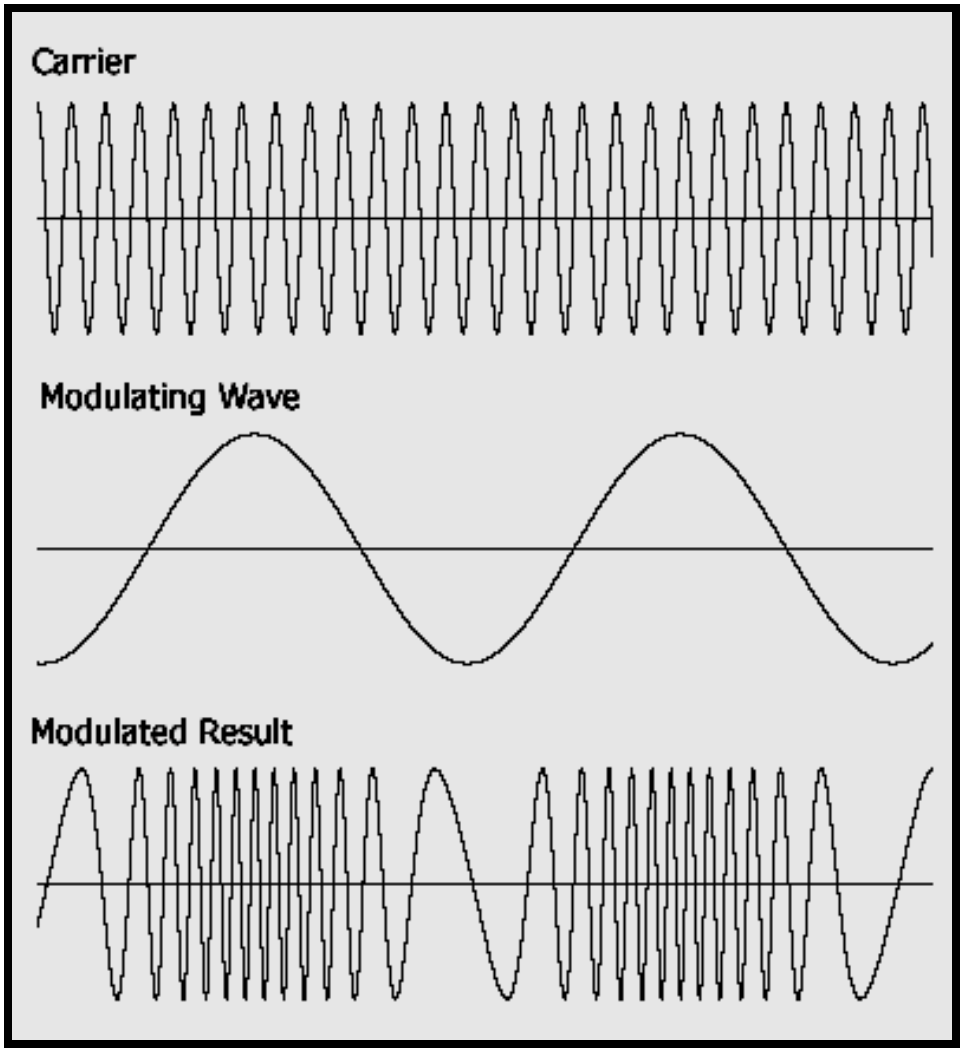


Figure : Frequency modulation (FM)

2.3.3. Phase modulation (PM).

- In phase modulation, the instantaneous phase of a carrier wave is varied from its reference value by an amount proportional to the instantaneous amplitude of the modulating signal.
- Phase Modulation is abbreviated *PM*.

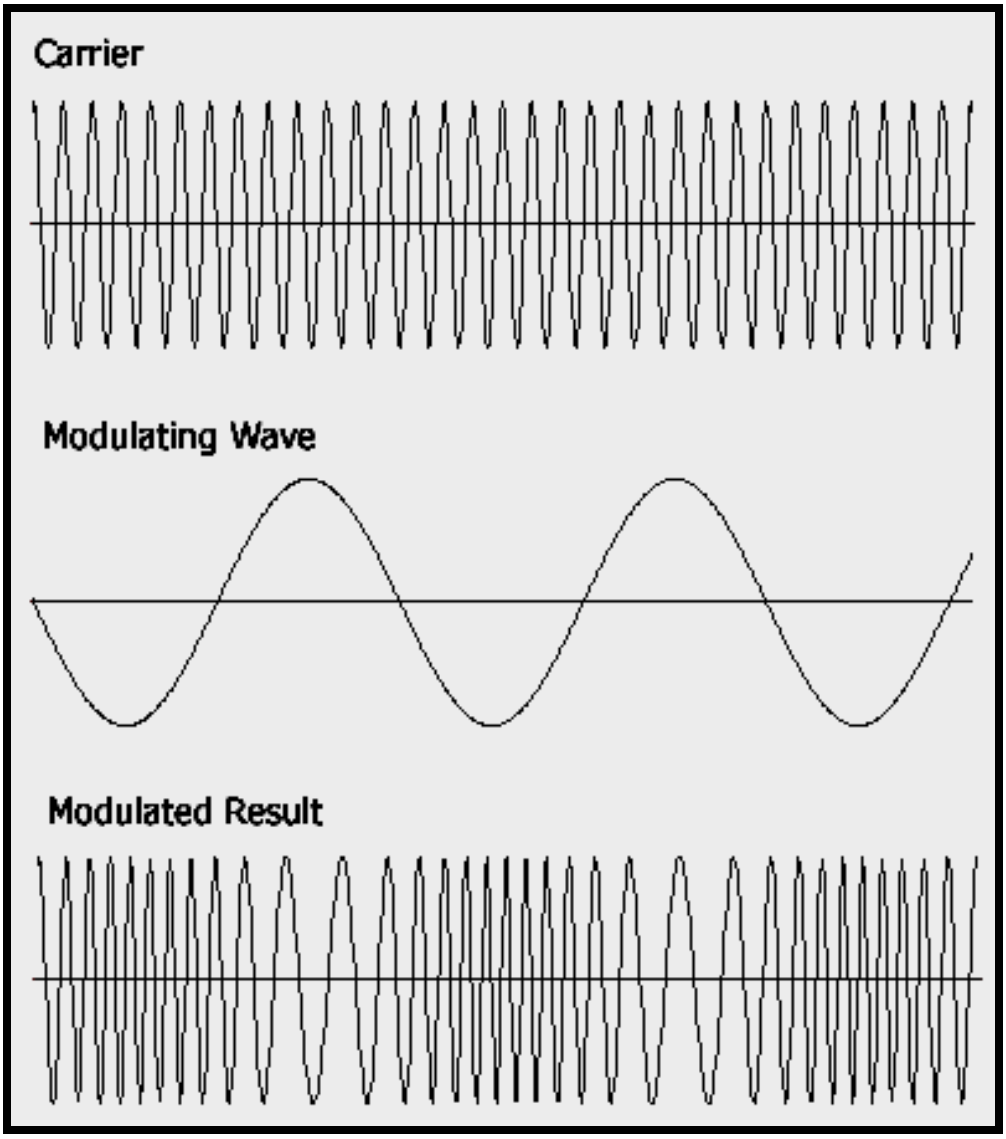


Figure : Phase modulation (PM).

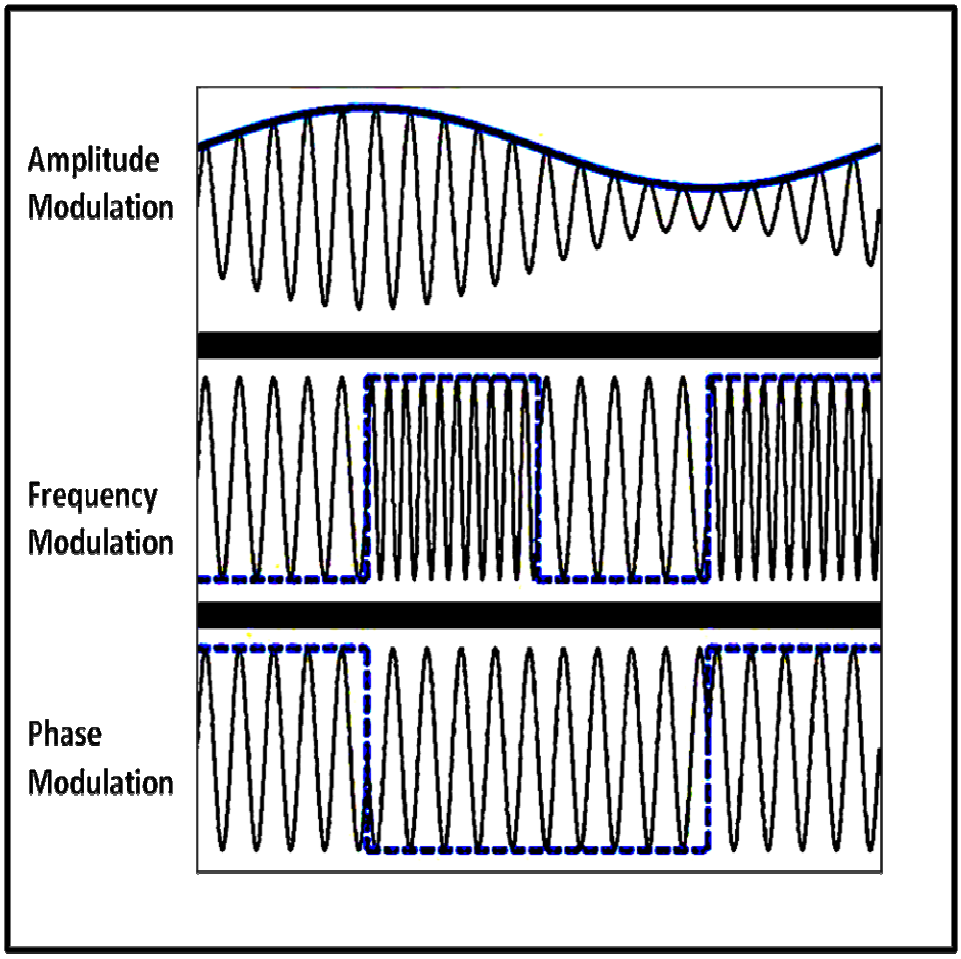


Figure : Comparison of AM, FM & PM

2.4. DIGITAL DATA TO ANALOG SIGNAL

- **Digital modulation** is used to convert digital data to analog signal.
- **It can be accomplished in the following ways:**

  1. ASK
  2. FSK
  3. PSK
  4. QAM

2.4.1. Amplitude Shift Keying (ASK)

- In amplitude shift keying, the amplitude of the carrier signal is varied to create signal elements.
- Both frequency and phase remain constant while the amplitude changes.

- **Binary ASK (BASK)**

  ASK is normally implemented using only two levels and is hence called binary amplitude shift keying.

  Bit 1 is transmitted by a carrier of one particular amplitude.

  To transmit Bit 0 we change the amplitude keeping the frequency is kept constant

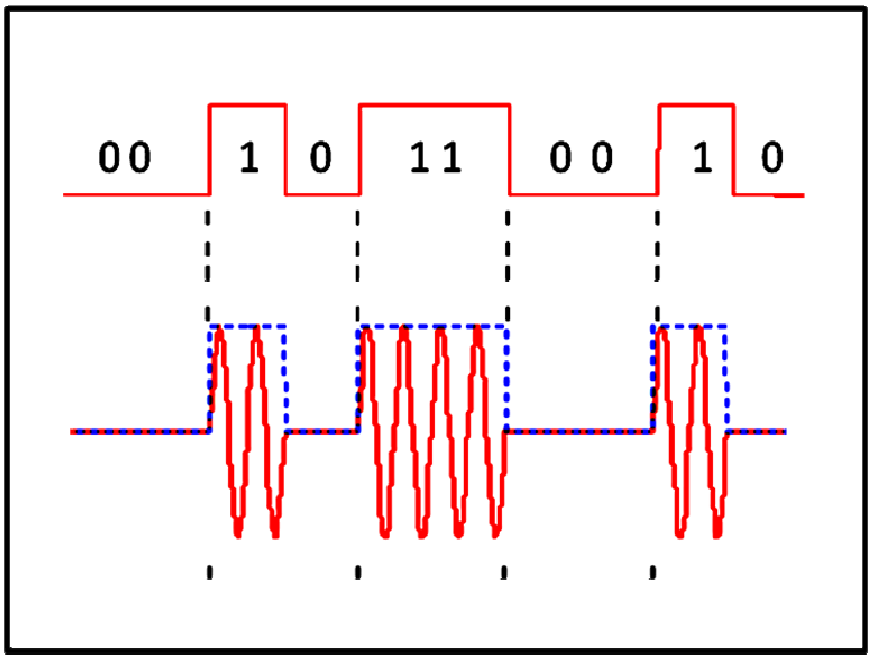


Figure : Amplitude Shift Keying (ASK

2.4.2. Frequency Shift Keying (FSK)

- In Frequency shift keying, we change the frequency of the carrier wave.
- Bit 0 is represented by a specific frequency, and bit 1 is represented by a different frequency.
- In the figure below frequency used for bit 1 is higher than frequency used for bit 0

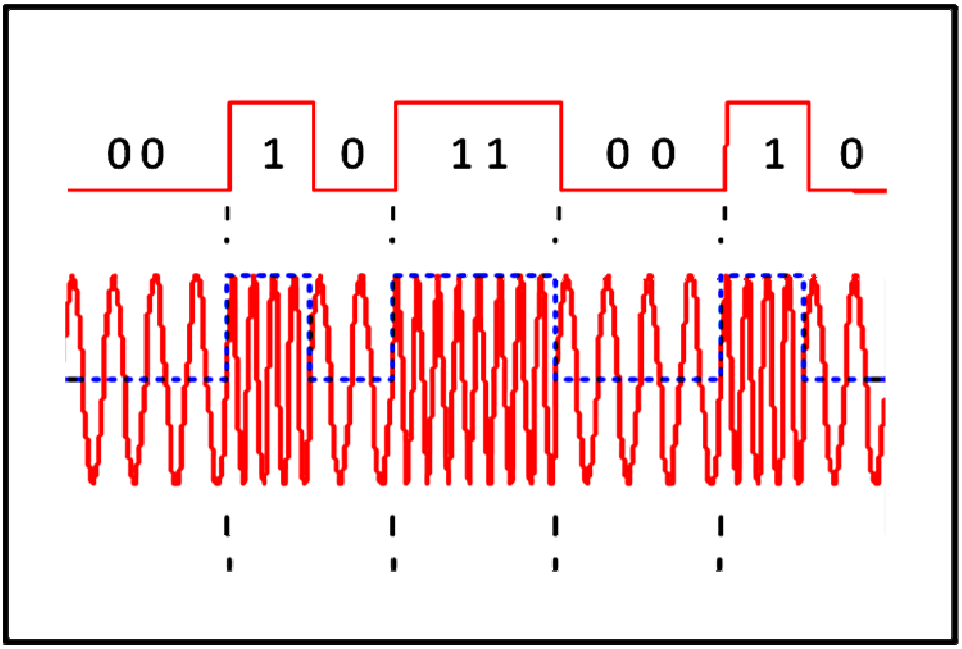


Figure : Frequency Shift Keying (FSK)

2.4.3. Phase Shift Keying (PSK)

- Phase shift keying (PSK) is a method of transmitting and receiving digital signals in which the phase of a transmitted signal is varied to convey information.
- Both amplitude and frequency remain constant as the phase changes.
- The simplest from of PSK has only two phases, 0 and 1.
- If the phase of the wave does not change, then the signal state stays the same (low or high).
- If the phase of the wave changes by 180 degrees, that is, if the phase reverses, then the signal state changes (from low to high or from high to low)

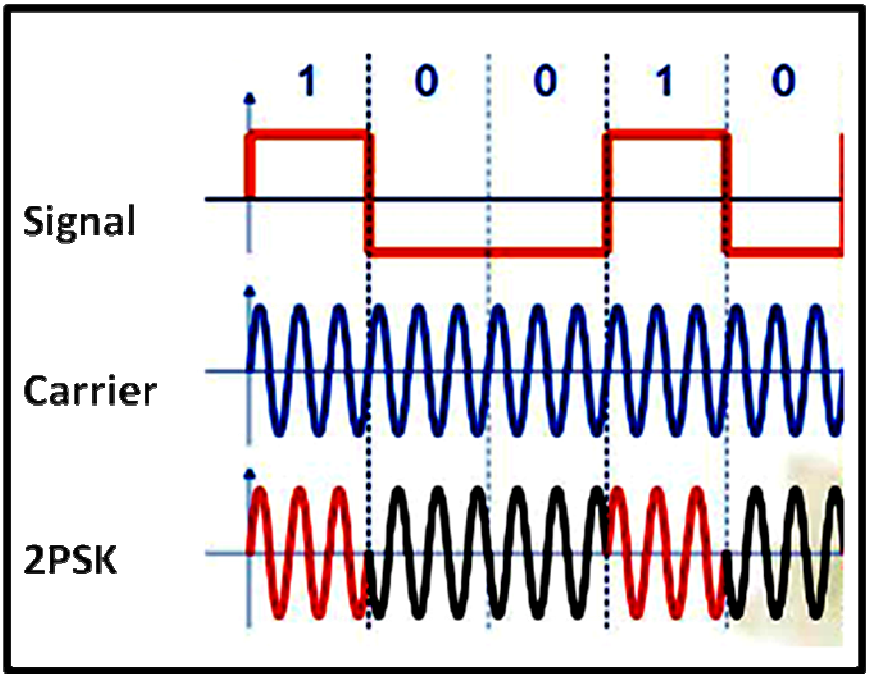


Figure: Phase Shift Keying (PSK)

2.4.4. QAM

- The concept of Quadrature Amplitude Modulation (QAM) involves use of two carriers, one for phase and the other for quadrature, with different amplitude levels for each carrier.
- It is a combination of ASK & PSK.

Further Reading & References

- Data Communication & Networking – Behrouz Forouzan.

❖❖❖❖

3

MULTIPLEXING

Unit Structure

3.1. Multiplexing
- 3.1.1. FDM
- 3.1.2. TDM
- 3.1.3. WDM

3.2. Modes of communication
- 3.2.1. Simplex
- 3.2.2. Half Duplex
- 3.2.3. Full Duplex

3.3. Switching Techniques
- 3.3.1. Circuit switching
- 3.3.2. Packet switching
- 3.3.2.1. Datagram Approach
- 3.3.2.2. Virtual Circuit Approach

Further Reading & References

3.1. MULTIPLEXING

- The concept of Multiplexing is closely related to bandwidth. Communication links have a definite bandwidth. A signal may require a bandwidth that is greater than the communication link which may require combining the several links together or the signal may require a very less portion of bandwidth which results in its wastage. It would be wise to share this link with other devices, which is done using Multiplexing techniques.

- **Multiplexing is a set of techniques. It allows simultaneous transmission of multiple signals across a single communication link.**

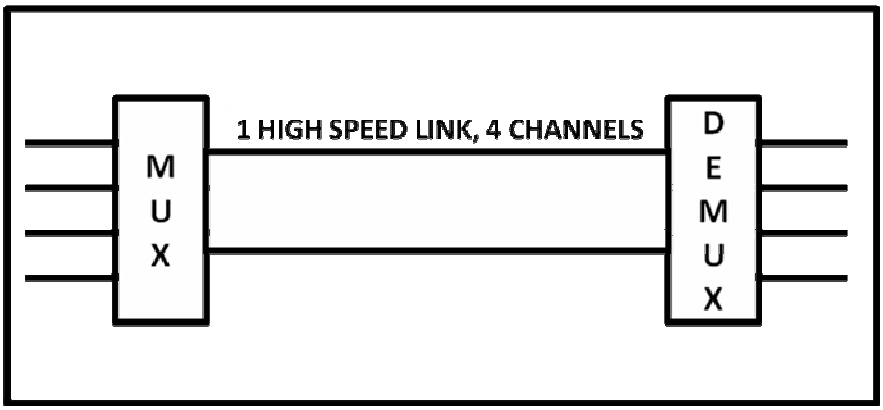


Fig. Multiplexing

- In the figure above, 4 low speed links are connected to a Multiplexer (MUX) which combines them into a single high speed link.
- A Multiplexer performs a many to one signal conversion (many low speed signals are converted into one high speed signal).
- This signal is fed to the Demultiplexer (DEMUX) at the other end which separates the high speed signal into its original component signals.
- Also, a channel is portion of a link, 1 link may have multiple (n) channels.
- Signals are of two types : Analog and Digital so multiplexing techniques fall under two categories A. Analog & B. Digital

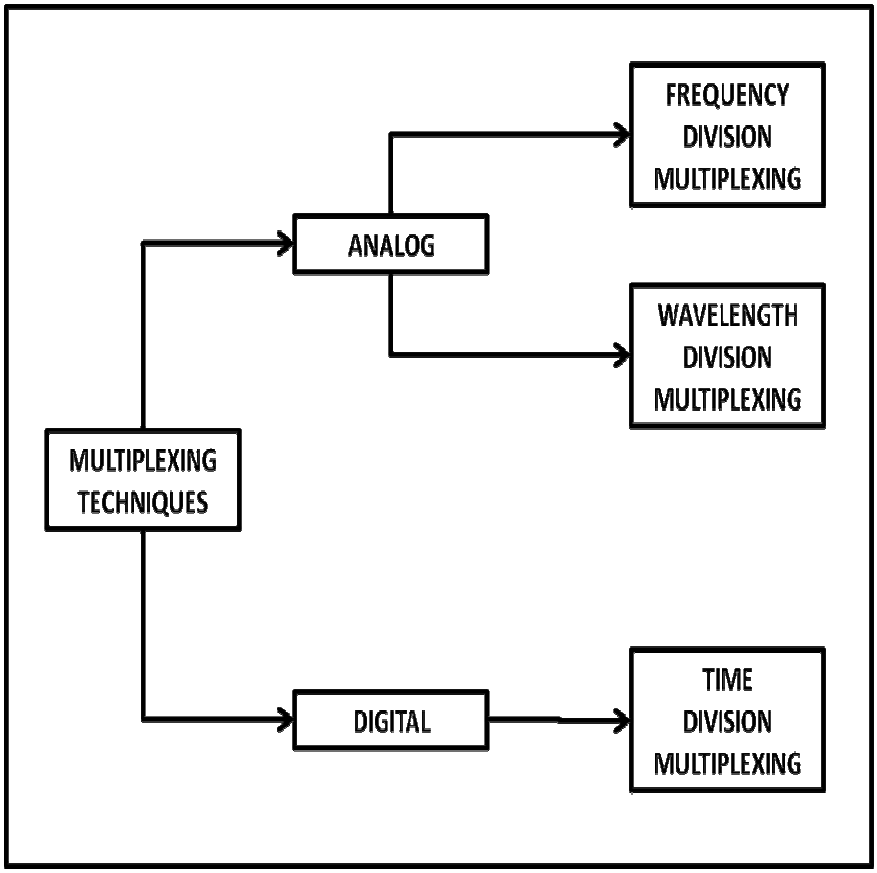


Fig: Types of Multiplexing

3.1.1. FDM (Frequency Division Multiplexing)

- FDM is applicable when the bandwidth of a link (in hertz) is more than the combined bandwidth of individual signals to be transmitted
- In FDM, signals generated by each sending device modulate different carrier frequencies. These modulated signals are then combined into a single composite signal that can be transported by the link.
- Carrier frequencies are separated by sufficient bandwidth to accommodate the modulated signal.
- Signals are prevented from overlapping by using strips of unused bandwidth called guard bands.

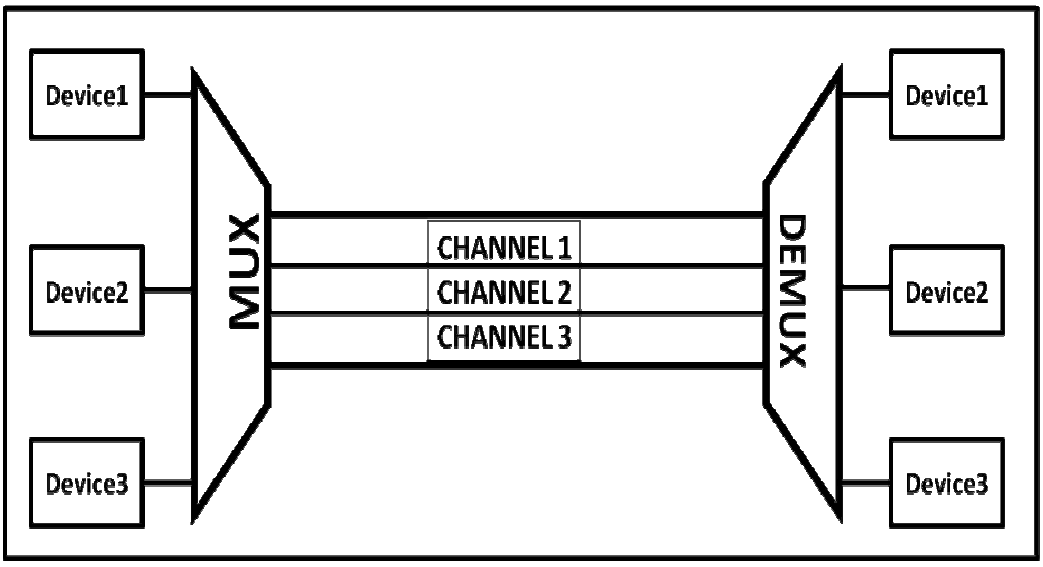


Figure : FDM (Frequency Division Multiplexing)

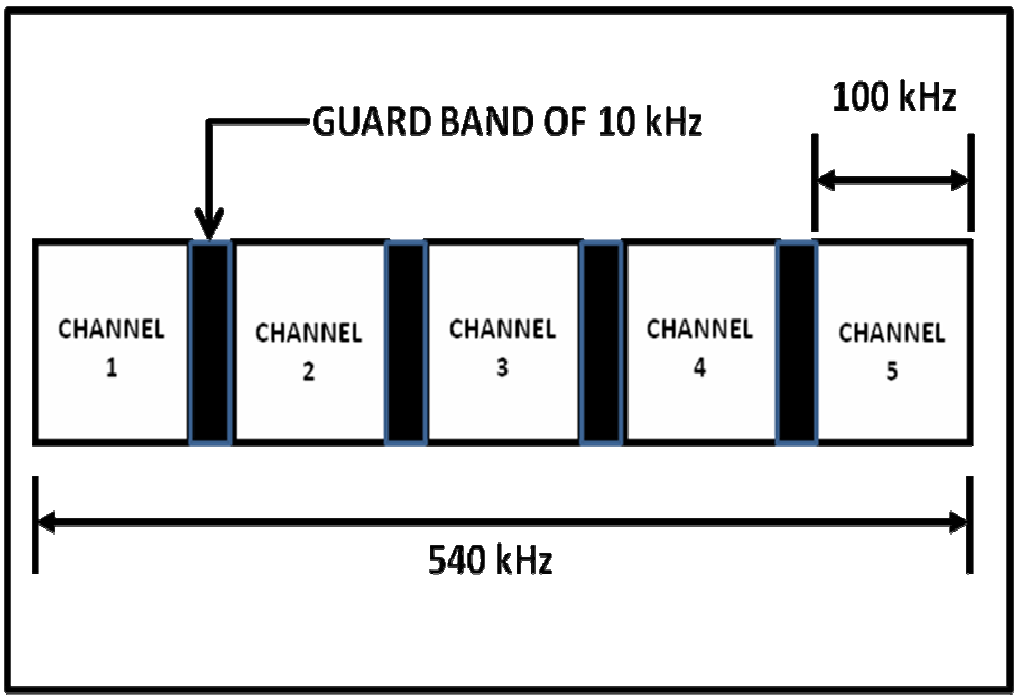


Figure: Guard Band Frequency in FDM

3.1.2. WDM (Wavelength Division Multiplexing)

- Wavelength-division multiplexing (WDM) is designed to use the high-data-rate capability of fiber-optic cable. The optical fiber data rate is higher than the data rate of metallic transmission cable. Using a fiber-optic cable for one single line wastes the available bandwidth.
- Multiplexing allows us to combine several lines into one.

- WDM is conceptually the same as FDM, except that the multiplexing and demultiplexing involve optical signals transmitted through fiber-optic channels.
- The idea is the same: We are combining different signals of different frequencies. The difference is that the frequencies are very high.

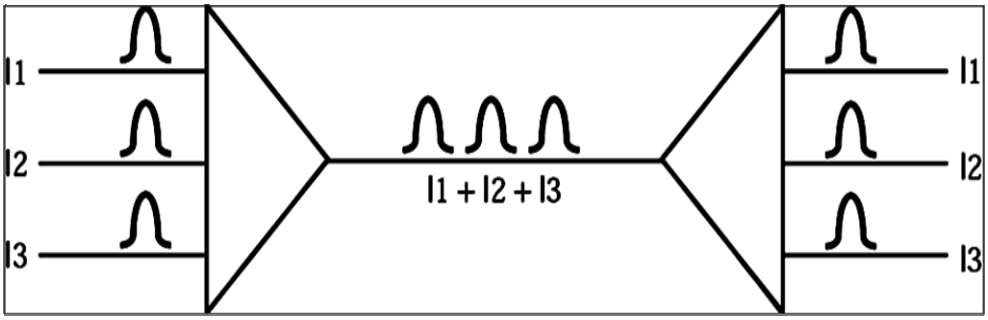


Figure : WDM (Wavelength Division Multiplexing)

- Figure above gives a conceptual view of a WDM multiplexer and demultiplexer.
- Very narrow bands of light from different sources are combined to make a wider band of light. At the receiver, the signals are separated by the demultiplexer.

3.1.3. TDM (Time-division multiplexing)

- It is a digital process that allows several connections to share the high bandwidth of a link.
- Instead of sharing a portion of the bandwidth as in FDM, time is shared. Each connection occupies a portion of time in the link.
- The next Figure gives a conceptual view of TDM.
- The same link is used as in FDM; here, however, the link is shown sectioned by time rather than by frequency. In the figure, portions of signals 1,2,3, and 4 occupy the link sequentially.

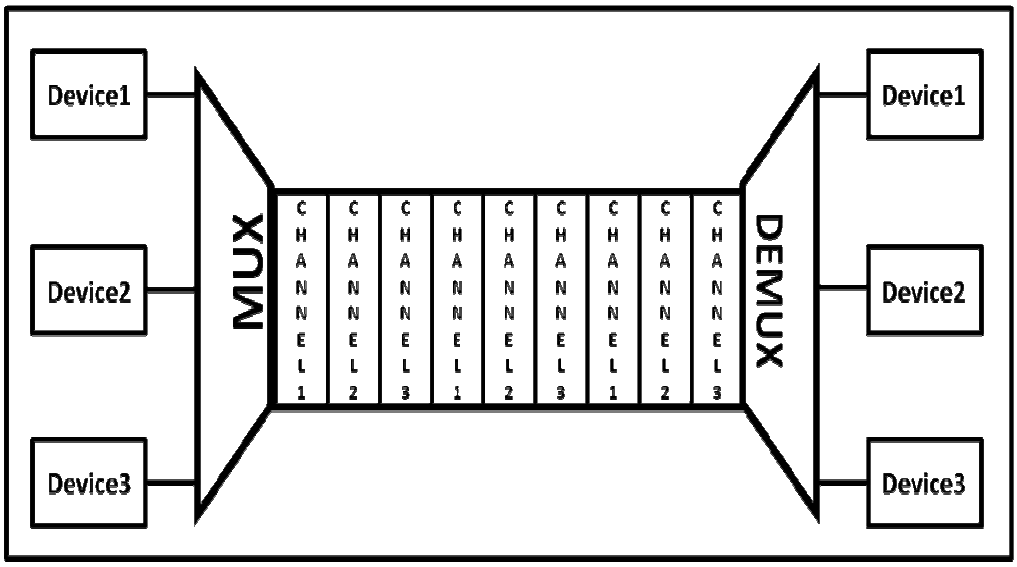


Figure: TDM (Time-division multiplexing)

3.2. MODES OF COMMUNICATION

Two devices communicate with each other by sending and receiving data. This can be done in the following ways.

3.2.1. Simplex

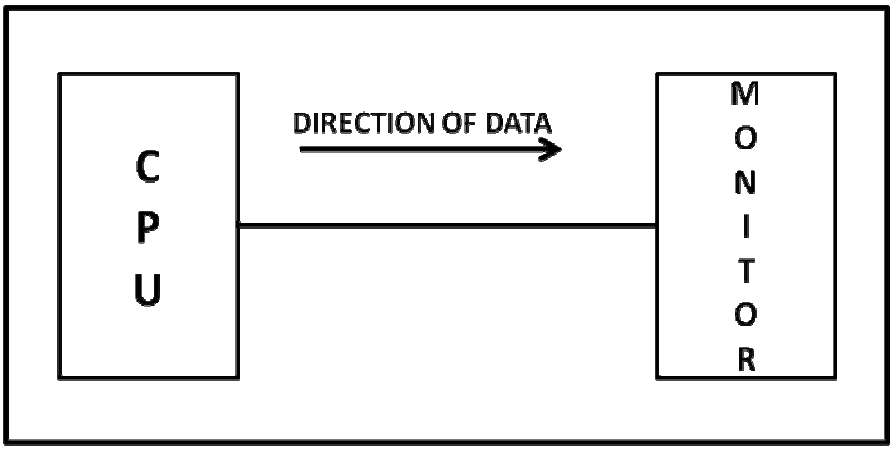


Figure: Simplex mode of communication

- In Simplex, communication is unidirectional
- One of the devices only sends the data and the other one only receives the data.
- Example: in the above diagram: a cpu send data while a monitor only receives data.

3.2.2. Half Duplex

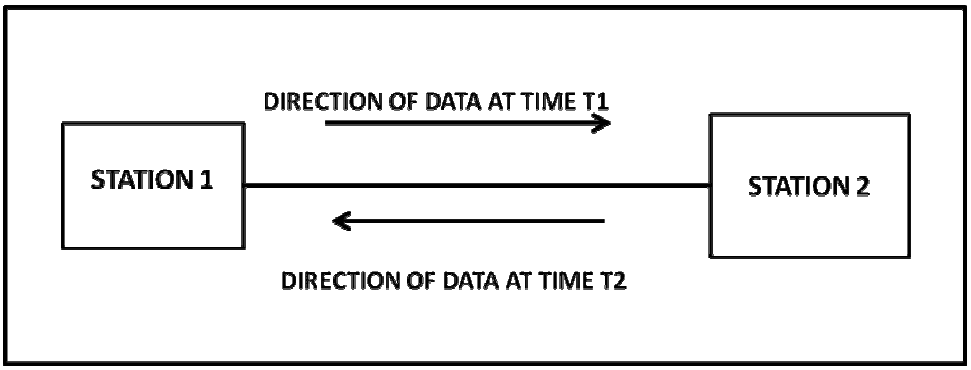


Figure: Half Duplex Mode of Communication

- In half duplex both the stations can transmit as well as receive but not at the same time.
- When one device is sending other can only receive and vice-versa (as shown in figure above.)
- Example: A walkie-talkie.

3.2.3. Full Duplex

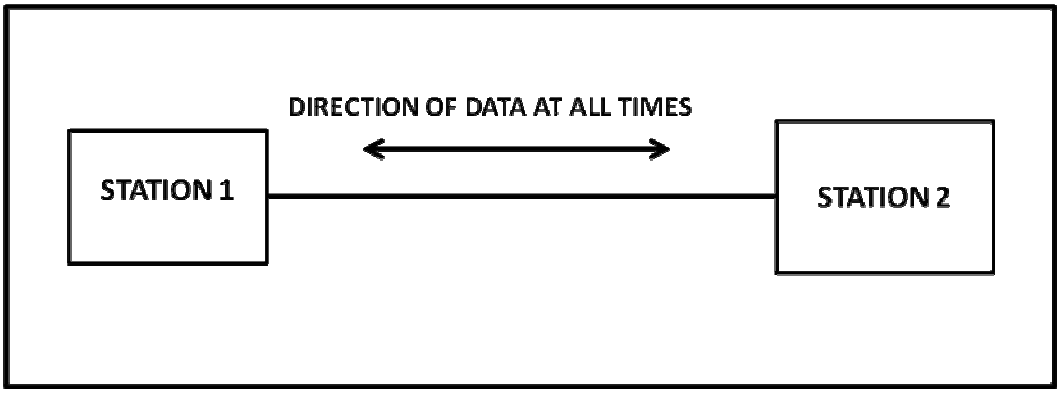
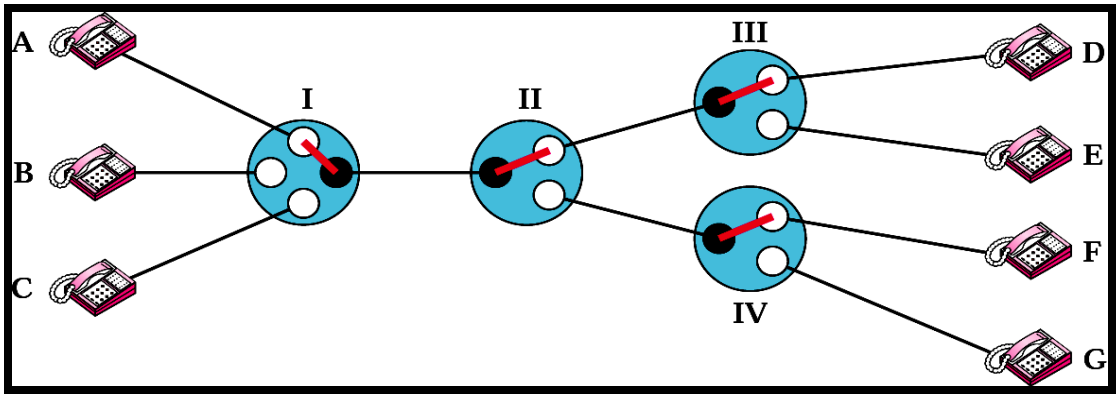


Figure: Full Duplex Mode of Communication

- In Full duplex mode, both stations can transmit and receive at the same time.
- Example: mobile phones

3.3. SWITCHING

- Switching is a solution or an alternate to using meshed network. A network is a collection of connected nodes. When we have multiple devices, there's the question of how to connect them, the best way is to connect a device to every other device to form a mesh topology, but that would result in a waste of resources.
- A Switch is a device capable of creating temporary connections between two or more devices linked to the switch.
- A Switched network consists of a collection of interlinked nodes called switches.
- In a switched network some of the nodes are connected to the end devices while others are only used for routing.



- In the above figure the Switches are labeled I, II, III, IV and the end nodes are labeled as A, B, C, D, E, F, G. Switch I is connected to end nodes A, B, C and switch III is connected to end node D and E while Switch II is only used for routing.
- There are three types of Switching techniques :
    1. Circuit Switching
    2. Packet Switching
    3. Message Switching.

  Message Switching is no longer used so our discussion is limited to circuit and packet switching.

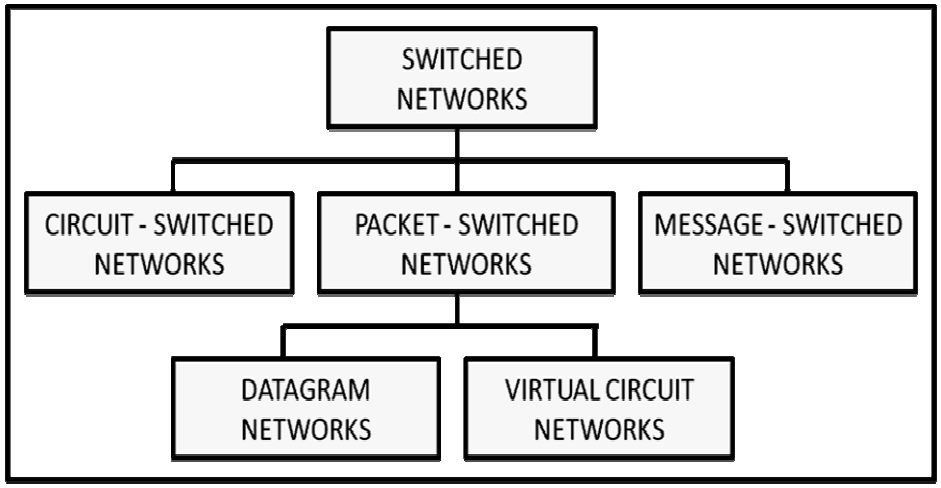


Figure Taxonomy of switched networks

3.3.1. CIRCUIT-SWITCHED NETWORKS

- Circuit switching takes place at the physical layer
- A circuit switching network is one that establishes a circuit (or channel) between nodes and terminals before the users may communicate, as if the nodes were physically connected with an electrical circuit.
- An important property of circuit switching is the need to set up an end-to-end path before any data can be sent.
- Example: A telephone call
- The switching equipment within the telephone system seeks out a physical path all the way from sender's telephone to the receiver's telephone.
- The actual communication in a circuit-switched network requires three phases:
    i. **connection setup**
    ii. **data transfer, and**
    iii. **connection teardown**.

Connection Setup

- Before the two parties (or multiple parties in a conference call) can communicate, a dedicated circuit (combination of channels in links) needs to be established.
- The end systems are normally connected through dedicated lines to the switches, so connection setup means creating dedicated channels between the switches.

Data Transfer Phase

- After the establishment of the dedicated circuit (channels), the two parties can transfer data.

Teardown Phase

- When one of the parties needs to disconnect, a signal is sent to each switch to release the resources.

3.3.2. PACKET SWITCHED NETWORKS

- In packet switching network, there is no call setup hence no particular path to be followed by the packets i.e different packets can follow different paths depending upon the network conditions, so the packets may arrive out of order at the receiver.
- Packet Switching is more fault tolerant than circuit switching. If a switch goes down, all of the circuits using it are terminated and no more traffic can be sent on any of them. With packet switching packets can be routed around dead switches.
- Packet switching uses store and forward transmission. A packet is stored in the routers memory and then sent on to the next router. With circuit switching bits just flow through the wire continuously. The store and forward technique adds delay to packet switching networks.
- Packet Switching network differ from circuit switching network in the way in which users are charged.
- With Circuit switching charging is done based on distance and time.

  Ex. STD calls are charged more compared to local calls.
- Packet switching networks charge for the volume of traffic.

  Ex. MTNL allows 400Mb of internet browsing at Rs 200/- After this limit of 400Mb is exceeded every MB is charged explicitly.
- In a packet-switched network, there is no resource reservation; resources are allocated on demand

There are two approaches to packet switching

- Datagram Approach
- Virtual Circuit Approach

3.3.2.1. Datagram Switching

- Using Datagram transmission, each packet is treated as a separate entity and contains a header with the full information about the intended recipient.

- The intermediate nodes examine the header of a packet and select an appropriate link to an intermediate node which is nearer the destination.

- In this system packets do not follow a pre-established route, and the intermediate nodes (usually routers) do not require prior knowledge of the routes that will be used.

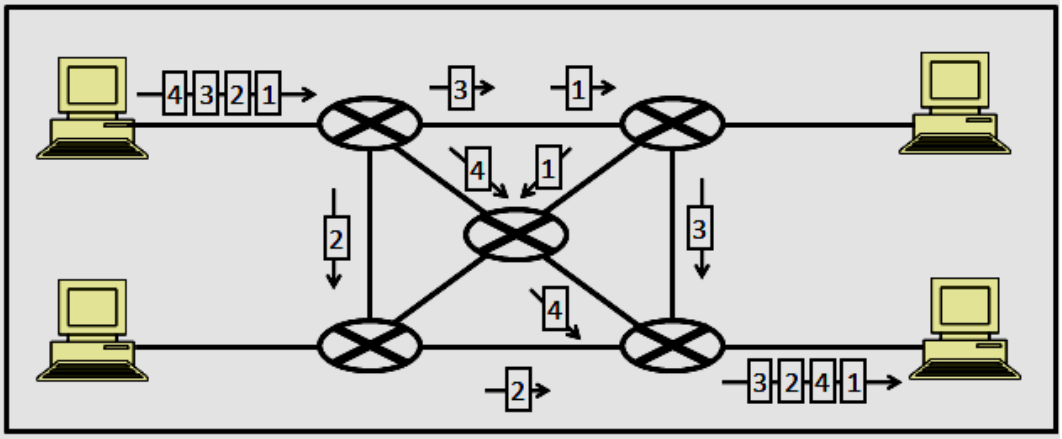


Figure: A Datagram Network

- Datagram switching is normally done at the network layer
- In a datagram network as packets are treated as independent entities they may take different routes to reach the destination hence may arrive out of sequence at the destination.
- The datagram networks are sometimes referred to as connectionless networks.
- The term ***connectionless*** here means that the switch (packet switch) does not keep information about the connection state. There are no setup or teardown phases. Each packet is treated the same by a switch regardless of its source or destination.
- **Routing Table**
- Each switch (or packet switch) has a routing table which is based on the destination address. The routing tables are dynamic and are updated periodically.
- The destination addresses and the corresponding forwarding output ports are recorded in the tables.
- This is different from the table of a circuit switched network in which each entry is created when the setup phase is completed and deleted when the teardown phase is over.

- **Destination Address**
  - Every packet in a datagram network carries a header that contains, among other information, the destination address of the packet.
  - When the switch receives the packet, this destination address is examined; the routing table is consulted to find the corresponding port through which the packet should be forwarded.

3.3.2.2. Virtual-circuit networks

A virtual-circuit network is a cross between a circuit-switched network and a datagram network. It has some characteristics of both.

1. As in a circuit-switched network, there are setup and teardown phases in addition to the data transfer phase.
2. Resources can be allocated during the setup phase, as in a circuit-switched network, or on demand, as in a datagram network.
3. As in a datagram network, data are packetized and each packet carries an address in the header.
4. As in a circuit-switched network, all packets follow the same path established during the connection.
5. A virtual-circuit network is normally implemented in the data link layer, while a circuit-switched network is implemented in the physical layer and a datagram network in the network layer.

Addressing

- In a virtual-circuit network, two types of addressing are involved:
  Global and local (virtual-circuit identifier).

Global Addressing

- A source or a destination needs to have a global address-an address that can be unique in the scope of the network or internationally if the network is part of an international network.

Virtual-Circuit Identifier

- The identifier that is actually used for data transfer is called the virtual-circuit identifier (VCI).
- A VCI, unlike a global address, is a small number that has only switch scope; it is used by a frame between two

switches. When a frame arrives at a switch, it has a VCI; when it leaves, it has a different VCI.

Three Phases

A source and destination need to go through three phases in a virtual-circuit network:

1. setup,
2. data transfer, and
3. teardown

Setup Phase

In the setup phase, a switch creates an entry for a virtual circuit.

For example, suppose source A needs to create a virtual circuit to B. Two steps are required:

1. the setup request and
2. the acknowledgment.

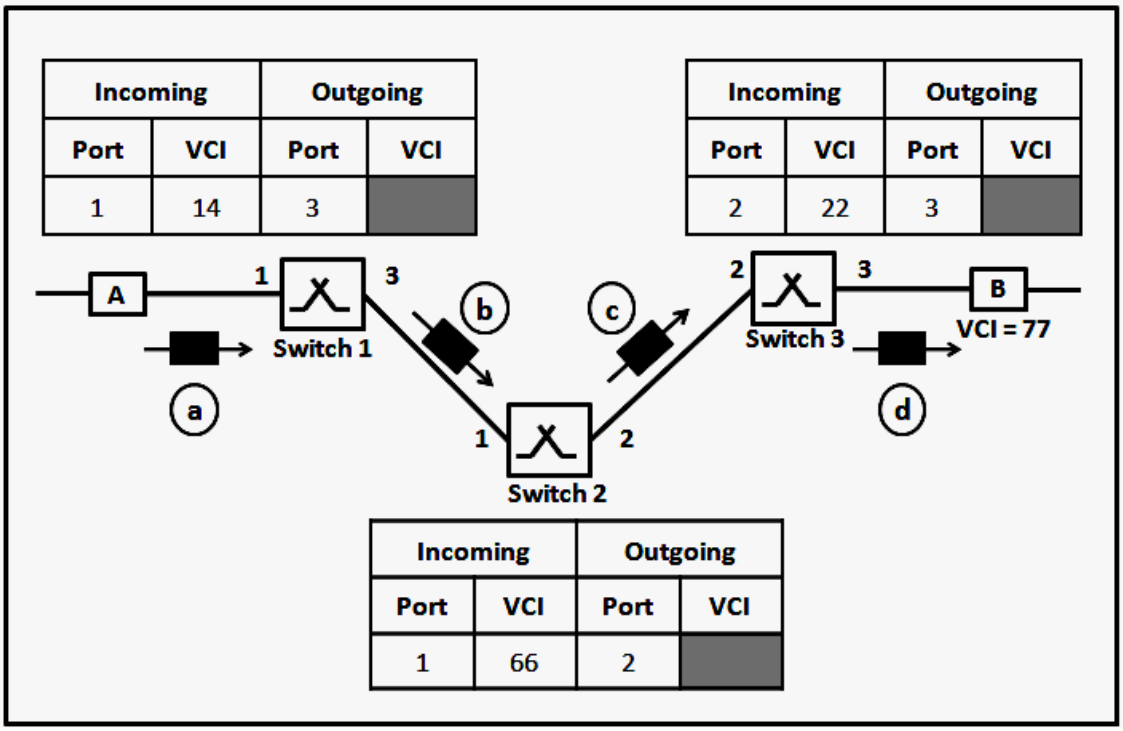


Figure: Setup Request

Setup Request

A setup request frame is sent from the source to the destination. Figure above shows the process.

a. Source A sends a setup frame to switch 1.

b. Switch 1 receives the setup request frame. It knows that a frame going from A to B goes out through port 3.

- It has a routing table which is different from the switching table
- The switch creates an entry in its table for this virtual circuit, but it is only able to fill three of the four columns.
- The switch assigns the incoming port (1) and chooses an available incoming VCI (14) and the outgoing port (3).
- It does not yet know the outgoing VCI, which will be found during the acknowledgment step. The switch then forwards the frame through port 3 to switch 2.

c. Switch 2 receives the setup request frame. The same events happen here as at switch 1; three columns of the table are completed: in this case, incoming port (I), incoming VCI (66), and outgoing port (2).

d. Switch 3 receives the setup request frame. Again, three columns are completed: incoming port (2), incoming VCI (22), and outgoing port (3).

e. Destination B receives the setup frame, and if it is ready to receive frames from A, it assigns a VCI to the incoming frames that come from A, in this case 77.

This VCI lets the destination know that the frames come from A, and not other sources.

Acknowledgment

A special frame, called the acknowledgment frame, completes the entries in the switching tables Figure below shows the process

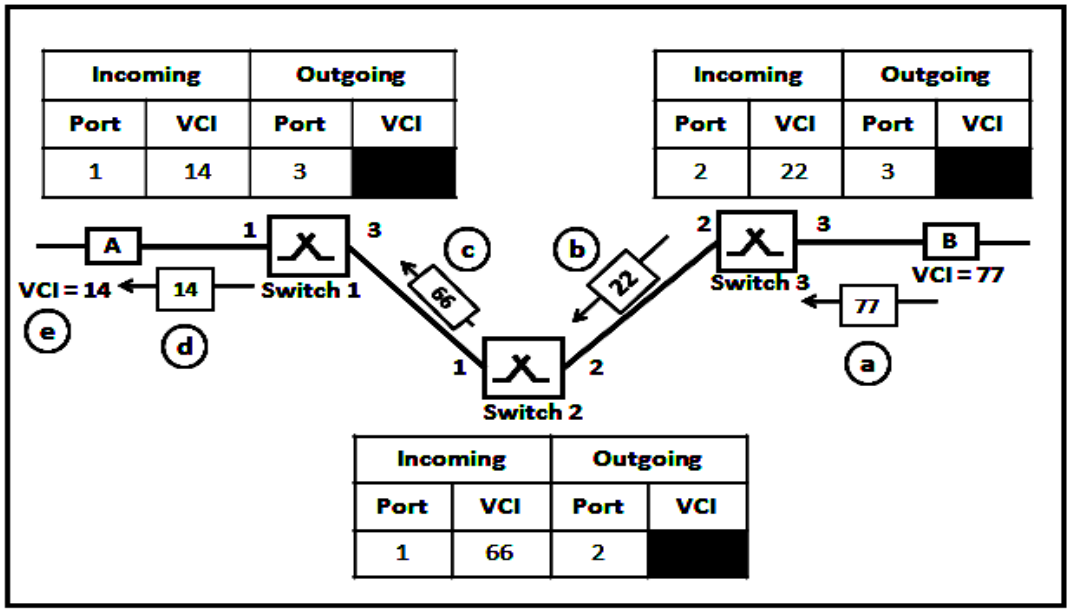


Figure: Setup Acknowledgement

a. The destination sends an acknowledgment to switch 3.
   - The acknowledgment carries the global source and destination addresses so the switch knows which entry in the table is to be completed.
   - The frame also carries VCI 77, chosen by the destination as the incoming VCI for frames from A.
   - Switch 3 uses this VCI to complete the outgoing VCI column for this entry.
   - 77 is the incoming VCI for destination B, but the outgoing VCI for switch 3.

b. Switch 3 sends an acknowledgment to switch 2 that contains its incoming VCI in the table, chosen in the previous step. Switch 2 uses this as the outgoing VCI in the table.

c. Switch 2 sends an acknowledgment to switch 1 that contains its incoming VCI in the table, chosen in the previous step. Switch 1 uses this as the outgoing VCI in the table.

d. Finally switch 1 sends an acknowledgment to source A that contains its incoming VCI in the table, chosen in the previous step.

e. The source uses this as the outgoing VCI for the data frames to be sent to destination B.

Data Transfer Phase

To transfer a frame from a source to its destination, all switches need to have a table entry for this virtual circuit.

The table has four columns.

INCOMING		OUTGOING	
PORT	VCI	PORT	VCI
1	14	3	22
1	77	2	41

Teardown Phase

- In this phase, source A, after sending all frames to B, sends a special frame called a ***teardown request***.
- Destination B responds with a teardown confirmation frame. All switches delete the corresponding entry from their tables.

Further Reading & References

- Data Communication & Networking – Behrouz Forouzan.

UNIT 2
Introductions

4

INTRODUCTION TO COMPUTER NETWORKS

Unit Structure

4.1 Uses of computer network
- 4.1.1 Introduction
- 4.1.2 Purpose
- 4.1.3 Network Classification
- 4.1.4 Basic Hardware Components

4.2 LANs, WANs, MANs

4.3 Wireless Networks
- 4.3.1 Classification of Wireless networks
- 4.3.2 Uses of Wireless Networks
- 4.3.3 Problems with Wireless Networks

4.4 Internet work
- 4.4.1 VPN
- 4.4.2 Overlay Network

4.1 USES OF COMPUTER NETWORK

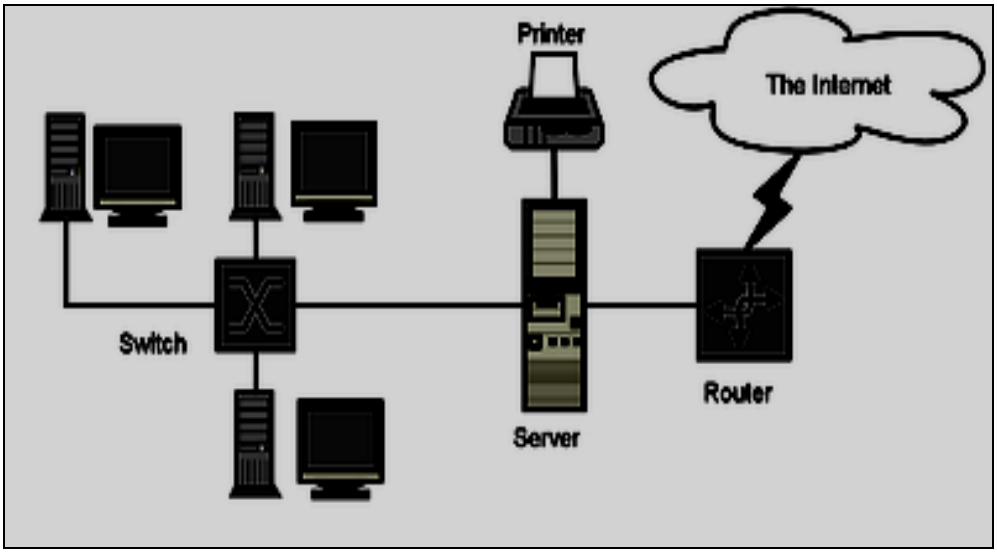
4.1.1 Introduction:

A computer network, often simply referred to as a network, is a group of computers and devices interconnected by communications channels that facilitate communications among users and allows users to share resources and information. In the 1960s, the Advanced Research Projects Agency (ARPA) started funding the design of the Advanced Research Projects Agency Network (ARPANET) for the United States Department of Defense. It was the first computer network in the world. Development of the network began in 1969, based on designs developed during the 1960s.

4.1.2 Purpose:

Computer networks can be used for several purposes:

- **Facilitating communications.** Using a network, people can communicate efficiently and easily via email, instant messaging, chat rooms, telephone, video telephone calls, and video conferencing.
- **Sharing hardware**. In a networked environment, each computer on a network may access and use hardware resources on the network, such as printing a document on a shared network printer.
- **Sharing files, data, and information.** In a network environment, authorized user may access data and information stored on other computers on the network. The capability of providing access to data and information on shared storage devices is an important feature of many networks.
- **Sharing software.** Users connected to a network may run application programs on remote computers.



4.1.3 Network classification:

The following list presents categories used for classifying networks.

1. **Connection method**
   Computer networks can be classified according to the hardware and software technology that is used to interconnect the individual devices in the network, such as optical fiber, Ethernet, wireless LAN, HomePNA, power line communication or G.hn.
   Ethernet uses physical wiring to connect devices. Frequently deployed devices include hubs, switches, bridges, or routers. Wireless LAN technology is designed to connect devices without wiring. These devices use radio

waves or infrared signals as a transmission medium. ITU-T G.hn technology uses existing home wiring (coaxial cable, phone lines and power lines) to create a high-speed (up to 1 Gigabit/s) local area network.

a) Wired technologies

- **Twisted pair wire** is the most widely used medium for telecommunication. Twisted-pair wires are ordinary telephone wires which consist of two insulated copper wires twisted into pairs and are used for both voice and data transmission. The use of two wires twisted together helps to reduce crosstalk and electromagnetic induction. The transmission speed ranges from 2 million bits per second to 100 million bits per second.

- **Coaxial cable** is widely used for cable television systems, office buildings, and other worksites for local area networks. The cables consist of copper or aluminum wire wrapped with insulating layer typically of a flexible material with a high dielectric constant, all of which are surrounded by a conductive layer. The layers of insulation help minimize interference and distortion. Transmission speed range from 200 million to more than 500 million bits per second.

- **Optical fiber cable** consists of one or more filaments of glass fiber wrapped in protective layers. It transmits light which can travel over extended distances. Fiber-optic cables are not affected by electromagnetic radiation. Transmission speed may reach trillions of bits per second. The transmission speed of fiber optics is hundreds of times faster than for coaxial cables and thousands of times faster than a twisted-pair wire.

b) Wireless technologies

- **Terrestrial microwave** – Terrestrial microwaves use Earth-based transmitter and receiver. The equipment looks similar to satellite dishes. Terrestrial microwaves use low-gigahertz range, which limits all communications to line-of-sight. Path between relay stations spaced approx, 30 miles apart. Microwave antennas are usually placed on top of buildings, towers, hills, and mountain peaks.

- **Communications satellites** – The satellites use microwave radio as their telecommunications medium which are not deflected by the Earth's atmosphere. The satellites are stationed in space, typically 22,000 miles (for geosynchronous satellites) above the equator. These Earth-orbiting systems are capable of receiving and relaying voice, data, and TV signals.

- **Cellular and PCS systems** – Use several radio communications technologies. The systems are divided to different geographic areas. Each area has a low-power transmitter or radio relay antenna device to relay calls from one area to the next area.

- **Wireless LANs** – Wireless local area network use a high-frequency radio technology similar to digital cellular and a low-frequency radio technology. Wireless LANs use spread spectrum technology to enable communication between multiple devices in a limited area. An example of open-standards wireless radio-wave technology is IEEE.

- **Infrared communication** , which can transmit signals between devices within small distances not more than 10 meters peer to peer or (face to face) without anybody in the line of transmitting.

2. **Scale:**
   Networks are often classified as
   - local area network (LAN),
   - wide area network (WAN),
   - metropolitan area network (MAN),
   - personal area network (PAN),
   - virtual private network (VPN),
   - campus area network (CAN),
   - storage area network (SAN), and others, depending on their scale, scope and purpose, e.g., controller area network (CAN) usage, trust level, and access right often differ between these types of networks.

LANs tend to be designed for internal use by an organization's internal systems and employees in individual physical locations, such as a building, while WANs may connect physically separate parts of an organization and may include connections to third parties.

3. **Functional relationship (network architecture)**
   Computer networks may be classified according to the functional relationships which exist among the elements of the network, e.g., active networking, client–server and peer-to-peer (workgroup) architecture.

4. **Network topology**
   Computer networks may be classified according to the network topology upon which the network is based, such as bus network, star network, ring network, mesh network. Network topology is the coordination by which devices in the network are arranged in their logical relations to one another,

independent of physical arrangement. Even if networked computers are physically placed in a linear arrangement and are connected to a hub, the network has a star topology, rather than a bus topology. In this regard the visual and operational characteristics of a network are distinct. Networks may be classified based on the method of data used to convey the data; these include digital and analog networks.

4.1.4 Basic hardware components

All networks are made up of basic hardware building blocks to interconnect network nodes, such as Network Interface Cards (NICs), Bridges, Hubs, Switches, and Routers. Less common are microwave links or optical cable ("optical fiber").

Network interface cards:

A network card, network adapter, or NIC (network interface card) is a piece of computer hardware designed to allow computers to communicate over a computer network. It provides physical access to a networking medium and often provides a low-level addressing system through the use of MAC addresses.

Repeaters:

A repeater is an electronic device that receives a signal, cleans it of unnecessary noise, regenerates it, and retransmits it at a higher power level, or to the other side of an obstruction, so that the signal can cover longer distances without degradation. In most twisted pair Ethernet configurations, repeaters are required for cable that runs longer than 100 meters. Repeaters work on the Physical Layer of the OSI model.

Hubs:

A network hub contains multiple ports. When a packet arrives at one port, it is copied unmodified to all ports of the hub for transmission. The destination address in the frame is not changed to a broadcast address. It works on the Physical Layer of the OSI model.

Bridges:

A network bridge connects multiple network segments at the data link layer (layer 2) of the OSI model. Bridges broadcast to all ports except the port on which the broadcast was received. However, bridges do not promiscuously copy traffic to all ports, as hubs do, but learn which MAC addresses are reachable through specific ports. Once the bridge associates a port and an address, it will send traffic for that address to that port only.

Bridges learn the association of ports and addresses by examining the source address of frames that it sees on various ports. Once a frame arrives through a port, its source address is stored and the bridge assumes that MAC address is associated

with that port. The first time that a previously unknown destination address is seen, the bridge will forward the frame to all ports other than the one on which the frame arrived.

Bridges come in three basic types:

- **Local bridges**: Directly connect local area networks (LANs)
- **Remote bridges**: Can be used to create a wide area network (WAN) link between LANs. Remote bridges, where the connecting link is slower than the end networks, largely have been replaced with routers.
- **Wireless bridges**: Can be used to join LANs or connect remote stations to LANs.

Switches:

A network switch is a device that forwards and filters OSI layer 2 datagrams (chunk of data communication) between ports (connected cables) based on the MAC addresses in the packets. A switch is distinct from a hub in that it only forwards the frames to the ports involved in the communication rather than all ports connected. A switch breaks the collision domain but represents itself as a broadcast domain. Switches make forwarding decisions of frames on the basis of MAC addresses. A switch normally has numerous ports, facilitating a star topology for devices, and cascading additional switches. Some switches are capable of routing based on Layer 3 addressing or additional logical levels; these are called multi-layer switches. The term **switch** is used loosely in marketing to encompass devices including routers and bridges, as well as devices that may distribute traffic on load or by application content (e.g., a Web URL identifier).

Routers:

A router is an internetworking device that forwards packets between networks by processing information found in the datagram or packet (Internet protocol information from Layer 3 of the OSI Model). In many situations, this information is processed in conjunction with the routing table (also known as forwarding table). Routers use routing tables to determine what interface to forward packets (this can include the "null" also known as the "black hole" interface because data can go into it, however, no further processing is done for said data).

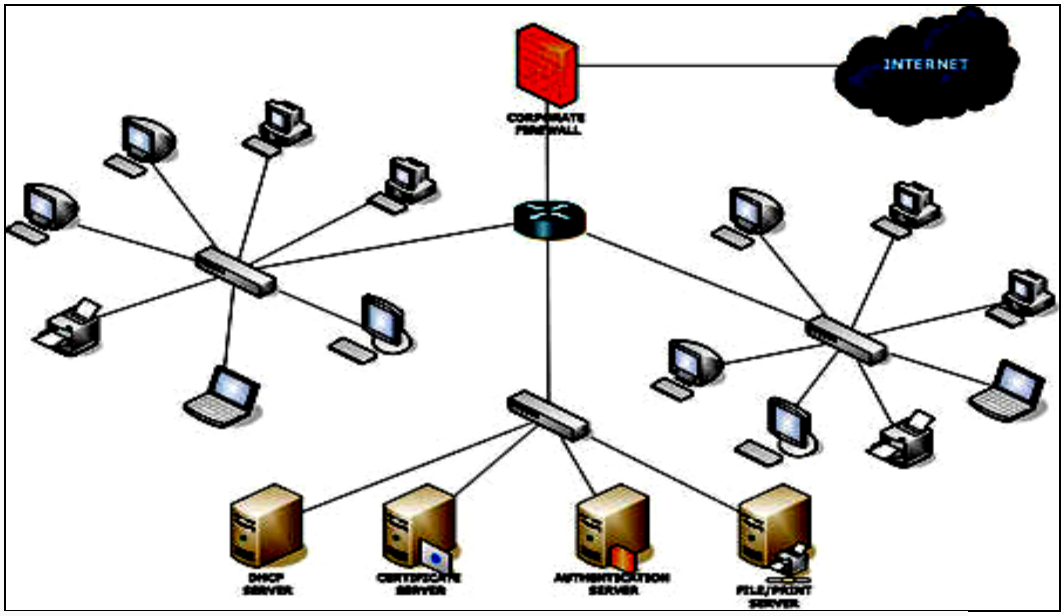
4.2 TYPES OF NETWORKS BASED ON PHYSICAL SCOPE

Common types of computer networks may be identified by their scale.

I. Local area network

A local area network (LAN) is a network that connects computers and devices in a limited geographical area such as home, school, computer laboratory, office building, or closely positioned group of buildings. Each computer or device on the network is a node. Current wired LANs are most likely to be based on Ethernet technology, although new standards like ITU-T G.hn also provide a way to create a wired LAN using existing home wires (coaxial cables, phone lines and power lines).

All interconnected devices must understand the network layer (layer 3), because they are handling multiple subnets (the different colors). Those inside the library, which have only 10/100 Mbit/s Ethernet connections to the user device and a Gigabit Ethernet connection to the central router, could be called "layer 3 switches" because they only have Ethernet interfaces and must understand IP. It would be more correct to call them access routers, where the router at the top is a distribution router that connects to the Internet and academic networks' customer access routers.



The defining characteristics of LANs, in contrast to WANs (Wide Area Networks), include their higher data transfer rates, smaller geographic range, and no need for leased telecommunication lines. Current Ethernet or other IEEE 802.3 LAN technologies operate at speeds up to 10 Gbit/s. This is the data transfer rate. IEEE has projects investigating the standardization of 40 and 100 Gbit/s.

a) Intranets and extranets:

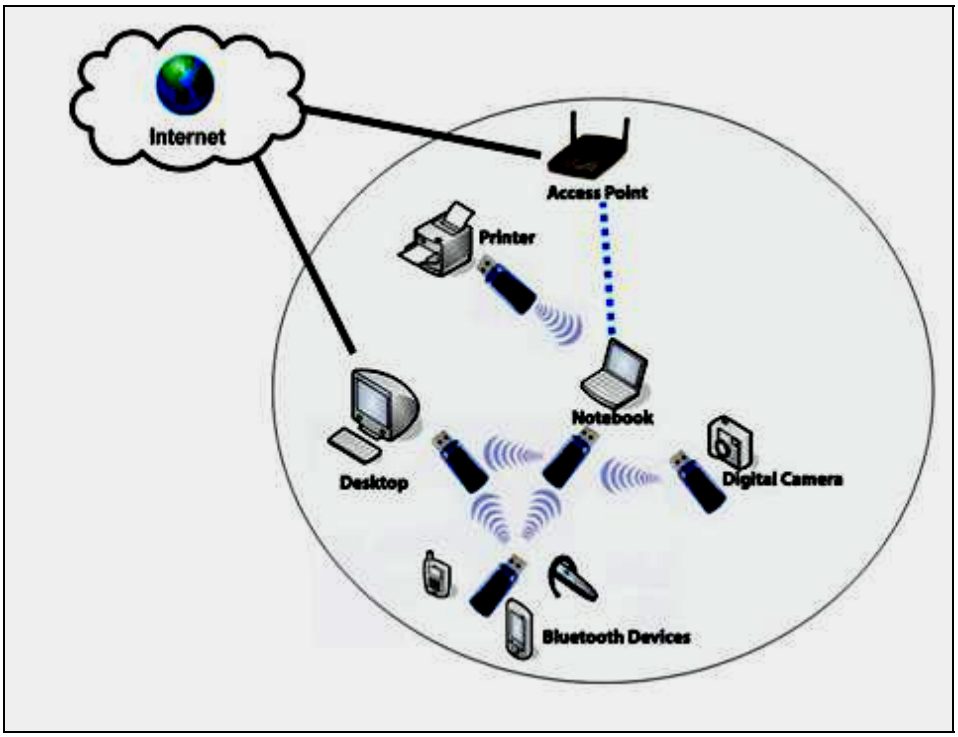
Intranets and extranets are parts or extensions of a computer network, usually **a local area network.**

An intranet is a set of networks, using the Internet Protocol and IP-based tools such as web browsers and file transfer applications that are under the control of a single administrative entity. That administrative entity closes the intranet to all but specific, authorized users. Most commonly, an intranet is the internal network of an organization. A large intranet will typically have at least one web server to provide users with organizational information.

An extranet is a network that is limited in scope to a single organization or entity and also has limited connections to the networks of one or more other usually, but not necessarily, trusted organizations or entities—a company's customers may be given access to some part of its intranet—while at the same time the customers may not be considered *trusted* from a security standpoint. Technically, an extranet may also be categorized as a CAN, MAN, WAN, or other type of network, although an extranet cannot consist of a single LAN; it must have at least one connection with an external network.

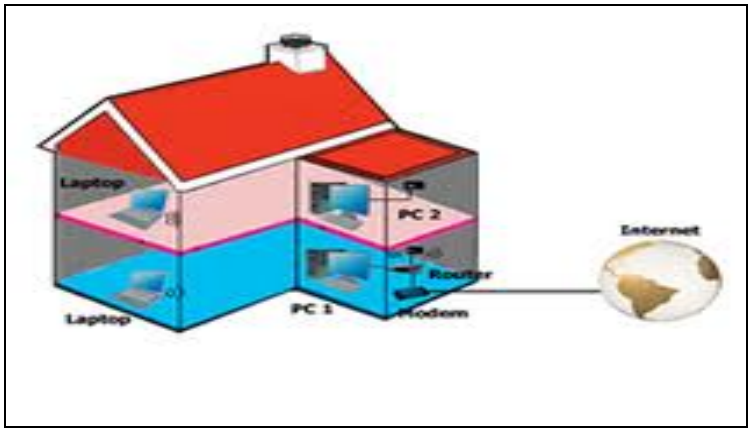
II) Personal area network:

A personal area network (PAN) is a computer network used for communication among computer and different information technological devices close to one person. Some examples of devices that are used in a PAN are personal computers, printers, fax machines, telephones, PDAs, scanners, and even video game consoles. A PAN may include wired and wireless devices. The reach of a PAN typically extends to 10 meters. A wired PAN is usually constructed with USB and Fire wire connections while technologies such as Bluetooth and infrared communication typically form a wireless PAN.



III) Home area network:

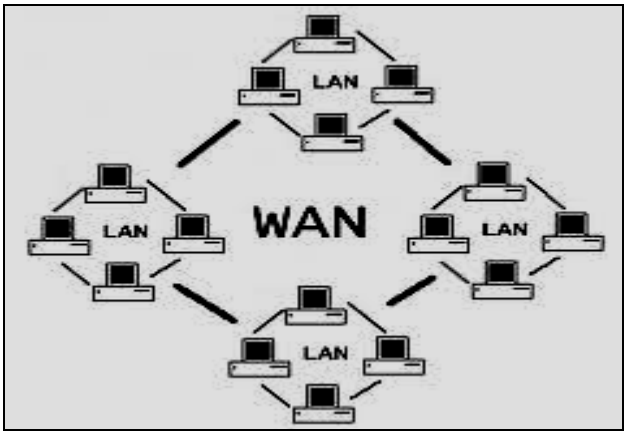
A home area network (HAN) is a residential LAN which is used for communication between digital devices typically deployed in the home, usually a small number of personal computers and accessories, such as printers and mobile computing devices. An important function is the sharing of Internet access, often a broadband service through a CATV or Digital Subscriber Line (DSL) provider. It can also be referred as an office area network (OAN).



(IV) Wide area network:

Wide area network (WAN) is a computer network that covers a large geographic area such as a city, country, or spans even intercontinental distances, using a communications channel

that combines many types of media such as telephone lines, cables, and air waves. A WAN often uses transmission facilities provided by common carriers, such as telephone companies. WAN technologies generally function at the lower three layers of the OSI reference model: the physical layer, the data link layer, and the network layer.



Several options are available for WAN connectivity. They are as follows:

Option:	Description	Advantages	Disadvantages	Bandwidth range	Sample protocols used
Leased line	Point-to-Point connection between two computers or Local Area Networks (LANs)	Most secure	Expensive		PPP,HDLC ,SDLC, HNAS
Circuit switching	A dedicated circuit path is created between end points. Best example is dialup conn ections	Less Expensive	Call Setup	28 - 144 kbit/s	PPP,ISDN

Packet switching	Devices transport packets via a shared single point-to-point or point-to-multipoint link across a carrier internet work. Variable length packets are transmitted over Permanent Virtual Circuits (PVC) or Switched Virtual Circuits (SVC)		Shared media across link		X.25Frame -Relay
Cell relay	Similar to packet switching, but uses fixed length cells instead of variable length packets. Data is divided into fixed-length cells and then transported across virtual circuits	Best for simultaneous use of voice and data	Overhead can be considerable		ATM

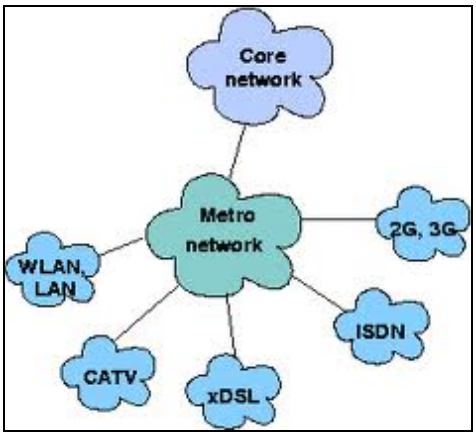
(V) Campus network:

A campus network is a computer network made up of an interconnection of local area networks (LAN's) within a limited geographical area. The networking equipments (switches, routers) and transmission media (optical fiber, copper plant, Cat5 cabling etc.) are almost entirely owned (by the campus tenant / owner: an enterprise, university, government etc.).

In the case of a university campus-based campus network, the network is likely to link a variety of campus buildings including; academic departments, the university library and student residence halls.

(VI) Metropolitan area network:

A Metropolitan area network is a large computer network that usually spans a city or a large campus.



The IEEE 802-2001 standard describes a MAN as being:

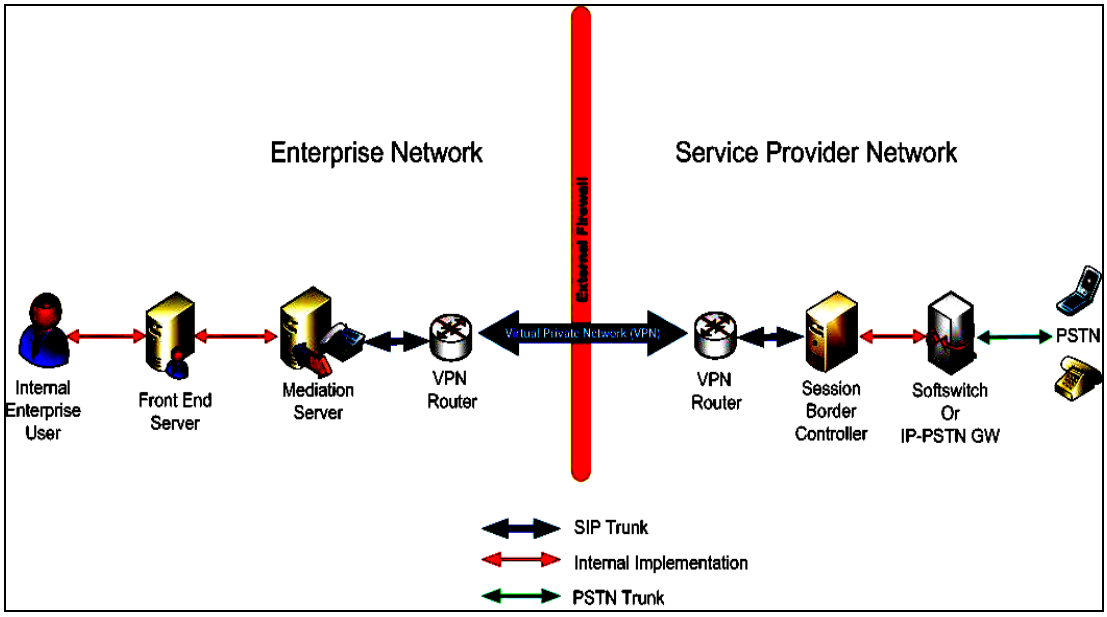
" A MAN is optimized for a larger geographical area than a LAN, ranging from several blocks of buildings to entire cities. MANs can also depend on communications channels of moderate-to-high data rates. A MAN might be owned and operated by a single organization, but it usually will be used by many individuals and organizations. MANs might also be owned and operated as public utilities. They will often provide means for internetworking of local networks. "

(VII) Enterprise private network:

An enterprise private network is a network build by an enterprise to interconnect various company sites, e.g., production sites, head offices, remote offices, shops, in order to share compute resources.

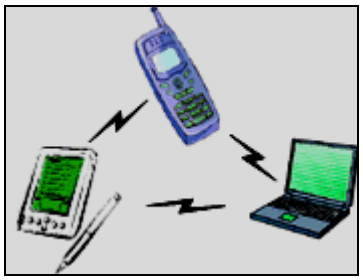
4.3 WIRELESS NETWORK

It refers to any type of computer network that is wireless, and is commonly associated with a telecommunications network whose interconnections between nodes are implemented without the use of wires. Wireless telecommunications networks are generally implemented with some type of remote information transmission system that uses electromagnetic waves, such as radio waves, for the carrier and this implementation usually takes place at the physical level or "layer" of the network.



4.3.1 Types of wireless connections:

- **Wireless PAN**
  Wireless Personal Area Networks (WPANs) interconnect devices within a relatively small area, generally within reach of a person. For example, Bluetooth provides a WPAN for interconnecting a headset to a laptop. ZigBee also supports WPAN applications. Wi-Fi PANs are also getting popular as vendors have started integrating Wi-Fi in variety of consumer electronic devices. Intel My WiFi and Windows 7 virtual Wi-Fi capabilities have made Wi-Fi PANs simpler and easier to set up and configure.

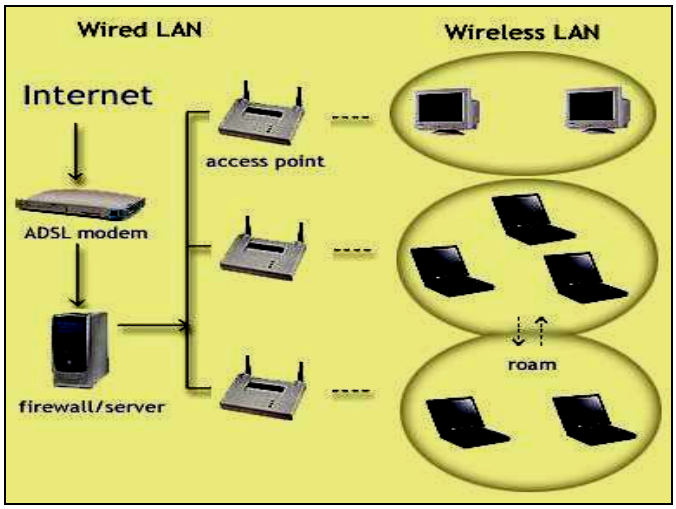


- **Wireless LAN**
  A wireless local area network (WLAN) links two or more devices using a wireless distribution method (typically spread-spectrum or OFDM radio), and usually providing a connection through an access point to the wider internet. This gives users the mobility to move around within a local coverage area and still be connected to the network.

  Wi-Fi: Wi-Fi is increasingly used as a synonym for 804.11 WLANs, although it is technically a certification of interoperability between 804.11 devices.

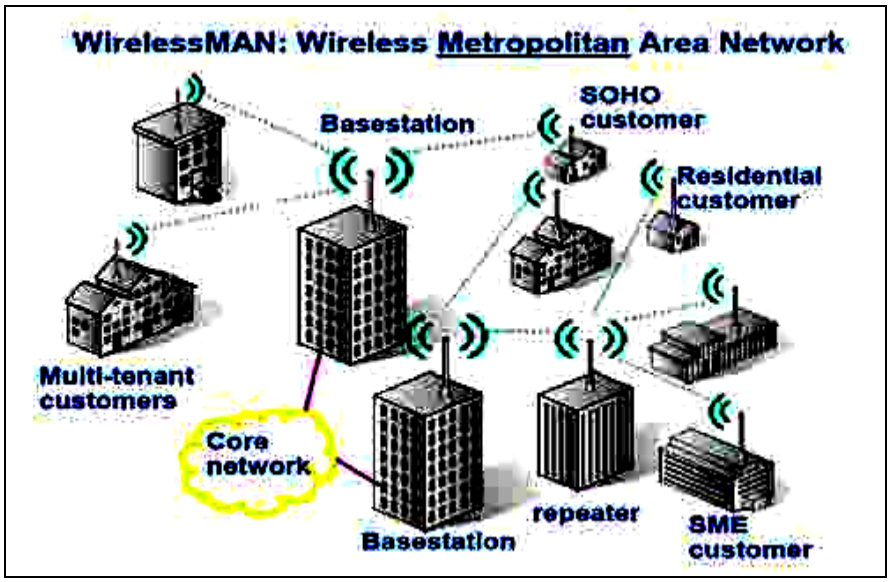
Fixed Wireless Data:

This implements point to point links between computers or networks at two locations. It is often using dedicated microwave or laser beams over line of sight paths. It is often used in cities to connect networks in two or more buildings without physically wiring the buildings together.



- **Wireless MAN**

Wireless Metropolitan area networks are a type of wireless network that connects several Wireless LANs. WiMAX is the term used to refer to wireless MANs.



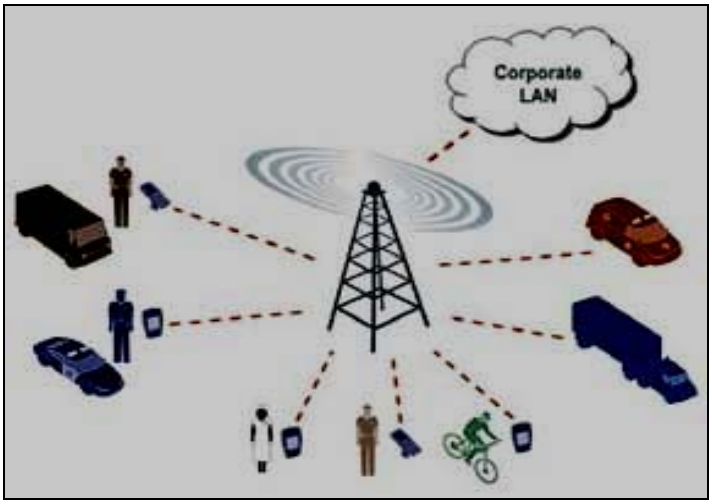
- **Wireless WAN**
  Wireless wide area networks are wireless networks that typically cover large outdoor areas. These networks can be used to connect branch offices of business or as a public internet access system. They are usually deployed on the 4.4 GHz band. A typical system contains base station gateways, access points and wireless bridging relays. Other configurations are mesh systems where each access point acts as a relay also. When combined with renewable energy systems such as photo-voltaic solar panels or wind systems they can be stand alone systems.

  - **Mobile devices networks**
    With the development of smart phones, cellular telephone networks routinely carry data in addition to telephone conversations:

- **Global System for Mobile Communications (GSM)**: The GSM network is divided into three major systems: the switching system, the base station system, and the operation and support system. The cell phone connects to the base system station which then connects to the operation and support station; it then connects to the switching station where the call is transferred to where it needs to go. GSM is the most common standard and is used for a majority of cell phones.

- **Personal Communications Service (PCS)**: PCS is a radio band that can be used by mobile phones in North America and South Asia. Sprint happened to be the first service to set up a PCS.

- **D-AMPS:** Digital Advanced Mobile Phone Service, an upgraded version of AMPS, is being phased out due to advancement in technology. The newer GSM networks are replacing the older system.



4.3.2 Uses of Wireless Network:

- Wireless networks have continued to develop and their uses have grown significantly.
- Cellular phones are part of huge wireless network systems.
- People use these phones daily to communicate with one another.
- Sending information overseas is possible through wireless network systems using satellites and other signals to communicate across the world.
- Emergency services such as the police department utilize wireless networks to communicate important information quickly.
- People and businesses use wireless networks to send and share data quickly whether it be in a small office building or across the world.
- Another important use for wireless networks is as an inexpensive and rapid way to be connected to the Internet in countries and regions where the telecom infrastructure is poor or there is a lack of resources, as in most developing countries.

4.3.3 Problems with Wireless Network:

- Compatibility issues also arise when dealing with wireless networks. Different components not made by the same company may not work together, or might require extra work to fix these issues.
- Wireless networks are typically slower than those that are directly connected through an Ethernet cable.
- A wireless network is more vulnerable, because anyone can try to break into a network broadcasting a signal. Many networks offer WEP - **Wired Equivalent Privacy** - security systems which have been found to be vulnerable to intrusion. Though WEP does block some intruders, the security problems have caused some businesses to stick with wired networks until security can be improved. Another type of security for wireless networks is **WPA** - Wi-Fi Protected Access. WPA provides more security to wireless networks than a WEP security set up. The use of **firewalls** will help with security breaches which can help to fix security problems in some wireless networks that are more vulnerable.

Environmental concerns and health hazard:

In recent times, there have been increased concerns about the safety of wireless communications, despite little evidence of health risks so far. The president of Lakehead University refused to agree to installation of a wireless network citing a California Public Utilities Commission study which said that the possible risk of tumors and other diseases due to exposure to electromagnetic fields (EMFs) needs to be further investigated.

4.4 INTERNETWORK

Interconnection of networks

Internetworking started as a way to connect disparate types of networking technology, but it became widespread through the developing need to connect two or more networks via some sort of wide area network. The original term for an internetwork was catenet.

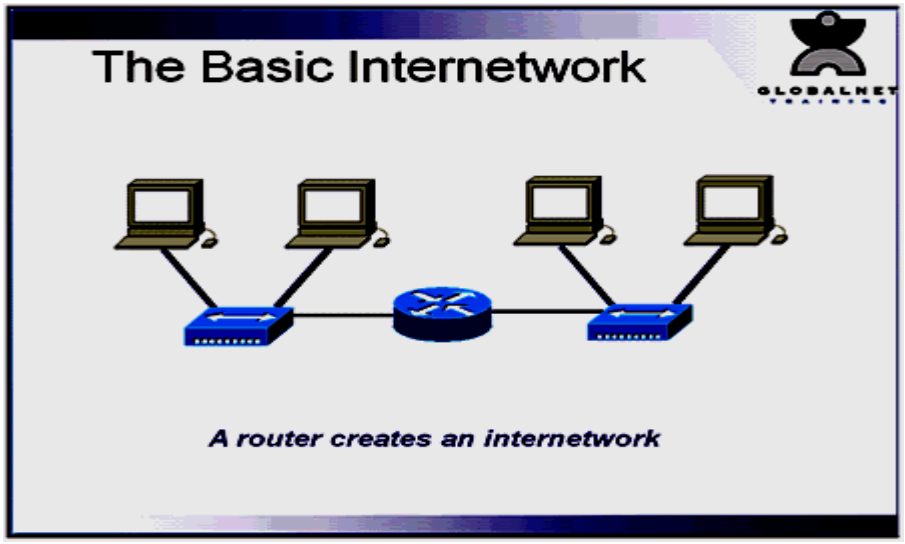
The definition of an internetwork today includes the connection of other types of computer networks such as personal area networks.

The network elements used to connect individual networks in the ARPANET, the predecessor of the Internet, were originally called gateways, but the term has been deprecated in this context, because of possible confusion with functionally different devices. Today the interconnecting gateways are called Internet routers.

Another type of interconnection of networks often occurs within enterprises at the Link Layer of the networking model, i.e. at the hardware-centric layer below the level of the TCP/IP logical interfaces. Such interconnection is accomplished with network bridges and network switches. This is sometimes incorrectly termed internetworking, but the resulting system is simply a larger, single sub-network, and no internetworking protocol, such as Internet Protocol, is required to traverse these devices. However, a single computer network may be converted into an internetwork by dividing the network into segments and logically dividing the segment traffic with routers.

The Internet Protocol is designed to provide an unreliable (not guaranteed) packet service across the network. The architecture avoids intermediate network elements maintaining any state of the network. Instead, this function is assigned to the endpoints of each communication session. To transfer data reliably, applications must utilize an appropriate Transport Layer protocol, such as Transmission Control Protocol (TCP), which provides a reliable stream. Some applications use a simpler, connection-less

transport protocol, User Datagram Protocol (UDP), for tasks which do not require reliable delivery of data or that require real-time service, such as video streaming. An internetwork is the connection of two or more private computer networks via a common routing technology (OSI Layer 3) using routers. The Internet is an aggregation of many internetworks; hence its name was shortened to Internet.



Global area network:

A global area network (GAN) is a network used for supporting mobile communications across an arbitrary number of wireless LANs, satellite coverage areas, etc. The key challenge in mobile communications is handing off the user communications from one local coverage area to the next. In IEEE Project 802, this involves a succession of terrestrial wireless LANs.

Internet:

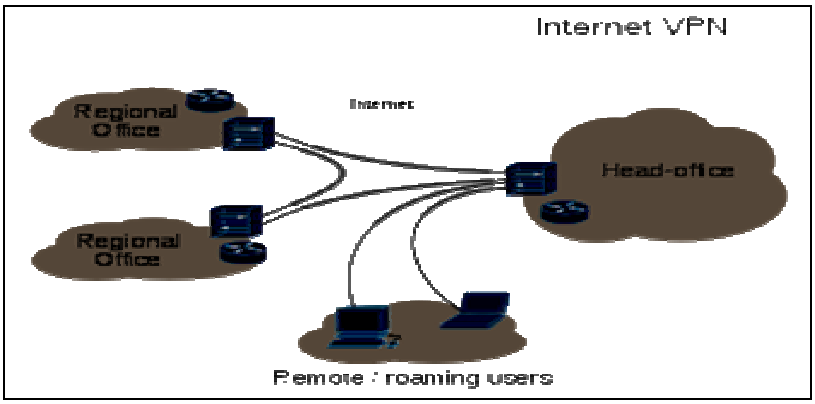
The Internet is a global system of interconnected governmental, academic, corporate, public, and private computer networks. It is based on the networking technologies of the Internet Protocol Suite. It is the successor of the Advanced Research Projects Agency Network (ARPANET) developed by DARPA of the United States Department of Defense. The Internet is also the communications backbone underlying the World Wide Web (WWW).

Participants in the Internet use a diverse array of methods of several hundred documented, and often standardized, protocols compatible with the Internet Protocol Suite and an addressing system (IP addresses) administered by the Internet Assigned Numbers Authority and address registries. Service providers and large enterprises exchange information about the reachability of their address spaces through the Border Gateway Protocol (BGP), forming a redundant worldwide mesh of transmission paths.

4.4.1 Virtual private network:

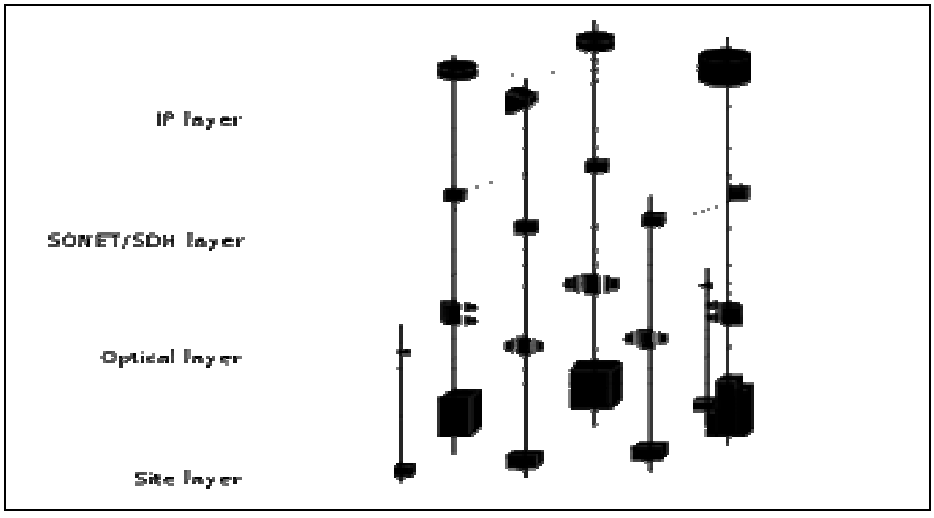
A virtual private network (VPN) is a computer network in which some of the links between nodes are carried by open connections or virtual circuits in some larger network (e.g., the Internet) instead of by physical wires. The data link layer protocols of the virtual network are said to be tunneled through the larger network when this is the case. One common application is secure communications through the public Internet, but a VPN need not have explicit security features, such as authentication or content encryption. VPNs, for example, can be used to separate the traffic of different user communities over an underlying network with strong security features.

A VPN may have best-effort performance, or may have a defined service level agreement (SLA) between the VPN customer and the VPN service provider. Generally, a VPN has a topology more complex than point-to-point.



4.4.2 Overlay network:

An overlay network is a virtual computer network that is built on top of another network. Nodes in the overlay are connected by virtual or logical links, each of which corresponds to a path, perhaps through many physical links, in the underlying network.



For example, many peer-to-peer networks are overlay networks because they are organized as nodes of a virtual system of links run on top of the Internet. The Internet was initially built as an overlay on the telephone network .

Overlay networks have been around since the invention of networking when computer systems were connected over telephone lines using modem, before any data network existed.

Nowadays the Internet is the basis for many overlaid networks that can be constructed to permit routing of messages to destinations not specified by an IP address. For example, distributed hash tables can be used to route messages to a node having a specific logical address, whose IP address is not known in advance.

Overlay networks have also been proposed as a way to improve Internet routing, such as through quality of service guarantees to achieve higher-quality streaming media. On the other hand, an overlay network can be incrementally deployed on end-hosts running the overlay protocol software, without cooperation from Internet service providers. The overlay has no control over how packets are routed in the underlying network between two overlay nodes, but it can control, for example, the sequence of overlay nodes a message traverses before reaching its destination.

5

NETWORK MODEL

Unit Structure

5.1 Introduction
3.2 The OSI Reference model
5.3 The TCP/IP Reference model
5.4 A comparison of the OSL and TCP Reference models

5.1 INTRODUCTION

Networking models

Two architectural models are commonly used to describe the protocols and methods used in internetworking.

The Open System Interconnection (OSI) reference model was developed under the auspices of the International Organization for Standardization (ISO) and provides a rigorous description for layering protocol functions from the underlying hardware to the software interface concepts in user applications. Internetworking is implemented in the Network Layer (Layer 3) of the model.

The Internet Protocol Suite, also called the TCP/IP model of the Internet was not designed to conform to the OSI model and does not refer to it in any of the normative specifications in Requests and Internet standards. Despite similar appearance as a layered model, it uses a much less rigorous, loosely defined architecture that concerns itself only with the aspects of logical networking. It does not discuss hardware-specific low-level interfaces, and assumes availability of a Link Layer interface to the local network link to which the host is connected. Internetworking is facilitated by the protocols of its Internet Layer.

Connection:

TCP is a connection-oriented protocol. It establishes a virtual path between the source and destination. All the segments belonging to a message are then sent over this virtual path. Using a single virtual pathway for the entire message facilitates the acknowledgment process as well as retransmission of damaged or lost frames. In TCP, connection-oriented transmission requires two procedures:

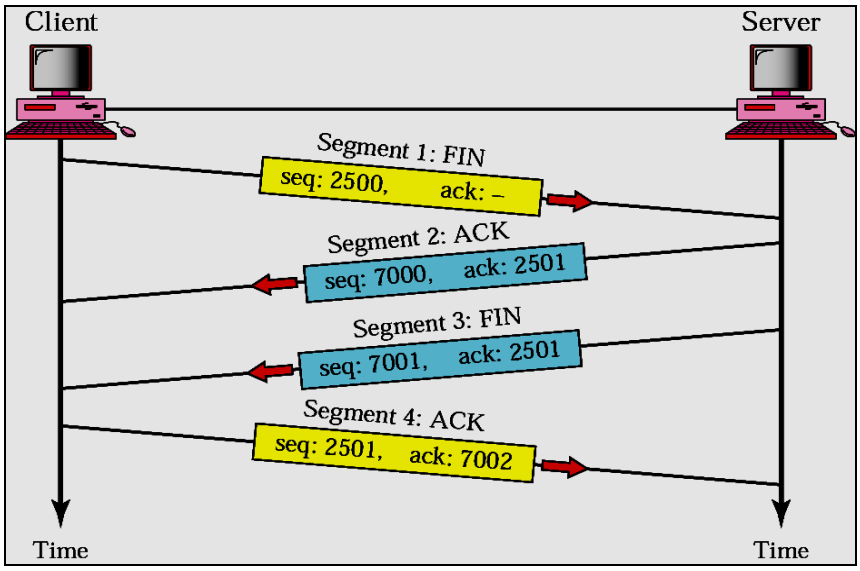
(1) Connection Establishment and
(2) Connection Termination.

Connection Establishment

TCP transmits data in full-duplex mode. When two TCPs in two machines are connected, they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data transfer.

Four steps are needed to establish the connection, as discussed before.

However, the second and third steps can be combined to create a three-step connection, **called a three-way handshake,** as shown in Figure.



The steps of the process are as follows:

(1) The client sends the first segment, a SYN segment. The segment includes the source and destination port numbers. The destination port number clearly defines the server to which the client wants to be connected. The segment also contains the client initialization sequence number (ISN) used for numbering the bytes of data sent from the client to the server.

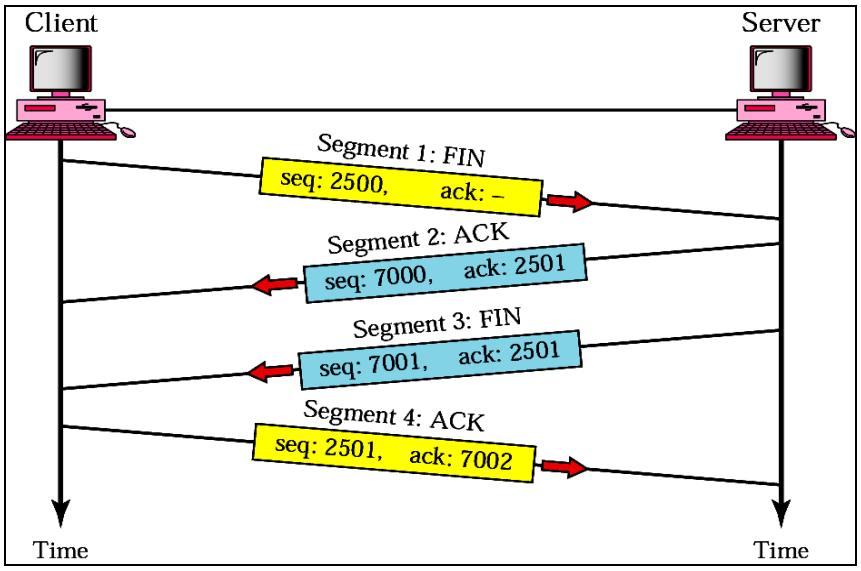
(2) The server sends the second segment; a SYN and an ACK segment. This segment has a dual purpose. First, it acknowledges the receipt of the first segment, using the ACK flag and acknowledgment number field. Note that the acknowledgment number is the client initialization sequence number plus 1 because no user data have been sent in

segment 1. The server must also define the client window size. Second, the segment is used as the initialization segment for the server. It contains the initialization sequence number used to number the bytes sent from the server to the client.

(3) The client sends the third segment. This is just an ACK segment. It acknowledges the receipt of the second segment, using the ACK flag and acknowledgment number field. Note that the acknowledgment number is the server initialization sequence number plus 1 because no user data have been sent in segment 2. The client must also define the server window size. Data can be sent with the third packet.

Connection Termination

Any of the two parties involved in exchanging data (client or server) can close the connection. When connection in one direction is terminated, the other party can continue sending data in the other direction. Therefore, four steps are needed to close the connections in both directions, as shown in Figure.



The four steps are as follows:

(1) The client TCP sends the first segment, a FIN segment.

(2) The server TCP sends the second segment, an ACK segment, to confirm the receipt of the FIN segment from the client. Note that the acknowledgment number is 1 plus the sequence number received in the FIN segment because no user data have been sent in segment 1.

(3) The server TCP can continue sending data in the server-client direction. When it does not have any more data to send, it sends the third segment. This segment is a FIN segment.

(4) The client TCP sends the fourth segment, an ACK segment, to confirm the receipt of the FIN segment from the TCP server. Note that the acknowledgment number is 1 plus the sequence number received in the FIN segment from the server.

Connection Resetting:

TCP may request the resetting of a connection. Resetting here means that the current connection is destroyed. This happens in one of three cases:

(1) The TCP on one side has requested a connection to a nonexistent port. The TCP on the other side may send a segment with its RST bit set to annul the request.

(2) One TCP may want to abort the connection due to an abnormal situation. It can send an RST segment to close the connection.

(3) The TCP on one side may discover that the TCP on the other side has been idle for a long time. It may send an RST segment to destroy the connection

When is TCP open, TCP half opened?

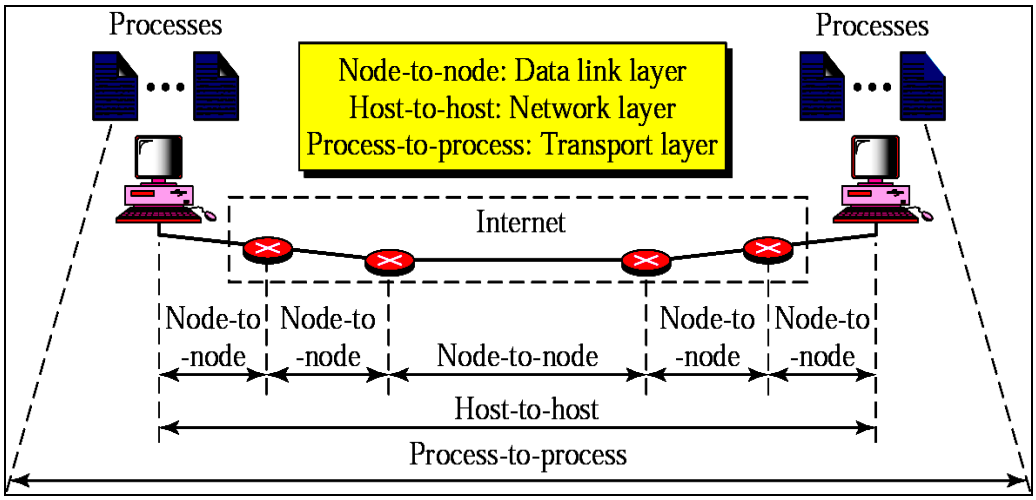
A three-step process is shown in Figure above. After the server receives the initial SYN packet, the connection is in a half-opened state. The server replies with its' own sequence number, and awaits an acknowledgment, the third and final packet of a TCP open.

Attackers have gamed this half-open state. SYN attacks flood the server with the first packet only, hoping to swamp the host with half-open connections that will never be completed. In addition, the first part of this three-step process can be used to detect active TCP services without alerting the application programs, which usually aren't informed of incoming connections until the three-packet handshake is complete.

The sequence numbers have another function. Because the initial sequence number for new connections changes constantly, it is possible for TCP to detect stale packets from previous incarnations of the same circuit (i.e., from previous uses of the same 4-tuple).

There is also a modest security benefit: A connection cannot be fully established until both sides have acknowledged the other's initial sequence number.

Node-to-Node, Host-to-Host and Process-to-Process deliveries:



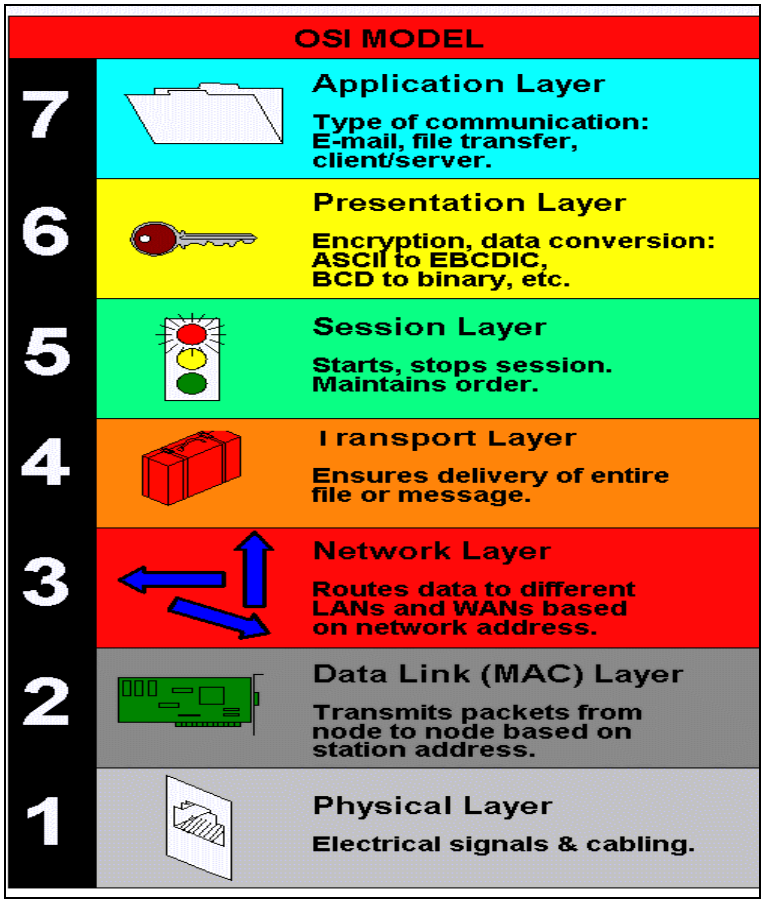
Connection oriented v/s connectionless deliveries:

Parameter	**Connection oriented**	**Connectionless**
Definition	A characteristic of a network system that requires a pair of computers to establish a connection before sending data. **Example: Telephone line**	A characteristic of network system that allows a computer to send data to any other computer at any time without any prerequisite of destination connection **Example: Postal system**
PDU movement	Sequential	To transmit data in such a way that each PDU is treated independently of all prior PDUs
Three way handshake	Connection establishment requires "three-way hand shake"	Nothing of this sort
Modus Operandi	Three simple steps: **(1)** Connection establishment **Agree on :** **(a)** Syntax, **(b)** Semantics & **(c)** Timing **(2)** Data Transfer & **(3)** Connection termination	Nothing of this sort

Decision on path	Only at the beginning	At every node
Sequence	Keeps the sequence	Can arrive out of sequence
Example	TCP	UDP
Reliability	Reliable	Unreliable

5.2 OSI REFERENCE MODEL:

In 1978, work on a layered model of network architecture was started and the International Organization for Standardization (ISO) began to develop its OSI framework architecture. OSI has two major components: an abstract model of networking, called the Basic Reference Model or seven-layer model, and a set of specific protocols.



The **Open Systems Interconnection model** (**OSI model**) is a product of the Open Systems Interconnection effort at the International Organization for Standardization. It is a way of sub-dividing a communications system into smaller parts called layers. A layer is a collection of conceptually similar functions that provide services to the layer above it and receives services from

the layer below it. On each layer an **instance** provides services to the instances at the layer above and requests service from the layer below.

For example, a layer that provides error-free communications across a network provides the path needed by applications above it, while it calls the next lower layer to send and receive packets that make up the contents of the path. Conceptually two instances at one layer are connected by a horizontal protocol connection on that layer.

7. Application Layer
NNTP · SIP · SSI · DNS · FTP ·Gopher · HTTP · NFS · NTP · SMPP ·SMTP · DHCP · SNMP · Telnet ·(more)
6. Presentation Layer
MIME · XDR · TLS · SSL
5. Session Layer
Named Pipes · NetBIOS · SAP · SIP ·L2TP · PPTP
4. Transport Layer
TCP · UDP · SCTP · DCCP
3. Network Layer
IP (IPv4, IPv6) · ICMP · IPSec · IGMP ·IPX · AppleTalk
2. Data Link Layer
ARP · CSLIP · SLIP · Ethernet · Frame relay · ITU-T G.hn DLL · PPP
1. Physical Layer
RS-232 · RS-449 SONET/SDH ·OTN · DSL . PHY ·Ethernet · USB · Bluetooth · Firewire (IEEE 1394)

Description of OSI layers

According to X.200 Recommendation, there are 7 layers, each one is generically called N layer. The N+1 entity ask for transmission services to the N entity.

At each level two entities (N-entity) interacts by means of the (N) protocol by transmitting Protocol Data Units (PDU). Service Data Unit (SDU) is a specific unit of data that has been passed down from an OSI layer, to a lower layer, and has not yet been encapsulated into a Protocol data Unit (PDU), by the lower layer. It

is a set of data that is sent by a user of the services of a given layer, and is transmitted semantically unchanged to a peer service user. The PDU at any given layer, layer 'n', is the SDU of the layer below, layer 'n-1'. In effect the SDU is the 'payload' of a given PDU. That is, the process of changing a SDU to a PDU consists of an encapsulation process, performed by the lower layer. All the data contained in the SDU becomes encapsulated within the PDU. The layer n-1 adds headers or footers, or both, to the SDU, transforming it into a PDU of layer n-1. The added headers or footers are part of the process used to make it possible to get data from a source to a destination.

OSI Model			
	Data unit	**Layer**	**Function**
Host layers	**Data**	**7. Application**	**Network process to application**
		6. Presentation	**Data representation, encryption and decryption, convert machine dependent data to machine independent data**
		5. Session	**Inter-host communication**
	Segments	**4. Transport**	**End-to-end connections and reliability, Flow control**
Media layers	**Packet**	**3. Network**	**Path determination and logical addressing**
	Frame	**2. Data Link**	**Physical addressing**
	Bit	**1. Physical**	**Media, signal and binary transmission**

Some orthogonal aspects, such as management and security, involve every layer.

Layer 1: Physical Layer

The Physical Layer defines the electrical and physical specifications for devices. In particular, it defines the relationship between a device and a transmission medium, such as a copper or optical cable. This includes the layout of pins, voltages, cable specifications, hubs, repeaters, network adapters, host bus adapters (HBA used in storage area networks) and more.

To understand the function of the Physical Layer, contrast it with the functions of the Data Link Layer. Think of the Physical Layer as concerned primarily with the interaction of a single device with a medium, whereas the Data Link Layer is concerned more with the interactions of multiple devices (i.e., at least two) with a

shared medium. Standards such as RS-232 do use physical wires to control access to the medium.

The major functions and services performed by the Physical Layer are:

- Establishment and termination of a connection to a communications medium.
- Participation in the process whereby the communication resources are effectively shared among multiple users. For example, contention resolution and flow control.
- Modulation or conversion between the representation of digital data in user equipment and the corresponding signals transmitted over a communications channel. These are signals operating over the physical cabling (such as copper and optical fiber) or over a radio link.

Parallel SCSI buses operate in this layer, although it must be remembered that the logical SCSI protocol is a Transport Layer protocol that runs over this bus. Various Physical Layer Ethernet standards are also in this layer; Ethernet incorporates both this layer and the Data Link Layer. The same applies to other local-area networks, such as token ring, FDDI, ITU-TG.hn and IEEE 802.11, as well as personal area networks such as Bluetooth and IEEE 802.15.4.

Layer 2: Data Link Layer

The Data Link Layer provides the functional and procedural means to transfer data between network entities and to detect and possibly correct errors that may occur in the Physical Layer. Originally, this layer was intended for point-to-point and point-to-multipoint media, characteristic of wide area media in the telephone system. Local area network architecture, which included broadcast-capable multi-access media, was developed independently of the ISO work in IEEE Project 802. IEEE work assumed sub layering and management functions not required for WAN use. In modern practice, only error detection, not flow control using sliding window, is present in data link protocols such as Point-to-Point Protocol (PPP), and, on local area networks, the IEEE 802.2 LLC layer is not used for most protocols on the Ethernet, and on other local area networks, its flow control and acknowledgment mechanisms are rarely used. Sliding window flow control and acknowledgment is used at the Transport Layer by protocols such as TCP, but is still used in niche where X.25 offers performance advantages.

Both WAN and LAN service arranges bits, from the Physical Layer, into logical sequences called frames. Not all Physical Layer

bits necessarily go into frames, as some of these bits are purely intended for Physical Layer functions. For example, every fifth bit of the FDDI bit stream is not used by the Layer.

WAN Protocol architecture

Connection-oriented WAN data link protocols, in addition to framing, detect and may correct errors. They are also capable of controlling the rate of transmission. A WAN Data Link Layer might implement a sliding window flow control and acknowledgment mechanism to provide reliable delivery of frames; that is the case for SDLC and HDLC.

Layer 3: Network Layer

The Network Layer provides the functional and procedural means of transferring variable length data sequences from a source to a destination via one or more networks, while maintaining the quality of service requested by the Transport Layer. The Network Layer performs network routing functions, and might also perform fragmentation and reassembly, and report delivery errors. Routers operate at this layer—sending data throughout the extended network and making the Internet possible. This is a logical addressing scheme – values are chosen by the network engineer. The addressing scheme is not hierarchical.

Careful analysis of the Network Layer indicated that the Network Layer could have at least **3 sub layers:**

1. Sub-network Access - that considers protocols that deal with the interface to networks, such as X.25;

2. Sub-network Dependent Convergence - when it is necessary to bring the level of a transit network up to the level of networks on either side;

3. Sub-network Independent Convergence - This handles transfer across multiple networks. The best example of this latter case is CLNP, or IPv7 ISO 8473. It manages the connectionless transfer of data one hop at a time, from end system to ingress router, router to router, and from egress router to destination end system. It is not responsible for reliable delivery to a next hop, but only for the detection of errored packets so they may be discarded. In this scheme, IPv4 and IPv6 would have to be classed with X.25 as Subnet Access protocols because they carry interface addresses rather than node addresses.

A number of layer management protocols belong to the Network Layer. These include routing protocols, multicast group management, Network Layer information and error, and Network Layer address assignment. It is the function of the payload that

makes these belong to the Network Layer, not the protocol that carries them.

Layer 4: Transport Layer

The Transport Layer provides transparent transfer of data between end users, providing reliable data transfer services to the upper layers. The Transport Layer controls the reliability of a given link through flow control, segmentation/desegmentation, and error control. Some protocols are state and connection oriented. This means that the Transport Layer can keep track of the segments and retransmit those that fail. The Transport layer also provides the acknowledgement of the successful data transmission and if no error free data was transferred then sends the next data.

Although not developed under the OSI Reference Model and not strictly conforming to the OSI definition of the Transport Layer, typical examples of Layer 4 are the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

Of the actual OSI protocols, there are five classes of connection-mode transport protocols ranging from class 0 (which is also known as TP0 and provides the least features) to class 4 (TP4, designed for less reliable networks, similar to the Internet). Class 0 contains no error recovery, and was designed for use on network layers that provide error-free connections. Class 4 is closest to TCP, although TCP contains functions, such as the graceful close, which OSI assigns to the Session Layer. Also, all OSI TP connection-mode protocol classes provide expedited data and preservation of record boundaries, both of which TCP is incapable. Detailed characteristics of TP0-4 classes are shown in the following table:

Feature Name	TP0	TP1	TP2	TP3	TP4
Connection oriented network	Yes	Yes	Yes	Yes	Yes
Connectionless network	No	No	No	No	Yes
Concatenation and separation	No	Yes	Yes	Yes	Yes
Segmentation and reassembly	Yes	Yes	Yes	Yes	Yes
Error Recovery	No	Yes	No	Yes	Yes
Reinitiate connection (if an excessive number of PDUs are unacknowledged)	No	Yes	No	Yes	No
Multiplexing and de-multiplexing	No	No	Yes	Yes	Yes

over a single virtual circuit					
Explicit flow control	No	No	Yes	Yes	Yes
Retransmission on timeout	No	No	No	No	Yes
Reliable Transport Service	No	Yes	No	Yes	Yes

Perhaps an easy way to visualize the Transport Layer is to compare it with a Post Office, which deals with the dispatch and classification of mail and parcels sent. Do remember, however, that a post office manages the outer envelope of mail. Higher layers may have the equivalent of double envelopes, such as cryptographic presentation services that can be read by the addressee only. Roughly speaking, tunneling protocols operate at the Transport Layer, such as carrying non-IP protocols such as IBM's SNA or Novell's IPX over an IP network, or end-to-end encryption with IPSec. While **Generic Routing Encapsulation** (GRE) might seem to be a Network Layer protocol, if the encapsulation of the payload takes place only at endpoint, GRE becomes closer to a transport protocol that uses IP headers but contains complete frames or packets to deliver to an endpoint. L2TP carries PPP frames inside transport packet.

Layer 5: Session Layer

The Session Layer controls the dialogues (connections) between computers. It establishes, manages and terminates the connections between the local and remote application. It provides for full-duplex, half-duplex, or simplex operation, and establishes check pointing, adjournment, termination, and restart procedures. The OSI model made this layer responsible for graceful close of sessions, which is a property of the Transmission Control Protocol, and also for session check pointing and recovery, which is not usually used in the Internet Protocol Suite. The Session Layer is commonly implemented explicitly in application environments that use remote procedure calls.

Layer 6: Presentation Layer

The Presentation Layer establishes context between Application Layer entities, in which the higher-layer entities may use different syntax and semantics if the presentation service provides a mapping between them. If a mapping is available, presentation service data units are encapsulated into session protocol data units, and passed down the stack.

This layer provides independence from data representation (e.g., encryption) by translating between application and network formats. The presentation layer transforms data into the form that

the application accepts. This layer formats and encrypts data to be sent across a network. It is sometimes called the syntax layer.

The original presentation structure used the basic encoding rules of **Abstract Syntax Notation One (ASN.1),** with capabilities such as converting an EBCDIC-coded text file to an ASCII-coded file, or serialization of objects and other data structures from and to XML.

Layer 7: Application Layer

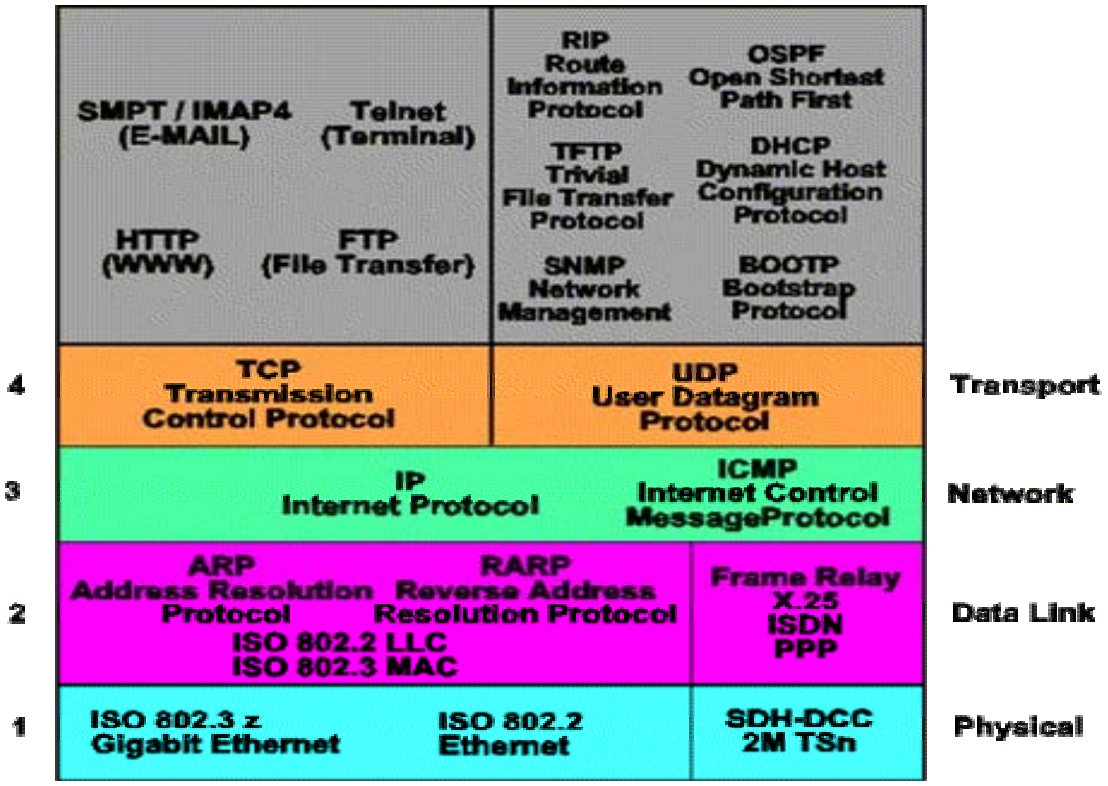
The Application Layer is the OSI layer closest to the end user, which means that both the OSI application layer and the user interact directly with the software application. This layer interacts with software applications that implement a communicating component. Such application programs fall outside the scope of the OSI model. Application layer functions typically include identifying communication partners, determining resource availability, and synchronizing communication. When identifying communication partners, the application layer determines the identity and availability of communication partners for an application with data to transmit. When determining resource availability, the application layer must decide whether sufficient network or the requested communication exists. In synchronizing communication, all communication between applications requires cooperation that is managed by the application layer. Some examples of application layer implementations include Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP) and X.400 Mail.

5.3 TCP/IP - INTERNET TRANSPORT SERVICES

5.3.1 Introduction and Background

TCP/IP has matured to become the backbone of the Information superhighway. The TCP/IP protocols provide users with the backbone services used to carry popular services such as the World Wide Web (WWW), E-Mail, and others. From its beginnings in the late 70's as a U.S. Government Advanced Research Projects Agency (ARPA) research and development project, the "ARPANET" has grown to provide connections to systems worldwide, helping bring the information age into reality.

With a layered architecture, the TCP/IP suite can be understood as a set of discrete components. The lower layers (transport and below) of the TCP/IP suite can be viewed as shown in Figure 1.



Internet Protocol Suite - Architecture

The link protocols provide the delivery of packets between adjacent nodes in an internetwork; this delivery only implies that delivered information is intact, not that it has arrived in any particular order or that delivery guarantees are implied. The internetwork layer is responsible for the addressing and routing of packets between source and destination nodes within the network. Finally, the transport layer provides a range of services to transfer information between. As shown in Figure 1, the Internet applications make use of a combination of the services provided by TCP and/or UDP.

Transport Services

Two general classes of transport services are most often associated with the Internet protocol suite, the Transmission Control Protocol (TCP), and the User Datagram Protocol (UDP). Both of these protocols provide multiplexing services that permit the transmission of information to any one of a number of applications residing on the addressed hosts. UDP provides an inherently unreliable packet transfer service, and TCP is a more full-featured protocol, providing reliable delivery of an information stream.

UDP, as its name suggests, is a datagram protocol that will make a single attempt to transfer every application packet to the appropriate destination, if the network is operating efficiently, most packets will reach their destination. UDP is used by

application protocols such as the **Simple Network Management Protocol** that supports network management, the Trivial File Transfer Protocol. The UDP Header is quite simple, specifying only the length of the datagram and identifying the source and destination ports that are involved in the conversation.

TCP has been developed to support applications requiring reliable, ordered delivery of information between two participating network entities. Unlike UDP's packet orientation, TCP is a byte oriented stream protocol, with all information acknowledgments being based on acknowledgment of individual bytes in the stream rather than complete packets. The protocol has been designed to automatically adjust to differences in the communication channels that exist between the two processes participating in a conversation over the network. A sliding window acknowledgment scheme controls the orderly delivery of information presented to the TCP stream. Retry timers are based entirely on route turnaround times, with exponential increases being applied to the time applied to each subsequent retransmission of information.

Network Services

The Internet Protocol network layer provides the smarts required to identify the computers and sub-networks that receive and transmit packets. Services provided within the network layer include packet addressing, routing of traffic between systems. Protocols that are important in providing these services include the Internet Protocol (the IP of TCP/IP), a series of routing support protocols that help in identifying the most appropriate route for each packet in the network, and additional control protocols. The current addressing scheme provided by IP makes use of a 32-bit address field that can be divided into up to four eight bit fields that are used to describe the network a host is participating in, with a local address that specifies the address of the specific unit.

Three types of direct station address formats are supported by IP, class A, in which the high-order 8 (actually seven) bits are used to describe the network, class B, in which the high-order 16 bits defining the network, and the low order 16 defining the host, and class C addresses, in which the high-order 21 bits define the sub-network, and the low order 8 bits define the specific hosts (machines) on the local network. The type of address is defined by the high-order bits in the address. A zero in the highest order bit indicates Class A addressing; a 0x10 in the high-order two bits indicates that the address is a class B address. Class C addresses is defined as having a 0x110 in the high-order three bits of the address. The next generation Internet protocols, still under active discussion, have addressed the limitations of these addressing schemes through the definition of address spaces that can be scaled as the network size grows.

In addition to addressing and delivery of packets, IP provides fragmentation and reassembly services. Each link in the network has a parameter termed the Maximum Transmission Unit (MTU) that represents the largest payload that can be carried. When upper layer packets exceed these values, the IP includes services that split each packet into two or more packets that are subsequently routed through the network. Reassembly of the packets can occur either within a local network, or more typically can be reassembled at the receiving host. Timing information is transferred with each IP packet to permit intermediate and end nodes on a route to identify and discard stale packets.

With IP providing only a small set of addressing, fragmentation, and reassembly services, additional route and link management services are required. For error reporting and fault diagnosis, the Internet Control Message Protocol (**ICMP**) is frequently used.

Packet routing through the Internet is handled through a variety of techniques. In the simplest case, a system with no routing services directs all traffic through a single gateway that in turn is responsible for determining the most appropriate routes. A key distinction in the definition of routing services is between autonomous networks attaching to the network through a single access point, and internal network routing that can involve several routing choices. These routing protocols are grouped into the External Gateway Protocols (**EGP**) for attaching autonomous networks to the Internet, and Internal Gateway Protocols that manage more complicated routing decisions within a network.

The EGP is used to exchange routing information between autonomous systems. The routing lists consist of a list of sub-networks that can be reached through the reporting gateway. These reports are limited to only those sub-networks and hosts that the gateway provides direct service to (its own autonomous network) and not those destinations made available by the peer gateways.

For more sophisticated routing services, the Internal Gateway Protocols actively compute reachability and cost information to gain access to other systems on the network. Two general algorithms are used to perform this service, a **distance vector algorithm, and a shortest path first (SPF) algorithm**.

In a distance vector based system, each routing node recalculates and reports cost information to reach the destination nodes, paring its tree to some predetermined number of hops. The Route Information Protocol (RIP) is based on the simpler distance vector algorithm. In an effort to provide additional standardization and a

more robust set of services, the Open SPF (OSPF) protocol has been proposed in a separate set of Requests For Comment (RFC).

Link Services

At the link-level, the Internet protocols need only provide delivery of completed packets. While reliable service can be implemented, it is not necessary, and in some cases could actually impede the performance of TCP/IP based systems with additional retransmissions. Where reliability is required, TCP based services are generally used. Standards for exchange of Internet traffic have been developed for a large number of physical links including Ethernet, serial lines, Token Ring LANs, ATM, ISDN and others. The Point To Point protocol is often used to support the exchange of IP packets between interconnected nodes. In general the link protocol services include mechanisms to perform the following:

- **Intact delivery** - all information that is delivered is as intended by the sending station.
- **Address binding** - services are usually provided to map physical connections (such as Ethernet addresses, phone numbers, stations...) with the Internet address. One of the better known mechanisms for address binding is the Address Resolution Protocol (ARP) that maps IP addresses to Ethernet addresses.
- **Security** - some of the link protocols include access authorization mechanisms. This is particularly important in the case of dial-in protocols, where some form of protection is often required to protect critical network resources against unauthorized use.

For effective communication, the link services should provide a reasonable level of reliability. While the upper layer protocols have been designed to operate under a range of error environments, frequent errors can result in annoying, and sometimes unpredictable system performance.

Boot Services

Several boot service protocols have been developed to help new nodes learn their role in a network, and to inform the network of their presence. In many cases, such as dial-up protocols, diskless workstations, and embedded systems, a node entering a TCP/IP network may not have a pre-assigned network address. Some of the more popular boot protocols include:

- **Reverse Address Resolution Protocol (RARP)** - one of the earlier booting protocols, RARP allows a station that knows its own physical ethernet address to request an IP address from a RARP server. This protocol is still in use, with its popularity on a decline due to the fact that its services are limited to ethernet equipped devices, and no standard mechanism for defining a boot file image.

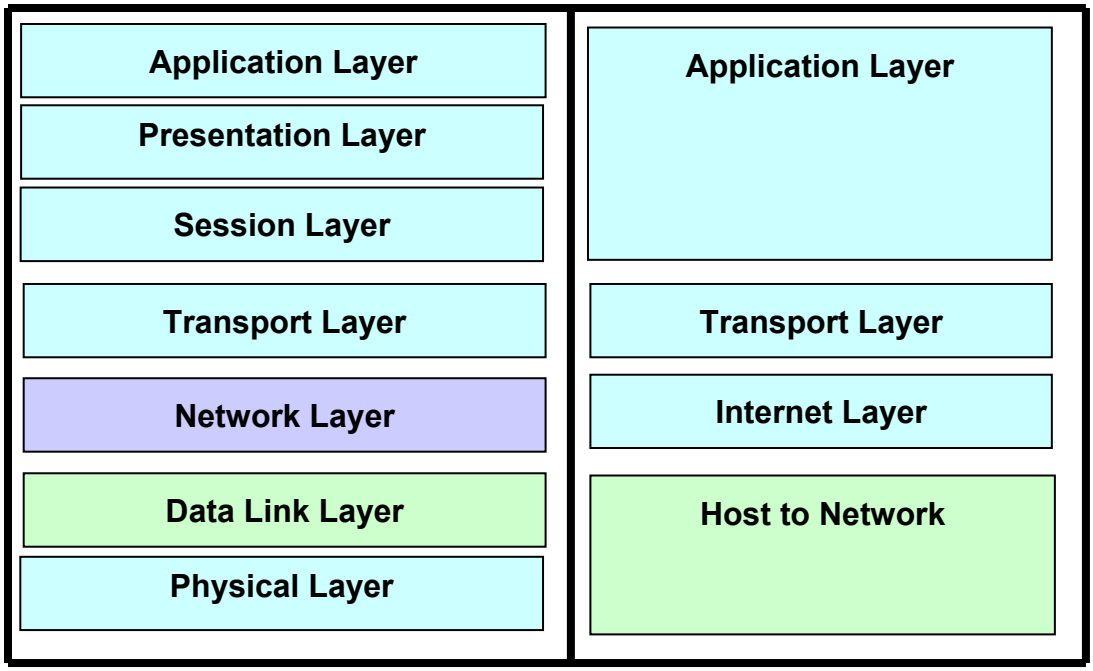
- **BOOTP -** addresses the limitations of RARP, providing address discovery to systems not containing an Ethernet port, and provides a mechanism for the BOOTP server to provide the diskless equipment with a name for the bootstrap code image. It is up to the client equipment to know how to process the received boot image name. Most typically, the image is downloaded through a file transfer protocol such as the Trivial File Transfer Protocol (TFTP). Also, unlike RARP, BOOTP relies on UDP and IP for the transfer of its requests. This permits operation in installations that may include several local subnets with a single centralized BOOTP server.

- The Dynamic Host Configuration Protocol (**DHCP**) further extends the services provided by BOOTP to permit the negotiation and transfer of operating parameters to the booting client system.

Application Transport Selections

The Internet applications typically use either UDP or TCP, with the rationale for the use of the different protocols depending on the nature of the application. Where reliable information with a specific connection context is required, TCP is the preferred protocol. Examples of application protocols that use UDP are Telnet, Rlogin, and the File Transfer Protocol. These application protocols have been designed to leverage the reliable connection-oriented TCP services with enhancements to support the respective applications. The reasons for using UDP range from simplicity of implementation to predictable effect on network operations. Protocols such as TFTP when combined with UDP; being simple to implement in constrained code spaces as may be found in diskless systems. In the case of network management protocols such as SNMP, UDP can be used without running the risk of piling additional traffic onto networks that may already be stuck in congested states.

5.4 TCP/IP VERSUS OSI



Comparison between OSI and TCP/IP

This topic gives a brief comparison between OSI and TCP/IP protocols with a special focus on the similarities and on how the protocols from both worlds map to each other. The adoption of TCP/IP does not conflict with the OSI standards because the two protocol stacks were developed concurrently. In some ways, TCP/IP contributed to OSI, and vice-versa. Several important differences do exist, though, which arise from the basic requirements of TCP/IP which are:

- A common set of applications
- Dynamic routing
- Connectionless protocols at the networking level
- Universal connectivity
- Packet-switching

The main differences between the OSI architecture and that of TCP/IP relate to the layers above the transport layer (layer 4) and those at the network layer (layer 3). OSI has both, the session layer and the presentation layer, whereas TCP/IP combines both into an application layer. The requirement for a connectionless protocol also required TCP/IP to combine OSI's physical layer and data link layer into a network level.

Physical Layer

The physical layer may be Ethernet, SDH-DCC, or some timeslot of a PDH (**Plesiochronous Digital Hierarchy)** signal.

Either OSI protocols and TCP/IP protocols build on the same physical layer standards, thus there is no difference between OSI and TCP/IP in this aspect.

Data Link Layer

The purpose of the data link layer is to provide error free data transmission even on noisy links. This is achieved by framing of data and retransmission of every frame until it is acknowledged from the far end, using flow control mechanisms. Error detection is done by means of error detection codes.

The data link layer in the OSI world makes use of the Q.921 LapD protocol which must support an information field length of at least 512 octets according to G.784. LapD is based on HDLC framing.

In the internet world there is no real data link layer protocol, but the subnet protocol which has quite many similarities. The subnet protocol consists of the IMP-IMP protocol which aims to provide a reliable connection between neighbored IMPs.

For Ethernet based networks e.g. LANs (Local Area Network), the data link protocol LLC (Logical Link Control) is equally used in OSI and TCP/IP networks.

Network Layer

The network layer provides routing capabilities between source and destination system.

OSI uses the CLNS (Connectionless Network Service) protocols ES-IS for communication of an end system to an intermediate system and IS-IS for communication between intermediate systems.

TCP divides messages in datagrams of up to 64k length. Each datagram consists of a header and a text part. Besides some other information, the header contains the source and the destination address of the datagram. IP routes these datagrams through the network using e.g. the protocol OSPF (Open Shortest Path First) or RIP (Route Information Protocol) for path calculation purposes. The service provided by IP is not reliable. Datagrams may be received in the wrong order or they may even get lost in the network.

Transport Layer

The transport layer provides a reliable end-to-end connection between source and destination system on top of the network layer. It builds an integral part of the whole OSI layering principle and of the internet protocol.

The OSI transport layer protocol (TP4) and the internet transport protocol (TCP) have many similarities but also some remarkable differences. Both protocols are built to provide a reliable connection-oriented end-to-end transport service on top of an unreliable network service. The network service may lose packets, store them, deliver them in the wrong order or even duplicate packets. Both protocols have to be able to deal with the most severe problems e.g. sub-network stores valid packets and send them at a later date. TP4 and TCP have connect, transfer and a disconnect phase. The principles of doing this are also quite similar. One difference between TP4 and TCP to be mentioned is that TP4 uses nine different TPDU (Transport Protocol Data Unit) types whereas TCP knows only one. This makes TCP simpler but every TCP header has to have all possible fields and therefore the TCP header is at least 20 bytes long whereas the TP4 header takes at least 5 bytes.

Another difference is the way both protocols react in case of a call collision. TP4 opens two bi-directional connections between the TSAPs whereas TCP opens just one connection.

TP4 uses a different flow control mechanism for its messages; it also provides means for quality of service measurement.

The differences are given in the table format as under.

Parameter	**O.S.I. Model**	**T.C.P./I.P.**
Expand the acronym	Open System Interconnect	Transmission Control Protocol / Internet Protocol
No. of layers	7	4
Protocols	Good as a model. The protocols are not very popular	The model is just description of protocols. Not so good as a model but protocols are more useful
Orientation	Both connection oriented and connection less in the Network Layer Only connection oriented in the transport Layer	Only connectionless in the Network layer Supports both (connection oriented and connectionless) in the transport layer

Services	OSI differentiates clearly between specification and the implementations O.S.I. Made the distinction between the following concepts explicitly:	Does not clearly distinguish the concepts of Service, interface and protocol
	(1) Services (2) Interface (3) Protocols	
Suitability	More general protocols	Only for TCP/IP protocols Cannot describe "Blue tooth"
Physical layer	Data Link & Physical are separate	Doesn't even mention about these
Top layers merged	Separate Application, Presentation and session layers	TCP/IP does not have separate Session and Presentation Layer. It is a part of Application Layer

UNIT 3
Physical Layer

6

TRANSMISSION MEDIUM

Unit Structure

6.1. Transmission Media
- 6.1.1. Guided Media
  - 6.1.1.1. Twisted Pair
  - 6.1.1.2. Coaxial Cable
  - 6.1.1.3. Fiber Optics
- 6.2.2. Unguided Media

6.2. Wireless transmission
6.3. Electromagnetic Spectrum
6.4. Radio Transmission
6.5. Microwave transmission
6.6. Infra red waves
- 6.6.1. Millimeter waves
- 6.6.2. Applications of millimeter waves

6.7. Light wave transmission

6.1. TRANSMISSION MEDIA:

Any data communication is in need of three components

1) Media
2) Data
3) Transmitter and the receiver

The data generated by the transmitter is carried by the medium to the receiver. This medium can be classified into two categories.

1. Guided media
2. Unguided media

6.1.1. Guided media:

Guided Transmission Media uses a "cabling" system that guides the data signals along a specific path. The data signals are bound by the "cabling" system. Guided Media is also known as

Bound Media. Cabling is meant in a generic sense in the previous sentences and is not meant to be interpreted as copper wire cabling only.

Unguided Transmission Media consists of a means for the data signals to travel but nothing to guide them along a specific path. The data signals are not bound to a cabling media and as such are often called Unbound Media.

There 4 basic types of Guided Media:
Open Wire
Twisted Pair
Coaxial Cable
Optical Fiber

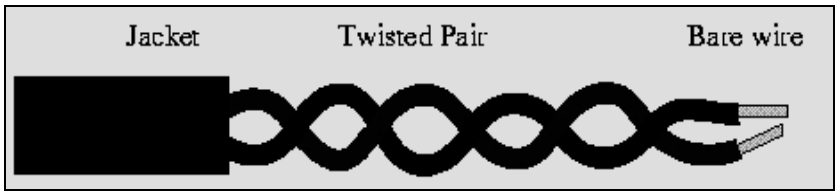
6.1.1.1. Twisted pair:

Twisted pair cable consists of a pair of insulated wires twisted together. It is a cable type used in telecommunication for very long time. Cable twisting helps to reduce noise pickup from outside sources and crosstalk on multi-pair cables.

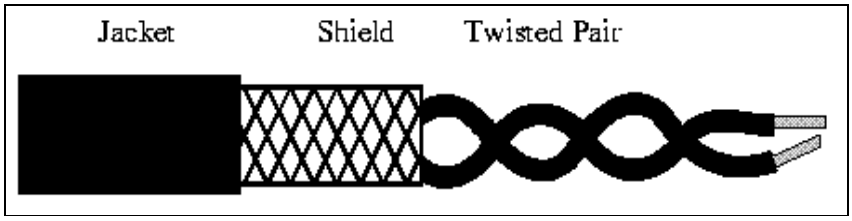
Shielded Twisted Pair Cable is used to eliminate inductive and capacitive coupling. Twisting cancels out inductive coupling, while the shield eliminates capacitive coupling. Most applications for this cable are between equipment, racks and buildings. Shielding adds usually some attenuation to the cable (compared to unshielded), but usually not because in the case of balanced transmission, the complementing signals will effectively cancel out any shield currents, so shield current losses are negligible.

The noise pickup characteristics of twisted pair cable is determined by the following cable characteristics: number of twists per meter (generally more twists per meter gives better performance), uniform cable construction, capacitance balance (less capacitance difference to ground, the better), cable diameter (less are between wires is better) and the amount of shielding (more shielding, the better).

The wires in Twisted Pair cabling are twisted together in pairs. Each pair would consist of a wire used for the +ve data signal and a wire used for the -ve data signal. Any noise that appears on 1 wire of the pair would occur on the other wire. Because the wires are opposite polarities, they are 180 degrees out of phase (180 degrees - phase definition of opposite polarity). When the noise appears on both wires, it cancels or nulls itself out at the receiving end. Twisted Pair cables are most effectively used in systems that use a balanced line method of transmission: polar line coding (Manchester Encoding) as opposed to unipolar line coding (TTL logic).



The degree of reduction in noise interference is determined specifically by the number of turns per foot. Increasing the number of turns per foot reduces the noise interference. To further improve noise rejection, a foil or wire braid shield is woven around the twisted pairs. This "shield" can be woven around individual pairs or around a multi-pair conductor (several pairs).



Cables with a shield are called Shielded Twisted Pair and commonly abbreviated STP. Cables without a shield are called Unshielded Twisted Pair or UTP. Twisting the wires together results in characteristic impedance for the cable. Typical impedance for UTP is 100 ohm for Ethernet 10BaseT cable

Twisted pair cable is good for transferring balanced differential signals. The practice of transmitting signals differentially dates back to the early days of telegraph and radio. The advantages of improved signal-to-noise ratio, crosstalk, and ground bounce that balanced signal transmission are particularly valuable in wide bandwidth and high fidelity systems. By transmitting signals along with a 180 degree out-of-phase complement, emissions and ground currents are theoretically canceled. This eases the requirements on the ground and shield compared to single ended transmission and results in improved EMI performance.

The most commonly used form of twisted pair is unshielded twisted pair (UTP). It is just two insulated wires twisted together. Any data communication cables and normal telephone cables are this type. Shielded twisted pair (STP) differs from UTP in that it has a foil jacket that helps prevent crosstalk and noise from outside source. In data communications there is a cable type called FTP (foil shielded pairs) which consists of four twisted pair inside one common shield (made of aluminum foil).

When cable twisted at constant twist rate over the length of the cable, a cable with well defined characteristic impedance is formed. Characteristic impedance of twisted pair is determined by

the size and spacing of the conductors and the type of dielectric used between them. Balanced pair, or twin lines, has a Zo which depends on the ratio of the wire spacing to wire diameter and the foregoing remarks still apply. For practical lines, Zo at high frequencies is very nearly, but not exactly, a pure resistance. Because the impedance of a cable is actually a function of the spacing of the conductors, so separating the conductors significantly changes the cable impedance at that point.

When many twisted pairs are put together to form a multi-pair cable, individual conductors are twisted into pairs with varying twists to minimize crosstalk. Specified color combinations to provide pair identification.

The most commonly used twisted pair cable impedance is 100 ohms. It is widely used for data communications and telecommunications applications in structured cabling systems. In most twisted pair cable applications the cable impedance is between 100 ohms and 150 ohms. When a cable has a long distance between the conductors, higher impedances are possible. Typical wire conductor sizes for cables used in telecommunications 26, 24, 22 or 19 AWG.

Here is some common impedance related to twisted pair lines:

- 100 ohms: This impedance is the standardized impedance to be used in the twisted pair wiring used in structured wiring systems standardized EIA/TIA 568 standard. Both unshielded and shielded "CAT5 and better" cables used on this kind of applications have 100 ohms impedance (usually at +-15% or better accuracy). Nowadays the most common LAN standard, Ethernet, is designed for 100 ohms twisted pair cable. Many telecommunication twisted pair cables have impedance of around 100 ohms, and many modern digital communication system are matched to this impedance. Nowadays practically all modern in-building twisted pair wiring for telecom applications has 100 ohms impedance.

- 110 ohms: 110 ohms shielded twisted pair cable is standardized as the cable type to be used for digital AES/EBU sound interface.

- 120 ohms: 120 ohms shielded cable is generally used for for RS485 communications in industrial networking. There are many industrial "control and data" cables which have impedance of around 120 ohms. Also some telecom cables (both shielded and unshielded) have impedance of 120 ohms, and there are digital telecom systems matched to this impedance also (for example some E1 systems).

- 150 ohms: This was the impedance used in shielded twisted pair wiring IBM cabling system and Token Ring network. There are also many shielded "control and data" cables that have impedance of around 150 ohms in use nowadays. Some modern microphone cabling (shielded twisted pair) has impedance of around 150 ohms at high frequencies and you can sometimes hear 150 ohms impedance mentioned in analogue audio applications (typical dynamic professional microphones have impedance of 150-200 ohms usually).

- 300 ohms: The twin lead wire used in some antenna applications has impedance of 300 ohms. This is a very low loss antenna cable type. 300 ohms is generally not used for anything else than some antenna applications.

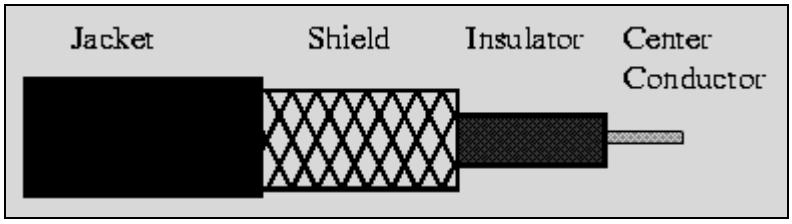
- 600 ohms: 600 ohms is standardized impedance used in telephone world. The first long telephone air lines (two wires on the poles separated from each other at some distance) used to have impedance of around 600 ohms. In practice the modern telephone cables do not have impedance of 600 ohms, but for historical reasons this impedance is spoken often and many telephone equipment are still matched to this impedance. You can sometimes (quite rarely nowadays) hear 600 ohms matching also in audio world.

6.1.1.2. Coaxial cable:

Introduction:

A coaxial cable is one that consists of two conductors that share a common axis. The inner conductor is typically a straight wire, either solid or stranded and the outer conductor is typically a shield that might be braided or a foil.

Coaxial Cable consists of 2 conductors. The inner conductor is held inside an insulator with the other conductor woven around it providing a shield. An insulating protective coating called a jacket covers the outer conductor.



The outer shield protects the inner conductor from outside electrical signals. The distance between the outer conductor (shield) and inner conductor plus the type of material used for

insulating the inner conductor determine the cable properties or impedance. Typical impedances for coaxial cables are 75 ohms for Cable TV, 50 ohms for Ethernet Thinnet and Thicknet. The excellent control of the impedance characteristics of the cable allow higher data rates to be transferred than Twisted Pair cable.

Coaxial cable is a cable type used to carry radio signals, video signals, measurement signals and data signals. Coaxial cables exist because we can't run open-wire line near metallic objects (such as ducting) or bury it. We trade signal loss for convenience and flexibility. Coaxial cable consists of an insulated center conductor which is covered with a shield. The signal is carried between the cable shield and the center conductor. This arrangement give quite good shielding against noise from outside cable, keeps the signal well inside the cable and keeps cable characteristics stable.

Coaxial cables and systems connected to them are not ideal. There is always some signal radiating from coaxial cable. Hence, the outer conductor also functions as a shield to reduce coupling of the signal into adjacent wiring. More shield coverage means less radiation of energy (but it does not necessarily mean less signal attenuation).

Coaxial cables are typically characterized with the impedance and cable loss. The length has nothing to do with coaxial cable impedance. Characteristic impedance is determined by the size and spacing of the conductors and the type of dielectric used between them. For ordinary coaxial cable used at reasonable frequency, the characteristic impedance depends on the dimensions of the inner and outer conductors. The characteristic impedance of a cable (Zo) is determined by the formula 138 log b/a, where b represents the inside diameter of the outer conductor (read: shield or braid), and a represents the outside diameter of the inner conductor.

Most common coaxial cable impedances in use in various applications are 50 ohms and 75 ohms. 50 ohms cable is used in radio transmitter antenna connections, many measurement devices and in data communications (Ethernet). 75 ohms coaxial cable is used to carry video signals, TV antenna signals and digital audio signals. There are also other impedances in use in some special applications (for example 93 ohms). It is possible to build cables at other impedances, but those mentioned earlier are the standard ones that are easy to get. It is usually no point in trying to get something very little different for some marginal benefit, because standard cables are easy to get, cheap and generally very good. Different impedances have different characteristics. For maximum power handling, somewhere between 30 and 44 Ohms is the

optimum. Impedance somewhere around 77 Ohms gives the lowest loss in a dielectric filled line. 93 Ohms cable gives low capacitance per foot. It is practically very hard to find any coaxial cables with impedance much higher than that.

Here is a quick overview of common coaxial cable impedances and their main uses:

- 50 ohms: 50 ohms coaxial cable is very widely used with radio transmitter applications. It is used here because it matches nicely to many common transmitter antenna types, can quite easily handle high transmitter power and is traditionally used in this type of applications (transmitters are generally matched to 50 ohms impedance). In addition to this 50 ohm coaxial cable can be found on coaxial Ethernet networks, electronics laboratory interconnection (foe example high frequency oscilloscope probe cables) and high frequency digital applications (for example ECL and PECL logic matches nicely to 50 ohms cable). Commonly used 50 Ohm constructions include RG-8 and RG-58.

- 60 Ohms: Europe chose 60 ohms for radio applications around 1950s. It was used in both transmitting applications and antenna networks. The use of this cable has been pretty much phased out, and nowadays RF system in Europe uses either 50 ohms or 75 ohms cable depending on the application.

- 75 ohms: The characteristic impedance 75 ohms is an international standard, based on optimizing the design of long distance coaxial cables. 75 ohms video cable is the coaxial cable type widely used in video, audio and telecommunications applications. Generally all base band video applications that use coaxial cable (both analogue and digital) are matched for 75 ohm impedance cable. Also RF video signal systems like antenna signal distribution networks in houses and cable TV systems are built from 75 ohms coaxial cable (those applications use very low loss cable types). In audio world digital audio (S/PDIF and coaxial AES/EBU) uses 75 ohms coaxial cable, as well as radio receiver connections at home and in car. In addition to this some telecom applications (for example some E1 links) use 75 ohms coaxial cable. 75 Ohms is the telecommunications standard, because in a dielectric filled line, somewhere around 77 Ohms gives the lowest loss. For 75 Ohm use common cables are RG-6, RG-11 and RG-59.

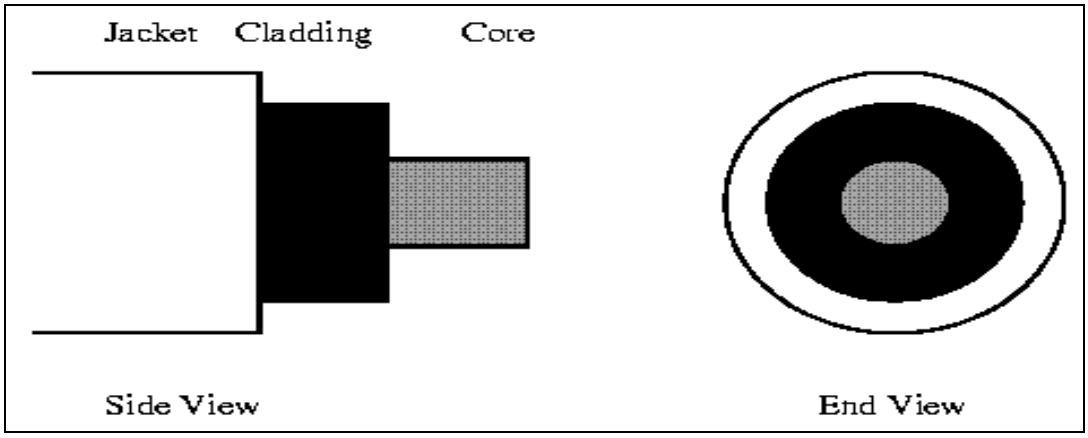
- 93 Ohms: This is not much used nowadays. 93 ohms was once used for short runs such as the connection between

computers and their monitors because of low capacitance per foot which would reduce the loading on circuits and allow longer cable runs. In addition this was used in some digital communication systems (IBM 3270 terminal networks) and some early LAN systems.

6.1.1.3. Fiber optics

Optical Fibre

Optical Fibre consists of thin glass fibres that can carry information at frequencies in the visible light spectrum and beyond. The typical optical fibre consists of a very narrow strand of glass called the Core. Around the Core is a concentric layer of glass called the Cladding. A typical Core diameter is 62.5 microns (1 micron = 10-6 meters). Typically Cladding has a diameter of 125 microns. Coating the cladding is a protective coating consisting of plastic, it is called the Jacket.



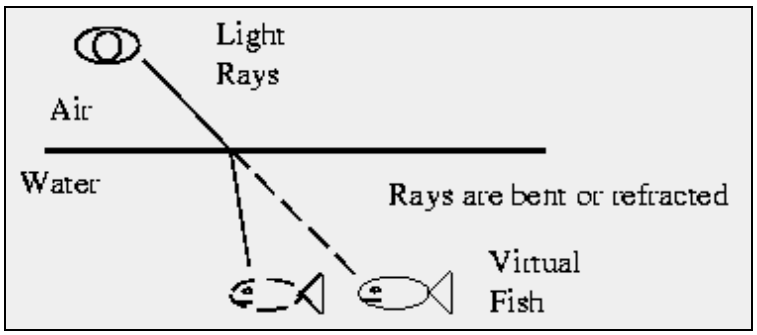
Important characteristic of Fibre Optics is Refraction. Refraction is the characteristic of a material to either pass or reflect light. When light passes through a medium, it "bends" as it passes from one medium to the other. An example of this is when we look into a pond of water.

If the angle of incidence is small, the light rays are reflected and do not pass into the water. If the angle of incident is great, light passes through the media but is bent or refracted.

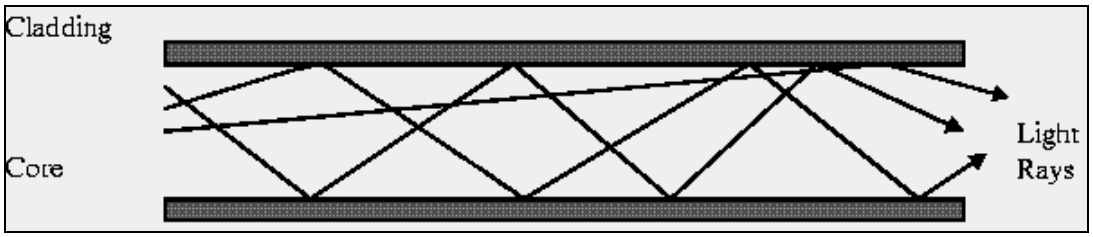
Optical Fibres work on the principle that the core refracts the light and the cladding reflects the light. The core refracts the light and guides the light along its path. The cladding reflects any light back into the core and stops light from escaping through it.

Optical Transmission Modes

There are 3 primary types of transmission modes using optical fibre.
They are
a) Step Index
b) Grade Index
c) Single Mode

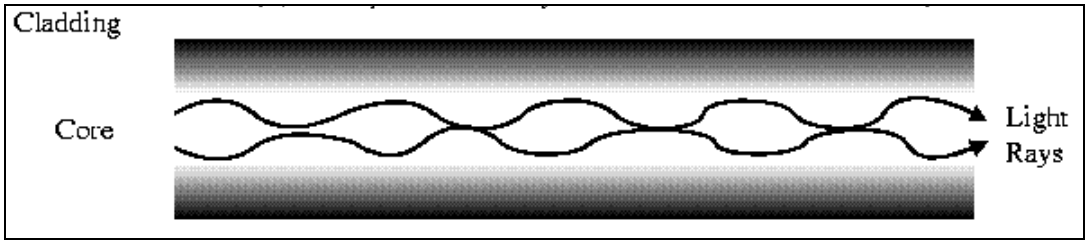


Step Index has a large core the light rays tend to bounce around, reflecting off the cladding, inside the core. This causes some rays to take a longer or shorted path through the core. Some take the direct path with hardly any reflections while others bounce back and forth taking a longer path. The result is that the light rays arrive at the receiver at different times. The signal becomes longer than the original signal. LED light sources are used. Typical Core: 62.5 microns.



Step Index Mode

Grade Index has a gradual change in the Core's Refractive Index. This causes the light rays to be gradually bent back into the core path. This is represented by a curved reflective path in the attached drawing. The result is a better receive signal than Step Index. LED light sources are used. Typical Core: 62.5 microns.



Grade Index Mode

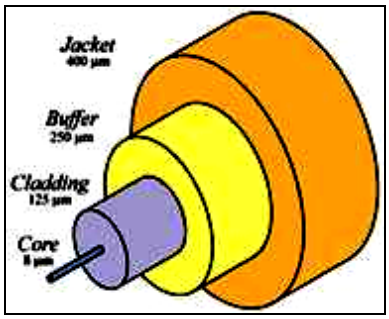
Single Mode has separate distinct Refractive Indexes for the cladding and core. The light ray passes through the core with relatively few reflections off the cladding. Single Mode is used for a single source of light (one color) operation. It requires a laser and the core is very small: 9 microns.

Single Mode:

When the fiber core is so small that only light ray at 0° incident angle can stably pass through the length of fiber without much loss, this kind of fiber is called single mode fiber. The basic requirement for single mode fiber is that the core be small enough to restrict transmission to a singe mode. This lowest-order mode can propagate in all fibers with smaller cores (as long as light can physically enter the fiber).

The most common type of single mode fiber has a core diameter of 8 to 10 μm and is designed for use in the near infrared (the most common are 1310nm and 1550nm). Please note that the mode structure depends on the wavelength of the light used, so that this fiber actually supports a small number of additional modes at visible wavelengths. Multi mode fiber, by comparison, is manufactured with core diameters as small as 50um and as large as hundreds of microns.

The following picture shows the fiber structure of a single mode fiber.



Comparison among the three different models:

Factor	UTP	Twisted Pair	Coaxial Cable	Fiber Optic
Cost	Lowest	Moderate	Moderate	Highest
Installation	Easy	Fairly easy	Fairly easy	Difficult
Bandwidth	Typically 10 Mbps	Typically 10 Mbps	Typically 16 Mbps	2 Gbps
Attenuation	High (Range few hundred meters)	High (Range few hundred meters)	Lower (Range of a few kilo meters)	Lowest (Range of tens of kilometers)
Electromagnetic interference (EMI)	Most vulnerable to EMI	Less vulnerable that UTP	Less vulnerable that UTP	Not effected by EMI

6.2. WIRELESS TRANSMISSION:

The wireless transmission is the sending and receiving of the data packets over the distance without the use of wires. The wireless network transmission is the ideal for the locations where the physical mediums like coaxial cables, UPT/STP and fiber optic is not possible to deploy. The demand of the wireless communications is increasing exponentially.

The wireless transmission is the sending and receiving of the data packets over the distance without the use of wires. The wireless network transmission is the ideal for the locations where the physical mediums like coaxial cables, UPT/STP and fiber optic is not possible to deploy. The demand of the wireless communications is increasing exponentially.

The wireless communication can be performed in a variety of ways such as wireless Ethernet, GSM, Bluetooth, Infrared, Wi-Fi and Wi-Max. Similarly the broadband wireless is an emerging

wireless technology that allows the simultaneous delivery of the voice, video and data signals.

All these technologies based on the different standards and specifications. The wireless communication uses the radio and electromagnetic waves to transmit the data. Most of the wireless transmission standards are based on the 802.11 specifications.

Common examples of wireless equipment in use today include:

- Cellular phones and pagers -- provide connectivity for portable and mobile applications, both personal and business
- Global Positioning System (GPS) -- allows drivers of cars and trucks, captains of boats and ships, and pilots of aircraft to ascertain their location anywhere on earth
- Cordless computer peripherals -- the cordless mouse is a common example; keyboards and printers can also be linked to a computer via wireless
- Cordless telephone sets -- these are limited-range devices, not to be confused with cell phones
- Home-entertainment-system control boxes -- the VCR control and the TV channel control are the most common examples; some hi-fi sound systems and FM broadcast receivers also use this technology
- Remote garage-door openers -- one of the oldest wireless devices in common use by consumers; usually operates at radio frequencies
- Two-way radios -- this includes Amateur and Citizens Radio Service, as well as business, marine, and military communications
- Baby monitors -- these devices are simplified radio transmitter/receiver units with limited range
- Satellite television -- allows viewers in almost any location to select from hundreds of channels
- Wireless LANs or local area networks -- provide flexibility and reliability for business computer users

6.3. ELECTROMAGNETIC SPECTRUM:

The **electromagnetic spectrum** is the range of all possible frequencies of electromagnetic radiation. The "electromagnetic spectrum" of an object is the characteristic distribution of electromagnetic radiation emitted or absorbed by that particular object.

The electromagnetic spectrum extends from low frequencies used for modern radio to gamma radiation at the short-wavelength end, covering wavelengths from thousands of kilometers down to a fraction of the size of an atom. The long wavelength limit is the size of the universe itself, while it is thought that the short wavelength limit is in the vicinity of the Planck length, although in principle the spectrum is infinite and continuous.

EM waves are typically described by any of the following three physical properties:

Frequency *f*,
Wavelength λ, or
Photon energy *E*.

Electromagnetic radiation interacts with matter in different ways in different parts of the spectrum. The types of interaction can be so different that it seems to be justified to refer to different types of radiation. At the same time, there is a continuum containing all these "different kinds" of electromagnetic radiation. Thus we refer to a spectrum, but divide it up based on the different interactions with matter.

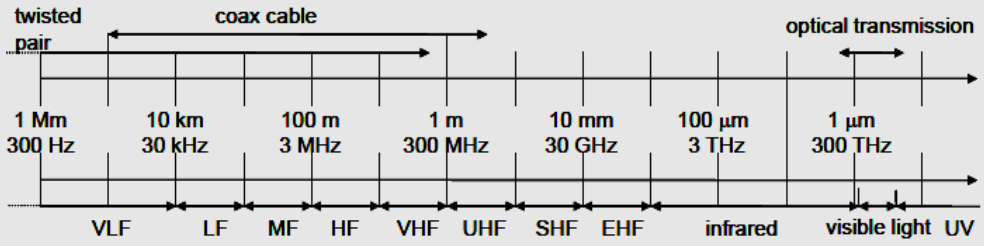
Region of the spectrum	**Main interactions with matter**
Radio	Collective oscillation of charge carriers in bulk material (plasma oscillation). An example would be the oscillation of the electrons in an antenna.
Microwave through far infrared	Plasma oscillation, molecular rotation
Near infrared	Molecular vibration, plasma oscillation (in metals only)
Visible	Molecular electron excitation (including pigment molecules found in the human retina), plasma oscillations (in metals only)
Ultraviolet	Excitation of molecular and atomic valence electrons, including ejection of the electrons (photoelectric effect)
X-rays	Excitation and ejection of core atomic electrons, Compton scattering (for low atomic numbers)

Gamma rays	Energetic ejection of core electrons in heavy elements, Compton scattering (for all atomic numbers), excitation of atomic nuclei, including dissociation of nuclei
High energy gamma rays	Creation of particle-antiparticle pairs. At very high energies a single photon can create a shower of high energy particles and antiparticles upon interaction with matter.

Frequencies for communication

- VLF = Very Low Frequency
- LF = Low Frequency
- MF = Medium Frequency
- HF = High Frequency
- VHF = Very High Frequency
- UHF = Ultra High Frequency
- SHF = Super High Frequency
- EHF = Extra High Frequency
- UV = Ultraviolet Light

- Frequency and wave length
  - $\lambda = c/f$
  - wave length λ, speed of light $c \cong 3x10^8 m/s$, frequency f



6.4. RADIO TRANSMISSION:

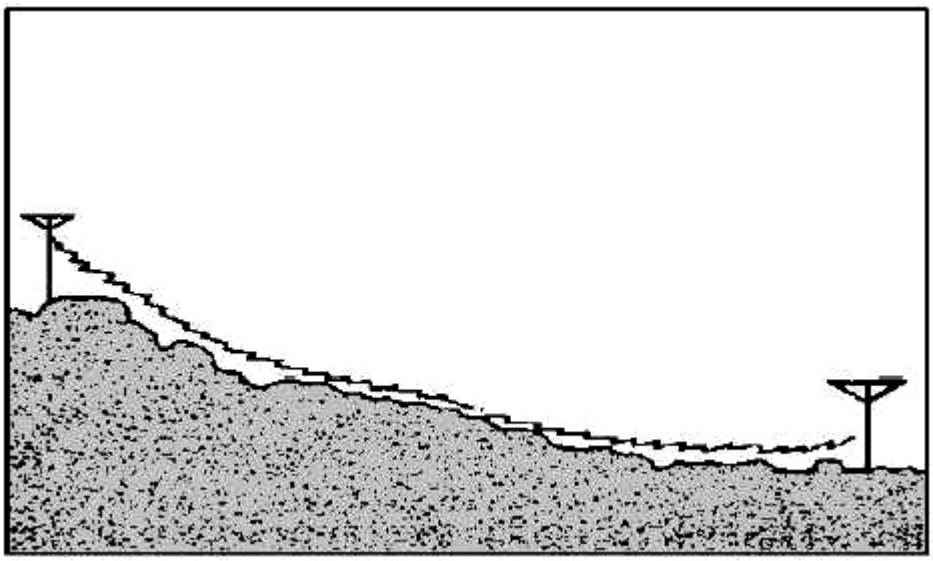
Radio waves are easy to generate, can travel long distances, and can penetrate buildings easily, so they are widely used for communication, both indoors and outdoors. Radio waves also are omnidirectional, meaning that they travel in all directions from the source, so the transmitter and receiver do not have to be carefully aligned physically.

There are two principal ways in which electromagnetic (radio) energy travels from a transmitting antenna to a receiving antenna. One way is by GROUND WAVES and the other is by SKY WAVES. Ground waves are radio waves that travel near the surface of the Earth (surface and space waves). Sky waves are radio waves that are reflected back to Earth from the ionosphere.

Ground Waves The ground wave is actually composed of two separate component waves. These are known as the SURFACE WAVE and the SPACE WAVE. The determining factor in whether a ground wave component is classified as a space wave or a surface wave is simple. A surface wave travels along the surface of the Earth. A space wave travels over the surface.

SURFACE WAVE:

The surface wave reaches the receiving site by traveling along the surface of the ground as shown in the following figure. A surface wave can follow the contours of the Earth because of the process of diffraction. When a surface wave meets an object and the dimensions of the object do not exceed its wavelength, the wave tends to curve or bend around the object. The smaller the object, the more pronounced the diffractive action will be.

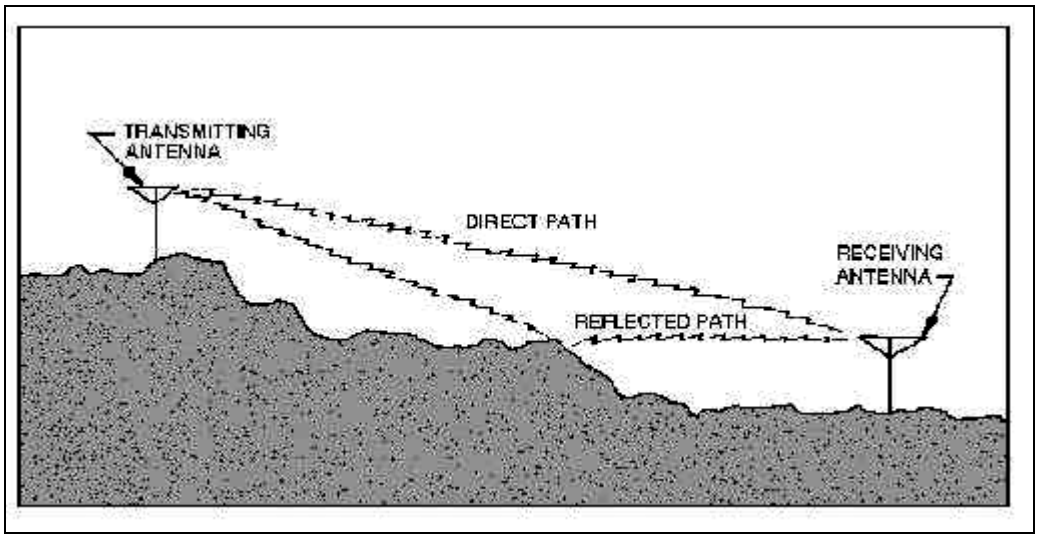


As a surface wave passes over the ground, the wave induces a voltage in the Earth. The induced voltage takes energy away from the surface wave, thereby weakening, or attenuating, the wave as it moves away from the transmitting antenna. To reduce the attenuation, the amount of induced voltage must be reduced. This is done by using vertically polarized waves that minimize the extent to which the electric field of the wave is in contact with the Earth. When a surface wave is horizontally polarized, the electric field of the wave is parallel with the surface of the Earth and, therefore, is constantly in contact with it. The wave is then completely attenuated within a short distance from the transmitting site. On the other hand, when the surface wave is vertically polarized, the electric field is vertical to the Earth and merely dips into and out of the Earth's surface. For this reason, vertical polarization is vastly superior to horizontal polarization for surface wave propagation.

The attenuation that a surface wave undergoes because of induced voltage also depends on the electrical properties of the terrain over which the wave travels. The best type of surface is one that has good electrical conductivity. The better the conductivity, the less the attenuation.

SPACE WAVE:

The space wave follows two distinct paths from the transmitting antenna to the receiving antenna—one through the air directly to the receiving antenna, the other reflected from the ground to the receiving antenna. The primary path of the space wave is directly from the transmitting antenna to the receiving antenna. So, the receiving antenna must be located within the radio horizon of the transmitting antenna. Because space waves are refracted slightly, even when propagated through the troposphere, the radio horizon is actually about one-third farther than the line-of-sight or natural horizon.



Although space waves suffer little ground attenuation, they nevertheless are susceptible to fading. This is because space waves actually follow two paths of different lengths (direct path and ground reflected path) to the receiving site and, therefore, may arrive in or out of phase. If these two component waves are received in phase, the result is a reinforced or stronger signal. Likewise, if they are received out of phase, they tend to cancel one another, which results in a weak or fading signal.

6.5. MICROWAVE TRANSMISSION:

Microwave transmission also requires line of sight in order to work properly. In order to allow two way communications two frequencies are used. However, this does not mean that there has to be two antennas because the frequencies can be dealt with by one antenna at both ends.

The distance covered by microwave signals is based upon the height of the antenna. In order to increase this coverage each antenna has a built-in repeater that regenerates the signal before passing it on to the next antenna in line. The placement of the antenna to do this is approximately 25 miles.

Microwave transmission refers to the technology of transmitting information by the use of the radio waves whose wavelengths are conveniently measured in small numbers of centimeters, by using various electronic technologies. These are called ***microwaves***. This part of the radio spectrum ranges across frequencies of roughly 1.0 gigahertz (GHz) to 30 GHz. Also by using the formula **λ = c/f** , these correspond to wavelengths from 30 centimeters down to 1.0 cm. [In the above equation, the Greek letter λ (lambda) is the wavelength in meters; **c** is the speed of light in meters per second; and **f** is the frequency in hertz (Hz).]

In the microwave frequency band, antennas are usually of convenient sizes and shapes, and also the use of metal waveguides for carrying the radio power works well. Furthermore, with the use of the modern solid-state electronics and traveling wave tube technologies that have been developed since the early 1960s, the electronics used by microwave radio transmission have been readily used by expert electronics engineers.

Microwave radio transmission is commonly used by communication systems on the surface of the Earth, in satellite communications, and in deep space radio communications. Other parts of the microwave radio band are used for radars, radio navigation systems, sensor systems, and radio astronomy.

The main drawback of microwave signals is that they can be affected by weather, especially rain.

Microwave transmission relies on three key elements:

- Use of radio frequency to achieve the transmissions (operating between 1Ghz to 170Ghz)
- Clear line-of-sight with no obstacles in the way
- Regular relay stations required due to line of site and cost considerations

Advantages:

- No cables needed
- Multiple channels available
- Wide bandwidth

Disadvantages:

- Line-of-sight will be disrupted if any obstacle, such as new buildings, are in the way
- Signal absorption by the atmosphere. Microwaves suffer from attenuation due to atmospheric conditions.
- Towers are expensive to build.

6.6. INFRA RED WAVES:

Infrared light is a form of electromagnetic waves. Infrared can be visible light or can be in the form of microwaves. In its visible form, infrared can be seen as red or violet. Visible infrared waves are very short, measuring about 750 nm in length; longer infrared waves which are closest to microwaves are about 1mm in length, about the size of a microbial cell.

While shorter infrared waves can be seen as colors, as infrared waves get longer they give off radiation in the form of heat. Infrared waves are considered to be thermal. For instance, the sun gives off infrared wave lengths, your toaster gives off infrared wave lengths and even living organisms such as humans, dogs and cats give off thermal heat in the form of infrared wave lengths.

While most infrared wave lengths give off heat, you are probably aware of one of the most popular uses of infrared wave lengths- the TV remote control. Your remote control works via short infrared wave lengths, the visible kind and don't worry, these short wave lengths do not give off any heat.

Your TV remote is able to produce visible light waves, usually red that can be seen and distinguished by sensors in your TV set. Because infrared waves at the visible level are safe and easily seen and separated from other light waves, remote controls are very effective in controlling items.

Another effective use for infrared is the ability to see objects that give off infrared radiation. For instance, practically every object gives off thermal heat. Thermal heat is shown as infrared radiation. For instance, an oven gives off thermal heat, so does the skin of a human, and even an ice cream sundae has heat which can be seen via infrared radiation. As an object gets warmer, it gives off more infrared radiation. A toaster gives off far more radiation than an ice cube.

6.6.1. Millimeter waves:

The following table is indicating some frequency bands, exact frequencies and approximate wavelengths.

Band	Starts at	Ends at
VHF	30 MHz	300 MHz
-	10 meters	1 meter
UHF	300 MHz	3 GHz
-	1 meter	10 cm< /TR>
SHF	3 GHz	30 GHz
-	10 cm	1 cm
EHF	30 GHz	300 GHz
-	1 cm	1 mm
Sub Millimeter	300 GHz	Infinity
-	1 mm	zero

EHF is considered "Millimeter Waves" because the range of wavelengths is between 1 mm and 1 cm. Above 300 GHz is considered "Sub-millimeter waves" because the wavelength is below one millimeter. Frequencies above 1 THz (1000 GHz) are called "Terahertz frequencies" or "Very long wave infrared"

Various designators and names have been associated with the wavelengths between 300 GHz and 300 THz. They usually are called Infrared in one form or another, but as you can see, this range covers three orders of magnitude - or one thousand fold.

300 THz - or 1 um (one micrometer, or 1000 nano-meters, or 10,000 angstroms) is still infrared, because visual sensitivity begins around 480 THz or about 625 nm which is a very deep red color.

6.6.2. Applications of millimeter waves:

The millimeter-wave region of the electromagnetic spectrum is usually considered to be the range of wavelengths from 10 millimeters (0.4 inches) to 1 millimeter (0.04 inches). This means they are larger than infrared waves or x-rays, for example, but smaller than radio waves or microwaves. The millimeter-wave region of the electromagnetic spectrum corresponds to radio band frequencies of 30 GHz to 300 GHz and is sometimes called the Extremely High Frequency (EHF) range. The high frequency of millimeters waves as well as their propagation characteristics (that is, the ways they change or interact with the atmosphere as they travel) makes them useful for a variety of applications including transmitting large amounts of computer data, cellular communications, and radar.

One of the greatest and most important uses of millimeter waves is in transmitting large amounts of data. Every kind of wireless communication, such as the radio, cell phone, or satellite, uses specific range of wavelengths or frequencies. Each application provider (such as a local television or radio broadcaster) has a unique “channel” assignment, so that they can all communicate at the same time without interfering with each other. These channels have “bandwidths” (also measured in either wavelength or frequency) that must be large enough to pass the information from the broadcaster’s transmitter to the user. For example, a telephone conversation requires only about 6 kHz of bandwidth, while a TV broadcast, which carries much larger amounts of information, requires about 6 MHz. (A kilohertz, is one thousand cycles per second; a megahertz is one million cycles per second). Increases in the amount of information transmitted require the use of higher frequencies. This is where millimeter waves come in. Their high frequency makes them a very efficient way of sending large amounts of data such as computer data, or many simultaneous television or voice channels.

Radar is another important use of millimeter waves, which takes advantage of another important property of millimeter wave propagation called beam width. Beam width is a measure of how a transmitted beam spreads out as it gets farther from its point of origin. In radar, it is desirable to have a beam that stays narrow, rather than fanning out. Small beam widths are good in radar because they allow the radar to “see” small distant objects, much like a telescope. A carefully designed antenna allows microwaves to be focused into a narrow beam, just like a magnifying glass focuses sunlight unfortunately, small beam widths require large antenna sizes, which can make it difficult to design a good radar set that will fit, for example, inside a cramped airplane cockpit.

6.7. LIGHT WAVE TRANSMISSION:

Unguided optical signaling has been in use for centuries. Coherent optical signaling using lasers is inherently unidirectional, so each building needs its own laser and its own photo detector. This scheme offers very high bandwidth ad very low cost. It is also relatively easy install.

UNIT 4
Data Link Layer

7

DATA LINK LAYER

Unit Structure

7.1. Design Issues
- 7.1.1. Reliable Delivery:
- 7.1.2. Best Effort:
- 7.1.3. Acknowledged Delivery:
- 7.1.4. Framing
- 7.1.5. Length Count:
- 7.1.6. Bit Stuffing:
- 7.1.7. Character stuffing:
- 7.1.8. Encoding Violations:
- 7.1.9.Error Control

7.2. Error correction and detection
- 7.2.1. General strategy:
- 7.2.2. Error-correcting codes :
- 7.2.3. Hamming Code (1 bit error correction)
- 7.2.4. Error-detecting codes
- 7.2.5. Modulo 2 arithmetic
- 7.2.6. Error detection with CRC

7.1. DESIGN ISSUES:

The goal of the data link layer is to provide reliable, efficient communication between adjacent machines connected by a single communication channel. Specifically:

1. Group the physical layer bit stream into units called *frames*. Note that frames are nothing more than ``packets'' or ``messages''. By convention, we'll use the term ``frames'' when discussing DLL packets.
2. Sender checksums the frame and sends checksum together with data. The checksum allows the receiver to determine when a frame has been damaged in transit.

3. Receiver recomputes the checksum and compares it with the received value. If they differ, an error has occurred and the frame is discarded.
4. Perhaps return a *positive* or *negative acknowledgment* to the sender. A positive acknowledgment indicate the frame was received without errors, while a negative acknowledgment indicates the opposite.
5. Flow control. Prevent a fast sender from overwhelming a slower receiver. For example, a supercomputer can easily generate data faster than a PC can consume it.
6. In general, provide service to the network layer. The network layer wants to be able to send packets to its neighbors without worrying about the details of getting it there in one piece.

At least, the above is what the OSI reference model suggests. As we will see later, not everyone agrees that the data link layer should perform all these tasks.

Design Issues

If we don't follow the OSI reference model as gospel, we can imagine providing several alternative service semantics:

7.1.1. Reliable Delivery:

Frames are delivered to the receiver reliably and in the same order as generated by the sender.

Connection state keeps track of sending order and which frames require retransmission. For example, receiver state includes which frames have been received, which ones have not, etc.

7.1.2. Best Effort:

The receiver does not return acknowledgments to the sender, so the sender has no way of knowing if a frame has been successfully delivered.

When would such a service be appropriate?

1. When higher layers can recover from errors with little loss in performance. That is, when errors are so infrequent that there is little to be gained by the data link layer performing the recovery. It is just as easy to have higher layers deal with occasional lost packet.

2. For real-time applications requiring ``better never than late'' semantics. Old data may be *worse* than no data. For example, should an airplane bother calculating the proper wing flap angle using old altitude and wind speed data when newer data is already available?

7.1.3. Acknowledged Delivery:

The receiver returns an acknowledgment frame to the sender indicating that a data frame was properly received. This sits somewhere between the other two in that the sender keeps connection state, but may not necessarily retransmit unacknowledged frames. Likewise, the receiver may hand received packets to higher layers in the order in which the arrive, regardless of the original sending order.

Typically, each frame is assigned a unique *sequence number*, which the receiver returns in an *acknowledgment frame* to indicate which frame the ACK refers to. The sender must *retransmit* unacknowledged (e.g., lost or damaged) frames.

7.1.4. Framing

The DLL translates the physical layer's raw bit stream into discrete units (messages) called *frames*. How can the receiver detect frame boundaries? That is, how can the receiver recognize the start and end of a frame?

7.1.5. Length Count:

Make the first field in the frame's header be the length of the frame. That way the receiver knows how big the current frame is and can determine where the next frame ends.

Disadvantage: Receiver loses synchronization when bits become garbled. If the bits in the count become corrupted during transmission, the receiver will think that the frame contains fewer (or more) bits than it actually does. Although checksum will detect the incorrect frames, the receiver will have difficulty resynchronizing to the start of a new frame. This technique is not used anymore, since better techniques are available.

7.1.6. Bit Stuffing:

Use reserved bit patterns to indicate the start and end of a frame. For instance, use the 4-bit sequence of 0111 to delimit consecutive frames. A frame consists of everything between two delimiters.

Problem: What happens if the reserved delimiter happens to appear in the frame itself? If we don't remove it from the data, the receiver will think that the incoming frame is actually two smaller frames!

Solution: Use *bit stuffing*. Within the frame, replace every occurrence of two consecutive 1's with 110. E.g., append a zero bit after each pair of 1's in the data. This prevents 3 consecutive 1's from ever appearing in the frame.

Likewise, the receiver converts two consecutive 1's followed by a 0 into two 1's, but recognizes the 0111 sequence as the end of the frame.

Example: The frame ``1011101'' would be transmitted over the physical layer as ``0111101101010111''.

Note: When using bit stuffing, locating the start/end of a frame is easy, even when frames are damaged. The receiver simply scans arriving data for the reserved patterns. Moreover, the receiver will resynchronize quickly with the sender as to where frames begin and end, even when bits in the frame get garbled.

The main disadvantage with bit stuffing is the insertion of additional bits into the data stream, wasting bandwidth. How much expansion? The precise amount depends on the frequency in which the reserved patterns appear as user data.

7.1.7. Character stuffing:

Same idea as bit-stuffing, but operates on bytes instead of bits.

Use reserved characters to indicate the start and end of a frame. For instance, use the two-character sequence DLE STX (Data-Link Escape, Start of TeXt) to signal the beginning of a frame, and the sequence DLE ETX (End of TeXt) to flag the frame's end.

Problem: What happens if the two-character sequence DLE ETX happens to appear in the frame itself?

Solution: Use *character stuffing*; within the frame, replace every occurrence of DLE with the two-character sequence DLE DLE. The receiver reverses the processes, replacing every occurrence of DLE DLE with a single DLE.

Example: If the frame contained ``A B DLE D E DLE'', the characters transmitted over the channel would be ``DLE STX A B DLE DLE D E DLE DLE DLE ETX''.

Disadvantage: character is the smallest unit that can be operated on; not all architectures are byte oriented.

7.1.8. Encoding Violations:

Send an signal that doesn't conform to any legal bit representation. In Manchester encoding, for instance, 1-bits are represented by a high-low sequence, and 0-bits by low-high sequences. The start/end of a frame could be represented by the signal low-low or high-high.

The advantage of encoding violations is that no extra bandwidth is required as in bit-stuffing. The IEEE 802.4 standard uses this approach.

Finally, some systems use a combination of these techniques. IEEE 802.3, for instance, has both a length field and special frame start and frame end patterns.

7.1.9.Error Control

Error control is concerned with insuring that all frames are eventually delivered (possibly in order) to a destination. How? Three items are required.

Acknowledgements:

Typically, reliable delivery is achieved using the ``acknowledgments with retransmission'' paradigm, whereby the receiver returns a special *acknowledgment* (ACK) frame to the sender indicating the correct receipt of a frame.

In some systems, the receiver also returns a *negative acknowledgment* (NACK) for incorrectly-received frames. This is nothing more than a hint to the sender so that it can retransmit a frame right away without waiting for a timer to expire.

Timers:

One problem that simple ACK/NACK schemes fail to address is recovering from a frame that is lost, and as a result, fails to solicit an ACK or NACK. What happens if an ACK or NACK becomes lost?

Retransmission timers are used to resend frames that don't produce an ACK. When sending a frame, schedule a timer to expire at some time after the ACK should have been returned. If the timer goes off, retransmit the frame.

Sequence Numbers:

Retransmissions introduce the possibility of duplicate frames. To suppress duplicates, add sequence numbers to each frame, so that a receiver can distinguish between new frames and old copies.

Flow Control

Flow control deals with throttling the speed of the sender to match that of the receiver. Usually, this is a dynamic process, as the receiving speed depends on such changing factors as the load, and availability of buffer space.

One solution is to have the receiver extend *credits* to the sender. For each credit, the sender may send one frame. Thus, the receiver controls the transmission rate by handing out credits.

Link Management

In some cases, the data link layer service must be ``opened'' before use:

- The data link layer uses open operations for allocating buffer space, control blocks, agreeing on the maximum message size, etc.
- Synchronize and initialize send and receive sequence numbers with its peer at the other end of the communications channel.

7.2. ERROR CORRECTION AND DETECTION:

All error-detection and correction methods only work below a certain error rate .

If we allow *any* number of errors in data bits *and* in check bits, then *no* error-detection (or correction) method can *guarantee* to work, since any valid pattern can be transformed into any other valid pattern.

All methods of error-detection and correction only work if we assume the number of bits changed by error is below some threshold. If *all* bits can be changed, no error detection method can work even in theory. All methods only work below a certain error rate. Above that rate, the line is simply not usable.

7.2.1. General strategy:

Coding scheme so that a small no. of errors in this block won't change one legal pattern into another legal pattern:

- Frame or codeword length n = m (data) + r (redundant or check bits).
- *Any* data section (length m) is valid (we allow *any* 0,1 bit string).
- *Not* every codeword (length n) is valid.
- Make it so that:

(no. of *valid* codewords) is a very small subset of (all possible 0,1 bit strings of length n) so very *unlikely* that even large no. of errors will transform it into a *valid* codeword.

- Price to pay: Lots of extra check bits (high r).
- Obviously this works up to some error rate - won't work if no. of errors is *large* enough (e.g. = n).
- Error-check says "I will work if less than p errors in this block"

If errors still getting through: Reduce block size, so will get less error per block.

- Reduce m. Reduce the amount of info you transmit before doing error-checking. This can vastly reduce the probability of multiple errors per block.
- e.g. Say we have average 1 error per 1000. Transmit blocks of 10000. Average 10 errors each.
- Transmit blocks of 10. Average 1 error per 100 blocks. Almost never 2 errors in a block.

7.2.2. Error-correcting codes :

Frame or codeword length n = m (data) + r (redundant or check bits).

All 2^m patterns are legal.
Not all 2^n patterns are legal.
Basic idea: If illegal pattern, find the legal pattern closest to it. That might be the original data (before errors corrupted it).

Given two bit strings, XOR gives you the number of bits that are different. This is the Hamming distance.

If two codeword are Hamming distance d apart, it will take d one-bit errors to convert one into the other.

1. To *detect* (but not correct) up to d errors per length n, you need a coding scheme where code words are at least (d+1) apart in Hamming distance. Then d errors can't change into another legal code, so we know there's been an error.
2. To *correct* d errors, need code words (2d+1) apart. Then even with d errors, bit string will be d away from original and (d+1) away from nearest legal code. Still closest to original. Original can be reconstructed.

Error-detection example: Parity bit
Appended to data so that no. of 1 bits is even (or odd). Codeword distance 2.
Can detect (but not correct) 1 error.
Can't detect 2 errors.

Error-correction example: Sparse codeword
Let's say only 4 valid codeword, 10 bits:
0000000000
0000011111
1111100000
1111111111
Minimum distance 5.
Can detect and *correct* 1,2 errors.
Can detect but not *correct* 3,4 errors.
Can't detect 5 errors.
Note: Can detect *some* (even most) 5 bit errors.
Just can't guarantee to detect *all* 5 bit errors.
e.g. Encode every 2 bits this way.
Need **5 times the bandwidth** to send same amount of data.
But can communicate even if **almost every other bit is an error** (4 out of 10).

Theoretical limit of 1-bit error-correction
Detect and *correct* all 1 errors.
Need distance 3.
We need to have 2^m legal messages.
If change 1 bit, must get illegal (and an illegal which is 1 bit away from *this* message, but *not* 1 bit away from any *other* legal message).
So each of the 2^m must have n illegal codeword at distance 1 from it (systematically change each bit). These are *my* illegal, can't overlap with the illegal that are 1 bit away from other patterns.
Each message needs (n+1) patterns reserved for it.
$(n+1)\ 2^m <= 2^n$
$(n+1) <= 2^{n-m}$
$(m+r+1) <= 2^r$
For large r, this is always true.
i.e. This is putting a limit on how *small* r can be.
e.g. Let m=64. Then we need:
$r+65 <= 2^r$
Remember powers of 2.
i.e. r >= 7

What block size?
It scales well. Let m=1000. then r=10.
Question is, will 1000 bits only have 1 error?

e.g. Could send 1 M bits, need only 20 check bits to error-correct 1 bit error! But if there are 2 errors the system fails
.
If errors getting through: Reduce m until almost never get more than 1 error per block.

7.2.3. Hamming Code (1 bit error correction)

Achieves the theoretical limit for minimum number of check bits to do 1-bit error-correction.

Bits of codeword are numbered: bit 1, bit 2, ..., bit n. Check bits are inserted at positions 1,2,4,8,.. (all powers of 2). The rest are the m data bits.

Each check bit checks (as parity bit) a number of data bits. Each check bit checks a different collection of data bits. Check bits only check data, not other check bits.

Hamming Codes used in:

Wireless comms, e.g. fixed wireless broadband. High error rate. Need correction not detection.

Any number can be written as sum of powers of 2

First note every number can be written in base 2 as a sum of powers of 2 multiplied by 0 or 1.
i.e. As a simple sum of powers of 2.

Number is sum of these:	1	2	4	8	16
Number:					
1	x				
2		x			
3	x	x			
4			x		
5	x		x		
6		x	x		
7	x	x	x		
8				x	
9	x			x	
10		x		x	
11	x	x		x	
12			x	x	
13	x		x	x	
14		x	x	x	
15	x	x	x	x	
16					x
17	x				x
18		x			x
19	x	x			x
20			x		x
21	x		x		x
22		x	x		x
23	x	x	x		x
24				x	x
25	x			x	x

26		x		x	x
27	x	x		x	x
28			x	x	x
29	x		x	x	x
30		x	x	x	x
31	x	x	x	x	x

...

Scheme for check bits

Now here is our scheme for which bits each check bit checks:

Checked by check bit:	1	2	4	8	16

Bit:

1 (not applicable - this is a check bit)					
2 (n/a)					
3	x	x			
4 (n/a)					
5	x		x		
6		x	x		
7	x	x	x		
8 (n/a)					
9	x			x	
10		x		x	
11	x	x		x	
12			x	x	
13	x		x	x	
14		x	x	x	
15	x	x	x	x	
16 (n/a)					
17	x				x
18		x			x
19	x	x			x
20			x		x
21	x		x		x
22		x	x		x
23	x	x	x		x
24				x	x
25	x			x	x
26		x		x	x
27	x	x		x	x
28			x	x	x
29	x		x	x	x
30		x	x	x	x
31	x	x	x	x	x
32 (n/a)					

...

Check bit records odd or even parity of all the bits it covers, so any *one-bit* error in the data will lead to error in the check bit.

Assume one-bit error:
If any data bit bad, then *multiple* check bits will be bad (never just one check bit).

- Calculating the Hamming Code (check bits do *even* parity here)

How it works
21 (as sum of powers of 2) = 1 + 4 + 16
Bit 21 is checked by check bits 1, 4 and 16.

No other bit is checked by *exactly* these 3 check bits. If assume one-bit error, then if *exactly* these 3 check bits are bad, then we know that data bit 21 was bad *and no other*.

Assume one-bit error:
1. **Error in a data bit:**
   Will cause *multiple* errors in check bits. Will cause errors in *exactly* the check bits that correspond to the powers of 2 that the bit number can be written as a sum of.

2. **Error in a check bit:**
   Will affect nothing except that check bit. i.e. *One* bad check bit (not multiple bad check bits as above).

7.2.4. Error-detecting codes

Parity bit for 1 bit error detection

Parity bit can detect 1 error. Won't detect 2. *Will* detect 3, won't detect 4, etc.

If large block has 1 parity bit and is badly damaged, the odds of detecting this are only 0.5.

Parity bit for n bit burst error detection
Each block is considered as matrix, width n, height k. Parity bit calculated for each column.

Last row of parity bits appended: (parity bit for col 1, col 2, ..., col n)
Transmit n(k+1) block row by row.

1. Any burst of length up to n *in the data bits* will leave at most 1 error in each col.
   Can be detected.
2. If burst is in *the parity bits row*, they will be wrong and we will *detect* there has been an error.

Note: Don't know where error was (this is detection not correction). Will ask for re-transmit (don't know that data bits were actually ok).

For longer bursts, damaging the whole block:

Prob. of any given damaged column having correct parity by chance = 1/2

Prob. of all columns having correct parity by chance = $(1/2)^n$
Reasonable chance we'll detect it.

(If every parity bit in last line ok, it is prob. because there was no error, rather than by chance.)

Isolated errors

Isolated errors more difficult:

Any 2-bit error, where errors come at time t and t+n, will *not* be detected.

Polynomial codes for error detection

Also called CRC (Cyclic Redundancy Check)
Data is sent with a checksum.

When arrives, checksum is recalculated. Should match the one that was sent.

Bitstring represents polynomial.

e.g. 110001 represents:

$1 . x^5 + 1 . x^4 + 0 . x^3 + 0 . x^2 + 0 . x^1 + 1 . x^0$
$= x^5 + x^4 + x^0$

The **order** of a polynomial is the power of the highest non-zero coefficient. This is polynomial of order 5.

Special case: We don't allow bit string = all zeros. Easy to use **framing or stuffing** to make framed-and-stuffed transmission *never* all-zero,while still allowing payload within it to be all-zero.

7.2.5. Modulo 2 arithmetic

We are going to define a particular field (or here), in fact the smallest field there is, with only 2 members.

We define addition and subtraction as modulo 2 with no carries or borrows.

This means addition = subtraction = XOR.

Here are the rules for addition:

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0
Multiplication:
0 * 0 = 0
0 * 1 = 0
1 * 0 = 0
1 * 1 = 1
Subtraction: if 1+1=0, then 0-1=1, hence:
0 - 0 = 0
0 - 1 = 1
1 - 0 = 1
1 - 1 = 0
Long division is as normal, except the subtraction is modulo 2.

Example
No carry or borrow:
011 + (or minus)
110

101
Consider the polynomials:
$x + 1 + x^2 + x$

$x^2 + 2x + 1$
$= x^2 + 1$
We're saying the polynomial arithmetic is modulo 2 as well, so that:
$2\ x^k = 0$ for all k.

7.2.6. Error detection with CRC

Consider a message 110010 represented by the polynomial $M(x) = x^5 + x^4 + x$
Consider a *generating polynomial* $G(x) = x^3 + x^2 + 1$ (1101)
This is used to generate a 3 bit CRC = C(x) to be appended to M(x).
Note this G(x) is prime.

Steps:
1. Multiply M(x) by x^3 (highest power in G(x)). i.e. Add 3 zeros. 110010000
2. Divide the result by G(x). The remainder = C(x). 1101 long division into 110010000 (with subtraction mod 2) = 100100 remainder 100

Special case: This won't work if bit string = all zeros. We don't allow such an M(x).

But M(x) bit string = 1 will work, for example. Can divide 1101 into 1000.

If: x div y gives remainder c
that means: x = n y + c
Hence (x-c) = n y
(x-c) div y gives remainder 0
Here (x-c) = (x+c)
Hence (x+c) div y gives remainder 0
110010000 + 100 should give remainder 0.

3. Transmit 110010000 + 100
   To be precise, transmit: $T(x) = x^3M(x) + C(x)$
   = 110010100

4. Receiver end: Receive T(x). Divide by G(x), should have remainder 0.

Note if G(x) has order n - highest power is x^n,
then G(x) will cover (n+1) bits
and the *remainder* will cover n bits.
i.e. Add n bits to message.

8

DATA LINK LAYER PROTOCOLS

Unit Structure

8.1. Elementary data link protocols

- 8.1.1. An unrestricted simplex protocol
- 8.1.2. A simplex stop-and-wait protocol
- 8.1.3. A simplex protocol for a noisy channel

8.1. ELEMENTARY DATA LINK PROTOCOLS

Assumptions:

- The physical layer, data link layer, and network layer are independent processes.
- The network layer contacts the data link layer by generating proper *events*.

  There exist suitable library procedures for the data link layer to delivery a packet, and to fetch a packet.

  The data link layer provides a reliable, connection-oriented service to the network layer.
- The physical layer computes/checks checksums in outgoing/incoming frames, and contacts the data link layer by generating proper *events*.

  There exist suitable library procedures for the data link layer to send a frame ( ), and to fetch a frame ( ).

```
#define MAX_PKT 1024 /* determines packet size in bytes */


typedef enum {false, true} boolean; /* boolean type */
typedef unsigned int seq_nr;  /* sequence or ack numbers */
typedef struct {unsigned char data[MAX_PKT];} packet; /*
packet def. */
typedef  enum {data, ack, nak} frame_kind;  /* frame_kind
definition */
typedef  struct {       /* frames are transported in this layer */
  frame_kind kind;      /* what kind of a frame is it ? */
  seq_nr     seq;       /* sequence number */
```

```
  seq_nr     ack;       /* acknowledgement number */
  packet     info;      /* the network layer packet */
} frame;

/* Wait for an event to happen; return its type in event. */
void wait_for_event(event_type *event);

/* Fetch/pass a packet from/to the network layer */
void from_network_layer(packet *p);
void to_network_layer(packet *r);

/* Fetch/pass an inbound/outbound frame from/to the
physical layer. */
void from_physical_layer(frame *r);
void to_physical_layer(frame *s);

/* Start/stop the clock running and enable/disable the timeout
event */
void start_timer(seq_nr k);
void stop_timer(seq_nr k);

/* Start/stop an auxiliary timer and enable/disable the
ack_timeout event */
void start_ack_timer(void);
void stop_ack_timer(void);

/* Allow/disallow the network layer to cause a
network_layer_ready event */
void enable_network_layer(void);
void disable_network_layer(void);

/* Macro inc is expanded in-line: Increment k circularly */
#define inc(k) if(k < MAX_SEQ) k = k + 1; else k = 0
```

8.1.1. An unrestricted simplex protocol

Assumptions:

- Data transmission in one direction only.
- Channel is error free.
- The receiver is able to process incoming data infinitely fast.

```
/* Protocol 1 (utopia) provides for data transmission in one
direction only, from the sender to receiver. The communication
channel is assumed  to be error free, and the receiver is assumed
to be able to process all  the input infinitely fast. Consequently, the
sender just sits in a loop  pumping data out onto the line as fast as
it can */
```

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender1(void)
{
  frame s;                  /* buffer for an outbound frame */
  packet buffer;              /* buffer for an outbound packet */

  while (true) {
    from_network_layer(&buffer);/* get something to send */
    s.info = buffer; /* copy it into s for transmission */
    to_physical_layer(&s); /* send it on its way */
  }
}

void receiver1(void)
{
  frame r;
  event_type event  /* filled in by wait, but not used here */

  while(true) {
    wait_for_event(&event); /* only possiblity is frame_arrival */
    from_physical_layer(&r);   /* go get the inbound frame */
    to_network_layer(&r.info); /* pass the data to the network
layer */
}
```

8.1.2. A simplex stop-and-wait protocol

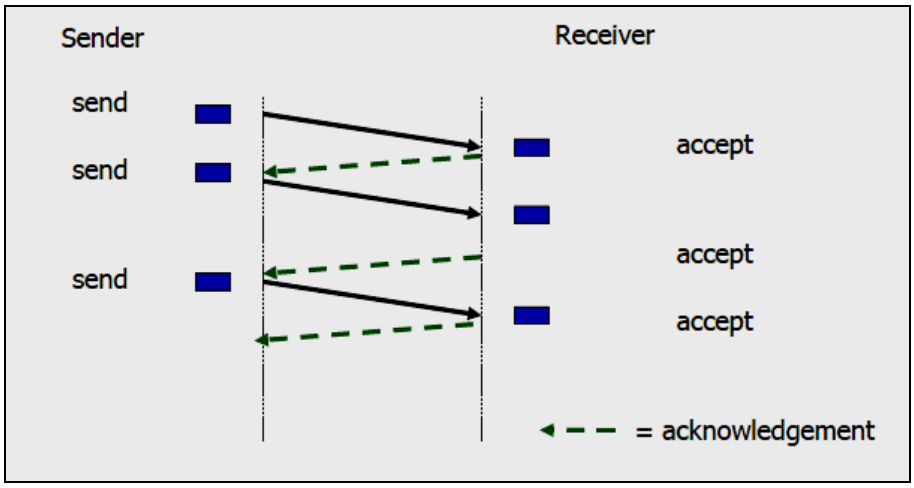
Assumptions:

- Data transmission in one direction only.
- Channel is error free.

How to prevent the sender from flooding the receiver with data ?

A general solution:

- After finishing the process of an incoming frame, the receiver sends a dummy frame back to the sender.
- Only when the sender has got the acknowledgement (a dummy frame) from the receiver, does the sender transmit another frame.



The above strategy is called **stop-and-wait**.

Data traffic is still simplex, but frames do travel in both directions.

The communication channel needs to be capable of bidirectional information transfer (a half-duplex physical channel suffices here).

/* Protocol 2 (stop-and-wait) also provides for a one-directional flow of data from sender to receiver. The communication channel is once agian assumed to be error free. However, the receiver has only a finite buffer capacity and a finite processing speed, so the protocol must explicitly prevent the sender from flooding the receiver with data faster than it can be handled. */

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"


void sender2(void)
{
  frame s;   /* buffer for an outbound frame */
  packet buffer;  /* buffer for an outbound packet */
  event_type event;  /* frame_arrival is the only possiblity */


  while (true) {
    from_network_layer(&buffer);/* get something to send */
    s.info = buffer;  /* copy it into s for transmission */
    to_physical_layer(&s);  /* send it on its way */
    wait_for_event(&event); /* do not proceed until given the go
ahead */
```

```
  }
}
void receiver2(void)
{
  frame r, s;  /* buffers for frames */
  event_type event  /* filled in by wait, but not used here */
  while(true) {
    wait_for_event(&event);    /* only possiblity is frame_arrival */
    from_physical_layer(&r);   /* go get the inbound frame */
    to_network_layer(&r.info); /* pass the data to the network layer */
    to_physical_layer(&s);   /* send a dummy frame to awaken
sender */


}
```

8.1.3. A simplex protocol for a noisy channel

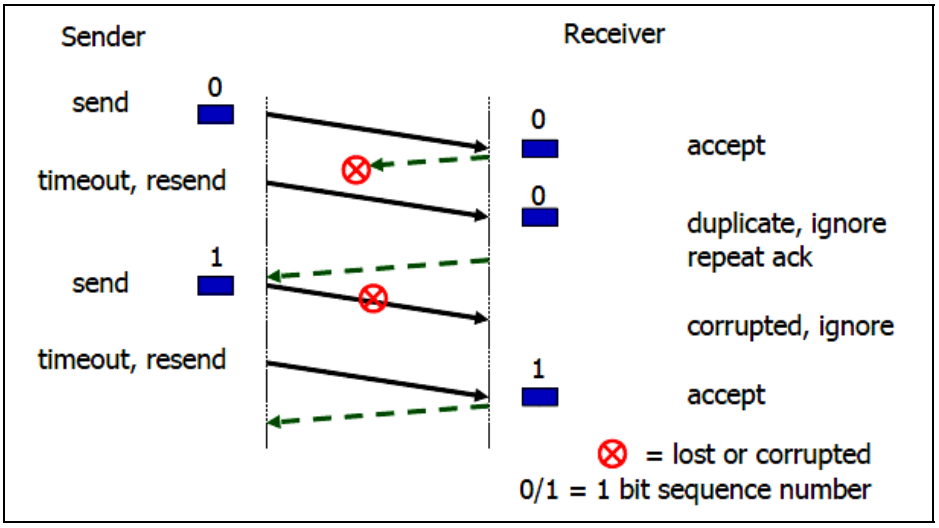
Assumptions:

- Data transmission in one direction only.
- Channel is NOT error free, but a damaged frame can be detected by the receiver hardware by computing the checksum

.

A scheme using **timer** and **acknowledgement**:

- The sender starts a timer after sending a frame.
- The receiver sends back an acknowledgement only when the incoming frame were correctly received.
- The sender will retransmit the frame if the timer expired before receiving an acknowledgement.



What is the fatal flaw in the above scheme ?

The fatal flaw in the above mentioned scheme is the Loss of an acknowledgement duplicated frame. To distinguish a frame being seen for the first time from a retransmission by using a sequence number in the header of each frame.

What is the minimum number of bits needed for the sequence number ?

A 1-bit sequence number (0 or 1) suffices here.

```
/*Protocol 3 (par) */
#define MAX_SEQ 1              /* must be 1 for protocol 3 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"


void sender3(void) {
  seq_nr next_frame_to_send;     /* seq number of next outgoing frame */
  frame s;                  /* buffer for an outbound frame */
  packet buffer;              /* buffer for an outbound packet */
  event_type event;           /* frame_arrival is the only possiblity */
  next_frame_to_send = 0;          /* initialize outbound sequence numbers */
  from_network_layer(&buffer); /* fetch first packet */
  while (true) {
    s.info = buffer;          /* construct a frame for transmission */
    s.seq = next_frame_to_send; /* insert sequence number in frame */
    to_physical_layer(&s);      /* send it on its way */
    start_timer(s.seq);         /* if answer takes too long, time out */
    wait_for_event(&event);     /* frame_arrival, cksum_err, timeout */
    if(event == frame_arrival) {
       from_network_layer(&buffer);/* get the next one to send */
       inc(next_frame_to_send); /* invert next_frame_to_send */
    }
  }
}
```

```
void receiver3(void) {
  seq_nr frame_expected;
  frame r, s;                 /* buffers for frames */
  event_type event;           /* filled in by wait, but not used here */
  frame_expected = 0;
  while(true) {
    wait_for_event(&event);    /* frame_arrival, cksum_err */
    if(event == frame_arrival) { /* a valid frame has arrived */
      from_physical_layer(&r);   /* go get the inbound frame */
      if(r.seq == frame_expected){
        to_network_layer(&r.info);/*pass the data to the network
layer*/
        inc(frame_expected); /* next time expect the other seq nr */
      }
      to_physical_layer(&s);  /* none of the fields are used */
    }
  }
}
```

9

SLIDING WINDOW PROTOCOLS

Unit Structure

9.1. SLIDING WINDOW PROTOCOLS
- 9.1.1 A One-Bit Sliding Window Protocol
- 9.1.2 A Protocol Using Go Back N
- 9.1.3. A Protocol Using Selective Repeat

9.2. HDLC
- 9.2.1. HDLC STATIONS AND CONFIGURATIONS
- 9.2.2. Stations
- 9.2.3. HDLC Operational Modes
- 9.2.4. HDLC Non-Operational Modes
- 9.2.5. HDLC Frame Structure
- 9.2.6. HDLC COMMANDS AND RESPONSES

9.1. SLIDING WINDOW PROTOCOLS

In the previous protocols, data frames were transmitted in one direction only. In most practical situations, there is a need for transmitting data in both directions. One way of achieving full-duplex data transmission is to have two separate communication channels and use each one for simplex data traffic (in different directions).

If this is done, we have two separate physical circuits, each with a ‘‘forward’’ channel (for data) and a ‘‘reverse’’ channel (for acknowledgements). In both cases the bandwidth of the reverse channel is almost entirely wasted. In effect, the user is paying for two circuits but using only the capacity of one.

A better idea is to use the same circuit for data in both directions. After all, in protocols 2 and 3 it was already being used to transmit frames both ways, and the reverse channel has the same capacity as the forward channel. In this model the data frames from *A* to *B* are intermixed with the acknowledgement frames from *A* to *B*. By looking at the *kind* field in the header of an incoming frame, the receiver can tell whether the frame is data or acknowledgement.

Although interleaving data and control frames on the same circuit is an improvement over having two separate physical circuits, yet another improvement is possible. When a data frame arrives, instead of immediately sending a separate control frame, the receiver restrains itself and waits until the network layer passes it the next packet. The acknowledgement is attached to the outgoing data frame (using the *ack* field in the frame header). In effect, the acknowledgement gets a free ride on the next outgoing data frame. The technique of temporarily delaying outgoing acknowledgements so that they can be hooked onto the next outgoing data frame is known as **piggybacking**.

The principal advantage of using piggybacking over having distinct acknowledgement frames is a better use of the available channel bandwidth. The *ack* field in the frame header costs only a few bits, whereas a separate frame would need a header, the acknowledgement, and a checksum. In addition, fewer frames sent means fewer ‘‘frame arrival’’ interrupts, and perhaps fewer buffers in the receiver, depending on how the receiver’s software is organized. In the next protocol to be examined, the piggyback field costs only 1 bit in the frame header. It rarely costs more than a few bits.

However, piggybacking introduces a complication not present with separate acknowledgements. How long should the data link layer wait for a packet onto which to piggyback the acknowledgement? If the data link layer waits longer than the sender’s timeout period, the frame will be retransmitted, defeating the whole purpose of having acknowledgements. If the data link layer were an oracle and could foretell the future, it would know when the next network layer packet was going to come in and could decide either to wait for it or send a separate acknowledgement immediately, depending on how long the projected wait was going to be. Of course, the data link layer cannot foretell the future, so it must resort to some ad hoc scheme, such as waiting a fixed number of milliseconds. If a new packet arrives quickly, the acknowledgement is piggybacked onto it; otherwise, if no new packet has arrived by the end of this time period, the data link layer just sends a separate acknowledgement frame.

The next three protocols are bidirectional protocols that belong to a class called **sliding window** protocols. The three differ among themselves in terms of efficiency, complexity, and buffer requirements, as discussed later. In these, as in all sliding window protocols, each outbound frame contains a sequence number, ranging from 0 up to some maximum. The maximum is usually $2^n - 1$ so the sequence number fits exactly in an n-bit field. The stop-and-wait sliding window protocol uses $n = 1$, restricting the sequence numbers to 0 and 1, but more sophisticated versions can use arbitrary n.

The essence of all sliding window protocols is that at any instant of time, the sender maintains a set of sequence numbers corresponding to frames it is permitted to send. These frames are said to fall within the **sending window**. Similarly, the receiver also maintains a **receiving window** corresponding to the set of frames it is permitted to accept. The sender's window and the receiver's window need not have the same lower and upper limits or even have the same size. In some protocols they are fixed in size, but in others they can grow or shrink over the course of time as frames are sent and received.

Although these protocols give the data link layer more freedom about the order in which it may send and receive frames, we have definitely not dropped the requirement that the protocol must deliver packets to the destination network layer in the same order they were passed to the data link layer on the sending machine.

Nor have we changed the requirement that the physical communication channel is ‘‘wire-like,’’ that is, it must deliver all frames in the order sent.

The sequence numbers within the sender's window represent frames that have been sent or can be sent but are as yet not acknowledged. Whenever a new packet arrives from the network layer, it is given the next highest sequence number, and the upper edge of the window is advanced by one. When an acknowledgement comes in, the lower edge is advanced by one. In this way the window continuously maintains a list of unacknowledged frames.

Since frames currently within the sender's window may ultimately be lost or damaged in transit, the sender must keep all these frames in its memory for possible retransmission. Thus, if the maximum window size is *n*, the sender needs *n* buffers to hold the unacknowledged frames. If the window ever grows to its maximum size, the sending data link layer must forcibly shut off the network layer until another buffer becomes free.The receiving data link layer's window corresponds to the frames it may accept. Any frame falling outside the window is discarded without comment. When a frame whose sequence number is equal to the lower edge of the window is received, it is passed to the network layer, an acknowledgement is generated, and the window is rotated by one. Unlike the sender's window, the receiver's window always remains at its initial size. Note that a window size of 1 means that the

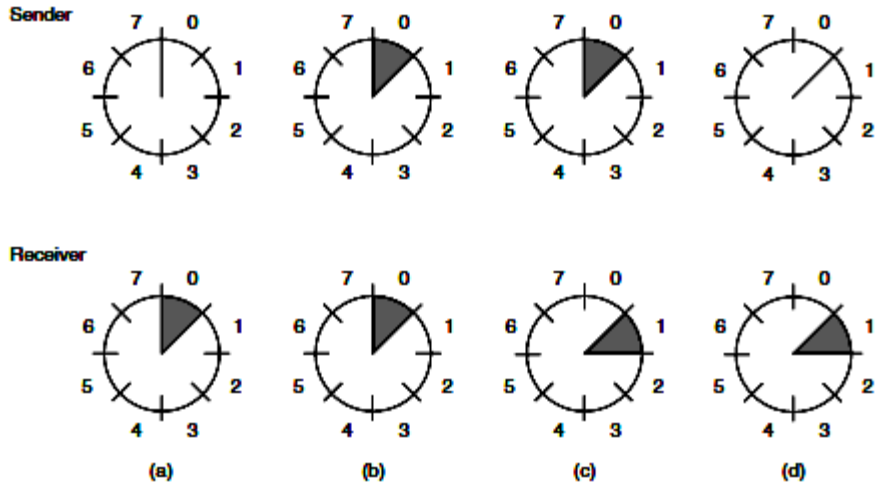


Figure 3-13. A sliding window of size 1, with a 3-bit sequence number. (a) Initially. (b) After the first frame has been sent. (c) After the first frame has been received. (d) After the first acknowledgement has been received.

(a) Initially. (b) After the first frame has been sent. (c) After the first frame has been received. (d) After the first acknowledgement has been received. data link layer only accepts frames in order, but for larger windows this is not so. The network layer, in contrast, is always fed data in the proper order, regardless of the data link layer's window size.

The figure shows an example with a maximum window size of 1. Initially, no frames are outstanding, so the lower and upper edges of the sender's window are equal, but as time goes on, the situation progresses as shown.

9.1.1 A One-Bit Sliding Window Protocol

Before tackling the general case, let us first examine a sliding window protocol with a maximum window size of 1. Such a protocol uses stop-and-wait since the sender transmits a frame and waits for its acknowledgement before sending the next one.

Like the others, it starts out by defining some variables. *Next*3*frame*3*to*3*send* tells which frame the sender is trying to send. Similarly, *frame*3*expected* tells which frame the receiver is expecting. In both cases, 0 and 1 are the only possibilities.

Under normal circumstances, one of the two data link layers goes first and transmits the first frame. In other words, only one of the data link layer programs should contain the *to*3*physical*3*layer* and *start*3*timer* procedure calls outside the main loop

```
/* Protocol 4 (sliding window) is bidirectional. */
#define MAX3SEQ 1 /* must be 1 for protocol 4 */
typedef enum {frame3arrival, cksum3err, timeout} event3type;
#include "protocol.h"
void protocol4 (void)
{
seq3nr next3frame3to3send; /* 0 or 1 only */
seq3nr frame3expected; /* 0 or 1 only */
frame r, s; /* scratch variables */
packet buffer; /* current packet being sent */
event3type event;
next3frame3to3send = 0; /* next frame on the outbound stream */
frame3expected = 0; /* frame expected next */
from3network3layer(&buffer); /* fetch a packet from the network layer */
s.info = buffer; /* prepare to send the initial frame */
s.seq = next3frame3to3send; /* insert sequence number into frame */
s.ack = 1 − frame3expected; /* piggybacked ack */
to3physical3layer(&s); /* transmit the frame */
start3timer(s.seq); /* start the timer running */
while (true) {
wait3for3event(&event); /* frame3arrival, cksum3err, or timeout */
if (event == frame3arrival) { /* a frame has arrived undamaged. */
from3physical3layer(&r); /* go get it */
if (r.seq == frame3expected) { /* handle inbound frame stream. */
to3network3layer(&r.info); /* pass packet to network layer */
inc(frame3expected); /* invert seq number expected next */
}
if (r.ack == next3frame3to3send) { /* handle outbound frame stream. */
stop3timer(r.ack); /* turn the timer off */
```

```
from3network3layer(&buffer); /* fetch new pkt from network layer */
inc(next3frame3to3send); /* invert sender's sequence number */
}
}
s.info = buffer; /* construct outbound frame */
s.seq = next3frame3to3send; /* insert sequence number into it */
s.ack = 1 − frame3expected; /* seq number of last received frame */
to3physical3layer(&s); /* transmit a frame */
start3timer(s.seq); /* start the timer running */
}
}
```

The starting machine fetches the first packet from its network layer, builds a frame from it, and sends it. When this (or any) frame arrives, the receiving data link layer checks to see if it is a duplicate, just as in protocol 3. If the frame is the one expected, it is passed to the network layer and the receiver's window is slid up.

The acknowledgement field contains the number of the last frame received without error. If this number agrees with the sequence number of the frame the sender is trying to send, the sender knows it is done with the frame stored in *buffer* and can fetch the next packet from its network layer. If the sequence number disagrees, it must continue trying to send the same frame. Whenever a frame is received, a frame is also sent back.

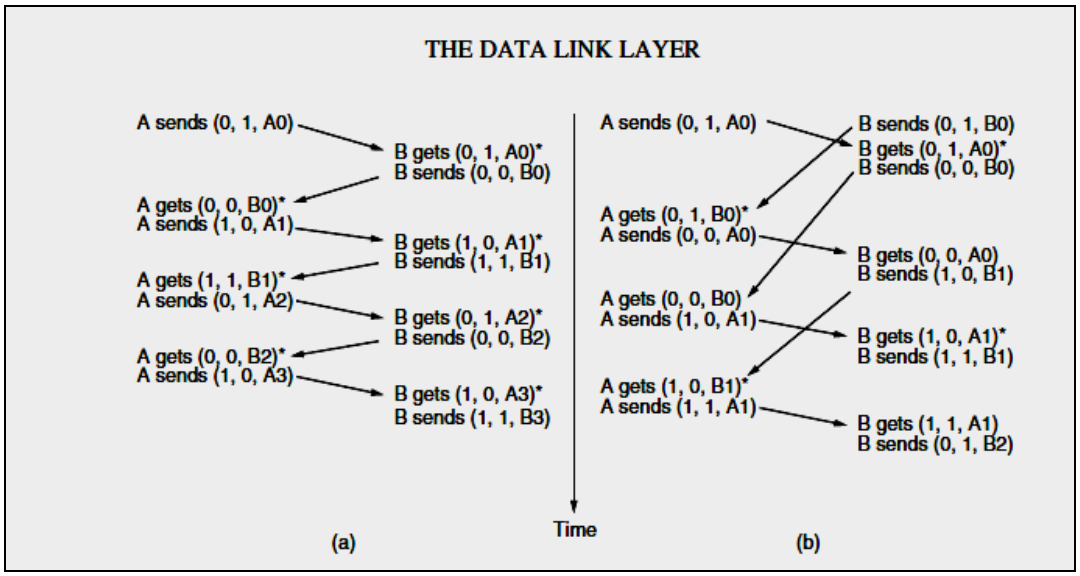
Now let us examine protocol 4 to see how resilient it is to pathological scenarios. Assume that computer *A* is trying to send its frame 0 to computer *B* and that *B* is trying to send its frame 0 to *A*. Suppose that *A* sends a frame to *B*, but *A*'s timeout interval is a little too short. Consequently, *A* may time out repeatedly, sending a series of identical frames, all with *seq* = 0 and *ack* = 1.

When the first valid frame arrives at computer *B*, it will be accepted and *frame*3*expected* will be set to 1. All the subsequent frames will be rejected because *B* is now expecting frames with sequence number 1, not 0. Furthermore, since all the duplicates have *ack* = 1 and *B* is still waiting for an acknowledgement of 0, *B* will not fetch a new packet from its network layer.

After every rejected duplicate comes in, *B* sends *A* a frame containing *seq* = 0 and *ack* = 0. Eventually, one of these arrives correctly at *A*, causing *A* to begin sending the next packet. No combination of lost frames or premature timeouts can cause the protocol to deliver duplicate packets to either network layer, to skip a packet, or to deadlock.

However, a peculiar situation arises if both sides simultaneously send an initial packet. This synchronization difficulty is illustrated by the following figure. In part (a), the normal operation of the protocol is shown. In (b) the peculiarity is illustrated.

If *B* waits for *A*'s first frame before sending one of its own, the sequence is as shown in (a), and every frame is accepted. However, if *A* and *B* simultaneously initiate communication, their first frames cross, and the data link layers then get into situation (b). In (a) each frame arrival brings a new packet for the network layer; there are no duplicates. In (b) half of the frames contain duplicates, even though there are no transmission errors. Similar situations can occur as a result of premature timeouts, even when one side clearly starts first. In fact, if multiple premature timeouts occur, frames may be sent three or more times.



9.1.2 A Protocol Using Go Back N

Until now we have made the tacit assumption that the transmission time required for a frame to arrive at the receiver plus the transmission time for the acknowledgement to come back is negligible. Sometimes this assumption is clearly false. In these situations the long round-trip time can have important implications for the efficiency of the bandwidth utilization. As an example, consider a 50-kbps satellite channel with a 500-msec round-trip propagation delay. Let us imagine trying to use protocol 4 to send 1000-bit frames via the satellite. At $t = 0$ the sender starts sending

the first frame. At t = 20 msec the frame has been completely sent. Not until t = 270 msec has the frame fully arrived at the receiver, and not until t = 520 msec has the acknowledgement arrived back at the sender, under the best of circumstances (no waiting in the receiver and a short acknowledgement frame).

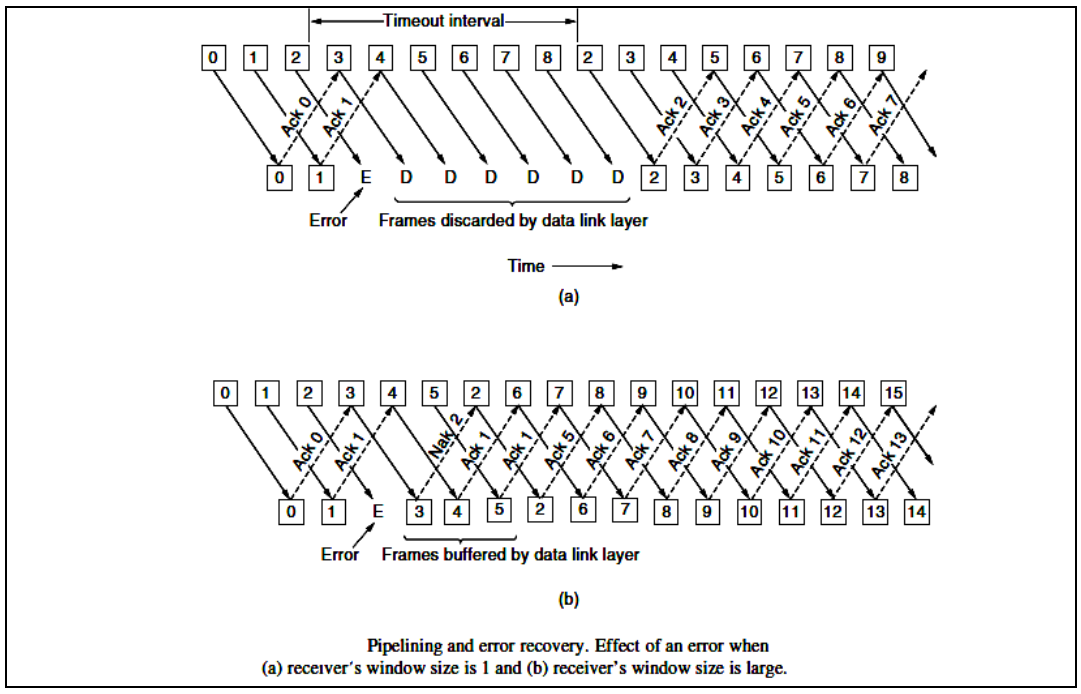
This means that the sender was blocked during 500/520 or 96 percent of the time. In other words, only 4 percent of the available bandwidth was used. Clearly, the combination of a long transit time, high bandwidth, and short frame length is disastrous in terms of efficiency.

The problem described above can be viewed as a consequence of the rule requiring a sender to wait for an acknowledgement before sending another frame. If we relax that restriction, much better efficiency can be achieved. Basically, the solution lies in allowing the sender to transmit up to w frames before blocking, instead of just 1. With an appropriate choice of w the sender will be able to continuously transmit frames for a time equal to the round-trip transit time without filling up the window. In the example above, w should be at least 26. The sender begins sending frame 0 as before. By the time it has finished sending 26 frames, at t = 520, the acknowledgement for frame 0 will have just arrived. Thereafter, acknowledgements arrive every 20 msec, so the sender always gets permission to continue just when it needs it. At all times, 25 or 26 unacknowledged frames are outstanding. Put in other terms, the sender's maximum window size is 26.

The need for a large window on the sending side occurs whenever the product of bandwidth × round-trip-delay is large. If the bandwidth is high, even for a moderate delay, the sender will exhaust its window quickly unless it has a large window. If the delay is high (e.g., on a geostationary satellite channel), the sender will exhaust its window even for a moderate bandwidth. The product of these two factors basically tells what the capacity of the pipe is, and the sender needs the ability to fill it without stopping in order to operate at peak efficiency.

This technique is known as **pipelining**. If the channel capacity is b bits/sec, the frame size l bits, and the round-trip propagation time R sec, the time required to transmit a single frame is l/b sec. After the last bit of a data frame has been sent, there is a delay of $R/2$ before that bit arrives at the receiver and another delay of at least $R/2$ for the acknowledgement to come back, for a total delay of R. In stop-and-wait the line is busy for l/b and idle for R, giving line utilization = $l/(l + bR)$ If $l < bR$, the efficiency will be less than 50 percent. Since there is always a nonzero delay for the acknowledgement to propagate back, pipelining can, in principle, be used to keep the line busy during this interval, but if the interval is small, the additional complexity is not worth the trouble.

Pipelining frames over an unreliable communication channel raises some serious issues. First, what happens if a frame in the middle of a long stream is damaged or lost? Large numbers of succeeding frames will arrive at the receiver before the sender even finds out that anything is wrong. When a damaged frame arrives at the receiver, it obviously should be discarded, but what should the receiver do with all the correct frames following it? Remember that the receiving data link layer is obligated to hand packets to the network layer in sequence. In the following figure, we see the effects of pipelining on error recovery. We will now examine it in some detail.



Pipelining and error recovery. Effect of an error when (a) receiver's window size is 1 and (b) receiver's window size is large.

Two basic approaches are available for dealing with errors in the presence of pipelining. One way, called **go back n**, is for the receiver simply to discard all subsequent frames, sending no acknowledgements for the discarded frames. This strategy corresponds to a receive window of size 1. In other words, the data link layer refuses to accept any frame except the next one it must give to the network layer. If the sender's window fills up before the timer runs out, the pipeline will begin to empty.

Eventually, the sender will time out and retransmit all unacknowledged frames in order, starting with the damaged or lost one. This approach can waste a lot of bandwidth if the error rate is high.

In Fig (a) we see go back n for the case in which the receiver's window is large. Frames 0 and 1 are correctly received and acknowledged. Frame 2, however, is damaged or lost. The sender, unaware of this problem, continues to send frames until the

timer for frame 2 expires. Then it backs up to frame 2 and starts all over with it, sending 2, 3, 4, etc. all over again.

The other general strategy for handling errors when frames are pipelined is called **selective repeat**. When it is used, a bad frame that is received is discarded, but good frames received after it are buffered. When the sender times out, only the oldest unacknowledged frame is retransmitted. If that frame arrives correctly, the receiver can deliver to the network layer, in sequence, all the frames it has buffered. Selective repeat is often combined with having the receiver send a negative acknowledgement (NAK) when it detects an error, for example, when it receives a checksum error or a frame out of sequence. NAKs stimulate retransmission before the corresponding timer expires and thus improve performance.

In Fig. (b), frames 0 and 1 are again correctly received and acknowledged and frame 2 is lost. When frame 3 arrives at the receiver, the data link layer there notices that is has missed a frame, so it sends back a NAK for 2 but buffers 3. When frames 4 and 5 arrive, they, too, are buffered by the data link layer instead of being passed to the network layer. Eventually, the NAK 2 gets back to the sender, which immediately resends frame 2. When that arrives, the data link layer now has 2, 3, 4, and 5 and can pass all of them to the network layer in the correct protocol 5 (go back n) allows multiple outstanding frames.

The sender may transmit up to MAX3SEQ frames without waiting for an ack. In addition, unlike in the previous protocols, the network layer is not assumed to have a new packet all the time. Instead, the network layer causes a network3layer3ready event when there is a packet to send.

```
*/ #define MAX3SEQ 7 /* should be 2^n − 1 */

typedef enum {frame3arrival, cksum3err, timeout,
network3layer3ready} event3type;

#include "protocol.h"

static boolean between(seq3nr a, seq3nr b, seq3nr c)

{

/* Return true if a <=b < c circularly; false otherwise. */

if (((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) &&
(c < a)))

return(true);

else

return(false);
```

```
}
static void send3data(seq3nr frame3nr, seq3nr frame3expected, packet buffer[ ])
{
/* Construct and send a data frame. */
frame s; /* scratch variable */
s.info = buffer[frame3nr]; /* insert packet into frame */
s.seq = frame3nr; /* insert sequence number into frame */
s.ack = (frame3expected + MAX3SEQ) % (MAX3SEQ + 1);/* piggyback ack */
to3physical3layer(&s); /* transmit the frame */
start3timer(frame3nr); /* start the timer running */
}
void protocol5(void)
{
seq3nr next3frame3to3send; /* MAX3SEQ > 1; used for outbound stream */
seq3nr ack3expected; /* oldest frame as yet unacknowledged */
seq3nr frame3expected; /* next frame expected on inbound stream */
frame r; /* scratch variable */
packet buffer[MAX3SEQ + 1]; /* buffers for the outbound stream */
seq3nr nbuffered; /* # output buffers currently in use */
seq3nr i; /* used to index into the buffer array */
event3type event;
enable3network3layer(); /* allow network3layer3ready events */
ack3expected = 0; /* next ack expected inbound */
next3frame3to3send = 0; /* next frame going out */
frame3expected = 0; /* number of frame expected inbound */
nbuffered = 0; /* initially no packets are buffered */
```



```
while (true) {
```

```
wait3for3event(&event); /* four possibilities: see event3type above */
switch(event) {
case network3layer3ready: /* the network layer has a packet to send */
/* Accept, save, and transmit a new frame. */
from3network3layer(&buffer[next3frame3to3send]); /* fetch new packet */
nbuffered = nbuffered + 1; /* expand the sender's window */
send3data(next3frame3to3send, frame3expected, buffer);/* transmit the frame */
inc(next3frame3to3send); /* advance sender's upper window edge */
break;
case frame3arrival: /* a data or control frame has arrived */
from3physical3layer(&r); /* get incoming frame from physical layer */
if (r.seq == frame3expected) {
/* Frames are accepted only in order. */
to3network3layer(&r.info); /* pass packet to network layer */
inc(frame3expected); /* advance lower edge of receiver's window */
}
/* Ack n implies n − 1, n − 2, etc. Check for this. */
while (between(ack3expected, r.ack, next3frame3to3send)) {
/* Handle piggybacked ack. */
nbuffered = nbuffered − 1; /* one frame fewer buffered */
stop3timer(ack3expected); /* frame arrived intact; stop timer */
inc(ack3expected); /* contract sender's window */
}
break;
case cksum3err: break; /* just ignore bad frames */
case timeout: /* trouble; retransmit all outstanding frames */
next3frame3to3send = ack3expected; /* start retransmitting here */
for (i = 1; i <= nbuffered; i++) {
```

```
send3data(next3frame3to3send, frame3expected, buffer);/*
resend 1 frame */
inc(next3frame3to3send); /* prepare to send the next one */
}
}
if (nbuffered < MAX3SEQ)
enable3network3layer();
else
disable3network3layer();
}
}
```

order. It can also acknowledge all frames up to and including 5, as shown in the figure. If the NAK should get lost, eventually the sender will time out for frame 2 and send it (and only it) of its own accord, but that may be a quite a while later. In effect, the NAK speeds up the retransmission of one specific frame. Selective repeat corresponds to a receiver window larger than 1. Any frame within the window may be accepted and buffered until all the preceding ones have been passed to the network layer.

This approach can require large amounts of data link layer memory if the window is large. These two alternative approaches are trade-offs between bandwidth and data link layer buffer space. Depending on which resource is scarcer, one or the other can be used. The above figure shows a pipelining protocol in which the receiving data link layer only accepts frames in order; frames following an error are discarded.

In this protocol, for the first time we have dropped the assumption that the network layer always has an infinite supply of packets to send. When the network layer has a packet it wants to send, it can cause a *network*3*layer*3*ready* event to happen. However, to enforce the flow control rule of no more than *MAX*3*SEQ* unacknowledged frames outstanding at any time, the data link layer must be able to keep the network layer from bothering it with more work. The library procedures *enable*3*network*3*layer* and *disable*3*network*3*layer* do this job.

Note that a maximum of *MAX*3*SEQ* frames and not *MAX*3*SEQ* + 1 frames may be outstanding at any instant, even though there are *MAX*3*SEQ* + 1 distinct sequence numbers: 0, 1, 2, . . . , *MAX*3*SEQ.* To see why this restriction is required, consider the following scenario with *MAX*3*SEQ* = 7*.*

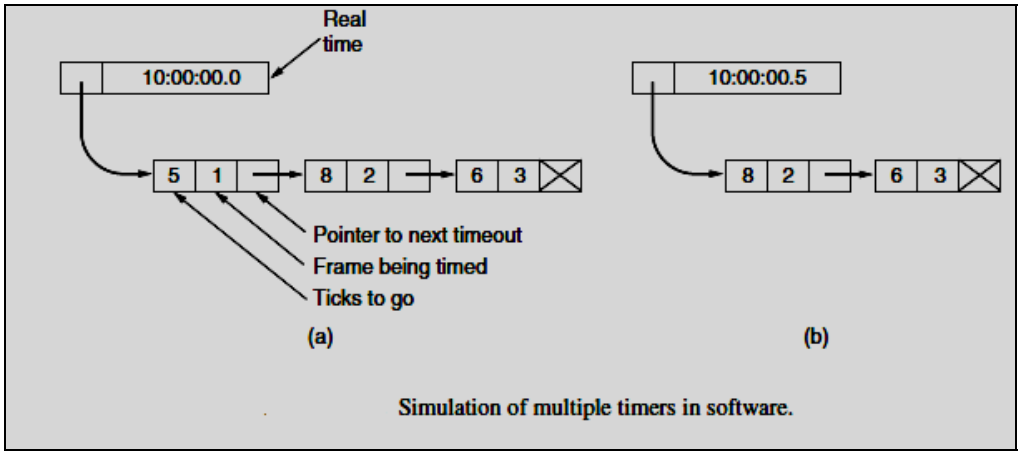
1. The sender sends frames 0 through 7.
2. A piggybacked acknowledgement for frame 7 eventually comes back to the sender.
3. The sender sends another eight frames, again with sequence numbers 0 through 7.
4. Now another piggybacked acknowledgement for frame 7 comes in.

The question is this: Did all eight frames belonging to the second batch arrive successfully, or did all eight get lost (counting discards following an error as lost)? In both cases the receiver would be sending frame 7 as the acknowledgement. The sender has no way of telling. For this reason the maximum number of outstanding frames must be restricted to *MAX*3*SEQ*.

Although protocol 5 does not buffer the frames arriving after an error, it does not escape the problem of buffering altogether. Since a sender may have to retransmit all the unacknowledged frames at a future time, it must hang on to all transmitted frames until it knows for sure that they have been accepted by the receiver. When an acknowledgement comes in for frame *n*, frames *n* − 1, *n* − 2, and so on are also automatically acknowledged. This property is especially important when some of the previous acknowledgement-bearing frames were lost or garbled. Whenever any acknowledgement comes in, the data link layer checks to see if any buffers can now be released. If buffers can be released (i.e., there is some room available in the window), a previously blocked network layer can now be allowed to cause more *network*3*layer*3*ready* events.

For this protocol, we assume that there is always reverse traffic on which to piggyback acknowledgements. If there is not, no acknowledgements can be sent. Protocol 4 does not need this assumption since it sends back one frame every time it receives a frame, even if it has just already sent that frame. In the next protocol we will solve the problem of one-way traffic in an elegant way.

Because protocol 5 has multiple outstanding frames, it logically needs multiple timers, one per outstanding frame. Each frame times out independently of all the other ones. All of these timers can easily be simulated in software, using a single hardware clock that causes interrupts periodically. The pending timeouts form a linked list, with each node of the list telling the number of clock ticks until the timer expires, the frame being timed, and a pointer to the next node.



Simulation of multiple timers in software.

As an illustration of how the timers could be implemented, consider the example of the above figure(a). Assume that the clock ticks once every 100 msec. Initially, the real time is 10:00:00.0; three timeouts are pending, at 10:00:00.5, 10:00:01.3, and 10:00:01.9. Every time the hardware clock ticks, the real time is updated and the tick counter at the head of the list is decremented. When the tick counter becomes zero, a timeout is caused and the node is removed from the list, as shown in Fig. (b). Although this organization requires the list to be scanned when *start*3*timer* or *stop*3*timer* is called, it does not require much work per tick.

In protocol 5, both of these routines have been given a parameter, indicating which frame is to be timed.

9.1.3. A Protocol Using Selective Repeat

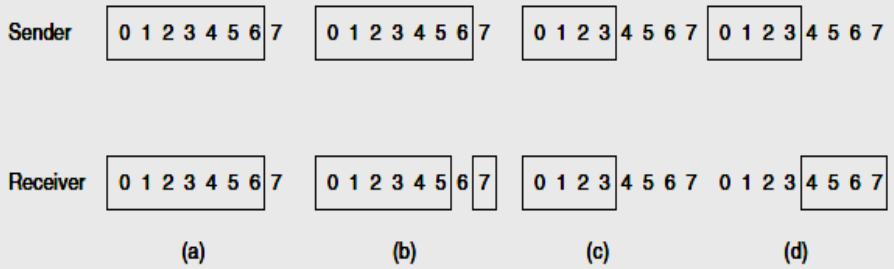
Protocol 5 works well if errors are rare, but if the line is poor, it wastes a lot of bandwidth on retransmitted frames. An alternative strategy for handling errors is to allow the receiver to accept and buffer the frames following a damaged or lost one. Such a protocol does not discard frames merely because an earlier frame was damaged or lost.

In this protocol, both sender and receiver maintain a window of acceptable sequence numbers. The sender's window size starts out at 0 and grows to some predefined maximum, *MAX*3*SEQ*. The receiver's window, in contrast, is always fixed in size and equal to *MAX*3*SEQ*. The receiver has a buffer reserved for each sequence number within its fixed window. Associated with each buffer is a bit (*arrived*) telling whether the buffer is full or empty. Whenever a frame arrives, its sequence number is checked by the function *between* to see if it falls within the window. If so and if it has not already been received, it is accepted and stored.

This action is taken without regard to whether or not it contains the next packet expected by the network layer. Of course,

it must be kept within the data link layer and not passed to the network layer until all the lower-numbered frames have already been delivered to the network layer in the correct order. A protocol using this algorithm is given in the above mentioned figure.

Non sequential receive introduces certain problems not present in protocols in which frames are only accepted in order. We can illustrate the trouble most easily with an example. Suppose that we have a 3-bit sequence number, so that the sender is permitted to transmit up to seven frames before being required to wait for an acknowledgement. Initially, the sender's and receiver's windows are as shown in the following figure (a).



(a) Initial situation with a window of size seven. (b) After seven frames have been sent and received but not acknowledged. (c) Initial situation with a window size of four. (d) After four frames have been sent and received but not acknowledged.

The sender now transmits frames 0 through 6. The receiver's window allows it to accept any frame with sequence number between 0 and 6 inclusive. All seven frames arrive correctly, so the receiver acknowledges them and advances its window to allow receipt of 7, 0, 1, 2, 3, 4, or 5, as shown in Fig. (b). All seven buffers are marked empty.

It is at this point that disaster strikes in the form of a lightning bolt hitting the telephone pole and wiping out all the acknowledgements. The sender eventually times out and retransmits frame 0. When this frame arrives at the receiver, a check is made to see if it falls within the receiver's window. Unfortunately, in Fig. (b) Frame 0 is within the new window, so it will be accepted. The receiver sends a piggybacked acknowledgement for frame 6, since 0 through 6 have been received.

The sender is happy to learn that all its transmitted frames did actually arrive correctly, so it advances its window and immediately sends frames 7, 0, 1, 2, 3, 4, and 5. Frame 7 will be accepted by the receiver and its packet will be passed directly to

the network layer. Immediately thereafter, the receiving data link layer checks to see if it has a valid frame 0 already, discovers that it does, and passes the embedded packet to the network layer. Consequently, the network layer gets Protocol 6 (selective repeat), accepts frames out of order but passes packets to the network layer in order. Associated with each outstanding frame is a timer. When the timer expires, only that frame is retransmitted, not all the outstanding frames, as in protocol 5 an incorrect packet and the protocol fail.

```
 */#define MAX3SEQ 7 /* should be 2^n − 1 */

#define NR3BUFS ((MAX3SEQ + 1)/2)

typedef enum {frame3arrival, cksum3err, timeout, network3layer3ready, ack3timeout} event3type;

#include "protocol.h"

boolean no3nak = true; /* no nak has been sent yet */

seq3nr oldest3frame = MAX3SEQ + 1; /* initial value is only for the simulator */

static boolean between(seq3nr a, seq3nr b, seq3nr c)

{

/* Same as between in protocol5, but shorter and more obscure. */

return ((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a));

}

static void send3frame(frame3kind fk, seq3nr frame3nr, seq3nr frame3expected, packet buffer[ ])

{

/* Construct and send a data, ack, or nak frame. */

frame s; /* scratch variable */

s.kind = fk; /* kind == data, ack, or nak */

if (fk == data) s.info = buffer[frame3nr % NR3BUFS];

s.seq = frame3nr; /* only meaningful for data frames */

s.ack = (frame3expected + MAX3SEQ) % (MAX3SEQ + 1);

if (fk == nak) no3nak = false; /* one nak per frame, please */

to3physical3layer(&s); /* transmit the frame */

if (fk == data) start3timer(frame3nr % NR3BUFS);

stop3ack3timer(); /* no need for separate ack frame */

}
```

```
void protocol6(void)
{
seq3nr ack3expected; /* lower edge of sender's window */
seq3nr next3frame3to3send; /* upper edge of sender's window + 1 */
seq3nr frame3expected; /* lower edge of receiver's window */
seq3nr too3far; /* upper edge of receiver's window + 1 */
int i; /* index into buffer pool */
frame r; /* scratch variable */
packet out3buf[NR3BUFS]; /* buffers for the outbound stream */
packet in3buf[NR3BUFS]; /* buffers for the inbound stream */
boolean arrived[NR3BUFS]; /* inbound bit map */
seq3nr nbuffered; /* how many output buffers currently used */
event3type event;
enable3network3layer(); /* initialize */
ack3expected = 0; /* next ack expected on the inbound stream */
next3frame3to3send = 0; /* number of next outgoing frame */
frame3expected = 0;
too3far = NR3BUFS;
nbuffered = 0; /* initially no packets are buffered */
for (i = 0; i < NR3BUFS; i++) arrived[i] = false;
SEC. 3.4 SLIDING WINDOW PROTOCOLS 225
while (true) {
wait3for3event(&event); /* five possibilities: see event3type above */
switch(event) {
case network3layer3ready: /* accept, save, and transmit a new frame */
nbuffered = nbuffered + 1; /* expand the window */
from3network3layer(&out3buf[next3frame3to3send % NR3BUFS]); /* fetch new packet */
send3frame(data, next3frame3to3send, frame3expected, out3buf);/* transmit the frame */
```

```
inc(next3frame3to3send); /* advance upper window edge */
break;
case frame3arrival: /* a data or control frame has arrived */
from3physical3layer(&r); /* fetch incoming frame from physical layer */
if (r.kind == data) {
/* An undamaged frame has arrived. */
if ((r.seq != frame3expected) && no3nak)
send3frame(nak, 0, frame3expected, out3buf); else start3ack3timer();
if (between(frame3expected, r.seq, too3far) && (arrived[r.seq%NR3BUFS] == false)) {
/* Frames may be accepted in any order. */
arrived[r.seq % NR3BUFS] = true; /* mark buffer as full */
in3buf[r.seq % NR3BUFS] = r.info; /* insert data into buffer */
while (arrived[frame3expected % NR3BUFS]) {
/* Pass frames and advance window. */
to3network3layer(&in3buf[frame3expected % NR3BUFS]);
no3nak = true;
arrived[frame3expected % NR3BUFS] = false;
inc(frame3expected); /* advance lower edge of receiver's window */
inc(too3far); /* advance upper edge of receiver's window */
start3ack3timer(); /* to see if a separate ack is needed */
}
}
}
if((r.kind==nak) && between(ack3expected,(r.ack+1)%(MAX3SEQ+1),next3frame3to3send))
send3frame(data, (r.ack+1) % (MAX3SEQ + 1), frame3expected, out3buf);
while (between(ack3expected, r.ack, next3frame3to3send)) {
nbuffered = nbuffered − 1; /* handle piggybacked ack */
stop3timer(ack3expected % NR3BUFS); /* frame arrived intact */
```

```
inc(ack3expected); /* advance lower edge of sender's window */
}
break;
case cksum3err:
if (no3nak) send3frame(nak, 0, frame3expected, out3buf); /* damaged frame */
break;
case timeout:
send3frame(data, oldest3frame, frame3expected, out3buf); /* we timed out */
break;
case ack3timeout:
send3frame(ack,0,frame3expected, out3buf); /* ack timer expired; send ack */
}
if (nbuffered < NR3BUFS) enable3network3layer(); else disable3network3layer();
}
}
```

The essence of the problem is that after the receiver advanced its window, the new range of valid sequence numbers overlapped the old one. Consequently, the following batch of frames might be either duplicates (if all the acknowledgements were lost) or new ones (if all the acknowledgements were received). The poor receiver has no way of distinguishing these two cases.

The way out of this dilemma lies in making sure that after the receiver has advanced its window, there is no overlap with the original window. To ensure that there is no overlap, the maximum window size should be at most half the range of the sequence numbers, as is done in Fig. (c) and Fig. (d). For example, if 4 bits are used for sequence numbers, these will range from 0 to 15.

Only eight unacknowledged frames should be outstanding at any instant. That way, if the receiver has just accepted frames 0 through 7 and advanced its window to permit acceptance of frames 8 through 15, it can unambiguously tell if subsequent frames are retransmissions (0 through 7) or new ones (8 through 15). In general, the window size for protocol 6 will be (*MAX*3*SEQ* + 1*)*/2. Thus, for 3- bit sequence numbers, the window size is four.

An interesting question is: How many buffers must the receiver have? Under no conditions will it ever accept frames whose sequence numbers are below the lower edge of the window or frames whose sequence numbers are above the upper edge of the window. Consequently, the number of buffers needed is equal to the window size, not to the range of sequence numbers. In the above example of a 4- bit sequence number, eight buffers, numbered 0 through 7, are needed. When frame *i* arrives, it is put in buffer *i* mod 8. Notice that although *i* and (*i* + 8) mod 8 are ‘‘competing’’ for the same buffer, they are never within the window at the same time, because that would imply a window size of at least 9.

For the same reason, the number of timers needed is equal to the number of buffers, not to the size of the sequence space. Effectively, a timer is associated with each buffer. When the timer runs out, the contents of the buffer are retransmitted.

In protocol 5, there is an implicit assumption that the channel is heavily loaded. When a frame arrives, no acknowledgement is sent immediately. Instead, the acknowledgement is piggybacked onto the next outgoing data frame. If the reverse traffic is light, the acknowledgement will be held up for a long period of time. If there is a lot of traffic in one direction and no traffic in the other direction, only *MAX*3*SEQ* packets are sent, and then the protocol blocks, which is why we had to assume there was always some reverse traffic.

In protocol 6 this problem is fixed. After an in-sequence data frame arrives, an auxiliary timer is started by *start*3*ack*3*timer*. If no reverse traffic has presented itself before this timer expires, a separate acknowledgement frame is sent. An interrupt due to the auxiliary timer is called an *ack*3*timeout* event. With this arrangement, one-directional traffic flow is now possible because the lack of reverse data frames onto which acknowledgements can be piggybacked is no longer an obstacle. Only one auxiliary timer exists, and if *start*3*ack*3*timer* is called while the timer is running, it is reset to a full acknowledgement timeout interval. It is essential that the timeout associated with the auxiliary timer be appreciably shorter than the timer used for timing out data frames. This condition is required to make sure a correctly received frame is acknowledged early enough that the frame’s retransmission timer does not expire and retransmit the frame. Protocol 6 uses a more efficient strategy than protocol 5 for dealing with errors. Whenever the receiver has reason to suspect that an error has occurred, it sends a negative acknowledgement (NAK) frame back to the sender. Such a frame is a request for retransmission of the frame specified in the NAK. There are two cases when the receiver should be suspicious: a damaged frame has arrived or a frame

other than the expected one arrived (potential lost frame). To avoid making

Multiple requests for retransmission of the same lost frame, the receiver should keep track of whether a NAK has already been sent for a given frame. The variable *no*3*nak* in protocol 6 is true if no NAK has been sent yet for *frame*3*expected*. If the NAK gets mangled or lost, no real harm is done, since the senders will eventually time out and retransmit the missing frame anyway. If the wrong frame arrives after a NAK has been sent and lost, *no*3*nak* will be true and the auxiliary timer will be started. When it expires, an ACK will be sent to resynchronize the sender to the receiver's current status.

In some situations, the time required for a frame to propagate to the destination, be processed there, and have the acknowledgement come back is (nearly) constant. In these situations, the sender can adjust its timer to be just slightly larger than the normal time interval expected between sending a frame and receiving its acknowledgement. However, if this time is highly variable, the sender is faced with the choice of either setting the interval to a small value (or risking unnecessary retransmissions), or setting it to a large value (and going idle for a long period after an error).

Both choices waste bandwidth. If the reverse traffic is sporadic, the time before acknowledgement will be irregular, being shorter when there is reverse traffic and longer when there is not. Variable processing time within the receiver can also be a problem here. In general, whenever the standard deviation of the acknowledgement interval is small compared to the interval itself, the timer can be set ‘‘tight’’ and NAKs are not useful. Otherwise the timer must be set ‘‘loose,’’ to avoid unnecessary retransmissions, but NAKs can appreciably speed up retransmission of lost or damaged frames.

Closely related to the matter of time outs and NAKs is the question of determining which frame caused a timeout. In protocol 5, it is always *ack*3*expected*, because it is always the oldest. In protocol 6, there is no trivial way to determine who timed out. Suppose that frames 0 through 4 have been transmitted, meaning that the list of outstanding frames is 01234, in order from oldest to youngest. Now imagine that 0 times out, 5 (a new frame) is transmitted, 1 times out, 2 times out, and 6 (another new frame) is transmitted. At this point the list of outstanding frames is 3405126, from oldest to youngest. If all inbound traffic (i.e., acknowledgement- bearing frames) is lost for a while, the seven outstanding frames will time out in that order.

To keep the example from getting even more complicated than it already is, we have not shown the timer administration. Instead, we just assume that the variable *oldest3frame* is set upon timeout to indicate which frame timed out.

9.2. HDLC

High-Level Data Link Control, also know as HDLC, is a bit oriented, switched and non-switched protocol. It is a data link control protocol, and falls within layer 2, the Data Link Layer, of the Open Systems Interface(OSI) model as shown in the following figure.

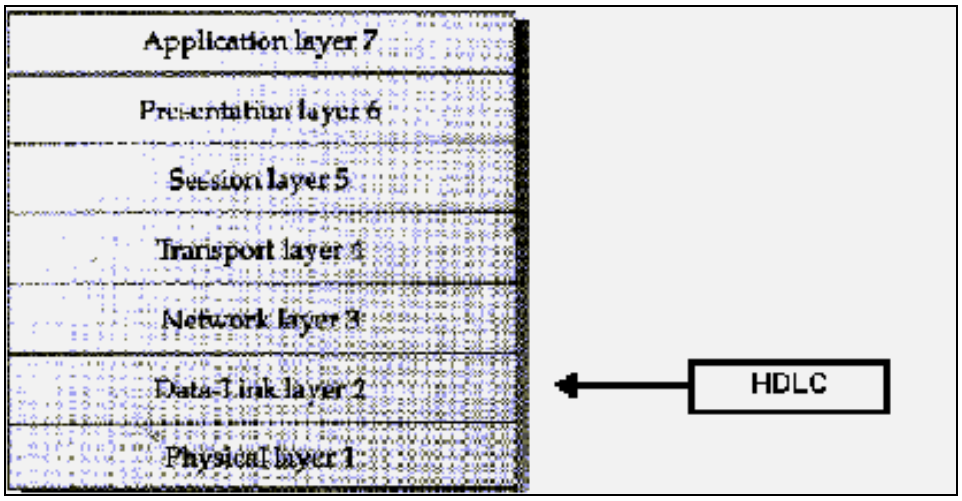


Figure 1

HDLC is a protocol developed by the International Organization for Standardization (ISO). It falls under the ISO standards ISO 3309 and ISO 4335. It has found itself being used throughout the world. It has been so widely implemented because it supports both half duplex and full duplex communication lines, point to point(peer to peer) and multi-point networks, and switched or non-switched channels. The procedures outlined in HDLC are designed to permit synchronous, code-transparent data transmission. Other benefits of HDLC are that the control information is always in the same position, and specific bit patterns used for control differ dramatically from those in representing data, which reduces the chance of errors.

It has also led to many subsets. Two subsets widely in use are Synchronous Data Link Control(SDLC) and Link Access Procedure-Balanced (LAP-B).

This technical overview will be concerned with the following aspects of HDLC:

- Stations and Configurations
- Operational Modes
- Non-Operational Modes
- Frame Structure
- Commands and Responses
- HDLC Subsets(SDLC and LAPB)

9.2.1. HDLC STATIONS AND CONFIGURATIONS

HDLC specifies the following three types of stations for data link control:

- Primary Station
- Secondary Station
- Combined Station

PRIMARY STATION

Within a network using HDLC as it's data link protocol, if a configuration is used in which there is a primary station, it is used as the controlling station on the link. It has the responsibility of controlling all other stations on the link(usually secondary stations). Despite this important aspect of being on the link, the primary station is also responsible for the organization of data flow on the link. It also takes care of error recovery at the data link level(layer 2 of the OSI model).

SECONDARY STATION

If the data link protocol being used is HDLC, and a primary station is present, a secondary station must also be present on the data link. The secondary station is under the control of the primary station. It has no ability, or direct responsibility for controlling the link. It is only activated when requested by the primary station. It only responds to the primary station. The secondary station's frames are called responses. It can only send response frames when requested by the primary station.

COMBINED STATION

A combined station is a combination of a primary and secondary station. On the link, all combined stations are able to send and receive commands and responses without any permission from any other stations on the link. Each combined station is in full control of itself, and does not rely on any other stations on the link. No other stations can control any combined station.

9.2.2. Stations:

HDLC also defines three types of configurations for the three types of stations:

- Unbalanced Configuration
- Balanced Configuration
- Symmetrical Configuration

UNBALANCED CONFIGURATION

The unbalanced configuration in an HDLC link consists of a primary station and one or more secondary stations. The unbalanced occurs because one stations controls the other stations. In a unbalanced configurations, any of the following can be used:

- Full - Duplex or Half - Duplex operation
- Point to Point or Multi-point networks

An example of an unbalanced configuration can be found below in figure 2.a

BALANCED CONFIGURATION

The balanced configuration in an HDLC link consists of two or more combined stations. Each of the stations have equal and complimentary responsibility compared to each other. Balanced configurations can used only the following:

- Full - Duplex or Half - Duplex operation
- Point to Point networks

An example of a balanced configuration can be found below in figure 2.b

Figure 2.a An unbalanced configuration

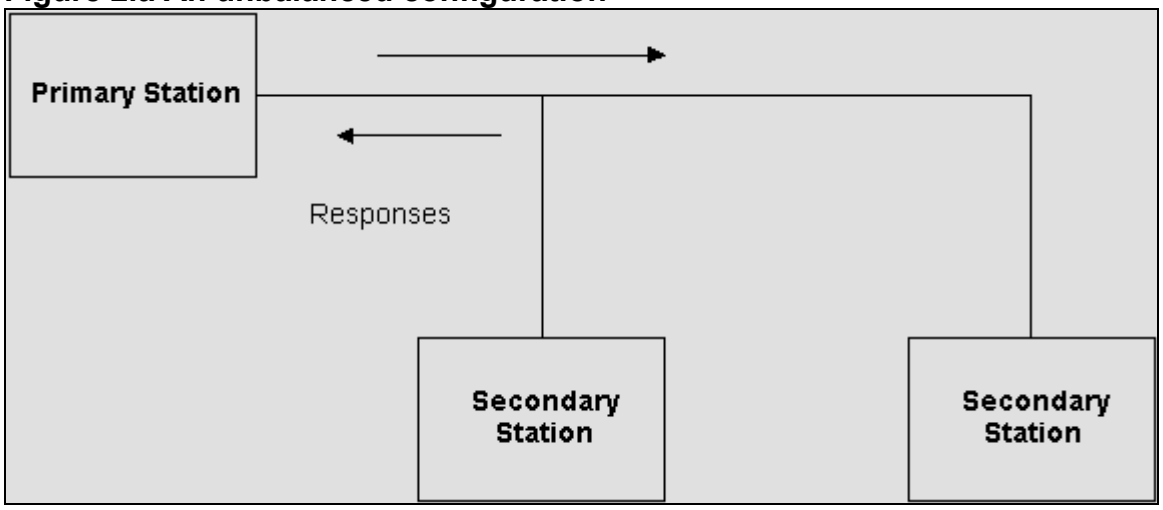
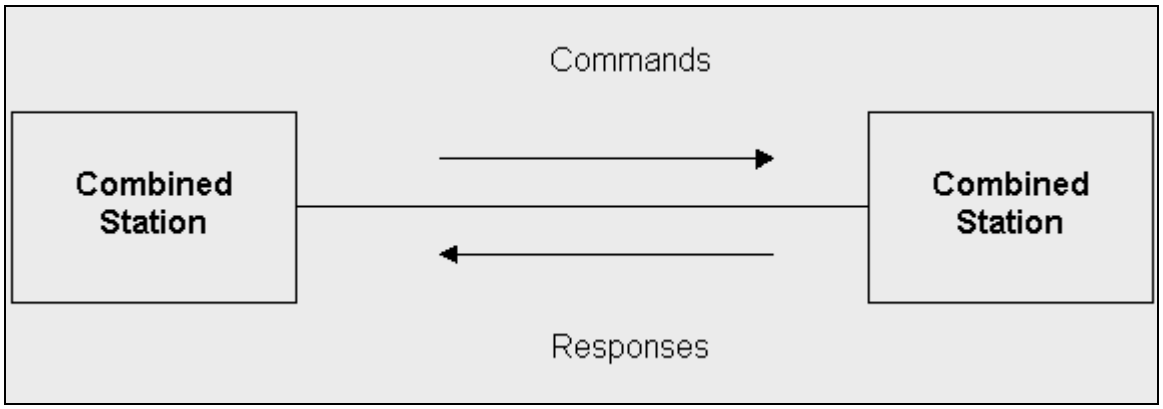


Figure 2.b A balanced configuration



SYMMETRICAL CONFIGURATION

This third type of configuration is not widely in use today. It consists of two independent point to point, unbalanced station configurations. In this configurations, each station has a primary and secondary status. Each station is logically considered as two stations.

9.2.3. HDLC Operational Modes

HDLC offers three different modes of operation. These three modes of operations are:

- Normal Response Mode(NRM)
- Asynchronous Response Mode(ARM)
- Asynchronous Balanced Mode(ABM)

Normal Response Mode

This is the mode in which the primary station initiates transfers to the secondary station. The secondary station can only transmit a response when, and only when, it is instructed to do so by the primary station. In other words, the secondary station must receive explicit permission from the primary station to transfer a response. After receiving permission from the primary station, the secondary station initiates it's transmission. This transmission from the secondary station to the primary station may be much more than just an acknowledgment of a frame. It may in fact be more than one information frame. Once the last frame is transmitted by the secondary station, it must wait once again from explicit permission to transfer anything, from the primary station. Normal Response Mode is only used within an unbalanced configuration.

Asynchronous Response Mode

In this mode, the primary station doesn't initiate transfers to the secondary station. In fact, the secondary station does not have to wait to receive explicit permission from the primary station to transfer any frames. The frames may be more than just acknowledgment frames. They may contain data, or control information regarding the status of the secondary station. This mode can reduce overhead on the link, as no frames need to be

transferred in order to give the secondary station permission to initiate a transfer. However some limitations do exist. Due to the fact that this mode is Asynchronous, the secondary station must wait until it detects and idle channel before it can transfer any frames. This is when the ARM link is operating at half-duplex. If the ARM link is operating at full-duplex, the secondary station can transmit at any time. In this mode, the primary station still retains responsibility for error recovery, link setup, and link disconnection.

Asynchronous Balanced Mode

This mode uses combined stations. There is no need for permission on the part of any station in this mode. This is because combined stations do not require any sort of instructions to perform any task on the link.

Normal Response Mode is used most frequently in multi-point lines, where the primary station controls the link. Asynchronous Response Mode is better for point to point links, as it reduces overhead. Asynchronous Balanced Mode is not used widely today.

The "asynchronous" in both ARM and ABM does not refer to the format of the data on the link. It refers to the fact that any given station can transfer frames without explicit permission or instruction from any other station.

9.2.4. HDLC Non-Operational Modes

HDLC also defines three non-operational modes. These three non-operational modes are:

- Normal Disconnected Mode(NDM)
- Asynchronous Disconnected Mode(ADM)
- Initialization Mode(IM)

The two disconnected modes(NDM and ADM) differ from the operational modes in that the secondary station is logically disconnected from the link(note the secondary station is not physically disconnected from the link). The IM mode is different from the operations modes in that the secondary station's data link control program is in need of regeneration or it is in need of an exchange of parameters to be used in an operational mode.

9.2.5. HDLC Frame Structure

HDLC uses the term "frame" to indicate an entity of data (or a protocol data unit) transmitted from one station to another. Figure below is a graphical representation of a HDLC frame with an information field.

An HDLC frame with an information field.

Field Name	Size(in bits)
Flag Field(F)	8 bits
Address Field(A)	8 bits
Control Field(C)	8 or 16 bits
Information Field(I)	Variable; Not used in some frames
Frame Check Sequence(FCS)	16 or 32 bits
Closing Flag Field(F)	8 bits

THE FLAG FIELD

Every frame on the link must begin and end with a flag sequence field(F). Stations attached to the data link must continually listen for a flag sequence. The flag sequence is an octet looking like 01111110. Flags are continuously transmitted on the link between frames to keep the link active. Two other bit sequences are used in HDLC as signals for the stations on the link. These two bit sequences are:

- Seven 1's, but less than 15 signal an abort signal. The stations on the link know there is a problem on the link.
- 15 or more 1's indicate that the channel is in an idle state.

The time between the transmissions of actual frames is called the interframe time fill. The interframe time fill is accomplished by transmitting continuous flags between frames. The flags may be in 8 bit multiples.

HDLC is a code-transparent protocol. It does not rely on a specific code for interpretation of line control. This means that if a bit at position N in an octet has a specific meaning, regardless of the other bits in the same octet. If an octet has a bit sequence of 01111110, but is not a flag field, HLDC uses a technique called bit-stuffing to differentiate this bit sequence from a flag field. Once the transmitter detects that it is sending 5 consecutive 1's, in inserts a 0 bit to prevent a "phony" flag.

At the receiving end, the receiving station inspects the incoming frame. If it detects 5 consecutive 1's it looks at the next bit. If it is a 0, it pulls it out. If it is a 1, it looks at the 8^{th} bit. If the 8^{th} bit is a 0, it knows an abort or idle signal has been sent. It then proceeds to inspect the following bits to determine appropriate action. This is the manner in which HDLC achieves code-transparency. HDLC is

not concerned with any specific bit code inside the data stream. It is only concerned with keeping flags unique.

THE ADDRESS FIELD

The address field (A) identifies the primary or secondary stations involvement in the frame transmission or reception. Each station on the link has a unique address. In an unbalanced configuration, the A field in both commands and responses refers to the secondary station. In a balanced configuration, the command frame contains the destination station address and the response frame has the sending station's address.

THE CONTROL FIELD

HDLC uses the control field(C) to determine how to control the communications process. This field contains the commands, responses and sequences numbers used to maintain the data flow accountability of the link, defines the functions of the frame and initiates the logic to control the movement of traffic between sending and receiving stations. There three control field formats:

- **Information Transfer Format**

The frame is used to transmit end-user data between two devices.

- **Supervisory Format**

The control field performs control functions such as acknowledgment of frames, requests for re-transmission, and requests for temporary suspension of frames being transmitted. Its use depends on the operational mode being used.

- **Unnumbered Format**

This control field format is also used for control purposes. It is used to perform link initialization, link disconnection and other link control functions.

THE POLL/FINAL BIT(P/F)

The 5^{th} bit position in the control field is called the poll/final bit, or p/f bit. It can only be recognized when it is set to 1. If it is set to 0, it is ignored. The poll/final bit is used to provide dialogue between the primary station and secondary station. The primary station uses P=1 to acquire a status response from the secondary station. The P bit signifies a poll. The secondary station responds to the P bit by transmitting a data or status frame to the primary station with the P/F bit set to F=1. The F bit can also be used to signal the end of a transmission from the secondary station under Normal Response Mode.

THE INFORMATION FIELD

This field is not always in a HDLC frame. It is only present when the Information Transfer Format is being used in the control

field. The information field contains the actually data the sender is transmitting to the receiver.

THE FRAME CHECK SEQUENCE FIELD

This field contains a 16 bit, or 32 bit cyclic redundancy check. It is used for error detection.

9.2.6. HDLC COMMANDS AND RESPONSES

The set of commands and responses in HDLC is summarized in table 1.

INFORMATION TRANSFER FORMAT COMMAND AND RESPONSE

The functions of the information command and response is to transfer sequentially numbered frames, each containing an information field, across the data link.

SUPERVISORY FORMAT COMMANDS AND RESPONSES

Supervisory(S) commands and responses are used to perform numbered supervisory functions such as acknowledgment, polling, temporary suspension of information transfer, or error recovery. Frames with the S format control field cannot contain an information field. A primary station may use the S format command frame with the P bit set to 1 to request a response from a secondary station regarding its status. Supervisory Format commands and responses are as follows:

- *Receive Ready (RR)* is used by the primary or secondary station to indicate that it is ready to receive an information frame and/or acknowledge previously received frames.
- *Receive Not Ready (RNR)* is used to indicate that the primary or secondary station is not ready to receive any information frames or acknowledgments.
- *Reject (REJ)* is used to request the retransmission of frames.
- *Selective Reject (SREJ)* is used by a station to request retransmission of specific frames. An SREJ must be transmitted for each erroneous frame; each frame is treated as a separate error. Only one SREJ can remain outstanding on the link at any one time.

UNNUMBERED FORMAT COMMANDS RESPONSES

The unnumbered format commands and responses are used to extend the number of data link control functions. The unnumbered format frames have 5 modifier bits which allow for up to 32 additional commands and 32 additional response functions.

Below, 13 command functions, and 8 response functions are described.

- *Set Normal Response Mode(SNRM)* places the secondary station into NRM. NRM does not allow the secondary station to send any unsolicited frames. Hence the primary station has control of the link.
- *Set Asynchronous Response Mode (SARM)* allows a secondary station to transmit frames without a poll from the primary station.
- *Set Asynchronous Balanced Mode (SABM)* sets the operational mode of the link to ABM.
- *Disconnect(DISC)* places the secondary station in to a disconnected mode.
- *Set Normal Response Mode Extended (SNRME)* increases the size of the control field to 2 octets instead of one in NRM. This is used for extended sequencing. The same applies for *SARME* and *SABME*.
- *Set Initialization Mode (SIM)* is used to cause the secondary station to initiate a station-specific procedure(s) to initialize its data link level control functions.
- *Unnumbered Poll (UP)* polls a station without regard to sequencing or acknowledgment.
- *Unnumbered Information (UI)* is used to send information to a secondary station.
- *Exchange Identification (XID)* is used to cause the secondary station to identify itself and provide the primary station identifications characteristics of itself.
- *Reset (RSET)* is used to reset the receive state variable in the addressed station.
- *Test (TEST)* is used to cause the addressed secondary station to respond with a TEST response at the first response opportunity. It performs a basic test of the data link control.
- *Unnumbered Acknowledgment (UA)* is used by the secondary station to acknowledge the receipt and acceptance of an *SNRM, SARM, SABM, SNRME, SARME, SABME, RSET, SIM,* or *DISC* commands.
- *Disconnected Mode (DM)* is transmitted from a secondary station to indicate it is in disconnected mode(non-operational mode.)
- *Request Initialization Mode (RIM)* is a request from a secondary station for initialization to a primary station. Once

the secondary station sends *RIM*, it can only respond to *SIM, DSIC, TEST* or *XID* commands.

- *Request Disconnect (RD)* is sent by the secondary station to inform the primary station that it wishes to disconnect from the link and go into a non-operational mode(NDM or ADM).
- *Frame Reject (FRMR)* is used by the secondary station in an operation mode to report that a condition has occurred in transmission of a frame and retransmission of the frame will not correct the condition.

TABLE 1 HDLC Commands and Responses

Information Transfer	**Information Transfer**
Format Commands	**Format Responses**
I - Information	I - Information
Supervisory Format	**Supervisory Format**
Commands	**Responses**
RR - Receive ready	RR - Receive ready
RNR - Receive not ready	RNR - Receive not ready
REJ - Reject	REJ - Reject
SREJ - Selective reject	SREJ - Selective reject
Unnumbered Format	**Unnumbered Format**
Commands	**Commands**
SNRM - Set Normal Response Mode	UA - Unnumbered Acknowledgment
SARM - Set Asynchronous Response Mode	DM - Disconnected Mode
SABM - Set Asynchronous Balanced Mode	RIM - Request Initialization Mode
DISC - Disconnect	RD - Request Disconnect
SNRME - Set Normal Response Mode Extended	UI - Unnumbered Information
SARME - Set Asynchronous Response Mode Extended	XID - Exchange Identification
SABME - Set Asynchronous Balanced Mode Extended	FRMR - Frame Reject
SIM - Set Initialization Mode	TEST - Test
UP - Unnumbered Poll	

UI - Unnumbered Information	
XID - Exchange identification	
RSET - Reset	
TEST - Test	

HDLC SUBSETS

Many other data link protocols have been derived from HDLC. However, some of them reach beyond the scope of HDLC. Two other popular offsets of HDLC are Synchronous Data Link Control (SDLC), and Link Access Protocol, Balanced(LAP-B). SDLC is used and developed by IBM. It is used in a variety of terminal to computer applications. It is also part of IBM's SNA communication architecture. LAP-B was developed by the ITU-T. It is derived mainly from the asynchronous response mode (ARM) of HDLC. It is commonly used for attaching devices to packet-switched networks.

10

THE MEDIUM ACCESS SUB LAYER

Under this unit, we are going to discuss the importance of the medium access sub layer and the protocols support the functionalities. Following Protocols are going to be discussed:

Unit Structure

10.1 Multiple Access Protocols
10.1.1ALOHA (Pure, slotted, reservation)
10.2 Carrier Sense Multiple Access Protocols (CSMA)
Collision free Protocols
IEEE Standard 802.3, 802.4, 802.5, 802.6
10.3 High speed LANs – FDDI
10.4 Satellite Networks – Polling, ALOHA, FDMA, TDMA, CDMA
10.5 Categories of satellites – GEO, MEO, LEO

10.1 MULTIPLE ACCESS PROTOCOLS

In telecommunications and computer networks, a **channel access method** or **multiple access method** allows several terminals connected to the same multi-point transmission medium to transmit over it and to share its capacity. Examples of shared physical media are wireless networks, bus networks, ring networks, hub networks and half-duplex point-to-point links.

A channel-access scheme is based on a multiplexing method that allows several data streams or signals to share the same communication channel or physical medium. Multiplexing is in this context provided by the physical layer. Note that multiplexing also may be used in full-duplex point-to-point communication between nodes in a switched network, which should not be considered as multiple access.

A channel-access scheme is also based on a multiple access protocol and control mechanism, also known as media access control (MAC). This protocol deals with issues such as addressing, assigning multiplex channels to different users, and avoiding collisions. The MAC-layer is a sub-layer in Layer 2 (Data Link Layer) of the OSI model and a component of the Link Layer of the TCP/IP model.

Let us see the protocols that support multiple access:

10.1.1 ALOHA:

Aloha, also called the Aloha method, refers to a simple communications scheme in which each source (transmitter) in a network sends data whenever there is a frame to send. If the frame successfully reaches the destination (receiver), the next frame is sent. If the frame fails to be received at the destination, it is sent again. This protocol was originally developed at the University of Hawaii for use with satellite communication systems in the Pacific.

In a wireless broadcast system or a half-duplex two-way link, Aloha works perfectly. But as networks become more complex, for example in an Ethernet system involving multiple sources and destinations in which data travels many paths at once, trouble occurs because data frames collide (conflict). The heavier the communications volume, the worse the collision problems become. The result is degradation of system efficiency, because when two frames collide, the data contained in both frames is lost.

To minimize the number of collisions, thereby optimizing network efficiency and increasing the number of subscribers that can use a given network, a scheme called slotted Aloha was developed. This system employs signals called beacons that are sent at precise intervals and tell each source when the channel is clear to send a frame. Further improvement can be realized by a more sophisticated protocol called **Carrier Sense Multiple Access with Collision Detection (CSMA).**

History about ALOHA:

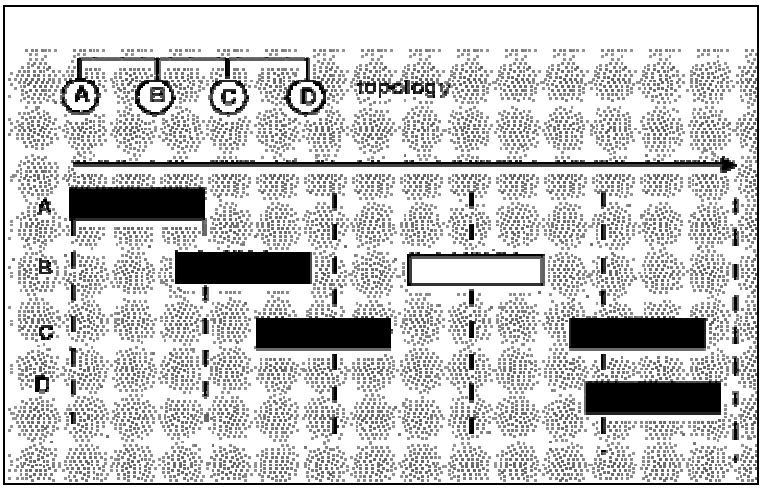
In 1070s, Norman Abranson and his colleagues at the University of Hawaii devised a new and elegant method to solve the channel allocation problem. Many researchers have extended their work since then. Although Abranson's work, called the Aloha System, used ground-based radio broadcasting, the basic idea is applicable to any system in which uncoordinated users are completing for the use of a single shared channel.

The two several of ALOHA are:

- PURE ALOHA
- SLOTTED ALOHA

Pure Aloha:

- Very simple one.
- Send the packet. In case of collision try to resend it.



Analyzing the Aloha Protocol:

Goal: quantitative understanding of performance of Aloha protocol

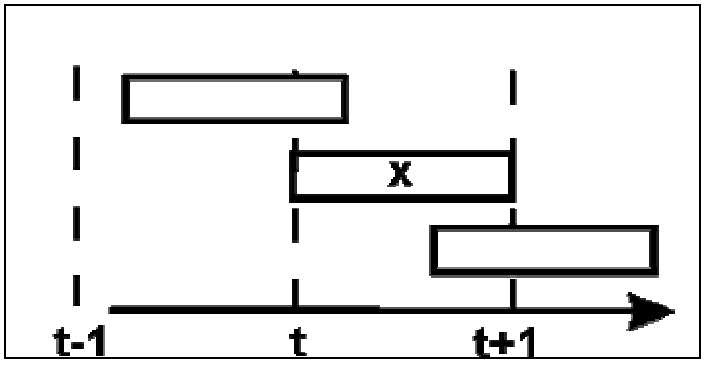
- Supports fixed length Packets.
- Packet transmission time is unit of time.
- Throughput S: number Packets successfully (without collision) transmitted per unit time.
- Offered load G: number Packet transmissions attempted per unit time.
    - note: S<G, but S depends on G

Poisson model: probability of k Packet transmission attempts in t time units:

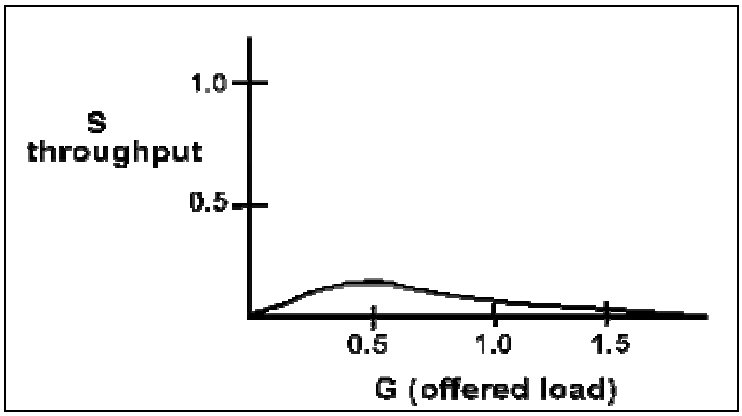
Prob[k trans in t] = ((Gt)k)(e**(-Gt))/(k!)**

- capacity of multiple access protocol: maximum value of S over all values of G

Focus on a given attempted packet transmission



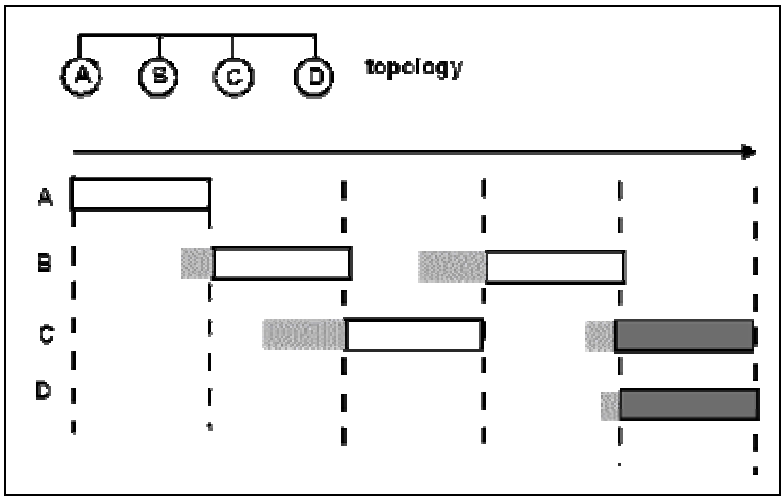
Pure Aloha throughput



Note: maximum throughput is 18% of physical channel capacity you buy 1 Mb link, throughput will never be more than 180Kb!

Slotted Aloha:

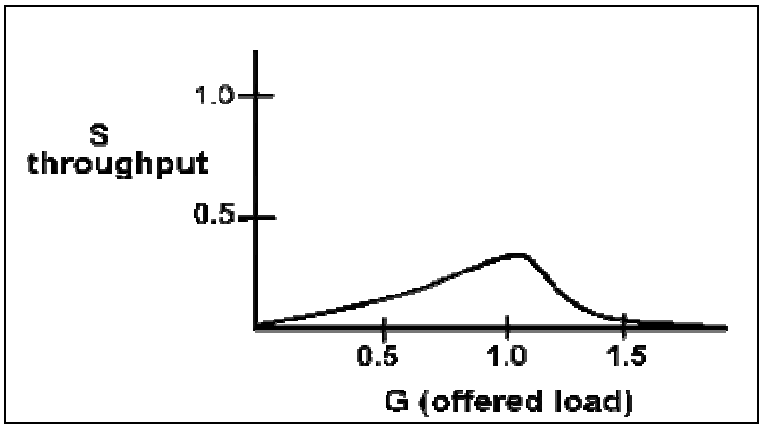
- It works as synchronous system where time divided into slots.
- The slot size equals fixed packet transmission time.
- When Packet is ready for transmission, wait until start of next slot.
- Thus, packets overlap completely or not at all.



Slotted Aloha performance:

S = G*prob[no other transmissions overlap]
= G*prob[0 other attempted transmissions]
= G*prob[0 other arrivals in previous slot]
= Ge**(-G)

Slotted Aloha Throughput:



Conclusion about the performance of Aloha: Aloha is inefficient (and rude!): doesn't listen before talking!

10.2 CARRIER SENSE MULTIPLE ACCESS: CSMA

Carrier Sense Multiple Access (**CSMA**) is a probabilistic Media Access Control (MAC) protocol in which a node verifies the absence of other traffic before transmitting on a shared transmission medium, such as an electrical bus, or a band of the electromagnetic spectrum.

"**Carrier Sense**" describes the fact that a transmitter uses feedback from a receiver that detects a carrier wave before trying to send. That is, it tries to detect the presence of an encoded signal from another station before attempting to transmit. If a carrier is sensed, the station waits for the transmission in progress to finish before initiating its own transmission.

"**Multiple Access**" describes the fact that multiple stations send and receive on the medium. Transmissions by one node are generally received by all other stations using the medium.

In our analysis the access methods to control multichannel system and the data multichannel system is based on slotted ALOHA protocol. Both control and data channels use the same time reference which we call **cycle**. We define as cycle, the time interval that includes onetime unit for control packet transmission followed by a data packet transmission period. Thus the cycle time duration is **T=L+1** time units. The stations are synchronized for the transmission on the control and data packet during a cycle. A station generating or retransmitting a data packet, waits the beginning of the next cycle, selects randomly one of the v wavelengths

λ_{c1},..., $\lambda_{}$and sends a control packet over the λ_{cm} control channel at the first time unit of the cycle. The control packet is consisting of the transmitter address, the receiver address and the wavelength, λ_{dk}, of the data channel. Immediately after transmission of the control packet the data packet is transmitted over λ_{dk} wavelength data channel.

Types of CSMA

- **1-persistent CSMA**

When the sender (station) is ready to transmit data, it checks if the physical medium is busy. If so, it senses the medium continually until it becomes idle, and then it transmits a piece of data (a frame). In case of a collision, the sender waits for a random period of time and attempts to transmit again.

- **p-persistent CSMA**

When the sender is ready to send data, it checks continually if the medium is busy. If the medium becomes idle, the sender transmits a frame with a probability *p*. If the station chooses not to transmit (the probability of this event is *1-p*), the sender waits until the next available time slot and transmits again with the same probability *p*. This process repeats until the frame is sent or some other sender stops transmitting. In the latter case the sender monitors the channel, and when idle, transmits with a probability *p*, and so on.

- **o-persistent CSMA**

Each station is assigned a transmission order by a supervisor station. When medium goes idle, stations wait for their time slot in accordance with their assigned transmission order. The station assigned to transmit first transmits immediately. The station assigned to transmit second waits one time slot (but by that time the first station has already started transmitting). Stations monitor the medium for transmissions from other stations and update their assigned order with each detected transmission (i.e. they move one position closer to the front of the queue).

Protocol modifications/variations:

Carrier Sense Multiple Access With Collision Detection (CSMA/CD) is a modification of CSMA. CSMA/CD is used to improve CSMA performance by terminating transmission as soon as a collision is detected, and reducing the probability of a second collision on retry.

Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) is a modification of CSMA. Collision avoidance is used to improve the performance of CSMA by attempting to be less "greedy" on the channel. If the channel is sensed busy before transmission then the transmission is deferred for a "random" interval. This reduces the probability of collisions on the channel.

10.3 HIGH SPEED LANS (FDDI)

Function and Frame Format of FDDI

Fiber Distributed-Data Interface (FDDI) :

FDDI (Fiber-Distributed Data Interface) is a standard for data transmission on fiber optic lines in that can extend in range up to 200 km (124 miles). The FDDI protocol is based on the token ring protocol. In addition to being large geographically, an FDDI local area network can support thousands of users.

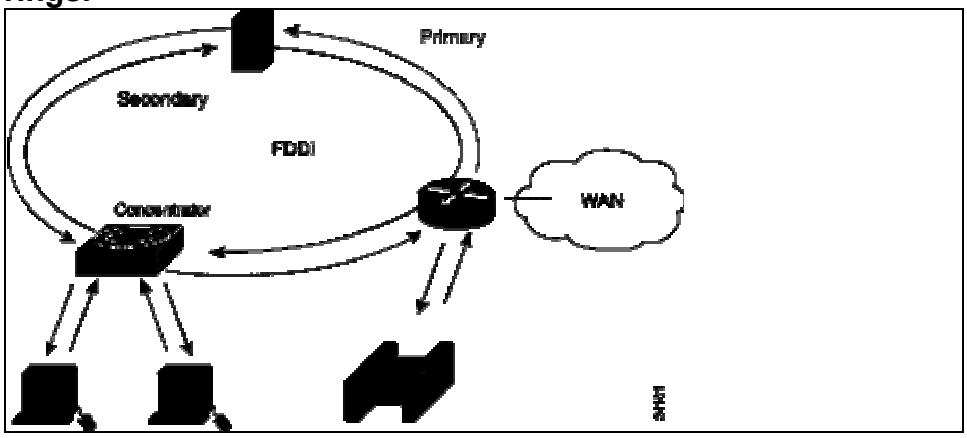
An FDDI network contains two token rings, one for possible backup in case the primary ring fails. The primary ring offers up to 100 Mbps capacity. If the secondary ring is not needed for backup, it can also carry data, extending capacity to 200 Mbps. The single ring can extend the maximum distance; a dual ring can extend 100 km (62 miles).

FDDI is a product of American National Standards Committee X3-T9 and conforms to the open system interconnect (OSI) model of functional layering. It can be used to interconnect LANs using other protocols. FDDI-II is a version of FDDI that adds the capability to add circuit-switched service to the network so that voice signals can also be handled. Work is underway to connect FDDI networks to the developing Synchronous Optical Network.

Function of FDDI
Background
The Fiber Distributed Data Interface (FDDI) specifies a 100-Mbps token-passing, dual-ring LAN using fiber-optic cable. FDDI is frequently used as high-speed backbone technology because of its support for high bandwidth and greater distances than copper. It should be noted that relatively recently, a related copper specification, called Copper Distributed Data Interface (CDDI) has emerged to provide 100-Mbps service over copper. CDDI is the implementation of FDDI protocols over twisted-pair copper wire. This chapter focuses mainly on FDDI specifications and operations, but it also provides a high-level overview of CDDI.

FDDI uses a dual-ring architecture with traffic on each ring flowing in opposite directions (called counter-rotating). The dual-rings consist of a primary and a secondary ring. During normal operation, the primary ring is used for data transmission, and the secondary ring remains idle. The primary purpose of the dual rings, as will be discussed in detail later in this chapter, is to provide superior reliability and robustness. Figure 1 shows the counter-rotating primary and secondary FDDI rings.

Figure 1: FDDI uses counter-rotating primary and secondary rings.



FDDI Specifications

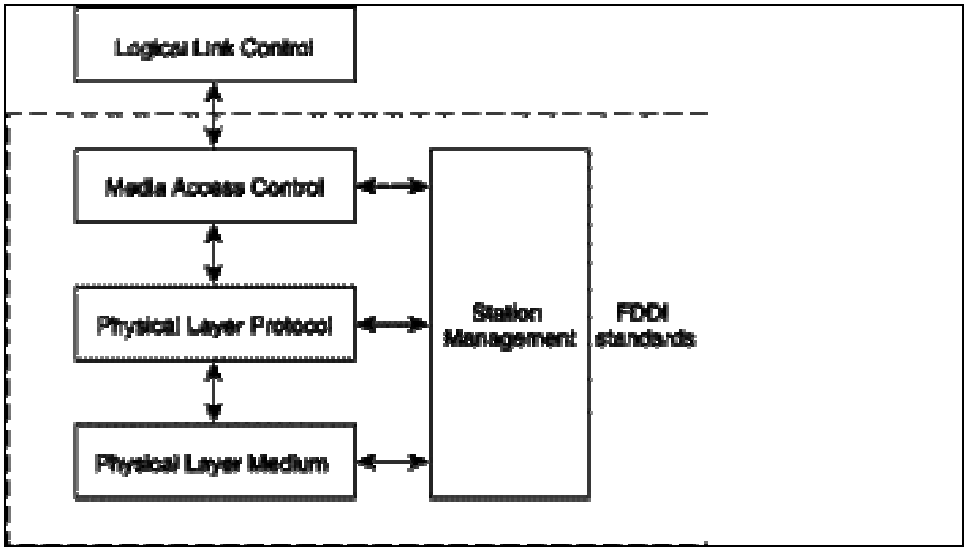
FDDI specifies the physical and media-access portions of the OSI reference model. FDDI is not actually a single specification, but it is a collection of four separate specifications each with a specific function. Combined, these specifications have the capability to provide high-speed connectivity between upper-layer protocols such as TCP/IP and IPX, and media such as fiber-optic cabling.

FDDI's four specifications are the Media Access Control (MAC), Physical Layer Protocol (PHY), Physical-Medium Dependent (PMD), and Station Management (SMT). The MAC specification defines how the medium is accessed, including frame format, token handling, addressing, algorithms for calculating cyclic redundancy check (CRC) value, and error-recovery mechanisms. The PHY specification defines data encoding/decoding procedures, clocking requirements, and framing, among other functions. The PMD specification defines the characteristics of the transmission medium, including fiber-optic links, power levels, bit-error rates, optical components, and connectors. The SMT specification defines FDDI station configuration, ring configuration, and ring control features, including station insertion and removal, initialization, fault isolation and recovery, scheduling, and statistics collection.

FDDI is similar to IEEE 802.3 Ethernet and IEEE 802.5 Token Ring in its relationship with the OSI model. Its primary purpose is to

provide connectivity between upper OSI layers of common protocols and the media used to connect network devices. Figure 3 illustrates the four FDDI specifications and their relationship to each other and to the IEEE-defined Logical-Link Control (LLC) sublayer. The LLC sublayer is a component of Layer 2, the MAC layer, of the OSI reference model.

Figure 2: FDDI specifications map to the OSI hierarchical model.



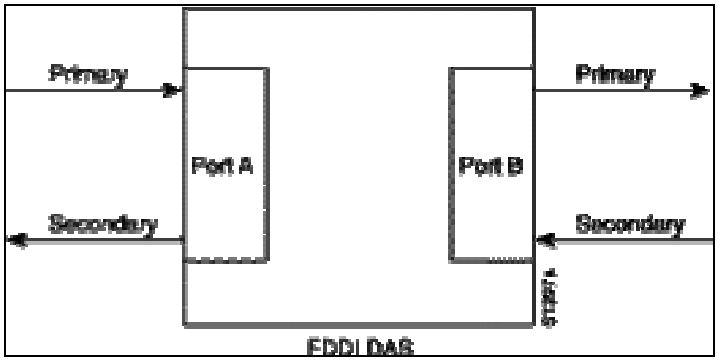
FDDI Station-Attachment Types

One of the unique characteristics of FDDI is that multiple ways actually exist by which to connect FDDI devices. FDDI defines three types of devices: single-attachment station (SAS), dual-attachment station (DAS), and a concentrator.

An SAS attaches to only one ring (the primary) through a concentrator. One of the primary advantages of connecting devices with SAS attachments is that the devices will not have any effect on the FDDI ring if they are disconnected or powered off. Concentrators will be discussed in more detail in the following discussion.

Each FDDI DAS has two ports, designated A and B. These ports connect the DAS to the dual FDDI ring. Therefore, each port provides a connection for both the primary and the secondary ring. As you will see in the next section, devices using DAS connections will affect the ring if they are disconnected or powered off. Figure 3 shows FDDI DAS A and B ports with attachments to the primary and secondary rings.

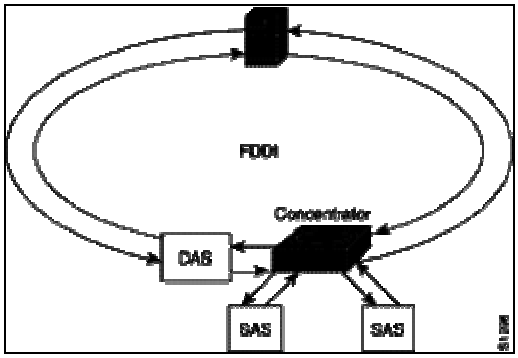
Figure 3: FDDI DAS ports attach to the primary and secondary rings.



An FDDI concentrator (also called a *dual-attachment concentrator* [DAC]) is the building block of an FDDI network. It attaches directly to both the primary and secondary rings and ensures that the failure or power-down of any SAS does not bring down the ring. This is particularly useful when PCs, or similar devices that are frequently powered on and off, connect to the ring. Figure 4 shows the ring attachments of an FDDI SAS, DAS, and concentrator.

TOP

Figure 4: A concentrator attaches to both the primary and secondary rings.



FDDI Fault Tolerance

FDDI provides a number of fault-tolerant features. In particular, FDDI's dual-ring environment, the implementation of the optical bypass switch, and dual-homing support make FDDI a resilient media technology.

Dual Ring

FDDI's primary fault-tolerant feature is the dual ring. If a station on the dual ring fails or is powered down, or if the cable is damaged, the dual ring is automatically *wrapped* (doubled back onto itself) into

a single ring. When the ring is wrapped, the dual-ring topology becomes a single-ring topology. Data continues to be transmitted on the FDDI ring without performance impact during the wrap condition. Figure 5 and Figure 6 illustrate the effect of a ring wrapping in FDDI.

Figure 5: A ring recovers from a station failure by wrapping.

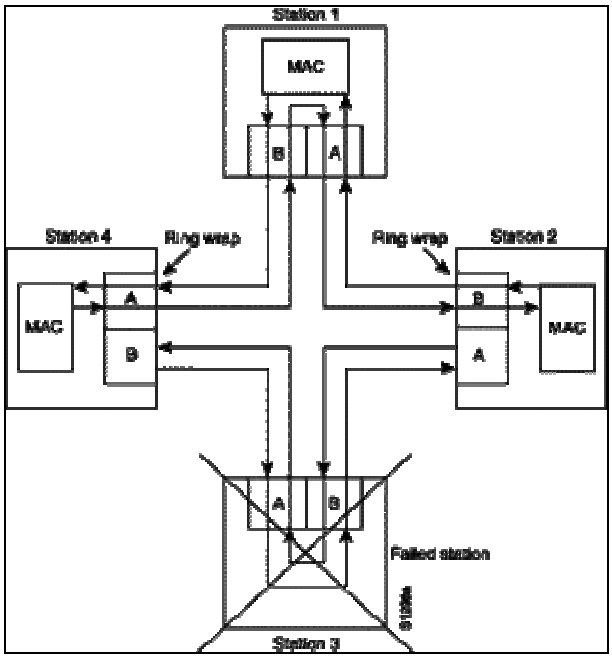
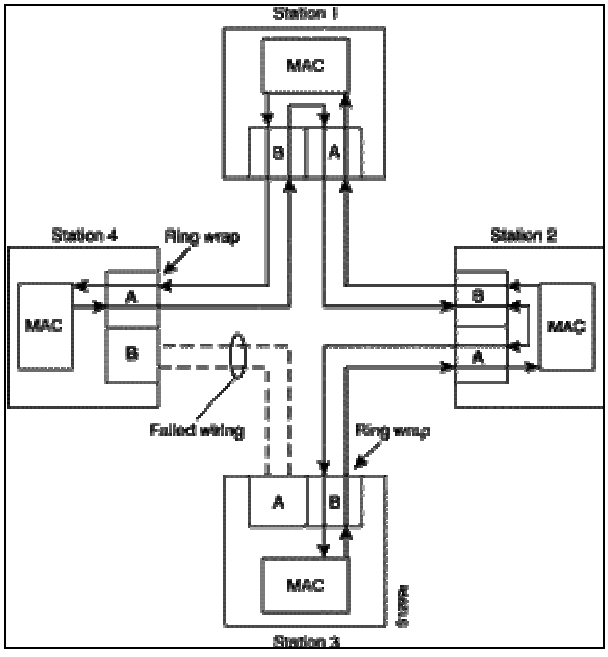


Figure 6: A ring also wraps to withstand a cable failure.



When a single station fails, as shown in Figure 5, devices on either side of the failed (or powered down) station wrap, forming a single ring. Network operation continues for the remaining stations

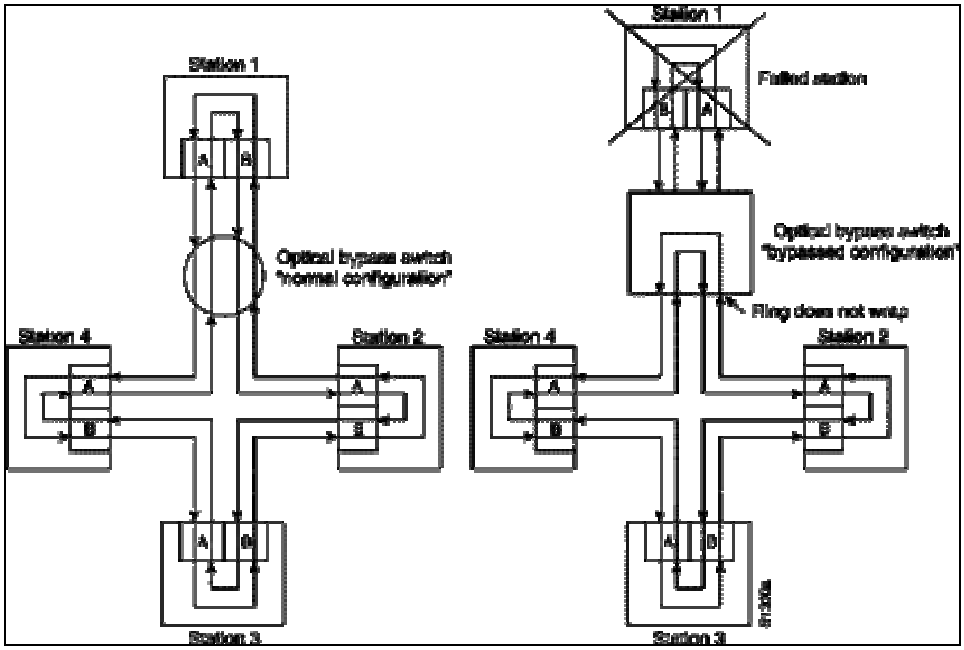
on the ring. When a cable failure occurs, as shown in Figure 6, devices on either side of the cable fault wrap. Network operation continues for all stations.

It should be noted that FDDI truly provides fault-tolerance against a single failure only. When two or more failures occur, the FDDI ring segments into two or more independent rings that are unable to communicate with each other.

Optical Bypass Switch

An optical bypass switch provides continuous dual-ring operation if a device on the dual ring fails. This is used both to prevent ring segmentation and to eliminate failed stations from the ring. The optical bypass switch performs this function through the use of optical mirrors that pass light from the ring directly to the DAS device during normal operation. In the event of a failure of the DAS device, such as a power-off, the optical bypass switch will pass the light through itself by using internal mirrors and thereby maintain the ring's integrity. The benefit of this capability is that the ring will not enter a wrapped condition in the event of a device failure. Figure 7 shows the functionality of an optical bypass switch in an FDDI network.

Figure 7: The optical bypass switch uses internal mirrors to maintain a network.

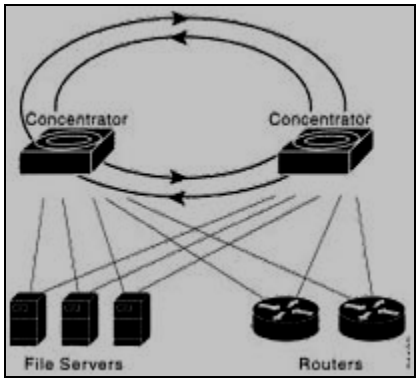


Dual Homing

Critical devices, such as routers or mainframe hosts, can use a fault-tolerant technique called *dual homing* to provide additional redundancy and to help guarantee operation. In dual-

homing situations, the critical device is attached to two concentrators. Figure 8 shows a dual-homed configuration for devices such as file servers and routers.

Figure 8: A dual-homed configuration guarantees operation.

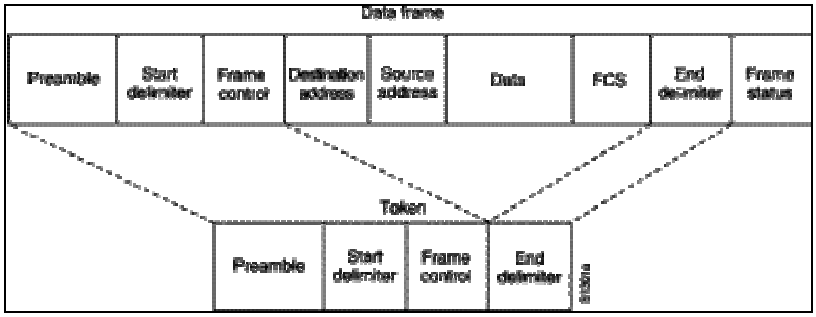


One pair of concentrator links is declared the active link; the other pair is declared passive. The passive link stays in back-up mode until the primary link (or the concentrator to which it is attached) is determined to have failed. When this occurs, the passive link automatically activates.

FDDI Frame Format

The FDDI frame format is similar to the format of a Token Ring frame. This is one of the areas where FDDI borrows heavily from earlier LAN technologies, such as Token Ring. FDDI frames can be as large as 4,500 bytes. Figure 9 shows the frame format of an FDDI data frame and token.

Figure 9: The FDDI frame is similar to that of a Token Ring frame.



FDDI Frame Fields

The following descriptions summarize the FDDI data frame and token fields illustrated in Figure 9.

Preamble---A unique sequence that prepares each station for an upcoming frame.

Start Delimiter---Indicates the beginning of a frame by employing a signaling pattern that differentiates it from the rest of the frame.

Frame Control---Indicates the size of the address fields and whether the frame contains asynchronous or synchronous data, among other control information.

Destination Address---Contains a unicast (singular), multicast (group), or broadcast (every station) address. As with Ethernet and Token Ring addresses, FDDI destination addresses are 6 bytes long.

Source Address---Identifies the single station that sent the frame. As with Ethernet and Token Ring addresses, FDDI source addresses are 6 bytes long.

Data---Contains either information destined for an upper-layer protocol or control information.

Frame Check Sequence (FCS)---Filed by the source station with a calculated *cyclic redundancy check* value dependent on frame contents (as with Token Ring and Ethernet). The destination address recalculates the value to determine whether the frame was damaged in transit. If so, the frame is discarded.

End Delimiter---Contains unique symbols, which cannot be data symbols, that indicate the end of the frame.

Frame Status---Allows the source station to determine whether an error occurred and whether the frame was recognized and copied by a receiving station.

FDDI Frame Format

FDDI Frame

Frame Control (FC): 8 bits

has bit format CLFFZZZZ

C indicates synchronous or asynchronous frame

L indicates use of 16 or 48 bit addresses

FF indicates whether it is a LLC, MAC control or reserved frame

in a control frame ZZZZ indicates the type of control

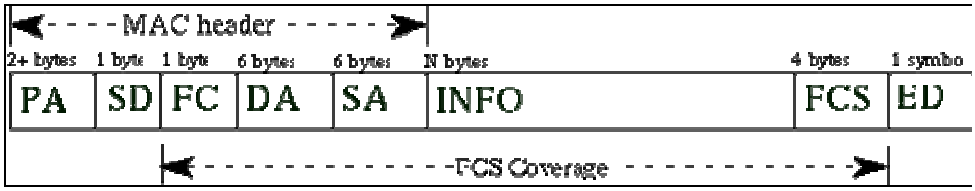
Destination Address (DA): 16 or 48 bits

specifies station for which the frame is intended

Source Address (SA): 16 or 48 bits

specifies station that sent the frame

Here is what the FDDI frame format looks like:



FDDI Frame Format

PA - Preamble 16 symbols

SD - Start Delimiter 2 symbols

FC - Frame Control 2 symbols

DA - Destination Address 4 or 12 symbols

SA - Source Address 4 or 12 symbols

FCS - Frame Check Sequence 8 symbols, covers the FC, DA, SA and Information

ED - End Delimiter 1 or 2 symbols

FS - Frame Status 3 symbols Token is just the PA, SD, FC and ED

Preamble

The Token owner as a minimum of transmits the preamble 16 symbols of Idle. Physical Layers of the subsequent repeating stations can change the length of the Idle pattern according to the Physical Layer requirements. Therefore, each repeating station may see a variable length preamble from the original preamble. Tokens will be recognized as long as its preamble length is greater than zero. If a valid token is received and cannot be processed (repeated), due to expiration of ring timing or latency constraints the station will issue a new token to be put on the ring.

A given MAC implementation is not required to be capable of copying frames received with less than 12 symbols of preamble; Nevertheless, with such frames, it cannot be correctly repeated.

Since the preamble cannot be repeated, the rest of the frame will not be repeated as well.

Starting Delimiter

This field of the frame denodes the start of the frame. It can only have symbols 'J' and 'K'. These symbols will not be used anywhere else but in the starting delimiter of a token or a frame.

Frame Control

Frame Control field descibes what type of data it is carrying inthe INFO field. Here are the most common values that are allowed in the FC field:

40: Void Frame.

41,4F: Station Management (SMT) Frame.

C2,C3: MAC Frame.

50,51: LLC Frame.

60: Implementor Frame.

70: Reserved Frame.

Please note that the list here are only the most common values that can be formed by a 48 bit addressed synchronous data frames.

Destination Address

Destination Address field contains 12 symbols that identifies the station that is receiving this particular frame. When FDDI is first setup, each station is given a unique address that identifies themselves from the others. When a frame passed by the station, the station will compare its address against the DA field of the frame. If it is a match, station then copies the frame into its buffer area waiting to be processed. There is not restriction on the number of stations that a frame can reach at a time. If the first bit of the DA field is set to '1', then the address is called a *group* or *global* address. If the first bit is '0', then the address is called*individual* address. As the name suggests, a frame with a *global* address setting can be sent to multiple stations on the network. If the frame is intended for everyone on the network, the address bits will be set to all 1's. Therefore, a global address contains all 'F' symbols. There are also two different ways of administer these addresses. One's called *local* and the other's called*universal*. The second bit of the address field determine whether or not the address is locally or universally administered. If the second bit is '1' then it is locally administered address. If the second bit is a '0', then it is universally administered adress.A locally administer address are addresses that havebeen assigned by the network administrator and a universally administered addresses are pre-assigned by the manufacturer's OUI.

Source Address

A Source Address identifies the station that created the frame. This field is used for remove frames from the ring. Each time a frame is sent, it travels around the ring, visiting each station,and eventually (hopefully) comes back to the station that originally sent that frame. If the address of a station matches the SA field inthe frame, the station will strip the frame off the ring. Each stationis responsible for removing its own frame from the ring.

Information Field

INFO field is the heart and soul of the frame. Every components of the frame is designed around this field; Who to send it to, where's this coming from, how it is received and so on.The type of information in the INFO field can be found by looking in the FC field of the frame. For example:'50'(hex) denodes a LLC frame. So, the INFO field will have a LLC header followed by other upper layer headers. For example SNAP, ARP, IP, TCP, SNMP, etc.'41'(hex or '4F'(hex) denode s SMT (Station Management) frame. Therefore, a SMT header will appear in the INFO field.

Frame Check Sequence

Frame Check Sequence field is used to check or verify the traversingframe for any bit errors. FCS information is generated by the stationthat sends the frame, using the bits in FC, DA, SA, INFO, and FCSfields. To verify if there are any bit errors in the frame, FDDI uses 8 symbols (32 bits) CRC (Cyclic Redundancy Check) to ensure the transmission of a frame on the ring.

End Delimiter

As the name suggests, the end delimiter denodes the end of the frame. The ending delimiter consist of a 'T' symbol. This 'T' symbols indicates that the frame is complete or ended. Any data sequence that does not end with this 'T' symbol is not considered to be a frame.

Frame Status

Frame Status (FS) contains 3 indicators that dictates the condition of the frame. Each indicator can have two values: *Set* ('S') or Reset ('R'). The indicators could possibly be corrupted. In this case, the indicators is neither 'S' nor 'R'. All frame are initially set to 'R'.Three types of indicators are as follows:Error (E):This indicator is set if a station determines an error for that frame. Might be a CRC failiure or other causes. If a frame has its E indicator set, then, that frame is discarded by the first station that encounters the frame.Acknowledge(A): Sometime this indicator is called 'address recognized'. This indicator is set whenever a frame in properly received; meaning the frame has reached its destination address. Copy (C): This indicator is set whenever a station is able to copy the received frame into its buffer section. Thus, Copy and Acknowledge indicators are usually set at the same time. But, sometimes when a station is receiving too many frames and cannot copy all the incoming frames. If this happens, it would re-transmit the frame with indicator 'A' set indicator 'C' left on reset.

Communications, Protocols and Polling on Distributed Satellite-Local Networks

Distributed local area networks can be organized in many different fashions. Some of the approaches used for network configurations are:

- Star network--Each device is connected separately to a central controlling point in the network.
- Bus network (or multidrop)--Places all the devices in the network together on a single shared line.
- Ring network (or loop)--Where each device is connected to each other in a closed circular fashion, providing two pathways around the ring, going in either direction. Breaking the ring at one spot would not affect the networks' functionally.

A feasible approach in providing a satellite network protocol would be a slotted Aloha protocol technique. To describe briefly its approach, one should know that the following must be employed:

- Establish a synchronized time interval over a satellite channel, according to the time to transmit the length of the largest packet message.
- Broadcast a packet only at the beginning of this discrete-time interval, for example every 20 milliseconds.
- Provide a round-robin polling approach, with the ability to reprogram the order of poll and the priority.

The packer itself would contain a hello/message/goodbye structure. Collisions occasionally occur due to quarter-second (250 ms) delays in transmissions. Uplinks would listen to their own messages and compare with that was transmitted; if a collision has occurred, then it would retry the transmission when its time slice comes up again.

"Packer narrowcasting" would be employed, which implies that the packet contains source and destination addresses, as well as a repetition counter for repeat transmits of the same packets.

"Capture effect" could also be used, where different uplinks would broadcast signals at different power levels, thereby providing a pseudo-priority basis for resolving collision conflicts.

In other words, stronger signals to the satellite provide a higher priority by increasing the probability that its packet would get through. If the highest priority makes it through the satellite transponder successfully, then you have one less uplink packet to retransmit in the event of a collision.

In a local area network, polling performs a request for status in a round-robin fashion, over a twisted-wire pair cable. Each poll

expects a response, within a specified time frame, form the device. In order that future requirements be accommodated, the current polling module being used for projects has been designed to allow for the polling of SA buses (or, in fact, any type bus interface-able through a DEC-compatible interface board, RS-232C, RS-422 and others) in any order, as well as poll the devices on a bus in any order.

Satellite Networks-ALOHA

Aloha Satellites offers an array of competitive communications services including bulk basic analog video packages, bulk basic digital video packages, premium video programming services, access to hundreds of additional a-la-carte digital video services, local network broadcast channels, pay-per-view video programming, HDTV, interactive TV guide, CD quality audio programming, broadband high-speed internet and access to Voice-over-IP (VoIP). The possible combinations of services are endless.

There are many factors that must be considered prior to determining the right package for your situation including but not limited to property size, property type (apartment, condo, hotel, etc.) construction type (high-rise or garden style), number of outlets, type and condition of existing wiring, etc.). Then conduct an in-depth site analysis to gather critical site data that will be used as a basis to estimate system build-out costs and determine the economic feasibility of the proposed system.

While private cable operators adopts and delivers “one size fits all" approach to your commercial business or property, Aloha Satellites, adopts and delivers the approach to your business not as a number but as an individual entity with its own distinct individual needs. The effort is taken to deliver a solution that is custom fit to your application .Thus; Aloha Satellites will deliver the most competitive combination of quality products, service and price.

Satellite delivered digital video services are enjoyed by more than 30 million subscribers daily, more than any single cable TV company or telephone company. Although private cable operators may not like to admit it, their video services are also delivered to their headed via satellite. The difference is in the delivery of the video signals. In Aloha Satellites, video signals are delivered directly from the satellite to your business or property. After receiving and processing the satellite video signals, the cable TV company must then send these signals through a distribution network consisting of hundreds of miles of overhead and underground cable plant, thousands of active and passive electronic devices, thousands of connectors, dozens of power supplies and multiple power grids before finally arriving at your business or property. Outages or compromised service may occur

at anytime and anywhere in this maze as a result of faulty active or passive devices, loose connectors, water damage, cut cables or fiber, power outages, automobile accidents, rodents, equipment not properly balanced, corrosion, acts of God, and so on. Thus the delivery system of Aloha Satellites is more reliable than others.

Furthermore, our commercial delivery systems are designed and built to be even more robust and reliable than our single-family resident installations. Utilization of more efficient satellite dishes with one dish dedicated to each satellite in use, Heavy-duty mounting hardware and auto-leveling amplifiers provide consistent and unwavering signal strength to professionally designed delivery network located at your business or property. Thus troubleshooting steps are simpler to handle.

Satellite Networks-FDMA

Frequency Division Multiple Access or **FDMA** is a channel access method used in multiple-access protocols as a channelization protocol. FDMA gives users an individual allocation of one or several frequency bands, or channels. It is particularly commonplace in satellite communication. FDMA, like other Multiple Access systems, coordinates access between multiple users. Alternatives include TDMA, CDMA, or SDMA. These protocols are utilized differently, at different levels of the theoretical OSI model.

Disadvantage:

Crosstalk may cause interference among frequencies and disrupt the transmission.

Features

- In FDMA all users share the satellite simultaneously but each user transmits at single frequency.
- FDMA can be used with both analog and digital signal.
- FDMA requires high-performing filters in the radio hardware, in contrast to TDMA and CDMA.
- FDMA is not vulnerable to the timing problems that TDMA has. Since a predetermined frequency band is available for the entire period of communication, stream data (a continuous flow of data that may not be packetized) can easily be used with FDMA.
- Due to the frequency filtering, FDMA is not sensitive to near-far problem which is pronounced for CDMA.
- Each user transmits and receives at different frequencies as each user gets a unique frequency slot

There are two main techniques:

1. Multi-channel per-carrier (MCPC)

With **multiple channels per carrier** (**MCPC**), several subcarriers are combined into a single bitstream before being modulated onto a carrier transmitted from a single location to one or more remote sites. This uses time-division multiplexing (TDM). It is a heteronym of sorts, as it was the only way radio networks were transmitted ("piggybacked" on television networks) until SCPC.

FDMA is distinct from frequency division duplexing (FDD).

While FDMA allows multiple users simultaneous access to a transmission system, FDD refers to how the radio channel is shared between the uplink and downlink (for instance, the traffic going back and forth between a mobile-phone and a mobile phone base station). Frequency-division multiplexing (FDM) is also distinct from FDMA. FDM is a physical layer technique that combines and transmits low-bandwidth channels through a high-bandwidth channel. FDMA, on the other hand, is an access method in the data link layer.

FDMA also supports demand assignment in addition to fixed assignment. *Demand assignment* allows all users apparently continuous access of the radio spectrum by assigning carrier frequencies on a temporary basis using a statistical assignment process. The first FDMA *demand-assignment* system for satellite was developed by COMSAT for use on the *Intelsat*series *IVA* and *V* satellites.

In digital radio and digital television, an ensemble or other multiplex or multichannel stations can be considered MCPC, though the term is generally only applied to satellites.

The major disadvantage of MCPC is that all of the signals must be sent to a single place first, and then combined for retransmission — a major reason for using SCPC instead.

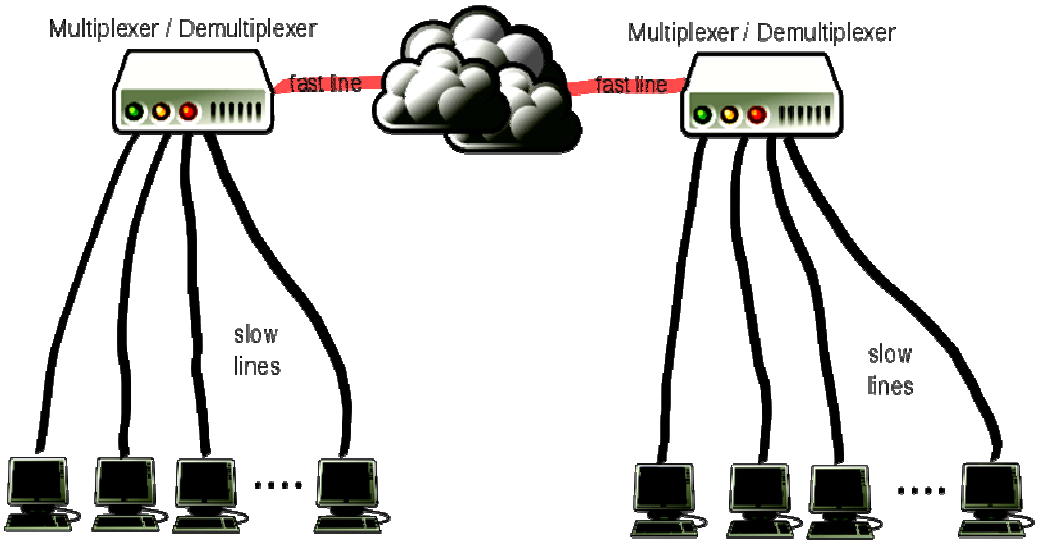
- In telecommunications and computernetworks, **multiplexing** (al so known as **muxing**) is a method by which multiple analog message signals or digital data streams are combined into one signal over a shared medium. The aim is to share an expensive resource. For example, in telecommunications, several telephone calls may be carried using one wire. Multiplexing originated in telegraphy, and is now widely applied in communications.

- The multiplexed signal is transmitted over a communication channel, which may be a physical transmission medium. The

multiplexing divides the capacity of the low-level communication channel into several higher-level logical channels, one for each message signal or data stream to be transferred. A reverse process, known as demultiplexing, can extract the original channels on the receiver side.

- A device that performs the multiplexing is called a multiplexer (MUX), and a device that performs the reverse process is called a demultiplexer (DEMUX).

- Inverse multiplexing (IMUX) has the opposite aim as multiplexing, namely to break one data stream into several streams, transfer them simultaneously over several communication channels, and recreate the original data stream
-



2. Single- Channel per-carrier (SCPC)

Single channel per carrier (**SCPC**) refers to using a single signal at a given frequency and bandwidth. Most often, this is used on broadcast satellites to indicate that radio stations are not multiplexed as subcarriers onto a single video carrier, but instead independently share a transponder. It may also be used on other communications satellites, or occasionally on non-satellite transmissions.

In an SCPC system, satellite bandwidth is dedicated to a single source. This makes sense if it is being used for something like satellite radio, which broadcasts continuously. Another very common application is voice, where a small amount of fixed bandwidth is required. However, it does not make sense for burst transmissions like satellite internet access or telemetry, since a

customer would have to pay for the satellite bandwidth even when they were not using it.

Where multiple-access is concerned, SCPC is essentially FDMA. Some applications use SCPC instead of TDMA, because they require guaranteed, unrestricted bandwidth. As satellite TDMA technology improves however, the applications for SCPC are becoming more limited.

Advantages:

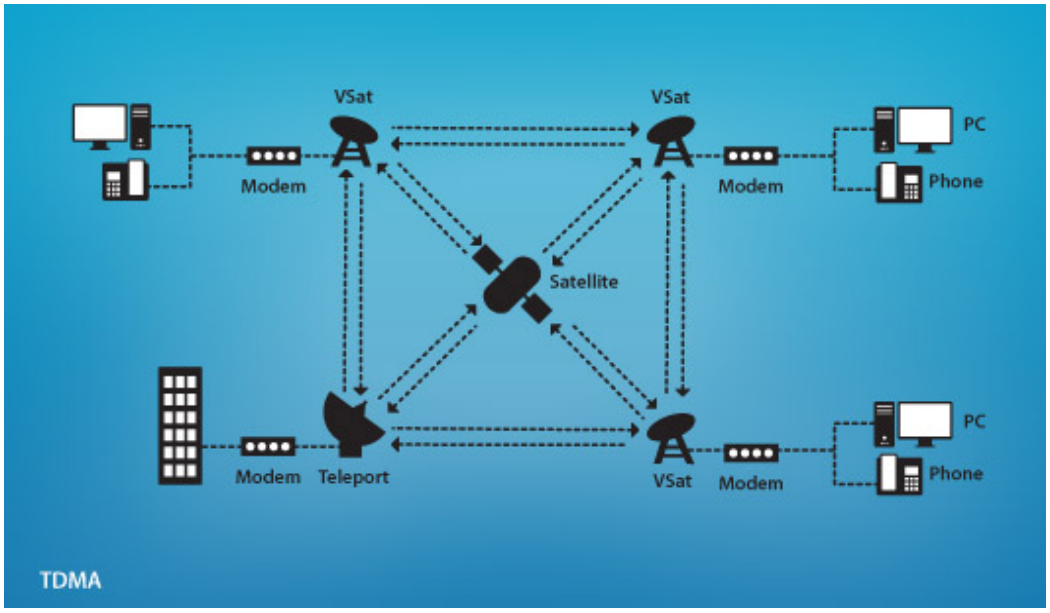
- simple and reliable technology
- low-cost equipment
- any bandwidth (up to a full transponder)
- usually 64 kbit/s to 50 Mbit/s
- easy to add additional receive sites (earth stations)

Disadvantages

- inefficient use of satellite bandwidth for burst transmissions, typically encountered with packet data transmission
- usually requires on-site control
- When used in remote locations, the transmitting dish must be protected.
- A dish which is moved out of alignment can result in fines as high as $1,100 per minute (as of 2003) from the satellite operator.

Satellite Networks-TDMA

When you require efficient satellite bandwidth utilization for multiple sites with infrequent or low usage requirements, TDMA technologies provide low cost, highly effective solutions – perfect for Satellite Internet applications.



OVERVIEW :

No other technology is quite as suited to the bursty nature of a good deal of IP traffic as TDMA. Many sites can share bandwidth and timeslots allocated dynamically to enhance bandwidth utilization. TDMA networks are highly flexible, extendable and bandwidth efficient. Ideal for applications that are not constantly resource intensive, then we have to create TDMA VSAT satellite networks that share network capacity on-demand

KEY FEATURES

This network is available in point-to-multipoint and fully meshed configurations.

TDMA networks offer the following benefits:

- Low cost Internet access
- Dynamic, demand-assigned bandwidth allocation
- Cost-effective, flexible and expandable network architecture
- Reliability and quality of service guaranteed by SLA
- One stop solution for network design, implementation and support
- Centralized management.

KEY APPLICATIONS

You should consider an TDMA VSAT satellite network for the following:

- Low cost, multi-site connectivity
- Satellite Internet
- VoIP
- Content distribution

Satellite Networks-CDMA

The CDMA signal provides excellent data and voice capacity through the satellite phone network of 48 satellites. The CDMA signal is the foundation for 3G communication services worldwide. CDMA converts speech signal into digital format and then transmits it from the Satellite phone up to the satellite systems and down to the ground station. Every call over the satellite network has its own unique code which distinguishes it from the other calls sharing the airwaves at the same time. **The CDMA signal is without interference, cross talk or static**. CDMA was introduced in 1995 and soon become the fastest growing wireless technology

---

Key features of satellite phone CDMA:

- Unique forward and reverse links
- Direct sequence spread spectrum
- Seamless soft handoff
- Universal frequency reuse
- Multi-path propagation for diversity
- Variable rate transmission

Introduction to Global Satellite Systems

Several different types of global satellite communications systems are in various stages of development. Each system either planned or existing, has a unique configuration optimized to support a unique business plan based on the services offered and the markets targeted.

In the last few years more than 60 global systems have been proposed to meet the growing demand for international communications services. More are being planned and these are in addition to a large number of new regional systems.

Some of the global systems intend to provide global phone service, filling in where ground-based wireless systems leave off or providing seamless connectivity between different systems.

Others intend to provide global data connectivity, either for low-cost short message applications such as equipment monitoring, or for high-speed Internet access anywhere in the world.

The global phone systems will target two very different markets. The first is the international business user, who wants the ability to use a single mobile wireless phone anywhere in the world. This is impossible today on terrestrial systems because mobile phone standards are different from region to region. The second market is unserved and underserved communities where mobile and even basic telecommunications services are unavailable.

Because global and regional satellite systems are relatively new in non-military communications, these market approaches still are untested and it is likely that economics, user acceptance rates, technical difficulties and other factors will cause adjustments in the business plans of many of these systems.

The Types of Satellite Systems

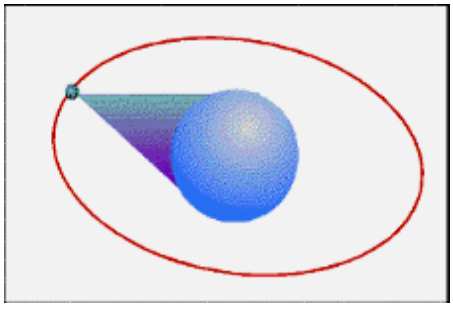
The design of a satellite system is closely tied to the market it is intended to serve and the type of communications services it is intended to offer. There are four general system designs, which are differentiated by the type of orbit in which the satellites operate:

- Geostationary Orbit (GEO),
- Low-earth Orbit(LEO),
- Medium-earth Orbit (MEO), and
- Highly Elliptical Orbit (HEO).

Each of these has various strengths and weaknesses in its ability to provide particular communications services.

Outside of the well-defined GEO universe, the differences between these systems are often not absolute and the acronyms applied to a system can be confusing and sometimes misleading. Several systems, for example, are variously described as LEOs and MEOs. Constantly evolving technology along with newly developing markets and service definitions combine to blur the lines between one satellite system and another.

The definitions below are meant to describe the general characteristics of GEOs, MEOs, LEOs and HEOs. Although examples of commercial systems employing these satellites are given, keep in mind that each system has unique characteristics that may not match precisely the general descriptions. The same caution should be applied to ascribing a particular satellite type's limitations to any one commercial system, since each uses several strategies for minimizing or overcoming the limitations inherent in satellite designs. For example, some systems may employ more than one type of satellite.



GEOSTATIONARY (GEO)

GEO systems orbit the Earth at a fixed distance of 35,786 kilometers (22,300 miles). The satellite's speed at this altitude matches that of the Earth's rotation, thereby keeping the satellite stationary over a particular spot on the Earth. Examples of GEO systems include INTELSAT, Inmarsat, and PanAmSat.

Geostationary satellites orbit the Earth above the equator and cover one third of the Earth's surface at a time. The majority of communications satellites are GEOs and these systems will continue to provide the bulk of the communications satellite capacity for many years to come.

GEOs support voice, data, and video services, most often providing fixed services to a particular region. For example, GEO satellites provide back-up voice capacity for the majority of the U.S. long distance telephone companies and carry the bulk of nation-wide television broadcasts, which commonly are distributed via from a central point to affiliate stations throughout the country.

Until recently, the large antennae and power requirements for GEO systems limited their effectiveness for small-terminal and mobile services. However, newer high-powered GEO satellites using clusters of concentrated "spot beams" can operate with smaller terrestrial terminals than ever before and can support some mobile applications. GEO satellite coverage typically degrades beyond 20 degrees North Latitude and 20 degrees South Latitude.

GEO systems have a proven track record of reliability and operational predictability not yet possible for the more sophisticated orbital designs now being deployed. GEO systems are also less complicated to maintain because their fixed location in the sky requires relatively little tracking capability in ground equipment. In addition, their high orbital altitude allows GEOs to remain in orbit longer than systems operating closer to Earth. These characteristics, along with their high bandwidth capacity, may provide a cost advantage over other system types.

However, their more distant orbit also requires relatively large terrestrial antennae and high-powered equipment and are subject to transmission delays. In addition, since only a few large satellites carry the load for the entire system, a GEO satellite loss is somewhat more consequential than for the systems described below.

Summary of GEO Pros and Cons

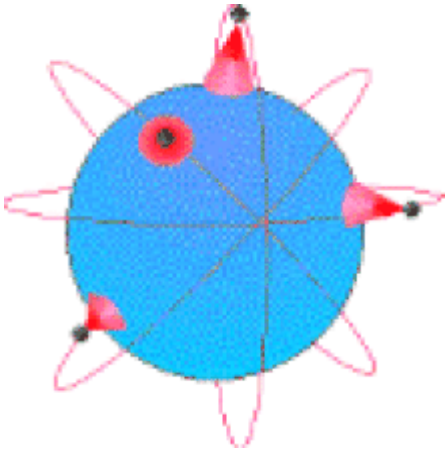
- PRO: GEO systems have significantly greater available bandwidth than the LEO and MEO systems described below. This permits them to provide two-way data, voice and broadband services that may be unpractical for other types of systems.

- PRO: Because of their capacity and configuration, GEOs are often more cost-effective for carrying high-volume traffic, especially over long-term contract arrangements. For example, excess capacity on GEO systems often is reserved

in the form of leased circuits for use as a backup to other communications methods.

- CON: GEO systems, like all other satellite systems, require line-of-sight communication paths between terrestrial antennae and the satellites. But, because GEO systems have fewer satellites and these are in a fixed location over the Earth, the opportunities for line of sight communication are fewer than for systems in which the satellites "travel" across the sky. This is a significant disadvantage of GEO systems as compared to LEO and MEO systems, especially for mobile applications and in urban areas where tall buildings and other structures may block line-of-sight communication for hand-held mobile terminals.

- CON: Some users have expressed concern with the transmission delays associated with GEO systems, particularly for high-speed data. However, sophisticated echo cancellation and other technologies have permitted GEOs to be used successfully for both voice and high-speed data applications.



LOW-EARTH ORBIT (LEO)

LEO systems fly about 1,000 kilometers above the Earth (between 400 miles and 1,600 miles) and, unlike GEOs, travel across the sky. A typical LEO satellite takes less than two hours to orbit the Earth, which means that a single satellite is "in view" of ground equipment for a only a few minutes. As a consequence, if a transmission takes more than the few minutes that any one satellite is in view, a LEO system must "hand off" between satellites in order to complete the transmission. In general, this can be accomplished by constantly relaying signals between the satellite and various ground stations, or by communicating between the satellites themselves using "inter-satellite links."

In addition, LEO systems are designed to have more than one satellite in view from any spot on Earth at any given time, minimizing the possibility that the network will loose the

transmission. Because of the fast-flying satellites, LEO systems must incorporate sophisticated tracking and switching equipment to maintain consistent service coverage. The need for complex tracking schemes is minimized, but not obviated, in LEO systems designed to handle only short-burst transmissions.

The advantage of the LEO system is that the satellites' proximity to the ground enables them to transmit signals with no or very little delay, unlike GEO systems. In addition, because the signals to and from the satellites need to travel a relatively short distance, LEOs can operate with much smaller user equipment (e.g., antennae) than can systems using a higher orbit. In addition, a system of LEO satellites is designed to maximize the ability of ground equipment to "see" a satellite at any time, which can overcome the difficulties caused by obstructions such as trees and buildings.

There are two types of LEO systems, Big LEOs and Little LEOs, each describing the relative mass of the satellites used as well as their service characteristics.

Little LEO satellites are very small, often weighing no more than a human being, and use very little bandwidth for communications. Their size and bandwidth usage limits the amount of traffic the system can carry at any given time. However, such systems often employ mechanisms to maximize capacity, such as frequency reuse schemes and load delay tactics.

Little LEO systems support services that require short messaging and occasional low-bandwidth data transport, such as paging, fleet tracking and remote monitoring of stationary monitors for everything from tracking geo-platonic movements to checking on vending machine status. The low bandwidth usage may allow a LEO system to provide more cost effective service for occasional-use applications than systems that maximize their value based on bulk usage. Examples of Little LEO systems include Orbcomm, Final Analysis and Leo One.

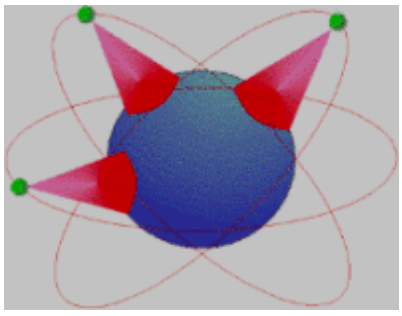
Big LEO systems are designed to carry voice traffic as well as data. They are the technology behind "satellite phones" or "global mobile personal communications system" (GMPCS) services now being developed and launched.

Most Big LEO systems also will offer mobile data services and some system operators intend to offer semi-fixed voice and data services to areas that have little or no terrestrial telephony infrastructure. Smaller Big LEO constellations also are planned to serve limited regions of the globe. Examples of Big LEO systems include Iridium, Globalstar and the regional Constellation and ECO-8 systems.

An emerging third category of LEO systems is the so-called "super LEOs" or "mega LEOs," which will handle broadband data. The proposed Teledesic and Skybridge systems are examples of essentially Big LEO systems optimized for packet-switched data rather than voice. These systems share the same advantages and drawbacks of other LEOs and intend to operate with inter-satellite links to minimize transmission times and avoid dropped signals.

Summary of LEO Pros and Cons

- PRO: The transmission delay associated with LEO systems is the lowest of all of the systems.
- CON: The small coverage area of a LEO satellite means that a LEO system must coordinate the flight paths and communications hand-offs a large number of satellites at once, making the LEOs dependent on highly complex and sophisticated control and switching systems.
- PRO: Because of the relatively small size of the satellites deployed and the smaller size of the ground equipment required, the Little LEO systems are expected to cost less to implement than the other satellite systems discussed here.
- CON: LEO satellites have a shorter life span than other systems mentioned here. There are two reasons for this: first, the lower LEO orbit is more subject to the gravitational pull of the Earth and second, the frequent transmission rates necessary in LEO systems mean that LEO satellites generally have a shorter battery life than others.



MEDIUM EARTH ORBIT (MEO)

MEO systems operate at about 10,000 kilometers (between 1,500 and 6,500 miles) above the Earth, which is lower than the GEO orbit and higher than most LEO orbits. The MEO orbit is a compromise between the LEO and GEO orbits. Compared to LEOs, the more distant orbit requires fewer satellites to provide coverage than LEOs because each satellite may be in view of any particular location for several hours. Compared to GEOs, MEOs can operate effectively with smaller, mobile equipment and with less latency (signal delay).

Although MEO satellites are in view longer than LEOs, they may not always be at an optimal elevation. To combat this difficulty, MEO systems often feature significant coverage overlap from satellite to satellite, this in turn requires more sophisticated tracking and switching schemes than GEOs.

Typically, MEO constellations have 10 to 17 satellites distributed over two or three orbital planes. Most planned MEO systems will offer phone services similar to the Big LEOs. In fact, before the MEO designation came into wide use, MEO systems were considered Big LEOs. Examples of MEO systems include ICO Global Communications and the proposed Orblink from Orbital Sciences.

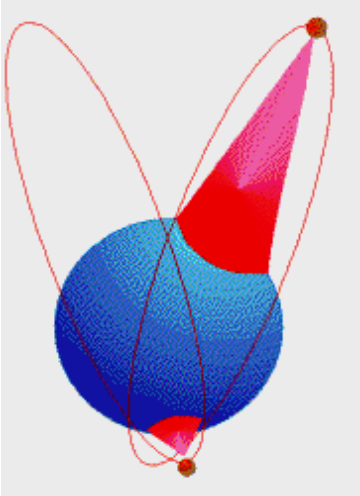
Summary of MEO Pros and Cons

- PRO: MEO systems will require far fewer satellites than LEOs, reducing overall system complexity and cost, while still requiring fewer technological fixes to eliminate signal delay than GEOs.
- CON: MEO satellites, like LEOs, have a much shorter life expectancy than GEOs, requiring more frequent launches to maintain the system over time.
- PRO: MEO systems's larger capacity relative to LEOs may enable them to be more flexible in meeting shifting market demand for either voice or data services.
- CON: MEO systems, as well as some Big LEOs, targeted at the voice communications market may have a disadvantage when compared with cellular and other terrestrial wireless networks. A satellite signal is inherently weaker and is more subject to interference than those of terrestrial systems, thus requiring a larger antenna than a traditional mobile phone. By contrast, the trend in the mobile phone market is toward smaller and smaller phones.

HIGHLY ELLIPTICAL ORBIT (HEO)

HEO systems operate differently than LEOs, MEOs or GEOs. As the name implies, the satellites orbit the Earth in an elliptical path rather than the circular paths of LEOs and GEOs. The HEO path typically is not centered on the Earth, as LEOs, MEOs and GEOs

are. This orbit causes the satellite to move around the Earth faster when it is traveling close to the Earth and slower the farther away it gets. In addition, the satellite's beam covers more of the Earth from farther away, as shown in the illustration.



The orbits are designed to maximize the amount of time each satellite spends in view of populated areas. Therefore, unlike most LEOs, HEO systems do not offer continuous coverage over outlying geographic regions, especially near the south pole. Several of the proposed global communications satellite systems actually are hybrids of the four varieties reviewed above. For example, all of the proposed HEO communications systems are hybrids, most often including a GEO or MEO satellite orbital plane around the equator to ensure maximum coverage in the densely populated zone between 40 degrees North Latitude and 40 degrees South Latitude. Examples of HEO systems include Ellipso and the proposed Pentriad.

Summary of HEO Pros and Cons

- **PRO:** The HEO orbital design maximizes the satellites' time spent over populated areas, thus requiring fewer satellites than LEOs and providing superior line-of-sight in comparison to most LEOs or GEOs.

- **CON:** Coverage of a typical HEO system is not as complete as other orbital designs, although they will provide good coverage over most population centers.

11

THE NETWORK LAYER

Unit Structure

11.1. Introduction
11.2. Network Layer Design issues
11.3. Routing Algorithms
    11.3.1. Adaptive & Non Adaptive
    11.3.2. The Optimality Principle & Sink Tree
11.4. Shortest Path routing
11.5. Flooding
11.6. Distance vector routing
11.7. Link state routing
11.8. Broadcast routing
11.9. Multicast routing
11.10. Internetworking

11.1. INTRODUCTION - THE NETWORK LAYER

- The Network Layer deals with end to end transmission of data, unlike data link layer where the responsibility is node to node delivery.
- The source and the destination usually have many intermediate nodes connecting them. To transmit the data from the sender to the receiver involves routing through these nodes(routers) taking into consideration the various network conditions like congestion, the shortest path to the receiver, etc.

11.2. NETWORK LAYER DESIGN ISSUES

Following are the issues faced by network designers. These are related to the services provided to the transport layer and internal subnet design:

1. Store-and-Forward Packet Switching

- The sender and receiver may be connected by many intermediate nodes like routers. It is the routers that connect the user to the carriers network.
- When the sender has data to send it sends it in the form of packets to the nearest router where it is stored until the complete packet has arrived so that it can be checked for errors.
- This packet is then forwarded to the next router where the same process is repeated.
- The packet is stored & forwarded at every router until it reaches its destination.
- Here every router may be similar in terms of construction, software and design but the concern here is the algorithm that runs on every router that may complicate the process.

2. Services Provided to the Transport Layer

- The network layer provides services to the Transport layer. It is designed to achieve the following goals:
    1. The services should be independent of the router technology.
    2. The transport layer should be shielded from the number, type, and topology of the routers present.
    3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.
- With the above goals there are two ways to implement network layer services: **Connectionless service & Connection oriented service**
- The Internet is an example of a connectionless network layer while the ATM networks have a connection oriented network layer.
- Both approaches have their own advantages and disadvantages.

3. Implementation of Connectionless Service

- A connectionless service is where the user simply groups the data together into a unit and sends it to the network assuming that the network will work out a way to carry the data to the intended receiver.

- There is no reliability in connectionless service that the data will reach the receiver.
- A letter dropped in the mail box is an example of connectionless service. The sender cannot be sure that the receiver will receive the message but can only hope for it.
- The network layer implements connectionless service using independent data packets called **datagrams.**
- With Datagrams the routes from source to destination are not worked out in advance.
- Every data packet that is sent may be routed independently which means packets belonging to a piece of data may arrive at the receiver from different routes and out of order.
- At every router the decision is made as to which router will be the next node to transmit the datagram.
- With datagrams there is more work for the routers but it is a robust way to deal with network failures and congestion.
- The other advantage of using datagram approach is that the routers need not maintain the information about routes in its tables specific to packets being sent.

4. Implementation of Connection-Oriented Service

- A connection oriented service is where the user is given a reliable (dedicated) end to end connection to the destination.
- To communicate the sender has to establish a connection with the receiver initially and only then can send the data.
- The connection exists for the duration of data transfer.
- A telephone call is an example of Connection oriented service.
- The network layer implements connection oriented service by establishing connections called **virtual circuits.**
- The mechanism of virtual circuits is to choose only one route from source to destination to send the data.
- When the circuit (connection) is established, it is used for sending all the data.
- When there is no data to be sent the circuit is terminated.
- With Virtual circuits, every router needs to maintain a table of routes.
- Every packet has a virtual circuit number and uses circuit establish, data transfer and tear down phase as explained in virtual circuit networks in the previous chapters.

11.3. ROUTING ALGORITHMS

- The main function of the network layer is routing packets from the source to destination.
- The algorithms that choose the routes and the data structures that they use are a major area of network layer design.
- The **Routing algorithm** is a part of the network layer software. It is responsible for deciding which output line an incoming packet should be transmitted on.
- Certain properties that are desirable in a routing algorithm are : correctness, simplicity, robustness, stability, fairness, and optimality.

11.3.1. Types of Routing Algorithms

They are of two types:

1. Non-adaptive Algorithms
2. Adaptive Algorithms

1. Non-adaptive Algorithms

- The choice of which route to be selected is planned in advance.
- It does not take into account any kind of metric, measurements or estimates about traffic and topology.
- They are also called **Static Routing.**

2. Adaptive Algorithms

- The choice of which route to be taken is calculated at runtime.
- The routing decision can be changed if there are any changes in the topology or traffic.
- It takes into account several metrics such as distance, number of hops, estimated transit time, etc.
- They are also called **Dynamic Routing.**

11.3.2. The Optimality Principle

- The **Optimality Principle** is a general statement about optimal routes without any concern about the network topology or traffic.
- It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.

- To explain this, call the part of the route from I to J as r1 and the rest of the route r2.
- If a route better than r2 existed from J to K, it could be concatenated with r1 to improve the route from I to K, contradicting our statement that r1r2 is optimal.

Sink tree

- From the optimality principle, we can see that the set of optimal routes from all sources to a given destination form a tree rooted at the destination.
- Such a tree is called a sink tree and is shown in the diagram below, where the distance metric is the number of hops.
- Since a sink tree is a tree that does not contain any loops, so each packet will be delivered within a finite and bounded number of hops.

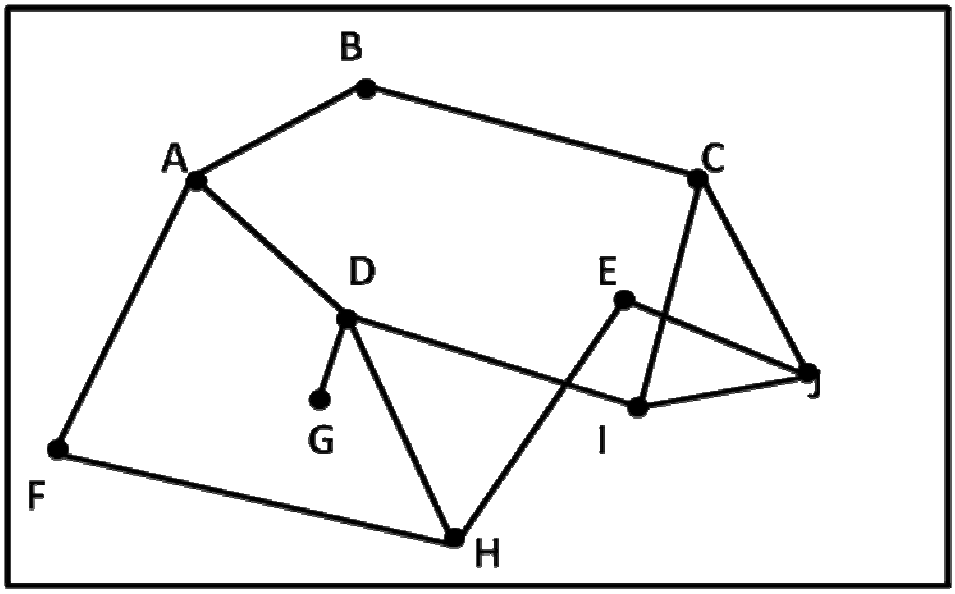


Figure : A Subnet

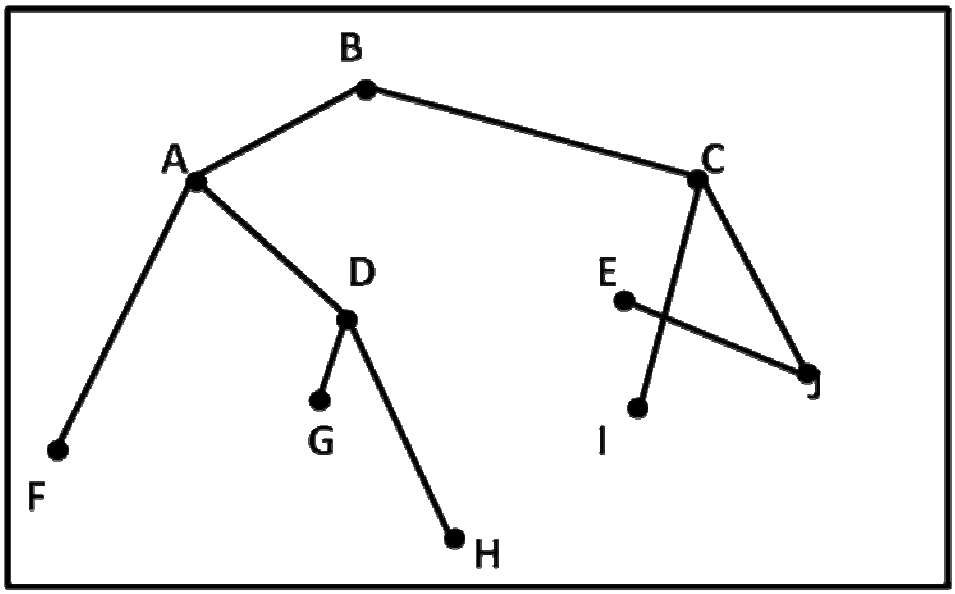


Figure : A sink tree for a router

Static Routing Algorithms
Some examples of Static algorithms are:

1. Shortest Path routing
2. Flooding

11.4. SHORTEST PATH ROUTING

- This algorithm is based on the simplest and most widely used principle.
- A graph of subnet is built, where each node represents a router and each arc represents a communication link.
- This algorithm simply finds the shortest path between the routers so that we can choose it as path.
- The shortest path could be found out by counting the number of hops or by measuring the geographical distance in kilometers.
- Other metrics besides hops and physical distance are also possible. For example, each arc could be labeled with the mean queuing and transmission delay and find the shortest path as the fastest path rather than the path with the fewest arcs or kilometers.
- The labels on the arcs can be computed as distance, traffic, mean queue length, cost of communication, measured delay etc.

- The algorithm weighs various parameters and computes the shortest path, based on any combination of criteria as stated.
- Examples of Shortest Path Algorithms are : Dijkstra Algorithm, Bellman Ford Algorithm.

11.5. FLOODING

- In **Flooding** every incoming packet is sent out on every outgoing line except the one it arrived on.
- The main disadvantage is that Flooding results in vast numbers of duplicate packets, almost an infinite number unless some measures are taken to damp the process.

1. **Using a hop count**: A hop count may be included in the header of every packet, which could be used to suppress onwards transmission of packets after the number of hops is exceeds the network diameter.
2. **Keep a track of the flooded packets:**
    - This is done to avoid sending the packets multiple times.
    - This can be achieved by having the source router put a sequence number in each packet it receives from its hosts.
    - Each router maintains a list of routers and sequence numbers which contains the sequence numbers originating from that router.
    - If an incoming packet is on the list, it is not flooded.
3. **Selective Flooding:**
    - This is a more practical approach to flooding.
    - In this approach, every incoming packet is not sent on every outgoing line, instead it is sent only on those lines which are approximately going in the right direction.

Applications of Flooding

- It does not have many practical applications
- But it is used in military purposes like flooding a large number of routers to blow them up.
- In distributed database applications it may be used to update all the databases concurrently.

Dynamic Routing Algorithms

- The disadvantages of static routing algorithms described above is that they do not take the current network load into account.
- Following are two Dynamic Routing algorithms
  1. Distance vector routing
  2. Link state routing

11.6. DISTANCE VECTOR ROUTING

- These algorithms operate by having each router maintain a table (i.e, a vector) giving the best known distance to each destination and which line to use to get there.
- These tables are updated by exchanging information with the neighbors.
- It is also known as :
  1. Distributed Bellman-Ford routing algorithm
  2. Ford-Fulkerson algorithm
- In distance vector routing, each router maintains a routing table that is indexed by, and containing one entry for, each router in the subnet.
- This entry contains two parts:
  1. First, the preferred outgoing line to use for that destination and
  2. Second, an estimate of the time or distance to that destination.
- The router is assumed to know the "distance" to each of its neighbors.
- In Distance Vector Routing, a router tells its neighbors its distance to every other router in the network.

11.7.LINK STATE ROUTING

The idea behind link state routing is simple and can be stated as five parts.

Each router must do the following:

1. Discover its neighbors and learn their network addresses.
2. Measure the delay or cost to each of its neighbors.
3. Construct a packet telling all it has just learned.

4. Send this packet to all other routers.
5. Compute the shortest path to every other router.

11.8. BROADCAST ROUTING

Some applications requires to send messages to multiple or all hosts. This is achieved by using Broadcast Routing.
The following techniques could be used:

1. Flooding
As discussed above

2. Multi-destination Routing

- Every packet contains a list of destinations.
- When a packet arrives at a router, it is checked for all the destinations to determine the set of output lines that can be used to reach the destination.
- The router creates new copies of the packet one for each output line and each packet contains only those destinations which could be reached on those output lines. (i.e. the number of destination addresses is filtered out)
- After a few hops each packet will contain only one address like a normal packet.

3. Spanning Tree

- This approach uses the Sink tree for the router that initiates the broadcast.
- Spanning tree is a subset of the subnet that contains all the routers but contains no loops.
- Using this approach the router can broadcast the incoming packet on all the spanning tree lines except the one it arrived on.
- This approach makes excellent use of bandwidth and generates the minimum number of packets necessary to get the job done.
- It is mandatory for the routers to know the spanning tree.

4. Reverse Path Forwarding

- This algorithm tries to approximate the behavior of spanning tree approach when the routers have no information about the spanning tree.

- **Mechanism :**

  When a broadcasted packet is received by a router, it checks whether the same line is used to broadcast packets to the source. Hence the packet has itself followed the best available path.

  In case the broadcast packet arrives at a line other than the one used to send data to the source the packet is discarded.

- **Advantages of this approach**

  1. It is easy to implement and is efficient.
  2. The routers do not need to know the spanning tree.
  3. There is no overhead involved in maintaining the destination list.
  4. No mechanism is required to stop the process (damping like hop counter) as is required in flooding.

11.9.MULTICAST ROUTING

- Broadcast routing has a disadvantage.
- Ex. It can be used to inform 1000 users on a million node network. But this will be inefficient as most of the users will not be interested in the broadcasted message. Also it poses a threat of revealing the message to users who are not authorized to see it.
- Sending a message to a group of nodes in a large size network is called **multicasting** and the routing algorithms used are called **Multicast Routing Algorithms.**
- Multicasting requires **Group Management.**
- It involves methods to create and destroy groups, and to allow processes to join and leave groups.
- It is necessary for the router to know which host belongs to which groups. When a process joins a particular group the router informs its host about this fact.
- Changes in group membership may be communicated either by host to the router or could be queried by the routers to the hosts.
- Each router has to compute a spanning tree that comprises of all the routers in the network.
- When a router receives a multicast packet destined to a group, it will first examine the spanning tree and prune it to remove the output lines that do not lead to hosts that are member to the group.

- Various ways are possible to prune the spanning tree.
  I. With **Link state routing,** the routing has information of the complete topology, including the information on hosts and their groups. The pruning begins from the end of the path to the root removing the routers that are not needed for transmission.
  II. **Distance Vector Routing** uses Reverse path forwarding to prune the spanning tree.

11.10. INTERNETWORKING

- We cannot always make an assumption that the network being used contains homogeneous elements using same protocols in every element.
- The internet today is a network of networks joining together many LAN, MAN & WANS with a wide variety of protocols being used in every network.
- The following issues may arise when we try to connect two networks together
  I. The existing network operating on a particular technology and protocol obviously has a large customer base. It would be very difficult to migrate users from one network to another.
  II. As the cost of computers and setting up of networks has become cheaper the decision as to which kind of network is to be set up is not taken by top management but the department heads. This may lead to two departments of the same company having Unix and MAC machines running on TCP/IP & AppleTalk respectively having to be connected together. If the top management would have taken a decision it would have been uniform for all the departments saving the trouble of discrepancies between the two types that arises when exchanging data.
  III. As technology keeps on changing new protocols and networks keep evolving.

- **Difference between networks**

I. Two networks can differ in many ways as summarized below:

No	Criteria	Description
1	Service Offered	Service offered by the network could be connection oriented or connectionless
2	Protocol	The network could operate on totally different set of protocol suites ex. IP, SNA, ATM, MPLS, APPLETALK, etc.
3	Addressing	The type of addressing used could also differ. Ex. Flat vs Hierarchical addressing
4	Multicasting	Some networks may use Multicasting or broadcasting others may not.
5	Packet Size	Every network has its maximum packet size defined.
6	Quality of Service	Different networks may have or not any mechanisms to achieve quality of service (present or not)
7	Error Handling	The networks may use error handling mechanisms which may be Reliable, ordered or unordered.
8	Flow Control	Various mechanism to achieve Flow Control are available. Ex. Sliding Window,
9	Congestion Control	The networks may use open loop or closed loop congestion control techniques.
10	Security	One or more security constraints can be implemented like encryption.

II. A packet sent from one network to another may have to face the issues listed above before it reaches the destination.

- **Connecting different networks**

I. Networks can be connected by devices that operate at physical layer, data link layer, network layer & application layer.

II. In the physical layer, networks can be connected by repeaters or hubs, which move the bits from one network to another. These are analog devices and they just regenerate signals.

III. In the data link layer, networks can be connected by bridges and switches which accept frames, examine the MAC addresses, and then forward the frames to a different network.

IV. In the network layer, networks can be connected by routers. In case the two networks have a different network layer the router may translate the packet from source to destination network format. Such a router handling multiple protocols is called **multiprotocol router**.

V. At the transport layer, transport gateways form an interface between transport connections. ex. A transport gateway facilitates the flow of packets from TCP network to SNA.

VI. At the application layer, application gateways translate message semantics. Ex. Application gateways for email messages.

- **Connectionless Internetworking**

I. The other alternative to using concatenated virtual circuits is connectionless internetworking i.e. Datagram Approach.

II. In this approach the network layer allows the transport layer to simply inject packets into the network nothing else.

III. The routing decision is taking at every router through which the packet passes.

IV. Disadvantages of this approach are:

a. There is no guarantee that packets will arrive at the destination and in order.

b. Since packets are simply injected into the network without any consideration of the intermediate networks, their protocol and addressing schemes it would be difficult to route such packets whose addressing is unknown. A solution to this problem is to have a standard format for packets that will be known to every router. Ex. IP

- **Tunneling**

I. The concept of tunneling is used when the source and destination networks work on same protocol but the intermediate network connecting those works on different protocol.

II. Ex. Source A and Destination B operate on TCP/IP on Ethernet and connected by an ATM network. When A wants to send a packet containing the address of Destination B, it puts the IP packet in an Ethernet frame and sends it to the router. The router removes the IP packet from the Ethernet frame puts it in

its Packet format and sends it forward. When the packet is handed over at the destination router it removes the IP packet puts it in a Ethernet frame and sends it to the destination B.

- **Fragmentation**

I. Each network imposes some maximum size on its packets. The problem appears when a large packet is sent to a network whose maximum packet size is too small.

II. The solution to the problem is to allow gateways to break up packets into fragments, sending each fragment as a separate internet packet.

III. The problem to fragmentation is combining the fragments into the original packet. There are various approaches to this problem. First, the fragments can be recombined at the gateway and then send the packet to the destination. Second, the fragments need not be combined at the gateway but directly by the destination.

Third, when the packets are being fragmented use such a smaller size that it can pass through every network.

References & Further Reading

Computer Networks – Andrew Tannenbaum

12

IP PROTOCOL

Unit Structure

12.1 Introduction – Network layer in the internet

12.2. IP Protocol – IPv4

- 12.2.1. IP addresses
- 12.2.2. Address Space
- 12.2.3. Notation
- 12.2.4. Classful Addressing
- 12.2.5. Netid & Hostid
- 12.2.6. Subnetting
- 12.2.7.CIDR
- 12.2.8. NAT

12.3. Protocols

- 12.3.1. ICMP
- 12.3.2. IGMP
- 12.3.3. ARP
- 12.3.4. RARP, BOOTP, DHCP

12.4. IPv6

12.1. NETWORK LAYER IN THE INTERNET

he internet is a network of networks. Actually at the network layer the internet is a collection of Autonomous Systems that are interconnected. The internet has no structure but it has several backbones that are made of high bandwidth lines and high speed routers. At the network layer the internet is a packet switched network. It used the datagram approach for switching which is connectionless. The reason for the internet to use datagram approach (connectionless) is that it is made of so many heterogeneous networks that it is impossible to make a connection from source to destination without knowing the nature of networks in advance.

The whole Internet is held together by the network layer protocol, IP (Internet Protocol)

12.2. IP PROTOCOL – IPV4

Packets in the IPv4 format are called datagram. An IP datagram consists of a header part and a text part. The header has a 20-byte fixed part and a variable length optional part. It is transmitted in big-endian order: from left to right, with the high-order bit of the Version field going first.

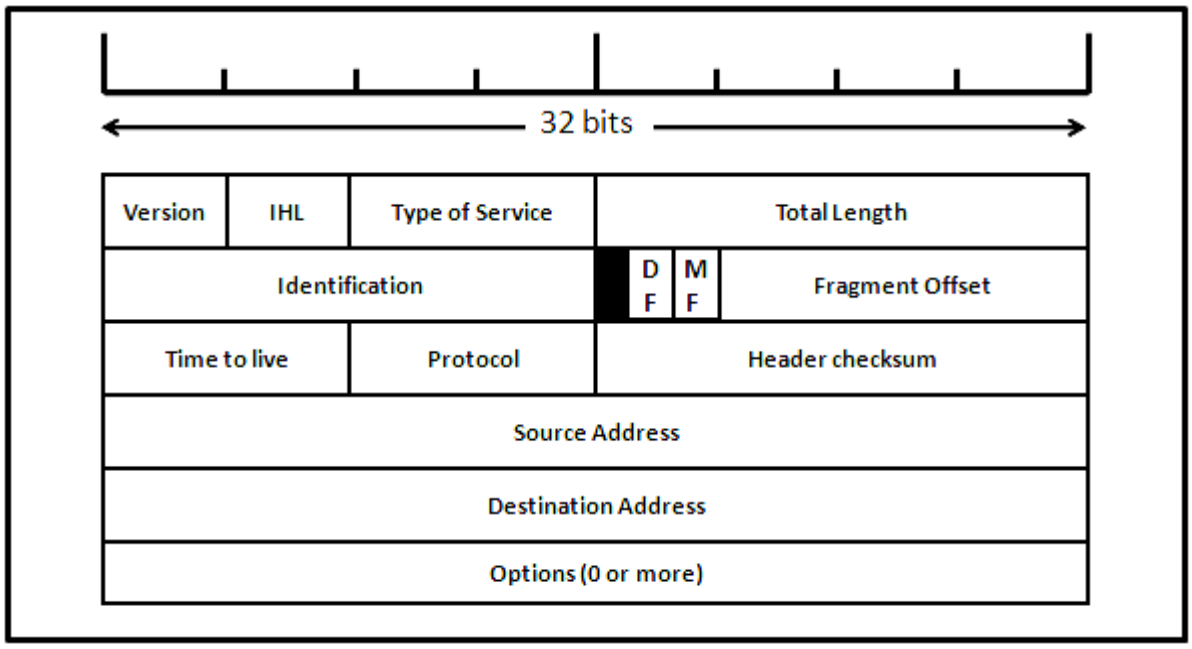


Figure: The IPv4 (Internet Protocol) header

The description of the fields shown in the diagram is as follows:

No	Field Name	Description
1	Version	Keeps track of the version of the protocol the datagram belongs to (IPV4 or IPv6)
2	IHL	Used to indicate the length of the Header. Minimum value is 5 Maximum value 15
3	Type of service	Used to distinguish between different classes of service
4	Total length	It includes everything in the datagram—both header and data. The maximum length is 65,535 bytes
5	Identification	Used to allow the destination host to identify which datagram a newly arrived fragment belongs to. All the fragments of a datagram contain the same Identification value

6	DF	1 bit field. It stands for Don't Fragment. Signals the routers not to fragment the datagram because the destination is incapable of putting the pieces back together again
7	MF	MF stands for More Fragments. All fragments except the last one have this bit set. It is needed to know when all fragments of a datagram have arrived.
8	Fragment offset	Used to determine the position of the fragment in the current datagram.
9	Time to live	It is a counter used to limit packet lifetimes. It must be decremented on each hop. When it hits zero, the packet is discarded and a warning packet is sent back to the source host.
10	Header checksum	It verifies Header for errors.
11	Source address	IP address of the source
12	Destination address	IP address of the destination
13	Options	The options are variable length. Originally, five options were defined: 1. Security : specifies how secret the datagram is 2. Strict source routing : Gives complete path to be followed 3. Loose source routing : Gives a list of routers not to be missed 4. Record route: Makes each router append its IP address 5. Timestamp: Makes each router append its IP address and timestamp

12.2.1. IP addresses

- Every host and router on the Internet has an IP address, which encodes its network number and host number.
- The combination is unique: in principle, no two machines on the Internet have the same IP address.
- An IPv4 address is 32 bits long

- They are used in the Source address and Destination address fields of IP packets.
- An IP address does not refer to a host but it refers to a network interface.

12.2.2. Address Space

- An address space is the total number of addresses used by the protocol. If a protocol uses *N* bits to define an address, the address space is 2^N because each bit can have two different values (0 or 1) and *N* bits can have 2^N values.
- IPv4 uses 32-bit addresses, which means that the address space is 2^{32} or 4,294,967,296 (more than 4 billion).

12.2.3.Notations

- There are two notations to show an IPv4 address:

  1. Binary notation

  The IPv4 address is displayed as 32 bits.

  ex. 11000001 10000011 00011011 11111111

  2. Dotted decimal notation

  To make the IPv4 address easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separating the bytes.

  Each byte (octet) is 8 bits hence each number in dotted-decimal notation is a value ranging from 0 to 255.

  Ex. 129.11.11.239

12.2.4. Classful addressing

In classful addressing, the address space is divided into five classes: A, B, C, D,and E.

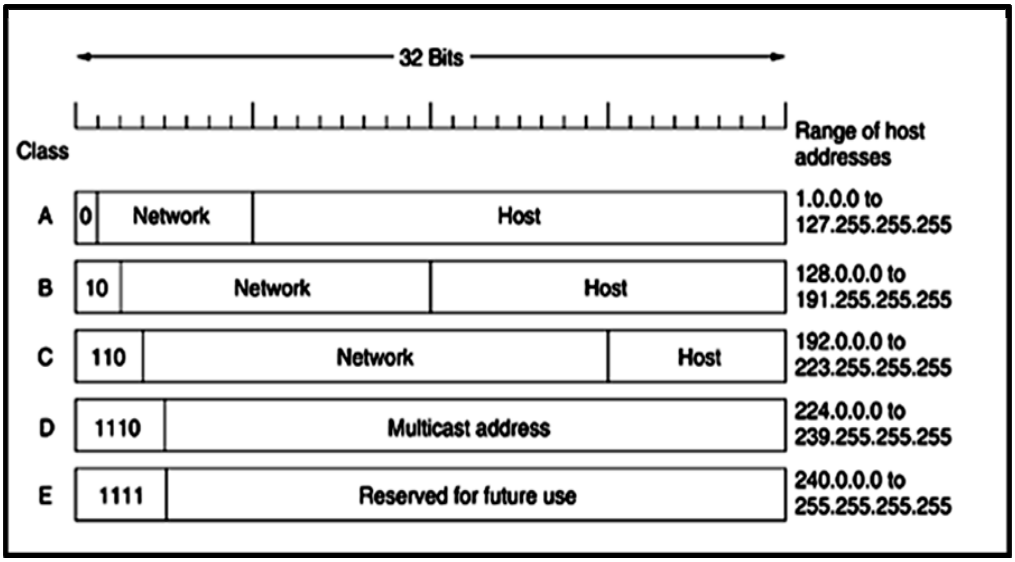


Figure: Classful addressing : IPv4

12.2.5. Netid and Hostid

- In classful addressing, an IP address in class A, B, or C is divided into netid and hostid.
- These parts are of varying lengths, depending on the class of the address as shown above.
  Information on the Number of networks and host in each class is given below:

Class	Number of Networks	Number of Hosts	Application
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

- The IP address 0.0.0.0 is used by hosts when they are being booted.
- All addresses of the form 127.xx.yy.zz are reserved for loopback testing, they are processed locally and treated as incoming packets.

12.2.6. Subnetting

- It allows a network to be split into several parts for internal use but still act like a single network to the outside world.

- To implement subnetting, the router needs a subnet mask that indicates the split between network + subnet number and host. Ex. 255.255.252.0/22. A"/22" to indicate that the subnet mask is 22 bits long.
- Consider a class B address with 14 bits for the network number and 16 bits for the host number where some bits are taken away from the host number to create a subnet number.
-

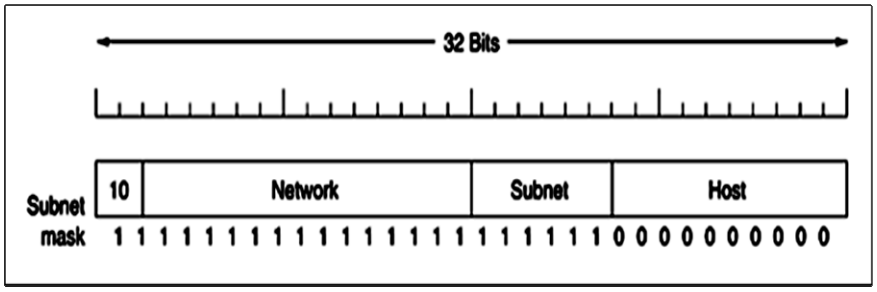


Fig: A Class B network subnetted into 64 subnets.

- If 6 bits from the host Id are taken for subnet then available bits are :

  14 bits for network + 6 bits for subnet + 10 bits for host
- With 6 bits for subnet the number of possible subnets is 2^6 which is 64.
- With 10 bits for host the number of possible host are 2^{10} which is 1022 (0 & 1 are not available)

12.2.7. CIDR

A class B address is far too large for most organizations and a class C network, with 256 addresses is too small.This leads to granting Class B address to organizations who do not require all the address in the address space wasting most of it.

This is resulting in depletion of Address space.

A solution is CIDR (Classless InterDomain The basic idea behind CIDR, is to allocate the remaining IP addresses in variable-sized blocks, without regard to the classes.

Ex. If a site needs, say, 2000 addresses, it is given a block of 2048 addresses on a 2048-byte boundary. Routing.

12.2.8. NAT (Network Address Translation)

- The scarcity of network addresses in IPv4 led to the development of IPv6.
- IPv6 uses a 128 bit address, hence it has 2^{128} addresses in its address space which is larger than 2^{32} addresses provided by IPv4.
- Transition from IPv4 to IPv6 is slowly occurring, but will take years to complete, because of legacy hardware and its incompatibility to process IPv6 address.
- NAT (Network Address Translation) was used to speed up the transition process
- The only rule is that no packets containing these addresses may appear on the Internet itself. The three reserved ranges are:
  10.0.0.0 – 10.255.255.255/8 (16,777,216 hosts)
  172.16.0.0 – 172.31.255.255/12 (1,048,576 hosts)
  192.168.0.0 – 192.168.255.255/16 (65,536 hosts)

- **Operation:**
  Within the Organization, every computer has a unique address of the form 10.x.y.z. However, when a packet leaves the organization, it passes through a NAT box that converts the internal IP source address, 10.x.y.z, to the organizations true IP address, 198.60.42.12 for example.

12.3. PROTOCOLS – ICMP, IGMP, ARP, RARP, BOOTP, DHCP

Apart from IP the network layer in the Internet has the following protocols: ICMP, ARP, RARP, BOOTP, and DHCP

12.3.1. The Internet Control Message Protocol (ICMP)

It reports the occurrences of unexpected events to source. The types of messages are as follows:

No	Message Type	Description
1	DESTINATION UNREACHABLE	It is used when a router cannot locate the destination or when a packet with the DF bit cannot be delivered to a network which allows smaller packet size.

2	TIME EXCEEDED	It sent when a packet is dropped because its hop count has reached zero suggesting that there may be congestion in the network or the timer value is set too low.
3	PARAMETER PROBLEM	It indicates that an illegal value has been detected in a header field.
4	SOURCE QUENCH	It is used to signal a fast sending source to slow down the rate of pumping packets into the network.
5	REDIRECT	It is used when a router notices that a packet seems to be routed wrong. It is used to inform the source about this.
6	ECHO	It is used to get the status of a machine : dead or alive
7	ECHO REPLY	Upon receiving and ECHO message the destination is expected to send an ECHO REPLY message if its alive
8	TIMESTAMP REQUEST	Same as ECHO, but with TIMESTAMP
9	TIMESTAMP REPLY	Same as ECHO REPLY but arrival time of the message and the departure time of the reply are recorded in the reply

12.3.2. Internet Group Management Protocol (IGMP)

- The Internet Group Management Protocol (IGMP) is one of the protocols that is involved in multicasting. Multicasting is where some processes sometimes need to send the same
- message to a large number of receivers simultaneously
- IGMP is not a multicasting routing protocol; it is a protocol that manages group

  membership.
- The IGMP protocol gives the multicast routers information about the membership status of hosts (routers) connected to the network.
- IGMP helps the multicast router create and update this list of groups in the network.

IGMP Messages

It has three types of messages:

1. **The query**

   There are two types of query messages: general and special

   a. **General Query:** used to learn which groups have members on an attached network.

   b. **Special Query:** It is a group specific query used to learn if a particular group has any members on an attached network.

2. **The membership report**

   A membership report message is sent by a host whenever it joins a multicast group, and when responding to Membership Queries sent by an IGMP router that is functioning as a Querier.

3. **The leave report**.

   This message is sent when a host leaves a multicast group. This message is sent to the 'all-routers' multicast address of 224.0.0.2. The router then sends out a *special query* to the network to verify if the last member of a group has left.

12.3.3. ARP—The Address Resolution Protocol

- The machines knowing their IP address cannot simply start sending data to other machines. The IP addresses have to be mapped onto data link layer addresses (MAC addresses) since the data link layer does not understand the 32 bit IP address.
- The MAC address is a 48-bit Ethernet address imprinted on the Network Interface card by the manufacturer.
- ARP is the means by which an IP address is translated into a physical MAC address. This is done using a broadcast request to all hosts on the network.
- By using ARP, a packet destined for a logical IP address is delivered to a computer with a Network Interface Card (NIC) with a specific hardware address.

12.3.4 RARP, BOOTP, and DHCP

ARP is used for mapping logical to physical address.

There are two occasions in which a host knows its physical address, but needs to know its

logical address. They are:

1. A diskless station is just booted. The station can find its physical address by checking

   its interface, but it does not know its IP address.

2. An organization does not have enough IP addresses to assign to each station; it needs

   to assign IP addresses on demand. The station can send its physical address and ask for a short time lease.

RARP (Reverse Address Resolution Protocol)

- Reverse Address Resolution Protocol (RARP) defines how a 48 bit MAC address is translated into a 32 bit host IP address.
- The RARP protocol was created to establish a dynamic or automated method of obtaining the IP address from a given MAC address.
- When a computer wants to find its logical address using physical address an RARP request is created and broadcasted on the local network.
- Another machine on the local network that knows all the IP addresses will respond with
- a RARP reply. The requesting machine must be running a RARP client program; the responding machine must be running a RARP server program.
- RARP has disadvantage. With broadcasting, it can send the request only to the machines in the local network. This means that if an administrator has several networks, it needs to assign a RARP server for each network.

BOOTP (Bootstrap Protocol)

- The Bootstrap Protocol (BOOTP) is a client/server protocol designed to translate physical address to logical address.
- In other words, it is a network protocol used by a network client to obtain an IP address from a server.
- BOOTP is an application layer protocol. The client and the server may be on the same network or on different networks.
- BOOTP messages are encapsulated in a UDP packet, and the UDP packet itself is encapsulated in an IP packet. The client simply uses all 0s as the source address and all 1s as the destination address.
- The BOOTP request is broadcast because the client does not know the IP address of the server.

DHCP (Dynamic Host Configuration Protocol)

- BOOTP is not a dynamic configuration protocol.
- When a client requests its IP address, the BOOTP server consults a table that matches the physical address of the client with its IP address. There is a mapping that exists between the physical and logical address that is predetermined. This approach is static.
- The disadvantage is that if a host needs to temporarily move to another network the mapping between physical and logical address that is stored in a table needs to be manually changed.
- DHCP provides static and dynamic address allocation that can be manual or automatic
- **Static Address Allocation**:

  Here DHCP acts as BOOTP. A DHCP server has a database that statically binds physical addresses to IP addresses.

- **Dynamic Address Allocation:**

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

- The dynamic aspect of DHCP is needed when a host moves from network to network. DHCP provides temporary IP addresses for a limited time. The addresses assigned from the pool are temporary addresses.

12.4. IPV6

IPv6 is an answer to the shortcomings of IPv4.

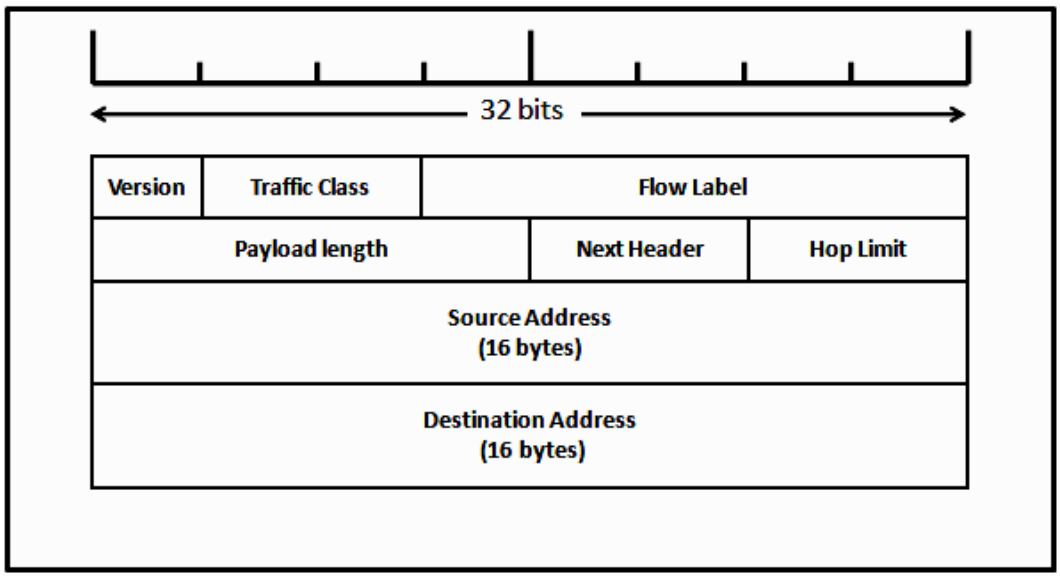
It is the other name given to SIPP (Simple Internet Protocol Plus**)**

The main features of IPv6 are:

1. Larger Address Base of 2^{128} compared to IPv4's $2^{32.}$
2. Simplification of the header. It contains only seven fields versus 13 in IPv4. This allows routers to process packets faster and thus improve throughput and delay.

3. Better support for options.
4. Provision to implement security.
5. Improved Quality of Service.

IPv6 Header



The description of the fields shown in the diagram is as follows:

No	Field Name	Description
1	Version	Value is 6 for IPv6 and 4 for IPV4
2	Traffic class	It is used to differentiate between packets with different real-time delivery requirements.
3	Flow label	This field is also still experimental. It is supposed to allow a source and destination establish a pseudo connection that has specific properties and requirements.
4	Payload length	Used to indicate the number of bytes that will follow the header.
5	Next header	There are six additional extension headers (optional). This field indicates which of the six extension headers follows the main one.
6	Hop limit	It is a counter used to limit packet lifetimes. . It must be decremented on each hop. When it hits zero, the packet is discarded and a warning packet is sent back to the source host.

7	Source address and Destination address	Both are 16 byte addresses. They are written as eight groups of four hexadecimal digits with colons between the groups, like this: 8000:0000:0000:0000:0123:4567:89AB:CDEF

References & Further Reading:

1. Computer Networks – Andrew Tannenbaum
2. Data communication & Networking Standards – Behrouz Forouzan

13

TRANSPORT LAYER

Unit Structure

13.1. Introduction
13.2. Elements of Transport protocols
    13.2.1 .Addressing
    13. 2 .2. Establishing a connection
    13.2.3. Releasing a connection
    13.2.4: Timer-based connection management
    13.2.4: Timer-based connection management
    13.2.5. Flow control and buffering
    13.2.6. Multiplexing
    13.2.7. Crash recovery
13.3. The TCP service model
    13.3.1. TCP segment header
        13.3.1.1. TCP Header Format
13.4. UDP
    13.4.1. Format
    13.4.2. Fields
13.5. Congestion control
    13.5.1.Congestion prevention

13.1. INTRODUCTION:

The first three layers of the OSI Reference Model—the physical layer, data link layer and network layer—are very important layers for understanding how networks function. The physical layer moves bits over wires; the data link layer moves frames on a network; the network layer moves datagrams on an internet work. Taken as a whole, they are the parts of a protocol stack that are responsible for the actual “nuts and bolts” of getting data from one place to another.

Immediately above these we have the fourth layer of the OSI Reference Model: the transport layer, called the host-to-host transport layer in the TCP/IP model. This layer is interesting in that

it resides in the very architectural center of the model. Accordingly, it represents an important transition point between the hardware-associated layers below it that do the “grunt work”, and the layers above that are more software-oriented and abstract.

13.2. ELEMENTS OF TRANSPORT PROTOCOLS:

Transport protocols resemble the data link protocols: error control, sequencing, and flow control.

There are major dissimilarities between the environments in which the two protocols operate.

- In the data link layer, it is not necessary for a router to specify which router it wants to talk to.

  In the transport layer, explicit addressing of destination is required.
- Initial connection setup is much more complicated in transport layer.
- The potential existence of storage capacity in the subnet requires special transport protocols.
- Buffering and flow control are needed in both layers, but the presence of a large and dynamically varying number of connections in the transport layer may require a different approach than the data link layer approach (e.g., sliding window buffer management).

13.2.1 .Addressing

How to specify the transport users?

The method normally used is to define transport addresses to which processes can listen for connection requests:

- In Internet, these end points are **(IP address, local port)** pairs.
- In ATM networks, they are **AAL-SAPs**.

We will use the neutral term **TSAP (Transport Service Access Point)**.

A possible connection scenario:

- A time-of-day server process on machine attaches itself to TSAP 122 to wait for an incoming call.
- A process on machine wants to find out the time-of-day, so it issues a CONNECT request specifying TASP 6 as the source and TSAP 122 as the destination.

- The transport entity on selects an NSAP on its machine (if it has more than one) and on the destination machine, and sets up a network connection (e.g., virtual circuit) between them (with a connectionless subnet, establishing this network layer connection would not be done). Using this network connection, it can talk to the transport entity on.
- The first thing the transport entity on says to its peer on is:``Good morning. I would like to establish a transport connection between my TSAP 6 and your TSAP 122. What do you say ?"
- The transport entity on then asks the time-of-day server at TSAP 122 if it is willing to accept a new connection. If it agrees, the transport connection is established.

How does the user process on know that the time-of-day server is attached to TSAP 122 ?

Scheme 1: Stable TSAP addresses for long life servers.
What happens if user processes want to talk to other user processes that only exist for a short time and do not have a TSAP address that is known in advance?

Scheme 2: Initial connection protocol used by UNIX hosts on the Internet.

- Each machine that wishes to offer service to remote users has a special **process server** that acts as a proxy for less-heavily used servers.
- Whenever the process server is idle, it listens to a set of ports at the same time, waiting for a TCP connection request.
- Potential users of any service must begin by doing a CONNECT request, specifying the TSAP address (TCP port) of the service they want. If no server is waiting for them, they

  get a connection to the process server, as shown Fig (a).
- After it gets the incoming request, the process server spawns off the requested server, allowing it to inherit the existing connection with the user.
- The new server then does the requested work, while the process server goes back to listening for new requests, as

  shown Fig. -(b).

How to deal with situations in which services exist independently of the process server? E.g., a file server.

Scheme 3: name server or directory server.

- To find the TSAP address corresponding to a given service name, a user sets up a connection to the name server (which listens to a well-known TSAP).
- The user then sends a message specifying the service name, and the name server sends back the TSAP address.
- Then the user releases the connection with the name server and establishes a new one with the desired service.

In this model, when a new service is created, it must register itself with the name server.

How does the local transport entity know on which machine a TSAP is located ?

More specifically, how does the transport entity know which NSAP to use to set up a network connection to the remote transport entity that manages the TSAP requested ?

The answer depends on the structure of TSAP addresses.

Hierarchical TSAP addresses:

address = <galaxy><star><planet><country><network><host><port>

TSAP = <NSAP><PORT>

Flat address space: using a name server to map TSAP into NSAP.

13. 2 .2. Establishing a connection

The problem with establishing a connection occurs when the subnet can lose, store, and duplicate packets.

Consider the following scenario:

- A user establishes a connection with a bank,
- sends messages telling the bank to transfer a large amount of money to the account of a not entirely trustworthy person,
- and then release the connection.

What happens if each packet in the above process is duplicated and stored in the subnet ?

After the connection has been released, all the packets pop out of the subnet and arrive at the destination in order, asking the bank to establish a new connection, transfer money (again), and release the connection. The bank has no way of telling that these are duplicates.

How to deal with the problem of delayed duplicated and establish connections in a reliable way?

Method 1: use throwaway TSAP addresses.

- Each time a TSAP address is needed, a new, unique address is generated, typically based on the current time.
- When a connection is released, the addresses are discarded forever.

This strategy makes the process server model impossible.

Method 2:

- Each connection is assigned a connection identifier (i.e., a sequence number incremented for each connection established), chosen by the initiating party, and put in each TPDU, including the one requesting the connection.
- After each connection is released, each transport entity could update a table (of indefinite size!) listing obsolete connections as *(peer transport entity, connection identifier)* pair.
- Whenever a connection request came in, it could be checked against the table, to see if it belonged to a previously released connection.

What happens if a machine crashes and loses its memory?

If we can ensure that no packet lives longer than some known time, the problem becomes somewhat more manageable.

What we need is a mechanism to kill off very old packets that are still wandering about.

1. Restricted subnet design. Any method preventing packets from looping.
2. Putting a hop counter in each packet.
3. Time stamping each packet.

In practice, we will need to guarantee not only that a packet is dead, but also that all acknowledgements to it are also dead.

Method 3: (due to Tomlinson)

Let be some small multiple of the true maximum packet lifetime. is protocol-dependent. If we wait a time after a packet has been sent, we can be sure that all traces of it are gone.

To get around the problem of a machine losing all memory of where it was after a crash, Tomlinson proposed to equip each host with a time of day clock.

- Each clock takes the form of a binary counter that increments itself at uniform intervals.

- The number of bits in the counter must equal or exceed the number of bits in the sequence numbers.
- The clock survives the host crashes.
- The clocks at different hosts need not be synchronized.

The **basic idea** is to ensure that two identically numbered TPDUs are never outstanding at the same time.

How to number TPDUs?

When a connection is set up, the low-order bits of the clock are used as the initial sequence number (also bits). So each connection starts numbering its TPDUs with a different sequence number.

The sequence space should be so large (e.g., bits) that by the time sequence numbers wrap around, old TPDUs with the same sequence number are long gone.

Once both transport entities have agreed (how?) on the initial sequence number, any sliding window protocol can be used for data flow control.

What to do when a host comes back after a crash?

Solution 1: keep idle for sec to let all old TPDUs die off. In a complex internet work may be too large.

Solution 2: introduce a new restriction (**forbidden region**) on the use of sequence numbers.
Suppose, and the clock tick once per second.

It is required that a sequence number should not be used if it has the potential of being used as an initial sequence number within a time.

What's the problem with this solution?

- Data transmission rate should not be faster than the clock rate, otherwise sequence numbers would fall in the forbidden region.

  This argues for a short clock tick.

- If data rate is less than the clock rate, sequence numbers may fall into the forbidden region from the left (above).

  Therefore, before sending any TPDU, the transport entity must check to see if it is about to enter the forbidden region, and if so, either delay the TPDU for time or resynchronize the sequence numbers.

The clock based method only solves the delayed duplicate problem for data TPDUs (after the connection has been established).

Since control TPDUs (e.g., connection setup TPDUs) may also be delayed, there is a potential problem in getting both sides to agree on the initial sequence number.

The **three-way handshake** establishment protocol:

13.2.3. Releasing a connection

Asymmetric release is abrupt and may result in data loss. One way to avoid data loss is to use symmetric release, in which each direction is released independently of the other one.

A more sophisticated release protocol is required to avoid data loss.

- says:``I am done. Are you done too ?"
- If responds:``I am done too. Goodbye."

This way does not always work.

The two-army problem:

- A white army is encamped in a valley.
- On both of the surrounding hillsides are blue armies.
- The white army is larger than either of the blue armies alone, but together they are larger than the white army.
- If either blue army attacks by itself, it will be defeated, but if the two blue armies attack simultaneously, they will be victorious.
- The communication medium between the two blue armies is to send messengers on foot down into the valley, where they might be captured and the message lost.

The question is, does a protocol exist that allows the blue armies to win ?

The answer is there is **NO** such protocol exists.

Just substitute ``disconnect" for ``attack". If neither side is prepared to disconnect until it is convinced that the other side is prepared to disconnect too, the disconnection will never happen.

In practice, one is usually prepared to take more risks when releasing connections than attacking white armies, so the situation is not entirely hopeless.

Three-way handshake combined with timers

In theory, this protocol can fail if the initial DR and retransmissions are all lost. The sender will give up and delete the connection, while the other side knows nothing at all about the attempts to disconnect and is still fully active. This situation results in a **half-open** connection.

One way to kill off half-open connections: If no TPSUs have arrived for a certain number of seconds, the connection is automatically disconnected.

13.2.4: Timer-based connection management

The heart of the scheme:

- When a sender wishes to send a stream of TPDUs to a receiver, it creates a **connection record** internally and starts a timer for it.
- The first TPDU contains a 1-bit flag DRF (Data Run Flag). When a TPDU with DRF flag set arrives, the receiver creates a connection record and starts a timer for it.
- If there is no TPDU sent or received for some time, the connection record timer will expire and connection record is deleted.

The connection record timer intervals for sender and receiver are different and carefully chosen so that receiver will always time out and delete its connection record well before the sender.

When any TPDU is sent, a retransmission timer is started. If after retransmissions, there is still no acknowledgement, the sender gives up. This give-up time plays an important role in the protocol.

If the first arrived TPDU does not have the DRF flag set, it is discarded.

After a connection record is created, if an incoming TPDU is in order, it is passed to the transport user; otherwise it may be buffered.

An acknowledgement is always sent but it does not imply that all the previous TPDUs have been received unless a flag ARF (Acknowledgement Run Flag) is set.

A simple case of how this protocol works:

- A sequence of TPDUs is sent and all are received in order and acknowledged.
- When the sender gets the acknowledgement of the final TPDU (with ARF flag set), it stops all the retransmission timers.
- If no more data to send, the receiver's connection record times out first and then the sender's does too.

No explicit establishment or release TPDUs is needed.

What happens if the TPDU bearing the DRF flag is lost?

The sender will time out and retransmit until it is acknowledged or the give-up timer expires.

Consider a situation: a stream of TPDUs is sent and correctly received, but some of the acknowledgements are lost. Subsequent retransmissions are also lost, so the receiver's connection record times out and is deleted.

How to prevent an old duplicate of the TPDU with the DRF flag from appearing at the receiver and triggering a new connection record?

The trick is to make the receiver's connection record timer much longer than the sender's give-up timer plus the maximum TPDU lifetime.

What happens if the transport user sends a burst of TPDUs and all acknowledged, then it gets around to sending a new burst of TPDUs?

One of three situations must hold:

1. Both sender and receiver still have their connection records

The next TPDU will carry a DRF flag and will start a new run; it will be numbered one higher than the previous TPDU (using the existing connection record).

The receiver will accept it (using the existing connection record).

2. The sender has its connection record but the receiver does not

The next TPDU will carry a DRF flag and will start a new run; it will be numbered one higher than the previous TPDU (using the existing connection record).

The receiver will create a create a new connection record and regard it as the start of a new run.

3. Both connection records have been deleted

The sender will create a new connection record and the next TPDU will not be numbered in sequence with the previous ones.

The receiver will create a new connection record and regard it as the start of a new run (it has lost its memory anyway).

One situation has been carefully avoided that the receiver knows (by existing connection record) which TPDU to expect but the sender does not know which one to send.

This protocol has an interesting mixture of connectionless and connection-oriented properties:

- For transmitting a minimum of two TPDUs (for query-response application), it is as efficient as a connectionless protocol.
- For transmitting a sequence of TPDUs, they are guaranteed to be delivered in order like a connection-oriented protocol.
- Connections are released automatically.

13.2.5. Flow control and buffering

How connections are managed while they are in use?

For flow control, a sliding window is needed on each connection to keep a fast transmitter from overrunning a slow receiver (the same as the data link layer).

Since a host may have numerous connections, it is impractical to implement the same data link buffering strategy (using dedicated buffers for each line).

The sender should always buffer outgoing TPDUs until they are acknowledged.

The receiver may not dedicate specific buffers to specific connections. Instead, a single buffer pool may be maintained for all connections. When a TPDU comes in, if there is a free buffer available, the TPDU is accepted, otherwise it is discarded.

However, for high-bandwidth traffic (e.g., file transfers), it is better if the receiver dedicate a full window of buffers, to allow the data to flow at maximum speed.

How large the buffer size should be ?

As the traffic pattern changes, the transport protocol should allow a sending host to request buffer space at the other end. Alternatively, the receiver could tell the sender ``I have reserved buffers for you.''

A general way to manage dynamic buffer allocation is to decouple the buffering from the acknowledgements.

Dynamic buffer management a variable-sized window.

- Initially, the sender requests a certain number of buffers, based on its perceived needs.
- The receiver then grants as many of these as it can afford.
- Every time the sender transmits a TPDU, it must decrement its allocation, stopping altogether when the allocation reaches zero.

- The receiver then **separately** piggybacks both acknowledgements and buffer allocations onto reverse traffic.

To prevent deadlock caused by loss of control TPDUs, each host should periodically send control TPDUs giving the acknowledgement and buffer status on each connection.

The sender's window size could be dynamically adjusted not only by the availability of the buffers at the receiver side, but also by the capacity and traffic of the subnet: the biger the subnet's carrying capacity and the lighter the traffic, the larger the sender's window (assuming buffer availability is no longer a problem).

13.2.6. Multiplexing

The reasons for multiplexing:

- To share the price of a virtual circuit connection: mapping multiple transport connections to a single network connection (**upward multiplexing**).
- To provide a high bandwidth: mapping a single transport connection to multiple network connections (**downward multiplexing**).

13.2.7. Crash recovery

In case of a **router crash**, the two transport entities must exchange information after the crash to determine which TPDUs were received and which were not. The crash can be recovered by retransmitting the lost ones.

It is very difficult to recover from a **host crash**.

Suppose that the sender is sending a long file to the receiver using a simple stop-and-wait protocol. Part way through the transmission the receiver crashes.

When the receiver comes back up, it might send a broadcast TPDU to all other hosts, requesting the other hosts to inform it of the status of all open connections before the crash.

The sender can be in one of two states: one TPDU outstanding, , or no TPDUs outstanding, .

What happens if the sender retransmits only if it is in?

Think about the situation when the receiver first sends an ACK and then perform the write (to the output stream), but crash occurs in the middle (lost TPDU).

How about reversing the order of sending ACK and performing the write?

It does not help (duplicated TPDU).

No matter how the sender and receiver are programmed, there are always situations where the protocol fails to recover properly.

13.3. The TCP service model:

TCP service is obtained by both the sender and receiver creating end points, called sockets. Each socket has a socket number consisting of the IP address of the host and a 16-bit number local to that host, called a port. A port is the TCP name for a TSAP. For TCP service to be obtained, a connection must be explicitly established between a socket on the sending machine and a socket on the receiving machine.

A socket may be used for multiple connections at the same time. In other words, two or more connections may terminate at the same socket. Connections are identified by the socket identifiers at both ends, that is, (Socket 1, Socket 2). No virtual circuit numbers or other identifiers are used.

Port numbers below 1024 are called well-known ports and are reserved for standard services. All TCP connections are full duplex and point-to-point. Full duplex means that traffic can go in both directions at the same time. Point-to-point means that each connection has exactly two end points. TCP does not support multicasting or broadcasting.

A TCP connection is a byte stream not a message stream. Message boundaries are not preserved end to end. For example, if the sending process does four 512 – byte writes to a TCP stream, these data may be delivered to the receiving process as four 512-byte chunks, two 1024-byte chunks, one 2048-byte chunks or some other way. There is no way for the receiver to detect the unit in which the data were written.

When an application passes data to TCP, TCP may send it immediately or buffer it , at its discretion. For example, suppose a user is logged in to a remote machine, After a command line has been finished and the carriage return typed, it is essential that the line be shipped off to the remote machine immediately and not buffered until the next line comes in. To force data out, applications can use the PUSH flag, which tells TCP not to delay the transmission.

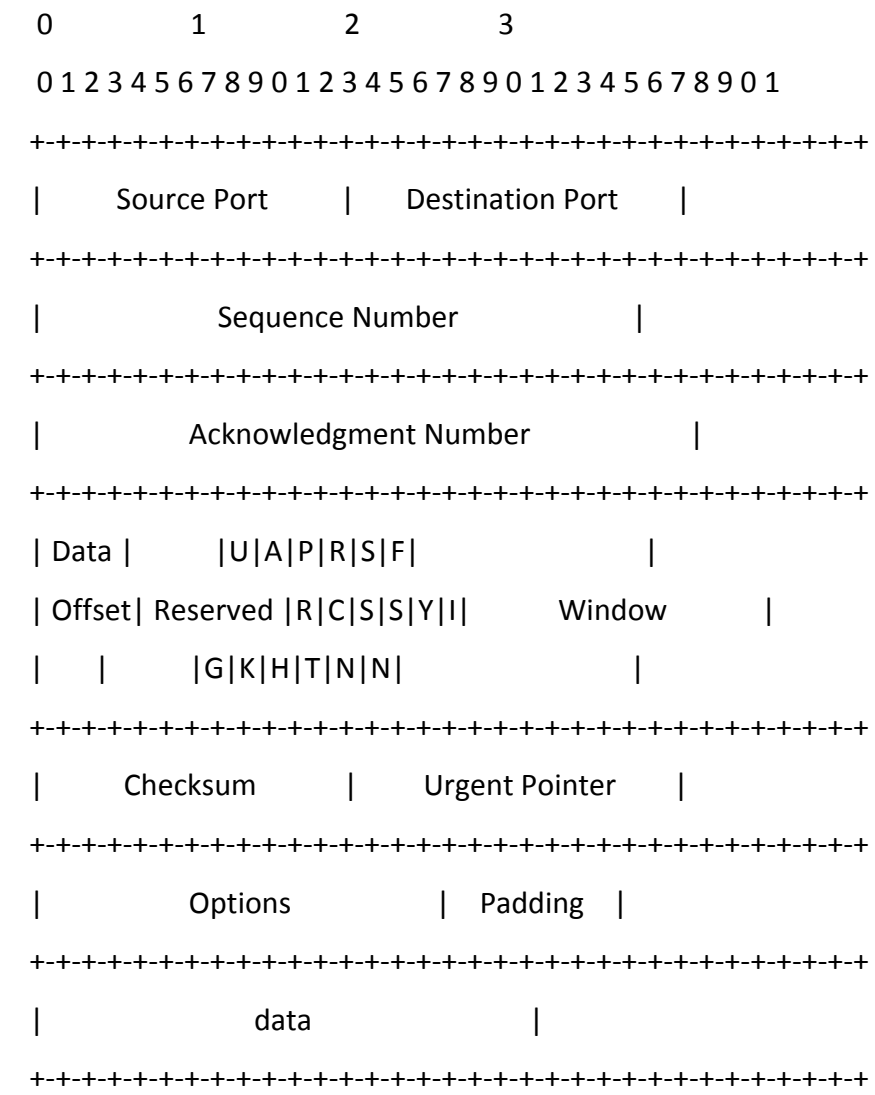
This trick sometimes works, it sometimes fails since not all implementations o TCP pass the PUSH flag to the application on the receiving side. One last feature of the TCP service that is worth mentioning here is urgent data. The end of the urgent data is marked so the application knows when it is over. The start of the urgent data is not marked. It is up to the application to figure that

out. This scheme basically provides a crude signaling mechanism and leaves everything else up to the application.

13.3.1. TCP segment header:

TCP segments are sent as internet datagrams. The Internet Protocol header carries several information fields, including the source and destination host addresses. A TCP header follows the internet header, supplying information specific to the TCP protocol. This division allows for the existence of host level protocols other than TCP.

13.3.1.1. TCP Header Format



TCP Header Format

Source Port: 16 bits indicates the source port number.

Destination Port: 16 bits indicates the destination port number.

Sequence Number: 32 bits

The sequence number of the first data octet in this segment (except when SYN is present). If SYN is present the sequence number is the initial sequence number (ISN) and the first data octet is ISN+1.

Acknowledgment Number: 32 bits

If the ACK control bit is set this field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established this is always sent.

Data Offset: 4 bits

The number of 32 bit words in the TCP Header. This indicates where the data begins. The TCP header (even one including options) is an integral number of 32 bits long.

Reserved: 6 bits, Reserved for future use. Must be zero.

Control Bits: 6 bits (from left to right):

URG: Urgent Pointer field significant

ACK: Acknowledgment field significant

PSH: Push Function

RST: Reset the connection

SYN: Synchronize sequence numbers

FIN: No more data from sender

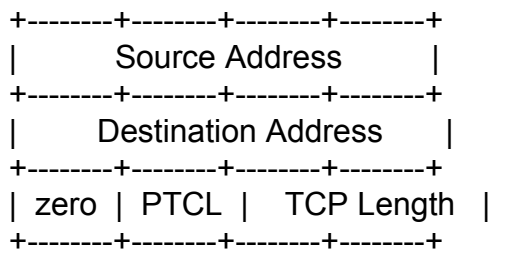
Window: 16 bits

The number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept.

Checksum: 16 bits

The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and text. If a segment contains an odd number of header and text octets to be check summed, the last octet is padded on the right with zeros to form a 16 bit word for checksum purposes. The pad is not transmitted as part of the segment. While computing the checksum, the checksum field itself is replaced with zeros.

The checksum also covers a 96 bit pseudo header conceptually prefixed to the TCP header. This pseudo header contains the Source Address, the Destination Address, the Protocol, and TCP length. This gives the TCP protection against misrouted segments. This information is carried in the Internet Protocol and is transferred across the TCP/Network interface in the arguments or results of calls by the TCP on the IP.



The TCP Length is the TCP header length plus the data length in octets (this is not an explicitly transmitted quantity, but is computed), and it does not count the 12 octets of the pseudo header.

Urgent Pointer: 16 bits

This field communicates the current value of the urgent pointer as a positive offset from the sequence number in this segment. The urgent pointer points to the sequence number of the octet following the urgent data. This field is only be interpreted in segments with the URG control bit set.

Options: variable

Options may occupy space at the end of the TCP header and are a multiple of 8 bits in length. All options are included in the checksum. An option may begin on any octet boundary. There are two cases for the format of an option:

Case 1: A single octet of option-kind.

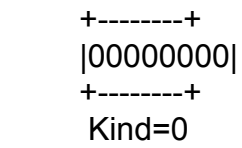
Case 2: An octet of option-kind, an octet of option-length, and the actual option-data octets.

The option-length counts the two octets of option-kind and option-length as well as the option-data octets. Note that the list of options may be shorter than the data offset field might imply. The content of the header beyond the End-of-Option option must be header padding (i.e., zero).

A TCP must implement all options. Currently defined options include (kind indicated in octal):

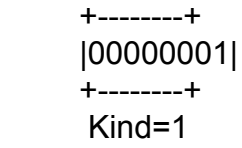
Kind	Length	Meaning
0	-	End of option list.
1	-	No-Operation.
2	4	Maximum Segment Size.

Specific Option Definitions
End of Option List



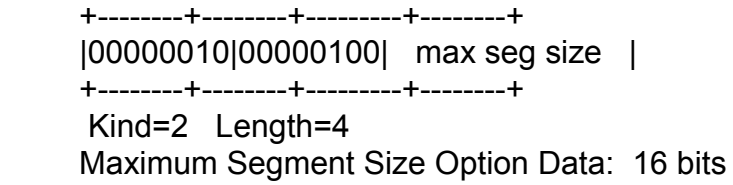
This option code indicates the end of the option list. This might not coincide with the end of the TCP header according to the Data Offset field. This is used at the end of all options, not the end of each option, and need only be used if the end of the options would not otherwise coincide with the end of the TCP header.

No-Operation



This option code may be used between options, for example, to align the beginning of a subsequent option on a word boundary. There is no guarantee that senders will use this option, so receivers must be prepared to process options even if they do not begin on a word boundary.

Maximum Segment Size



If this option is present, then it communicates the maximum receive segment size at the TCP which sends this segment. This field must only be sent in the initial connection request (i.e., in segments with the SYN control bit set). If this option is not used, any segment size is allowed.

Padding: variable

The TCP header padding is used to ensure that the TCP header ends and data begins on a 32 bit boundary. The padding is composed of zeros.

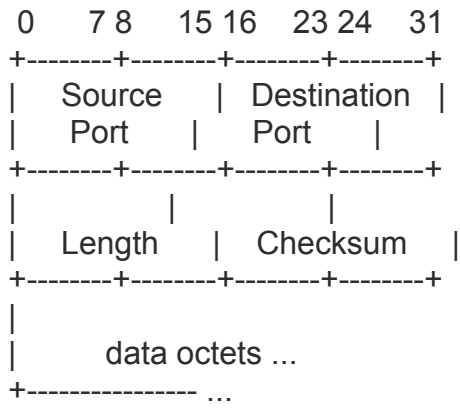
13.4. UDP:

This User Datagram Protocol (UDP) is defined to make available a datagram mode of packet-switched computer

communication in the environment of an interconnected set of computer networks. This protocol assumes that the Internet Protocol (IP) is used as the underlying protocol.

This protocol provides a procedure for application programs to send messages to other programs with a minimum of protocol mechanism. The protocol is transaction oriented, and delivery and duplicate protection are not guaranteed. Applications requiring ordered reliable delivery of streams of data should use the Transmission Control Protocol (TCP).

13.4.1. Format:



User Datagram Header Format

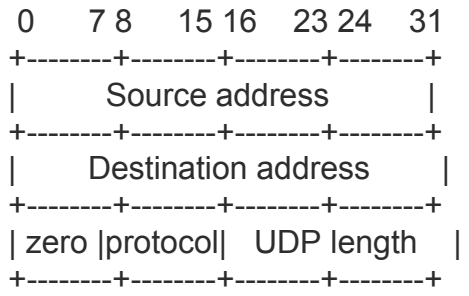
13.4.2. Fields:

Source Port is an optional field, when meaningful, it indicates the port of the sending process, and may be assumed to be the port to which a reply should be addressed in the absence of any other information. If not used, a value of zero is inserted.

Fields Destination Port has a meaning within the context of a particular Internet destination address. Length is the length in octets of this user datagram including this header and the data. (This means the minimum value of the length is eight.)

Checksum is the 16-bit one's complement of the one's complement sum of a pseudo header of information from the IP header, the UDP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

The pseudo header conceptually prefixed to the UDP header contains the source address, the destination address, the protocol, and the UDP length. This information gives protection against misrouted datagrams. This checksum procedure is the same as is used in TCP.



If the computed checksum is zero, it is transmitted as all ones (the equivalent in one's complement arithmetic). An all zero transmitted checksum value means that the transmitter generated no checksum (for debugging or for higher level protocols that don't care).

User Interface

A user interface should allow the creation of new receive ports, receive operations on the receive ports that return the data octets and an indication of source port and source address, and an operation that allows a datagram to be sent, specifying the data, source and destination ports and addresses to be sent.

13.5. CONGESTION CONTROL:

When the load to any network is more than it can handle, the congestion is increasing. Although the network layer tries to manage congestion, most of the heavy lifting is done by TCP because the real solution to congestion is to slow down the data rate.

The first step in managing congestion is detecting it. In the olden days, detecting congestion was difficult. A timeout caused by a lost packet could have been caused by either

(A) Noise on a transmission line or
(B) Packet discards at a congested router.

Nowadays, packet loss due to transmission errors is relatively rare because most long-haul trunks are fiber. Consequently, most transmission timeouts on the Internet are due to congestion.

13.5.1. Congestion prevention:

When a connection is established, a suitable window size has to be chosen. The receiver can specify a window based on its buffer size. If the sender sticks to this window size, problems will not occur due to buffer overflow at the receiving end, but they may occur due to internal congestion within the network.

Solutions:

1. The first solution is to realize that two potential problems exist – network capacity and receiver capacity which needs to be dealt with separately. To avoid the same each sender maintains two windows.
   a) The window the receiver has granted
   b) The congestion window.

   Each reflects the number of bytes the sender may transmit. The number of bytes that may be sent is the minimum of the two windows. Thus, the effective window is the minimum of what the sender and the receiver thinks as the maximum capacity

2) When a connection is established, the sender initializes the congestion window to the size of the maximum segment in use on the connection. It then sends one maximum segment. IF this segment is acknowledged before the timer goes off, it adds one segment's worth of bytes to the congestion window to make it two maximum size segments and sends two segments. As each of these segments is acknowledged, the congestion window is increased by one maximum segment size. When the congestion window is n segments, if all n are acknowledged on time, the congestion window is increased by the byte count corresponding to n segments.

3) The congestion window keeps growing exponentially until either a timeout occurs or the receiver's window is reached.

4) The idea behind is that if bursts of size 1024,2048 and 4096 bytes work fine but a burst of 8192 bytes gives a timeout, the congestion window should be set to 4096 to avoid congestion. As long as the congestion window remains at 4096, no bursts longer than that will be sent. This algorithm is called slow start.

5) Another way of implementing congestion control algorithm is by using third parameter, the threshold, initially 64KB, in addition to the receiver and congestion windows. When timeout occurs, the threshold is set to half of the current congestion window, and the congestion window is reset to one maximum segment. Slow start is then used to determine what the network can handle, except that exponential growth stops when the threshold is hit.

6) From that point on, successful transmissions grow the congestion window linearly instead of one per segment.

14

THE APPLICATION LAYER

In this Unit, we are going to discuss in detail about the following services.

Unit Structure

14.1 WWW,
14.2 HTTP,
14.3 DNS,
14.4 SNMP,
14.5 FTP,
14.6 Remote logging/Telnet and
14.7 E-mail.

14.1. THE WORLD WIDE WEB(WWW)

The Web came to us in 1994 (commercially) and allowed for everyone to work on the Internet, even though many had no idea what they were working on. The browser became the interface, a simple–to–use interface, and this was the start of the commercialization of the Web. This is when “corporate” money became involved. However, the idea started out way back in 1981 with a program called **Enquire, developed by Tim Berners–Lee**.

A program known as **Mosaic** was released in November 1993 as freeware written by the cofounder of NetScape, Marc Andreeson, at the U.S. National Center for Supercomputer Applications (NCSA). Mosaic allowed text and graphics on the same Web page and was the basis for NetScape’s Navigator browser and Microsoft’s Internet Explorer.

First and foremost, the Web allows anyone, especially nontechnical people, instant access to an infinite amount of information. You can get stock reports, information from a library, order a book, reserve airline tickets, page someone, find that long–lost friend through the yellow pages, order a data line for your house, check your credit card statement, check on the availability of that one–and–only car, provide computer–based training, or attend a private (video and audio) meeting. And yes, you can send an email.

Unlike other online services such as CompuServe, Prodigy, and America Online (at the time), anyone can create a Web page as well—not too hard to do, the language to create a Web page is pretty much English. Millions of ideas are available, and there is a pulldown menu in the browser that allows you to see the source code (the basic instructions that tell the Web server how to format a page) of any Web page.

By 1995, companies known as Internet Service Providers (ISPs) were advertising their ability to put you on the Web for a low price of $19.95. In fact, today, most professional ISPs give you space on their servers (a small amount, but enough to get started) for you to create your Web page, at no charge! Point and click to access any information that you would like; you do not have to know an operating system to move around the Web. No other "**cyberspace"** provider has the rich simplicity of the browser.

One click and you can be on a server in Japan, video conference to California, send an email to your friend in England, or plan a vacation to Breckenridge, Colorado. Other online providers had information, but it was the simplicity and combination of text and still pictures on the same page that catapulted the Web into every home. Virtually anything that you want to check on, you can do on the Web and you do not have to remember IP addresses, directory commands for DOS and Unix, file compression, executing the TAR command, printing to a postscript printer, and so on. Simply stated, the Web allows everyone access to network data with a simple click of the mouse.

On the application front, more and more applications are being written towards (or have embedded) the most common Internet interface: a browser. A browser allows the Internet to be accessed graphically using icons and pictures and a special text language known as Hypertext Markup Language, or HTML. For platform independence in writing applications for the Web, the Java language was created.

What is the downfall of the Internet? No, connectivity is generally not the problem. ISPs can be a problem, but even they are manageable. The biggest problem with the Internet is its biggest asset: **information.** You may find yourself scratching your head while traveling the Internet. Anyone can create content and post it, so there is a lot of old information on the Internet. Web pages are not kept up. Web pages are not written correctly and contain too many slow–loading graphics. Many links that are embedded in other Web pages no longer exist. Information is posted without having validity checks. Remember, no one entity owns the Internet or the Web application.

Some companies with Web pages are no longer around. All Web pages are not created equal; some take an eternity to write to your browser, while others take a minimal amount of time. Also, all ISPs are not created equal. An ISP is your connection to the Internet. Test out your ISP for service and connectivity.

Search engines bring back many undesired Web pages which require advanced searching techniques. Be careful when scrutinizing the Internet. Make sure the data is reputable (i.e., can be verified). The Internet really introduced us to the concept of trying something for free. Another concept that the Internet was not used for was known **as *shareware***, where the free samples of applications range from severely crippled (lacking many of the full–version features such as printing abilities) to the full–blown version of the software. The Web combined the two concepts, and the marketing concept really took hold when the Internet came into the business world. Every business sponsoring a Web page will give you something if you purchase something—a very old concept brought to life again via the Internet.

Most of us try a free sample before purchasing. This is still known as shareware, and payment is expected, which leads to another big problem for the Internet: How and when do you charge for something? Most users expect to surf the Internet, pick up what they want for free, and then sign off. Sorry dears, we don't live in a free world, and eventually you must pay. Unfortunately, there are those out there who continue to download software and not pay for it. If this continues, shareware will not be available, and you will end up with a pay–first, try–later attitude.

Another problem of the Internet is the spread of viruses. Protect your workstation with some type of antiviral software before downloading anything from the Internet. Most protection schemes are dynamic in that they are constantly checking for viruses even during an email download or a file transfer. Here is where the other online providers do have an advantage. Private online providers such as America Online and CompuServe make every effort to test uploaded software and generally do not allow for content to be written to their servers. You will find those services more protected and watched over than the Internet.

14.2 HTTP

Introduction:

HTTP defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands. For example, when you enter a URL in your browser, this actually sends an HTTP command to the Web server directing it to fetch and transmit the requested Web

page. The other main standard that controls how the World Wide Web works is HTML, which covers how Web pages are formatted and displayed.

HTTP is called a *stateless* protocol because each command is executed independently, without any knowledge of the commands that came before it. This is the main reason that it is difficult to implement Web sites that react intelligently to user input. This shortcoming of HTTP is being addressed in a number of new technologies, including ActiveX, Java, JavaScript and cookies.

S-HTTP is an extension to the HTTP protocol to support sending data securely over the World Wide Web. S-HTTP was developed by Enterprise Integration Technologies (EIT), which was acquired by Verifone, Inc. in 1995. Not all Web browsers and servers support S-HTTP.

Another technology for transmitting secure communications over the World Wide Web -- *Secure Sockets Layer (SSL)* -- is more prevalent.

However, SSL and S-HTTP have very different designs and goals so it is possible to use the two protocols together. Whereas SSL is designed to establish a secure connection between two computers, S-HTTP is designed to send individual messages securely. Both protocols have been submitted to the Internet Engineering Task Force (IETF) for approval as a standard.

The **Hypertext Transfer Protocol** (**HTTP**) is a networking protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web. The standards development of HTTP has been coordinated by the Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C), culminating in the publication of a series of Requests for Comments (RFCs), most notably RFC 2616 (June 1999), which defines HTTP/1.1, the version of HTTP in common use.

Technical overview:

HTTP functions as a request-response protocol in the client-server computing model. In HTTP, a web browser, for example, acts as a *client*, while an application running on a computer hosting a web site functions as a *server*. The client submits an HTTP *request* message to the server. The server, which stores content, or provides *resources*, such as HTML files, or performs other functions on behalf of the client, returns a response message to the client. A response contains completion status information about the request and may contain any content requested by the client in its message body.

A client is often referred to as a *user agent* (UA). As well as web browsers, web crawlers are another common user agent. These include the indexing software used by search providers. Voice browsers are another less common but important class of user agent.

The HTTP protocol is designed to permit intermediate network elements to improve or enable communications between clients and servers. High-traffic websites often benefit from web cache servers that deliver content on behalf of the original, so-called *origin server* to improve response time. HTTP proxy servers at network boundaries facilitate communication when clients without a globally routable address are located in private networks by relaying the requests and responses between clients and servers.

HTTP is an Application Layer protocol designed within the framework of the Internet Protocol Suite. The protocol definitions presume a reliable Transport Layer protocol for host-to-host data transfer.[2] The Transmission Control Protocol (TCP) is the dominant protocol in use for this purpose. However, HTTP has found application even with unreliable protocols, such as the User Datagram Protocol (UDP) in methods such as the Simple Service Discovery Protocol (SSDP).

HTTP Resources are identified and located on the network by Uniform Resource Identifiers (URIs)—or, more specifically, Uniform Resource Locators (URLs)—using the http or https URI. URIs and the Hypertext Markup Language (HTML), form a system of inter-linked resources, called hypertext documents, on the Internet.

The original version of HTTP (HTTP/1.0) was revised in HTTP/1.1. HTTP/1.0 uses a separate connection to the same server for every request-response transaction, while HTTP/1.1 can reuse a connection multiple times, to download, for instance, images for a just delivered page. Hence HTTP/1.1 communications experience less latency as the establishment of TCP connections presents considerable overhead.

HTTP session:

An HTTP session is a sequence of network request-response transactions. An HTTP client initiates a request. It establishes a Transmission Control Protocol (TCP) connection to a particular port on a host (typically port 80). An HTTP server listening on that port waits for a client's request message. Upon receiving the request, the server sends back a status line, such as "HTTP/1.1 200 OK", and a message of its own, the body of which is perhaps the requested resource, an error message, or some other information.

Request message:

The request message consists of the following:

- Request line, such as GET /images/logo.png HTTP/1.1, which requests a resource called /images/logo.png from server
- Headers, such as Accept-Language: en
- An empty line

An optional message body:

The request line and headers must all end with <CR><LF> (that is, a carriage return followed by a line feed). The empty line must consist of only <CR><LF> and no other whitespace. Although <CR><LF> is required <LF> alone is also accepted by most servers In the HTTP/1.1 protocol, all headers except Host are optional. A request line containing only the path name is accepted by servers to maintain compatibility with HTTP clients before the HTTP/1.0 specification in RFC1945

Request methods:

An HTTP request made using telnet. The request, response headers and response body are highlighted. HTTP defines nine methods (sometimes referred to as "verbs") indicating the desired action to be performed on the identified **resource**. What this resource represents, whether pre-existing data or data that is generated dynamically, depends on the implementation of the server. Often, the resource corresponds to a file or the output of an executable residing on the server.

1. **HEAD-** Asks for the response identical to the one that would correspond to a GET request, but without the response body. This is useful for retrieving meta-information written in response headers, without having to transport the entire content.

2. **GET-** Requests a representation of the specified resource. Requests using GET (and a few other HTTP methods) "SHOULD NOT have the significance of taking an action other than retrieval". The W3C has published guidance principles on this distinction, saying, "Web application design should be informed by the above principles, but also by the relevant limitations."

3. **POST**- Submits data to be processed (e.g., from an HTML form) to the identified resource. The data is included in the body of the request. This may result in the creation of a new resource or the updates of existing resources or both.

4. **PUT-** Uploads a representation of the specified resource.

5. **DELETE-** Deletes the specified resource.

6. **TRACE-** Echoes back the received request, so that a client can see what (if any) changes or additions have been made by intermediate servers.
7. **OPTIONS-** Returns the HTTP methods that the server supports for specified URL. This can be used to check the functionality of a web server by requesting '*' instead of a specific resource.
8. **CONNECT -** Converts the request connection to a transparent TCP/IP tunnel, usually to facilitate SSL-encrypted communication (HTTPS) through an unencrypted HTTP proxy.
9. **PATCH- Is** used to apply partial modifications to a resource. HTTP servers are required to implement at least GET and HEAD methods and, whenever possible, also the OPTIONS method.

Safe methods:

Some methods (for example, HEAD, GET, OPTIONS and TRACE) are defined as *safe*, which means they are intended only for information retrieval and should not change the state of the server. In other words, they should not have side effects, beyond relatively harmless effects such as logging, caching and the serving of banner advertisements or incrementing a web counter. Making arbitrary GET requests without regard to the context of the application's state should therefore be considered safe.

By contrast, methods such as POST, PUT and DELETE are intended for actions that may cause side effects either on the server, or external side effects such as transactions or transmission of email. Such methods are therefore not usually used by conforming web robots or web crawlers; some that do not conform tend to make requests without regard to context or consequences.

Despite the prescribed safety of *GET* requests, in practice their handling by the server is not technically limited in any way. Therefore, careless or deliberate programming can cause non-trivial changes on the server. This is discouraged, because it can cause problems for Web caching, search engines and other automated agents, which can make unintended changes on the server.

Furthermore, methods such as TRACE, TRACK and DEBUG are considered potentially 'unsafe' by some security professionals, because they can be used by attackers to gather information or bypass security controls during attacks. Security software tools such as Tenable Nessus and Microsoft URLScan report on the presence of these methods as being security issues.

Idempotent methods and web applications:

Methods PUT and DELETE are defined to be idempotent, meaning that multiple identical requests should have the same effect as a single request. Methods GET, HEAD, OPTIONS and TRACE, being prescribed as safe, should also be idempotent, as HTTP is a stateless protocol.

In contrast, the POST method is not necessarily idempotent, and therefore sending an identical POST request multiple times may further affect state or cause further side effects (such as financial transactions). In some cases this may be desirable, but in other cases this could be due to an accident, such as when a user does not realize that their action will result in sending another request, or they did not receive adequate feedback that their first request was successful. While web browsers may show alert dialog boxes to warn users in some cases where reloading a page may re-submit a POST request, it is generally up to the web application to handle cases where a POST request should not be submitted more than once.

Note that whether a method is idempotent is not enforced by the protocol or web server. It is perfectly possible to write a web application in which (for example) a database insert or other non-idempotent action is triggered by a GET or other request. Ignoring this recommendation, however, may result in undesirable consequences, if a user agent assumes that repeating the same request is safe when it isn't.

Status codes:

In HTTP/1.0 and since, the first line of the HTTP response is called the *status line* and includes a numeric *status code* (such as "404") and a textual *reason phrase* (such as "Not Found"). The way the user agent handles the response primarily depends on the code and secondarily on the response headers. Custom status codes can be used since, if the user agent encounters a code it does not recognize, it can use the first digit of the code to determine the general class of the response. Also, the standard *reason phrases* are only recommendations and can be replaced with "local equivalents" at the web developer's discretion. If the status code indicated a problem, the user agent might display the *reason phrase* to the user to provide further information about the nature of the problem. The standard also allows the user agent to attempt to interpret the *reason phrase*, though this might be unwise since the standard explicitly specifies that status codes are machine-readable and *reason phrases* are human-readable.

Persistent connections:

In HTTP/0.9 and 1.0, the connection is closed after a single request/response pair. In HTTP/1.1 a keep-alive-mechanism was

introduced, where a connection could be reused for more than one request. Such *persistent connections* reduce request latency perceptibly, because the client does not need to re-negotiate the TCP connection after the first request has been sent. Version 1.1 of the protocol made bandwidth optimization improvements to HTTP/1.0. For example, HTTP/1.1 introduced chunked transfer encoding to allow content on persistent connections to be streamed, rather than buffered. HTTP pipelining further reduces lag time, allowing clients to send multiple requests before a previous response has been received to the first one. Another improvement to the protocol was byte serving, which is when a server transmits just the portion of a resource explicitly requested by a client.

HTTP session state:

HTTP is a stateless protocol. A stateless protocol does not require the server to retain information or status about each user for the duration of multiple requests. For example, when a web server is required to customize the content of a web page for a user, the web application may have to track the user's progress from page to page. A common solution is the use of HTTP cookies. Other methods include server side sessions, hidden variables (when the current page is a form), and URL-rewriting using URI-encoded parameters,
e.g., /index.php?session_id=some_unique_session_code.

Secure HTTP:

There are three methods of establishing a secure HTTP connection: HTTP Secure, Secure Hypertext Transfer Protocol and the HTTP/1.1 Upgrade header. Browser support for the latter two is, however, nearly non-existent,[*citation needed*] so HTTP Secure is the dominant method of establishing a secure HTTP connection.

Example session:

Below is a sample conversation between an HTTP client and an HTTP server running on www.example.com, port 80.

Client request

GET /index.html HTTP/1.1
Host: www.example.com

A client request (consisting in this case of the request line and only one header) is followed by a blank line, so that the request ends with a double newline, each in the form of a carriage return followed by a line feed. The "Host" header distinguishes between various DNS names sharing a single IP address, allowing name-based virtual hosting. While optional in HTTP/1.0, it is mandatory in HTTP/1.1.

Server response:

- HTTP/1.1 200 OK
- Date: Mon, 23 May 2005 22:38:34 GMT
- Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
- Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
- Etag: "3f80f-1b6-3e1cb03b"
- Accept-Ranges: bytes
- Content-Length: 438
- Connection: close
- Content-Type: text/html; charset=UTF-8

A server response is followed by a blank line and text of the requested page. The ETag (entity tag) header is used to determine if a cached version of the requested resource is identical to the current version of the resource on the server. *Content-Type* specifies the Internet media type of the data conveyed by the http message, while *Content-Length* indicates its length in bytes. The HTTP/1.1 web server publishes its ability to respond to requests for certain byte ranges of the document by setting the header *Accept-Ranges: bytes*. This is useful, if the client needs to have only certain portions of a resource sent by the server, which is called byte serving. When *Connection: close* is sent in a header, it means that the web server will close the TCP connection immediately after the transfer of this response.

Most of the header lines are optional. When *Content-Length* is missing the length is determined with other ways. Chunked transfer encoding uses a chunk size of 0 to mark the end of the content. *Identity* encoding without *Content-Length* reads content until the socket is closed.

A *Content-Encoding* like *gzip* can be used to reduce the amount of data.

14.3. DOMAIN NAME SERVICE (DNS)

There are millions of hosts on the Internet today representing even more millions of users. Most users have no idea what the underlying protocols are doing, nor do they care. But most of them would if they had to memorize IP addresses and determine other functions such as mail.

Actually, most would be frustrated by the numbering system and the Internet would not be as popular as it is. When the Internet

was young, an early method of mapping the 32-bit address to a hostname required downloading a file maintained by (at the time) the Network Information Center (NIC). It was a single file (hosts.txt) that contained a simple mapping of Internet addresses to hostnames. This file was usually contained in the /etc subdirectory on a workstation and various TCP/IP applications could access the information in this file. Not having this file meant that a user had to type in the 32-bit address for connectivity to a remote host.

Secondly, population of the Internet was becoming very diverse and more autonomous. In the 1980s the Internet was known as the ARPAnet (now shut down) and the hosts were primarily time shared. More and more connections to the Internet were sites that had LANs installed and connected to these LANs were mainframe and minicomputers or even personal computers. These sites were administering their own names and addresses in the hosts.txt file, but had to wait for the NIC to change hosts.txt to make changes visible to the Internet at large.

Lastly, with the additions of more sites to the Internet, the applications on the Internet were getting more sophisticated and creating a need for a general-purpose name service.

After many experimental RFCs, the global name system for the Internet became known as the Domain Name System (DNS). DNS is comprised of three components:

- a *name server*,
- a *database*, and
- a *name resolver*.

Name servers make information available to the resolvers. The information the name servers contain is IP addresses, aliases, mail information, and so forth. The resolvers usually reside on users' workstations and are embedded in the applications of TCP such as TELNET and FTP. They are not separate programs. The name server is a separate program and resides anywhere on a network answering queries from the resolvers. The domain servers each maintain a portion of the hierarchical database under separate administrative authority and control. Redundancy is obtained by transferring data between cooperating servers (primary masters and secondary masters).

DNS Structure:

The Domain Name Space is very much like a file system on Unix or DOS. It starts with a root and branches attach from this root to give an endless array of paths. Each branch in the file system is given a directory name, whereas in DNS it is called a ***label*.** Each label can be 63 characters in length, but most are far less than that. This means that each text word between the dots can be 63

characters in length, with the total domain name (all the labels) limited to 255 bytes in overall length assuming a screen line length of 80 characters this is just 3 screen lines.

The IP protocol mandates the use of IP addresses. Any user may use this address to connect to any service on the network; however, for a user to remember the addresses of the entire network servers on the network are an impossible task. Users are more likely to remember names than they are to remember numbers. For those familiar with database environments, the domain name server are simply a database (consisting of information such as names and IP addresses, and much more) to which any station on the network can make queries using the domain name resolver.

The domain name system is not necessarily complex, but it is involved. It is based on a hierarchical structure. The domain name is simply that: a name assigned to a domain. For example, isi.edu, cisco.com, and 3Com.com represent the domain name at those companies or educational institutions.

DNS Components:

DNS does much more than the name-to-address translation. It also allows for the following components.

- Domain Name Space and resource records
- Name servers
- Resolvers

The Domain Name Space and resource records:

This is the database of grouped names and addresses that are strictly formatted using a tree-structured name space and data associated with the names. The domain system consists of separate sets of local information called ***zones*.** The database is divided up into sections called *zones*, which are distributed among the name servers.

While name servers can have several optional functions and sources of data, the essential task of a name server is to answer queries using data in its zones. Conceptually, each node and leaf of the domain name space tree names a set of information, and query operations are attempts to extract specific types of information from a particular set. A query names the domain name of interest and describes the type of resource information that is desired.

Name servers:

These are workstations that contain a database of information about hosts in zones. This information can be about well-known services, mail exchanger, or host information. A name server may cache structure or set information about any part of the

domain tree, but in general, a particular name server has complete information about a subset of the domain space, and pointers to other name servers that can be used to lead to information from any part of the domain tree. Name servers know the parts of the domain tree for which they have complete information; a name server is said to be an authority for these parts of the name space.

Authoritative information is organized into units called *zones*, and these zones can be automatically distributed to the name servers that provide redundant service for the data in a zone. The name server must periodically refresh its zones from master copies in local files or foreign name servers.

Resolvers:

These are programs that generally reside on users' workstations and send requests over the network to servers on behalf of the users. Resolvers must be able to access at least one name server and use that name server's information to answer a query directly, or pursue the query using referrals to other name servers.

When a DNS server responds to a resolver, the requester attempts a connection to the host using the IP address and not the name. The preceding example could have used only part of a name: host. This is known as a ***relative name***. It is part of a larger name known as the ***absolute name***. The absolute name for the preceding example could be host.research.Naugle.com. This name would be in the domain name server. Most resolvers will step through a preconfigured list of suffixes (in order of configured input), append it to the name, and attempt a lookup when the full DNS (absolute) name is not specified.

Domain Structure:

DNS is hierarchical in structure, as shown previously. A domain is a sub-tree of the domain name space. The advantage of this structure is that at the bottom, the network administrator can assign the names. From the root, the assigned top-level domains (TLD) are as follows:

- GOV Government body.
- EDU Educational body.
- COM Commercial entity.
- MIL Military.
- ORG Any other organization not previously listed.
- CON- Any country using the ISO standard 3166 for names of countries As stated in RFC 1591, "the IANA is not in the business of what is and what is not a country."

Therefore, it is up to ISO to determine who is on that list.

Now let's look at the generalized format for a domain name.

Going down the tree, we can pick out a domain name, such as research.Naugle.com. This would signify the Research department (which is a sub-domain of domain Naugle.com) at Naugle Enterprises, which is defined as a commercial entity of the Internet. Naugle.com can be a node in the domain acting as a name server, or there may be different name servers for Naugle.com.

A user at workstation 148.1.1.2 types in the TELNET command and the domain name of host1.research.Naugle.com. This workstation must have the domain name resolver installed on it. This program would send out the translation request to a domain name server to resolve the hostname-to-IP address. If the hostname is found, the domain name server would return the IP address to the workstation. If the name is not found, the server may search for the name elsewhere and return the information to the requesting workstation, or return the address of a name server that the workstation (if able) can query to get more information.

A domain contains all hosts whose domain names are within a certain domain. A domain name is a sequence of labels separated by dots. A domain is a sub-domain of another domain if it is contained within that domain. This relationship can be tested by seeing if the sub-domain's name ends with the containing domain's name. For example, research.Naugle.com is a sub-domain of Naugle.com. Naugle.com is a sub-domain of .com, and " " (root).

There are special servers on the Internet that provide guidance to all name servers.

These are known as root name servers and, as of this writing, there are nine of them. They do not contain all information about every host on the Internet, but they do provide direction as to where domains are located (the IP address of the name server for the uppermost domain a server is requesting). The root name server is the starting point to find any domain on the Internet. If access to the root servers ceased, transmission over the Internet would eventually come to a halt.

14.4. SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

Network management is divided into five categories:

- Account management
- Fault management
- Security
- Configuration management
- Performance

Account management: Gathers information on which users or departments are employing which network services.

Fault management: Includes troubleshooting, finding, and correcting failed or damaged components, monitoring equipment for early problem indicators, and tracking down distributed problems.

Security: Includes authorization, access control, data encrypting, and management of encrypting keys.

Configuration management: Tracks hardware and software information. Included with this are administration tasks, such as day-to-day monitoring and maintenance of the current physical and logical state of the network, as well as recognition and registrations of applications and services on the network.

Performance: The monitoring of traffic on the network.

Elements of SNMP:

There are several elements that comprise SNMP and they all must work together in order for SNMP to operate. They are

- Management Server
- Management Clients
- Agent and
- MIB

Management server: The network station that runs the management application to monitor or control the management clients.

Management clients: The network station that contains the agent (a software component), which enables the management server to control and monitor them.

The **agent** can be located in any directly attached network device such as router, a PC, a switch, etc. SNMP is a Request/Response protocol that allows for the exchange of information between the server and an agent. This protocol does not define the items that can be managed.

MIB: The Management Information Base is a collection of objects that contain information that will be utilized by a network management server. It contains all of this information under an entity known as *object*. Similar objects are placed together to form groups. In other words, the MIB is a database. It is a collection of objects formed into groups, each of which contains information that will be given based on a request from a management station.

SNMP Manager:

An SNMP manager is a software application that queries the agents for information, or sets information on the client agent. The returned information is then stored in a database to be manipulated by other application software that is not defined by SNMP. The information gathered can be used to display graphs of how many bytes of information are transmitted out a port, how many errors have occurred, and so forth. SNMP simply sets or gathers information in a node. Therefore, the server will be comprised of two things:

Management applications:

Applications that can receive and process the information gathered by the SNMP manager. These applications are the ones that have some type of user interface to allow the network manager to manipulate the SNMP protocol; for example, set the SNMP node that it would like to talk to, send that node information, get information from that node, and so on.

Databases:

The information that is stored in the database is from the configuration, performance, and audit data of the agents. There are multiple databases on the server: the MIB database, the Network Element database, and the Management Application databases (Topology database, History Log, Monitor Logs). All of this runs on top of SNMP. It is not necessary for SNMP to operate, but it does allow for the human factor to fit in.

Agent:

Agents are simple elements that have access to a network's elements (router, switch, PC, etc.) MIB. Agents are the interface from the network management server to the client MIB. They perform server-requested functions. When a server requests information from a client, it will build its SNMP request and send it, unicast, to the client. The agent receives this request, processes it, retrieves or sets the information in the MIB of the client, and generates some type of response to the server. Usually, agents only transmit information when requested by a server.

However, there is one instance in which an agent will transmit unwanted information.

It is known as a ***trap*****.** There are certain things on a network station that may force it to immediately notify the server. Some of these traps are defined by the RFC. Things such as cold/warm start and authentication failure are traps that are sent to the server. Most agent applications today permit the use of user-defined traps. This means that the network administrator of an SNMP compliant device can configure the router to send traps to the server when certain

conditions are met. For example, a router may send a trap to the server when its memory buffers constantly overflow or when too many ICMP redirects have been sent.

Another type of agent is known as the ***proxy agent***. This allows one station to become an agent for another network station that does not support SNMP. Basically, proxy agents' server acts as translators between servers and non-SNMP capable clients (for security, limited resources, etc.).

Management Information Base (MIB):

The MIB is a collection of objects that contain specific information that together form a group. You can think of a MIB as a database that contains certain information that was either preset (during configuration of the node) or was gathered by the agent and placed into the MIB. Simply stated, the MIB is a database that contains information about the client that is currently placed on it. The structure of management information is defined in RFC 1155 and defines the format of the MIB objects. This includes the following:

- Collection of objects that contain specific information that together form a group.
- Structure of Management Information is specified in RFC 1155.

The objects contain the following:

- Syntax
- Access
- Status
- Description
- Index
- DefVal
- Value notation

Syntax (required): It is the abstract notation for the object type. This defines the data type that models the object.

Access (required): It defines the minimal level of support required for the object types. It must be one of read-only, read-write, write-only, or not accessible.

Status (required): It denotes the status of the MIB entry. It can be mandatory, optional, obsolete, or deprecated (removed).

Description (Optional): It is a text description of the object type.

Index-Present only if the object type corresponds to a row in a table.

DefVal (Optional): It defines a default value that may be assigned to the object when a new instance is created by an agent.

Value Notation- The name of the object, which is an Object Identifier.

Example of a MIB Entry:

Object:

IfOperStatus {ifEntry 8}

Syntax:

INTEGER {
up (1) — ready to pass packets
down(2),
testing (3) — in some test mode
}

Definition:

The current operational state of the interface. The testing (3) state indicates that no operational packets can be passed.

Access:

Read-only.

Status:

Mandatory.

SNMP is the protocol that is used between a manager and a client. SNMP uses a series of SNMP commands and protocol data units (PDUs) to send and receive management information. SNMP was eventually to be migrated to the OSI management scheme. Therefore an encoding scheme known as Abstract Syntax Notation, or ASN.1, was used. Only the INTEGER, OCTET, STRING, OBJECT IDENTIFIER, NULL, SEQUENCE, AND SEQUENCE OF are used. SNMP uses UDP as its transport layer. The following are the SNMP protocol data unit (PDU Packet) types:

GetRequest- Requests an agent to return attribute values for a list of managed objects.

GetNextRequest- Used to traverse a table of objects. Since the object attributes are stored in lexicographical order, the result of the previous GetNextRequest can be used as an argument in a subsequent GetNextRequest. In this way, a manager can go through a variable-length table until it has extracted all the information for the same types of objects.

GetResponse- Returns attribute values for the selected objects or error indications for such conditions as invalid object name or nonexistent object.

SetRequest - Used to change the attribute values of selected objects.

Trap- Used by the agent. Traps are used to report certain error conditions and changes of state to the managing process. The

conditions are cold-start, warm-start, link-up, link-down, EGP neighbor loss (not much use for this one anymore), and authentication failure.

SNMP provides for a simple authentication process between the client and the server. This is known as the ***community string*** and it must match between a client and the server. This string is embedded in the protocol packet and if either side has a different entry, the received SNMP packet is discarded. This community string is manually configured on the server and the client. The problem is that it is not encrypted in any way when it is transmitted. Any protocol analyzer that is on the same link as this packet can see the community string name.

SNMP Encapsulation:

The community string is a simple password protection mechanism. Most are set to “public” for read access and “private” for read-write access. This field is established at set up time. The variable binding indicates which groups or which objects in the groups are being requested for information.

14.5. FILE TRANSFER PROTOCOL (FTP)

TELNET provides users with the ability to act as a local terminal even though users are not directly attached to the host. One other TCP/IP application that provides network services for users on a network is a ***file transfer protocol***. With TCP/IP, there are three popular types of file access protocols in use:

- FTP,
- Trivial File Transfer Protocol (TFTP), and
- Network File System (NFS).

This FTP protocol provides for files to be transferred reliably across the network under the complete control of the user. FTP is transaction based. Every command is replied to using a number scheme similar to the SMTP protocol.

FTP is very robust. The FTP protocol actually uses two port assignments (and therefore two connections): 20 and 21. Remember that most connections between two network stations are made via one source port and one destination port. A network station wanting a connection to a remote network station must connect to two ports on the destination station in order for FTP to work. Port 20 is used for the initial setup of the connection and as the control connection. No data passes over this circuit except for control information. Port 21 is used for user data (the file to be transferred) to pass over the connection.

Similar to the TELNET arguments, simply typing **FTP <domain name or IP address>** will establish the connection. The command line should then read **FTP>** (this depends on your application). With the advent of Windows and Windows-like operating systems, FTP now has a GUI interface in order to take some of the harshness out of the protocol.

After the connection is established, the server process awaits a command from the client. To transfer a file from the server to the client, the user types **get <a name of a file>**, which is transmitted over to the remote network station. With this, a second connection is established between the server and client FTP process. It is known as the ***data connection***. Now we have two connections, but only during the file transfer process.

Once the file is transferred, the data connection port is closed. This is the well-known (or assigned) FTP data port. From a user's standpoint, to establish a connection between itself and a remote station, the command is similar to TELNET: **FTP <domain name or IP address>**. A user could also type in FTP and wait for the FTP prompt. At the prompt, the user would use the OPEN command to establish the connection.

Commands of FTP:

The following are the available commands in FTP. There are a lot of commands listed but, in reality, only a few are used. They are:

- **Open-** Opens a connection to a remote resource and creates a connection between two hosts.
- **Close-** Closes a connection to a remote resource. Thus terminates a connection between two hosts.
- **bye-** End this FTP session.
- **Binary-** Indicate that the file transfer will be a file of binary type (i.e., executable file, Lotus file, etc.).
- **get -**Get a file from the remote resource; get <filename>
- **mget-** <multiple files,wildcards included>.
- **put -**Puts a file to the remote resource; put <filename>
- **mput-** <multiple files,wildcards included>.
- **Cd-** Change directory on the remote device.
- **Lcd-** to change the directory on the local end.
- **Dir-** Get a directory listing on the remote device.
- **Ldir-** to get a directory listing on the local end.
- **hash** Display hash marks on the screen to indicate a file is being transferred.

A Sample FTP Data Transfer:

Once the connection is established, file transfer actually occurs over the data port. If a user wants to establish an FTP connection between 148.1.1.2 and an FTP server process on 148.1.1.19, the following sequence of events take place on a DOS PC (for other operating systems, the prompt would change, but the commands are all the same in every FTP implementation):

C> FTP or FTP <domain name or IP address>

If multiple files are needed, the user can use the commands MGET or MPUT, which stands for Multiple GET and Multiple PUT, respectively. If the file we want is a binary file (a spreadsheet and an application are examples of binary files), the user has to type in the keyword **binary** at the FTP prompt. This indicates to the FTP program that the file to be transferred is a binary file. Any of the commands may be entered at the FTP prompt. The protocol is transaction based and the numbers preceding each line are for the node to interpret the next command.

C:\WINDOWS>ftp
ftp> open mnauglepc
Connected to mnauglepc.
220 mnauglepc FTP service (NEWT v4.01) ready for new user.
User (mnauglepc:(none)): mnaugle
331 mnaugle, please enter your password.
Password:
230 User mnaugle logged in.
ftp> pwd
257 "c:\" is the current directory.
ftp> lcd
Local directory now C:\WINDOWS
ftp> get autoexec.bat autoexec.002
200 PORT command successful.
150 Opening ASCII mode data connection for autoexec.bat.
226 File transfer complete.
1911 bytes received in 0.00 seconds (1911000.00 Kbytes/sec)
ftp> bye
221 Goodbye.

14.6. REMOTE LOGGING/TELNET

The TELNET connection simply allows a terminal service over the TCP/IP network as if the terminal were directly connected. Remember that computers and terminals were connected by a cable and the terminals were directly attached to the host computer. The TELNET service provides a terminal service for the

network. It allows for any terminal to attach to any computer over the network. It can emulate many different types of terminals, depending on the manufacturer of the TELNET program The advantage to the TELNET program is that a user may log on to any host on the TCP/IP internet (provided security options are allowed). Sessions are set up over the TCP/IP network.

The TELNET protocol uses TCP as its transport. The user starts the TELNET protocol at his or her workstation, usually by typing **TELNET <domain name or IP address>**. The TELNET application may be started with or without an argument. The argument allows a simpler procedure to be invoked so that the TELNET process will automatically try to connect to the host signified by the argument statement. The TELNET application starts and attempts to establish a connection to the remote device. If an argument is not supplied, the TELNET application waits for the user to issue an OPEN command connection using the DNS or an IP address.

Options of TELNET:

- Each side of the connection requests or tells its partner the options it wants or can do.
- Options are formatted in:
    - WILL or WON'T <option>
    - DO or DON'T <option>
    - Negotiates options such that symmetry can be set up between two stations.
- Options include:
    - Ability to echo
    - Terminal type
    - Setting line mode so that groups of characters can be sent

The TELNET program is extensible through the use of options. Each side of the connection requests or tells the remote end of the connection which of these options it can support and which one the remote end should support. This provides for symmetry. The TELNET protocol was written so that it would work on a variety of operating systems. Therefore, before a connection is made to the remote device, the TELNET protocol has some work to do in order to synchronize the connection with the remote device.

For example, the DOS operating system for personal computers requires that a CR-LF (carriage return-line feed) be used to terminate a line of text. Other systems such as Unix require a line of text to be terminated with an LF. Another example is the echoing of characters. Upon connection attempt, the TELNET protocol will negotiate with the remote device as to who will do the echoing of typed characters to the initiator of a connection. During the connection attempt between a source and destination station,

the two stations will communicate options. These options indicate how each end of the connections will respond on the TELNET connection. These options include:

1. The ability to change from 7-bit text to 8-bit binary
2. Allowing one side or the other to echo characters
3. Specifying a terminal type
4. Requesting the status of a TELNET option from the remote connection
5. Setting a timing mark to synchronize two ends of a connection
6. The ability to terminate a record with an EOR code
7. Setting line mode so that strings of characters may be sent instead of a Character-at-a-time transmit
8. Stopping the go-ahead signal after data

The options are negotiated between the two network stations in the following manner:

Request Response
WILL <option> DO or DON'T <option>
For example, WILL ECHO from station A is requesting that station A provide the echoing of characters. The response will either be DO ECHO, meaning the remote end agrees, or DON'T ECHO, meaning the remote end will not allow station A to echo. Agreement between the two TELNET ends communicated for a DO <option> will be responded to with a WILL <option> or WON'T <option>. Either side of the connection can provide the command or the response. One side provides services in exactly the same manner as the other side.

14.7. E-MAIL

Today it is well known as Electronic Mail, or email. RFCs 821, 822, 974 are handling this protocol. Email still cannot transport packages and other items. Email is very fast and guarantees delivery. Three protocols are used for today's email:

- **SMTP—operates over TCP**
- **POP—operates over TCP**
- **DNS—operates over UDP**
- SMTP allows for the sending/receiving of email.
- POP allows us to intermittently retrieve email.
- DNS makes it simple.

RFC 822 defines the structure for the message, and RFC 821 specifies the protocol that is used to exchange the mail

between two network stations. It truly is amazing how old the original mail protocol is, and it is still in use today. So we have email to send to one another, completely bypassing the postal system.

There are some who call the postal system “snail mail.” Many people today still immensely enjoy receiving a handwritten letter from a family member, friend, or a business correspondence through the postal system.

Email does have many, many advantages and one of the top advantages is speed. The biggest disadvantage is lack of emotion. Like everything else, email has its place, but it is not 100 percent of the pastry; it is merely another form of communication. In order to send and receive mail between users, there are actually two protocols (possibly three) that are used:

SMTP: Used for the actual transport of mail between two entities (mail servers).

POP (Post Office Protocol): A protocol that allows single users to collect their mail on one server.

DNS: Used to identify the mail hosts for a domain or hostname.

Mail can be sent and received using only SMTP, but the other protocol involvement makes it much easier to use and is more efficient. This is a protocol that allows users to transmit messages (Mail) between other users. It is one of the most widely used applications of the TCP/IP protocol.

Simple Mail Transfer Protocol (SMTP)

Functions of SMTP:

The protocol is relatively simple. A message will be created, properly addressed, and sent from a local application to the SMTP application, which will store the message. The server will then check (at periodic intervals) to see if there are any messages to deliver. If there are, the mail server will try to deliver the message. If the intended recipient is not available at the time of delivery, the mail server will try again later. The mail server will try a few times to deliver the message and, if it cannot, will either delete the message or return it to the sender. The address has the general format of local-part @ domain-name. By this, you should recognize the domain name format. For example, an address at the SMTP header could be **matt@engineering.naugle.com**. This would indicate that the message is addressed to a user named Matt in the domain of engineering.naugle.com. When DNS is used to look up the mail handler for Matt, it will have some sort of entry like: **engineering.naugle.co IN MX 10**

NT1mail_server.engineering.naugle.com. From this, the name will be looked up and the mail will be delivered to that host.

There are two entities to this system, the ***sender*** SMTP and the ***receiver*** SMTP that are used to transport mail between two systems. The sender SMTP will establish communications with a receiver SMTP. Attachments are allowed with Internet email but not directly with the protocol used in SMTP (send mail protocol).

Email applications convert using a variety of protocols like MIME (Multipurpose Internet Mail Extensions). SMTP (or more specific, send mail) can only handle text. Therefore, most email applications convert an attachment to text before sending. A common type is MIME. At the receiver, the email application converts the attachment back to its original format.

SMTP Flow:

The SMTP design is based on the following model of communication:

Once you have filled out the header and body section of your mail message, the sender SMTP establishes two-way communication to a receiver SMTP. The receiver SMTP may be either the ultimate destination or a transient stop on the way to the final destination. Commands are sent to the receiver by the sender SMTP and SMTP replies are sent from the receiver SMTP to the sender SMTP in response to each of the commands.

Once two-way communication has been established, a series of commands (of which you can see operate using some mail applications) are issued.

The sender SMTP will send a HELLO (HELO) command identifying who it is using its domain name to the receiver.

The receiver acknowledges this with a reply using its domain name. Next, the server issues a MAIL command to the receiver. In this will be the identification of the person (place, or thing) sending the mail. The receiver acknowledges this with an OK.

The sender SMTP then sends a RCPT command to the receiver, using the intended receiver name as an argument. Each recipient in the list is sent to the receiver one at a time, and each time the receiver acknowledges with an OK for those recipients that it knows about. For those that it does not know about (different domain name), it will send back a different reply that it is forwarding the message on.

For any intended recipients received from the SMTP sender for which it has no account, the receiver will reply to the sender that no such user(s) exists.

After the intended recipients have been ACK'd or NACK'd, the SMTP sender sends the DATA command and the SMTP receiver will OK this and indicate what the end of message identifier should be.

Once this is received (the ending identifier), the SMTP receiver will reply with an OK. Notice that all data is received, the ending identifier is received, and then a reply message is sent by the receiver.

If everything went okay, the sender ends the connection with a QUIT command. The SMTP receiver will reply indicating that the communication channel is closed. So the minimum commands that a receiver must support are

- HELO,
- MAIL,
- RCPT,
- DATA,
- RSET,
- NOOP, and QUIT.

Depending on the mail program that you use, the transaction between a recipient and sender of mail has been the same since RFC 821 was written. The interface allows you to complete the mail message, filling in the header (addresses and subject) and the body (text) of the letter. When you press the Send button, the following transaction takes place. Some mail programs actually place the mail commands and state numbers on display while the transaction is taking place. It should be noted here that sending mail is immediate. It may get queued for a small length of time on different routers, and transient mail servers, but not for long. This is for the transport of mail.

Most electronic mail today is sent via SMTP and will reside on your mail server host until you retrieve it using POP. Today, retrieving your mail does not mean that you have to run the SMTP protocol. A server host will accept mail messages directed to you on your behalf. Then you can sign on any time you want and retrieve your mail.

DNS Interaction for Mail:

A record known as the MX record in DNS identifies a mail exchanger for the purpose of identifying hosts for recipients. A mail exchanger is hosts that will either process or forward mail for the domain name. Processing means that the host will deliver it the

host to which it is addressed or hand it off to another transport, such as UUCP or BITNET.

Forwarding means that the host will forward the message on to the final destination or to another mail exchanger closer to the destination. There can be multiple entries for a mail exchanger in a DNS. Each MX entry will have a precedence number beside it and this signals the sender which mail host it should try first. If the precedence value is equal among MX records, then the sender will randomly pick one from the list. Once the mail sender has successfully delivered the mail to one of the MX hosts, its job is done. It is the job of the MX host to make sure it is forwarded on to its final destination. If there are no MX records for a domain name, it is up to the mailer application as to what happens next. Some will try to deliver it to the IP address of the mail destination.

Post Office Protocol (POP)

The original mail program RFC 821 (which is the one in use today) was set up to send messages directly to a user logged in to a terminal, as well as store these messages to a mailbox. The commands allowed for the receiver to determine if the user was logged on to a terminal (not a PC), if they were accepting messages, and if they were not, are there a mailbox to deliver some mail to. There were no message attachments and messages were sent and received in 7-bit ASCII (8th bit was set to 0); therefore, this would not allow for binary messages to be sent (i.e., no attachments). In fact, the original message was not to exceed 1000 characters (however, implementations that could go beyond this barrier were strongly encouraged to do so). So, to operate mail, the host must be operational (able to receive) all the time.

Today, terminals do exist, but more commonly, personal computers have taken their place. Therefore, the final recipient will be the personal computer. The personal computer will have both SMTP and POP. Even though a personal computer will retrieve its mail via POP, it will still use the SMTP functions to send its mail. Since SMTP expects to be able to deliver mail immediately, this would mean that all users would have to have their personal computers on 100 percent of the time in order to accept mail.

Second, to receive and read your mail, you must log on to a specific host. To operate a mail server generally requires that the mail server is available for a majority of the time, has the ability to store many mail messages, and is able to fully run SMTP and accept mail from an SMTP sender. While this may have been feasible for situations like terminal-to-host connectivity, it is not feasible for situations that we have today; namely, personal computers and mobile workers. SMTP is a very robust transaction-oriented protocol and requires the statements previously discussed to operate fully.

SMTP is set up to send and receive mail by hosts that are up full time. No rules for those hosts that is intermittent on the LAN. POP emulates you as a host on the network. It receives SMTP mail for you to retrieve later. POP accounts are set up for you by an ISP or your company.POP retrieves your mail and downloads it to your personal computer when you sign on to your POP account. What we need is the ability for SMTP to operate (drop off the mail, like a PO Box at the post office), and then another protocol to download to our personal computers (we drop by the post office and retrieve our mail from the post office box.). POP is the protocol to allow for this. Mail can be delivered to a drop-off point and POP allows us to log in and retrieve our mail.

When you sign up with an Internet Service Provider, a POP account is assigned to you; for example, mnaugle@POP3.ISP1.com. You use this when configuring your mail program. Also, when sending mail you must give the SMTP server name to the configuration program as well. The protocol of POP3 is not used for sending mail.

Operations of POP:

POP3 should be viewed only to ***retrieve*** your mail from the mail drop-off point. ***Sending*** mail is a different story. Your personal computer still has the ability to establish a TCP connection to a relay host (intermediate mail host) to send mail. Therefore, you should consider your host as having the ability to be an SMTP sender, and the SMTP protocol explained earlier applies. However, to retrieve your mail, POP3 comes into play. The client (your PC with a mail application such as Eudora), once established with TCP/IP on its network, builds a connection to the POP3 server. The POP3 server configuration is built during the installation of your mail program on the PC. The connection between your PC and the POP3 server is a TCP connection on TCP port 110. Similar to SMTP, once the connection is established, the server will respond with a greeting like **"POP3 server ready."** The POP3 protocol then enters the authentication state. During this phase, you must identify yourself with a username and password. The RFC does not indicate which authentication mechanism you should use. The most common is the simple username/password combination. However, other options are available, such as Kerberos and APOP.

Once you have been "authenticated," the POP3 server puts an exclusive lock on your mailbox, ensuring that no other transactions take place on the messages while you are retrieving your mail. The server now enters the transaction state in which each of the messages in your mailbox is assigned a number. This allows your client POP to indicate how many messages are in your mailbox.

Each message can be retrieved one at a time or all can be retrieved. Furthermore, you can instruct your client POP to delete messages as they are retrieved. This can be good and bad. It would be nice to hold on to your messages as a backup on the server, but this requires disk space that can be depleted quickly.

From here, you retrieve your messages and, depending on how you configured you PC mail program, the messages are marked for deletion after your session. After you retrieve your messages, your mail program will send the QUIT command, which closes the POP3 session down. Then the UPDATE process begins on the server, which is housekeeping work on the server (deleting messages, etc.).

From here, you can read your messages locally on your PC. You are now disconnected from the POP3 server and you can manipulate the messages locally. There are many other options available for POP3 which may or may not be implemented; however, from a users' point of view, they are not noticed.

SMTP, DNS, and POP Topology:

For example, mail is sent from your PC to J's PC. In order to accomplish this, the send mail (SMTP) program is established to the SMTP server on your ISP. A DNS lookup is accomplished using the root DNS server to find the domain of the intended recipient. A call is made to recipients DNS to find the mail server (which could be the same server as the DNS). Once the mail server is found (its IP address is found), mail is sent to that server. The POP function delivers it to your mail box on that mail server so that when J's PC retrieves mail; your mail message will be waiting.

15

THE APPLICATION LAYER AND ITS SECURITY

In this Unit, we are going to discuss in detail about the following options available to secure the application layer.

Unit Structure

14.8 Cryptography

14.9 Symmetric key and asymmetric key cryptography,

14.10 DES algorithm,

14.11 RSA algorithm,

14.12 Security services

14.13 Message and entity.

15.1. CRYPTOGRAPHY

History of Cryptography:

- It is existed long before the ubiquity of computers.
- Julius Caesar (100-44 BC) used a simple substitution with the normal alphabet (just shifting the letters a fixed amount) in government communications".
- Cryptography, the science of encrypting and decrypting information, dates as far back as 1900 BC.
- Thomas Jefferson, invented a wheel cipher in the 1790's,
- It was used extensively during both the world wars.
- Cipher machines were created to encrypt messages by the Nazis called by the allies as Enigma.
- They were used by bootleggers in the 1930s for liquor smuggling.
- In 1970s, Dr. Horst Feistel established the precursor to today's Data Encryption Standard (DES) with his 'family' of ciphers, while working at IBM's Watson Research Laboratory.
- Also in 1976, two contemporaries of Feistel, Whitfield Diffie and Martin Hellman first introduced the idea of public key cryptography.

- 1977, Rivest, Shamir and Adleman introduced to the world their RSA cipher, applicable to public key cryptography and digital signatures.
- Zimmerman released his first version of Pretty Good Privacy (PGP) in 1991 as a freeware product, which uses the IDEA algorithm.

Need of cryptography/Reason for its existence:

(1) Secrecy: Only intended receiver understands the message.

(2) Authentication: sender and receiver need to confirm each other's identity.

(3) Message Integrity: Ensure that their communication has not been altered, either maliciously or by accident during transmission.

Terms frequently used in Cryptography:

Cryptology: Originated for the Greek *kryptós logos*, meaning ``hidden word''.

Plaintext: The message to be encrypted.

Key: It is the object used to encrypt the plaintext.

Ciphertext: It is the encrypted text.

Encryption: The process of converting plaintext into Ciphertext using an appropriate key.

Decryption: The process of converting Ciphertext into plaintext using a appropriate key.

Cryptography: IT is the art or science of keeping communication classified.

Cryptographers: People who indulge in cryptography are known as cryptographers.

Cryptanalysis: The art or science of decrypting a Ciphertext without knowing the authorized key is known as cryptanalysis.

Cryptanalysts: People who indulge in cryptanalysis. Could be ethical or fraudsters.

Cipher: The method of decryption and encryption is generally known as cipher.

PURPOSE AND APPLICATION OF CRYPTOGRAPHY:

a) Secure Communication:

Secure communication is the most straightforward use of cryptography. To prevent eavesdropping

- War time communication.
- Business transactions.

b) Identification and Authentication:

Identification and authentication are two widely used applications of cryptography. Identification is the process of verifying someone's or something's identity.

c) Secret Sharing:

Another application of cryptography, is secret sharing, allows the trust of a secret to be distributed among a group of people.

d) Electronic Commerce:

Over the past few years there has been a growing amount of business conducted over the Internet - this form of business is called electronic commerce or e-commerce.

E.g.: Encrypted Credit Card Number

e) Certification:

Certification is a scheme by which trusted agents such as certifying authorities guarantee for unknown agents, such as users. The trusted agents issue vouchers called certificates which each have some inherent meaning. It was developed to make identification and authentication possible on a large scale

f) Key Recovery:

Key recovery is a technology that allows a key to be revealed under certain circumstances without the owner of the key revealing it.

E.g.: accidental erasure, fraudster, terrorism.

g) Remote Access:

Secure remote access is another important application of cryptography. The basic system of passwords certainly gives a level of security for secure access, but it may not be enough in some cases.

h) Cell Phones:

Used in cellular phones as a means of authentication; especially if phone is lost. This prevents people from stealing cellular phone numbers and access codes. Or to prevent eavesdropping of phone calls using voice encryption.

i) Access Control:

Cryptography is also used to regulate access to satellite and cable TV. Cable TV is set up so people can watch only the channels they pay for.

Principles of Cryptography:

(1) Symmetric (secret key):

- An identical key is used for encryption and decryption
- Strength of algorithm is determined by the size of the key, longer the key more difficult it is to crack.
- Key length is expressed in bits.
- Typical key sizes vary between 48bits and 448 bits.
- Set of possible keys for a cipher is called key space.
- For 40-bit key there are 240 possible keys.
- For 128-bit key there are 2128 possible keys.
- Each additional bit added to the key length doubles the security.
- To crack the key the hacker has to use brute-force (try all the possible keys till a key works is found).
- Super Computer can crack a 56-bit key in 24 hours.
- It will take 272 times longer to crack a 128-bit key (Longer than the age of the universe).

Primitive Ciphers:

Caesar Cipher is a method in which each letter shifted in the plaintext n places. Mono-alphabetic Cipher is a method in which any letter can be substituted for any other letter

Advantages of primitive ciphers:

- Relatively simple and significantly faster than the rest.
- Used in an environment where single authority manages the keys.
- Used in environments where secure secret key distribution can take place

Disadvantages:

- Key management (generation, transmission and storage of keys) may be a problem.
- People could repudiate sent messages claiming the receiver had compromised the key
- Third party involvement may be required for authentication of key. Database with keys of all

Types of Symmetric Ciphers:

(1) Stream cipher:

- Each bit or byte is encrypted or decrypted individually
- Simple substitution ciphers
- Used for a single message

(2) Block cipher:

A block cipher is a type of symmetric-key encryption algorithm that transforms a fixed-length block of plaintext data into a block of cipher text data of the same length

- Encrypt data one bit or one byte at a time
- Used if data is a constant stream of information
- Iterated block cipher is when ciphering is repeatedly done

(2) Asymmetric (public key):

In asymmetric, different keys (public and private) for encryption and decryption used. Encryption is done by public key and sent across. Decryption is done by private keys:

- A sends an encrypted message using a public key
- B receives the message and decrypts it using a private key.
- Anybody can send B a message but only B who knows his private key can open the message.

Public-key cryptography is not meant to replace secret-key cryptography, but rather to supplement it, to make it more secure

Advantages:

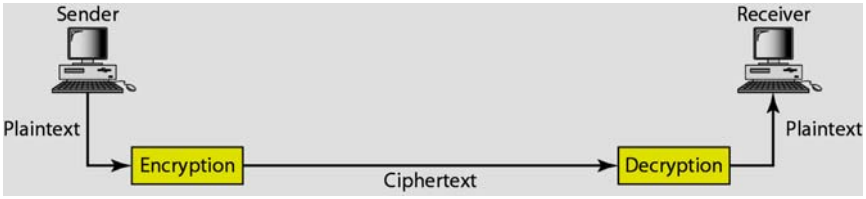
- The need for the sender and receiver to share keys is eliminated, no private key is ever shared
- Is the fundamental of digital signatures that cannot be repudiated.

Disadvantages:

- A disadvantage of using public-key cryptography for encryption is speed.
- It is problematic to get the key pair generated for the encryption.
- Vulnerable to man-in-the-middle attack.

15.2 INTRODUCTION TO BASIC ENCRYPTION AND DECRYPTION:

The term 'Cryptography' means the concept of encryption and decryption together. Cryptography is the technique in which the original 'plain text' message is 'encrypted' i.e. converted into a coded form called 'cipher text' at the sender's end, which is then transmitted to the receiver. The receiver then 'decrypts' i.e. converts the 'cipher text' back into the 'plain text' to get the original message back.



Cryptography is also called as an art or technique to achieve secure communication between the communicating parties by encoding the messages between them such that no third party can gain anything useful out of interception.

Various techniques are utilized for this purpose of cryptography. Broadly these techniques fall into two categories.

(1) Symmetric key cryptography: In which the 'key' element used, is the 'same' for both encryption as well as decryption and

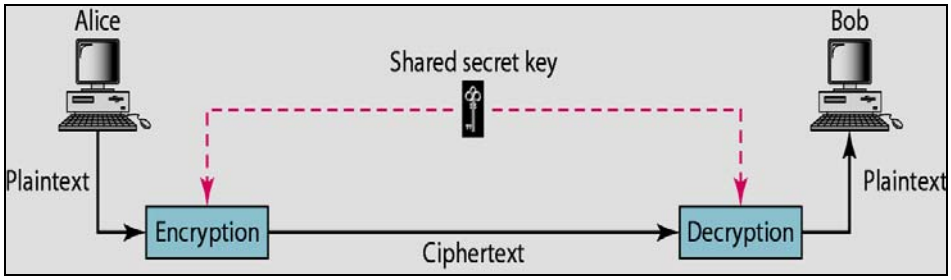
(2) Asymmetric key cryptography: In which the 'key' element used, is different for both encryption as well as decryption.

(a) Symmetric key cryptography is also known as 'private or secret key cryptography'; Whereas Asymmetric key cryptography is also known as 'public key cryptography'.

SYMMETRIC- KEY CRYPTOGRAPHY

We can divide all the cryptography algorithms in the world into two groups: symmetric-key (sometimes called secret-key) cryptography algorithms and public-key (sometimes called asymmetric) cryptography algorithms.

In symmetric-key cryptography, the same key is used by both parties. The sender uses this key and an encryption algorithm to encrypt data; the receiver uses the same key and the corresponding decryption algorithm to decrypt the data



In symmetric-key cryptography, the same key is used by the sender (for encryption) and the receiver (for decryption). The key is shared.

In symmetric-key cryptography, the algorithm used for decryption is the inverse of the algorithm used for encryption. This means that if the encryption algorithm uses a combination of addition and multiplication, the decryption algorithm uses a combination of division and subtraction.

Note that the symmetric-key cryptography algorithms are so named because the same key can be used in both directions.

In symmetric-key cryptography, the same key is used in both directions.

Symmetric-key algorithms are efficient; it takes less time to encrypt a message using a symmetric-key algorithm than it takes to encrypt using a public-key algorithm. The reason is that the key is usually smaller. For this reason, symmetric-key algorithms are used to encrypt and decrypt long messages.

Symmetric-key Cryptography is often used for Long Messages:

Disadvantages of symmetric key:

A symmetric-key algorithm has two major disadvantages.

(1) Each pair of users must have a unique symmetric key.

This means that if N people in the world want to use this method, there needs to be N(N - 1)/2 symmetric keys.

For example, for 1 thousand people to communicate, 1000 * 999 /2 = 4, 99, 500 (4 lakhs 99 thousand and five hundred symmetric keys are needed. The distribution of the keys between two parties can be difficult.

(2) The sender needs to exchange the key to the receiver. It may be hijacked in between!

15.3. DES (DATA ENCRYPTION STANDARD) CIPHER ALGORITHM

DES CIPHER:

It is a 16-round Feistel cipher with block size of 64 bits. DES stands for Data Encryption Standard. IBM developed DES in 1974 in response to a federal government public invitation for data encryption algorithms. In 1977, DES was published as a federal standard, FIPS PUB 46.

Algorithm:

Step 1: 64 bit plain text blocks is handed over to the initial permutation (IP) function.

Step 2: IP is performed on the plain text.

Step 3: IP produces 2 halves; say LPT and RPT, both of 32 bit each.

Step 4: Performs 16 rounds of encryption process each with its own key.

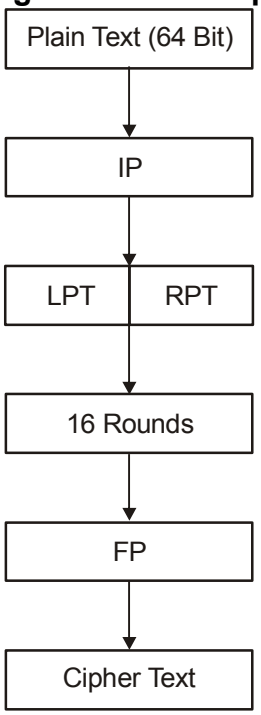
Rounds are defined as follows in the algorithm:

4a: Key transformation
4b: Expansion Permutation (EP)
4c: S-Box Substitution
4d: P-Box Permutation
4e: XOR and Swap.

Step 5: LPT and RPT are rejoined finally and a Final Permutation (FP) is performed on the combined block.

Step 6: The result of this process produces 64-bit cipher text.

Diagrammatical Representation:



Explanation of the Algorithm:

IP – Initial Permutation:

Comparing the IP table performs IP. It happens only once, and it happens before the first round. It suggests how the transposition in IP should proceed, as shown in the IP table.

After this IP, 64 bit plain text is divided into 2 halves normal LPT and RPT, 08 32 bits each.

Rounds:

In the rounds, **step 1 is key transformation.**

That is achieved by:

(a) Shifting the key position by considering the Round Table.

(b) Compare the Compression Table to get the sub key of 48 bits.

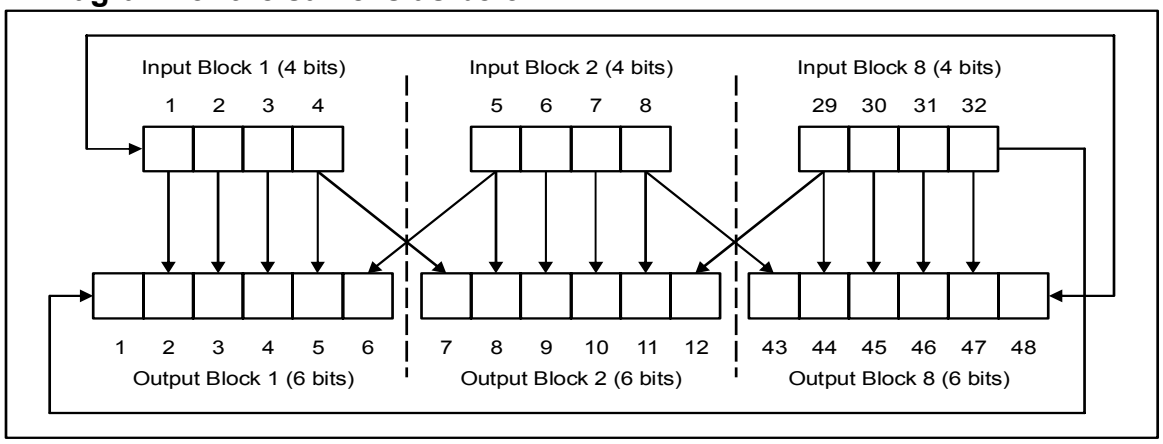
Step 2: is Expansion Permutation (EP):

In this step, the 32-bit RPT is expanded to 48 bits as it of key length. The process is shown as under:

The 32-bit text is divided into 8 blocks of 4 bits each.

Then by adding 2 bits extra that is the first bit of the block 1 is the last bit of the block 8 and the last bit of the block 8 is the first bit of the 7th block the 48-bit text is obtained.

Diagram for the same is as below:



After this expansion it will be compared with the Expansion Permutation Table.

Step 3: in Round is S-Box Substitution:

(1) This step reduces 48 bits RPT into 32 bits because LPT is of 32 bits.

(2) It accepts 48 bits, does some XOR logic and gives 32 bits.

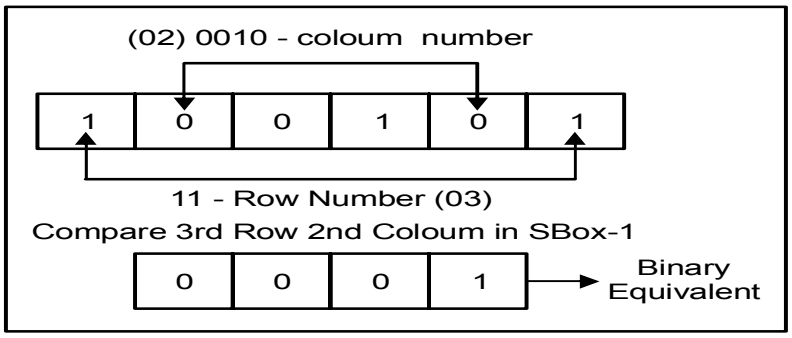
(a) The 48 bits key (Result of Step 1) and the 48 bits of RPT (Result of Step 2) will be XOR and the output will be 48 bits Input block and that will be given as the input for the S-Box Substitution.

(b) The 48-bit block text will be divided into 8 blocks of 6 bits each.

(c) Decimal equivalent of the first and last bit in a block denotes the row number and decimal equivalent of the bit 2, 3, 4 and 5 denotes the column number of the S-Box Substitution table.

(d) Check the value and take the binary equivalent of the number.

(e) The result is 4-bit binary number.



(f) For example if the 6-bit number is 100101 then the first and last bit is 11 and the decimal equivalent of the number is 3. The remaining bits are 0010 and the decimal equivalent of the number is 2. If it is the first block of input, then check the 3rd row 2nd column value in the Sbox-1 substitution table. It is given as 1 in the table. Binary equivalent of 1 is 0001.

(g) The input 100101 of 6-bit is now reduced to 0001 after S-Box Substitution.

Step 4: in Round is P-Box Permutation:

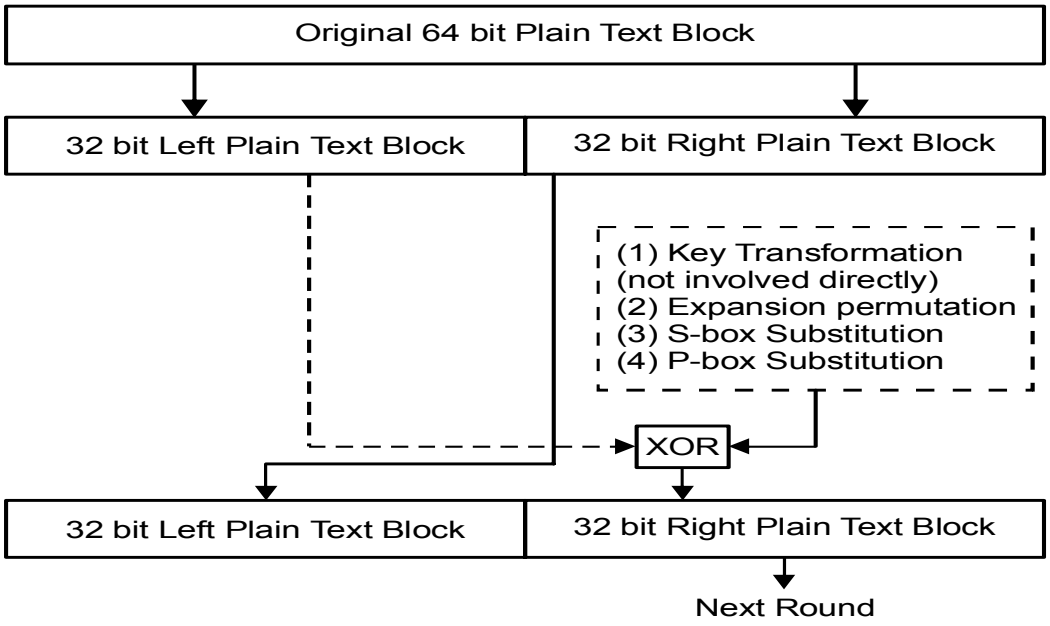
In this step, the output of S-Box, that is 32 bits are permuted using a p-box. This mechanism involves simple permutation that is replacement of each bit with another bit as specified in the p-Box table, without any expansion or compression. This is called as P-Box Permutation. The P-Box is shown below.

16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

For example, a 16 in the first block indicates that the bit at position 16 moves to bit at position 1 in the output.

Step 5: is XOR and Swap:

The untouched LPT, which is of 32 bits, is XORed with the resultant RPT that is with the output produced by P-Box permutation. The result of this XOR operation becomes the new right half. The old right half becomes the new left half in the process of swapping. This is shown below.



Final Permutation (FP):

At the end of 16 rounds, the Final Permutation is performed only once. This is a simple transposition based on the Final Permutation Table. The output of the Final permutation is the 64-bit encrypted block.

KEY DISTRIBUTION:

Every process of encryption and decryption is necessarily associated with a **'key'**- the combination used for encryption and/or decryption, and an algorithm i.e. the rules or steps used for both encryption and decryption. The requirement of 'same' key as in case of 'symmetric' key cryptography leads to a common problem called 'problem of key distribution', i.e. how the two parties should agree upon a 'common' key that has to be used for the process. This is as described below.

Problem of Key Distribution in Symmetric Key Cryptography:

As in case of symmetric key cryptography, the key that has to be used for both encryption and decryption should be the 'same' this leads to a problem that how the two parties requiring secure communication can 'agree' or 'decide' upon a common key, without letting any third person know about it? There can be many ways in which the two parties will try to communicate assuming it is secure, but it may not be so. e.g. even if they exchange letters, seal envelopes into locked boxes, talk over open media for the common key, or send the key along with the locked boxes, whatever may be the means used, it turns out to be practically non-viable or difficult to implement.

That is to say, there are very much chances of intercepting the communication between two parties if any of these methods are used. This is called the 'problem of key distribution'.

In order to come out of this problem, one good solution was given by two scientists jointly known as 'Diffie-Hellman key exchange algorithm'.

ASYMMETRIC KEY CRYPTOGRAPHY:

THE CONCEPT OF PUBLIC KEY AND PRIVATE KEY:

The Asymmetric key cryptography is also known as a 'public key cryptography', which uses a key-pair rather than a single key. The importance of this scheme is that only one key-pair is required to securely communicate between any numbers of other parties. (Unlike the huge number of keys that we've seen with earlier method.) Hence, one problem is overcome right away. One of these two keys is called public key (which can be announced to the world) and another is private key (obviously to be kept with oneself). This is to be followed by everyone who wants to communicate securely.

The working of public and private keys:

Asymmetric key cryptography (using public and private keys) works as under:

Consider the scenario, X wants to send a message to Y, without having to worry about its security.

(1) Then X and Y should each have a private key and a public key.

- **(a) X should keep its private key secret.**
- **(b) Y should keep its private key secret**.
- **(c)** X should inform Y about its public key.
- **(d)** Y should inform X about its public key

(Both now have their own set of keys ready.)

(2) When X wants to send message to Y, X encrypts with Y's public key (as it is known to everyone)

(3) X then sends this message to Y.

(4) Then, Y decrypts this message using his own private key (known only to Y)

[This ensures in this case, that the message can be encrypted & sent by anyone, but can only be decrypted by Y. Hence, any interception will not result in knowing the sensitive information as key is only with Y.]

Similarly, on the other side, if Y wants to send the message to X, reverse method is performed.

(5) Y encrypts the message using X's public key and sends this to X.

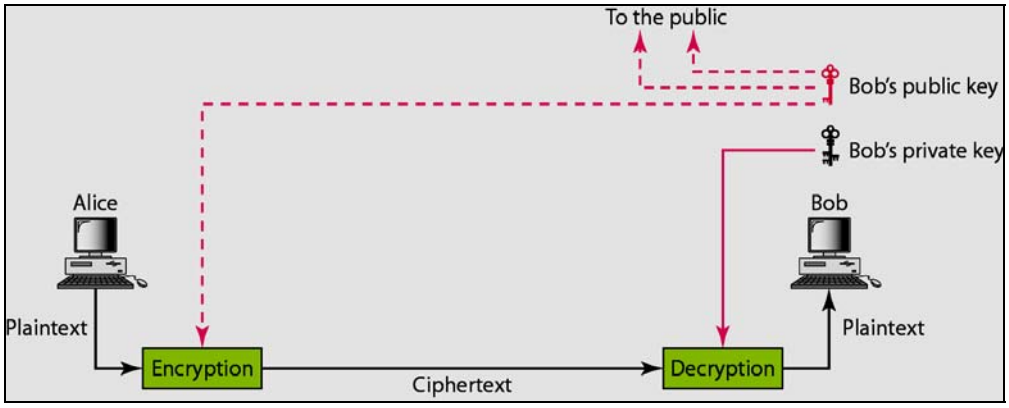
(6) On receiving the message, X can further decrypt it using his own private key.

The basis of this working lies in the assumption of large prime number with only two factors. If one of the factors is used for encryption process, only the other factor shall be used for decryption.

The **best example** of an asymmetric key cryptography algorithm is the famous **RSA algorithm** (developed by **Rivest, Shamir and Adleman** at MIT in 1978, based on the framework setup by Diffie & Hellman earlier).

In public-key cryptography, there are two keys: a private key and a public key. The receiver keeps the private key. The public key is announced to the public.

Imagine Alice, as shown in Figure wants to send a message to Bob. Alice uses the public key to encrypt the message. When Bob receives the message, the private key is used to decrypt the message.



In public-key encryption/decryption, the public key that is used for encryption is different from the private key that is used for decryption.

The public key is available to the public; the private key is available only to an individual.

Public-Key Encryption/Decryption has Two Advantages:

First, it removes the restriction of a shared symmetric key between two entities (e.g., persons) that need to communicate with each other. A shared symmetric key is shared by the two parties and cannot be used when one of them wants to communicate with a third party. In public-key encryption! Decryption, each entity creates a pair of keys; the private one is kept, and the public one is distributed. Each entity is independent, and the pair of keys created can be used to communicate with any other entity.

The second advantage is that the number of keys needed is reduced tremendously.

In this system, for I thousand users to communicate, only 1 thousand pairs of keys i.e. 2000 keys are needed, not 4,99,500, as was the case in symmetric-key cryptography.

Public-Key Cryptography also has Two Disadvantages:

The biggest disadvantage is the **complexity of the algorithm**. If we want the method to be effective, the algorithm needs large numbers. Calculating the cipher text from plaintext using the long keys takes a lot of time. That is the main reason that public-key cryptography is not recommended for large amounts of text. **Public-Key Algorithms are more efficient for Short Messages.**

The second disadvantage of the public-key method is that the association between an entity and its public key must be verified. If Alice sends her public key via an email to Bob, then Bob must be sure that the public key really belongs to Alice and nobody else. One point needs to re-mention that if your private key were made public you would Get Bankrupted in no time!

15.4. RSA ALGORITHM:

The algorithm to show the working of asymmetric key cryptography is as follows:

(1) Generate two large random primes, *p* and *q*, of approximately equal size

(2) Calculate N = PXQ

(3) Select the public key that is the encryption key E such that it is not a factor of (p-1) (q-1).

(4) Select the private key that is the decryption key D such that the following equation is true: **(DXE) mod (P-1) X (Q-1)=1**

(5) For encryption, calculate the cipher text CT as $CT=PT^E$ mod N.

(6) Send CT as the cipher text to the receiver.

(7) For decryption, calculate the plain text PT as $PT=CT^D$ mod N.

A Very Simple Example of RSA Encryption:

This is an extremely simple example using numbers you can work out on a pocket calculator (those of you over the age of 35 can probably even do it by hand).

(1) Select primes p=11, q=3.

(2) n = pq = 11.3 = 33 phi = (p-1)(q-1) = 10.2 = 20

(3) Choose e=3 Check gcd(e, p-1) = gcd(3, 10) = 1 (i.e. 3 and 10 have no common factors except 1), and check gcd(e, q-1) = gcd(3, 2) = 1therefore gcd(e, phi) = gcd(e, (p-1)(q-1)) = gcd(3, 20) = 1

(4) Compute d such that ed = 1 (mod phi) i.e. compute d = e^{-1} mod phi = 3^{-1} mod 20 i.e. find a value for d such that phi divides (ed-1) i.e. find d such that 20 divides 3d-1. Simple testing (d = 1, 2,...) gives d = 7 Check: ed-1 = 3.7 - 1 = 20, which is divisible by phi.

(5) Public key = (n, e) = (33, 3) Private key = (n, d) = (33, 7).

This is actually the smallest possible value for the modulus n for which the RSA algorithm works.

Now say we want to encrypt the message m = 7, c = m^e mod n = 7^3 mod 33 = 343 mod 33 = 13. Hence the cipher text c = 13.

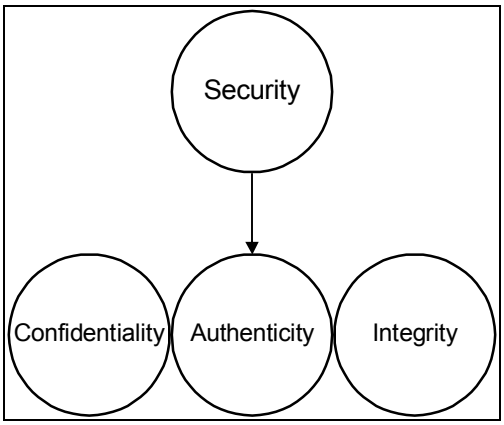
To check decryption we compute m' = c^d mod n = 13^7 mod 33 = 7. Note that we don't have to calculate the full value of 13 to the power 7 here. We can make use of the fact that a = bc mod n = (b mod n).(c mod n) mod n so we can break down a potentially large number into its components and combine the results of easier, smaller calculations to calculate the final value.

One-way of calculating m' is as follows: m' = 13^7 mod 33 = $13^{(3+3+1)}$ mod 33 = $13^3.13^3.13$ mod 33 = (13^3 mod 33). (13^3 mod 33).(13 mod 33) mod 33 = (2197 mod 33).(2197 mod 33).(13 mod 33) mod 33 = 19.19.13 mod 33 = 4693 mod 33 = 7.

15.5. PRINCIPLES OF SECURITY/SECURITY SERVICES:

Generally the security issues can be classified into the following categories.

(1) Confidentiality
(2) Authenticity
(3) Availability
(4) Auditability
(5) Access Control
(6) Integrity
(7) Non-repudiability



These three are the basic levels of issues, which will lead to the further loopholes.

Confidentiality:

Basically the threats can be in the area of network or in the area of application. Mainly the insiders who are having full access with the computer system create the application levels of threats. This can be easily detected can be avoided by using suitable mechanisms. Even though the application level threats are easy to detect, this is also creating high level of problems to the system.

The network levels of attacks are dangerous because of the major business transactions that are happening through the Internet. This is maintaining the actual secrecy of the message. The message that is traveling through the network should not be opened by any of the third parties who are not related with the transaction.

Nowadays majority of the bank transactions are happening through the network. So that confidentiality issue is creating lot of problems related with network, software and data.

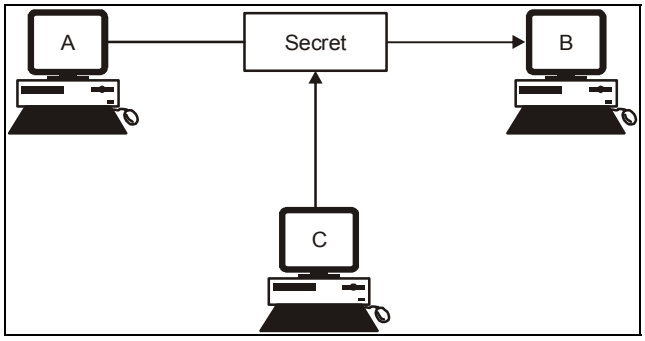
The confidentiality related issues could be belonging to any one of the following categories:

(1) IP Spoofing
(2) Packet sniffing
(3) Alteration of message
(4) Modification of message
(5) Man-in-middle
(6) Brute force attack
(7) Password cracking

By using any of the above-mentioned methods an user can enter into others message and can create problem related to the secrecy of the message. The intension of the user may be viewing the message without making many changes to the hacked message.

Normally in banking system the people used to receive their bank balance-using network. In this kind of situation any hacker who knows the IP address can get these information about others account balance.

In some situation people may hack the message and forward it to some unknown person, which will create confusion between the original sender and the receiver. This will also lead to the problem related with integrity.



Loss of Confidentiality

Authenticity:

This can be defined as an identity for a user to assure that the message is coming from the right person. This is also an another important issue along with confidentiality which may lead to the further security threats. This can be assured by any of the following factors:

(1) Something you have (like tokens, credit card, passport etc).

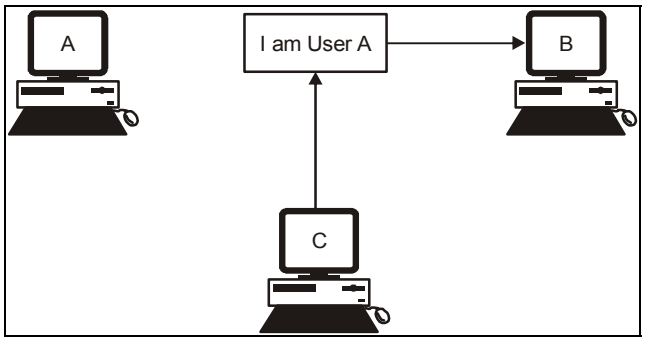
(2) Something you know (like PIN numbers, account number etc).

(3) Something you are (like fingerprints, signatures etc).

Generally with the computer system passwords are the very simple authentication mechanism, which help the system to authenticate a particular person. The people can use one-time passwords and key technology to assure authenticity during message transaction.

Various issues related to authenticity includes.

(1) Stealing password
(2) Fake login screen
(3) Information leakage Etc.



Absence of Authenticity

Fabrication is possible in the absence of proper authentication mechanisms.

Availability:

This can be defined as keeping the right information or resources available to the right person at the right time. This can happen either with the data or with the hardware resources. This will stop the person from accessing various resources by flooding the network.

There is a complexity with the availability of the resources and data. Because it can be identified as a issue only when the following conditions are existing in the system.

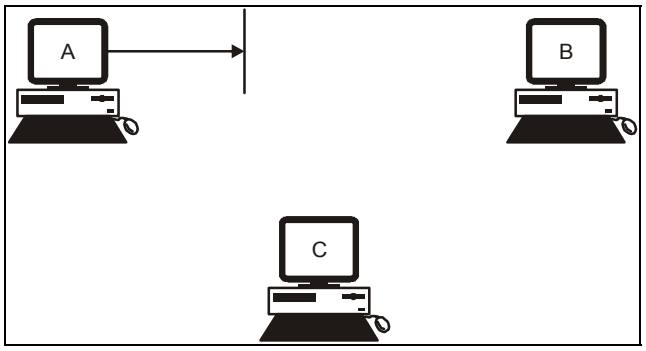
(1) The resources are completely available up to the users expectation.

(2) The content is present in a usable format.

(3) The access rights are used in a proper way.

The actual problem related with availability can be identified only when the above-mentioned things are assured before the problem identification. This is a very serious issue, which will totally stop the process and may lead the user to an idle condition without allowing him to proceed with the further process.

The main problem related with the availability is **DOS attack and DDOS attacks.** Flooding the network path by sending continuous packets, which will create a heavy traffic in the network, does this. This stops the right people accessing right information at the right time.



Attack on Availability

Access Control:

This is also an issue, which is dealing with the hardware resources, software and data. This is helping the operating system to allow access to a particular resource or data only to an authorized person. In this way it is interrelated with the authenticity and availability.

Because the authenticated users will be allotted with certain kinds of rights like **Read, Write, Read/Write, Owner** etc. These rights will be maintained by the operating system in a tabular format or in a linked list format. First the users authenticity has to be verified and then the authentic users rights will be verified against the table.

USERS/FILES	FILE1	FILE2	FILE3
USER1	RW	-	W
USER2	O	RW	W
USER3	-	OR	OW

The attacks related to authenticity and availability can also create the problem related with access control.

Attacks related to access control are as follows:

(1) Intrusion
(2) DDOS
(3) Interference
(4) Inference Etc.

The issues related with authenticity can be resolved by using hash algorithms.

Non-repudiability:

This is another issue, which is related with authenticity and integrity. Repudiability means refusing. This is an issue, which is actually created by the sender who is participating in the transaction. After sending a message a sender can refuse that he was not sending that message. This is done intentionally to create

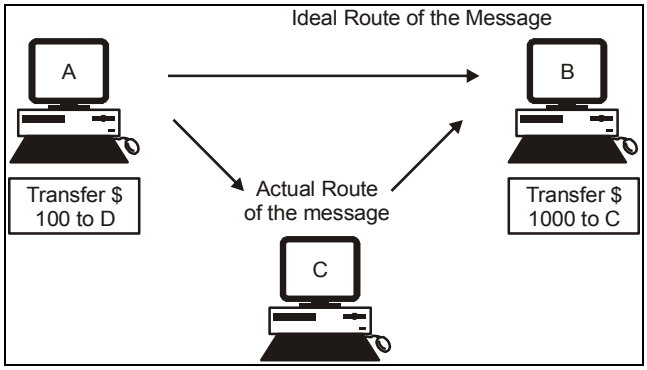
problems at the receiver's side, which creates confusion to the receiver.

This can be done from either side. It may also be from the receiver side. The receiver can deny after receiving the message that he doesn't receive any message. Non-repudiation does not allow the sender of the message to refute the claim of not sending that message.

The non-repudiability related issues can happen in any of the following three ways.

(1) Proof of origin
(2) Proof of receipt
(3) Proof of content

This can be assured by using digital signatures along with the hash algorithms. If the proper authenticity and integrity is achieved then the problems related with non-repudiability can be minimized. All above-mentioned issues are the basic issues of network security. Apart from these various other threats like natural disasters, attacks, software modifications are also creating problems with networks. But majority of the attacks are coming under the basic issues of the network.



Loss of Integrity

Wherever the problems are available accordingly solutions are also present. It is the responsibility of the user to categorize the problem and identify the suitable solution.

Generally the solutions can be categorized as follows.
(a) Security Issues
(b) Security Objectives
(c) Security Techniques

Security Issue	Security Objective	Security Technique
Confidentiality	Privacy of message	Encryption
Authentication	Origin Verification	Digital Signatures, Challenge-response, Passwords, and Biometric devices
Non-repudiation	Proof of origin, receipt and contents	Bi-directional hashing , Digital signatures, Transaction Certificates, Time stamps and Confirmation services
Access controls	Limiting entry to authorized users	Firewalls, Passwords and Biometric devices

15.6. MESSAGE AND ENTITIES OF APPLICATION LAYER

Message

Message Types

HTTP messages consist of requests from client to server and responses from server to client.

HTTP-message = Request | Response ; HTTP/1.1 messages

Request and Response messages use the generic message format of RFC 822 [9] for transferring **entities (the payload of the message**). Both types of message consist of a start-line, zero or more header fields (also known as "headers"), an empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields, and possibly a message-body.

```
generic-message = start-line
                  *(message-header CRLF)
                  CRLF
                  [ message-body ]
start-line      = Request-Line | Status-Line
```

In the interest of robustness, servers SHOULD ignore any empty line(s) received where a Request-Line is expected. In other words, if the server is reading the protocol stream at the beginning of a message and receives a CRLF first, it should ignore the CRLF.

Certain buggy HTTP/1.0 client implementations generate extra CRLF's after a POST request. To restate what is explicitly forbidden by the BNF, an HTTP/1.1 client MUST NOT preface or follow a request with an extra CRLF.

Message Headers

HTTP header fields, which include general-header (section 4.5), request-header (section 5.3), response-header (section 6.2), and entity-header (section 7.1) fields, follow the same generic format as that given in Section 3.1 of RFC 822 [9]. Each header field consists of a name followed by a colon (":") and the field value. Field names are case-insensitive. The field value MAY be preceded by any amount of LWS, though a single SP is preferred. Header fields can be extended over multiple lines by preceding each extra line with at least one SP or HT. Applications ought to follow "common form", where one is known or indicated, when generating HTTP constructs, since there might exist some implementations that fail to accept anything beyond the common forms.

```
message-header = field-name ":" [ field-value ]
field-name     = token
field-value    = *( field-content | LWS )
field-content  = <the OCTETs making up the field-value
                 and consisting of either *TEXT or combinations
                 of token, separators, and quoted-string>
```

The field-content does not include any leading or trailing LWS: linear white space occurring before the first non-whitespace character of the field-value or after the last non-whitespace character of the field-value. Such leading or trailing LWS MAY be removed without changing the semantics of the field value. Any LWS that occurs between field-content MAY be replaced with a single SP before interpreting the field value or forwarding the message downstream.

The order in which header fields with differing field names are received is not significant. However, it is "good practice" to send general-header fields first, followed by request-header or response- header fields, and ending with the entity-header fields.

Multiple message-header fields with the same field-name MAY be present in a message if and only if the entire field-value for that header field is defined as a comma-separated list [i.e., #(values)]. It MUST be possible to combine the multiple header fields into one "field-name: field-value" pair, without changing the semantics of the message, by appending each subsequent field-value to the first, each separated by a comma. The order in which header fields with the same field-name are received is therefore significant to the interpretation of the combined field value, and thus a proxy MUST NOT change the order of these field values when a message is forwarded.

Message Body

The message-body (if any) of an HTTP message is used to carry the entity-body associated with the request or response. The message-body differs from the entity-body only when a transfer-coding has been applied, as indicated by the Transfer-Encoding header field (section 14.41).

message-body = entity-body
| <entity-body encoded as per Transfer-Encoding>

Transfer-Encoding MUST be used to indicate any transfer-codings applied by an application to ensure safe and proper transfer of the message. Transfer-Encoding is a property of the message, not of the entity, and thus MAY be added or removed by any application along the request/response chain.

The rules for when a message-body is allowed in a message differ for requests and responses.

The presence of a message-body in a request is signaled by the inclusion of a Content-Length or Transfer-Encoding header field in the request's message-headers. A message-body MUST NOT be included in a request if the specification of the request method does not allow sending an entity-body in requests. A server SHOULD read and forward a message-body on any request; if the request method does not include defined semantics for an entity-body, then the message-body SHOULD be ignored when handling the request.

For response messages, whether or not a message-body is included with a message is dependent on both the request method and the response status code. All responses to the HEAD request method MUST NOT include a message-body, even though the presence of entity- header fields might lead one to believe they do. All 1xx (informational), 204 (no content), and 304 (not modified) responses MUST NOT include a message-body. All other responses do include a message-body, although it MAY be of zero length.

Message Length

The transfer-length of a message is the length of the message-body as it appears in the message; that is, after any transfer-codings have been applied. When a message-body is included with a message, the transfer-length of that body is determined by one of the following (in order of precedence):

1.Any response message which "MUST NOT" include a message-body (such as the 1xx, 204, and 304 responses and any response to a HEAD request) is always terminated by the first empty line

after the header fields, regardless of the entity-header fields present in the message.

2. If a Transfer-Encoding header field (section 14.41) is present and has any value other than "identity", then the transfer-length is defined by use of the "chunked" transfer-coding, unless the message is terminated by closing the connection.

3. If a Content-Length header field (section 14.13) is present, its decimal value in OCTETs represents both the entity-length and the transfer-length. The Content-Length header field MUST NOT be sent if these two lengths are different (i.e., if a Transfer-Encoding header field is present). If a message is received with both a Transfer-Encoding header field and a Content-Length header field, the latter MUST be ignored.

4. If the message uses the media type "multipart/byteranges", and the transfer-length is not otherwise specified, then this self-delimiting media type defines the transfer-length. This media type MUST NOT be used unless the sender knows that the recipient can parse it; the presence in a request of a Range header with multiple byte- range specifiers from a 1.1 client implies that the client can parse multipart/byteranges responses.

A range header might be forwarded by a 1.0 proxy that does not understand multipart/byte ranges; in this case the server MUST delimit the message using methods defined in items 1, 3 or 5 of this section.

5. By the server closing the connection. (Closing the connection cannot be used to indicate the end of a request body, since that would leave no possibility for the server to send back a response.) For compatibility with HTTP/1.0 applications, HTTP/1.1 requests containing a message-body MUST include a valid Content-Length header field unless the server is known to be HTTP/1.1 compliant. If a request contains a message-body and a Content-Length is not given, the server SHOULD respond with 400 (bad request) if it cannot determine the length of the message, or with 411 (length required) if it wishes to insist on receiving a valid Content-Length.

All HTTP/1.1 applications that receive entities MUST accept the "chunked" transfer-coding (section 3.6), thus allowing this mechanism to be used for messages when the message length cannot be determined in advance.

Messages MUST NOT include both a Content-Length header field and a non-identity transfer-coding. If the message does include a non- identity transfer-coding, the Content-Length MUST be ignored.

When a Content-Length is given in a message where a message-body is allowed, its field value MUST exactly match the number of OCTETs in the message-body. HTTP/1.1 user agents MUST notify the user when an invalid length is received and detected.

General Header Fields

There are a few header fields which have general applicability for both request and response messages, but which do not apply to the entity being transferred. These header fields apply only to the message being transmitted.

```
general-header = Cache-Control          ;
               | Connection             ;
               | Date                   ;
               | Pragma                 ;
               | Trailer                ;
               | Transfer-Encoding      ;
               | Upgrade                ;
               | Via                    ;
               | Warning                ;
```

General-header field names can be extended reliably only in combination with a change in the protocol version. However, new or experimental header fields may be given the semantics of general header fields if all parties in the communication recognize them to be general-header fields. Unrecognized header fields are treated as entity-header fields.