LECTURE NOTES ON
COMPUTER NETWORKS
(UPCIT5302)

Compiled By,

Debi Prasad Mishra

Faculty, IT, CET Bhubaneswar

DISCLAIMER

The lecturer notes are prepared based on the lectures delivered previously. Few of the material are extracted from various sources and majority is from the lectures notes delivered during the theory class. Care has been taken to cover the syllabus within the stipulated hours as prescribed therein. The students are required to consider it just an indicative of their curriculum, not as exhaustive course content. They are also requested to follow the prescribed text book in the syllabus and also practice the question papers at the end of the chapters.

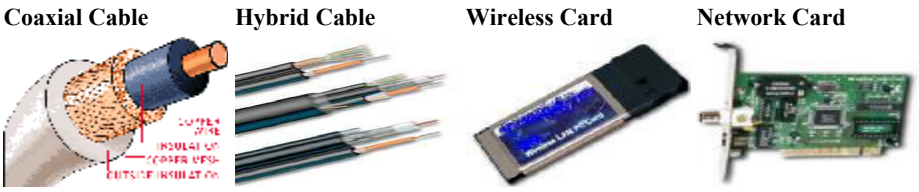
ISO-OSI 7-Layer Network Architecture

This lecture introduces the ISO-OSI layered architecture of Networks. According to the ISO standards, networks have been divided into 7 layers depending on the complexity of the fucntionality each of these layers provide. The detailed description of each of these layers is given in the notes below. We will first list the layers as defined by the standard in the increasing order of function complexity:

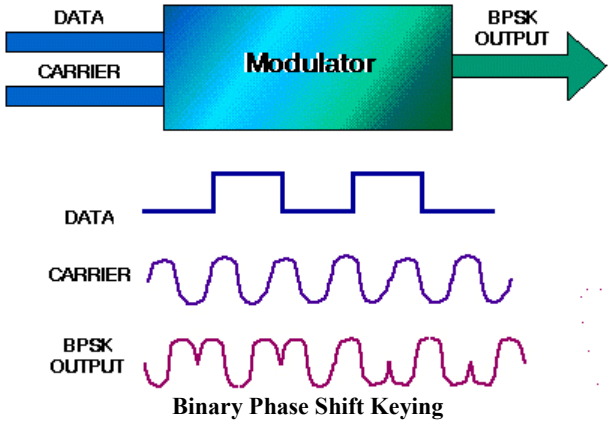
Physical Layer

This layer is the lowest layer in the OSI model. It helps in the transmission of data between two machines that are communicating through a physical medium, which can be optical fibres,copper wire or wireless etc. The following are the main functions of the physical layer:

1. **Hardware Specification:** The details of the physical cables, network interface cards, wireless radios, etc are a part of this layer.



2. **Encoding and Signalling:** How are the bits encoded in the medium is also decided by this layer. For example, on the coppar wire medium, we can use differnet voltage levels for a certain time interval to represent '0' and '1'. We may use +5mV for 1nsec to represent '1' and -5mV for 1nsec to represent '0'. All the issues of modulation is dealt with in this layer. eg, we may use Binary phase shift keying for the representation of '1' and '0' rather than using different volatage levels if we have to transfer in RF waves.



Binary Phase Shift Keying

3. **Data Transmission and Reception:** The transfer of each bit of data is the responsibility of this layer. This layer assures the transmissoin of each bit with a *high probability*. The transmission of the bits is not completely reliable as their is no error correction in this layer.
4. **Topology and Network Design:** The network design is the integral part of the physical layer. Which part of the network is the router going to be placed, where the switches will be used, where we will put the hubs, how many machines is each switch going to handle, what server is going to be placed where, and many such concerns are to be taken care of by the physical layer. The variosu kinds of netopologies that we decide to use may be ring, bus, star or a hybrid of these topologies depending on our requirements.

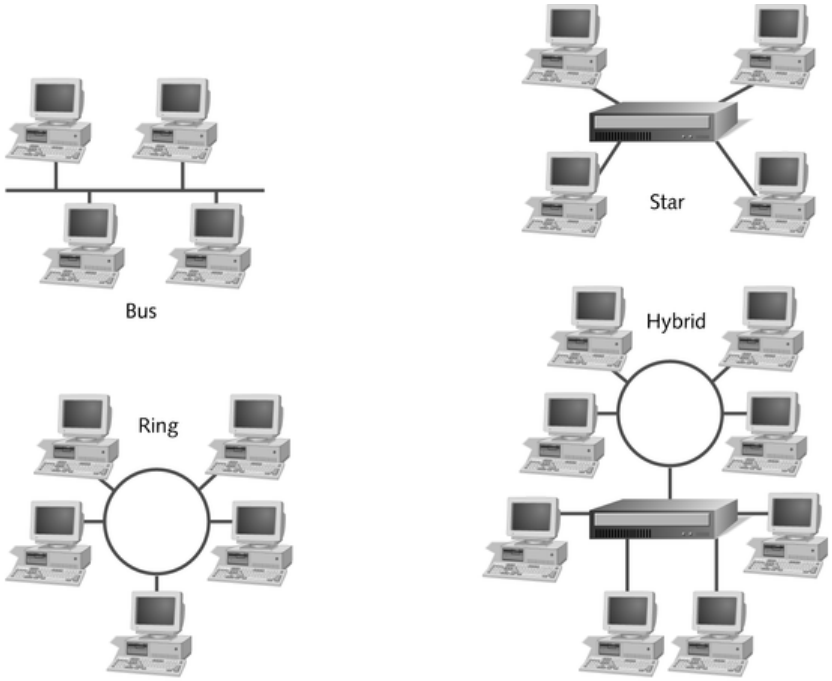
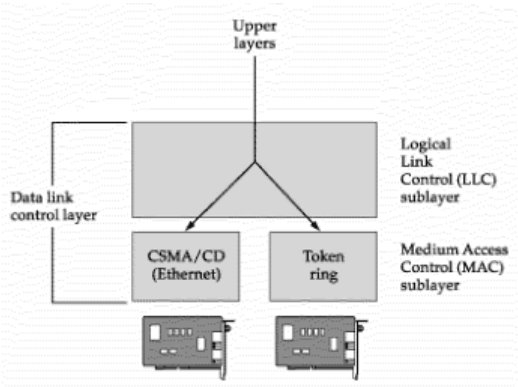


Figure 1-7 Commonly used network topologies

Data Link Layer

This layer provides reliable transmission of a packet by using the services of the physical layer which transmits bits over the medium in an unreliable fashion. This layer is concerned with :

1. Framing : Breaking input data into frames (typically a few hundred bytes) and caring about the frame boundaries and the size of each frame.
2. Acknowledgment : Sent by the receiving end to inform the source that the frame was received without any error.
3. Sequence Numbering : To acknowledge which frame was received.
4. Error Detection : The frames may be damaged, lost or duplicated leading to errors.The error control is on **link to link** basis.
5. Retransmission : The packet is retransmitted if the source fails to receive acknowledgment.
6. Flow Control : Necessary for a fast transmitter to keep pace with a slow receiver.

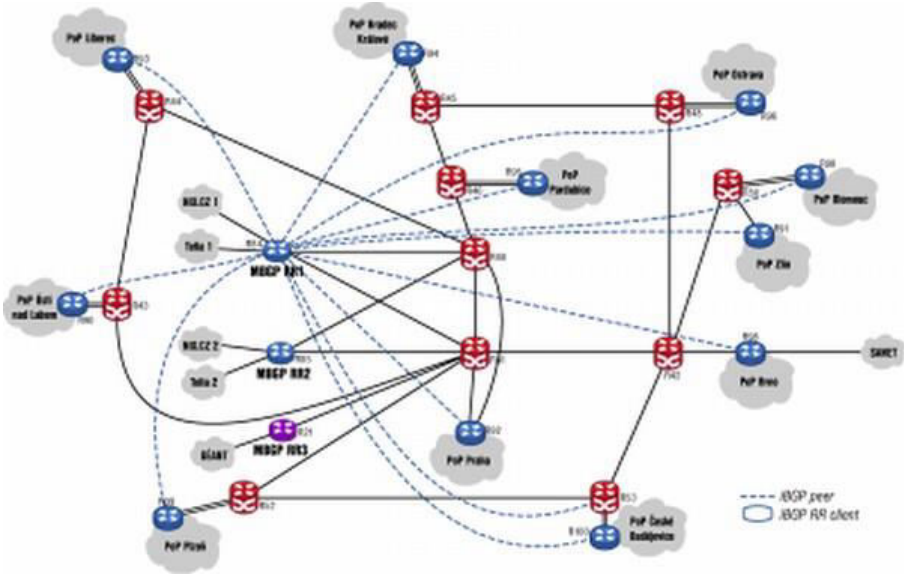


Data Link Layer

Network Layer

Its basic functions are routing and congestion control.
Routing: This deals with determining how packets will be routed (transferred) from source to destination. It can be of three types :

- Static : Routes are based on static tables that are "wired into" the network and are rarely changed.
- Dynamic : All packets of one application can follow different routes depending upon the topology of the network, the shortest path and the current network load.
- Semi-Dynamic : A route is chosen at the start of each conversation and then all the packets of the application follow the same route.



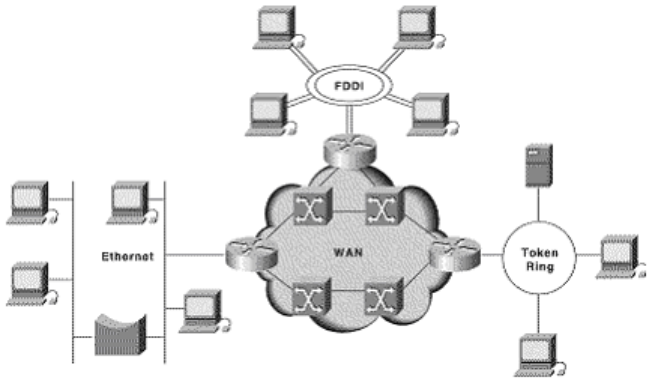
Routing

The services provided by the network can be of two types :

- **Connection less service:** Each packet of an application is treated as an independent entity. On each packet of the application the destination address is provided and the packet is routed.
- **Connection oriented service:** Here, first a connection is established and then all packets of the application follow the same route. To understand the above concept, we can also draw an analogy from the real life. Connection oriented service is modeled after the telephone system. All voice packets go on the same path after the connection is established till the connection is hung up. It acts like a tube ; the sender pushes the objects in at one end and the receiver takes them out in the same order at the other end. Connection less service is modeled after the postal system. Each letter carries the destination address and is routed independent of all the others. Here, it is possible that the letter sent first is delayed so that the second letter reaches the destination before the first letter.

Congestion Control: A router can be connected to 4-5 networks. If all the networks send packet at the same time with maximum rate possible then the router may not be able to handle all the packets and may drop some/all packets. In this context the dropping of the packets should be minimized and the source whose packet was dropped should be informed. The control of such congestion is also a function of the network layer. Other issues related with this layer are transmitting time, delays, jittering.

Internetworking: Internetworks are multiple networks that are connected in such a way that they act as one large network, connecting multiple office or department networks. Internetworks are connected by networking hardware such as routers, switches, and bridges.Internetworking is a solution born of three networking problems: isolated LANs, duplication of resources, and the lack of a centralized network management system. With connected LANs, companies no longer have to duplicate programs or resources on each network. This in turn gives way to managing the network from one central location instead of trying to manage each separate LAN. We should be able to transmit any packet from one network to any other network even if they follow different protocols or use different addressing modes.



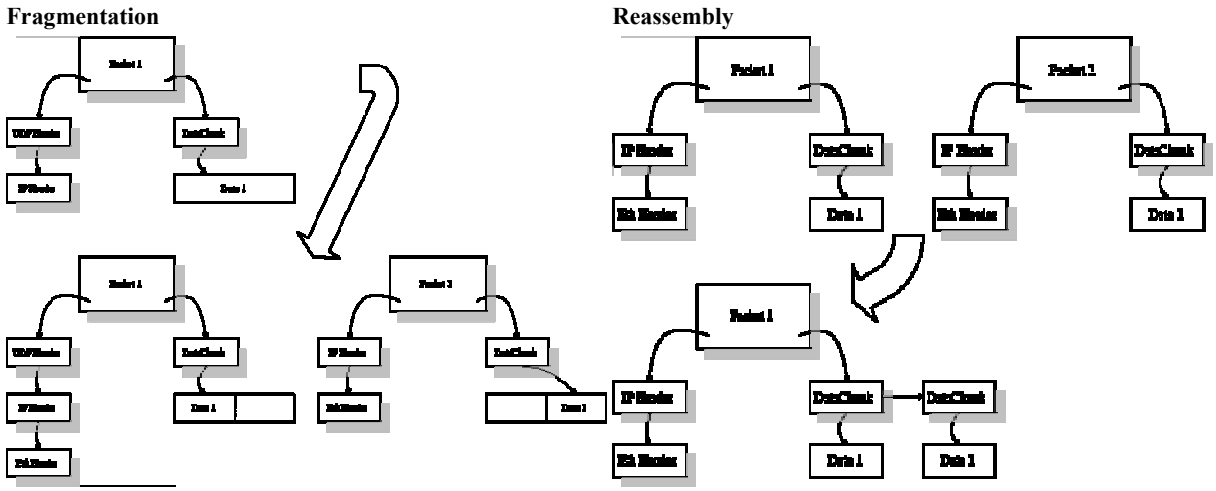
Inter-Networking

Network Layer **does not** guarantee that the packet will reach its intended destination. There are no reliability guarantees.

Transport Layer

Its functions are :

- **Multiplexing / Demultiplexing :** Normally the transport layer will create distinct network connection for each transport connection required by the session layer. The transport layer may either create multiple network connections (to improve throughput) or it may multiplex several transport connections onto the same network connection (because creating and maintaining networks may be expensive). In the latter case, demultiplexing will be required at the receiving end. A point to note here is that communication is always carried out between two processes and not between two machines. This is also known as process-to-process communication.
- **Fragmentation and Re-assembly :** The data accepted by the transport layer from the session layer is split up into smaller units (fragmentation) if needed and then passed to the network layer. Correspondingly, the data provided by the network layer to the transport layer on the receiving side is re-assembled.



- **Types of service :** The transport layer also decides the type of service that should be provided to the session layer. The service may be perfectly reliable, or may be reliable within certain tolerances or may not be reliable at all. The message may or may not be received in the order in which it was sent. The decision regarding the type of service to be provided is taken at the time when the connection is established.
- **Error Control :** If reliable service is provided then error detection and error recovery operations are also performed. It provides error control mechanism on **end to end** basis.
- **Flow Control :** A fast host cannot keep pace with a slow one. Hence, this is a mechanism to regulate the flow of information.
- **Connection Establishment / Release :** The transport layer also establishes and releases the connection across the network. This requires some sort of naming mechanism so that a process on one machine can indicate with whom it wants to communicate.

References of Images

- http://www.putergeek.com/.../ pci_combo_card_sm.jpg
- http://blue.utb.edu/libertad/clipart/pi_wireless_pc_card_b.jpg
- http://www.commscope.com/ images/hybrids.jpg hybrid cable
- http://www.cba.nau.edu/facstaff/maris-j/SavedStuff/Images/net_topo.gif
- http://www.ces.net/doc/2003/research/xl-unicast-routing.gif
- http://www.infinitygroup.com/images/internetworking.gif
- http://www.microway.com/.../ data_link_layer.gif
- http://searchnetworking.techtarget.com/WhatIs/images/coaxla.gif
- http://www.df.lth.se/~pkj/thesis_report/img13.gif
- http://www.df.lth.se/~pkj/ thesis_report/img12.gif
- http://www.ifla.org/VI/5/reports/rep3/rep3-2.gif
- http://msp.gsfc.nasa.gov/tdrss/bpsk.gif

Session Layer

It deals with the concept of **Sessions** i.e. when a user logins to a remote server he should be **authenticated** before getting access to the files and application programs. Another job of session layer is to establish and maintain sessions. If during the transfer of data between two machines the session breaks down, it is the session layer which re-establishes the connection. It also ensures that the data transfer starts from where it breaks keeping it transparent to the end user. e.g. In case of a session with a database server, this layer introduces **check points** at various places so that in case the connectoin is broken and reestablished, the transition running on the database is not lost even if the user has not committed. This activity is called **Synchronization**. Another function of this layer is **Dialogue Control** which determines whose turn is it to speak in a session. It is useful in video conferencing.

Presentation Layer

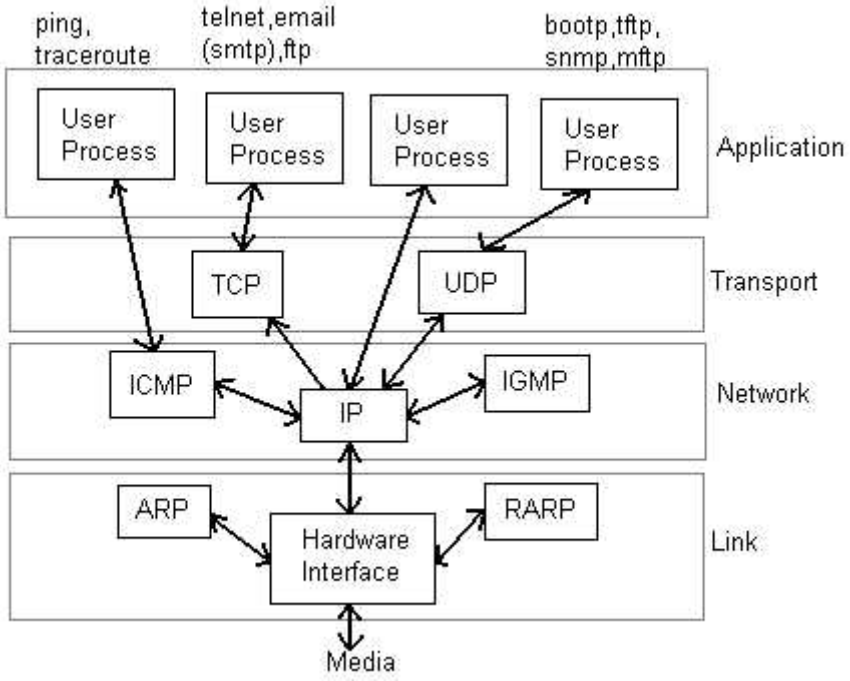
This layer is concerned with the syntax and semantics of the information transmitted. In order to make it possible for computers with different data representations to communicate data structures to be exchanged can be defined in abstract way alongwith standard encoding. It also manages these abstract data structres and allows higher level of data structres to be defined an exchange. It encodes the data in standard agreed way(network format). Suppose there are two machines A and B one follows 'Big Endian' and other 'Little Endian' for data representation. This layer ensures that the data transmitted by one gets converted in the form compatibale to othe machine. This layer is concerned with the syntax and semantics of the information transmitted.In order to make it possible for computers with different data representations to communicate data structures to be exchanged canbe defined in abstract way alongwith standard encoding. It also manages these abstract data structres and allows higher level of data structres to be defined an exchange. Other functions include compression, encryption etc.

Application Layer

The seventh layer contains the application protocols with which the user gains access to the network. The choice of which specific protocols and their associated functions are to be used at the application level is up to the individual user. Thus the boundary between the presentation layer and the application layer represents a separation of the protocols imposed by the network designers from those being selected and implemented by the network users.For example commonly used protocols are HTTP(for web browsing), FTP(for file transfer) etc.

Network Layers as in Practice

In most of the networks today, we do not follow the OSI model of seven layers. What is actually implemented is as follows. The functionality of Application layer and Presentation layer is merged into one and is called as the Application Layer. Functionalities of Session Layer is not implemented in most networks today. Also, the Data Link layer is split theoretically into **MAC (Medium Access Control) Layer** and **LLC (Link Layer Control)**. But again in practice, the LLC layer is not implemented by most networks. So as of today, the network architecture is of 5 layers only.



Network Layers in Internet Today

Some Related Links on OSI Model and TCP Model

- http://en.wikipedia.org/wiki/OSI_model
- http://www.tcpipguide.com/free/t_OSIReferenceModelLayers.htm
- http://www.geocities.com/SiliconValley/Monitor/3131/ne/osimodel.html
- http://www.tech-faq.com/osi-model.shtml
- http://www.networkdictionary.com/protocols/osimodel.php

Physical Layer

Physical layer is concerned with transmitting raw bits over a communication channel. The design issues have to do with making sure that when one side sends a 1 bit, it is recieved by the other side as 1 bit and not as 0 bit. In physical layer we deal with the communication medium used for transmission.

Types of Medium

Medium can be classified into 2 categories.

1. **Guided Media :** Guided media means that signals is guided by the prescence of physical media i.e. signals are under control and remains in the physical wire. For eg. copper wire.
2. **Unguided Media :** Unguided Media means that there is no physical path for the signal to propogate. Unguided media are essentially electro-magnetic waves. There is no control on flow of signal. For eg. radio waves.

Communication Links

In a nework nodes are connected through links. The communication through links can be classified as

1. **Simplex :** Communication can take place only in one direction. eg. T.V broadcasting.
2. **Half-duplex :** Communication can take place in one direction at a time. Suppose node A and B are connected then half-duplex communication means that at a time data can flow from A to B or from B to A but not simultaneously. eg. two persons talking to each other such that when speaks the other listens and vice versa.
3. **Full-duplex :** Communication can take place simultaneously in both directions. eg. A discussion in a group without discipline.

Links can be further classified as

1. **Point to Point :** In this communication only two nodes are connected to each other. When a node sends a packet then it can be recieved only by the node on the other side and none else.
2. **Multipoint** : It is a kind of sharing communication, in which signal can be recieved by all nodes. This is also called broadcast.

Generally two kind of problems are associated in transmission of signals.

1. **Attenuation :** When a signal transmitts in a network then the quality of signal degrades as the signal travels longer distances in the wire. This is called attenuation. To improve quality of signal amplifiers are used at regular distances.
2. **Noise :** In a communication channel many signals transmits simultaneously, certain random signals are also present in the medium. Due to interference of these signals our signal gets disrupted a bit.

Bandwidth

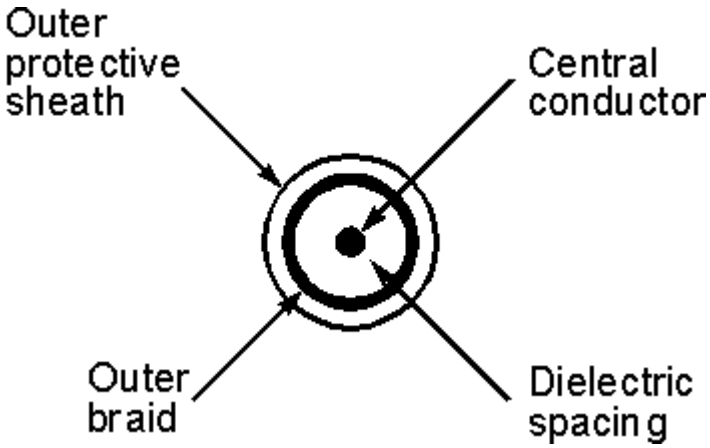
Bandwidth simply means how many bits can be transmitted per second in the communication channel. In technical terms it indicates the width of frequency spectrum.

Transmission Media

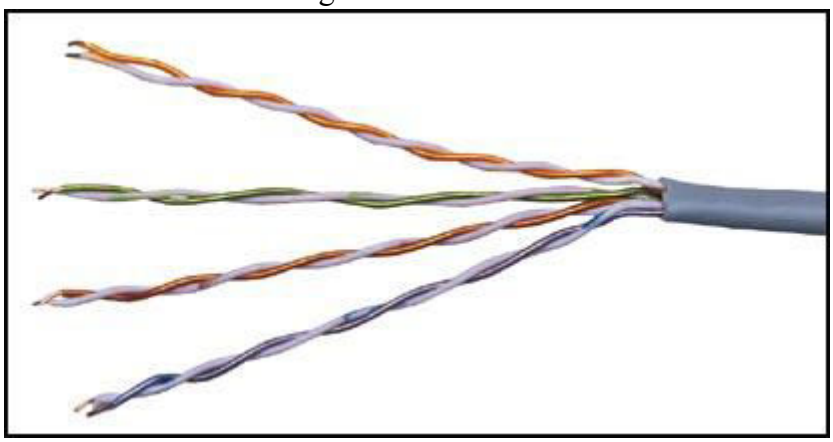
Guided Transmission Media
In Guided transmission media generally two kind of materials are used.

1. Copper
    - Coaxial Cable
    - Twisted Pair
2. Optical Fiber

1. **Coaxial Cable:** Coaxial cable consists of an inner conductor and an outer conductor which are seperated by an insulator. The inner conductor is usually copper. The outer conductor is covered by a plastic jacket. It is named coaxial because the two conductors are coaxial. Typical diameter of coaxial cable lies between 0.4 inch to 1 inch. The most application of coaxial cable is cable T.V. The coaxial cable has high bandwidth, attenuation is less.



2. **Twisted Pair:** A Twisted pair consists of two insulated copper wires, typically 1mm thick. The wires are twisted togather in a helical form the purpose of twisting is to reduce cross talk interference between several pairs. Twisted Pair is much cheaper then coaxial cable but it is susceptible to noise and electromagnetic interference and attenuation is large.



Twisted Pair can be further classified in two categories:
Unshielded twisted pair: In this no insulation is provided, hence they are susceptible to interference.
Shielded twisted pair: In this a protective thick insulation is provided but shielded twisted pair is expensive and not commonly used.

The most common application of twisted pair is the telephone system. Nearly all telephones are connected to the telephone company office by a twisted pair. Twisted pair can run several kilometers without amplification, but for longer distances repeaters are needed. Twisted pairs can be used for both analog and digital transmission. The bandwidth depends on the thickness of wire and the distance travelled. Twisted pairs are generally limited in distance, bandwidth and data rate.

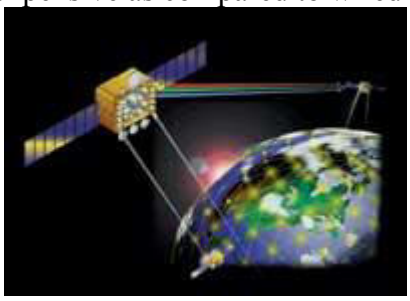
3. **Optical Fiber:** In optical fiber light is used to send data. In general terms prescence of light is taken as bit 1 and its absence as bit 0. Optical fiber consists of inner core of either glass or plastic. Core is surrounded by cladding of the same material but of different refrective index. This cladding is surrounded by a plastic jacket which prevents optical fiber from electromagnetic interferrence and harshy environments. It uses the principle of total internal reflection to transfer data over optical fibers. Optical fiber is much better in bandwidth as compared to copper wire, since there is hardly any attenuation or electromagnetic interference in optical wires. Hence there is less requirement to improve quality of signal, in long distance transmission. Disadvantage of optical fiber is that end points are fairly expensive. (eg. switches)

Differences between different kinds of optical fibers:

1. Depending on material
    - Made of glass
    - Made of plastic.
2. Depending on radius
    - Thin optical fiber
    - Thick optical fiber
3. Depending on light source
    - LED (for low bandwidth)
    - Injection lased diode (for high bandwidth)

Wireless Transmission

1. **Radio:** Radio is a general term that is used for any kind of frequency. But higher frequencies are usually termed as microwave and the lower frequency band comes under radio frequency. There are many application of radio. For eg. cordless keyboard, wireless LAN, wireless ethernet. but it is limited in range to only a few hundred meters. Depending on frequency radio offers different bandwidths.
2. **Terrestrial microwave:** In terrestrial microwave two antennas are used for communication. A focused beam emerges from an antenna and is recieved by the other antenna, provided that antennas should be facing each other with no obstacle in between. For this reason antennas are situated on high towers. Due to curvature of earth terristial microwave can be used for long distance communication with high bandwidth. Telecom department is also using this for long distance communication. An advantage of wireless communication is that it is not required to lay down wires in the city hence no permissions are required.
3. **Satellite communication:** Satellite acts as a switch in sky. On earth VSAT(Very Small Aperture Terminal) are used to transmit and recieve data from satellite. Generally one station on earth transmitts signal to satellite and it is recieved by many stations on earth. Satellite communication is generally used in those places where it is very difficult to obtain line of sight i.e. in highly irregular terristial regions. In terms of noise wireless media is not as good as the wired media. There are frequency band in wireless communication and two stations should not be allowed to transmit simultaneously in a frequency band. The most promising advantage of satellite is broadcasting. If satellites are used for point to point communication then they are expensive as compared to wired media.



References of Images

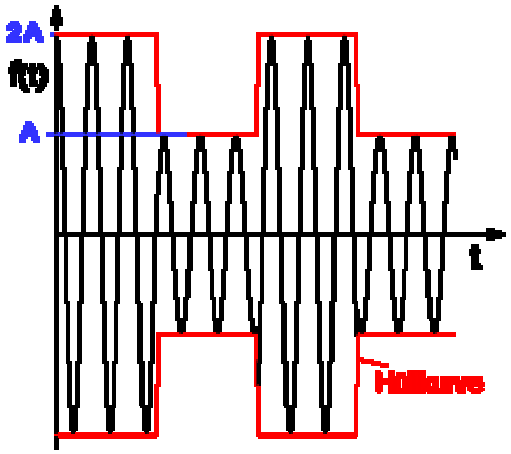
- http://ai3.asti.dost.gov.ph/sat/levels.jpg
- http://www.ll.mit.edu/Image-Lib/photos/color-Satellite.jpg
- http://oldsite.vislab.usyd.edu.au/photonics/revolution/technology/images/twisted_pair.jpg
- http://www.radio-electronics.com/info/antennas/coax/cross_section_thro_coax.gif

Data Encoding

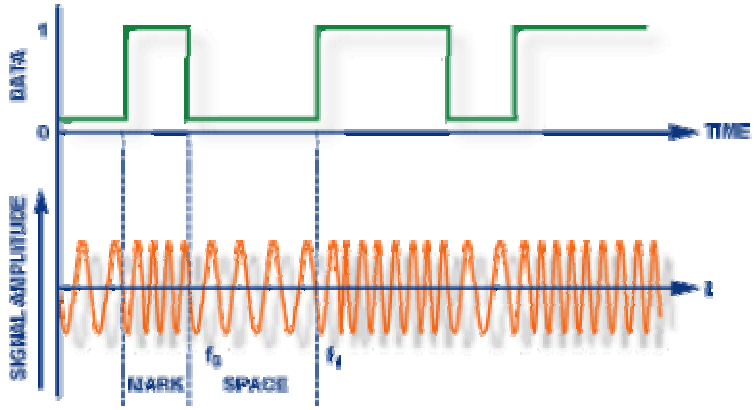
Digital data to analog signals

A modem (modulator-demodulator) converts digital data to analog signal. There are 3 ways to modulate a digital signal on an analog carrier signal.

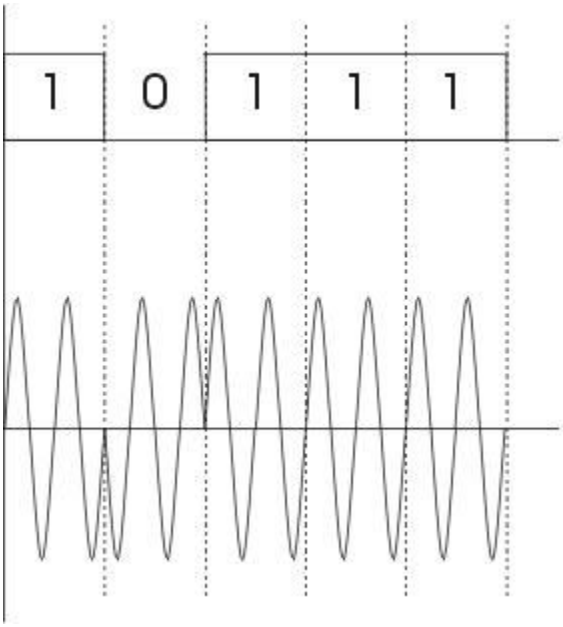
1. **Amplitude shift keying (ASK):** is a form of modulation which represents digital data as variations in the amplitude of a carrier wave. Two different amplitudes of carrier frequency represent '0' , '1'.



2. **Frequency shift keying (FSK):** In Frequency Shift Keying, the change in frequency define different digits. Two different frequencies near carrier frequency represent '0' ,"1'.



3. **Phase shift keying (PSK):** The phase of the carrier is discretely varied in relation either to a reference phase or to the phase of the immediately preceding signal element, in accordance with data being transmitted. Phase of carrier signal is shifted to represent '0' , '1'.



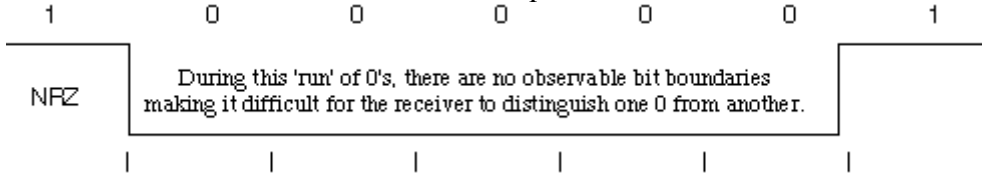
Digital data to digital signals

A digital signal is sequence of discrete , discontinuous voltage pulses. Each pulses a signal element. Encoding scheme is an important factor in how successfully the receiver interprets the incoming signal.

Encoding Techniques

Following are several ways to map data bits to signal elements.

- **Non return to zero(NRZ)** NRZ codes share the property that voltage level is constant during a bit interval. High level voltage = bit 1 and Low level voltage = bit 0. A problem arises when there is a long sequence of 0s or 1s and the volatage level is maintained at the same value for a long time. This creates a problem on the recieving end because now, the clock synchronization is lost due to lack of any transitions and hence, it is difficult to determine the exact number of 0s or 1s in this sequence.



The two variations are as follows:

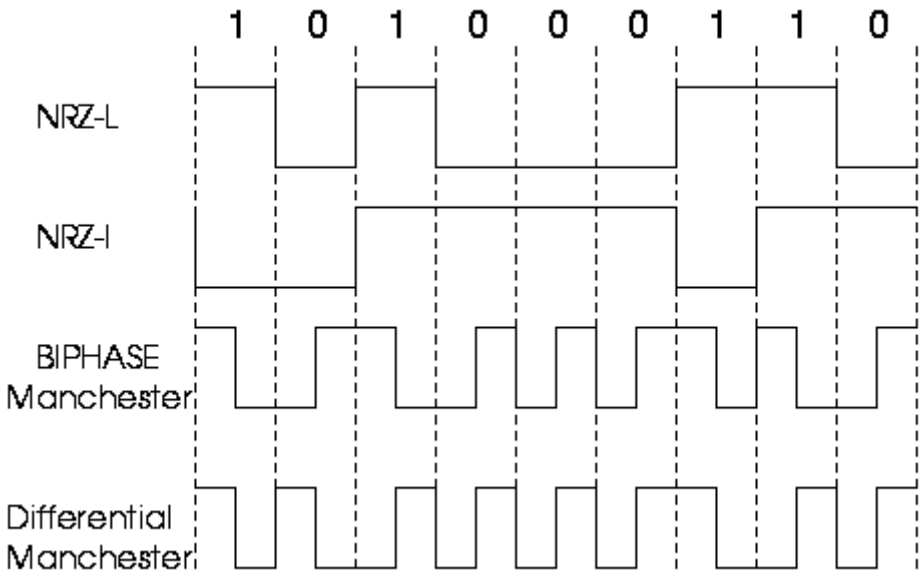
1. **NRZ-Level:** In NRZ-L encoding, the polarity of the signal changes only when the incoming signal changes from a 1 to a 0 or from a 0 to a 1. NRZ-L method looks just like the NRZ method, except for the first input one data bit. This is because NRZ does not consider the first data bit to be a polarity change, where NRZ-L does.
2. **NRZ-Inverted:** Transition at the beginning of bit interval = bit 1 and No Transition at beginning of bit interval = bit 0 or vicecersa. This technique is known as differential encoding.

NRZ-I has an advantage over NRZ-L. Consider the situation when two data wires are wrongly connected in each other's place.In NRZ-L all bit sequences will get reversed (B'coz voltage levels get swapped).Whereas in NAZ-I since bits are recognized by transition the

bits will be correctly interpreted. A disadvantage in NRZ codes is that a string of 0's or 1's will prevent synchronization of transmitter clock with receiver clock and a separate clock line need to be provided.

- **Biphase encoding:** It has following characteristics:
  1. Modulation rate twice that of NRZ and bandwidth correspondingly greater. (Modulation is the rate at which signal level is changed).
  2. Because there is predictable transition during each bit time,the receiver can synchronize on that transition i.e. clock is extracted from the signal itself.
  3. Since there can be transition at the beginning as well as in the middle of the bit interval the clock operates at twice the data transfer rate.

  Types of Encoding -->
  - **Biphase-manchester:** Transition from high to low in middle of interval = 1 and Transition from low to high in middle of interval = 0
  - **Differential-manchester:** Always a transition in middle of interval. No transition at beginning of interval=1 and Transition at beginning of interval = 0



  - **4B/5B Encoding:** In Manchester encoding scheme , there is a transition after every bit. It means that we must have clocks with double the speed to send same amount of data as in NRZ encodings. In other words, we may say that only 50% of the data is sent. This performance factor can be significantly improved if we use a better encoding scheme. This scheme may have a transition after fixed number of bits instead of every other bit. Like if we have a transition after every four bits, then we will be sending 80% data of actual capacity. This is a significant improvement in the performance.

    This scheme is known as **4B/5B**. So here we convert 4-bits to 5-bits, ensuring at least one transition in them. The basic idea here is that 5-bit code selected must have :

    - one leading 0
    - no more than two trailing 0s

    Thus it is ensured that we can never have more than three consecutive 0s. Now these 5-bit codes are transmitted using NRZI coding thus problem of consecutive 1s is solved.

The exact transformation is as follows :

4-bit Data	5-bit code	4-bit Data	5-bit code
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

Of the remaining 16 codes, 7 are invalid and others are used to send some control information like line idle(11111), line dead(00000), Halt(00100) etc.

There are other variants for this scheme viz. 5B/6B, 8B/10B etc. These have self suggesting names.

    - **8B/6T Encoding:** In the above schemes, we have used two/three voltage levels for a signal. But we may altogether use more than three voltage levels so that more than one-bit could be send over a single signal. Like if we use six voltage levels and we use 8-bits then the scheme is called **8B/6T**. Clearly here we have 729(3^6) combinations for signal and 256(2^8) combinations for bits.
- **Bipolar AIM:** Here we have 3 voltage levels: middle,upper,lower
    - Representation 1: Middle level =0 Upper,Lower level =1 such that successive 1's will be represented alternately on upper and lower levels.
    - Representation 2 (pseudoternary): Middle level =1 Upper,Lower level=0

Analog data to digital signal:

The process is called digitization. Sampling frequency must be at least twice that of highest frequency present in the the signal so that it may be fairly regenerated. Quantization - Max. and Min values of amplitude in the sample are noted. Depending on number of bits (say n) we use we divide the interval (min,max) into 2(^n) number of levels. The amplitude is then approximated to the nearest level by a 'n' bit integer. The digital signal thus consists of blocks of n bits.On reception the process is reversed to produce analog signal. But a lot of data can be lost if fewer bits are used or sampling frequency not so high.

- **Pulse code modulation(PCM):** Here intervals are equally spaced. 8 bit PCB uses 256 different levels of amplitude. In non-linear encoding levels may be unequally spaced.
- **Delta Modulation(DM):** Since successive samples do not differ very much we send the differences between previous and present sample. It requires fewer bits than in PCM.

Digital Data Communication Techniques:

For two devices linked by a transmission medium to exchange data ,a high degree of co-operation is required. Typically data is transmitted one bit at a time. The timing (rate, duration,spacing) of

these bits must be same for transmitter and receiver. There are two options for transmission of bits.

1. **Parallel** All bits of a byte are transferred simultaneously on separate parallel wires. Synchronization between multiple bits is required which becomes difficult over large distance. Gives large band width but expensive. Practical only for devices close to each other.
2. **Serial** Bits transferred serially one after other.Gives less bandwidth but cheaper. Suitable for transmission over long distances.

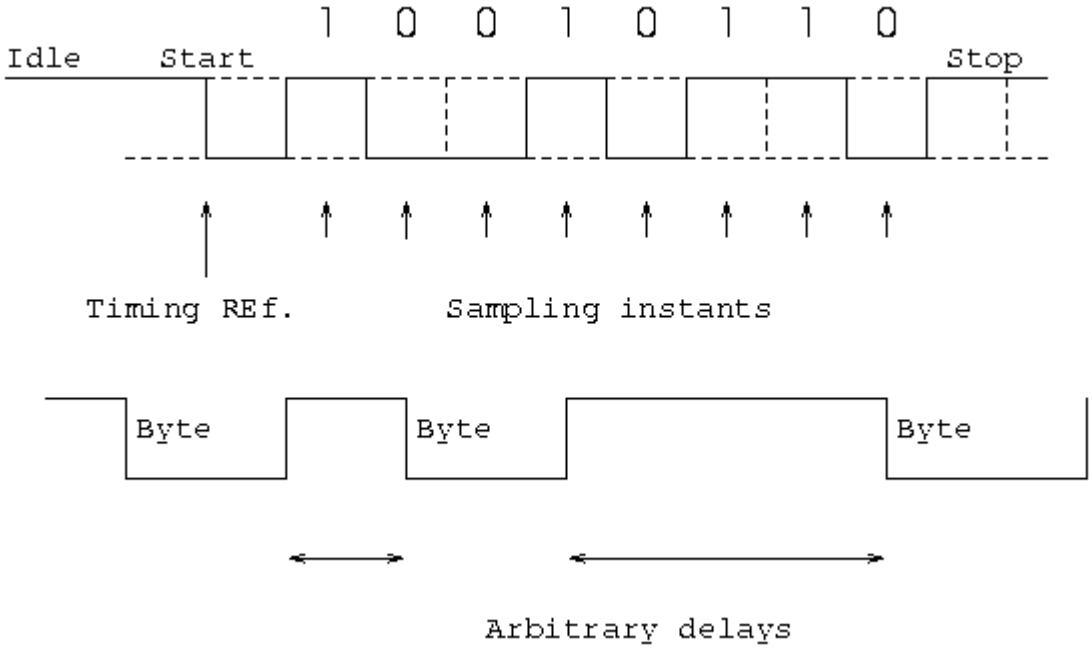
Transmission Techniques:

1. **Asynchronous:** Small blocks of bits(generally bytes) are sent at a time without any time relation between consecutive bytes .when no transmission occurs a default state is maintained corresponding to bit 1. Due to arbitrary delay between consecutive bytes,the time occurrences of the clock pulses at the receiving end need to be synchronized for each byte. This is achieved by providing 2 extra bits start and stop.

   Start bit: It is prefixed to each byte and equals 0. Thus it ensures a transition from 1 to 0 at onset of transmission of byte.The leading edge of start bit is used as a reference for generating clock pulses at required sampling instants. Thus each onset of a byte results in resynchronization of receiver clock.

   Stop bit: To ensure that transition from 1 to 0 is always present at beginning of a byte it is necessary that default state be 1. But there may be two bytes one immediately following the other and if last bit of first byte is 0, transition from 1 to 0 will not occur . Therefore a stop bit is suffixed to each byte equaling 1. It's duration is usually 1,1.5,2 bits.

   Asynchronous transmission is simple and cheap but requires an overhead of 3 bits i.e. for 7 bit code 2 (start ,stop bits)+1 parity bit implying 30% overhead.However % can be reduced by sending larger blocks of data but then timing errors between receiver and sender can not be tolerated beyond [50/no. of bits in block] % (assuming sampling is done at middle of bit interval). It will not only result in incorrect sampling but also misaligned bit count i.e. a data bit can be mistaken for stop bit if receiver's clock is faster.



2. **Synchronous** - Larger blocks of bits are successfully transmitted.Blocks of data are either treated as sequence of bits or bytes. To prevent timing drift clocks at two ends need to be synchronized.This can done in two ways:
    1. Provide a separate clock line between receiver and transmitter. OR
    2. Clocking information is embedded in data signal i.e. biphase coding for digital signals.

    Still another level of synchronization is required so that receiver determines beginning or end of block of data. Hence each block begins with a start code and ends with a stop code.These are in general same known as flag that is unique sequence of fixed no. of bits.In addition some control characters encompass data within these flags. **Data+control information** is called a frame. Since any arbitrary bit pattern can be transmitted there is no assurance that bit pattern for flag will not appear inside the frame thus destroying frame level synchronization. So to avoid this we use bit stuffing

    Bit Stuffing: Suppose our flag bits are 01111110 (six 1's). So the transmitter will always insert an extra 0 bit after each occurrence of five 1's (except for flags). After detecting a starting flag the receiver monitors the bit stream . If pattern of five 1's appear, the sixth is examined and if it is 0 it isdeleted else if it is 1 and next is 0 the combination is accepted as a flag. Similarly byte stuffing is used for byte oriented transmission.Here we use an escape sequence to prefix a byte similar to flag and 2 escape sequences if byte is itself a escape sequence.

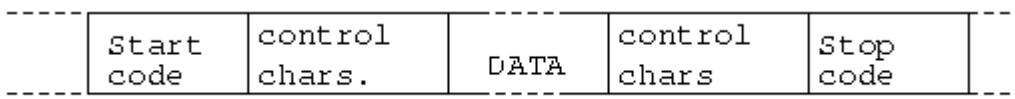


Image References:

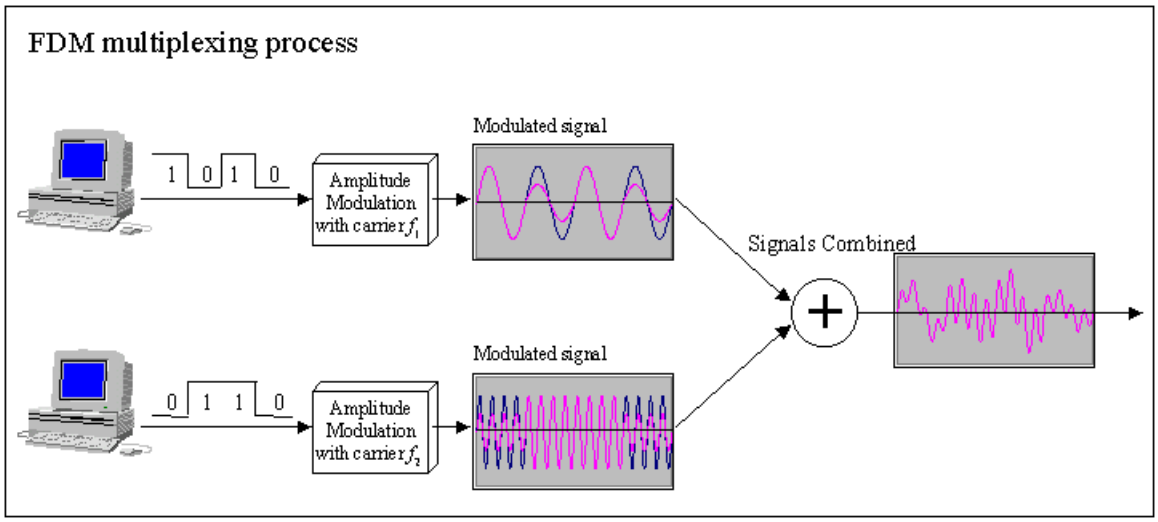
- http://www.analog.com/library/analogDialogue/archives/38-08/DDS_06.gif
- http://racon.net/archive/FHTW/projects/Richtfunk/2psk.png
- http://fara.cs.uni-potsdam.de/~rnitschk/wireless/antenna/pics/009001.gif

- http://www.erg.abdn.ac.uk/users/gorry/course/images/nrz-run.gif

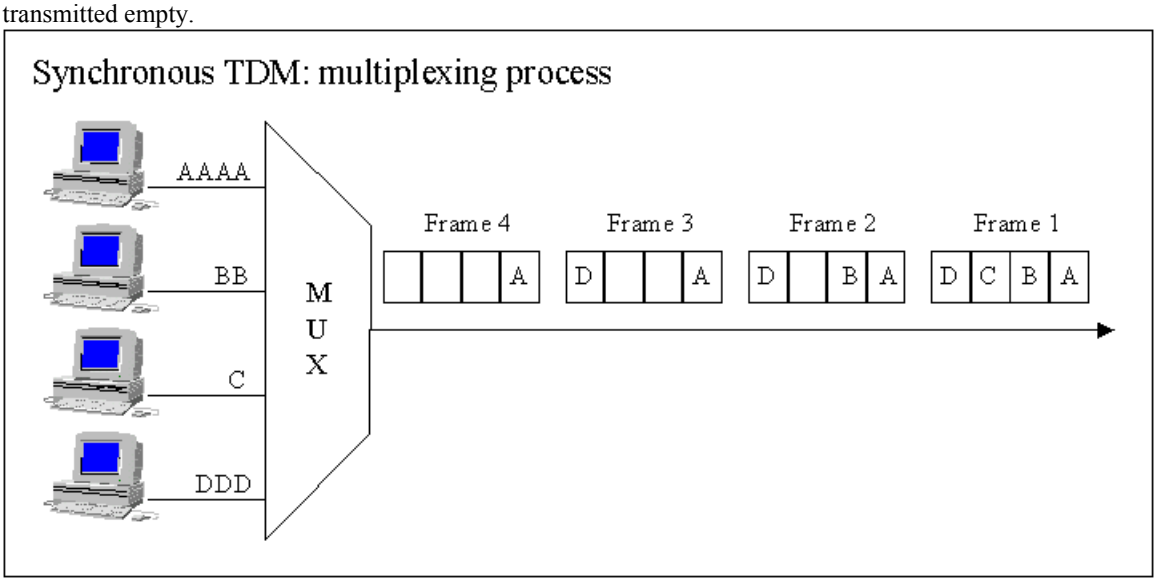
Multiplexing

When two communicating nodes are connected through a media, it generally happens that bandwidth of media is several times greater than that of the communicating nodes. Transfer of a single signal at a time is both slow and expensive. The whole capacity of the link is not being utilized in this case. This link can be further exploited by sending several signals combined into one. This combining of signals into one is called multiplexing.

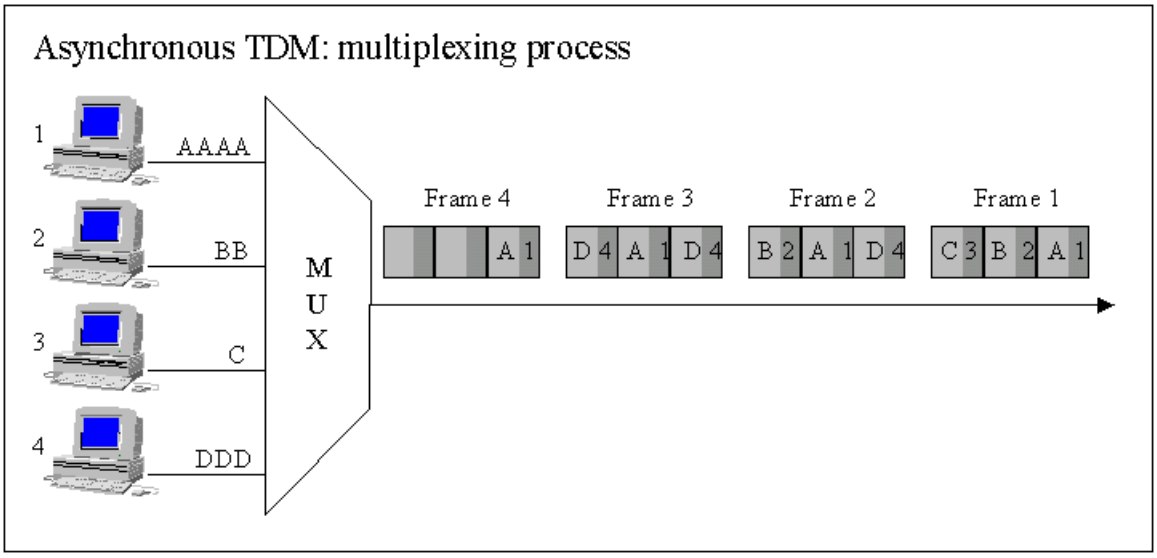
1. **Frequency Division Multiplexing (FDM):** This is possible in the case where transmission media has a bandwidth than the required bandwidth of signals to be transmitted. A number of signals can be transmitted at the same time. Each source is allotted a frequency range in which it can transfer it's signals, and a suitable frequency gap is given between two adjescent signals to avoid overlapping. This is type of multiplexing is commonly seen in the cable TV networks.



2. **Time Division Multiplexing (TDM):** This is possible when data transmission rate of the media is much higher than that of the data rate of the source. Multiple signals can be transmitted if each signal is allowed to be transmitted for a definite amount of time. These time slots are so small that all transmissions appear to be in parallel.
    1. **Synchronous TDM:** Time slots are preassigned and are fixed. Each source is given it's time slot at every turn due to it. This turn may be once per cycle, or several turns per cycle ,if it has a high data transfer rate, or may be once in a no. of cycles if it is slow. This slot is given even if the source is not ready with data. So this slot is transmitted empty.



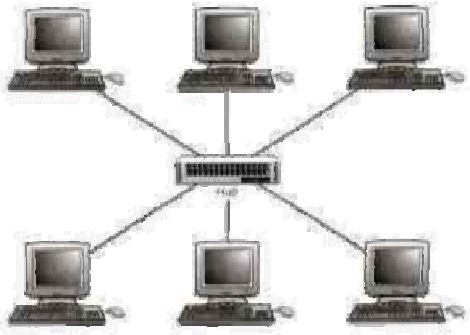
    2. **Asynchronous TDM:** In this method, slots are not fixed. They are allotted dynamically depending on speed of sources, and whether they are ready for transmission.



Network Topologies

A network topology is the basic design of a computer network. It is very much like a map of a road. It details how key network components such as nodes and links are interconnected. A network's topology is comparable to the blueprints of a new home in which components such as the electrical system, heating and air conditioning system, and plumbing are integrated into the overall design. Taken from the Greek work "Topos" meaning "Place," Topology, in relation to networking, describes the configuration of the network; including the location of the workstations and wiring connections. Basically it provides a definition of the components of a Local Area Network (LAN). A topology, which is a pattern of interconnections among nodes, influences a network's cost and performance. There are three primary types of network topologies which refer to the physical and logical layout of the Network cabling. They are:

1. **Star Topology:** All devices connected with a Star setup communicate through a central Hub by cable segments. Signals are transmitted and received through the Hub. It is the simplest and the oldest and all the telephone switches are based on this. In a star topology, each network device has a home run of cabling back to a network hub, giving each device a separate connection to the network. So, there can be multiple connections in parallel.



Advantages

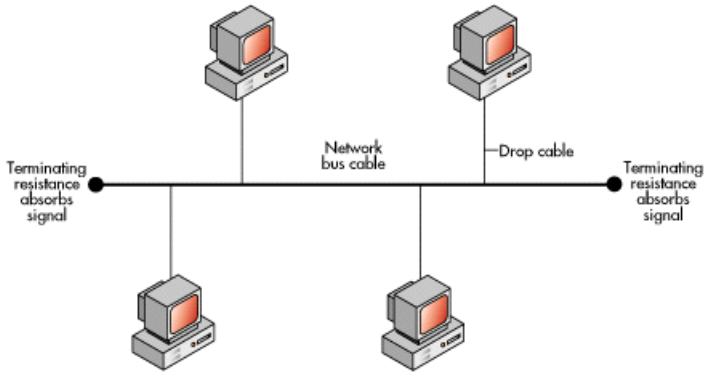
- Network administration and error detection is easier because problem is isolated to central node
- Networks runs even if one host fails
- Expansion becomes easier and scalability of the network increases
- More suited for larger networks

Disadvantages

- Broadcasting and multicasting is not easy because some extra functionality needs to be provided to the central hub
- If the central node fails, the whole network goes down; thus making the switch some kind of a bottleneck
- Installation costs are high because each node needs to be connected to the central switch

2. **Bus Topology:** The simplest and one of the most common of all topologies, Bus consists of a single cable, called a Backbone, that connects all workstations on the network using a single line. All transmissions must pass through each of the connected devices to complete the desired request. Each workstation has its own individual signal that

identifies it and allows for the requested data to be returned to the correct originator. In the Bus Network, messages are sent in both directions from a single point and are read by the node (computer or peripheral on the network) identified by the code with the message. Most Local Area Networks (LANs) are Bus Networks because the network will continue to function even if one computer is down. This topology works equally well for either peer to peer or client server.



The purpose of the terminators at either end of the network is to stop the signal being reflected back.

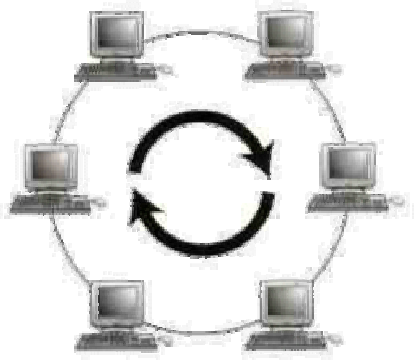
Advantages

- Broadcasting and multicasting is much simpler
- Network is redundant in the sense that failure of one node doesn't effect the network. The other part may still function properly
- Least expensive since less amount of cabling is required and no network switches are required
- Good for smaller networks not requiring higher speeds

Disadvantages

- Trouble shooting and error detection becomes a problem because, logically, all nodes are equal
- Less secure because sniffing is easier
- Limited in size and speed

3. **Ring Topology:** All the nodes in a Ring Network are connected in a closed circle of cable. Messages that are transmitted travel around the ring until they reach the computer that they are addressed to, the signal being refreshed by each node. In a ring topology, the network signal is passed through each network card of each device and passed on to the next device. Each device processes and retransmits the signal, so it is capable of supporting many devices in a somewhat slow but very orderly fashion. There is a very nice feature that everybody gets a chance to send a packet and it is guaranteed that every node gets to send a packet in a finite amount of time.



Advantages

- Broadcasting and multicasting is simple since you just need to send out one message
- Less expensive since less cable footage is required
- It is guaranteed that each host will be able to transmit within a finite time interval
- Very orderly network where every device has access to the token and the opportunity to transmit
- Performs better than a star network under heavy network load

Disadvantages

- Failure of one node brings the whole network down
- Error detection and network administration becomes difficult

- Moves, adds and changes of devices can effect the network
- It is slower than star topology under normal load

Generally, a BUS architecture is preferred over the other topologies - ofcourse, this is a very subjective opinion and the final design depends on the requirements of the network more than anything else. Lately, most networks are shifting towards the STAR topology. Ideally we would like to design networks, which physically resemble the STAR topology, but behave like BUS or RING topology.

Data Link Layer

Data link layer can be characterized by two types of layers:

1. Medium Access Layer (MAL)
2. Logical Link Layer

Aloha Protocols

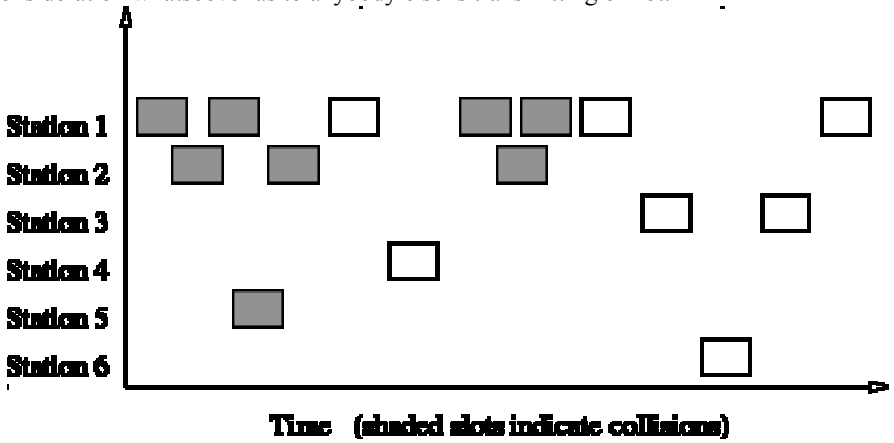
History

The Aloha protocol was designed as part of a project at the University of Hawaii. It provided data transmission between computers on several of the Hawaiian Islands using radio transmissions.

- Communications was typically between remote stations and a central sited named Menehune or vice versa.
- All message to the Menehune were sent using the same frequency.
- When it received a message intact, the Menehune would broadcast an ack on a distinct outgoing frequency.
- The outgoing frequency was also used for messages from the central site to remote computers.
- All stations listened for message on this second frequency.

Pure Aloha

Pure Aloha is an unslotted, fully-decentralized protocol. It is extremely simple and trivial to implement. The ground rule is - "when you want to talk, just talk!". So, a node which wants to transmits, will go ahead and send the packet on its broadcast channel, with no consideration whatsoever as to anybody else is transmitting or not.



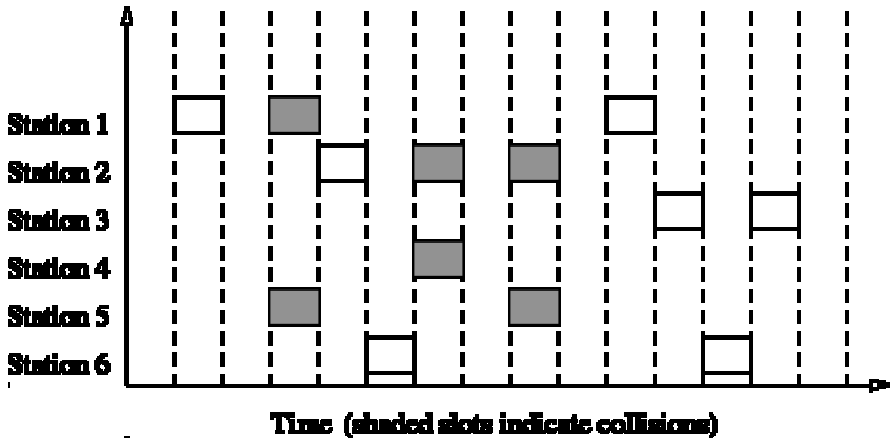
One serious drawback here is that, you dont know whether what you are sending has been received properly or not (so as to say, "whether you've been heard and understood?"). To resolve this, in Pure Aloha, when one node finishes speaking, it expects an acknowledgement in a finite amount of time - otherwise it simply retransmits the data. This scheme works well in small networks where the load is not high. But in large, load intensive networks where many nodes may want to transmit at the same time, this scheme fails miserably. This led to the development of Slotted Aloha.

Slotted Aloha

This is quite similar to Pure Aloha, differing only in the way transmissions take place. Instead of transmitting right at demand time, the sender waits for some time. This delay is specified as follows - the timeline is divided into equal slots and then it is required that transmission should take place only at slot boundaries. To be more precise, the slotted-Aloha makes the following assumptions:

- All frames consist of exactly L bits.
- Time is divided into slots of size L/R seconds (i.e., a slot equals the time to transmit one frame).

- Nodes start to transmit frames only at the beginnings of slots.
- The nodes are synchronized so that each node knows when the slots begin.
- If two or more frames collide in a slot, then all the nodes detect the collision event before the slot ends.



In this way, the number of collisions that can possibly take place is reduced by a huge margin. And hence, the performance become much better compared to Pure Aloha. collisions may only take place with nodes that are ready to speak at the same time. But nevertheless, this is a substantial reduction.

Carrier Sense Mutiple Access Protocols

In both slotted and pure ALOHA, a node's decision to transmit is made independently of the activity of the other nodes attached to the broadcast channel. In particular, a node neither pays attention to whether another node happens to be transmitting when it begins to transmit, nor stops transmitting if another node begins to interfere with its transmission. As humans, we have human protocols that allow allows us to not only behave with more civility, but also to decrease the amount of time spent "colliding" with each other in conversation and consequently increasing the amount of data we exchange in our conversations. Specifically, there are two important rules for polite human conversation:

1. **Listen before speaking:** If someone else is speaking, wait until they are done. In the networking world, this is termed carrier sensing - a node listens to the channel before transmitting. If a frame from another node is currently being transmitted into the channel, a node then waits ("backs off") a random amount of time and then again senses the channel. If the channel is sensed to be idle, the node then begins frame transmission. Otherwise, the node waits another random amount of time and repeats this process.
2. **If someone else begins talking at the same time, stop talking.** In the networking world, this is termed collision detection - a transmitting node listens to the channel while it is transmitting. If it detects that another node is transmitting an interfering frame, it stops transmitting and uses some protocol to determine when it should next attempt to transmit.

It is evident that the end-to-end channel propagation delay of a broadcast channel - the time it takes for a signal to propagate from one of the the channel to another - will play a crucial role in determining its performance. The longer this propagation delay, the larger the chance that a carrier-sensing node is not yet able to sense a transmission that has already begun at another node in the network.

CSMA- Carrier Sense Multiple Access

This is the simplest version CSMA protocol as described above. It does not specify any collision detection or handling. So collisions might and WILL occur and clearly then, this is not a very good protocol for large, load intensive networks.

So, we need an improvement over CSMA - this led to the development of CSMA/CD.

CSMA/CD- CSMA with Collision Detection

In this protocol, while transmitting the data, the sender simultaneously tries to receive it. So, as soon as it detects a collission (it doesn't receive its own data) it stops transmitting. Thereafter, the node waits for some time interval before attempting to transmit again. Simply put, **"listen while you talk"**. But, how long should one wait for the carrier to be freed? There are three schemes to handle this:

1. **1-Persistent:** In this scheme, transmission proceeds immediately if the carrier is idle. However, if the carrier is busy, then sender continues to sense the carrier until it becomes idle. The main problem here is that, if more than one transmitters are ready to send, a collision is GUARANTEED!!

2. **Non-Persistent:** In this scheme, the broadcast channel is not monitored continuously. The sender polls it at random time intervals and transmits whenever the carrier is idle. This decreases the probability of collisions. But, it is not efficient in a low load situation, where number of collisions are anyway small. The problems it entails are:
    - If back-off time is too long, the idle time of carrier is wasted in some sense
    - It may result in long access delays
3. **p-Persistent:** Even if a sender finds the carrier to be idle, it uses a probabilistic distribution to determine whether to transmit or not. Put simply, "toss a coin to decide". If the carrier is idle, then transmission takes place with a probability p and the sender waits with a probability 1-p. This scheme is a good trade off between the Non-persistent and 1-persistent schemes. So, for low load situations, p is high (example: 1-persistent); and for high load situations, p may be lower. Clearly, the value of p plays an important role in determining the performance of this protocol. Also the same p is likely to provide different performance at different loads.

CSMA/CD doesn't work in some wireless scenarios called **"hidden node"** problems. Consider a situation, where there are 3 nodes - A, B and C communicating with each other using a wireless protocol. Morover, B can communicate with both A and C, but A and C lie outside each other's range and hence can't communicate directly with each other. Now, suppose both A and C want to communicate with B simultaneously. They both will sense the carrier to be idle and hence will begin transmission, and even if there is a collision, neither A nor C will ever detect it. B on the other hand will receive 2 packets at the same time and might not be able to understand either of them. To get around this problem, a better version called CSMA/CA was developed, specially for wireless applications.

Image References:

- http://williams.comp.ncat.edu/Networks/multip10.gif
- http://williams.comp.ncat.edu/Networks/multip5.gif
- http://williams.comp.ncat.edu/Networks/multip13.gif
- http://www.e-networks.org/images/startopology.gif
- http://www.delmar.edu/Courses/CIS306/Primer/primf05.gif
- http://www.e-networks.org/images/ringtopology.gif
- http://www.laynetworks.com/images/aloha14.gif
- http://www.laynetworks.com/images/aloha15.gif

CSMA with Collision Avoidance

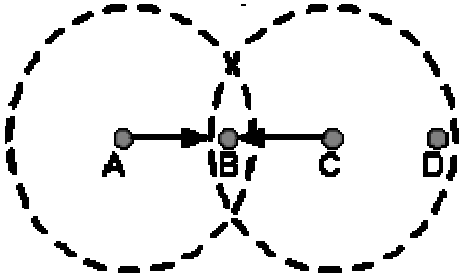
We have observed that CSMA/CD would break down in wireless networks because of hidden node and exposed nodes problems. We will have a quick recap of these two problems through examples.

Hidden Node Problem

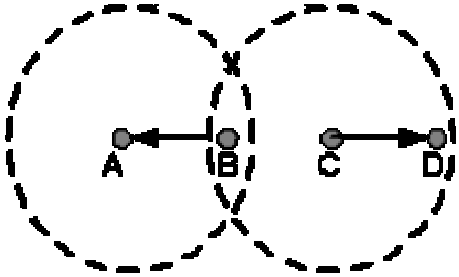
In the case of wireless network it is possible that A is sending a message to B, but C is out of its range and hence while "listening" on the network it will find the network to be free and might try to send packets to B at the same time as A. So, there will be a collision at B. The problem can be looked upon as if A and C are hidden from each other. Hence it is called the "hidden node problem".

Exposed Node Problem

If C is transmitting a message to D and B wants to transmit a message to A, B will find the network to be busy as B hears C trnasmitting. Even if B would have transmitted to A, it would not have been a problem at A or D. CSMA/CD would not allow it to transmit message to A, while the two transmissions could have gone in parallel.



Hidden node problem



Exposed node problem

Addressing hidden node problem (CSMA/CA)

Consider the figure above.Suppose A wants to send a packet to B. Then it will first send a small packet to B called **"Request to Send" (RTS)**. In response, B sends a small packet to A called **"Clear to Send" (CTS)**. Only after A receives a CTS, it transmits the actual data. Now, any of the nodes which can hear either CTS or RTS assume the network to be busy. Hence even if some other node which is out of range of both A and B sends an RTS to C (which can hear at least one of the RTS or CTS between A and B), C would not send a CTS to it and hence the communication would not be established between C and D.

One issue that needs to be addressed is how long the rest of the nodes should wait before they can transmit data over the network. The answer is that the RTS and CTS would carry some information about the size of the data that B intends to transfer. So, they can calculate time that would be required for the transmission to be over and assume the network to be free after that.Another interesting issue is what a node should do if it hears RTS but not a corresponding CTS. One possibility is that it assumes the recipient node has not responded and hence no transmission is going on, but there is a catch in this. It is possible that the node hearing RTS is just on the boundary of the node sending CTS. Hence, it does hear CTS but the signal is so

deteriorated that it fails to recognize it as a CTS. Hence to be on the safer side, a node will not start transmission if it hears either of an RTS or a CTS.

The assumption made in this whole discussion is that if a node X can send packets to a node Y, it can also receive a packet from Y, which is a fair enough assumption given the fact that we are talking of a local network where standard instruments would be used. If that is not the case additional complexities would get introduced in the system.

Does CSMA/CD work universally in the wired networks ?

The problem of range is there in wired networks as well in the form of deterioration of signals. Normally to counter this, we use repeaters, which can regenerate the original signal from a deteriorated one. But does that mean that we can build as long networks as we want with repeaters. The answer, unfortunately, is NO! The reason is the beyond a certain length CSMA/CD will break down.

The mechanism of collision detection which CSMA/CD follows is through listening while talking. What this means is so long as a node is transmitting the packet, it is listening on the cable. If the data it listens to is different from the data it is transmitting it assumes a collision. Once it has stopped transmitting the packet, and has not detected collision while transmission was going on, it assumes that the transmission was successful. The problem arises when the distance between the two nodes is too large. Suppose A wants to transmit some packet to B which is at a very large distance from B. Data can travel on cable only at a finite speed (usually 2/3c, c being the speed of light). So, it is possible that the packet has been transmitted by A onto the cable but the first bit of the packet has not yet reached B. In that case, if a collision occurs, A would be unaware of it occurring. Therefore there is problem in too long a network.

Let us try to parametrize the above problem. Suppose "t" is the time taken for the node A to transmit the packet on the cable and "T" is the time , the packet takes to reach from A to B. Suppose transmission at A starts at time t0. In the worst case the collision takes place just when the first packet is to reach B. Say it is at t0+T-e (e being very small). Then the collision information will take T-e time to propagate back to A. So, at t0+2(T-e) A should still be transmitting. Hence, for the correct detection of collision (ignoring e)

$$t > 2T$$

t increases with the number of bits to be transferred and decreases with the rate of transfer (bits per second). T increases with the distance between the nodes and decreases with the speed of the signal (usually 2/3c). We need to either keep t large enough or T as small. We do not want to live with lower rate of bit transfer and hence slow networks. We can not do anything about the speed of the signal. So what we can rely on is the minimum size of the packet and the distance between the two nodes. Therefore, we fix some minimum size of the packet and if the size is smaller than that, we put in some extra bits to make it reach the minimum size. Accordingly we fix the maximum distance between the nodes. Here too, there is a tradeoff to be made. We do not want the minimum size of the packets to be too large since that wastes lots of resources on cable. At the same time we do not want the distance between the nodes to be too small. Typical minimum packet size is 64 bytes and the corresponding distance is 2-5 kilometers.

Collision Free Protocols

Although collisions do not occur with CSMA/CD once a station has unambigously seized the channel, they can still occur during the contention period. These collisions adversely affect the efficiency of transmission. Hence some protocols have been developed which are contention free.

Bit-Map Method

In this method, there N slots. If node 0 has a frame to send, it transmit a 1 bit during the first slot. No other node is allowed to transmit during this period. Next node 1 gets a chance to transmit 1 bit if it has something to send, regardless of what node 0 had transmitted. This is done for all the nodes. In general node j may declare the fact that it has a frsme to send by inserting a 1 into slot j. Hence after all nodes have passed, each node has complete knowledge of who wants to send a frame. Now they begin transmitting in numerical order. Since everyone knows who is transmitting and when, there could never be any collision.

The basic problem with this protocol is its inefficiency during low load. If a node has to transmit and no other node needs to do so, even then it has to wait for the bitmap to finish. Hence the bitmap will be repeated over and over again if very few nodes want to send wasting valuable bandwidth.

Binary Countdown

In this protocol, a node which wants to signal that it has a frame to send does so by writing its address into the header as a binary number. The arbitration is such that as soon as a node sees that a higher bit position that is 0 in its address has been overwritten with a 1, it gives up. The final result is the address of the node which is allowed to send. After the node has transmitted the whole process is repeated all over again. Given below is an example situation.

Nodes	**Addresses**
A	0010
B	0101
C	1010
D	1001

	1010

Node C having higher priority gets to transmit. The problem with this protocol is that the nodes with higher address always wins. Hence this creates a priority which is highly unfair and hence undesirable.

Limited Contention Protocols

Both the type of protocols described above - Contention based and Contention - free has their own problems. Under conditions of light load, contention is preferable due to its low delay. As the load increases, contention becomes increasingly less attractive, because the overload associated with channel arbitration becomes greater. Just the reverse is true for contention - free protocols. At low load, they have high delay, but as the load increases , the channel efficiency improves rather than getting worse as it does for contention protocols.

Obviously it would be better if one could combine the best properties of the contention and contention - free protocols, that is, protocol which used contention at low loads to provide low delay, but used a cotention-free technique at high load to provide good channel efficiency. Such protocols do exist and are called Limited contention protocols.

It is obvious that the probablity of some station aquiring the channel could only be increased by decreasing the amount of competition. The limited contention protocols do exactly that. They first divide the stations up into (not necessarily disjoint) groups. Only the members of group 0 are permitted to compete for slot 0. The competition for aquiring the slot within a group is contention based. If one of the members of that group succeeds, it aquires the channel and transmits a frame. If there is collision or no node of a particular group wants to send then the members of the next group compete for the next slot. The probablity of a particular node is set to a particular value (optimum).

Adaptive Tree Walk Protocol

The following is the method of adaptive tree protocol. Initially all the nodes are allowed to try to aquire the channel. If it is able to aquire the channel, it sends its frame. If there is collision then the nodes are divided into two equal groups and only one of these groups compete for slot 1. If one of its member aquires the channel then the next slot is reserved for the other group. On the other hand, if there is a collision then that group is again subdivided and the same process is followed. This can be better understood if the nodes are thought of as being organised in a binary tree as shown in the following figure.

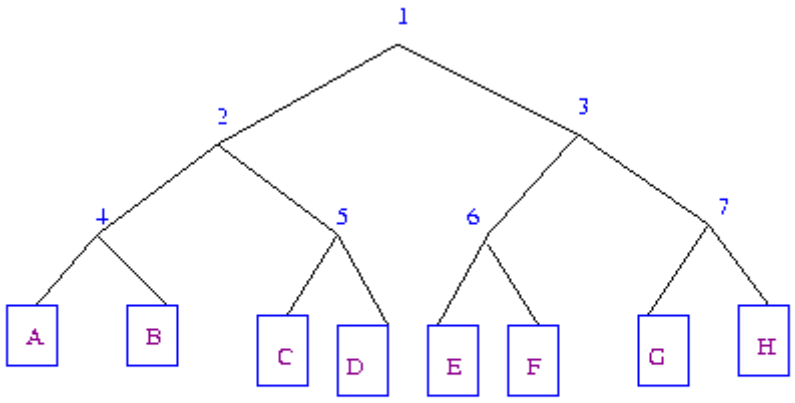


Fig. Adaptive Tree Walk abstraction of nodes in binary tree.

Many improvements could be made to the algorithm. For example, consider the case of nodes G and H being the only ones wanting to transmit. At slot 1 a collision will be detected and so 2 will be tried and it will be found to be idle. Hence it is pointless to probe 3 and one should directly go to 6,7.

Image References:

- http://www.ccs.neu.edu/course/csg150/Solutions-III_files/image002.gif

IEEE 802.3 and Ethernet

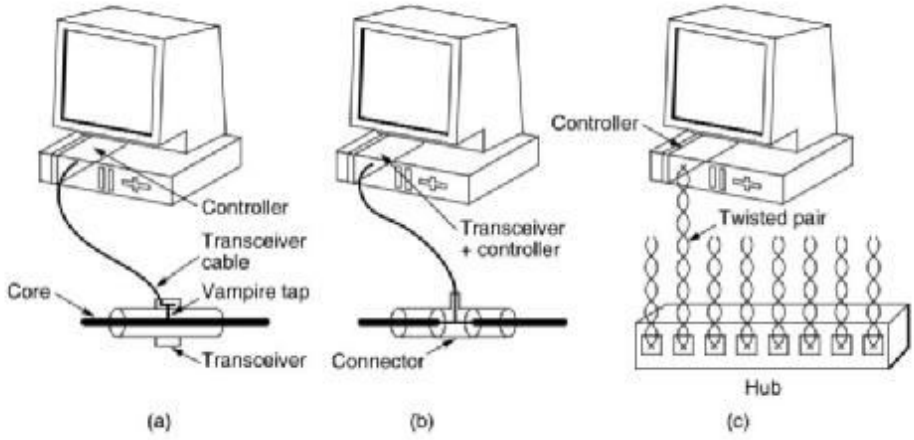
- Very popular LAN standard.
- Ethernet and IEEE 802.3 are distinct standards but as they are very similar to one another these words are used interchangeably.
- A standard for a 1-persistent CSMA/CD LAN.
- It covers the physical layer and MAC sublayer protocol.

Ethernet Physical Layer

A Comparison of Various Ethernet and IEEE 802.3 Physical-Layer Specifications

Characteristic	Ethernet Value	IEEE 802.3 Values					
		10Base5	10Base2	10BaseT	10BaseF	10 Base -TX	100BaseT4
Data rate (Mbps)	10	10	10	10	10	100	100
Signaling method	Baseband	Baseband	Baseband	Baseband	Baseband	Baseband	Baseband
Maximum segment length (m)	500	500	185	100	2,000	100	100
Media	50-ohm coax (thick)	50-ohm coax (thick)	50-ohm coax (thin)	Unshielded twisted-pair cable	Fiber-optic	Cat 5 UTP	Unshielded twisted
Nodes/segment	100	100	30	1024	1024		
Topology	Bus	Bus	Bus	Star	Point-to-point	Bus	Bus

10Base5 means it operates at 10 Mbps, uses baseband signaling and can support segments of up to 500 meters. The 10Base5 cabling is popularly called the Thick Ethernet. Vampire taps are used for their connections where a pin is carefully forced halfway into the co-axial cable's core as shown in the figure below. The 10Base2 or Thin Ethernet bends easily and is connected using standard BNC connectors to form T junctions (shown in the figure below). In the 10Base-T scheme a different kind of wiring pattern is followed in which all stations have a twisted-pair cable running to a central hub (see below). The difference between the different physical connections is shown below:



(a) 10Base5 (b)10Base2 (c)10Base-T

All 802.3 baseband systems use Manchester encoding , which is a way for receivers to unambiguously determine the start, end or middle of each bit without reference to an external clock. There is a restriction on the minimum node spacing (segment length between two nodes) in 10Base5 and 10Base2 and that is 2.5 meter and 0.5 meter respectively. The reason is that if two nodes are closer than the specified limit then there will be very high current which may cause trouble in detection of signal at the receiver end. Connections from station to cable of 10Base5 (i.e. Thick Ethernet) are generally made using vampire taps and to 10Base2 (i.e. Thin Ethernet) are made using industry standard BNC connectors to form T junctions. To allow larger networks, multiple segments can be connected by repeaters as shown. A repeater is a physical layer device. It receives, amplifies and retransmits signals in either direction.

Note: To connect multiple segments, amplifier is not used because amplifier also amplifies the noise in the signal, whereas repeater regenerates signal after removing the noise.

IEEE 802.3 Frame Structure

Preamble (7 bytes)	Start of Frame Delimiter (1 byte)	Dest. Address (2/6 bytes)	Source Address (2/6 bytes)	Length (2 bytes)	802.2 Header+Data (46-1500 bytes)	Frame Checksum (4 bytes)

A brief description of each of the fields

- **Preamble :**Each frame starts with a preamble of 7 bytes, each byte containing the bit pattern 10101010. Manchester encoding is employed here and this enables the receiver's clock to synchronize with the sender's and initialise itself.
- **Start of Frame Delimiter :**This field containing a byte sequence 10101011 denotes the start of the frame itself.
- **Dest. Address :**The standard allows 2-byte and 6-byte addresses. Note that the 2-byte addresses are always local addresses while the 6-byte ones can be local or global.

2-Byte Address - Manually assigned address

Individual(0)/Group(1) (1 bit)	Address of the machine (15 bits)

6-Byte Address - Every Ethernet card with globally unique address

Individual(0)/Group(1) (1 bit)	Universal(0)/Local(1) (1 bit)	Address of the machine (46 bits)

Multicast : Sending to group of stations. This is ensured by setting the first bit in either 2-byte/6-byte addresses to 1.
Broadcast : Sending to all stations. This can be done by setting all bits in the address field to 1.All Ethernet cards(Nodes) are a member of this group.

- **Source Address :**Refer to Dest. Address. Same holds true over here.
- **Length :** The Length field tells how many bytes are present in the data field, from a minimum of 0 to a maximum of 1500. The Data and padding together can be from 46bytes to 1500 bytes as the valid frames must be at least 64 bytes long, thus if data is less than 46 bytes the amount of padding can be found out by length field.
- **Data :**Actually this field can be split up into two parts - Data(0-1500 bytes) and Padding(0-46 bytes).
  Reasons for having a minimum length frame :
  1. To prevent a station from completing the transmission of a short frame before the first bit has even reached the far end of the cable, where it may collide with another frame. Note that the transmission time ought to be greater than twice the propagation time between two farthest nodes.
     transmission time for frame > 2*propagation time between two farthest nodes
  2. When a transceiver detects a collision, it truncates the current frame, which implies that stray bits and pieces of frames appear on the cable all the time. Hence to distinguish between valid frames from garbage, 802.3 states that the minimum length of valid frames ought to be 64 bytes (from Dest. Address to Frame Checksum).
- **Frame Checksum :** It is a 32-bit hash code of the data. If some bits are erroneously received by the destination (due to noise on the cable), the checksum computed by the destination wouldn't match with the checksum sent and therefore the error will be detected. The checksum algorithm is a cyclic redundancy checksum (CRC) kind. The checksum includes the packet from Dest. Address to Data field.

Ethernet Frame Structure

Preamble (8 bytes)	Dest. Address (2/6 bytes)	Source Address (2/6 bytes)	Type (2 bytes)	Data (46-1500 bytes)	Frame Checksum (4 bytes)

A brief description of the fields which differ from IEEE 802.3

- **Preamble :**The *Preamble* and *Start of Frame Delimiter* are merged into one in Ethernet standard. However, the contents of the first 8 bytes remains the same in both.
- **Type :**The length field of IEEE 802.3 is replaced by Type field, which denotes the type of packet being sent viz. IP, ARP, RARP, etc. If the field indicates a value less than 1500 bytes then it is length field of 802.3 else it is the type field of Ethernet packet.

Truncated Binary Exponential Back off

In case of collision the node transmitting backs off by a random number of slots , each slot time being equal to transmission time of 512 bits (64 Byte- minimum size of a packet) in the following fashion:

No of Collision	Random No of slots
1st	0-1
2nd	0-3
3rd	0-7
|	|
|	|
10th	0-1023
---	---
11th	0-1023
12th	0-1023
|	|
16th	0-1023

In general after i collisions a random number between 0-2^i-1 is chosen , and that number of slots is skipped. However, after 10 collisions have been reached the randomization interval is frozen at maximum of 1023 slots. After 16 collisions the controller reports failure back to the computer.

5-4-3 Rule

Each version of 802.3 has a maximum cable length per segment because long propagation time leads to difficulty in collision detection. To compensate for this the transmission time has to be increased which can be achieved by slowing down the transmission rate or increasing the packet size, neither of which is desirable. Hence to allow for large networks, multiple cables are connected via **repeaters**. Between any two nodes on an Ethernet network, there can be at most five segments, four repeaters and three populated segments (non-populated segments are those which do not have any machine connected between the two repeaters). This is known as the **5-4-3 Rule**.

Image References:

- http://homepages.ius.edu/rwisman/b438/Html/4-14.jpg

Data Link Layer

What is DLL(Data Link Layer)

The Data Link Layer is the second layer in the OSI model, above the Physical Layer, which ensures that the error free data is transferred between the adjacent nodes in the network. It breaks the datagrams passed down by above layers and convert them into frames ready for transfer. This is called Framing. It provides two main functionalities

- Reliable data transfer service between two peer network layers
- Flow Control mechanism which regulates the flow of frames such that data congestion is not there at slow receivers due to fast senders.

What is Framing?

Since the physical layer merely accepts and transmits a stream of bits without any regard to meaning or structure, it is upto the data link layer to create and recognize frame boundaries. This can be accomplished by attaching special bit patterns to the beginning and end of the frame. If these bit patterns can accidentally occur in data, special care must be taken to make sure these patterns are not incorrectly interpreted as frame delimiters. The four framing methods that are widely used are

- Character count
- Starting and ending characters, with character stuffing
- Starting and ending flags, with bit stuffing
- Physical layer coding violations

Character Count

This method uses a field in the header to specify the number of characters in the frame. When the data link layer at the destination sees the character count,it knows how many characters follow, and hence where the end of the frame is. The disadvantage is that if the count is garbled by a transmission error, the destination will lose synchronization and will be unable to locate the start of the next frame. So, this method is rarely used.

Character stuffing

In the second method, each frame starts with the ASCII character sequence DLE STX and ends with the sequence DLE ETX.(where DLE is Data Link Escape, STX is Start of TeXt and ETX is End of TeXt.) This method overcomes the drawbacks of the character count method. If the destination ever loses synchronization, it only has to look for DLE STX and DLE ETX characters. If however, binary data is being transmitted then there exists a possibility of the characters DLE STX and DLE ETX occurring in the data. Since this can interfere with the framing, a technique called character stuffing is used. The sender's data link layer inserts an ASCII DLE character just before the DLE character in the data. The receiver's data link layer removes this DLE before this data is given to the network layer. However character stuffing is closely associated with 8-bit characters and this is a major hurdle in transmitting arbitrary sized characters.

Bit stuffing

The third method allows data frames to contain an arbitrary number of bits and allows character codes with an arbitrary number of bits per character. At the start and end of each frame is a flag

byte consisting of the special bit pattern 01111110 . Whenever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a zero bit into the outgoing bit stream. This technique is called bit stuffing. When the receiver sees five consecutive 1s in the incoming data stream, followed by a zero bit, it automatically destuffs the 0 bit. The boundary between two frames can be determined by locating the flag pattern.

Physical layer coding violations

The final framing method is physical layer coding violations and is applicable to networks in which the encoding on the physical medium contains some redundancy. In such cases normally, a 1 bit is a high-low pair and a 0 bit is a low-high pair. The combinations of low-low and high-high which are not used for data may be used for marking frame boundaries.

Error Control

The bit stream transmitted by the physical layer is not guaranteed to be error free. The data link layer is responsible for error detection and correction. The most common error control method is to compute and append some form of a checksum to each outgoing frame at the sender's data link layer and to recompute the checksum and verify it with the received checksum at the receiver's side. If both of them match, then the frame is correctly received; else it is erroneous. The checksums may be of two types:
Error detecting : Receiver can only detect the error in the frame and inform the sender about it. # Error detecting and correcting : The receiver can not only detect the error but also correct it. Examples of Error Detecting methods:

- **Parity bit:**
  Simple example of error detection technique is parity bit. The parity bit is chosen that the number of 1 bits in the code word is either even(for even parity) or odd (for odd parity). For example when 10110101 is transmitted then for even parity an 1 will be appended to the data and for odd parity a 0 will be appended. This scheme can detect only single bits. So if two or more bits are changed then that can not be detected.
- **Longitudinal Redundancy Checksum:**
  Longitudinal Redundancy Checksum is an error detecting scheme which overcomes the problem of two erroneous bits. In this conceptof parity bit is used but with slightly more intelligence. With each byte we send one parity bit then send one additional byte which have the parity corresponding to the each bit position of the sent bytes. So the parity bit is set in both horizontal and vertical direction. If one bit get flipped we can tell which row and column have error then we find the intersection of the two and determine the erroneous bit. If 2 bits are in error and they are in the different column and row then they can be detected. If the error are in the same column then the row will differentiate and vice versa. Parity can detect the only odd number of errors. If they are even and distributed in a fashion that in all direction then LRC may not be able to find the error.
- **Cyclic Redundancy Checksum (CRC):**
  We have an n-bit message. The sender adds a k-bit Frame Check Sequence (FCS) to this message before sending. The resulting (n+k) bit message is divisible by some (k+1) bit number. The receiver divides the message ((n+k)-bit) by the same (k+1)-bit number and if there is no remainder, assumes that there was no error. How do we choose this number? For example, if k=12 then 1000000000000 (13-bit number) can be chosen, but this is a pretty crappy choice. Because it will result in a zero remainder for all (n+k) bit messages with the last 12 bits zero. Thus, any bits flipping beyond the last 12 go undetected. If k=12, and we take 1110001000110 as the 13-bit number (incidentally, in decimal representation this turns out to be 7238). This will be unable to detect errors only if the corrupt message

and original message have a difference of a multiple of 7238. The probablilty of this is low, much lower than the probability that anything beyond the last 12-bits flips. In practice, this number is chosen after analyzing common network transmission errors and then selecting a number which is likely to detect these common errors.

How to detect source errors?

In order ensure that the frames are delivered correctly, the receiver should inform the sender about incoming frames using positive or negative acknowledgements. On the sender's side the receipt of a positive acknowledgement implies that the frame has arrived at the destination safely while the receipt of a negative acknowledgement means that an error has occurred in the frame and it needs to be retransmitted. However, this scheme is too simplistic because if a noise burst causes the frame to vanish completely, the receiver will not respond at all and the sender would hang forever waiting for an acknowledgement. To overcome this drawback, timers are introduced into the data link layer. When the sender transmits a frame it also simultaneously starts a timer. The timer is set to go off after a interval long enough for the frame to reach the destination, be processed there, and have the acknowledgement propogate back to the sender. If the frame is received correctly the positive acknowledgment arrives before the timer runs out and so the timer is canceled. If however either the frame or the acknowledgement is lost the timer will go off and the sender may retransmit the frame. Since multiple transmission of frames can cause the receiver to accept the same frame and pass it to the network layer more than once, sequence numbers are generally assigned to the outgoing frames.
The types of acknowledgements that are sent can be classified as follows:

- Cumulative acknowledgements: A single acknowledgement informing the sender that all the frames upto a certain number have been received.
- Selective acknowledgements: Acknowledgement for a particular frame.

They may be also classified as:

- Individual acknowledgements: Individual acknowledgement for each frame.
- Group acknowledgements: A bit-map that specifies the acknowledgements of a range of frame numbers.

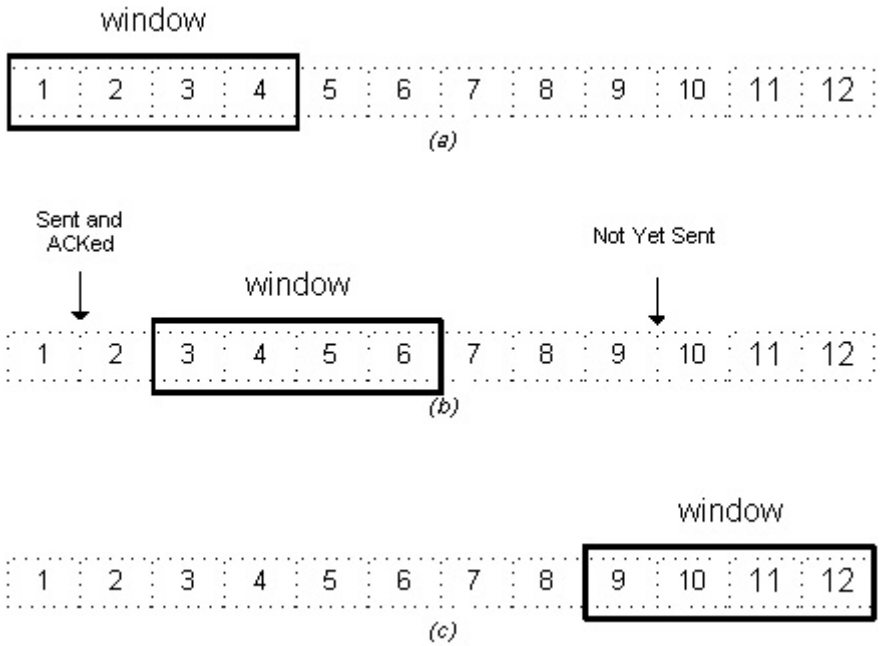
Flow Control

Consider a situation in which the sender transmits frames faster than the receiver can accept them. If the sender keeps pumping out frames at high rate, at some point the receiver will be completely swamped and will start losing some frames. This problem may be solved by introducing flow control. Most flow control protocols contain a feedback mechanism to inform the sender when it should transmit the next frame.

Mechanisms For Flow Control:

- **Stop and Wait Protocol:** This is the simplest file control protocol in which the sender transmits a frame and then waits for an acknowledgement, either positive or negative, from the receiver before proceeding. If a positive acknowledgement is received, the sender transmits the next packet; else it retransmits the same frame. However, this protocol has one major flaw in it. If a packet or an acknowledgement is completely destroyed in transit due to a noise burst, a deadlock will occur because the sender cannot proceed until it receives an acknowledgement. This problem may be solved using timers on the sender's side. When the frame is transmitted, the timer is set. If there is no response from the receiver within a certain time interval, the timer goes off and the frame may be retransmitted.

- **Sliding Window Protocols:** Inspite of the use of timers, the stop and wait protocol still suffers from a few drawbacks. Firstly, if the receiver had the capacity to accept more than one frame, its resources are being underutilized. Secondly, if the receiver was busy and did not wish to receive any more packets, it may delay the acknowledgement. However, the timer on the sender's side may go off and cause an unnecessary retransmission. These drawbacks are overcome by the sliding window protocols.
  In sliding window protocols the sender's data link layer maintains a 'sending window' which consists of a set of sequence numbers corresponding to the frames it is permitted to send. Similarly, the receiver maintains a 'receiving window' corresponding to the set of frames it is permitted to accept. The window size is dependent on the retransmission policy and it may differ in values for the receiver's and the sender's window. The sequence numbers within the sender's window represent the frames sent but as yet not acknowledged. Whenever a new packet arrives from the network layer, the upper edge of the window is advanced by one. When an acknowledgement arrives from the receiver the lower edge is advanced by one. The receiver's window corresponds to the frames that the receiver's data link layer may accept. When a frame with sequence number 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. If however, a frame falling outside the window is received, the receiver's data link layer has two options. It may either discard this frame and all subsequent frames until the desired frame is received or it may accept these frames and buffer them until the appropriate frame is received and then pass the frames to the network layer in sequence.



  In this simple example, there is a 4-byte sliding window. Moving from left to right, the window "slides" as bytes in the stream are sent and acknowledged.
  Most sliding window protocols also employ ARQ (Automatic Repeat reQuest) mechanism. In ARQ, the sender waits for a positive acknowledgement before proceeding to the next frame. If no acknowledgement is received within a certain time interval it retransmits the frame. ARQ is of two types :

  1. **Go Back 'n':** If a frame is lost or received in error, the receiver may simply discard all subsequent frames, sending no acknowledgments for the discarded frames. In this case the receive window is of size 1. Since no acknowledgements are being received the sender's window will fill up, the sender will eventually time out and retransmit all the unacknowledged frames in order starting from the damaged or lost frame. The

maximum window size for this protocol can be obtained as follows. Assume that the window size of the sender is n. So the window will initially contain the frames with sequence numbers from 0 to (w-1). Consider that the sender transmits all these frames and the receiver's data link layer receives all of them correctly. However, the sender's data link layer does not receive any acknowledgements as all of them are lost. So the sender will retransmit all the frames after its timer goes off. However the receiver window has already advanced to w. Hence to avoid overlap , the sum of the two windows should be less than the sequence number space.

w-1 + 1 < Sequence Number Space
i.e., w < Sequence Number Space
Maximum Window Size = Sequence Number Space - 1

2. **Selective Repeat:**In this protocol rather than discard all the subsequent frames following a damaged or lost frame, the receiver's data link layer simply stores them in buffers. When the sender does not receive an acknowledgement for the first frame it's timer goes off after a certain time interval and it retransmits only the lost frame. Assuming error - free transmission this time, the sender's data link layer will have a sequence of a many correct frames which it can hand over to the network layer. Thus there is less overhead in retransmission than in the case of Go Back n protocol.
In case of selective repeat protocol the window size may be calculated as follows. Assume that the size of both the sender's and the receiver's window is w. So initially both of them contain the values 0 to (w-1). Consider that sender's data link layer transmits all the w frames, the receiver's data link layer receives them correctly and sends acknowledgements for each of them. However, all the acknowledgemnets are lost and the sender does not advance it's window. The receiver window at this point contains the values w to (2w-1). To avoid overlap when the sender's data link layer retransmits, we must have the sum of these two windows less than sequence number space. Hence, we get the condition

Maximum Window Size = Sequence Number Space / 2

Image References:

- http://condor.depaul.edu/~jkristof/technotes/sliding-window.jpg
- http://www.eas.asu.edu/trace/eee459_sp02/applet/archana/gupta5463.html

Network Layer

What is Network Layer?

The network layer is concerned with getting packets from the source all the way to the destination. The packets may require to make many hops at the intermediate routers while reaching the destination. This is the lowest layer that deals with end to end transmission. In order to achieve its goals, the network layer must know about the topology of the communication network. It must also take care to choose routes to avoid overloading of some of the communication lines while leaving others idle. The network layer-transport layer interface frequently is the interface between the carrier and the customer, that is the boundary of the subnet. The functions of this layer include :

1. Routing - The process of transferring packets received from the Data Link Layer of the source network to the Data Link Layer of the correct destination network is called routing. Involves decision making at each intermediate node on where to send the packet next so that it eventually reaches its destination. The node which makes this choice is called a router. For routing we require some mode of addressing which is recognized by the Network Layer. This addressing is different from the MAC layer addressing.
2. Inter-networking - The network layer is the same across all physical networks (such as Token-Ring and Ethernet). Thus, if two physically different networks have to communicate, the packets that arrive at the Data Link Layer of the node which connects these two physically different networks, would be stripped of their headers and passed to the Network Layer. The network layer would then pass this data to the Data Link Layer of the other physical network..
3. Congestion Control - If the incoming rate of the packets arriving at any router is more than the outgoing rate, then congestion is said to occur. Congestion may be caused by many factors. If suddenly, packets begin arriving on many input lines and all need the same output line, then a queue will build up. If there is insufficient memory to hold all of them, packets will be lost. But even if routers have an infinite amount of memory, congestion gets worse, because by the time packets reach to the front of the queue, they have already timed out (repeatedly), and duplicates have been sent. All these packets are dutifully forwarded to the next router, increasing the load all the way to the destination. Another reason for congestion are slow processors. If the router's CPUs are slow at performing the bookkeeping tasks required of them, queues can build up, even though there is excess line capacity. Similarly, low-bandwidth lines can also cause congestion.

We will now look at these function one by one.

Addressing Scheme
IP addresses are of 4 bytes and consist of :
i) The network address, followed by
ii) The host address
The first part identifies a network on which the host resides and the second part identifies the particular host on the given network. Some nodes which have more than one interface to a network must be assigned separate internet addresses for each interface. This multi-layer addressing makes it easier to find and deliver data to the destination. A fixed size for each of these would lead to wastage or under-usage that is either there will be too many network addresses and few hosts in each (which causes problems for routers who route based on the network address) or there will be very few network addresses and lots of hosts (which will be a waste for small network requirements). Thus, we do away with any notion of fixed sizes for the network and host addresses. We classify networks as follows:

1. **Large Networks :** 8-bit network address and 24-bit host address. There are approximately 16 million hosts per network and a maximum of 126 (2^7 - 2) Class A networks can be defined. The calculation requires that 2 be subtracted because 0.0.0.0 is reserved for use as the default route and 127.0.0.0 be reserved for the loop back function. Moreover each Class A network can support a maximum of 16,777,214 (2^24 - 2) hosts per network. The host calculation requires that 2 be subtracted because all 0's are reserved to identify the network itself and all 1s are reserved for broadcast addresses. The reserved numbers may not be assigned to individual hosts.
2. **Medium Networks :** 16-bit network address and 16-bit host address. There are approximately 65000 hosts per network and a maximum of 16,384 (2^14) Class B networks can be defined with up to (2^16-2) hosts per network.

3. **Small networks :** 24-bit network address and 8-bit host address. There are approximately 250 hosts per network.

You might think that Large and Medium networks are sort of a waste as few corporations/organizations are large enough to have 65000 different hosts. (By the way, there are very few corporations in the world with even close to 65000 employees, and even in these corporations it is highly unlikely that each employee has his/her own computer connected to the network.) Well, if you think so, you're right. This decision seems to have been a mistak

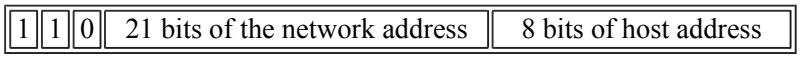
Address Classes

The IP specifications divide addresses into the following classes :

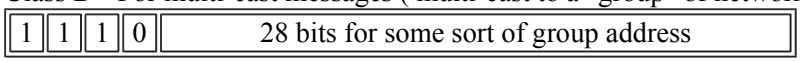
- Class A - For large networks

- Class B - For medium networks

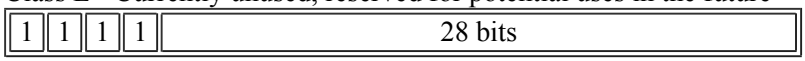
- Class C - For small networks



- Class D - For multi-cast messages (multi-cast to a "group" of networks)



- Class E - Currently unused, reserved for potential uses in the future



Internet Protocol

Special Addresses : There are some special IP addresses :

1. Broadcast Addresses They are of two types :
   (i) Limited Broadcast : It consists of all 1's, i.e., the address is 255.255.255.255 . It is used only on the LAN, and not for any external network.
   (ii) Directed Broadcast : It consists of the network number + all other bits as1's. It reaches the router corresponding to the network number, and from there it broadcasts to all the nodes in the network. This method is a major security problem, and is not used anymore. So now if we find that all the bits are 1 in the host no. field, then the packet is simply dropped. Therefore, now we can only do broadcast in our own network using Limited Broadcast.
2. Network ID = 0
   It means we are referring to this network and for local broadcast we make the host ID zero.
3. Host ID = 0
   This is used to refer to the entire network in the routing table.
4. Loop-back Address
   Here we have addresses of the type 127.x.y.z It goes down way upto the IP layer and comes back to the application layer on the same host. This is used to test network applications before they are used commercially.

Subnetting
Sub netting means organizing hierarchies within the network by dividing the host ID as per our network. For example consider the network ID : 150.29.x.y
We could organize the remaining 16 bits in any way, like :
4 bits - department
4 bits - LAN

8 bits - host
This gives some structure to the host IDs. This division is not visible to the outside world. They still see just the network number, and host number (as a whole). The network will have an internal routing table which stores information about which router to send an address to. Now consider the case where we have : 8 bits - subnet number, and 8 bits - host number. Each router on the network must know about all subnet numbers. This is called the subnet mask. We put the network number and subnet number bits as 1 and the host bits as 0. Therefore, in this example the subnet mask becomes : 255.255.255.0 . The hosts also need to know the subnet mask when they send a packet. To find if two addresses are on the same subnet, we can AND source address with subnet mask, and destination address with with subnet mask, and see if the two results are the same. The basic reason for sub netting was avoiding broadcast. But if at the lower level, our switches are smart enough to send directed messages, then we do not need sub netting. However, sub netting has some security related advantages.

Supernetting
This is moving towards class-less addressing. We could say that the network number is 21 bits (for 8 class C networks) or say that it is 24 bits and 7 numbers following that. For example : a.b.c.d / 21 This means only look at the first 21 bits as the network address.

Addressing on IITK Network
If we do not have connection with the outside world directly then we could have Private IP addresses (172.31) which are not to be publicised and routed to the outside world. Switches will make sure that they do not broadcast packets with such addressed to the outside world. The basic reason for implementing subnetting was to avoid broadcast. So in our case we can have some subnets for security and other reasons although if the switches could do the routing properly, then we do not need subnets. In the IITK network we have three subnets -CC, CSE building are two subnets and the rest of the campus is one subset

Packet Structure

<table>
<tr><td>Version Number (4 bits)</td><td>Header Length (4 bits)</td><td>Type of Service (8 bits)</td><td colspan="2">Total Length (16 bits)</td></tr>
<tr><td colspan="3">ID (16 bits)</td><td>Flags (3bits)</td><td>Flag Offset (13 bits)</td></tr>
<tr><td colspan="2">Time To Live (8 bits)</td><td>Protocol (8 bits)</td><td colspan="2">Header Checksum (16 bits)</td></tr>
<tr><td colspan="5">Source (32 bits)</td></tr>
<tr><td colspan="5">Destination (32 bits)</td></tr>
<tr><td colspan="5">Options</td></tr>
</table>

Version Number : The current version is Version 4 (0100).

1. **Header Length :** We could have multiple sized headers so we need this field. Header will always be a multiple of 4bytes and so we can have a maximum length of the field as 15, so the maximum size of the header is 60 bytes (20 bytes are mandatory).
2. **Type Of Service (ToS) :** This helps the router in taking the right routing decisions. The structure is :
   First three bits : They specify the precedences i.e. the priority of the packets.
   Next three bits :
   D bit - D stands for delay. If the D bit is set to 1, then this means that the application is delay sensitive, so we should try to route the packet with minimum delay.
   - T bit - T stands for throughput. This tells us that this particular operation is throughput sensitive.
   - R bit - R stands for reliability. This tells us that we should route this packet through a more reliable network.

   Last two bits: The last two bits are never used. Unfortunately, no router in this world looks at these bits and so no application sets them nowadays. The second word is meant for handling fragmentations. If a link cannot transmit large packets, then we fragment the packet and put sufficient information in the header for recollection at the destination.
3. **ID Field :** The source and ID field together will represent the fragments of a unique packet. So each fragment will have a different ID.

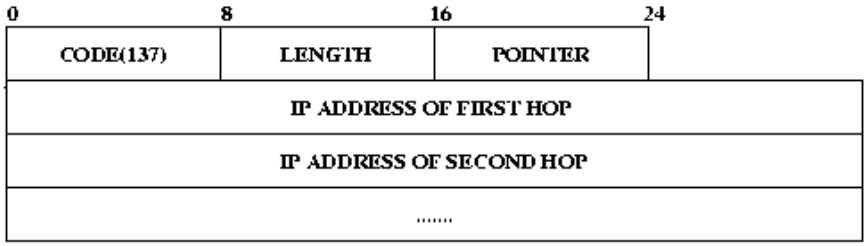
4. **Offset :** It is a 13 bit field that represents where in the packet, the current fragment starts. Each bit represents 8 bytes of the packet. So the packet size can be at most 64 kB. Every fragment except the last one must have its size in bytes as a multiple of 8 in order to ensure compliance with this structure. The reason why the position of a fragment is given as an offset value instead of simply numbering each packet is because refragmentation may occur somewhere on the path to the other node. Fragmentation, though supported by IPv4 is not encouraged. This is because if even one fragment is lost the entire packet needs to be discarded. A quantity M.T.U (Maximum Transmission Unit) is defined for each link in the route. It is the size of the largest packet that can be handled by the link. The Path-M.T.U is then defined as the size of the largest packet that can be handled by the path. It is the smallest of all the MTUs along the path. Given information about the path MTU we can send packets with sizes smaller than the path MTU and thus prevent fragmentation. This will not completely prevent it because routing tables may change leading to a change in the path.
5. **Flags :**It has three bits -
    - M bit : If M is one, then there are more fragments on the way and if M is 0, then it is the last fragment
    - DF bit : If this bit is sent to 1, then we should not fragment such a packet.
    - Reserved bit : This bit is not used.

   Reassembly can be done only at the destination and not at any intermediate node. This is because we are considering Datagram Service and so it is not guaranteed that all the fragments of the packet will be sent thorough the node at which we wish to do reassembly.
6. **Total Length :** It includes the IP header and everything that comes after it.
7. **Time To Live (TTL) :** Using this field, we can set the time within which the packet should be delivered or else destroyed. It is strictly treated as the number of hops. The packet should reach the destination in this number of hops. Every router decreases the value as the packet goes through it and if this value becomes zero at a particular router, it can be destroyed.
8. **Protocol :** This specifies the module to which we should hand over the packet (UDP or TCP). It is the next encapsulated protocol.

Value	Protocol
0	Pv6 Hop-by-Hop Option.
1	ICMP, Internet Control Message Protocol.
2	IGMP, Internet Group Management Protocol. RGMP, Router-port Group Management Protocol.
3	GGP, Gateway to Gateway Protocol.
4	IP in IP encapsulation.
5	ST, Internet Stream Protocol.
6	TCP, Transmission Control Protocol.
7	UCL, CBT.
8	EGP, Exterior Gateway Protocol.
9	IGRP.
10	BBN RCC Monitoring.
11	NVP, Network Voice Protocol.
12	PUP.
13	ARGUS.
14	EMCON, Emission Control Protocol.
15	XNET, Cross Net Debugger.
16	Chaos.
17	UDP, User Datagram Protocol.
18	TMux, Transport Multiplexing Protocol.
19	DCN Measurement Subsystems.
-	
-	
255	

9. **Header Checksum :** This is the usual checksum field used to detect errors. Since the TTL field is changing at every router so the header checksum (upto the options field) is checked and recalculated at every router.
10. **Source :** It is the IP address of the source node
11. **Destination :** It is the IP address of the destination node.

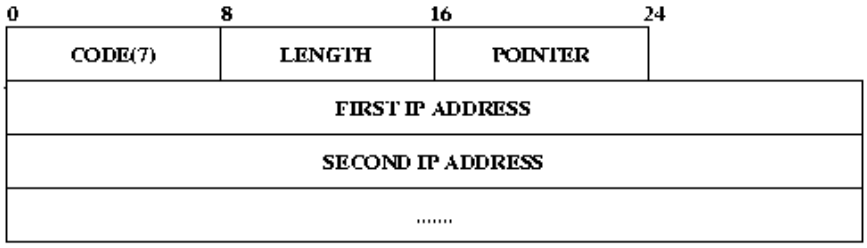
12. **IP Options :** The options field was created in order to allow features to be added into IP as time passes and requirements change. Currently 5 options are specified although not all routers support them. They are:
    - **Securtiy:** It tells us how secret the information is. In theory a military router might use this field to specify not to route through certain routers. In practice no routers support this field.
    - **Source Routing:** It is used when we want the source to dictate how the packet traverses the network. It is of 2 types
      -> Loose Source Record Routing (LSRR): It requires that the packet traverse a list of specified routers, in the order specified but the packet may pass though some other routers as well.
      -> Strict Source Record Routing (SSRR): It requires that the packet traverse only the set of specified routers and nothing else. If it is not possible, the packet is dropped with an error message sent to the host.



The format of Source Route options in an IP Datagram

The above is the format for SSRR. For LSRR the code is 131.

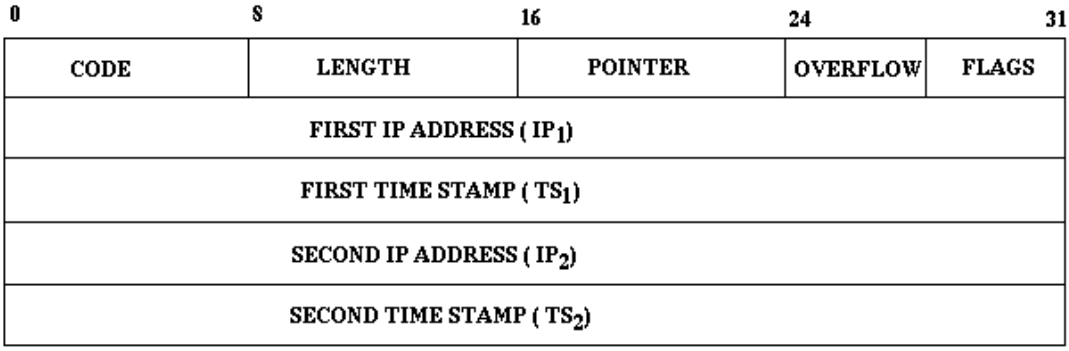
- **Record Routing :**



Format of the Record Route option in an IP Datagram

In this the intermediate routers put there IP addresses in the header, so that the destination knows the entire path of the packet. Space for storing the IP address is specified by the source itself. The pointer field points to the position where the next IP address has to be written. Length field gives the number of bytes reserved by the source for writing the IP addresses. If the space provided for storing the IP addresses of the routers visited, falls short while storing these addresses, then the subsequent routers do not write their IP addresses.

- **Time Stamp Routing :**



Format Of Timestamp Option

It is similar to record route option except that nodes also add their timestamps to the packet. The new fields in this option are

-> Flags: It can have the following values

- 0- Enter only timestamp.

- 1- The nodes should enter Timestamp as well as their IP.
- 3 - The source specifies the IPs that should enter their timestamp. A special point of interest is that only if the IP is the same as that at the pointer then the time is entered. Thus if the source specifies IP1 and IP2 but IP2 is first in the path then the field IP2 is left empty, even after having reached IP2 but before reaching IP1.

-> Overflow: It stores the number of nodes that were unable to add their timestamps to the packet. The maximum value is 15.

◦ **Format of the type/code field**

Copy Bit	Type of option	Option Number.

- **Copy bit:** It says whether the option is to be copied to every fragment or not. a value of 1 stands for copying and 0 stands for not copying.
- **Type:** It is a 2 bit field. Currently specified values are 0 and 2. 0 means the option is a control option while 2 means the option is for measurement
- **Option Number:** It is a 5 bit field which specifies the option number.

For all options a length field is put in order that a router not familiar with the option will know how many bytes to skip. Thus every option is of the form

◦ **TLV: Type/Length/Value.** This format is followed in not only in IP but in nearly all major protocols.

Network Layer (Continued...)

The network layer is concerned with getting packets from the source all the way to the destnation. The packets may require to make many hops at the intermediate routers while reaching the destination. This is the lowest layer that deals with end to end transmission. In order to achieve its goals, the network later must know about the topology of the communication network. It must also take care to choose routes to avoid overloading of some of the communication lines while leaving others idle. The main functions performed by the network layer are as follows:

- Routing
- Congestion Control
- Internetwokring

Routing

Routing is the process of forwarding of a packet in a network so that it reaches its intended destination. The main goals of routing are:

1. **Correctness:** The routing should be done properly and correctly so that the packets may reach their proper destination.
2. **Simplicity:** The routing should be done in a simple manner so that the overhead is as low as possible. With increasing complexity of the routing algorithms the overhead also increases.
3. **Robustness:** Once a major network becomes operative, it may be expected to run continuously for years without any failures. The algorithms designed for routing should be robust enough to handle hardware and software failures and should be able to cope with changes in the topology and traffic without requiring all jobs in all hosts to be aborted and the network rebooted every time some router goes down.
4. **Stability:** The routing algorithms should be stable under all possible circumstances.
5. **Fairness:** Every node connected to the network should get a fair chance of transmitting their packets. This is generally done on a first come first serve basis.
6. **Optimality:** The routing algorithms should be optimal in terms of throughput and minimizing mean packet delays. Here there is a trade-off and one has to choose depending on his suitability.

Classification of Routing Algorithms

The routing algorithms may be classified as follows:

1. **Adaptive Routing Algorithm:** These algorithms change their routing decisions to reflect changes in the topology and in traffic as well. These get their routing information from adjacent routers or from all routers. The optimization parameters are the distance, number of hops and estimated transit time. This can be further classified as follows:
    1. **Centralized:** In this type some central node in the network gets entire information about the network topology, about the traffic and about other nodes. This then transmits this information to the respective routers. The advantage of this is that only one node is required to keep the information. The disadvantage is that if the central node goes down the entire network is down, i.e. single point of failure.
    2. **Isolated:** In this method the node decides the routing without seeking information from other nodes. The sending node does not know about the status of a particular link. The disadvantage is that the packet may be send through a congested route resulting in a delay. Some examples of this type of algorithm for routing are:

- **Hot Potato:** When a packet comes to a node, it tries to get rid of it as fast as it can, by putting it on the shortest output queue without regard to where that link leads. A variation of this algorithm is to combine static routing with the hot potato algorithm. When a packet arrives, the routing algorithm takes into account both the static weights of the links and the queue lengths.
- **Backward Learning:** In this method the routing tables at each node gets modified by information from the incoming packets. One way to implement backward learning is to include the identity of the source node in each packet, together with a hop counter that is incremented on each hop. When a node receives a packet in a particular line, it notes down the number of hops it has taken to reach it from the source node. If the previous value of hop count stored in the node is better than the current one then nothing is done but if the current value is better then the value is updated for future use. The problem with this is that when the best route goes down then it cannot recall the second best route to a particular node. Hence all the nodes have to forget the stored informations periodically and start all over again.

3. **Distributed:** In this the node receives information from its neighbouring nodes and then takes the decision about which way to send the packet. The disadvantage is that if in between the the interval it receives information and sends the paket something changes then the packet may be delayed.

2. **Non-Adaptive Routing Algorithm:** These algorithms do not base their routing decisions on measurements and estimates of the current traffic and topology. Instead the route to be taken in going from one node to the other is computed in advance, off-line, and downloaded to the routers when the network is booted. This is also known as static routing. This can be further classified as:
    1. **Flooding:** Flooding adapts the technique in which every incoming packet is sent on every outgoing line except the one on which it arrived. One problem with this method is that packets may go in a loop. As a result of this a node may receive several copies of a particular packet which is undesirable. Some techniques adapted to overcome these problems are as follows:
        - **Sequence Numbers:** Every packet is given a sequence number. When a node receives the packet it sees its source address and sequence number. If the node finds that it has sent the same packet earlier then it will not transmit the packet and will just discard it.
        - **Hop Count:** Every packet has a hop count associated with it. This is decremented(or incremented) by one by each node which sees it. When the hop count becomes zero(or a maximum possible value) the packet is dropped.
        - **Spanning Tree:** The packet is sent only on those links that lead to the destination by constructing a spanning tree routed at the source. This avoids loops in transmission but is possible only when all the intermediate nodes have knowledge of the network topology.

        Flooding is not practical for general kinds of applications. But in cases where high degree of robustness is desired such as in military applications, flooding is of great help.
    2. **Random Walk:** In this method a packet is sent by the node to one of its neighbours randomly. This algorithm is highly robust. When the network is highly interconnected, this algorithm has the property of making excellent use of alternative routes. It is usually implemented by sending the packet onto the least queued link.

Delta Routing

Delta routing is a hybrid of the centralized and isolated routing algorithms. Here each node computes the cost of each line (i.e some functions of the delay, queue length, utilization, bandwidth etc) and periodically sends a packet to the central node giving it these values which then computes the **k** best paths from node **i** to node **j**. Let **Cij1** be the cost of the best **i-j** path, **Cij2** the cost of the next best path and so on.If **Cijn - Cij1 < delta**, (**Cijn** - cost of **n'th** best **i-j** path, **delta** is some constant) then path **n** is regarded equivalent to the best **i-j** path since their cost differ by so little. When **delta -> 0** this algorithm becomes centralized routing and when **delta -> infinity** all the paths become equivalent.

Multipath Routing

In the above algorithms it has been assumed that there is a single best path between any pair of nodes and that all traffic between them should use it. In many networks however there are several paths between pairs of nodes that are almost equally good. Sometimes in order to improve the performance multiple paths between single pair of nodes are used. This technique is called multipath routing or bifurcated routing. In this each node maintains a table with one row for each possible destination node. A row gives the best, second best, third best, etc outgoing line for that destination, together with a relative weight. Before forwarding a packet, the node generates a random number and then chooses among the alternatives, using the weights as probabilities. The tables are worked out manually and loaded into the nodes before the network is brought up and not changed thereafter.

Hierarchical Routing

In this method of routing the nodes are divided into regions based on hierarchy. A particular node can communicate with nodes at the same hierarchial level or the nodes at a lower level and directly under it. Here, the path from any source to a destination is fixed and is exactly one if the heirarchy is a tree.

Routing Algorithms

Non-Hierarchical Routing

In this type of routing, interconnected networks are viewed as a single network, where bridges, routers and gateways are just additional nodes.

- Every node keeps information about every other node in the network
- In case of adaptive routing, the routing calculations are done and updated for all the nodes.

The above two are also the disadvantages of non-hierarchical routing, since the table sizes and the routing calculations become too large as the networks get bigger. So this type of routing is feasible only for small networks.

Hierarchical Routing

This is essentially a 'Divide and Conquer' strategy. The network is divided into different regions and a router for a particular region knows only about its own domain and other routers. Thus, the network is viewed at two levels:

1. The Sub-network level, where each node in a region has information about its peers in the same region and about the region's interface with other regions. Different regions may have different 'local' routing algorithms. Each local algorithm handles the traffic between nodes of the same region and also directs the outgoing packets to the appropriate interface.
2. The Network Level, where each region is considered as a single node connected to its interface nodes. The routing algorithms at this level handle the routing of packets between two interface nodes, and is isolated from intra-regional transfer.

Networks can be organized in hierarchies of many levels; e.g. local networks of a city at one level, the cities of a country at a level above it, and finally the network of all nations.

In Hierarchical routing, the interfaces need to store information about:

- All nodes in its region which are at one level below it.
- Its peer interfaces.
- At least one interface at a level above it, for outgoing packages.

Advantages of Hierarchical Routing :

- Smaller sizes of routing tables.
- Substantially lesser calculations and updates of routing tables.

Disadvantage :

- Once the hierarchy is imposed on the network, it is followed and possibility of direct paths is ignored. This may lead to sub optimal routing.

Source Routing

Source routing is similar in concept to virtual circuit routing. It is implemented as under:

- Initially, a path between nodes wishing to communicate is found out, either by flooding or by any other suitable method.
- This route is then specified in the header of each packet routed between these two nodes. A route may also be specified partially, or in terms of some intermediate hops.

Advantages:

- Bridges do not need to lookup their routing tables since the path is already specified in the packet itself.
- The throughput of the bridges is higher, and this may lead to better utilization of bandwidth, once a route is established.

Disadvantages:

- Establishing the route at first needs an expensive search method like flooding.
- To cope up with dynamic relocation of nodes in a network, frequent updates of tables are required, else all packets would be sent in wrong direction. This too is expensive.

Policy Based Routing

In this type of routing, certain restrictions are put on the type of packets accepted and sent. e.g.. The IIT- K router may decide to handle traffic pertaining to its departments only, and reject packets from other routes. This kind of routing is used for links with very low capacity or for security purposes.

Shortest Path Routing

Here, the central question dealt with is 'How to determine the optimal path for routing ?' Various algorithms are used to determine the optimal routes with respect to some predetermined criteria. A network is represented as a graph, with its terminals as nodes and the links as edges. A 'length' is associated with each edge, which represents the cost of using the link for transmission. Lower the cost, more suitable is the link. The cost is determined depending upon the criteria to be optimized. Some of the important ways of determining the cost are:

- **Minimum number of hops:** If each link is given a unit cost, the shortest path is the one with minimum number of hops. Such a route is easily obtained by a breadth first search method. This is easy to implement but ignores load, link capacity etc.
- **Transmission and Propagation Delays:** If the cost is fixed as a function of transmission and propagation delays, it will reflect the link capacities and the geographical distances. However these costs are essentially static and do not consider the varying load conditions.
- **Queuing Delays:** If the cost of a link is determined through its queuing delays, it takes care of the varying load conditions, but not of the propagation delays.

Ideally, the cost parameter should consider all the above mentioned factors, and it should be updated periodically to reflect the changes in the loading conditions. However, if the routes are changed according to the load, the load changes again. This feedback effect between routing and load can lead to undesirable oscillations and sudden swings.

Routing Algorithms

As mentioned above, the shortest paths are calculated using suitable algorithms on the graph representations of the networks. Let the network be represented by graph G (V, E) and let the number of nodes be 'N'. For all the algorithms discussed below, the costs associated with the

links are assumed to be positive. A node has zero cost w.r.t itself. Further, all the links are assumed to be symmetric, i.e. if $d_{i,j}$ = cost of link from node i to node j, then $d_{i,j} = d_{j,i}$. The graph is assumed to be complete. If there exists no edge between two nodes, then a link of infinite cost is assumed. The algorithms given below find costs of the paths from all nodes to a particular node; the problem is equivalent to finding the cost of paths from a source to all destinations.

Bellman-Ford Algorithm

This algorithm iterates on the number of edges in a path to obtain the shortest path. Since the number of hops possible is limited (cycles are implicitly not allowed), the algorithm terminates giving the shortest path.

Notation:

$d_{i,j}$ = Length of path between nodes i and j, indicating the cost of the link.
h = Number of hops.
D[i,h] = Shortest path length from node i to node 1, with upto 'h' hops.
D[1,h] = 0 for all h .

Algorithm :

Initial condition : D[i, 0] = infinity, for all i (i != 1)

Iteration : $D[i, h+1] = \min \{ d_{i,j} + D[j,h] \}$ over all values of j .

Termination : The algorithm terminates when

D[i, h] = D [i, h+1] for all i .

Principle:

For zero hops, the minimum length path has length of infinity, for every node. For one hop the shortest-path length associated with a node is equal to the length of the edge between that node and node 1. Hereafter, we increment the number of hops allowed, (from h to h+1) and find out whether a shorter path exists through each of the other nodes. If it exists, say through node 'j', then its length must be the sum of the lengths between these two nodes (i.e. $d_{i,j}$) and the shortest path between j and 1 obtainable in upto h paths. If such a path doesn't exist, then the path length remains the same. The algorithm is guaranteed to terminate, since there are utmost N nodes, and so N-1 paths. It has time complexity of $O (N^3)$.

Dijkstra's Algorithm

Notation:

D_i = Length of shortest path from node 'i' to node 1.
$d_{i,j}$ = Length of path between nodes i and j .

Algorithm

Each node j is labeled with Dj, which is an estimate of cost of path from node j to node 1. Initially, let the estimates be infinity, indicating that nothing is known about the paths. We now iterate on the length of paths, each time revising our estimate to lower values, as we obtain them. Actually, we divide the nodes into two groups ; the first one, called set P contains the nodes whose shortest distances have been found, and the other Q containing all the remaining nodes. Initially P contains only the node 1. At each step, we select the node that has minimum cost path to node 1. This node is transferred to set P. At the first step, this corresponds to shifting the node closest to 1

in P. Its minimum cost to node 1 is now known. At the next step, select the next closest node from set Q and update the labels corresponding to each node using :

$$D_j = \min [D_j , D_i + d_{j,i}]$$

Finally, after N-1 iterations, the shortest paths for all nodes are known, and the algorithm terminates.

Principle
Let the closest node to 1 at some step be i. Then i is shifted to P. Now, for each node j , the closest path to 1 either passes through i or it doesn't. In the first case Dj remains the same. In the second case, the revised estimate of D_j is the sum $D_i + d_{i,j}$. So we take the minimum of these two cases and update D_j accordingly. As each of the nodes get transferred to set P, the estimates get closer to the lowest possible value. When a node is transferred, its shortest path length is known. So finally all the nodes are in P and the D_j's represent the minimum costs. The algorithm is guaranteed to terminate in N-1 iterations and its complexity is $O(N^2)$.

The Floyd Warshall Algorithm

This algorithm iterates on the set of nodes that can be used as intermediate nodes on paths. This set grows from a single node (say node 1) at start to finally all the nodes of the graph. At each iteration, we find the shortest path using given set of nodes as intermediate nodes, so that finally all the shortest paths are obtained.

Notation
$D_{i,j}[n]$ = Length of shortest path between the nodes i and j using only the nodes 1,2,....n as intermediate nodes.

Initial Condition
$Di,j[0] = d_{i,j}$ for all nodes i,j .

Algorithm
Initially, n = 0. At each iteration, add next node to n. i.e. For n = 1,2, ...N-1 ,

$$Di,j[n + 1] = \min \{ D_{i,j}[n] , D_{i,n+1}[n] + D_{n+1,j}[n] \}$$

Principle
Suppose the shortest path between i and j using nodes 1,2,...n is known. Now, if node n+1 is allowed to be an intermediate node, then the shortest path under new conditions either passes through node n+1 or it doesn't. If it does not pass through the node n+1, then $D_{i,j}[n+1]$ is same as $D_{i,j}[n]$. Else, we find the cost of the new route, which is obtained from the sum, $D_{i,n+1}[n] + D_{n+1,j}[n]$. So we take the minimum of these two cases at each step. After adding all the nodes to the set of intermediate nodes, we obtain the shortest paths between all pairs of nodes together. The complexity of Floyd-Warshall algorithm is $O (N^3)$.

It is observed that all the three algorithms mentioned above give comparable performance, depending upon the exact topology of the network.

ARP,RARP,ICMP Protocols

Address Resolution Protocol

If a machine talks to another machine in the same network, it requires its physical or MAC address. But ,since the application has given the destination's IP address it requires some mechanism to bind the IP address with its MAC address.This is done through Address Resolution protocol (ARP).IP address of the destination node is broadcast and the destination node informs the source of its MAC address.

1. Assume broadcast nature of LAN
2. Broadcast IP address of the destination
3. Destination replies it with its MAC address.
4. Source maintains a cache of IP and MAC address bindings

But this means that every time machine A wants to send packets to machine B, A has to send an ARP packet to resolve the MAC address of B and hence this will increase the traffic load too much, so to reduce the communication cost computers that use ARP maintains a cache of recently acquired IP_to_MAC address bindings, i.e. they dont have to use ARP repeatedly. ARP Refinements Several refinements of ARP are possible: When machine A wants to send packets to macine B, it is possible that machine B is going to send packets to machine A in the near future.So to avoid ARP for machine B, A should put its IP_to_MAC address binding in the special packet while requesting for the MAC address of B. Since A broadcasts its initial request for the MAC address of B, every machine on the network should extract and store in its cache the IP_to_MAC address binding of A When a new machine appears on the network (e.g. when an operating system reboots) it can broadcast its IP_to_MAC address binding so that all other machines can store it in their caches. This will eliminate a lot of ARP packets by all other machines, when they want to communicate with this new machine.

Example displaying the use of Address Resolution Protocol:

Consider a scenario where a computer tries to contact some remote machine using ping program, assuming that there has been no exchange of IP datagrams previously between the two machines and therefore arp packet must be sent to identify the MAC address of the remote machine.

The arp request message (who is A.A.A.A tell B.B.B.B where the two are IP addresses) is broadcast on the local area network with an Ethernet protocol type 0x806. The packet is discarded by all the machines except the target machine which responds with an arp response message (A.A.A.A is hh:hh:hh:hh:hh:hh where hh:hh:hh:hh:hh:hh is the Ethernet source address). This packet is unicast to the machine with IP address B.B.B.B. Since the arp request message included the hardware address (Ethernet source address) of the requesting computer, target machine doesn't require another arp message to figure it out.

Reverse Address Resolution Protocol

RARP is a protocol by which a physical machine in a local area network can request to learn its IP address from a gateway server's Address Resolution Protocol table or cache. This is needed since the machine may not have permanently attacded disk where it can store its IP address permanently. A network administrator creates a table in a local area network's gateway router that maps the physical machine (or Medium Access Control - MAC) addresses to corresponding

Internet Protocol addresses. When a new machine is set up, its RARP client program requests from the RARP server on the router to be sent its IP address. Assuming that an entry has been set up in the router table, the RARP server will return the IP address to the machine which can store it for future use.

Detailed Mechanism

Both the machine that issues the request and the server that responds use physical network addresses during their brief communication. Usually, the requester does not know the physical address. So, the request is broadcasted to all the machines on the network. Now, the requester must identify istelf uniquely to the server. For this either CPU serial number or the machine's physical network address can be used. But using the physical address as a unique id has two advantages.

- These addresses are always available and do not have to be bound into bootstrap code.
- Because the identifying information depends on the network and not on the CPU vendor, all machines on a given network will supply unique identifiers.

Request:
Like an ARP message, a RARP message is sent from one machine to the another encapsulated in the data portion of a network frame. An ethernet frame carrying a RARP request has the usual preamle, Ethernet source and destination addresses, and packet type fields in front of the frame. The frame conatins the value 8035 (base 16) to identify the contents of the frame as a RARP message. The data portion of the frame contains the 28-octet RARP message. The sender braodcasts a RARP request that specifies itself as both the sender and target machine, and supplies its physical network address in the target hardware address field. All machines on the network receive the request, but only those authorised to supply the RARP services process the request and send a reply, such machines are known informally as RARP servers. For RARP to succeed, the network must contain at least one RARP server.
Reply:
Servers answers request by filling in the target protocol address field, changing the message type from request to reply, and sending the reply back directly to the machine making the request.

Timing RARP Transactions
Since RARP uses the physical network directly, no other protocol software will time the response or retransmit the request. RARP software must handle these tasks. Some workstations that rely on RARP to boot, choose to retry indefinitely until the receive a response. Other implementations announce failure after only a few tries to avoid flooding the network with unnecessary broadcast.

Mulitple RARP Servers
Advantage: More reliability. Diadvantage: Overloading may result when all servers respond. So, to get away with disadvantage we have primary and secondary servers. Each machine that makes RARP request is assigned a primary server. Normally, the primary server responds but if it fails, then requester may time out and rebroadcast the request.Whenever a secondary server receives a second copy of the request within a short time of the first, it responds. But, still there might be a problem that all secondary servers respond, thus overloading the network. So, the solution adopted is to avoid having all secondary servers transmit responses simultaneously. Each secondary server that receives the request computes a random delay and then sends a response.

Drawbacks of RARP

- Since it operates at low level, it requires direct addresss to the network which makes it difficult for an application programmer to build a server.

- It doesn't fully utilizes the capability of a network like ethernet which is enforced to send a minimum packet size since the reply from the server contains only one small piece of information, the 32-bit internet address.

RARP is formally described in RFC903.

ICMP

This protocol discusses a mechanism that gateways and hosts use to communicate control or error information.The Internet protocol provides unreliable,connectionless datagram service,and that a datagram travels from gateway to gateway until it reaches one that can deliver it directly to its final destination. If a gateway cannot route or deliver a datagram,or if the gateway detects an unusual condition, like network congestion, that affects its ability to forward the datagram, it needs to instruct the original source to take action to avoid or correct the problem. The Internet Control Message Protocol allows gateways to send error or control messages to other gateways or hosts;ICMP provides communication between the Internet Protocol software on one machine and the Internet Protocol software on another. This is a special purpose message mechanism added by the designers to the TCP/IP protocols. This is to allow gateways in an internet to report errors or provide information about unexpecter circumstances. The IP protocol itself contains nothing to help the sender test connectivity or learn about failures.

Error Reporting vs Error Correction
ICMP only reports error conditions to the original source; the source must relate errors to individual application programs and take action to correct problems. It provides a way for gateway to report the error It does not fully specify the action to be taken for each possible error. ICMP is restricted to communicate with the original source but not intermediate sources.
ICMP Message Delivery
ICMP messages travel across the internet in the data portion of an IP datagram,which itself travels across the internet in the data portion of an IP datagram,which itself travels across each physical network in the data portion of a frame.Datagrams carryin ICMP messages are routed exactly like datagrams carrying information for users;there is no additional reliability or priority.An exception is made to the error handling procedures if an IP datagram carrying an ICMP messages are not generated for errors that result from datagrams carrying ICMP error messages.
ICMP Message Format
It has three fields;an 8-bit integer message TYPE field that identifies the message,an 8-bit CODE field that provides further information about the message type,and a 16-bit CHECKSUM field (ICMP uses the same additive checksum algorithm as IP,but the ICMP checksum only covers the ICMP message).In addition , ICMP messages that report errors always include the header and first 64 data bits of the datagram causing the problem. The ICMP TYPE field defines the meaning of the message as well as its format.
The Types include :

TYPE FIELD	ICMP MESSAGE TYPE
0	ECHO REPLY
3	DESTINATION UNREACHABLE
4	SOURCE QUENCH
5	REDIRECT(CHANGE A ROUTE)
8	ECHO REQUEST
11	TIME EXCEEDED FOR A DATAGRAM
12	PARAMETER PROBLEM ON A DATAGRAM
13	TIMESTAMP REQUEST

14	TIMESTAMP REPLY
15	INFORMATION REQUEST(OBSOLETE)
16	INFORMATION REPLY(OBSOLETE)
17	ADDRESS MASK REQUEST
18	ADDRESS MASK REPLY TESTING DESTINATION

Reachabilty and Status :
TCP/IP protocols provide facilities to help network managers or users identify network problems.One of the most frequently used debugging tools invokes the ICMP echo request and echo reply messages.A host or gateway sends an ICMP echo request message to a specified destination.Any machine that receives an echo request formulates an echo reply and returns to the original sender.The request contains an optional data area; the reply contains a copy of the data sent in the request.The echo request and associated reply can be used to test whether a destination is reachable and responding.Because both the request and reply travel in IP datagrams,successful receipt of a reply verifies that major pieces of the transport system work.
1.1 : IP software on the source must route the datagram
2.2 : Intermediate gateways between the source and destination must be operating and must route datagram correctly.
3.3 : The destination machine must be running , and both ICMP and IP software must be working.
4.4 : Routes in gateways along the return path must be correct.

Echo Request and Reply
The field listed OPTIONAL DATA is a variable length field that contains data to be returned to the sender.An echo reply always returns exactly the same data as was received in the request.Fields IDENTIFIER and SEQUENCE NUMBER are used by the sender to match replies to request.The value of the TYPE field specifies whether the message is a request(8) or a reply(0).

Reports of Unreachable Destinations
The Code field in a destination unreachable message contains an integer that further describes th problem.Possible values are :

CODE VALUE	MEANING
0	NETWORK UNREACHABLE
1	HOST UNREACHABLE
2	PROTOCOL UNREACHABLE
3	PORT UNREACHABLE
4	FRAGMENTATION NEEDED AND DF SET
5	SOURCE ROOT FAILED
6	DESTINATION NETWORK UNKNOWN
7	DESTINATION HOST UNKNOWN
8	SOURCE HOST ISOLATED
9	COMMUNICATION WITH DESTINATION NETWORK ADMINISTRATIVELY PROHIBITED
10	COMMUNICATION WTTH DESTINATION HOST ADMINISTRATIVELY PROHIBITED
11	NETWORK UNREACHABLE FOR TYPE OF SERVICE
12	HOST UNREACHABLE FOR TYPE OF SERVICE

Whenever an error prevents a gateway from routing or delivering a datagram, the gateway sends a destination unreachable message back to the source and then drops the datagram.Network unreachable errors usually imply roting failures ; host unreachable errors imply delivery failures.Because the message contains a short prefix of the datagram that caused the problem, the

source will know exactly which address is unreachable. Destinations may be unreachable because hardware is temporarily out of service, because the sender specified a nonexistent destination address, or because the gateway does not have a route to the destination network. Although gateways send destination unreachable messages if they cannot route or deliver datagrams, not all such errors can be detected.If the datagram contains the source route option with an incorrect route, it may trigger a source route failure message.If a gateway needs to fragment adatagram but the "don't fragment" bit is set, the gateway sends a fragmentation needed message back to the source.

Congestion and Datagram Flow Control :
Gateways cannot reserve memory or communication resources in advance of receiving datagrams because IP is connectionless. The result is, gateways can overrun with traffic, a condition known as congestion.Congestion arises due to two reasons :

1. A high speed computer may be able to generate traffic faster than a network can transfer it .
2. If many computers sumultaneously need to send datagrams through a single gateway , the gateway can experience congestion, even though no single source causes the problem.

When datagrams arrive too quickly for a host or a gateway to process, it enqueues them in memory temporarily.If the traffic continues, the host or gateway eventually exhausts menory ans must discard additional datagrams that arrive. A machine uses ICMP source quench messages to releive congestion. A source quench message is a request for the source to reduce its current rate of datagram transmission.
There is no ICMP messages to reverse the effect of a source quench.
Source Quench :
Source quench messages have a field that contains a datagram prefix in addition to the usual ICMP TYPE,CODE,CHECKSUM fields.Congested gateways send one source quench message each time they discard a datagram; the datagram prefix identifies the datagram that was dropped.

Route Change Requests From Gateways :
Internet routing tables are initialized by hosts from a configuration file at system startup, and system administrators seldom make routing changes during normal operations.Gateways exchange routing information periodically to accomadate network changes and keep their routes up-to-date.The general rule is , Gateways are assumed to know correct routes; host begin wint minimal routing information and learn new routes from gateways. The GATEWAY INTERNET ADDRESS field contains the address of a gateway that the host is to use to reach the destination mentioned in the datagram header. The INTERNET HEADER field contains IP header plus the next 64 bits of the datagram that triggered the message.The CODE field of an ICMP redirect message further specifies how to interpret the destination address, based on values assigned as follows :

Code Value	Meaning
0	REDIRECT DATAGRAMS FOR THE NET
1	REDIRECT DATAGRAMS FOR THE HOST
2	REDIRECT DATAGRAMS FOR THE TYPE OF SERVICE AND NET
3	REDIRECT DATAGRAMS FOR THE TYPE OF SERVICE AND HOST

Gateways only send ICMP redirect requests to hosts and not to other gateways.

Detecting Circular or Excessively Long Routes :
Internet gateways compute a next hop using local tables, errors in routing tables can produce a routing cycle for some destination. A routing cycle can consist of two gateways that each route a datagram for a particular destination to other, or it can consist of several gateways.To prevent datagrams from circling forever in a TCP/IP internet, each IP datagram contains a time-to-live

counter , sometimes called a hop count. A gateway decrements the time-to-live counter whenever it processes the datagram and discards the datagram when the count reaches zero. Whenever a gateway discards a datagram because its hop count has reached zero or because a timeout occured while waiting for fragments of a datagram ,it sends an ICMP time exceeded message back to the datagram's source, A gateway sends this message whenever a datagram is discarded because the time-to-live field in the datagram header has reached zero or because its reassembly timer expired while waiting for fragments.
The code field explains the nature of the timeout :

Code Value	Meaning
0	TIME-TO-LIVE COUNT EXCEEDED
1	FRAGMENT REASSEMBLY TIME EXCEEDED

Fragment reassembly refers to the task of collecting all the fragments from a datagram.

Reprting Other Problems :
When a gateway or host finds problems with a datagram not covered by previous ICMP error messages it sends a parameter problem message to the original source.To make the message unambigous, the sender uses the POINTER field in the message header to identify the octet in the datagram that caused the problem. Code 1 is used to report that a required option is missing; the POINTER field is not used for code 1.

Clock Synchronization nd Transmit the estimation :
ICMP messages are used to obtain the time from another machine.A requesting machine sends an ICMP timestamp request message to another machine, asking that the second machine return its current value of the time of day. The receiving machine returns a timestamp reply back to the machine making the request. TCP/IP protocol suite includes several protocols that can be used to synchronize clocks. This is one of the simplest techniques used by TCP/IP. The TYPE field idintifies the message as a request (13) or a reply (14); the IDENTIFIER and SEQUENCE NUMBER fields are used by the source to associate replies with requests.The ORIGINATE TIMESTAMP filed is filled in by the original sendet just before the packet is transmitted, the RECEIVE TIMESTAMP field is filled immediately upon receipt of a request, and the TRANSMIT TIMESTAMP field is filled immediately before the reply is transmitted. Hosts use the three timestamp fields to compute estimates of the delay time between them and to synchronize their clock.A host can compute the total time required for a request to travel to a destination, be transformed into a reply, and return. In practice, accurate estimation of round-trip delay can be difficult and substantially restirct the utility of ICMP timestanp messages.To obtain an accurate estimate to round trip delay one must take many measurements and average them.

Obtaining a Subnet Mask:
Subnet addressing is used by the hosts to extract some bits in the hostid portion of their IP address to identify a physical network.To participate in subnet addressing, hosts need to know which bits of the 32-bit internet address correspond to the physical network and which correspond to host identifiers. The information needed to interpret the address is represented in a 32-bit quatity called the subnet mask. To learn the subnet mask used for the local network, a machine can send an address mask request message to a gateway and receive an address mask reply. The TYPE field in an address mask message specifies whether the message is a request (17) or a reply (18). A reply contains the nework's subnet address mask in the ADDRESS MASK field.The IDENTIFIER and SEQUENCE NUMBER fields allow a machine to associate replies with requests.

Transport Layer Protocol

What is TCP?

TCP was specifically designed to provide a reliable end to end byte stream over an unreliable internetwork. Each machine supporting TCP has a TCP transport entity either a user process or part of the kernel that manages TCP streams and interface to IP layer. A TCP entity accepts user data streams from local processes, breaks them up into pieces not exceeding 64KB and sends each piece as a separate IP datagram. Client Server mechanism is not necessary for TCP to behave properly.

The IP layer gives no guarantee that datagram will be delivered properly, so it is up to TCP to timeout and retransmit, if needed. Duplicate, lost and out of sequence packets are handled using the sequence number, acknowledgements, retransmission, timers, etc to provide a reliable service. Connection is a must for this service.Bit errors are taken care of by the CRC checksum. One difference from usual sequence numbering is that each byte is given a number instead of each packet. This is done so that at the time of transmission in case of loss, data of many small packets can be combined together to get a larger packet, and hence smaller overhead.

TCP connection is a *duplex connection*. That means there is no difference between two sides once the connection is established.

TCP Connection establishment

The "three-way handshake" is the procedure used to establish a connection. This procedure normally is initiated by one TCP and responded to by another TCP. The procedure also works if two TCP simultaneously initiate the procedure. When simultaneous attempt occurs, each TCP receives a "SYN" segment which carries no acknowledgment after it has sent a "SYN". Of course, the arrival of an old duplicate "SYN" segment can potentially make it appear, to the recipient, that a simultaneous connection initiation is in progress. Proper use of "reset" segments can disambiguate these cases.

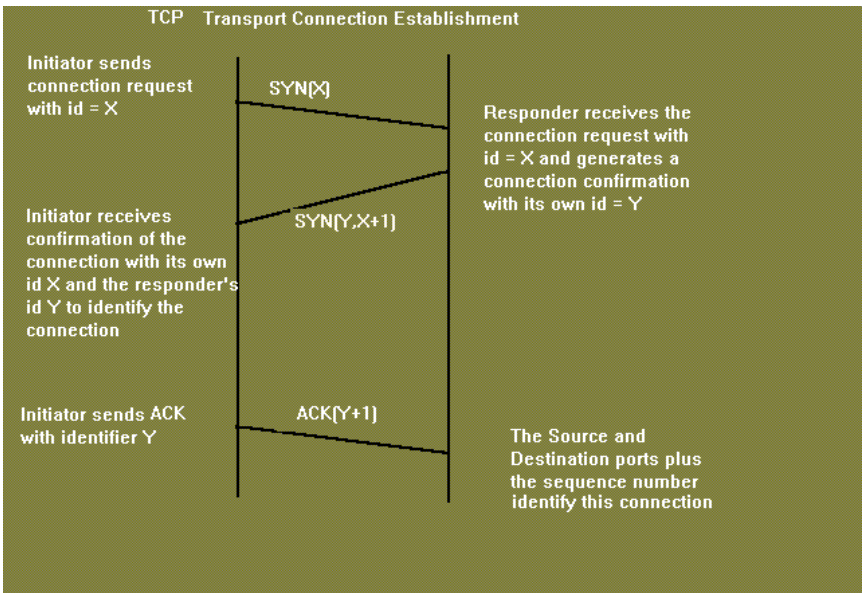
The three-way handshake reduces the possibility of false connections. It is the implementation of a trade-off between memory and messages to provide information for this checking.

The simplest three-way handshake is shown in figure below. The figures should be interpreted in the following way. Each line is numbered for reference purposes. Right arrows (-->) indicate departure of a TCP segment from TCP A to TCP B, or arrival of a segment at B from A. Left arrows (<--), indicate the reverse. Ellipsis (...) indicates a segment which is still in the network (delayed). TCP states represent the state AFTER the departure or arrival of the segment (whose contents are shown in the center of each line). Segment contents are shown in abbreviated form, with sequence number, control flags, and ACK field. Other fields such as window, addresses, lengths, and text have been left out in the interest of clarity.

```
     TCP A                                                TCP B

1.   CLOSED                                               LISTEN

2.   SYN-SENT    --> <SEQ=100><CTL=SYN>               --> SYN-RECEIVED

3.   ESTABLISHED <-- <SEQ=300><ACK=101><CTL=SYN,ACK>  <-- SYN-RECEIVED

4.   ESTABLISHED --> <SEQ=101><ACK=301><CTL=ACK>       --> ESTABLISHED

5.   ESTABLISHED --> <SEQ=101><ACK=301><CTL=ACK><DATA> --> ESTABLISHED

          Basic 3-Way Handshake for Connection Synchronisation
```

In line 2 of above figure, TCP A begins by sending a SYN segment indicating that it will use sequence numbers starting with sequence number 100. In line 3, TCP B sends a SYN and acknowledges the SYN it received from TCP A. Note that the acknowledgment field indicates TCP B is now expecting to hear sequence 101, acknowledging the SYN which occupied sequence 100.

At line 4, TCP A responds with an empty segment containing an ACK for TCP B's SYN; and in line 5, TCP A sends some data. Note that the sequence number of the segment in line 5 is the same as in line 4 because the ACK does not occupy sequence number space (if it did, we would wind up ACKing ACK's!).



Simultaneous initiation is only slightly more complex, as is shown in figure below. Each TCP cycles from CLOSED to SYN-SENT to SYN-RECEIVED to ESTABLISHED.

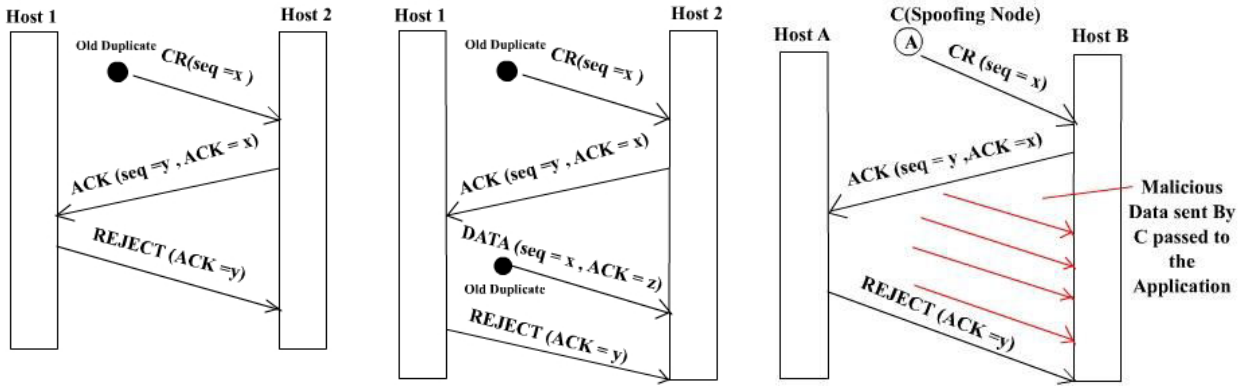
```
     TCP A                                          TCP B

1.   CLOSED                                         CLOSED

2.   SYN-SENT     --> <SEQ=100><CTL=SYN>             ...

3.   SYN-RECEIVED <-- <SEQ=300><CTL=SYN>             <-- SYN-SENT

4.                ... <SEQ=100><CTL=SYN>             --> SYN-RECEIVED

5.   SYN-RECEIVED --> <SEQ=100><ACK=301><CTL=SYN,ACK> ...

6.   ESTABLISHED  <-- <SEQ=300><ACK=101><CTL=SYN,ACK> <-- SYN-RECEIVED

7.                ... <SEQ=101><ACK=301><CTL=ACK>     --> ESTABLISHED

               Simultaneous Connection Synchronisation
```

Question: Why is three-way handshake needed? What is the problem if we send only two packets and consider the connection established? What will be the problem from application's point of view? Will the packets be delivered to the wrong application?

Problem regarding 2-way handshake

The only real problem with a 2-way handshake is that duplicate packets from a previous connection(which has been closed) between the two nodes might still be floating on the network. After a SYN has been sent to the responder, it might receive a duplicate packet of a previous connection and it would regard it as a packet from the current connection which would be undesirable.

Again spoofing is another issue of concern if a two way handshake is used.Suppose there is a node C which sends connection request to B saying that it is A.Now B sends an ACK to A which it rejects & asks B to close connection.Beteween these two events C can send a lot of packets which will be delievered to the application..

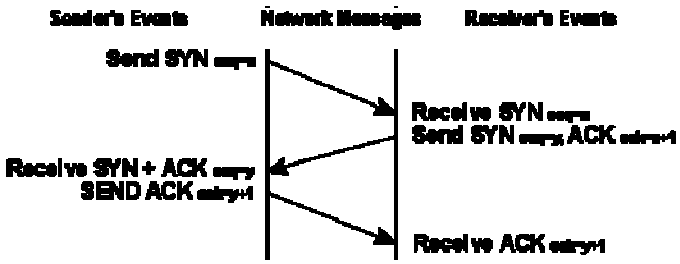


The first two figures show how a three way handshake deals with problems of duplicate/delayed connection requests and duplicate/delayed connection acknowledgements in the network.The third figure highlights the problem of spoofing associated with a two way handshake.

Some Conventions

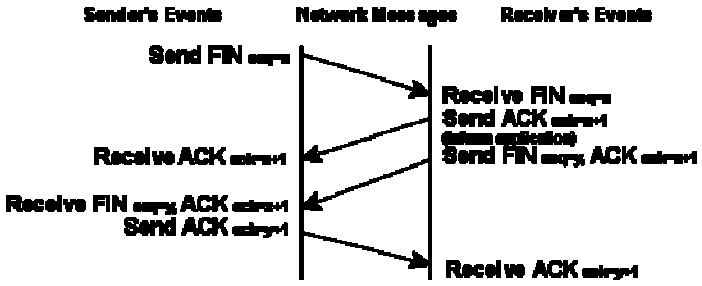
1. The ACK contains 'x+1' if the sequence number received is 'x'.
2. If 'ISN' is the sequence number of the connection packet then 1st data packet has the seq number 'ISN+1'
3. Seq numbers are 32 bit.They are byte seq number(every byte has a seq number).With a packet 1st seq number and length of the packet is sent.
4. Acknowlegements are cummulative.
5. Acknowledgements have a seq number of their own but with a length 0.So the next data packet have the seq number same as ACK.

Connection Establish



- The sender sends a SYN packet with serquence numvber say 'x'.
- The receiver on receiving SYN packet responds with SYN packet with sequence number 'y' and ACK with seq number 'x+1'
- On receiving both SYN and ACK packet, the sender responds with ACK packet with seq number 'y+1'
- The receiver when receives ACK packet, initiates the connection.

Connection Release



- The initiator sends a FIN with the current sequence and acknowledgement number.
- The responder on receiving this informs the application program that it will receive no more data and sends an acknowledgement of the packet. The connection is now closed from one side.
- Now the responder will follow similar steps to close the connection from its side. Once this is done the connection will be fully closed.

Image References

- http://www.renoir.vill.edu/~cassel/4900/transport.html
- www.uga.edu/~ucns/lans/tcpipsem/close.conn.gif
- www.uga.edu/~ucns/lans/tcpipsem/establish.conn.gif

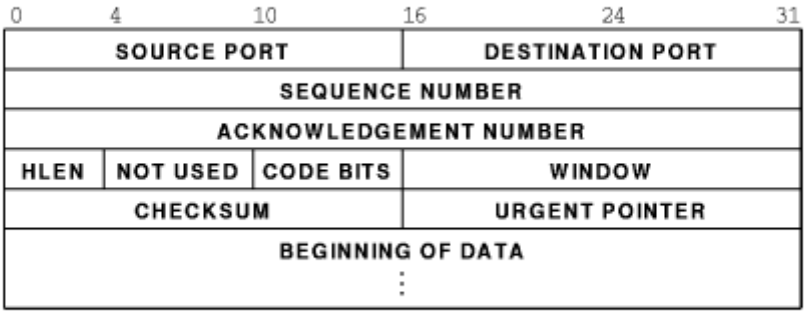
Transport Layer Protocol (continued)

TCP connection is a duplex connection. That means there is no difference between two sides once the connection is established.

Salient Features of TCP

- **Piggybacking of acknowledments:**The ACK for the last received packet need not be sent as a new packet, but gets a free ride on the next outgoing data frame(using the ACK field in the frame header). The technique is temporarily delaying outgoing ACKs so that they can be hooked on the next outgoing data frame is known as piggybacking. But ACK can't be delayed for a long time if receiver(of the packet to be acknowledged) does not have any data to send.
- **Flow and congestion control:**TCP takes care of flow control by ensuring that both ends have enough resources and both can handle the speed of data transfer of each other so that none of them gets overloaded with data. The term congestion control is used in almost the same context except that resources and speed of each router is also taken care of. The main concern is network resources in the latter case.
- **Multiplexing / Demultiplexing:** Many application can be sending/receiving data at the same time. Data from all of them has to be multiplexed together. On receiving some data from lower layer, TCP has to decide which application is the recipient. This is called demultiplexing. TCP uses the concept of port number to do this.

TCP segment header:



Explanation of header fields:

- **Source and destination port :**These fields identify the local endpoint of the connection. Each host may decide for itself how to allocate its own ports starting at 1024. The source and destination socket numbers together identify the connection.
- **Sequence and ACK number :** This field is used to give a sequence number to each and every byte transferred. This has an advantage over giving the sequence numbers to every packet because data of many small packets can be combined into one at the time of retransmission, if needed. The ACK signifies the next byte expected from the source and not the last byte received. The ACKs are cumulative instead of selective.Sequence number space is as large as 32-bit although 17 bits would have been enough if the packets were delivered in order. If packets reach in order, then according to the following formula:

  (sender's window size) + (receiver's window size) < (sequence number space)

the sequence number space should be 17-bits. But packets may take different routes and reach out of order. So, we need a larger sequence number space. And for optimisation, this is 32-bits.

- **Header length :**This field tells how many 32-bit words are contained in the TCP header. This is needed because the options field is of variable length.
- **Flags :** There are six one-bit flags.
    1. **URG :** This bit indicates whether the urgent pointer field in this packet is being used.
    2. **ACK :**This bit is set to indicate the ACK number field in this packet is valid.
    3. **PSH :** This bit indicates PUSHed data. The receiver is requested to deliver the data to the application upon arrival and not buffer it until a full buffer has been received.
    4. **RST :** This flag is used to reset a connection that has become confused due to a host crash or some other reason.It is also used to reject an invalid segment or refuse an attempt to open a connection. This causes an abrupt end to the connection, if it existed.
    5. **SYN :** This bit is used to establish connections. The connection request(1st packet in 3-way handshake) has SYN=1 and ACK=0. The connection reply (2nd packet in 3-way handshake) has SYN=1 and ACK=1.
    6. **FIN :** This bit is used to release a connection. It specifies that the sender has no more fresh data to transmit. However, it will retransmit any lost or delayed packet. Also, it will continue to receive data from other side. Since SYN and FIN packets have to be acknowledged, they must have a sequence number even if they do not contain any data.
- **Window Size :** Flow control in TCP is handled using a variable-size sliding window. The Window Size field tells how many bytes may be sent starting at the byte acknowledged. Sender can send the bytes with sequence number between (ACK#) to (ACK# + window size - 1) A window size of zero is legal and says that the bytes up to and including ACK# -1 have been received, but the receiver would like no more data for the moment. Permission to send can be granted later by sending a segment with the same ACK number and a nonzero Window Size field.
- **Checksum :** This is provided for extreme reliability. It checksums the header, the data, and the conceptual pseudoheader. The pseudoheader contains the 32-bit IP address of the source and destination machines, the protocol number for TCP(6), and the byte count for the TCP segment (including the header).Including the pseudoheader in TCP checksum computation helps detect misdelivered packets, but doing so violates the protocol hierarchy since the IP addresses in it belong to the IP layer, not the TCP layer.
- **Urgent Pointer :** Indicates a byte offset from the current sequence number at which urgent data are to be found. Urgent data continues till the end of the segment. This is not used in practice. The same effect can be had by using two TCP connections, one for transferring urgent data.
- **Options :** Provides a way to add extra facilities not covered by the regular header. eg,
    - Maximum TCP payload that sender is willing to handle. The maximum size of segment is called MSS (Maximum Segment Size). At the time of handshake, both parties inform each other about their capacity. Minimum of the two is honoured. This information is sent in the options of the SYN packets of the three way handshake.
    - Window scale option can be used to increase the window size. It can be specified by telling the receiver that the window size should be interpreted by shifting it left by specified number of bits. This header option allows window size up to 230.
- **Data :** This can be of variable size. TCP knows its size by looking at the IP size header.

Topics to be Discussed relating TCP

1. **Maximum Segment Size :** It refers to the maximum size of segment (MSS) that is acceptable to both ends of the connection. TCP negotiates for MSS using OPTION field. In Internet environment MSS is to be selected optimally. An arbitrarily small segment size will result in poor bandwith utilization since Data to Overhead ratio remains low. On the other hand extremely large segment size will necessitate large IP Datagrams which require fragmentation. As there are finite chances of a fragment getting lost, segment size above "fragmentation threshold " decrease the Throughput. Theoretically an optimum segment size is the size that results in largest IP Datagram, which do not require fragmentation anywhere enroute from source to destination. However it is very difficult to find such an optimum segmet size. In system V a simple technique is used to identify MSS. If H1 and H2 are on the same network use MSS=1024. If on different networks then MSS=5000.
2. **Flow Control :** TCP uses Sliding Window mechanism at octet level. The window size can be variable over time. This is achieved by utilizing the concept of "Window Advertisement" based on :
    1. **Buffer availabilty at the receiver**
    2. **Network conditions (traffic load etc.)**

    In the former case receiver varies its window size depending upon the space available in its buffers. The window is referred as RECEIVE WINDOW (Recv_Win). When receiver buffer begin to fill it advertises a small Recv_Win so that the sender does'nt send more data than it can accept. If all buffers are full receiver sends a "Zero" size advertisement. It stops all transmission. When buffers become available receiver advertises a Non Zero widow to resume retransmission. The sender also periodically probes the "Zero" window to avoid any deadlock if the Non Zero Window advertisement from receiver is lost. The Variable size Recv_Win provides efficient end to end flow control.

    The second case arises when some intermediate node (e.g. a router) controls the source to reduce transmission rate. Here another window referred as COGESTION WINDOW (C_Win) is utilized. Advertisement of C_Win helps to check and avoid congestion.
3. **Congestion Control :** Congestion is a condition of severe delay caused by an overload of datagrams at any intermediate node on the Internet. If unchecked it may feed on itself and finally the node may start dropping arriving datagrams.This can further aggravate congestion in the network resulting in congestion collapse. TCP uses two techniques to check congestion.
    1. **Slow Start :** At the time of start of a connection no information about network conditios is available. A Recv_Win size can be agreed upon however C_Win size is not known. Any arbitrary C_Win size can not be used because it may lead to congestion. TCP acts as if the window size is equal to the minimum of (Recv_Win & C_Win). So following algorithm is used.
        1. Recv_Win=X
        2. SET C_Win=1
        3. for every ACK received C_Win++
    2. **Multiplicative decrease :** This scheme is used when congestion is encountered (ie. when a segment is lost). It works as follows. Reduce the congestion window by half if a segment is lost and exponentially backoff the timer (double it) for the segments within the reduced window. If the next segment also gets lost continue the above process. For successive losses this scheme reduces traffic into the connection exponentially thus allowing the intermediate nodes to clear their queues. Once congestion ends SLOW START is used to scale up the transmission.
4. **Congestion Avoidance :** This procedure is used at the onset of congestion to minimize its effect on the network. When transmission is to be scaled up it should be done in such a way that it does'nt lead to congestion again. Following algorithm is used .
    1. At loss of a segment SET C_Win=1
    2. SET SLOW START THRESHOLD (SST) = Send_Win / 2
    3. Send segment

4. If ACK Received, C_Win++ till C_Win <= SST
5. else for each ACK C_Win += 1 / C_Win

5. **Time out and Retransmission :** Following two schemes are used :
    1. **Fast Retransmit**
    2. **Fast Recovery**

    When a source sends a segment TCP sets a timer. If this value is set too low it will result in many unnecessary treransmissions. If set too high it results in wastage of banwidth and hence lower throughput. In Fast Retransmit scheme the timer value is set fairly higher than the RTT. The sender can therefore detect segment loss before the timer expires. This scheme presumes that the sender will get repeated ACK for a lost packet.
6. **Round Trip Time (RTT) :** In Internet environment the segments may travel across different intermediate networks and through multiple routers. The networks and routers may have different delays, which may vary over time. The RTT therefore is also variable. It makes difficult to set timers. TCP allows varying timers by using an adaptive retransmission algorithm. It works as follows.
    1. Note the time (t1) when a segment is sent and the time (t2) when its ACK is received.
    2. Compute RTT(sample) = (t 2 - t 1)
    3. Again Compute RTT(new) for next segment.
    4. Compute Average RTT by weighted average of old and new values of RTT
    5. RTT(est) = a *RTT(old) + (1-a) * RTT (new) where 0 < a < 1
       A high value of 'a' makes the estimated RTT insensitive to changes that last for a short time and RTT relies on the history of the network. A low value makes it sensitive to current state of the network. A typical value of 'a' is 0.75
    6. Compute Time Out = b * RTT(est) where b> 1
       A low value of 'b' will ensure quick detection of a packet loss. Any small delay will however cause unnecessary retransmission. A typical value of 'b' is kept at .2

Image References

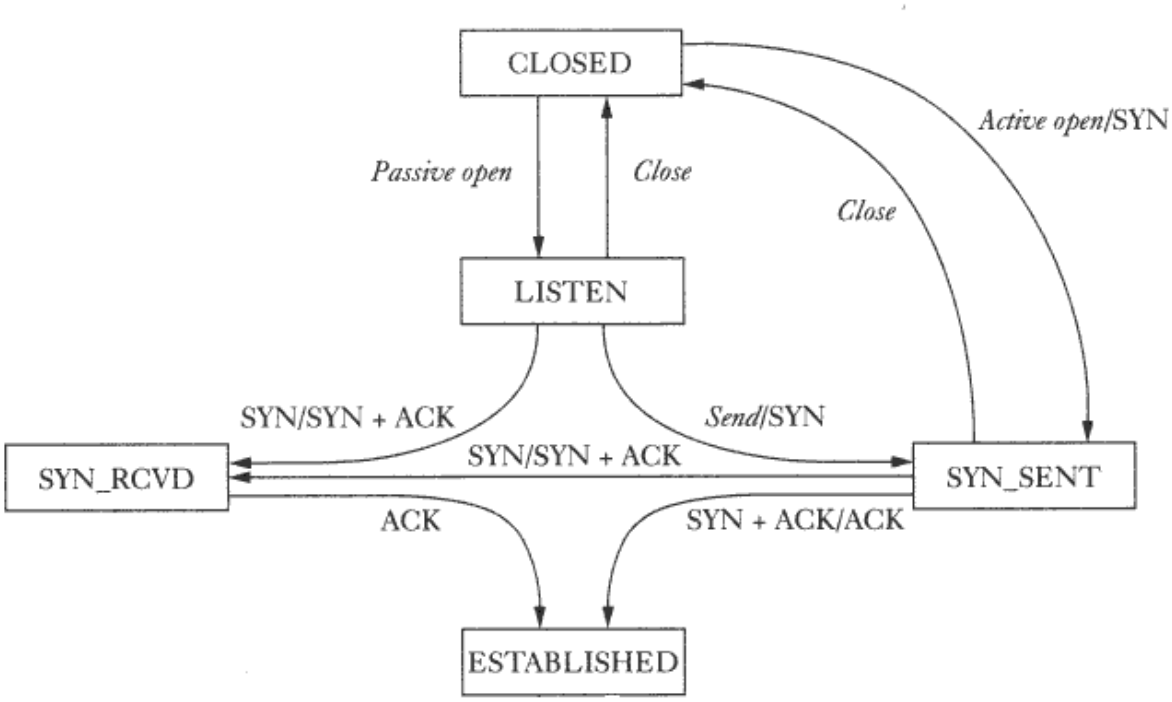
- http://plato.acadiau.ca/courses/comp/Eberbach/comp4343/lectures/transport/Com-TCP/f20_6.gif

Transport Layer Protocol- Implementation Issues

In this class we discussed about the TCP from the implementation point of view and addressed various issues like state diagram and other details which TCP Standard does not define but supported by commercial implementations.

State Diagram

The state diagram approach to view the TCP connection establishment and closing simplifies the design of TCP implementation. The idea is to represent the TCP connection state, which progresses from one state to other as various messages are exchanged. To simplify the matter, we considered two state diagrams, viz., for TCP connection establishment and TCP connection closing.
Fig 1 shows the state diagram for the TCP connection establishment and associated table briefly explains each state.



TCP Connection establishment

The table gives brief description of each state of the above diagram.

State Description Table 1.

Listen	Represents the state when waiting for connection request from any remote host and port. This specifically applies to a Server. From this state, the server can close the service or actively open a connection by sending SYN.
Syn-Sent	Represents waiting for a matching for a connection request after having sent a connection request. This applies to both server and client side. Even though server is considered as the one with passive open, it can also send a SYN packet actively.
Syn_Rcvd	Represents waiting for a confirmation connection request acknowledgment after having both received and sent connection request.
Estab	Represents an open connection. Data transfer can take place from this point onwards.

After the connection has been established, two end-points will exchange useful information and terminate the connection. Fig. 2 shows the state diagram for terminating an active connection.

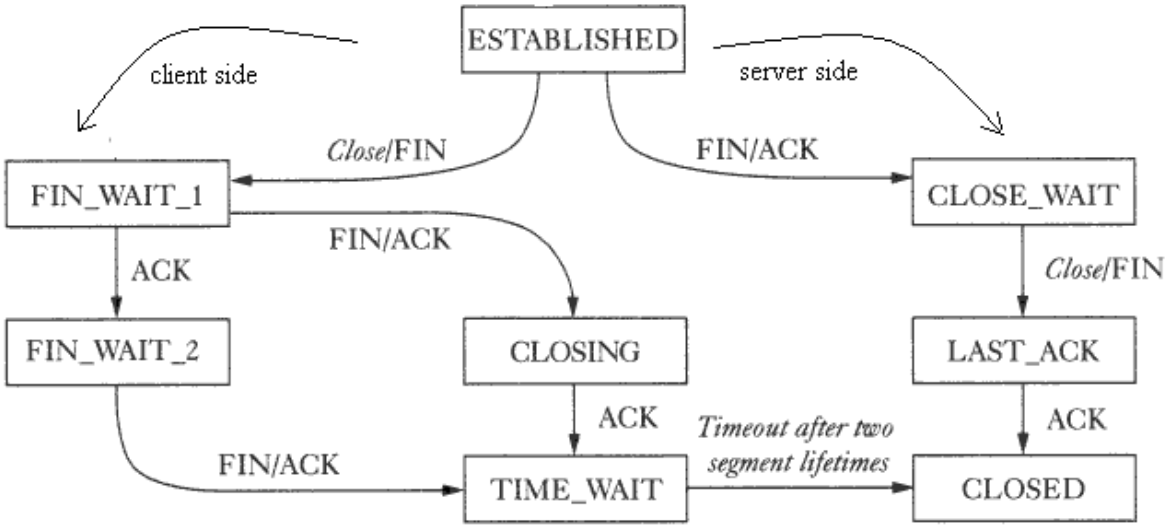


Fig 2. TCP Connection termination

State Description Table 2

FIN-WAIT-1	Represents connection termination request from the remote TCP peer, or an acknowledgment of the connection termination request previously sent. This state is entered when server issues close call.
FIN-WAIT-2	Represents waiting for a connection termination request from the remote TCP.
CLOSING	Represents connection termination request acknowledgment from the remote TCP.
TIME_WAIT	This represents waiting time enough for the packets to reach their destination. This waiting time is usually 4 min.
CLOSE_WAIT	Represents a state when the server receives a FIN from the remote TCP , sends ACK and issues close call sending FIN
LAST_ACK	Represents waiting for an ACK for the previously sent FIN-ACK to the remote TCP
CLOSE	Represents a closed TCP connection having received all the ACKs

Other implementation details

Quite Time

It might happen that a host currently in communication crashes and reboots. At startup time, all the data structures and timers will be reset to an initial value. To make sure that earlier connection packets are gracefully rejected, the local host is not allowed to make any new connection for a small period at startup. This time will be set in accordance with reboot time of the operating system.

Initial Sequence number :

Initial sequence number used in the TCP communication will be initialized at boot time randomly, rather than to 0. This is to ensure that packets from old connection should not interfere with a new connection. So the recommended method is to

- Initialize the ISN at boot time by a random number
- For every 500 ms, increment ISN by 64K
- With every SYN received, increment ISN by 64K

Maximum Request backlog at server

As we have seen in Unix Networking programming, *listen(sd,n),* sets a maximum to the number of requests to be obliged by the server at any time. So if there are already n requests for connection, and n+1 request comes, two things can be done.

- Drop the packet silently
- Ask the peer to send the request later.

The first option is recommended here because, the assumption is that this queue for request is a coincident and some time later, the server should be free to process the new request. Hence if we drop the packet, the client will go through the time-out and retransmission and server will be free to process it.

Also, Standard TCP does not define any strategy/option of knowing who requested the connection. Only Solaris 2.2 supports this option.

Delayed Acknowledgment

TCP will piggyback the acknowledgment with its data. But if the peer does not have the any data to send at that moment, the acknowledgment should not be delayed too long. Hence a timer for 200 ms will be used. At every 200 ms, TCP will check for any acknowledgment to be sent and send them as individual packets.

Small packets

TCP implementation discourages small packets. Especially if a previous relatively large packet has been sent and no acknowledgment has been received so far, then this small packet will be stored in the buffer until the situation improves.

But there are some applications for which delayed data is worse than bad data. For example, in *telnet,* each key stroke will be processed by the server and hence no delay should be introduced. As we have seen in Unix Networking programming, options for the socket can be set as NO_DELAY, so that small packets are not discouraged.

ICMP Source Quench

We have seen in ICMP that ICMP Source Quench message will be send for the peer to slow down. Some implementations discard this message, but few set the **current window size** to 1.
But this is not a very good idea.

Retransmission Timeout

In some implementation (E.g.. Linux), RTO = RTT + 4 * delay variance is used to instead of constant 2.

Also instead of calculating RTT(est) from the scratch, cache will be used to store the history from which new values are calculated as discussed in the previous classes.

Standard values for Maximum Segment Life (MSL) will be between 0.5 to 2 minutes and Time wait state = f(MSL)

Keep Alive Time

Another important timer in TCP is keep alive timer. It is basically used by a TCP peer to check whether the other end is up or down. It periodically checks this connection. If the other end did not respond, then that connection will be closed.

Persist Timer

As we saw in TCP window management, when source sends one full window of packets, it will set its window size to 0 and expects an ACK from remote TCP to increase its window size. Suppose such an ACK has been sent and is lost. Hence source will have current window size = 0 and cannot send & destination is expecting next byte. To avoid such a deadlock, a Persist Timer will be used. When this timer goes off, the source will send the last one byte again. So we hope that situation has improved and an ACK to increase the current window size will be received.

References

- http://www.ssfnet.org/Exchange/tcp/tcpTutorialNotes.html

Topics in TCP

TCP Congestion Control

If the receiver advertises a large window-size , larger than what the network en route can handle , then there will invariably be packet losses. So there will be re-transmissions as well . However , the sender cannot send all the packets for which ACK has not been received because this way it will be causing even more congestion in the network. Moreover , the sender at this point of time cannot be sure about how many packets have actually been lost . It might be that this is the only one that has been lost , and some following it have actually been received and buffered by the receiver. In that case , the sender will have unnecessarily sent a number of packets.

So the re-transmission of the packets also follows slow-start mechanism. However , we do indeed need to keep an upper bound on the size of the packets as it increases in slow start, to prevent it from increasing unbounded and causing congestion. This cap is put at half the value of the segment size at which packet loss started.

Congestion Window

We have already seen one bound on the size of the segments sent by the receiver-namely , the receiver window that the receiver advertises . However there could be a bottleneck created by some intermediate network that is getting clogged up. The net effect is that just having the receiver window is not enough. There should be some bound relating to the congestion of the network path - congestion window captures exactly this bound. Similar to receiver window, we have another window , the Congestion Window , and the maximum size of the segments sent are bounded by the minimum of the sizes of the two windows. E.g. If the receiver says "send 8K" (size of the receiver window) , but the sender knows that bursts of more than 4K (size of congestion window) clog the network up, then it sends 4K. On the other hand , if the congestion window was of size 32K , then the sender would send segments of maximum size 8K.

How do we calculate/manage the Congestion Window ?

The size of the congestion window is initialized to 1.For every ACK received , it is incremented by 1. Another field that we maintain is threshold which is equal to half the size of the congestion window. Whenever a packet loss takes place, the congestion window is set to 1.Then we keep increasing the congestion window by 1 on every ACK received till we reach the threshold value. Thereafter, we increment the congestion window size by 1 after every round trip time.

Notice that TCP always tries to keep the flow rate slightly below the maximum value. So if the network traffic fluctuates slightly, then a lot of packets might be lost. Packet losses cause a terrible loss in throughput.

In all these schemes, we have been assuming that any packet loss occurs only due to network congestion. What happens if some packet loss occurs not due to some congestion but due to some random factors?

When a packet is lost, the congestion window size is set to 1. Then when we retransmit the packet, if we receive a cumulative ACK for a lot of subsequent packets, we can assume that the packet loss was not due to congestion, but because of some random factors. So we give up slow start and straightaway set the size of Congestion Window to the threshold value.

Silly Window Syndrome

This happens when the application supplying data to the sender does do in large chunks, but the application taking data from receiver (probably an interactive application) does it in very small chunks, say 1 byte at a time. The sender keeps advertising windows of size 1 byte each as the application consumes the bytes one at a time.

Clark's Solution to this problem

We try to prevent the sender from advertising very small windows. The sender should try to wait until it has accumulated enough space in the window to send a full segment or half the receiver's buffer size, which it can estimate from the pattern of window updates that it received in the past.

Another problem: What if the same behavior is shown by an interactive application at the sender's end ? That is , what if the sender keeps sending in segments of very small size?

Nagle's algorithm

when data comes to the sender one byte at a time , send the first byte and buffer all the remaining bytes till the outstanding byte is acknowledged. Then send all the buffered characters in one segment and start buffering again till they are acknowledged. It can help reduce the bandwidth usage for example when the user is typing quickly into a telnet connection and the network is slow .

Persistent Timer

Consider the following deadlock situation . The receiver sends an ACK with 0 sized window, telling the sender to wait. Later it send an ACK with non-zero window, but this ACK packet gets lost. Then both the receiver and the sender will be waiting for each other to do something. So we keep another timer. When this timer goes off, the sender transmits a probe packet to the sender with an ACK number that is old. The receiver responds with an ACK with updated window size and transmission resumes.

Now we look at the solution of the last two problems ,namely **Problem of Random Losses** and **Sequence Number Wrap Around**.

Problem of Random Losses

How do we know if a loss is a congestion related loss or random loss ?If our window size is very large then we cannot say that one packet loss is random loss.So we need to have some mechanism to find what packets are lost. Cumulative Acknowledgement is not a good idea for this.

Solutions

Selective Acknowledgement

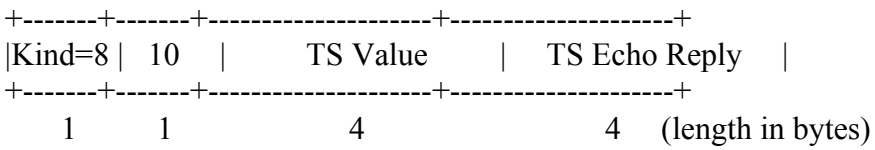
We need a selective acknowledgement but that creates a problem in TCP because we use byte sequence numbers .So what we we do is that we send the sequence number and the length. We may have to send a large number of such Selective Acknowledgements which will increase the overhead So whenever we get out of sequence packets we send the information a few time not in all the packets anyway. So we cannot rely on Selective Acknowledgement anyway. If we have 32

bit sequence number and 32 bit length,then already we will have too much of overhead .One proposal is to use 16 bit length field. If we have very small gaps then we will think that random losses are there and we need to fill them .If large gaps are there we assume that congestion is there and we need to slow down.

TCP Timestamps Option

TCP is a symmetric protocol, allowing data to be sent at any time in either direction, and therefore timestamp echoing may occur in either direction. For simplicity and symmetry, we specify that timestamps always be sent and echoed in both directions. For efficiency, we combine the timestamp and timestamp reply fields into a single TCP Timestamps Option.

Kind: 8
Length: 10 bytes



The Timestamps option carries two four-byte timestamp fields. The Timestamp Value field (TSval) contains the current value of the timestamp clock of the TCP sending the option. The Timestamp Echo Reply field (TSecr) is only valid if the **ACK** bit is set in the TCP header; if it is valid, it echos a times- tamp value that was sent by the remote **TCP** in the TSval field of a Timestamps option. When TSecr is not valid, its value must be zero. The TSecr value will generally be the time stamp for the last in-sequence packet received.

Example:

Sequence of packet send :	1 (t1)	2 (t2)	3 (t3)	4 (t4)	5 (t5)	6 (t6)
sequence of packets received:	1	2	4	3	5	6
time stamp copied in **ACK:**		t1	t2	t3		

PAWS: Protect Against Wrapped Sequence Numbers

PAWS operates within a single TCP connection, using state that is saved in the connection control block. PAWS uses the same TCP Timestamps option as the RTTM mechanism described earlier, and assumes that every received TCP segment (including data and ACK segments) contains a timestamp **SEG.TSval** whose values are monotone non-decreasing in time. The basic idea is that a segment can be discarded as an old duplicate if it is received with a timestamp **SEG.TSval** less than some timestamp recently received on this connection.

In both the PAWS and the RTTM mechanism, the "timestamps" are 32- bit unsigned integers in a modular 32-bit space. Thus, "less than" is defined the same way it is for TCP sequence numbers, and the same implementation techniques apply. If s and t are timestamp values, s < t if 0 < (t - s) < 2**31, computed in unsigned 32-bit arithmetic.

The choice of incoming timestamps to be saved for this comparison must guarantee a value that is monotone increasing. For example, we might save the timestamp from the segment that last advanced the left edge of the receive window, i.e., the most recent in- sequence segment. Instead, we choose the value **TS.Recent** for the RTTM mechanism, since using a common value for both PAWS and RTTM simplifies the implementation of both. **TS.Recent** differs from the timestamp from the last in-sequence segment only in the case of delayed ACKs, and therefore by less than one window. Either choice will therefore protect against sequence number wrap-around.

RTTM was specified in a symmetrical manner, so that TSval timestamps are carried in both data and ACK segments and are echoed in TSecr fields carried in returning ACK or data segments. PAWS submits all incoming segments to the same test, and therefore protects against duplicate ACK segments as well as data segments. (An alternative un-symmetric algorithm would protect against old duplicate ACKs: the sender of data would reject incoming ACK segments whose TSecr values were less than the TSecr saved from the last segment whose ACK field advanced the left edge of the send window. This algorithm was deemed to lack economy of mechanism and symmetry.)

TSval timestamps sent on {SYN} and {SYN,ACK} segments are used to initialize PAWS. PAWS protects against old duplicate non-SYN segments, and duplicate SYN segments received while there is a synchronized connection. Duplicate {SYN} and {SYN,ACK} segments received when there is no connection will be discarded by the normal 3-way handshake and sequence number checks of TCP.

Header Prediction

As we want to know that from which TCP connection this packet belongs. So for each new packet we have to match the header of each packet to the database that will take a lot of time so what we do is we first compare this header with the header of last received packet and on an average this will reduce the work. Assuming that this packet is from the same TCP connection from where we have got the last one (locality principal).

UDP (User Datagram Protocol)

UDP -- like its cousin the Transmission Control Protocol (TCP) -- sits directly on top of the base Internet Protocol (IP). In general, UDP implements a fairly "lightweight" layer above the Internet Protocol. It seems at first site that similar service is provided by both UDP and IP, namely transfer of data.But we need UDP for multiplexing/demultiplexing of addresses.

UDP's main purpose is to abstract network traffic in the form of datagrams. A datagram comprises one single "unit" of binary data; the first eight (8) bytes of a datagram contain the header information and the remaining bytes contain the data itself.

UDP Headers
The UDP header consists of four (4) fields of two bytes each:

Source Port	Destination Port
length	checksum

- source port number
- destination port number
- datagram size
- checksum

UDP port numbers allow different applications to maintain their own "channels" for data; both UDP and TCP use this mechanism to support multiple applications sending and receiving data concurrently. The sending application (that could be a client or a server) sends UDP datagrams through the source port, and the recipient of the packet accepts this datagram through the destination port. Some applications use static port numbers that are reserved for or registered to the application. Other applications use dynamic (unregistered) port numbers. Because the UDP port headers are two bytes long, valid port numbers range from 0 to 65535; by convention, values above 49151 represent dynamic ports.

The datagram size is a simple count of the number of bytes contained in the header and data sections . Because the header length is a fixed size, this field essentially refers to the length of the variable-sized data portion (sometimes called the payload). The maximum size of a datagram varies depending on the operating environment. With a two-byte size field, the theoretical maximum size is 65535 bytes. However, some implementations of UDP restrict the datagram to a smaller number -- sometimes as low as 8192 bytes.

UDP checksums work as a safety feature. The checksum value represents an encoding of the datagram data that is calculated first by the sender and later by the receiver. Should an individual datagram be tampered with (due to a hacker) or get corrupted during transmission (due to line noise, for example), the calculations of the sender and receiver will not match, and the UDP protocol will detect this error. The algorithm is not fool-proof, but it is effective in many cases. In UDP, check summing is optional -- turning it off squeezes a little extra performance from the system -- as opposed to TCP where checksums are mandatory. It should be remembered that check summing is optional only for the sender, not the receiver. If the sender has used checksum then it is mandatory for the receiver to do so.

Usage of the Checksum in UDP is optional. In case the sender does not use it, it sets the checksum field to all 0's. Now if the sender computes the checksum then the recipient must also compute the checksum an set the field accordingly. If the checksum is calculated and turns out to be all 1's then the sender sends all 1's instead of all 0's. This is since in the algorithm for checksum computation used by UDP, a checksum of all 1's if equivalent to a checksum of all 0's. Now the checksum field is unambiguous for the recipient, if it is all 0's then checksum has not been used, in any other case the checksum has to be computed.

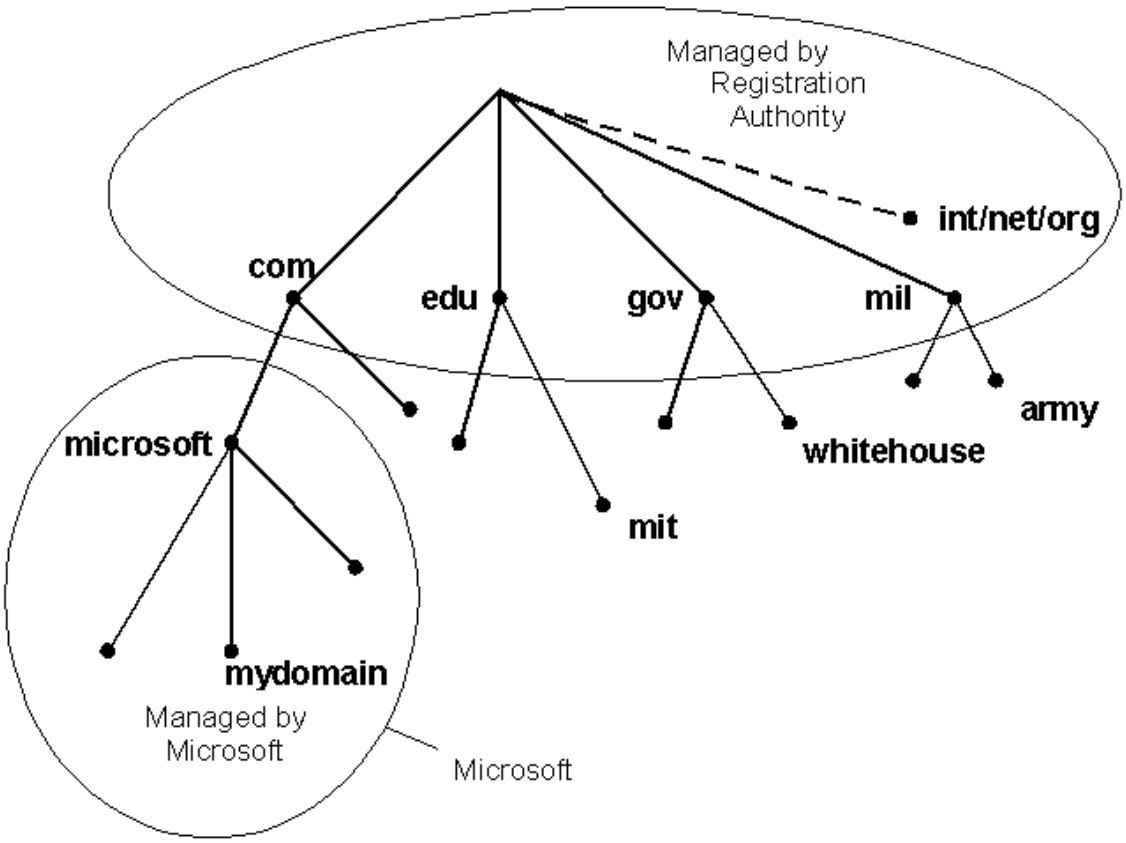
DNS (Domain Name Service)

The internet primarily uses IP addresses for locating nodes. However, its humanly not possible for us to keep track of the many important nodes as numbers. Alphabetical names as we see would be more convenient to remember than the numbers as we are more familiar with words. Hence, in the chaotic organization of numbers (IP addresses) we would be much relieved if we can use familiar sounding names for nodes on the network.

There is also another motivation for DNS. All the related information about a particular network (generally maintained by an organization, firm or university) should be available at one place. The organization should have complete control over what it includes in its network and how does it "organize" its network. Meanwhile, all this information should be available transparently to the outside world.

Conceptually, the internet is divide into several hundred top level domains where each domain covers many hosts. Each domain is partitioned in subdomains which may be further partitioned into subsubdomains and so on... So the domain space is partitioned in a tree like structure as shown below. It should be noted that this tree hierarchy has nothing in common with the IP address hierarchy or organization.

The internet uses a hierarchical tree structure of Domain Name Servers for IP address resolution of a host name.



The top level domains are either generic or names of countries. eg of generic top level domains are .edu .mil .gov .org .net .com .int etc. For countries we have one entry for each country as defined in ISO3166. eg. .in (India) .uk (United Kingdom).

The leaf nodes of this tree are target machines. Obviously we would have to ensure that the names in a row in a subdomain are unique. The max length of any name between two dots can be 63 characters. The absolute address should not be more than 255 characters. Domain names are case insensitive. Also in a name only letters, digits and hyphen are allowed. For eg. www.iitk.ac.in is a domain name corresponding to a machine named www under the subsubdomain iitk.ac.in.

Resource Records:

Every domain whether it is a single host or a top level domain can have a set of resource records associated with it. Whenever a resolver (this will be explained later) gives the domain name to DNS it gets the resource record

associated with it. So DNS can be looked upon as a service which maps domain names to resource records. Each resource record has five fields and looks as below:

Domain Name	Class	Type	Time to Live	Value

- Domain name: the domain to which this record applies.
- Class: set to IN for internet information. For other information other codes may be specified.
- Type: tells what kind of record it is.
- Time to live: Upper Limit on the time to reach the destination
- Value: can be an IP address, a string or a number depending on the record type.

Image Referecnes

- http://www.microsoft.com/technet/images/prodtechnol/windows2000serv/plan/images/w2kdns201_BIG.gif

DNS (Contd...)

Resource Record

A **Resource Record** (RR) has the following:

- **owner** which is the domain name where the RR is found.
- **type** which is an encoded 16 bit value that specifies the type of the resource in this resource record. It can be one of the following:
  - **A** a host address
  - **CNAME** identifies the canonical name of an alias
  - **HINFO** identifies the CPU and OS used by a host
  - **MX** identifies a mail exchange for the domain.
  - **NS** the authoritative name server for the domain
  - **PTR** a pointer to another part of the domain name space
  - **SOA** identifies the start of a zone of authority class which is an encoded 16 bit value which identifies a protocol family or instance of a protocol.
- **class** One of: **IN** the Internet system or **CH** the Chaos system
- **TTL** which is the time to live of the RR. This field is a 32 bit integer in units of seconds, an is primarily used by resolvers when they cache RRs. The TTL describes how long a RR can be cached before it should be discarded.
- **RDATA** Data in this field depends on the values of the type and class of the RR and a description for each is as follows:
  - for A: For the IN class, a 32 bit IP address For the CH class, a domain name followed by a 16 bit octal Chaos address.
  - for CNAME: a domain name.
  - for MX: a 16 bit preference value (lower is better) followed by a host name willing to act as a mail exchange for the owner domain.
  - for NS: a host name.
  - for PTR: a domain name.
  - for SOA: several fields.

Note: While short TTLs can be used to minimize caching, and a zero TTL prohibits caching, the realities of Internet performance suggest that these times should be on the order of days for the typical host. If a change can be anticipated, the TTL can be reduced prior to the change to minimize inconsistency during the change, and then increased back to its former value following the change. The data in the RDATA section of RRs is carried as a combination of binary strings and domain names. The domain names are frequently used as "pointers" to other data in the DNS.

Aliases and Cannonical Names

Some servers typically have multiple names for convenience. For example www.iitk.ac.in & yamuna.iitk.ernet.in identify the same server. In addition multiple mailboxes might be provided by some organizations. Most of these systems have a notion that one of the equivalent set of names is the canonical or primary name and all others are aliases.

When a name server fails to find a desired RR in the resource set associated with the domain name, it checks to see if the resource set consists of a CNAME record with a matching class. If so, the name server includes the CNAME record in the response and restarts the query at the domain name specified in the data field of the CNAME record.

Name Servers

Name servers are the repositories of information that make up the domain database. The database is divided up into sections called zones, which are distributed among the name servers. Name servers can answer queries in a simple manner; the response can always be generated using only local data, and either contains the answer to the question or a referral to other name servers "closer" to the desired information. The way that the name server answers the query depends upon whether it is operating in recursive mode or iterative mode:

- The simplest mode *for the server* is non-recursive, since it can answer queries using only local information: the response contains an error, the answer, or a referral to some other server "closer" to the answer. All name servers must implement non-recursive queries.
- The simplest mode *for the client* is recursive, since in this mode the name server acts in the role of a resolver and returns either an error or the answer, but never referrals. This service is optional in a name server, and the name server may also choose to restrict the clients which can use recursive mode.

Recursive Query vs Iterative Query

If the server is supposed to answer a recursive quesry then the response is either the reource record data or a error code. A server operating in this mode will never return the name of any forwarding name server but will contact the appropiate name server itself and try to get the information.

In iterative mode, on the other hand, if the server does not have the information requested locally then it return the address of some name server who might have the information about the query. It is then the responsibility of the contacting application to contact the next name server to resolve its query and do this iteratively until gets an answer or and error.

Relative Names

In place of giving full DNS names like cu2.cse.iitk.ac.in or bhaskar.cc.iitk.ac.in one can give just cu2 or bhaskar.This can be used by the server side as well as the client side.But for this one has to manually specify these extensions in the database of the servers holding the resource records.

BOOTP

The BOOTP uses UDP/IP. It is run when the machine boots. The protocol allows diskless machines to discover their IP address and the address of the server host. Additionally name of the file to be loaded from memory and executed is also supplied to the machine. This protocol is an improvement over RARP which has the follwing limitations:

1. Networks which do not have a broadcast method can't support RARP as it uses the broadcast method of the MAC layer underneath the IP layer.
2. RARP is heavily dependent on the MAC protocol.
3. RARP just supplies the IP address corresponding to a MAC address It doesn't support respond with any more data.
4. RARP uses the computer hardware's address to identify the machine and hence cannot be used in networks that dynamically assign hardware addresses.

Events in BOOTP

1. The Client broadcasts its MAC address (or other unique hardware identity number) asking for help in booting.
2. The BOOTP Server responds with the data that specifies how the Client should be configured (pre-configured for the specific client)

Note: BOOTP doesn't use the MAC layer broadcast but uses UDP/IP.

Configuration Information

The important informations provided are:

- IP address
- IP address of the default router for that particular subnet
- Subnet mask
- IP addresses of the primary and secondary nameservers

Additionaly it may also provide:

- Time offset from GMT
- The IP address of a time server
- The IP address of a boot server
- The name of a boot file (e.g. boot image for X terminals)
- The IP domain name for the client

But the problem with BOOTP is that it again can't be used for the dynamic IP's as in RARP servers.For getting dynamic IP's we use DHCP.

DHCP (Dynamic Host Configuration Protocol)

DHCP (Dynamic Host Configuration Protocol) is a protocol that lets network administrators manage centrally and automate the assignment of Internet Protocol (IP) addresses in an organization's network. If a machine uses Internet's set of protocol (TCP/IP), each machine that can connect to the Internet needs a unique IP address. When an organization sets up its computer users with a connection to the Internet, an IP address must be assigned to each machine. Without DHCP, the IP address must be entered manually at each computer and, if computers move to another location in another part of the network, a new IP address must be entered. DHCP lets a network administrator supervise and distribute IP addresses from a central point and automatically sends a new IP address when a computer is plugged into a different place in the network.

IP Address Allocation Mechanism

DHCP supports three mechanisms for IP address allocation.

- **Automatic allocation:** DHCP assigns a permanent IP address to a host.
- **Dynamic allocation:** DHCP assigns an IP address to a host for a limited period of time (or until the host explicitly relinquishes the address).
- **Manual allocation:** Host's IP address is assigned by the network administrator, and DHCP is used simply to convey the assigned address to the host. A particular network will use one or more of these mechanisms, depending on the policies of the network administrator.

Messages Used by DHCP

- **DHCP Discover** - Client broadcast to locate available servers. It is assumed atleast one of the servers will have resources to fulfill the request.(may include additional pointers to specific services required eg. particular subnet, minimum time limit etc).
- **DHCP Offer -** Server to client in response to DHCP Discover with offer of configration parameters.
- **DHCP Request -** Client broadcast to servers requesting offered parameters from one server and implicitly declining offers from all others.(also important in case of lease renewal if the alloted time is about to expire).
- **DHCP Decline -** Client to server indicating configration parameters invalid.
- **DHCP Release -** Client to server relinquishing network address and cancelling current lease.(in case of a graceful shut down DHCP server is sent a DHCP Release by the host machine).
- **DHCP Ack -** Server to client with configration parameters, including committed Network address.
- **DHCP Nack -** Server to client refusing request for configratin parameters (eg. requested network address already allocated).

Timers Used

Note that lease time is the time specified by the server for which the services have been provided to the client.

- **Lease Renewal Timer -** When this timer expires machine will ask the server for more time sending a DHCP Request.
- **Lease Rebinding Timer -** Whenever this timer expires, we have not been receiving any response from the server and so we can assume the server is down. Thus send a DHCP Request to all the servers using IP Broadcast facility. This is only point of difference between Lease renewal and rebinding.
- **Lease Expiry Timer -** Whenever this timer expires, the system will have to start crashing as the host does not have a valid IP address in the network.

Timer Configuration Policy

The timers have this usual setting which can be configured depending upon the usage pattern of the network. An example setting has been discussed below.

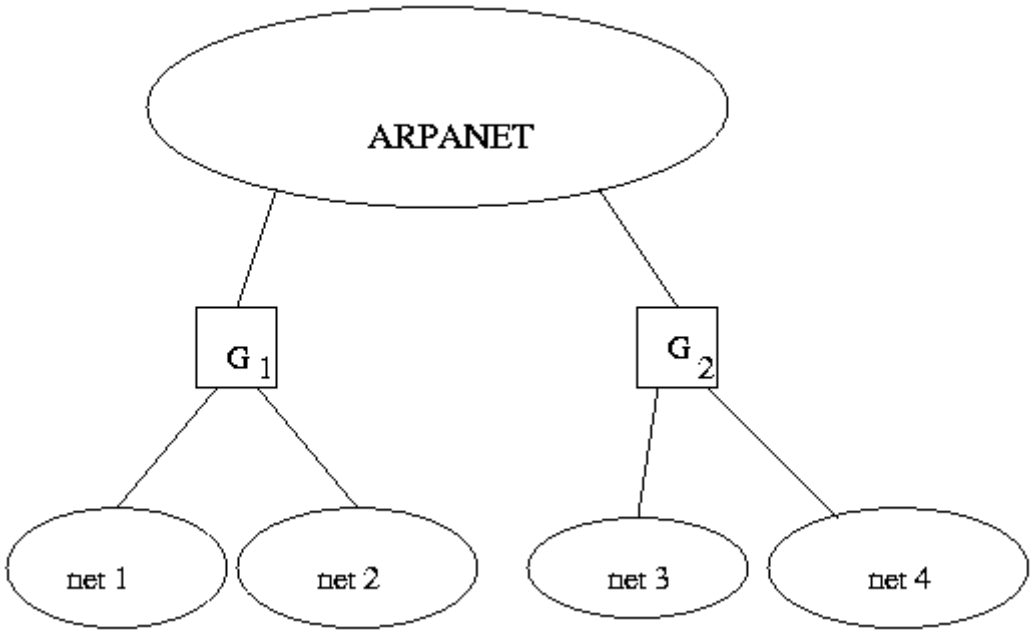
Lease Renewal = 50 % Lease time
Lease Rebinding = 87.5 % Lease time
Lease Expiry = 100 % Lease time

Routing in Internet

The Origin of Internet

The response of Internet to the issue of choosing routing tables with complete/par tail information is shown by the following architecture. There are a few nodes having complete routing information and a large number of nodes with partial information. The nodes with complete information, called core gateways, are well connected by a Backbone Network. These nodes talk to each other to keep themselves updated. The non-core gateways are connected to the core gateways. (Historically, this architecture comes from the ARPANET.)

The original internet was structured around a backbone of ARPANET with several core gateways connected to it .These core gateways connected some Local Area Networks (LANs) to the rest of the network. These core gateways talked to themselves and exchanged routing information's. Every core gateway contained complete information about all possible destinations.



How do you do routing ?

The usual IP routing algorithm employs an internet routing table (some times called an IP routing table) on each machine that Stores the information about the possible destinations, and how to reach them.

Default Routes

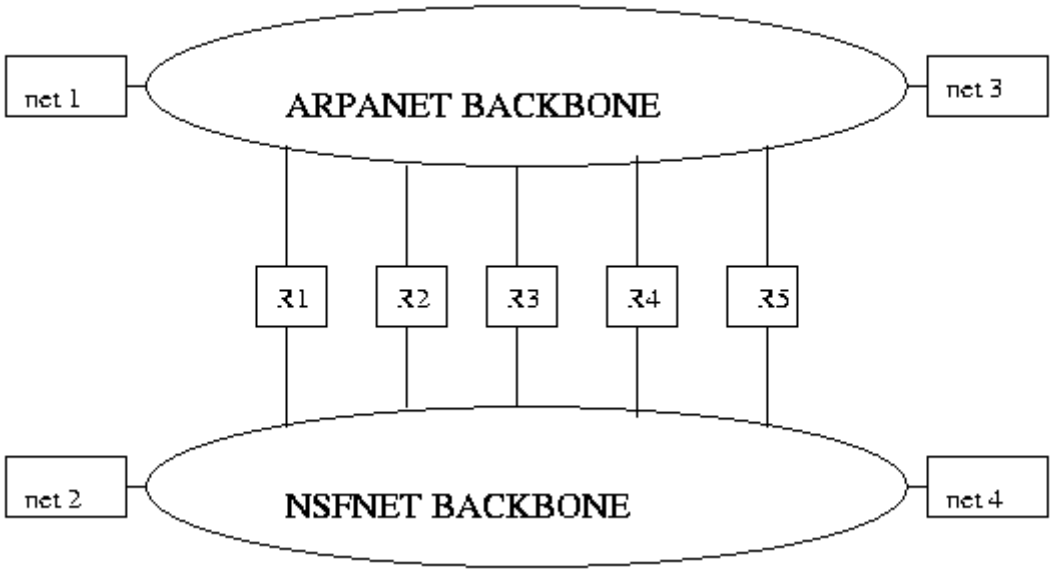
This technique used to hide information and keep routing table size small consolidates multiple entries into a default case. If no route appears in the routing table, the routing routine sends the data gram to the *default router*.

Default routing is especially useful when a site has a small set of local addresses and only one connection to the rest of the internet.

Host-Specific Routes

Most IP routing software allows per-host routes to be specified as a special case. Having per-host routes gives the local network administrator more control over network use, permits testing, and can also be used to control access for security purposes. when debugging network connections or routing tables, the ability to specify a special route to one individual machine turns out to be especially useful.

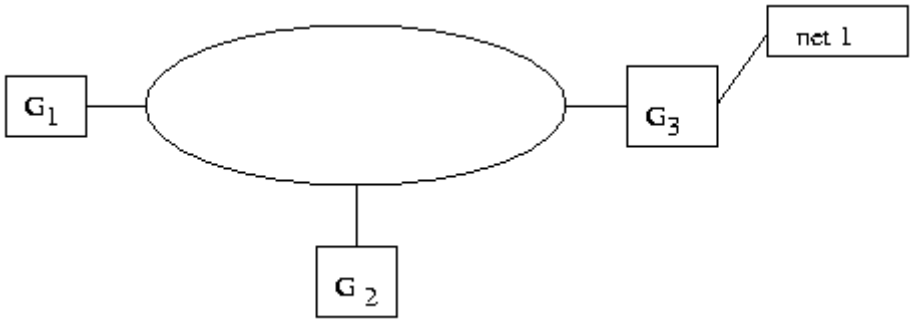
Internet with Two Backbones



As long as there was just one single router connecting ARPANET with NSFNET there was no problem. The core gateways of ARPANET had information about all destinations and the routers inside NSFNET contained information about local destinations and used a default route to send all non-NSFNET traffic to between NSFNET and ARPANET as both of them used different matrices to measure costs. the core gateways through the router between ARPANET and NSFNET. However as multiple connections were made between the two backbones, problems arise. Which route should a packet from net1 to net2 take? **Should it be R1 or R2 or R3 or R4 or R5?** For this some exchange of routing information between the two backbones was necessary. But, this was again a problem as how should we compare information.

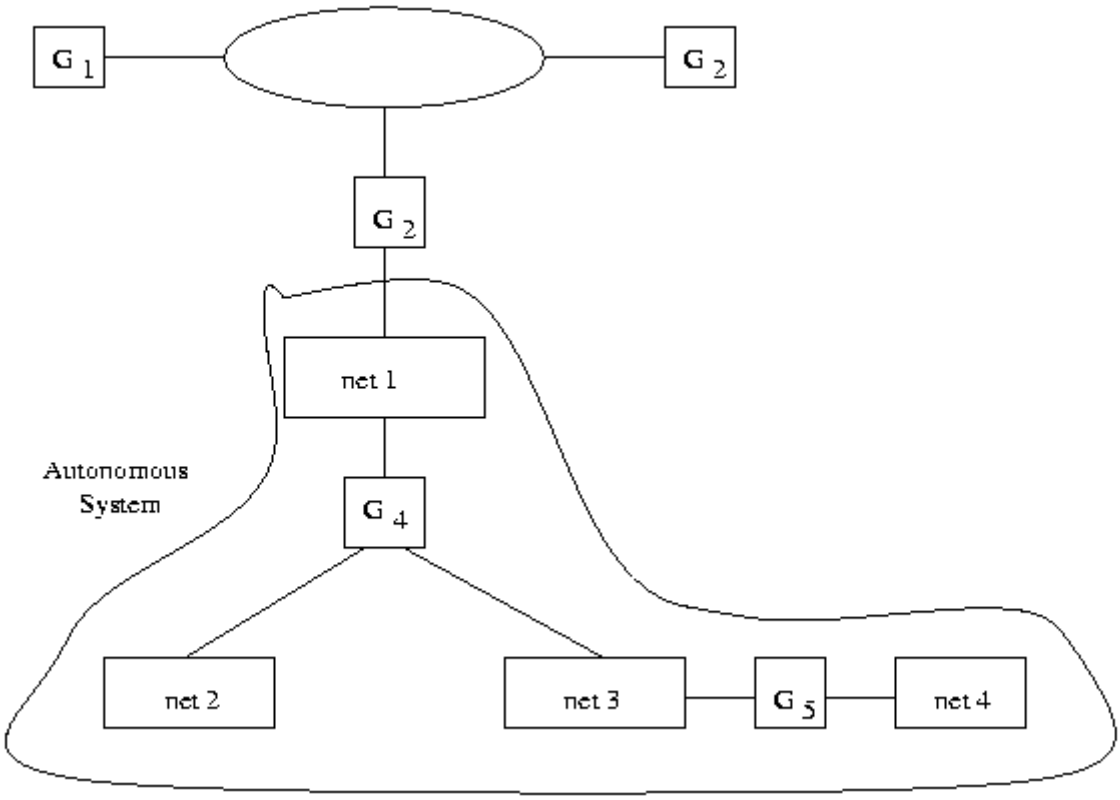
Gateway-To-Gateway Protocol (GGP)

This was the protocol used by the core-routers to exchange routing information among themselves. This is based on ***Distance Vector Algorithm*** and uses number of hops as the distance metric. This is a very poor metric as this does not take into account the load on the links and whether a link is slow or fast. A provision is made to manually increment the hop count in case a link is particularly slow.A protocol based on Shortest Path First Algorithm , known as **SPREAD** ,was also used for the same purpose.



Added Complexity To The Architecture Model

As the number of networks and routers increased, to reduce the load on the core gateways because of the enormous amount of calculations, routing was done with some core gateways keeping complete information and the non-core gateways keeping partial information.



In thisarchitecture, G_1 ,G_2 ,G_3 are all core gateways and G_4 and G_5 are non-core gateways. We must have a mechanism for someone to tell G_2 that it is connected to net2 , net3 and net4 , besides net1. Only G_5 can tell this to G_2 and so we must provide for a mechanism for G_2 to talk to G_5 . A concept of one backbone with core gateways connected to *Autonomous Systems* was developed. An *Autonomous system* is a group of networks controlled by a single administrative authority. Routers within an autonomous system are free to choose their own mechanisms for discovering , propagating ,validating , and checking the consistency of routes. Each autonomous system must agree to advertise network reachability information to other autonomous systems. Each advertisement propagates through a core router. The assumption made is that most of the routers in the autonomous system have complete information about the autonomous system. One such router will be assigned the task of talking to the core gateway.

Interior Gateway Protocols (IGP)

IGP is a type of protocols used by the routers in an autonomous system to exchange network reachability and routing information. Some of IGPs are given below.

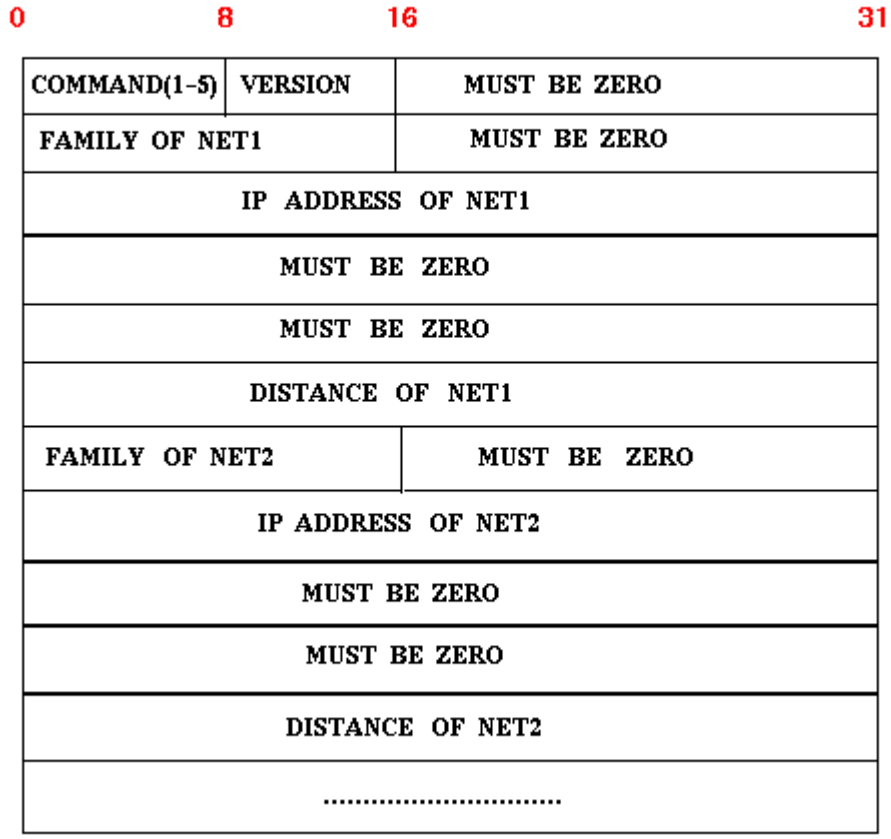
Routing Information Protocol (RIP)

This is one of the most widely used IGP. It was developed at Berkeley. This is also known by the name of the program that implements it, routed .This implements Distance Vector algorithm.Features of RIP:

- RIP uses a hop count metric to measure the distance to a destination. To compensate for differences in technologies, many RIP implementations allow managers to configure artificially high hop counts when advertising connections to slow networks. All routinfg updates are broadcast. This allows all hosts on the network to know about the routes.
- To prevent routes from oscillating between two or more equal cost paths, RIP specifies that existing routes should be retained until a new route has strictly lower cost. Since RIP does not explicitly detect routing loops, RIP must either assume participants can be trusted (being part of one autonomous system) or take precautions to prevent such loops.
- To prevent instabilities, RIP must use a low value for the maximum possible distance.RIP uses 16 as the maximum hop count. This restricts the maximum network diameter of the system to 16.
- To solve the slow convergence problem arising due to slow propagation of routing information, RIP uses Hold Down. If a particular link is down , any new information about that link is not accepted till some time. This is because the router must wait till the information aboutthe link being down propagates to another router before accepting information from that router about that down link.
- RIP runs on top of TCP/IP. RIP allows addresses to be of a maximum size of 14 Bytes. The Distance varies from 1 to 16 (where 16 is used to signify infinity). RIP address 0.0.0.0 denotes a default route. There is no explicit size of the RIP message and any number of routes can be advertized.

The message format is as shown:

FORMAT OF A RIP MESSAGE

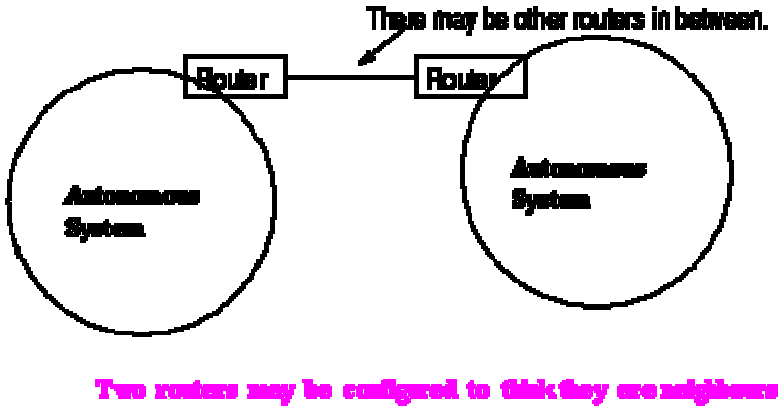


OSPF(Open Shortest Path First)

This is an Interior Gateway Protocol designed by the Internet Engineering Task Force (IETF). This algorithm scales better than the vector distance algorithms. This Protocol tackles several goals:

- OSPF includes type of service(ToS) routing. So, you can installmultiple routers to a given destination, one for each type of service. When routing a datagram, a router running OSPF uses both the destination address and type of service fields in the IP Header to choose a route.
- OSPF provides load balancing. If there are multiple routes to a given destination at the same cost, OSPF distributes traffic over all the routes equally.
- OSPF allows for creation of AREA HIERARCHIES. This makes the growth of the network easier and makes the network at a site easier to manage. Each area is self contained, so, multiple groups within a site can cooperate in the use of OSPF for routing.
- OSPF protocol specifies that all exchanges between the routers be authenticated. OSPF allows variety of authentication schemes, and even allows one area to choose a different scheme from the other areas.
- To accomodate multi-access networks like ethernet, OSPF allows every multi-access network to have a designated router(designated gateway).
- To permit maximum flexibility, OSPF allows the description of a virtual network topology that abstracts away from details of physical connections.
- OSPF also allows for routers to exchange routing information learned from other sites. The message format distinguishes between information acquired from external sources and

information acquired from routers interior to the site, so there is no ambiguity about the source or reliability of routes.
- It hastoo much overhead of sending LSPs but is gradually becoming popular.



Exterior Gateway Protocol (EGP)

If two routers belonging to two different autonomous systems exchange routing information ,the protocol used is called EGP . EGP consists of:

- **Acquisition Request:** A router sends a request to another neighbour router saying 'I want to talk'.
- **Acquisition Confirm:** This is a positive reply to the Acquisition request.
- **Acquisition Refuse:** This is a negative response to the Acquisition request.
- **Cease Request:** This requests termination of neighbour relationship.
- **Cease Confirm:** This is a confirmation response to the Cease Request.
- **Hello :** This is used to find if the neighbour router is up or down.This requests router to respond if alive.
- **I Heard You:** This is a response to the Hello message confirming that the router is alive. Because it is possible for Hello or I Heard You messages to be lost in transit, EGP uses a k-out-of-n rule to determine whether a network is down.At least k of the last n messages must fail for the router to declare its neighbour down.
- **Poll Request:** This is a request for network routing update.
- **Routing Update:** This conveys routing information about reachable networks to its EGP neighbour. The routing information is the distance vector of the reachable networks.
- **Error:** This is a response to an incorrect message.

EGP is used only to find network reachability and not for differentiating between good and bad routes. We can only use distance metric to declare a route plausible and not for comparing it with some other route (unless the two route form part of a same autonomous system). Since there cannot be two different routes to the same network, EGP restricts the topology of any internet to a tree structure in which a core system forms the root. There are no loops among other autonomous systems connected to it. This leads to several problems:

- Univerasal connectivity fails if the core gateway system fails.
- EGP can advertise only one path to a given network.
- EGP does not support load sharing on routers between arbitrary autonomous systems.
- Multiple backbone networks with multiple connections between them cannot be handled by EGP.

Border Gateway Protocol(BGP)

BGP is a distance-vector protocol used to communicate between different ASes. Instead of maintaining just the cost to each destination,each BGP router keeps track of the exact path used.Similarly,instead of periodically giving each neighbour its estimated cost to each destination, each BGP router tells its neighbours the path it is using.Every BGP router contains a module that examines routes to a given destination and scores them returning a number for destination to each route. Any route violating a policy constraint automatically gets a score of infinity. The router adapts a route with shortest distance.The scoring function is not a part of the BGP protocol and can be any function that the system managers want.BGP easily solves the count to infinity problem that plagues other distance-vector algorithms as whole path is known.

Routing (Continued)

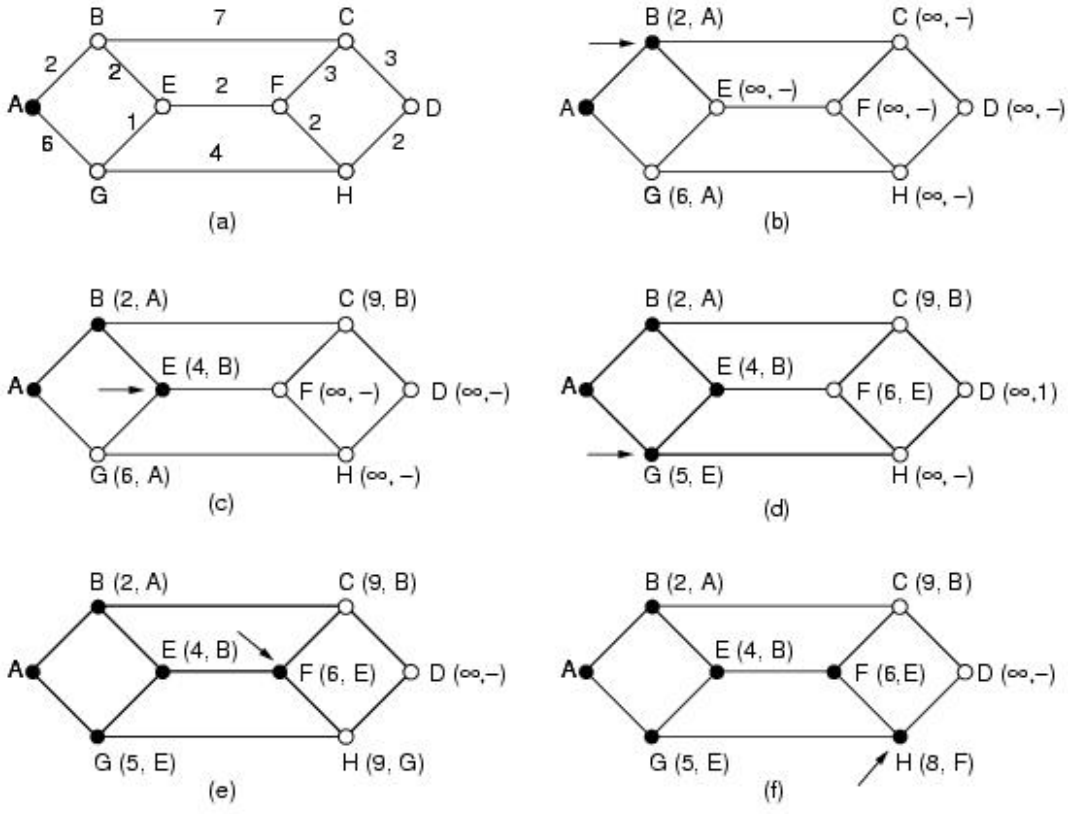
Shortest Path Algorithm

1. Dijktstra's Algorithm:

At the end each node will be labeled (see *Figure.1*) with its distance from source node along the best known path. Initially, no paths are known, so all nodes are labeled with infinity. As the algorithm proceeds and paths are found, the labels may change reflecting better paths. Initially, all labels are tentative. When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter.

Look at the weighted undirected graph of *Figure.1(a)*, where the weights represent, for example, distance. We want to find shortest path from A to D. We start by making node A as permanent, indicated by a filled in circle. Then we examine each of the nodes adjacent to A (the working node), relabeling each one with the distance to A. Whenever a node is relabeled, we also label it with the node from which the probe was made so that we can construct the final path later. Having examined each of the nodes adjacent to A, we examine all the tentatively labeled nodes in the whole graph and make the one with the smallest label permanent, as shown in *Figure.1(b).* This one becomes new working node.

We now start at B, and examine all nodes adjacent to it. If the sum of the label on B and the distance from B to the node being considered is less than the label on the node, we have a shorter path, so the node is relabeled. After all the nodes adjacent to the working node have been inspected and the tentative labels changed if possible, the entire graph is searched for the tentatively labeled node with the smallest value. This node is made permanent and becomes the working node for the next round. The *Figure. 1* shows the first five steps of the algorithm.



Note: Dijkstra's Algorithm is applicable only when cost of all the nodes is non-negative.

2. Bellman Ford's Algorithm:

We look at the distributed version which works on the premise that the information about far away nodes can be had from the adjoining links.

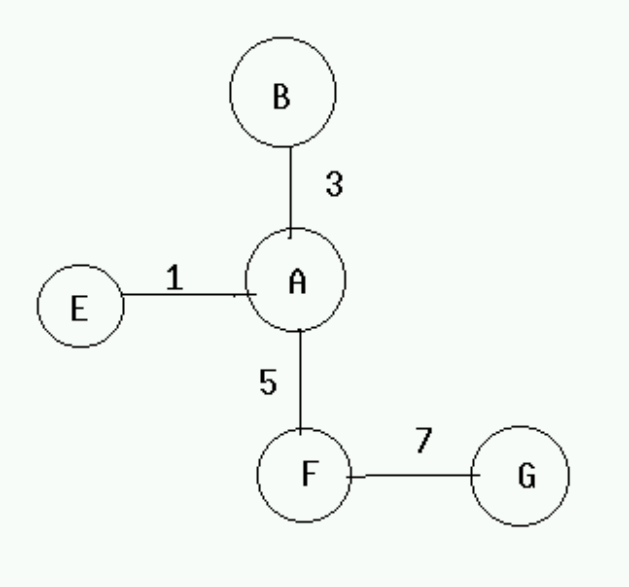
The algorithm works as follows.

- Compute the link costs from the starting node to every directly connected node .
- Select the cheapest links for every node (if there is more than one) .
- For every directly connected node, compute the link costs for all these nodes.
- Select the cheapest route for any node .
  Repeat until all nodes have been processed.

Every node should have the information about it's immediate neighbors and over a period of time we will have information about other nodes. Within n units of time , where n is the diameter of the network, every node will have the complete information. We do not need to be synchronized i.e. do not need to exchange information at the same time.

Routing algorithms based on Dijkstra's algorithm are called Link State Algorithms. Distance Vector Protocols are based on distributed Bellman's algorithm. In the former we are sending little information to many nodes while in the latter we send huge information to few neighbors.

Count-to-Infinity problem:



Suppose the link between A and E is down events may occur are:

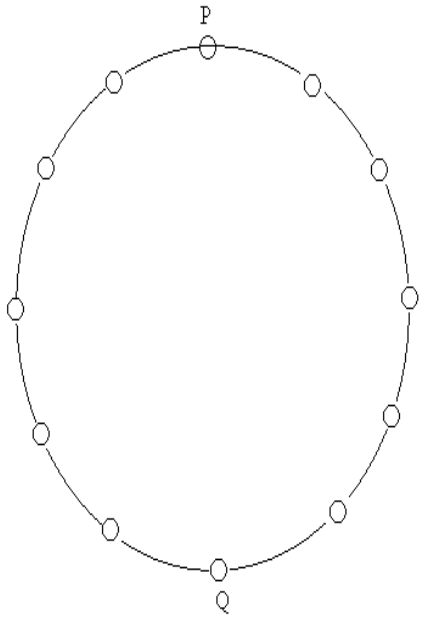
(1) F tells A that it has a path to E with cost 6
(2) A sets cost to E to be 11, and advertise to F again
(3) F sets the cost to E to be 16, and advertise to A again

This cycle will continue and the cost to E goes to infinity. The core of the problem is that when X tells Y that it has a path somewhere ,Y has no way to know whether it itself is on the path.

During this process of counting to infinity, packets from A or F destined to E are likely to loop back and forth between A and F, causing congestion for other packets.

Example to illustrate bad Routing Protocol :

Design of a bad routing protocol can lead to highly undesirable results. Consider the following scenario to understand this. We are having 16 nodes logically connected in a ring as shown in Figure1.



Each node sends one unit of data in unit time except one node Q which sends e (0<e<1) unit of data in unit time. We consider cost of the link as the traffic in that link. We consider P as the only receiver in the ring. Now Ideally we must have nodes left of the diagonal PQ sending data clockwise and that on the right of the PQ counterclockwise as shown in Figure 2. We may assume that Q sends data counterclockwise. Assume that this ideal distribution was achieved at some time. Now we can see that cost of links to the left of PQ are respectively 1,2,3 ..,7 while that on the right of PQ are e,1+e,2+e ..,7+e. Therefore when we reconsider the shortest path the node immediately to the right of Q will see traffic 28 to the left while 28+7e to the right and therefore will start sending data clockwise and same is true for Q also. This will heavily change the traffic on the network. Now the traffic load will shift to the left of PQ and next reconsideration will make a lot of nodes from left of PQ send data counterclockwise. This may keep oscillating and will cost a lot to the network. To prevent these two steps can be taken :

1. Assign a minimum cost to each link.

2. Do not change the route until you get a significant advantage.

Applications

FTP

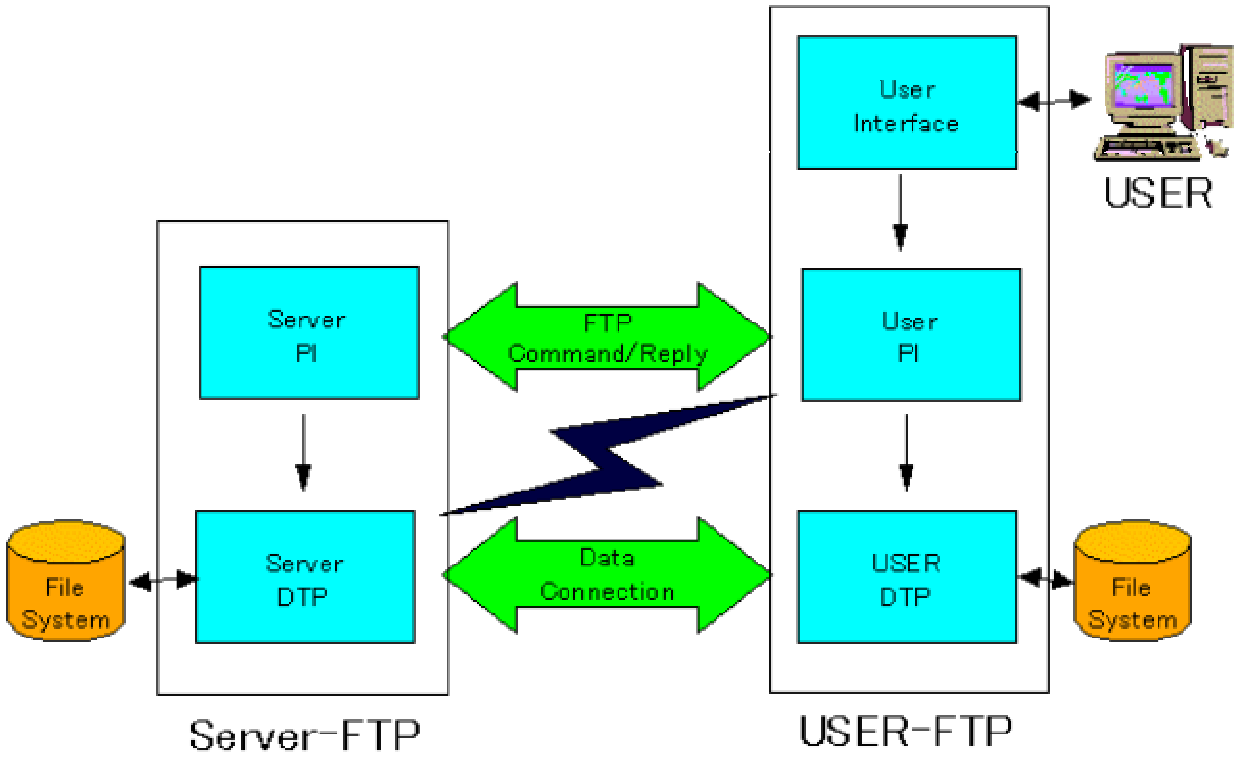
Given a reliable end-to-end trasport protocol like TCP, File Transfer might seem trivial. But, the details authorization, representation among heterogeneous machines make the protocol complex.

FTP offers many facilities :

- Interactive Access : Most implementations provide an interactive interface that allows humans to easily interact with remote servers.
- Format (representation) specification : FTP allows the client to specify the type and format of stored data.
- Authentication Control : FTP requires client to authorize themselves by sending a login name and password to the server before requesting file transfers.

FTP Process Model

FTP allows concurrent accesses by nultiple clients. Clients use TCP to connect to the server. A master server awaits connections and creates a slave process to handleeach connection. Unlike most servers, the slave process does not perform all the necessary computation. Instead the slave accepts and handles the control connection from the client, but uses an additinal process to handle a separate data transfer connection. The control connection carries the command that tells the server which file to transfer.



Data transfer connections and the data transfer processes that use them can be created dynamically when needed, but the control connection persists throughout a session. Once the control connection disappears, the session is terminated and the software at both ends terminates all data transfer processes.

In addition topassing user commands to the server, FTP uses the control connection to allow client and server processes to coordinate their use of dynamically assigned TCP protocol ports and the creation of data transfer processes that use those ports.

Proxy commands - allows one to copy files from any machine to any other arbitrary machine ie. the machine the files are being copied to need not be the client but any other machine.
Sometimes some **special processing** can be done which is not part of the protocol. eg. if a request for copying a file is made by issuing command 'get file_A.gz' and the zipped file does not exist but the file file_A does , then the file is automatically zipped and sent.
Consider what happens when the **connection breaks during a FTP session**. Two things may happen, certain FTP servers may again restart from the beginning and whatever portion of the file had been copied is overwritten. Other FTP servers may ask the client how much it has already read and it simply continues from that point.

TFTP

TFTP stands for Trivial File Transfer Protocol. Many applications do not need the full functionality of FTP nor can they afford the complexity. TFTP provides an inexpensive mechanism that does not need complex interactions between the client and the server. TFTP restricts operations to simple file transfer and does not provide authentication. Diskless devices have TFTP encoded in read-only memory(ROM) and use it to obtain an initial memory image when the machine is powered on. The advantage of using TFTP is that it allows bootstrapping code to use the same underlying TCP/IP protocols. that the operating system uses once it begins execution. Thus it is possible for a computer to bootstrap from a server on another physical network. TFTP does not have a reliable stream transport service. It runs on top of UDP or any other unreliable packet delivery system using timeout and retransmission to ensure that data arrives. The sending side transmits a file in fixed size blocks and awaits acknowledgements for each block before sending the next.

Rules for TFTP

The first packet sent requests file transfer and establishes connection between server and client. Other specifications are file name and whether it is to be transferred to client or to the server. Blocks of the file are numbered starting from 1 and each data packet has a header that specifies the number of blocks it carries and each acknowledgement contains the number of the block being acknowledged. A block of less than 512 bytes signals end of file. There can be five types of TFTP packets . The initial packet must use operation codes 1 or 2 specifying either a read request or a write request and also the filename. Once the read request or write request has been made the server uses the IP address and UDP port number of the client to identify subsequent operations.Thus data or ack msgs do not contain filename. The final message type is used to report errors.
TFTP supports symmetric retransmission. Each side has a timeout and retransmission.If the side sending data times out, then it retransmits the last data block. If the receiving side times out it retransmits the last acknowledgement. This ensures that transfer will not fail after a single packet loss.
Problem caused by symmetric retransmission - **Sorcerer's Apprentice Bug**
When an ack for a data packet is delayed but not lost then the sender retransmits the same data packet which the receiver acknowledges. Thus both the acks eventually arrives at the sender and the sender now transmits the next data packet once corresponding to each ack. Therefore a retransmission of all the subsequent packets are triggered . Basically the receiver will acknowledge both copies of this packet and send two acks which causes the sender in turn to send

two copies of the next packet.. The cycle continues with each packet being transmitted twice. TFTP supports multiple file types just like FTP ie. binary and ascii data. TFTP may also be integrated with email . When the file type is of type mail then the FILENAME field is to be considered as the name of the mailbox and instead of writing the mail to a new file it should be appended to it. However this implementation is not commonly used .

Now we look at another very common application EMAIL

EMAIL (electronic mail - SMTP , MIME , ESMTP)

Email is the most widely used application service which is used by computer users. It differs from other uses of the networks as network protocols send packets directly to destinations using timeout and retransmission for individual segments if no ack returns. However in the case of email the system must provide for instances when the remote machine or the network connection has failed and take some special action.Email applications involve two aspects -

- User-agent(pine, elm etc.)
- Transfer agent(sendmail daemon etc.)

When an email is sent it is the mail transfer agent (MTA) of the source that contacts the MTA of the destination. The protocol used by the MTA 's on the source and destination side is called SMTP. SMTP stands for **Simple Mail Transfer Protocol.**. There are some protocols that come between the user agent and the MTA eg. POP,IMAP which are discussed later.

Mail Gateways -

Mail gateways are also called mail relays, mail bridges and in such systems the senders machine does not contact the receiver's machine directly but sends mail across one or more intermediate machines that forward it on. These **intermediate machines** are called mail gateways.Mail gateways are introduce unreliablity.Once the sender sends to first intermediate m/c then it discards its local copy. So failure at an intermediate machine may result in message loss without informing the sender or the receiver. Mail gateways also introduce delays. Neither the sender nor the receiver can determine how long the delay will last or where it has been delayed.
However mail gateways have an advantage providing interoperability ie. they provide connections among standard TCP/IP mail systems and other mail systems as well as between TCP/IP internets and networks that do not support Internet protocols. So when there is a change in protocol then the mail gateway helps in translating the mail message from one protocol to another since it will be designed to understand both. .

SIMPLE MAIL TRANSFER PROTOCOL(SMTP)

TCP/IP protocol suite specifies a standard for the exchange of mail between machines. It was derived from the (MTP) Mail Transfer Protocol. it deals with how the nderlying mail delivery system passes messages across a link from one.machine to another. The mail is enclosed in what is called an **envelope** . The enveilope contains the To and From fields and these are followed by the mail . The mail consists of two parts namely the Header and the Data.
The Header has the To and From fields. If Headers are defined by us they should start with X. The standard headers do not start with X.
In SMTP data portion can contain only printable ASCII characters The old method of sending a binary file was to send it in uuencoded form but there was no way to distinguish between the many types of binary files possible eg. .tar , .gz , .dvi etc.

MIME(Multipurpose Internet Mail Extension)

This alllows the transmission of Non ASCII data througfh email, MIME allows arbitrary data to be encoded in ASCII and sent in a standard email message. Each MIME message includes information that tells the recipient the type of data and the type of encoding used.and this information alongwith the MIME version resides in the MIME header. Typical MIME header looks like -

MIME-Version: 1.0
Content-Description:
Content-Id:
Content-Type: image/gif
Content-Transfer-Encoding: base64

Content Descirption : contains the file name of the file that is being sent. Content -Type : is an important field that specifies the data format ie. tells what kind of data is being sent. It contains two identifiers a content type and a subtype separated by a slash. for e.g. image/gif
There are 7 Content Types -

1. text
2. image
3. video
4. audio
5. application
6. multipart
7. message

Content type - **Message**
It supports 3 subtypes namely

1. RFC822 - the old mail message format
2. Partial- means that ordinary message is just a part and the receiver should wait for all the parts before putting it in the mailbox.
3. external_body - destination MTA will fetch file from remote site.

Content Type - **Multipart**
Multiple messages which may have different content types can be sent together. It supports 4 subtypes namely

1. mixed -Look at each part independently
2. alternative - The same message is sent in multiple types and formats and the receiver may choose to read the message in any form he wishes.
3. parallel -The different parts of the message have to be read in parallel. ie.audio , video and text need to be read in a synchronised fashion
4. digest -There are multiple RFC messages in mail. The addresses of the receivers are in the form of a mailing list. Although file header is long it prevents cluttering of mail box.

PROBLEMS WITH SMTP

1. There is no convenient way to send nonprintable characters
2. There is no way to know if one has received mail or not or has read it or not.
3. Someone else can send a mail on my behalf.

So a better protocol was proposed - **ESMTP** ESMTP stands for Extended Simple Mail Transfer Protocol. It is compatible with SMTP. Just as the first packet sent in SMTP is HELO similarly in ESMTP the first packet is called EHELO. If the receiver supports ESMTP then it will answer to this EHELO packet by sending what data type and what kind of encoding it supports. Even a SMTP based receiver can reply to it. Also if there is an error message or there is no answer then the sender uses SMTP.

DELIVERY PROTOCOLS

The delivery protocols determine how the mail is transferred by the mail transfer agent to the user agent which provides an interface for reading mails.

There are 3 kinds

1. **POP3 (Post Office Protocol)** Here the mail person accesses the mail box from say a PC and the mail gets accumulated on a server. So in POP3 the mail is downloaded to the PC at a time interval which can be specified by the user. POP3 is used when the mail is always read from the same machine, so it helps to download the mail to it in advance.
2. **IMAP(Intermediate Mail Access Protocol)** Here the user may access the mail box on the server from different machines so there is no point in downloading the mail before hand. Instead when the mail has to be read one has to log on to the server. (IMAP thus provides **authentication**) The mailbox on the server can be looked upon as a **relational database.**
3. **DMSP(Distributive Mail System Protocol)** There are multiple mailboxes on different servers. To read the mail I connect to them from time to time and whenever I do so the mail will be downloaded. When a reply is sent then it will put the message in a queue. Thus DMSP is like a **pseudo MTA.**

Ensuring Network Security

1. How to ensure that nobody else reads your mail?
2. How to be sure that the mail has not been seen by someone else in your name?
3. Integrity ie. mail has not been tampered with
4. Non-Repudiability- means once I send a mail I cannot deny it, and this fact can be proved to a third person
5. Authentication

Mechanisms (PGP & PEM)

PGP (Pretty Good Privacy) - It uses some crytography algorithm to crypt the messages.
Symmetric PGP- The key used for encryption and decryption is the same.
Asymmetric PGP - The key used for encryption and decryption is different.Keys come in pairs - public (known to all) and private. which everybody has. Usually encryption is done using public key so that the private key is used for decryption by the receiver only for whom the message is meant.
Eg. of Symmetric PGP is DES, IDEA
Eg. of Asymmetric PGP is RSA
Symmetric is usually faster In asymmetric PGP there is a problem of key distribution. A hash function is applied on every message so that no two messages hash to the same value. Now the hash function is encrypted . If the hash function of source and destination matches then No tampering. If the key for encryption is private then not everybody can generate the message although anyone can read it . So this scheme **lacks privacy** tackles the other security issues.

References

1. http://hp.vector.co.jp/authors/VA019876/sokrpg/doc/img/ftpmodel.gif