UNIT -1

NETWORK MODELS

COMPUTER NETWORK:

A computer network is a number of computers connected by some communication links.

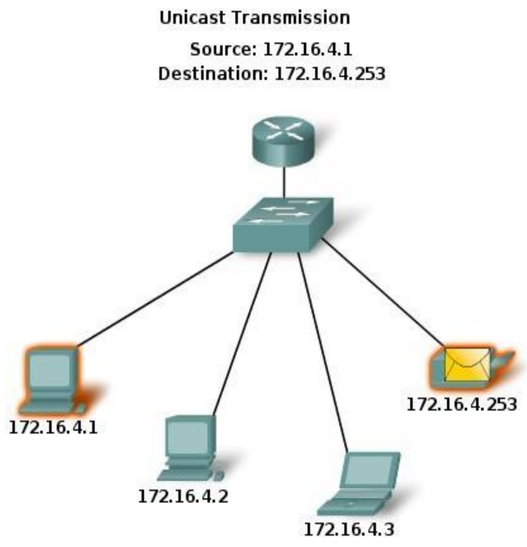
Communication links may be wired or wireless.

Two computer connected to the network can communicate each other through the other nodes if they are not directly connected.

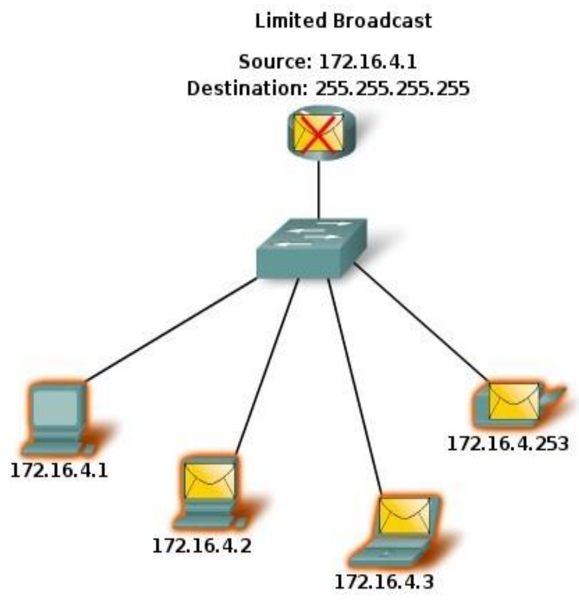
MODES OF COMMUNICATION:

1. Unicast
2. Broad cast
3. Multicast

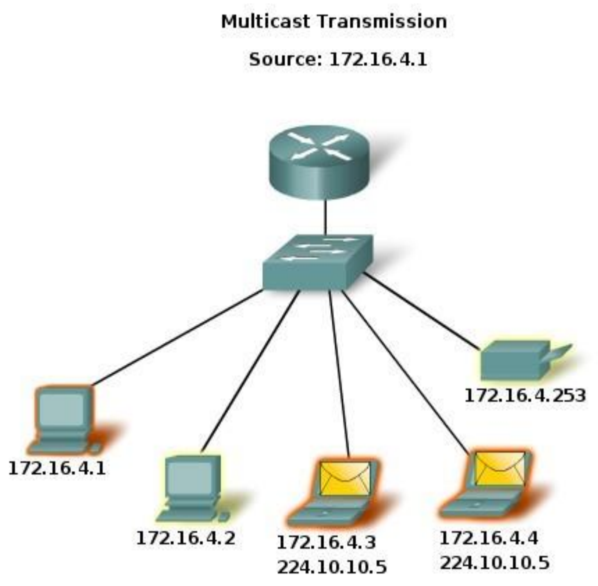
1. UNICAST: The process of sending a packet one host to individual host.



2. BROAD CAST: The process of sending a packet from one host to all other host present in the network.



3. MULTICAST: Sending a packet from one node to selected nodes.



MULTIPLEXING: Many number of signal are combined into a single signal and transmitted over the network.

SIMPLEX: One way communication.

HALF DUPLEX: Two way communication but either sender or receiver will work the same time.

FULL DUPLEX: Transmits and receives the data simultaneously.

NETWORK TOPOLOGY: The arrangement of nodes to form a computer network. Generally topology means layout for a system.

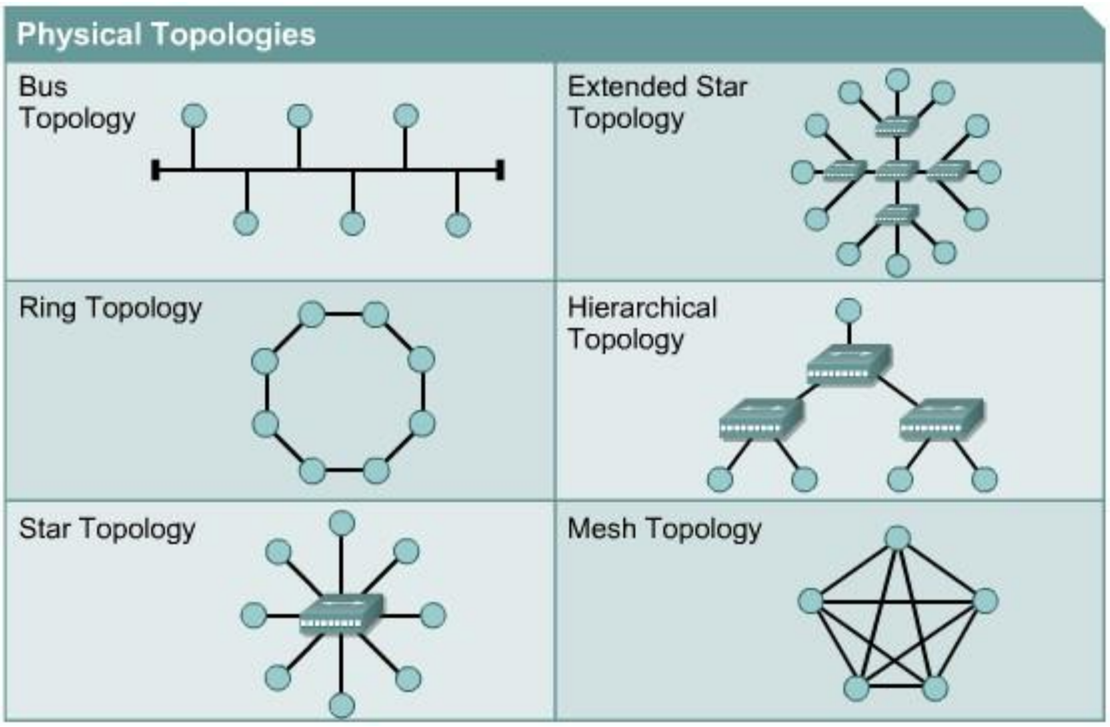
It can be viewed as 1. Physical topology
2. Logical topology

Physical topology: The placement of various modes to make a network.

Logical topology: It deals with the data flow from one system to other system in the network.

TYPES OF NETWORK TOPOLOGY:

1. BUS
2. RING
3. STAR
4. MESH
5. HYBRID



BUS TOPOLOGY: All data transmitted over the common transmission medium and is able to be received by all nodes present in the network.

A signal can be travelled in both directions.

ADVANTAGES:

1. Only one wire it is less expensive
2. If one mode failures it will not affect entire network.
3. Suited for temporary networks.

DISADVANTAGES:

1. If the transmission medium breaks the entire network will be collapsed.
2. No security
3. Limited cable length.

RING TOPOLOGY:

A Ring topology is a bus topology in a closed loop

Two connections are possible to its nearest neighbours.

It is unidirectional

Sending and receiving the data takes places with the help of token.

ADVANTAGES:

1. Performances better than bus topology.
2. All nodes have equal access.

DISADVANTAGES:

1. It takes long time process
2. It is unidirectional
3. No security

STAR TOPOLOGY: Every node is connected to a central node called hub or switch

It is centralized management

All traffic must pass through switch or hub

ADVANTAGES:

1. Easy to design and implement
2. Centralised administration

DISADVANTAGES:

1. Increased cost due to switch or hub
2. Single point of failure affects the whole network.

MESH TOPOLOGY:

Each node is connected directly to every other node present in the network

ADVANTAGES:

1. Fault tolerance
2. It is reliable

DISAVANTAGES:

It is very expensive an in practical for large networks

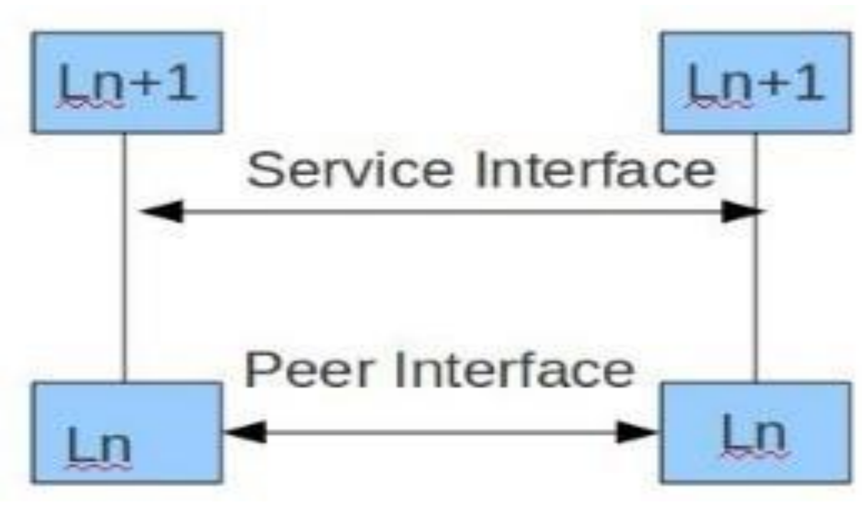
HYBRID TOPOLOGY:

The topology which combines two or more existing topology called as hybrid topology.

PROTOCAL: A Protocol defines the format and the order of messages exchanged between two or more computer

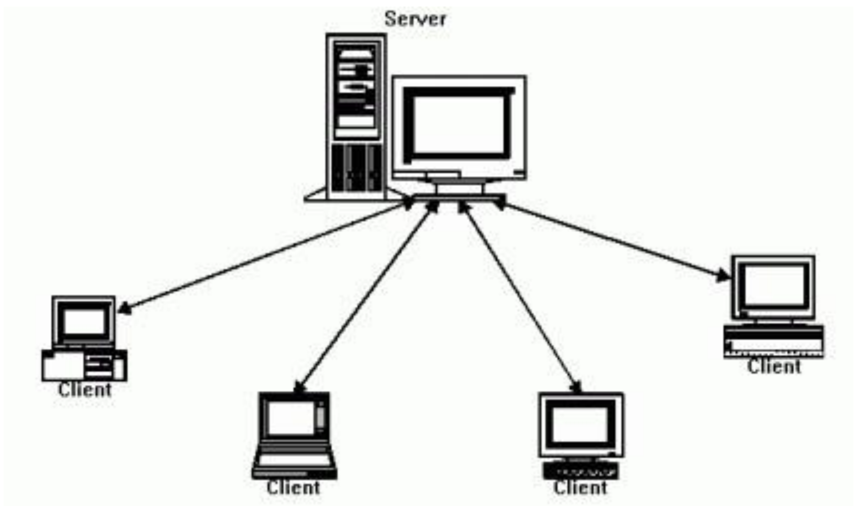
Each protoc0l has two interfaces 1. Service interface

2.peer to peer interface



SERVICE INTERFACE:

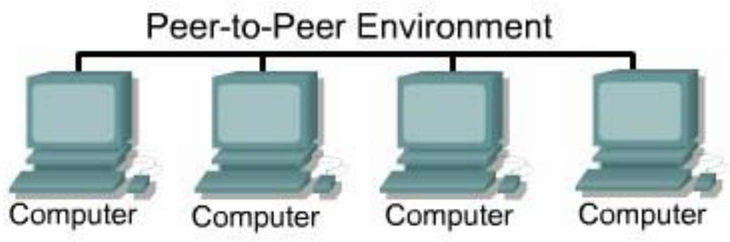
The interface which makes with in the host called service interface



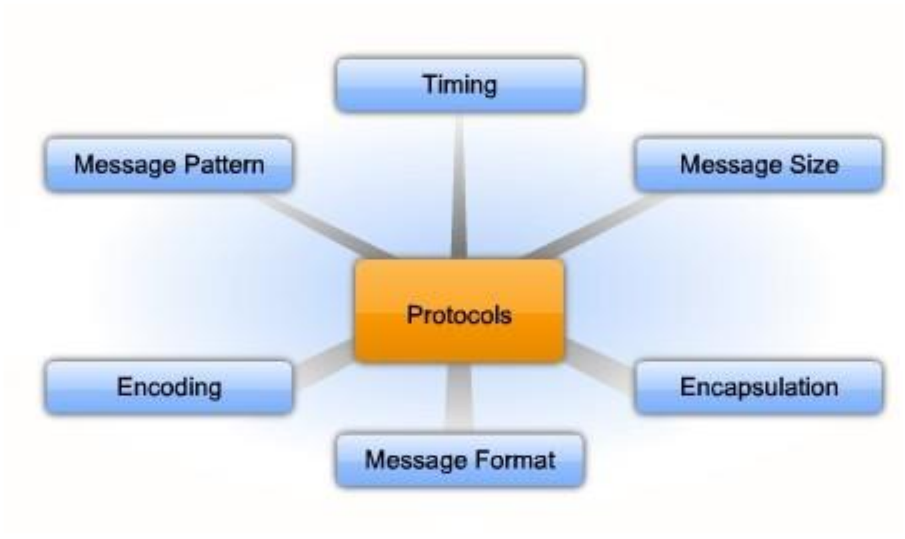
PEER TO PEER INTERFACE:

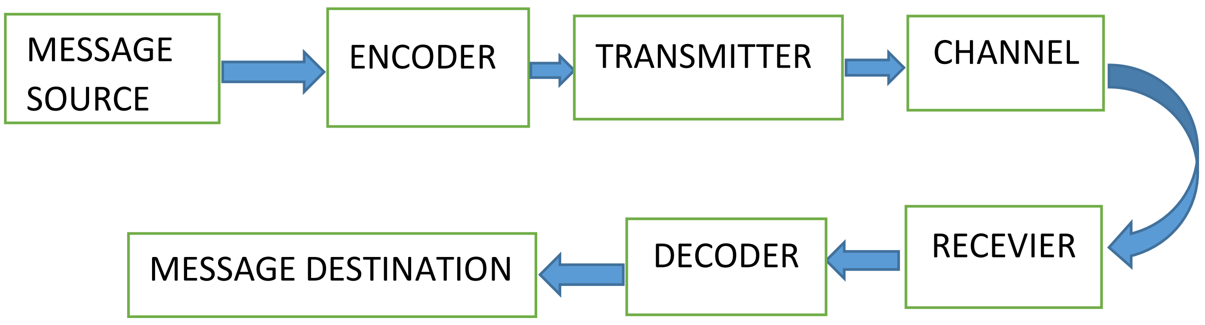
The interface which make connection between two host

Protocal is the building block of network architecture.



PRINCIPLE OF COMMUNICATION





The primary purpose of any network is to provide a communication.

All communication methods have three elements in common

1. TRANSMITTER
2. RECEIVER
3. CHANNEL

TRANSMITTER:

The first element called message source or people or electronic devices that need to communicate a message to other individual or devices

Second element destination it receives the message and interpet

The third element called channel provides the path way over which the messages can travel

Protocal defines the details of how the message is transmitted and delivered

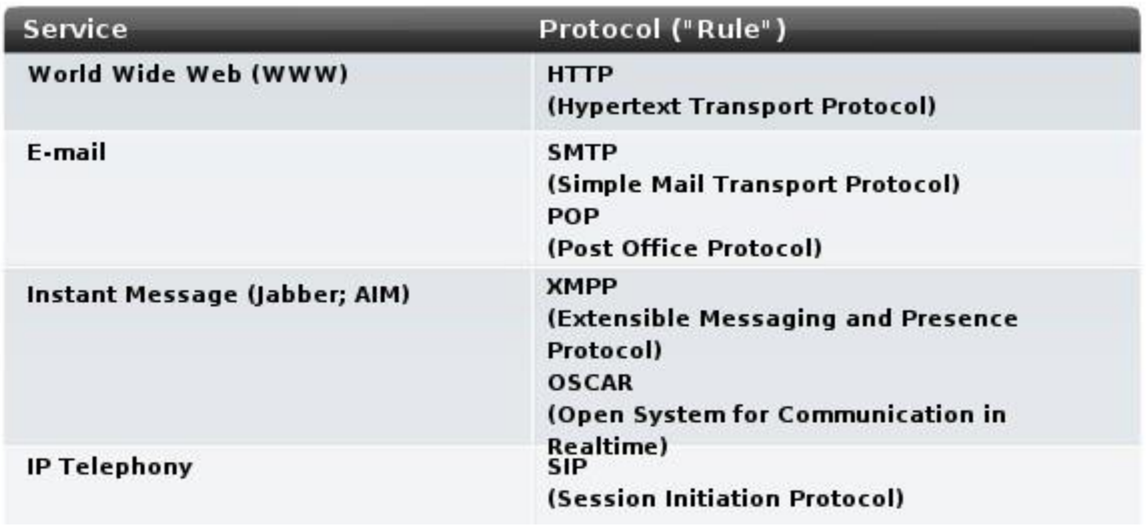
1. Message format
2. Message size
3. Timing
4. Encapsulation
5. Encoding
6. Standard message pattern

MESSAGE FORMAT: Message that is sent over a computer network follows specific format for it to delivered and processed

MESSAGE SIZE: When a long message is sent from one host to another over a network it is necessary to break the messages into smaller pieces

MESSAGE TIMING: One factor that affects how well a message is received and understood within a timing

Ex: response timeout



Service	Protocol ("Rule")
World Wide Web (WWW)	HTTP (Hypertext Transport Protocol)
E-mail	SMTP (Simple Mail Transport Protocol) POP (Post Office Protocol)
Instant Message (Jabber; AIM)	XMPP (Extensible Messaging and Presence Protocol) OSCAR (Open System for Communication in Realtime)
IP Telephony	SIP (Session Initiation Protocol)

ENCAPSUALTION: Just as a letter it is encapsulated in an envelope for safe delivery

Each computer message is encapsulated in a specific format called frames.

Before it is set over the network

A frame act like an envelope which has source and destination address

ENCODING:

The process of putting a sequences of characters into a specialized format (bit) for efficient transmission

Msg sent across the network are the first converted into bits by the sending host

Each bit is encoded into a pattern of sounds, lightwave depending on network media over which the bit or transmitter

The destination receives and decodes the signal in order to interpret the original message

TYPES OF NETWORK

There are three types of networks

LAN – Local area network

MAN – Metropolitan area network

WAN- Wide area network

LAN – LOCAL AREA NETWORK

It is connection of computers in a limited range.

Those are building, colleges, within bank

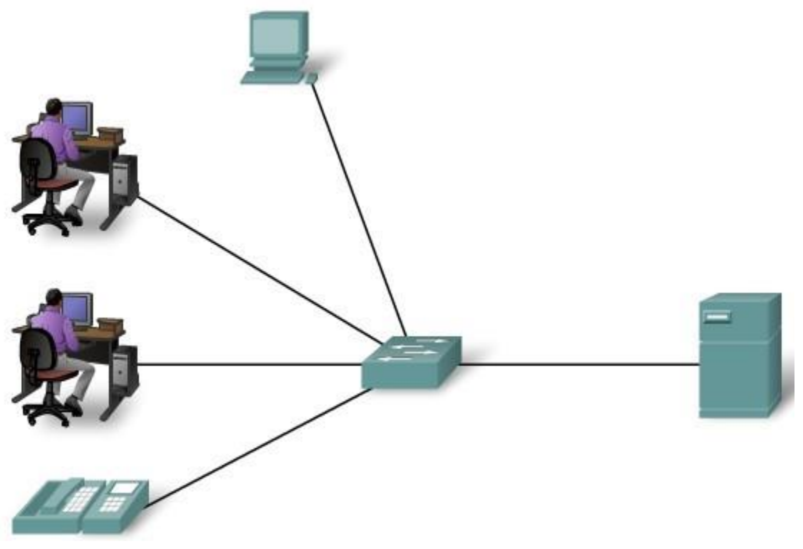
LAN is the sharing of information of resource

Resource – printing $\rightarrow$ software application $\rightarrow$ data

LAN is an high security network

DISADVANGES OF LAN:-

One of the major drawback of LAN is lack of privacy


A network serving a home, building or campus is considered a Local Area Network (LAN).

MAN – METROPOLITAN AREA NETWORK

MAN is a connection of different LAN and WAN

It is used in cities [within cities]

Best example for MAN banks [connection of all branches in a city]

MAN is the sharing of data network

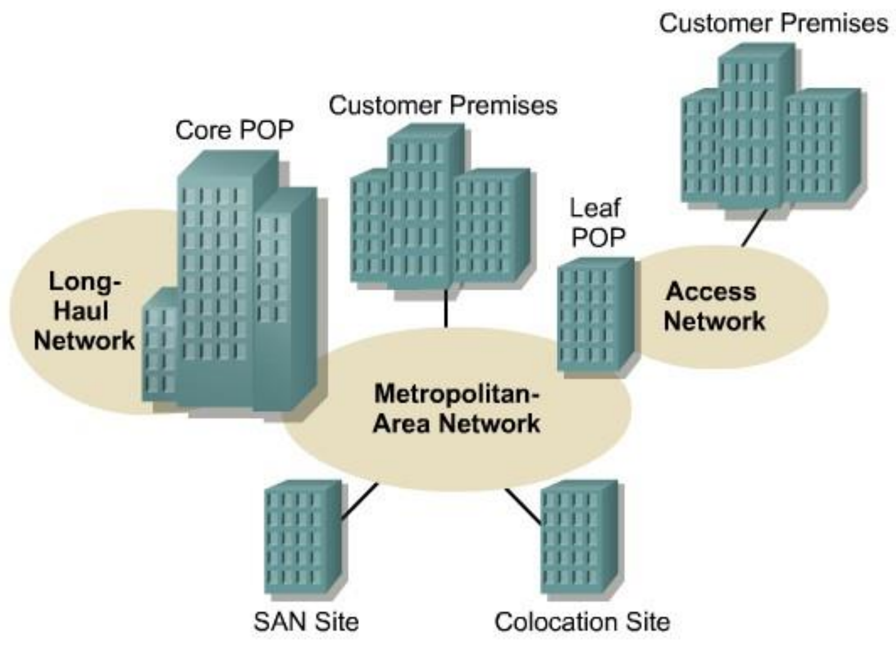
It is used for data centralized.

DISADVANTAGES:-

The main disadvantage of MAN is less security

It is connected by the fiber optics or cables

It required more cables



It is a connection among different cities and states

In this transmission media will be satellite, telephone cabled

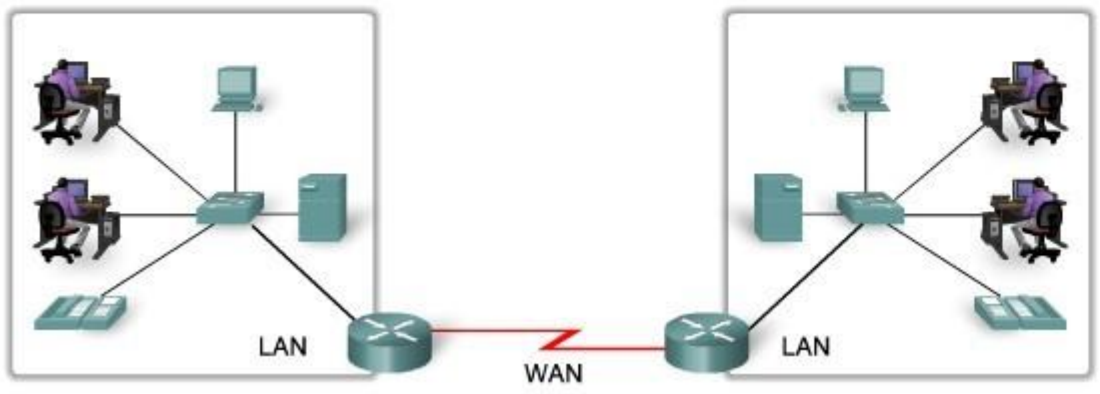
WAN is the connection of MAN’s

In this data sharing takes place by the roots

WAN is less security and need to fire wall

Best example for WAN is internet

LANs separated by geographic distance are connected by a network known as a Wide Area Network (WAN).



LAN Vs WAN

LAN	**WAN**
Connects host within a relatively small geographical area. • Same Building • Same room • Same Campus	Hosts may be widely dispersed. • Across Campuses • Acorss Cities/countries/continent
Faster	Slower
Cheaper	Expensive
Under a control of single ownership.	Not under a control of a single person.
Typical Speeds: 10 Mbps to 10Gbps	Typical Speed: 64 Kbps to 8 Mbps

STANDARDS

It provide guidelines to manufactures, govt agencies and other service provide to ensure the kind of inter connectivity necessary in today’s market places and international communication standard agreed upon rules

CATEGORIES OF STANDARD

DEFACTO:

By fact or approved by the organization (HTTP)

DE-JUNE:

By the law or government

Most of the common standards are de-June standards

STANDARD ORGANIZATIONS:

ISO (international organization for standard)

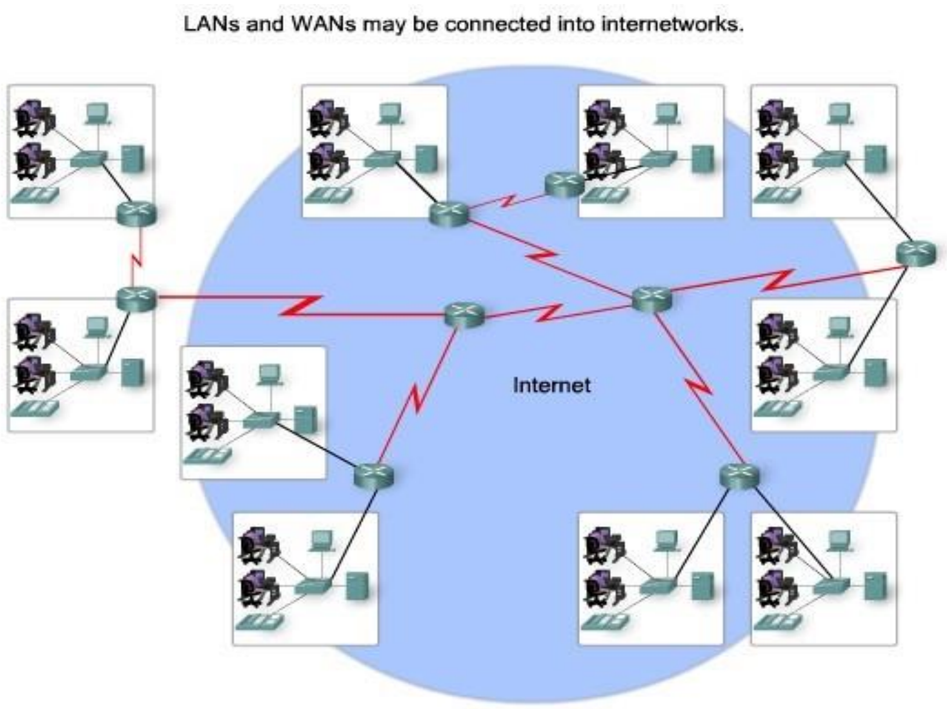
ITU-T (international telecommunication union – telecommunication standard sector)

ANST (American national standard institute)

IEEE (institute of electrical and electronic engineer)

EIA (electronic industry association)

INTERNET:-



Internet is a network of networks i.e used to interlink many different types of computers allows the world.

Or

When two or more networks to connected they are called internet work of internet.

It is defined as information super high o access information over the web world wide global s/m interconnected computer network

Internet uses standard internet protocols TCP/IP every computer in internet is identified by unique IP address. IP address is 110.22.33.44 which identifies computer (DNS-Domain name server) is used to give name to the IP address. So that the user can locate a computer by a name.

EVOLUTION:

The concept of the internet was originated in 1969 has undergone several technologies and infrastructural changes. The origin of the internet is from ARPANET [advanced research project agency network]. ARPANET developed by unitedstates. ARPANET provides communication among various bodies of government.

ADVANTAGES OF INTERNET:

Social network: used in fb, twitter,yahoo,google

Education and technology: various topics

Entertainment: online games, online TV, songs, videos

Online service: net banking, shopping, tickects.

DISADVANTAGES OF INTERNET:

Cyber crimes

Virus attack

Threat to personal information

Spamming i.e unwanted emails

ARCHITECTURE OF INTERNET:

Assume our client his/her clip over a dial up telephone line

The modem is a card within personal computer that converts digital signal into analog signal

These signals are passes to the telephone system again these signals are transferred to POP [point of presence] where they are removed from telephone system and injected into ISP region network

From this, it is fully digital

ISP is local telecommunication the POP will probably be located in telephone switching office where the telephone wires are client terminated If ISP is non local telecommunication it handle over ISP's backbone operator

Regional network consists if inter connected routers in various cities the ISP services provided international backbone network with 1000's of routers connected with high band width fiber optics

Larger co-operations that runs with the server directly connected to backbone.

Backbone is destiny for an internet service provider[ISP],it sends to the closer router and handled . To allow packets to hop between backbone all major backbones connected at NAP

A NAP is full of router Atleast one backbone

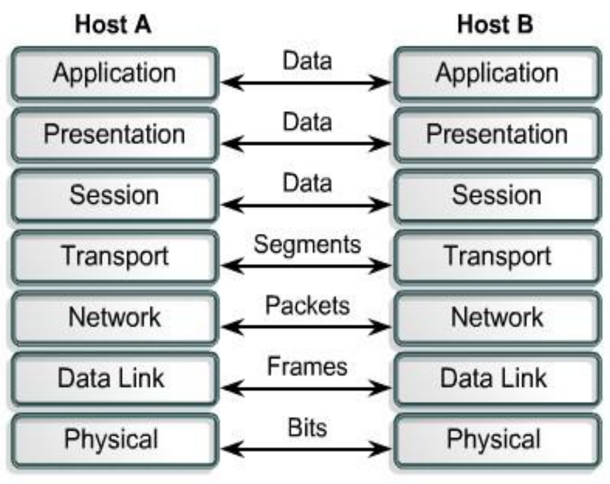
OSI/ISO REFERENCE MODEL:

OSI – open system interconnection.

To make the system compatible to communication with each

ISO has developed a standard i.e OSI

OSI is a seven layer architecture



DESIGN PRINCIPLE OF OSI REFERENCE MODEL:

A Different abstraction are needed a layer has to be create

Each layer should perform a well defined function.

The number of layer should be international standard

The number of layer should not be two layer

PHYSICAL LAYER: Physical medium to transfer bits

DATA LINK LAYER: Error free frames will be transmitted

NETWORK LAYER: Packets will be moved from source to destination.

TRANSPORT LAYER: Using protocol renewable message will be transmitted.

SESSION LAYER: Establishing terminating of session will be created.

PRESENTATION LAYER: Data compression, Encoding, encrypting

APPLICATION LAYER: Service will be provided directly to user.

1. PHYSICAL LAYER:
   It is the lower layer or hardware layer
   It is responsible for physical connection between two Device i.e establishment, maintenances, deactivation
   Information will be in the form of bits.
   In this layer transmission of signals are converted in 0's and 1's.

FUNCTIONS:

BIT SYNCHRONIZATION: By providing clock the synchronization will be achieved between bits.

BITRATE CONTROL: It defined number of bits transmitted per second.

PHYSICAL TOPOLOGY: It is response to know the arrangement of devices in a network either bus, ring, mesh or star.

TRANSMISSION MODE: In which mode data is to be transmitted. It mainly depends up on three types

SIMPLEX

HALF DUPLEX

FULL DUPLEX

SIMPLEX: It is unidirectional.

‘A’ Can send message to ‘B’ cannot send message to ‘A’.

Ex: monitor, keyboard

HALF DUPLEX: Data can be transmitted one after another.

Data cannot be transmitted at a time.

It is a bi-directional.

Ex: walki talkie

FULL DUPLEX: Data can be transmitted at same time from both the sender and receiver it is a bi-directional.

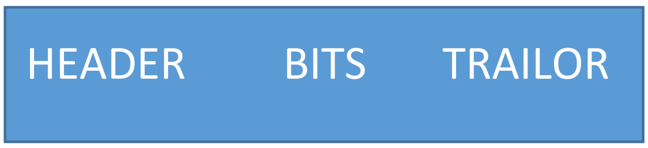
Ex: telephone lines

2. DATA LINK LAYER:

In data link layer data is represented in frames
It is responsible to transmitted error free data and
To get reliable and efficient communication
It is also responsible to define data in a network.

FRAMING:

Raw bits are converted into frames by adding few more bits as header and trailer. This frames will be transmitted from one devices to another devices.



FRAMES

PHYSICAL ADDRESSING: Destination hardware address will be included as harder

ERROR CONTROL: Error control mechanism will be implemented and calculated bits will be added in trailer

If any error or data corrupted the receiver send acknowledgment to retransmitted the corrupted data.

FLOW CONTROL: It means control of data to flow It is maintenances constant bit rate In this data will not get corrupted.

ACCESS CONTROL: If more than one device sharping same communication channel then data link layer (DLL) protocols. Are responsible to identify device which have to get control at a given time.

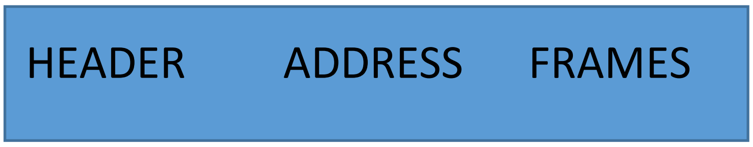
3. NETWORK LAYER:

This is the third layer of OSI model.

In this data is represented in the form of packets
Conversion of physical address to logical address i.e
Address will be included in packets as well as some
Bits will address in header some bits will be
Represented in router for destination.

Routing the packets from source to destination i.e routing will be done by networks layer. Here two different networks will be connected with help of routers. Packets will be transmitted from one network to another network.

LOGICAL ADDRESSING:



PACKETS

Multiple networks are connected with help of routers

Every routers will maintain forward table

INDEX	O/P
S1	R1
S2	R2
S3	R3

Every packets request at the router the header will be check in the index value if the header contain it will router to the packets to destination i.e output link.

INTER NETWORKING: Logical connection between different networks.

FRAGMENTATION: We know that every network will have different and different bandwidth. Based on bandwidth we can send data or packets to the destination. In such a case, avoids collusion in everything will be possible with the help of network layer.

FUNCTIONS

GAURENTEE DELEIVERED

In order of packet:-

What is the order transmitted the receiver receive the same

GUARENTEE MAXIMUM JITLER

The time between two successful transmission at the sender side is equal to destination side.

SECURITY SERVICE

In order to provide security for packets the source will be encrypted and destination will be decrypted

4. TRANSPORT LAYER

Data is represented in segments

It provides logical communication between the application on the different host i.e applications from sender side and application from receiver side. They will not be physical connection between applicant. But they will be logical communication between source and destination.

It is responsible for end to end message delivery

It provides acknowledgement for successful transmission of data

If error is occurred this layer is responsible to retransmit the data

Transmitter layer protocols is implemented in end system but not at routers

Two major protocols

1. TCP [transmission control protocol]
2. UDP[user defined protocol]

Provides servers to this transport layer

SERVERS:

End to end delivery

Reliable delivery [without any loss of information,without any error , without any interruption

ERROR CONTROL:

If any error during transmission that cannot be detected in datalink layer can be overcome in transmit layer

SEQUENCE CONTROL:

In this message is represented in segment. The message is divided into segments and each segment has one sequences number at the sender side.

We are sending number segment and at receiver side they have receiver all the transmitted segment

Sequences control two types

LOSS OF DATA

REASSEMBLING

LOSS CONTROL:

To avoid the data loss the loss control is used at the receiver to retransmit the signal.

DUPLICATION CONTROL:

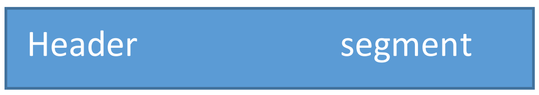
Sequences number provided to all the segment if any segment with the same sequences number that one automatically duplicate segment has received this layer avoid duplication segment.

FLOW CONTROL:

If the receiver having over loaded some segments are missing

Again the receiver request to retransmit the segment such type of avoidness can be controlled by flow control.

Addressing for every segment there will be header that header will be having some part address



It transmit the required application at receiver side

CONNECTION ORIENTED

CONNECTION ESTABLISHMENT

DATA TRANSFER

CONNECTION TERIMATION

Acknowledgement is given to free successful transmission of

Data

CONNECTION LESS: Data can transfer

No acknowledgement

5.SESSION LAYER:

Session is responsible to keep the data from each section to separate

It is responsible for setting up, managing and tearing down session

It is also provides dialog control and coordinate communication between the system

FUNCTION:

Section establishment, maintenances and termination

SYNCHORIZATION:

To avoid data loss

Sending some check points and if any data is loss will not go to the first packet

And continue with the packets were error is occurred

Ex: 100 pages of data from source to destination

100 pages of data, check point = 20 pages

After transmission 20 pages from destination will get the acknowledgment for those 20 pages

If there is source receives the acknowledgment number that means data is sequences required at the destination

If there is error in transmission at 65 pages can check from the 61 pages.

DIALOG CONTROL:

Allow two system to start communication either half duplex or full duplex.

6.PRESENTATION LAYER:

Is responsible for data translation and encoding

It will take data from application layer and translation into a generic format for transfer across the network

This layer also involved in data compression, decompression

Encryption and decryption.

FUNCTION:

TRANSLATION: Changing the one format of data into another format.

Encryption:

Sender → data → encrypted → cipher text

receiver → cipher text → decryption → plane text

COMPRESSION: The size of data will be reduced in Compression algorithm reduced the size of data and it will be transmitted to the network reduced the size of data.

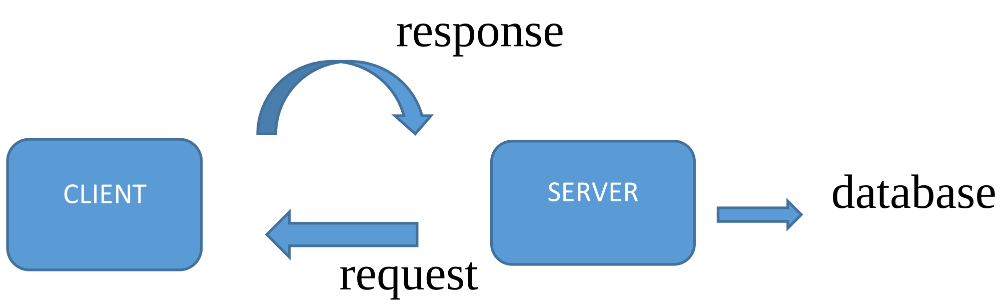
6.APPLICATION LAYER:

It is top layer of OSI model

It is a layer through which user can interact

It provide service to the user

Application layer program are based on client



Client is sending to the server (i.e opening of mail) giving user name, password the server will check with the data base when every server sending the response to client

There are different application to perform in system

Ex: web browser, messenger

Produce the data and that will be transmitted over a net work through all 6 layer

SERVICES:

File transfer, access and management

E-mail services

File or directory services

In this data actually will be transmitted for this layer

This layer provides services to display information in the required format

TCP/IP REFERENCE MODEL:

It is a implementation of OSI reference model. It consist of

4 layer

1. APPLICATION LAYER
2. TRANSPORT LAYER
3. INTERNET LAYER
4. HOST TO NETWORK LAYER

HOST TO NETWORK LAYER: It is a combination of datalink and physical layer

It is used for physical transmission data

It also defined the protocols connected to the host.

This protocols are different from one network to another network.

INTERNET LAYER: It is similar to network layer

In this packets deliver from source to destination

ROUTING: Packets will be routed from one network to another networks.

CONGESTION CONTROL (Bulk of data):

To avoid this congestion this layer is responsible

Each layer has protocol the main protocol for internet layer i.e IP (Internet protocol).

It is used to transfer packet from source to destination.

This layer is responsible to transmit packet independently to source host to destination host.

TRANSPORT LAYER:

It is responsible for segmenting splitting of data

The data will be divided into different segments splitting of

Data based up on bandwidth.

The data will be divided completely in transport layer

It also desire to send the data either in single path or multiple path

APPLICATION LAYER:

It is a combination of session and presentation layer

It act as a interface between host and server provides by the transport layer

It includes high level protocols

TELNET-Tele type network

It allow user on one machine to log in to another machine and work there

It is a two way communication

FTP- File transfer protocol

Transmitting the file data

SMTP- Simple mail transfer protocol

It transport electronics mails

DNS- Domain name system

Mapping host names on their address

APPLICATION LAYER	TELNET	FTP	SMTP	DNS
TRANSPORT LAYER	TCP		UDP	
INTERNET LAYER	IP			
HOST TO NETWORK LAYER	ETHER NET	FRAM E RELAY	TOKEN	RING

DIFFERENCE BETWEEN OSI AND TCP/IP:

OSI		TCP/IP	
1	Reference model	1	Implement of OSI
2	7layer	2	4 layer
3	Session and presentation layer are separated	3	It combines both session and presentation
4	Protocol are independent	4	Protocol are dependent standard
5	Supports to connection less and connection oriented	5	Supports only connection less

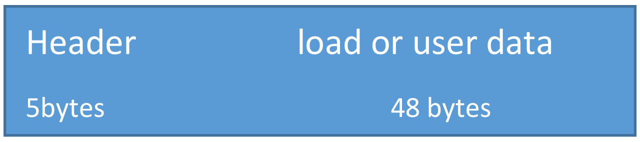
ATM:

ATM is also called as cell relay

Asynchronous in ATM devices do not send or receives information at affixed speed or using a timer but transmitted

Speed based on hardware and information flow

ATM transfer information fixed size unit each cell consist of 53 bytes



ATM CELL FORMAT

Cells are transmitted asynchronously

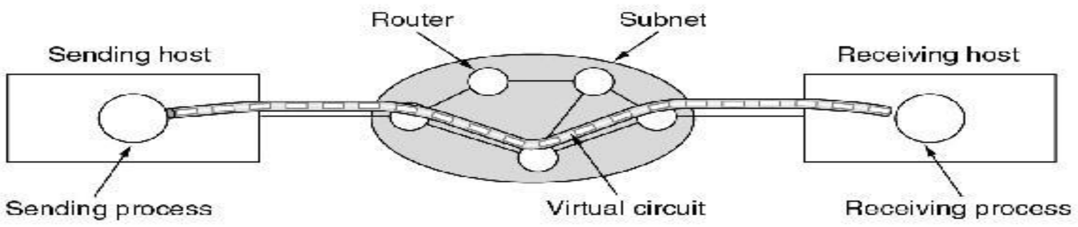
ATM network is connection oriented making an ATM call request 1st sending a message to setup a connection sub sequentially all cell follow the same path destination.

It can handle both constant rate traffic and variable length traffic

Thus it can carry multiple type of traffic with end to end quality of server.

ATM virtual circuit:

ATM Virtual Circuits



A virtual circuit.

ATM connection are connected oriented

Sending data requires 1st sending packets to setup the connection this setup packet went to the subnet on the path to all the routers no direction electrical connection from end to end connection are often called virtual circuit.

They are 2 types of virtual circuit

1. Permanent virtual circuit (PVC)
2. Switch virtual circuit (SVC)

PVC:

PVC are connection between 2 nodes that are establish statically by the network

Connection are long period of time

Any failure of this connection cannot be automatically connected by the network requires human help

This is one of the main drawback in PVC

SVC:-

ATM user use second type of connection SVC or temporary connection created for the purpose of information transfer

There are 4 steps to establish SVL connection

1. Call set up
2. Data transfer
3. Ideal
4. Call termination

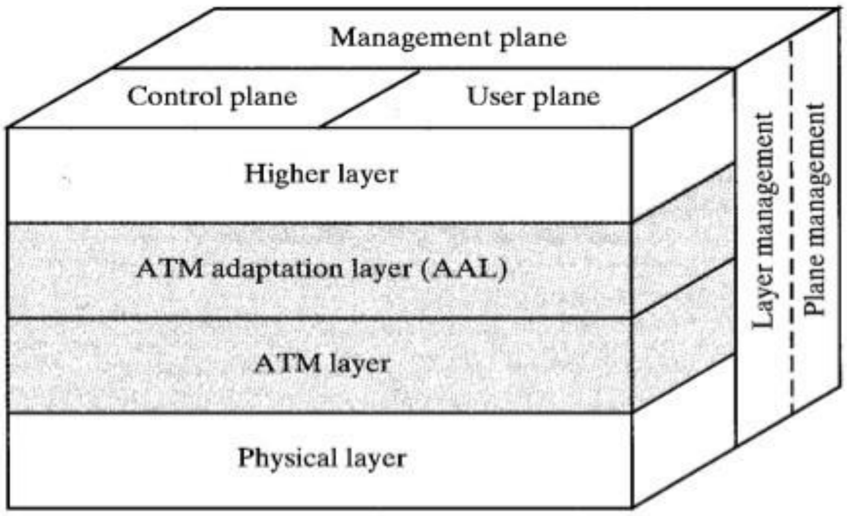
ATM REFERENCE MODEL

ATM has its own reference model difference from OSI model and also different from TCP/IP model

It consist of three layer and 3 dimensional

1. Physical layer

2. ATM layer
3. ATM adaption layer



CS – convergence sublayer

SAR – segmentation and reassemble sublayer

TC – transmission convergence sublayer

PMD – physical medium dependent sublayer

PHYSICAL LAYER

It deals with physical medium i.e voltage bit timing various other issues

ATM does not have set of rules

ATM clues can be send on wire or fiber by themselves

ATM designed to be independent of transmit media

ATM LAYER

It deals with the cell and cell transfer. It defines the layout of the cell and what header field means.

It deals with the establishment and of virtual circuit conjunction control is located

ATM ADAPTATION LAYER

ATM layer allows user to send packets layer than a cell

ATM adaptation layer segment. This packet transmitted to cell individually and reassemble them at other end

USER PLANE

Deals with data transport flow control error correction and other user function.

CONTROL PLANE: It helps in connection management layer and plane management.

LAYER AND PLANE MANAGEMENT

It related to the resource and inter layer management

Physical and ATM adaptive layer are divided into two sublayer

PMD [PHYSICAL MEDIUM DEPENDENT]:-

It move bits ON and OFF and handle the bit timing

TC [TRANSMISSION CONVENGENCE]

It covert bit stream into a cell stream for the ATM layer

SAR [SEGMENTATION AND REASSEMBLE]

Breaks the packets into cells or transmission side and put them again destination

CS [CONVENGENCE SUBLAYER]

ATM system offer different kind of services for different applications

Ex:- File transfers

ATM Layer	ATM sublayer	FUNCTION
AAL	CS	Providing the standard interface (convergence)
	SAR	Segmentation and reassembly
ATM		Flow control Cell header generation/extraction Virtual circuit/path management Cell multiplexing/Demultiplexing
PHYSICAL	TC	Cell rate decoupling Header checksum generation and verification Cell generation Packing/unpacking cells from the enclosing envelope Frame generation
	PMD	Bit timing Physical network access

DATA LINK LAYER

➤DESIGN ISSUSES:

- Error detection and correction
- Stop and wait protocol
- Sliding window protocols

Eg : data link protocols

➤THE MAC SUB LAYER:

- The channel allocation problem
- Multiple access protocols
- Wireless LANS-bridges-FDDI

DESIGN ISSUES

- Second layer of OSI model
- Most complicated layer and has complex functionalites

DLL IS DIVIDED INTO 2 SUBLAYERS

- Logical link control: It deal with protocols, flows, error controls
- Media access control: It deal with actual control of media

FUNCTIONALITY OF DATA LINK LAYER

- Farming
- Addressing
- Synchronization
- Error control
- Flow control
- Multi access

➤FRAMING:

- It takes packets from internet layer and encapsulates them into frames
- It sends each frame bit by bit to receiver.

➤ADDRESSING:

- DLL provides addressing destination hardware address will be included in header

➤SYNCHRONIZATION:

- To avoid data loose sending some check point and if any data is lost will not go with the first frame and continue with the same frame where error is occur

➤ERROR CONTROL:

- It can occur during transmission that can be detect in data link layer. Receiver sends acknowledgement to transmit the corrupted data

➤FLOW CONTROL:

- When a data frame is send from one host to other over a single medium, it is required that a sender and receiver should at a same speed
- It sender is sending to fast the receiver may be over loaded

➤MULTIACCESS:

- Multiple user can access a shared media among multiple system

ERROR DETECTION AND CORRECTION

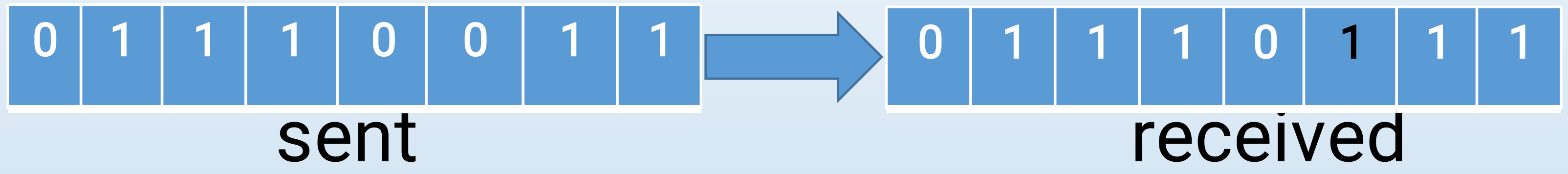
There are many mesons such as noise, cross-talk etc. which may help data to get corrupted during transmission. DLL uses some error control mechanism. To understand how errors is controlled, it is essential to known what types of errors may occur.

TYPES OF ERROR

THREE TYPES OF ERRORS

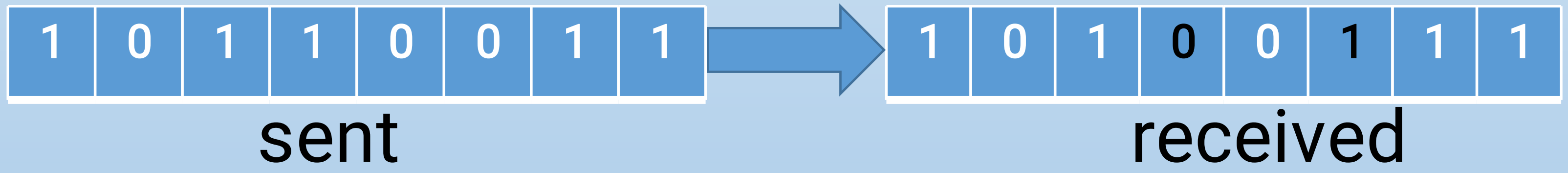
- Single bit error
- Multiple bit error
- Burst error

➤SINGLE BIT ERROR:



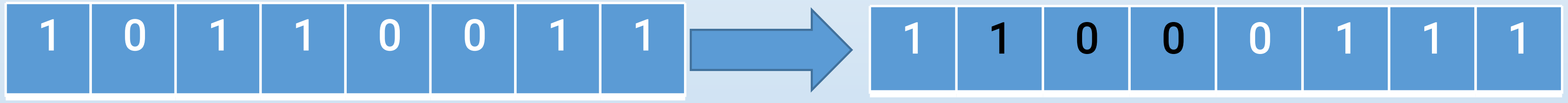
In a frame, there is only one bit, anywhere through, which is corrupt.

➤MULTIPLE BIT ERROR:



Frame is received with more than one bits in corrupted state

➤BURST ERROR:



sent received

Frame contains more than or consecutive bits corrupted

Error control mechanism may involve two possible ways

- Error detection
- Error correction

ERROR DETECTION

Errors in the received frames are detected by means of

- Parity check
- Cyclic redundancy check
- Check sum

PARITY CHECK:

One extra bit is sent along with the original bits to make number of is either even in case of even parity or odd in case of odd parity.

- Count of is should be even

$$\begin{tikzcd} 1\ 0\ 1\ 0\ 0 \arrow[r] & 10100\underline{0} \arrow[r] & 101000 \\ \text{sender data} & \text{even parity bit} & \text{receiver} \end{tikzcd}$$

This is even parity. The no. ‘1’ is in odd

- Odd parity:

$10101 \rightarrow 10100\underline{1} \longrightarrow 101011$

Sender data odd parity

The no. of ‘1’ is in even.

- Multiple error bits are not rectified by parity check

CYCLIC REDUNDANCY CHECK

- Data – 1 0 1 1 0 1
- CRC generator – 1 1 0 1
- CRC bits = n-1 =3

CHECK SUM:

In check sum error detection scheme, the data is divided into ‘k’ segments each of ‘m’ bits

- In the gender’s end the segments are added using 1’s complement arithmetic to get the sum. The sum is complemented to get check sum
- The check sum segment is sent along with the data segments
- At the receivers end, all received segments are added using is complement arithmetic to get the sum. The sum is complemented

- If the result is zero the received data is accepted, other wise discarded original data

ERROR CORRECTION TECHNIQUES:

Error correction techniques find out the exact number of bits that have been corrupted and as well as their locations.

Two principle ways

- Backward error correction (retransmission)
- Forward error correction

BACKWARD ERROR:

- Receiver detects an error in the incoming frame, it requests the sender to retransmit the frame
- Retransmitting is not expensive as in fiber optics.

FORWARD ERROR CORRECTION:

- Receiver detects some error in the incoming frame, it executes error correction code that generates the actual frame
- It there are too many errors, the frames need to be retransmitted.

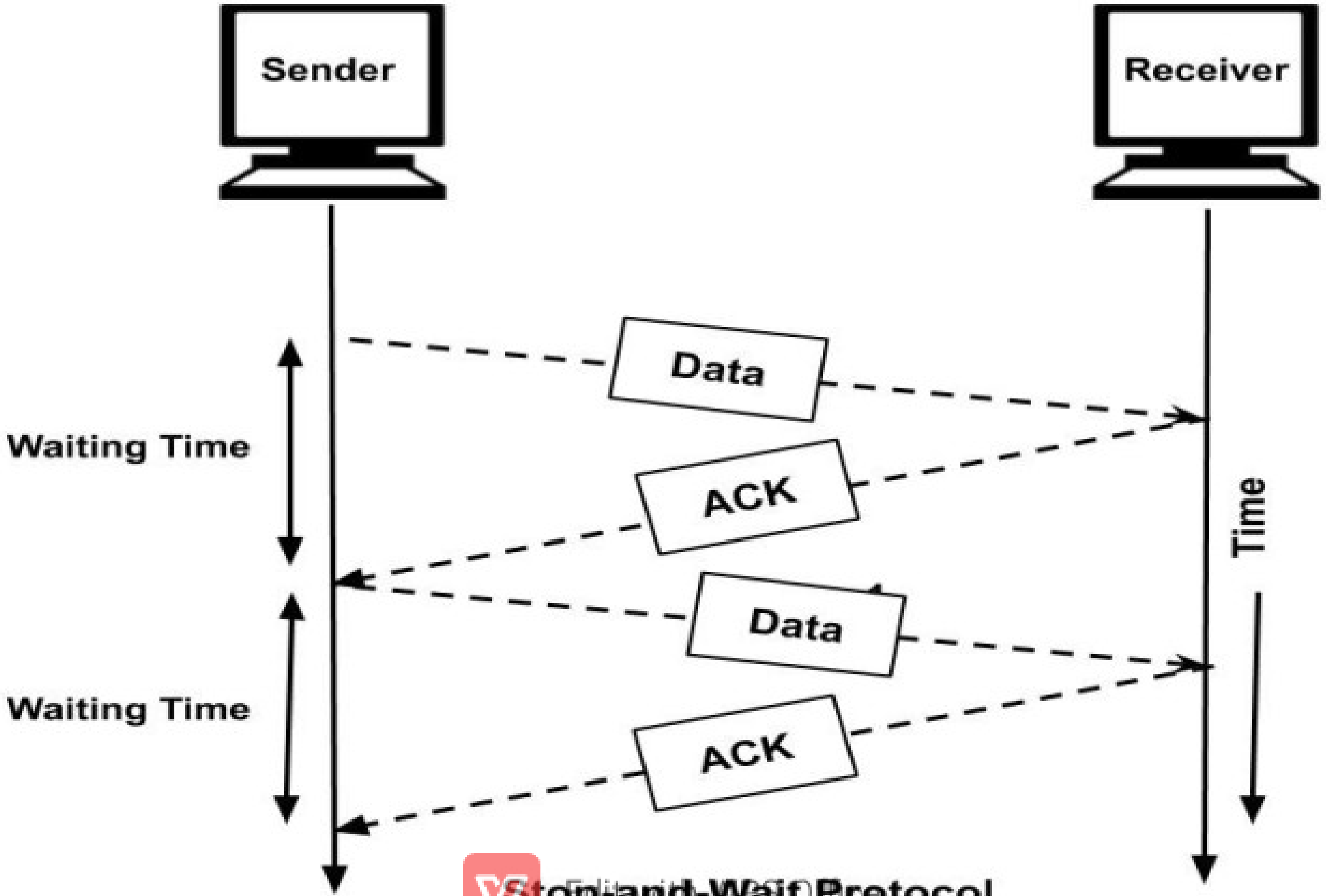
The two main error correction codes are

1.Hamming codes 2. binary convolution code

STOP AND WAIT

Two types of mechanism can be implemented to the control the flow

This flow control mechanism forces the sender after transmitting a data frame to stop and wait until the acknowledgement of the data frame send is received.



Stop-and-Wait Protocol

REQUIREMENTS FOR ERROR CONTROL MECHANISM

Error detection

Positive ack

Negative ack

Retransmission

- ERROR DETECTION:

The sender and receiver, either both or any, must know that there is some error in transit.

- POSTIVE ACK:

When the receiver receives the correct frame, it should acknowledgement it

- NEGATIVE ACK:

When the receiver receives a damage frame or duplicate frame,

It send a NACK. Back to the sender and the sender must retransmit the correct frame.

- RETRANSMISSION:

The sender maintains a clock and sets a time out period

It may acknowledgement of a data frame previously transmitted does not arrive before the time out the sender retransmit the frame, thinking the frame or it's acknowledgement is lost in transit

There are three types of techniques available in DLL

- Stop – and – wait (ARQ- Automatic repeat request)-
- go-back-N ARQ
- Selective repeat ARQ

SLIDING WINDOW:

In this flow control mechanism both sender and receiver agree on the number of data frames after which the acknowledgement should be sent stop and wait flow control mechanism. Wastes resource this protocol tries to make use of underlying resource as much as possible.

1. go-back-N
2. Selective repeat

Send multiple frames at a time

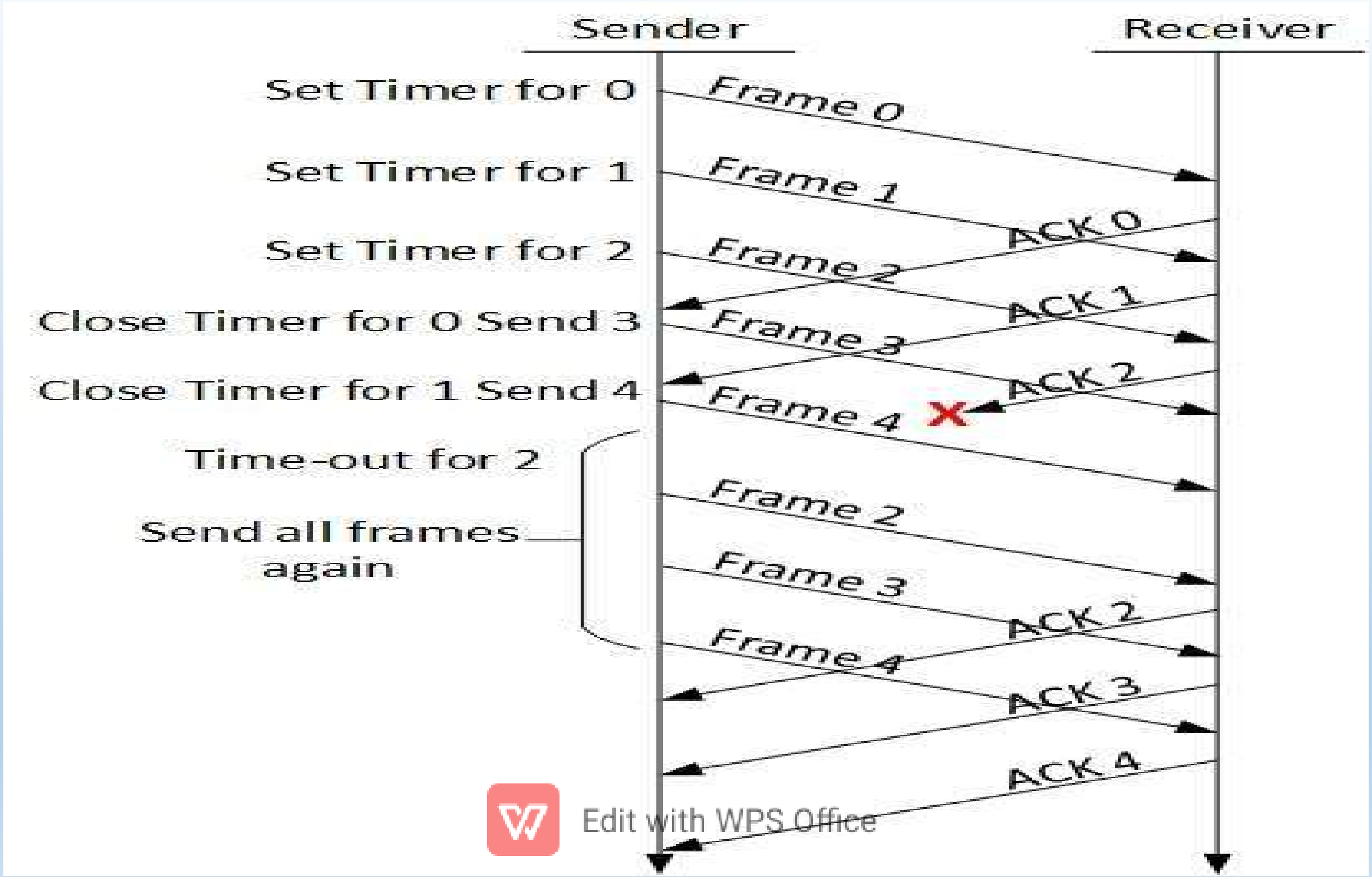
No of frames to be send is based on window sizes

Each frame is number sequences number.

- Go-back-N ARQ:

Stop and wait ARQ mechanism does not utilize the resources at there best.

When the acknowledgement is received the sender sits ideal and does nothing, in go-back-N ARQ method, both sender and receiver maintain the window


Sender
Receiver
Set Timer for 0
Frame 0
Set Timer for 1
Frame 1
ACK 0
Set Timer for 2
Frame 2
ACK 1
Close Timer for 0 Send 3
Frame 3
ACK 2
Close Timer for 1 Send 4
Frame 4
Time-out for 2
Frame 2
Send all frames again
Frame 3
ACK 2
Frame 4
ACK 3
ACK 4

- The sending window size enables the sender to send multiple frames without receiving the acknowledgement of the previous one. The receiving window enable the receiver to receive multiple frames and acknowledge them. The receiver keeps track of incoming frames in sequences number.
- When the sender sends all the frames in window, it checks up to what's sequences number it has receive positive acknowledgement. If all frames are positively acknowledged, the sender sends next set of frames. If sender finds that it has received "NACK" or has not receive any ack for a particular frame, it retransmits all the frames after to which does not receive any positive ack.

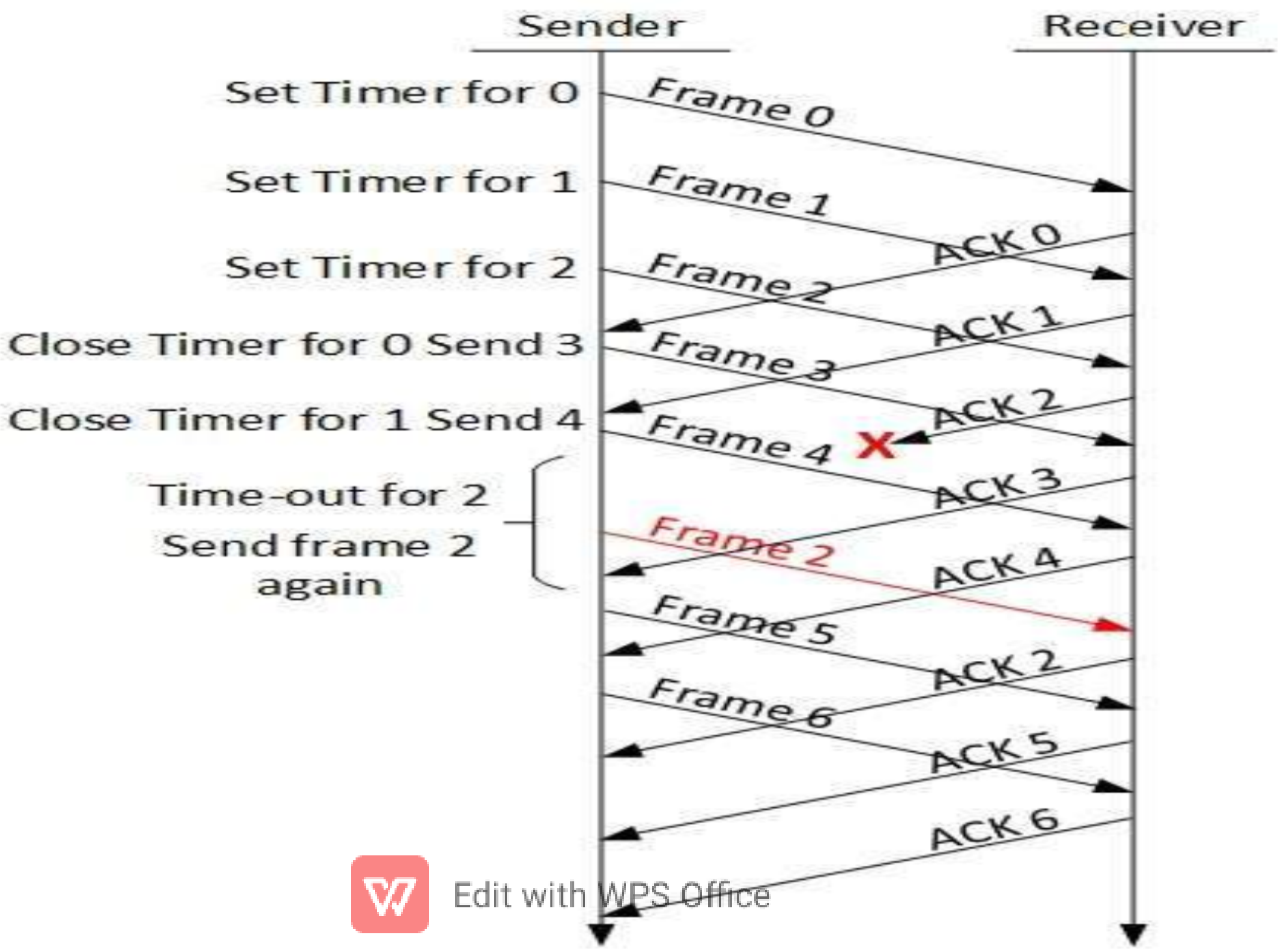
- SELECTIVE REPEAT ARQ:

In Go-back-N ARQ, it is assumed that the receiver does not have any buffer space for it’s window. Size and has to process each frame as it comes. This enforces the sender to retransmit

All the frames which are not acknowledged

In selective repeat ARQ, there is receiver by keeping track of sequences number, buffer. The frame in memory and send NACK for only framed which is missing or damaged

- The sender in this case sends only packet for which NACK is received.


Sender
Receiver
Set Timer for 0
Frame 0
Set Timer for 1
Frame 1
ACK 0
Set Timer for 2
Frame 2
ACK 1
Close Timer for 0 Send 3
Frame 3
ACK 2
Close Timer for 1 Send 4
Frame 4
X
ACK 3
Time-out for 2
Send frame 2 again
Frame 2
ACK 4
Frame 5
ACK 2
Frame 6
ACK 5
ACK 6

- Example data link protocols:

1. HDLC [High level data link control]
2. The data link layer in the internet

HDLC :

Derived from SDLC used in IBM Main framing

[Synchronous Data link protocols]

Bit oriented protocol used bit stuffing

Reliable protocol / selective repeat or go-back-N

Full duplex communication

There are three different classes or Frames used in HDLC

❖ Information frames: which carry actual information such frames can piggy back Ack.

- Supervisory frames:

Which are used for error and flow control purpose and hence contain send and sequence numbers

- Unnumbered frames:

Used in link set up and disconnection

HDLC Frames types:

1. Information frames
2. Supervisory frames
3. Unnumbered frames

FLAG	ADDRESS	CONTROL	USER INFROMATION	FCS	FLAG

INFORMATION FRAME

FLAG	ADDRESS	CONTROL	FCS	FLAG

SUPERVISORY FRAME

FLAG	ADDRESS	CONTROL	MANAGEMENT INFORMATION	FCS	FLAG

UNNUMBERED FRAME

- FLAG FIELD:

Is 8bits of a fixed pattern (01111110)

There is one flag at the beginning and one at the end frame

The ending flag of one frame can be used as the beginning flag of the next frame

To guarantee that the flag does not appear any where else in the frame

HDLC uses a process called bit suffing

BITS

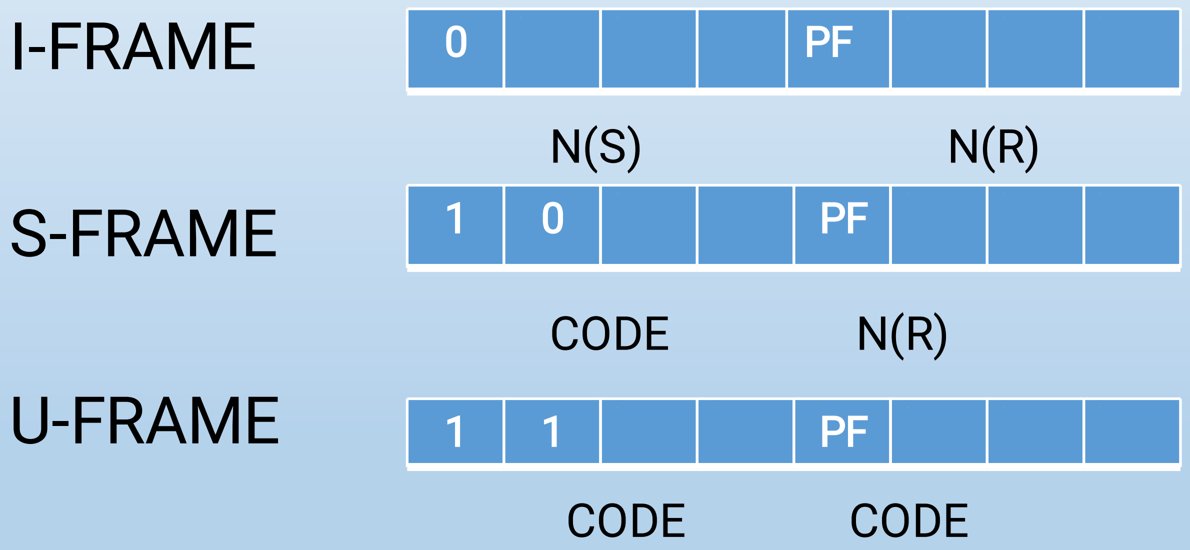
01111110	ADDRESS	CONTROL	DATA	CHECKSUM	01111110

- HDLC Control field:

Data link layer is highly responsible for hop to hop delivery

I- FRAME

FLAG	ADDRESS	CONTROL	INFORMATION	FCS	FLAG

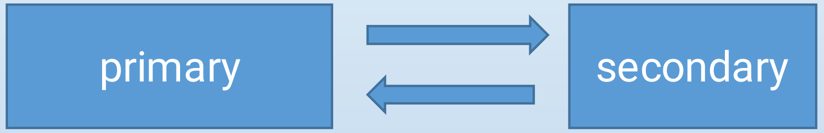


- POLL/FINAL:

P/F=1 poll or final

Poll if frame is send by the primary

Final if frame is sent by the secondary



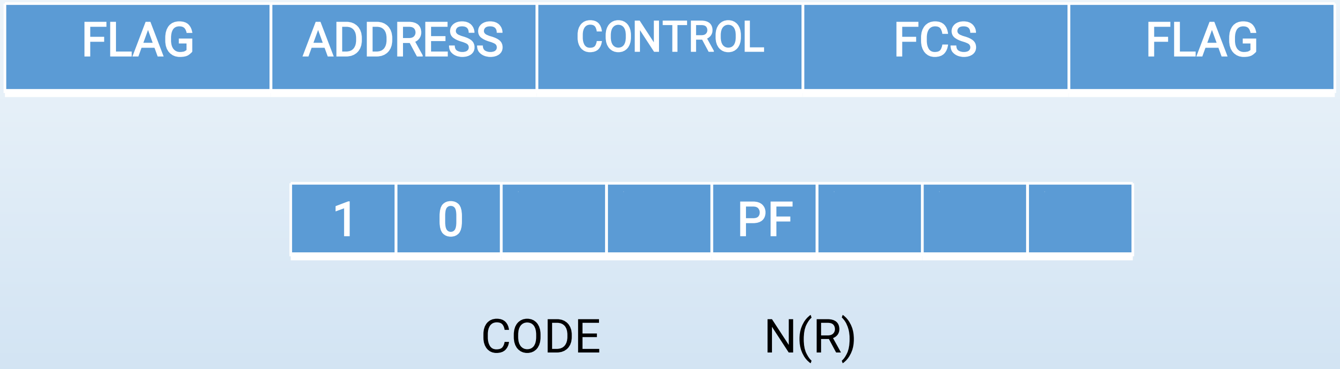
INFORMATION:

User data in an I-frame

Missing in an S-frame

Management information in a U-frame

- S-FRAMES



Code	command
00	RR-receiver ready
01	REJ-reject
10	RNR-receiver not ready
11	SRE-selective reject

- RECEIVER READY (RR):

Positive ack of received I-frame

- RECEIVER NON-READY (RNR):

Is RR frame with additional duties

It ack the receipt of a frame that the receiver is busy

- REJECT (REJ):

This is a NAK frame that can be used in go-back-N

- SELECTIVE REJECT (SREJ):

This is a NAK frame used in selective repeat ARQ

- U-FRAMES:

FLAG	ADDRESS	CONTROL	MANAGEMENT INFORMATION	FCS	FLAG

1	1			PF			

EG: 11 010 – disconnect connection

MAC sublayer:

- Is sublayer in which channel is allocation to multiple user
- MAC sublayer is important in LANS

CHANNEL ALLOCATION PROBLEM:

In which a single channel is divided allotted to multiple user is order to carry a user specific tasks

STATIC CHANNEL ALLOCATION:

It is a traditional approach of allocating a single channel among multiple users by FDM

- If these are 'N' users, the bandwidth PS divided into N equal sized portions each user being assigned one portion.
- Difference between no interface and user

$$T = \frac{1}{\mu C + \lambda}$$

T=Time delay

C=capacity of channel
$\lambda = arrival\ rate\ of\ frames$
$\mu = bits\ per\ frames$

DYNAMIC CHANNEL ALLOCATION

It is based up on possible

- Station model
- Single channel assumption
- Collision assumption
- Time

  Continuous

  Slotted
- Carrier sense
- No carrier sense

✓STATION MODEL:

- Model consists of N independent stations (Eg: computer, telephone or personal communication) each with a program
- Stations are sometime called terminates
- A frame being generated in an interval of length Δ t is

lamda Δ t

- Where lamda is a constant create of new frame
- One frame is generated, station is blocked and does nothing until

the frame has be successfully transmitted

✓SINSLE CHANNEL ASSUMPTION

- Single channel is available for all communication
- All stations can transmit on it and all can receive from it

✓COLLISION ASSUMPTION

- If two frames are transmitted simultaneously, they develop in time, tjis event is called a collision
- Collision frame must be transmitted again

✓TIME:

It can be divided into two types

1. Continuous time
2. Slotted time

- CONTINUOUS TIME:

frame transmission can begin at any instant. There is no matter clock dividing time into discrete intervals

- SLOTTED TIME:

Time is divided into discrete intervals (slots). Frame transmissions always begin at the start of a slot.

✓CARRIER SENSE

If the channel is in use before trying to use it, if the channel is busy no station will be use

✓NO CARRIER SENSE

Stations cannot sense the channel before trying to use it. Time of used to sense loss data.

UNIT IV
TRANSPORT LAYER

Transport layer - Services - Berkeley Sockets -Example – Elements of Transport protocols – Addressing - Connection Establishment - Connection Release - Flow Control and Buffering – Multiplexing – Congestion Control - Bandwidth Allocation - Regulating the Sending Rate –UDP-RPC – TCP - TCP Segment Header - Connection Establishment - Connection Release - TCP Congestion Control

Introduction:

The network layer provides end-to-end packet delivery using data-grams or virtual circuits. The transport layer builds on the network layer to provide data transport from a process on a source machine to a process on a destination machine with a desired level of reliability that is independent of the physical networks currently in use. It provides the abstractions that applications need to use the network.

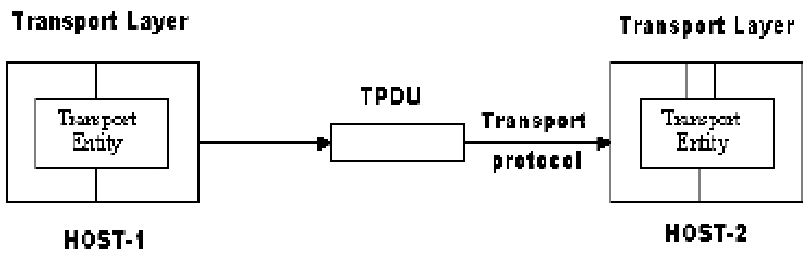
Transport Entity: The hardware and/or software which make use of services provided by the network layer, (within the transport layer) is called transport entity.

Transport Service Provider: Layers 1 to 4 are called Transport Service Provider.

Transport Service User: The upper layers i.e., layers 5 to 7 are called Transport Service User.

Transport Service Primitives: Which allow transport users (application programs) to access the transport service.

TPDU (Transport Protocol Data Unit): Transmissions of message between 2 transport entities are carried out by TPDU. The transport entity carries out the transport service primitives by blocking the caller and sending a packet the service. Encapsulated in the payload of this packet is a transport layer message for the server's transport entity. The task of the transport layer is to provide reliable, cost-effective data transport from the source machine to the destination machine, independent of physical network or networks currently in use.



TRANSPORT SERVICE

1. Services Provided to the Upper Layers

The ultimate goal of the transport layer is to provide efficient, **reliable, and cost-effective data transmission** service to its users, normally processes in the application layer. To achieve this, the transport layer makes use of the **services pro-vided by the network layer.** The software and/or hardware within the transport layer that does the work is called the **transport entity.** The transport entity can be located in the operating system kernel, in a library package bound into network applications, in a separate user process, or even on the network interface card.

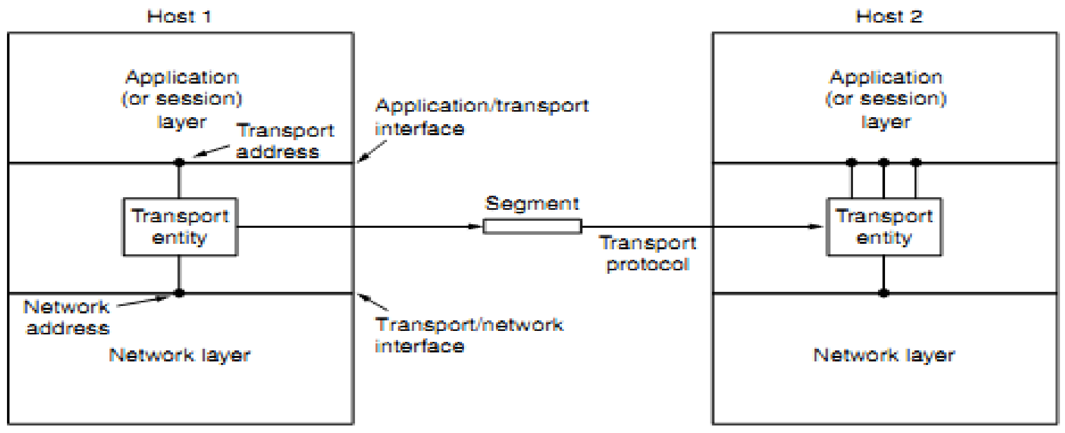


Fig 4.1: The network, Application and transport layer

There are two types of network service

- Connection-oriented
- Connectionless

Similarly, there are also two types of transport service. The connection-oriented transport service is similar to the connection-oriented network service in many ways.

In both cases, connections have three phases:

- Establishment
- Data transfer
- Release.

- Addressing and flow control are also similar in both layers. Furthermore, the connectionless transport service is also very similar to the connectionless network service.
- The bottom four layers can be seen as the transport service provider, whereas the upper layer(s) are the transport service user.

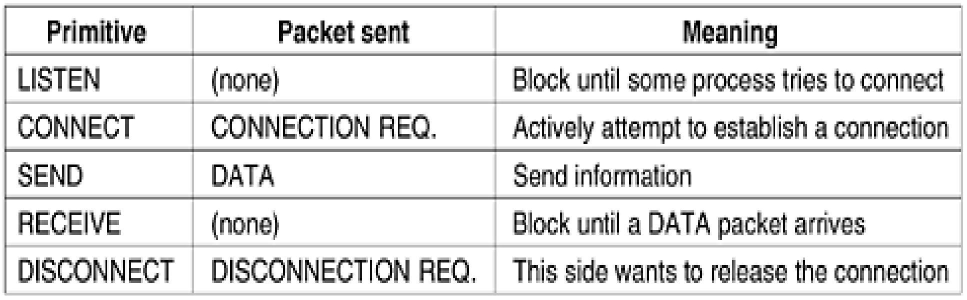
2. Transport Service Primitives

- To allow users to access the transport service, the transport layer must provide some operations to application programs, that is, a transport service interface. Each transport service has its own interface.
- The transport service is similar to the network service, but there are also some important differences.
- The **main difference** is that the network service is intended to model the service offered by real networks. Real networks can lose packets, so the network service is generally **unreliable.**
- The (connection-oriented) transport service, in contrast, is **reliable**

As an example, consider two processes connected by pipes in UNIX. They assume the connection between them is perfect. They do not want to know about acknowledgements, lost packets, congestion, or anything like that. What they want is a 100 percent reliable connection. Process A puts data into one end of the pipe, and process B takes it out of the other.

A **second difference** between the network service and transport service is **whom the services are intended for**. The network service is used only by the transport entities. Consequently, the transport service must be convenient and easy to use.

Table:4.1 - The primitives for a simple transport service.



Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	This side wants to release the connection

Eg: Consider an application with a server and a number of remote clients.

1. The server executes a “LISTEN” primitive by calling a library procedure that makes a System call to block the server until a client turns up.
2. When a client wants to talk to the server, it executes a “CONNECT” primitive, with “CONNECTION REQUEST” TPDU sent to the server.
3. When it arrives, the TE unblocks the server and sends a “CONNECTION ACCEPTED” TPDU back to the client.
4. When it arrives, the client is unblocked and the connection is established. Data can now be exchanged using “SEND” and “RECEIVE” primitives.
5. When a connection is no longer needed, it must be released to free up table space within the 2 transport entries, which is done with “DISCONNECT” primitive by sending “DISCONNECTION REQUEST”

TPDU. This disconnection can b done either by asymmetric variant (connection is released, depending on other one) or by symmetric variant (connection is released, independent of other one).

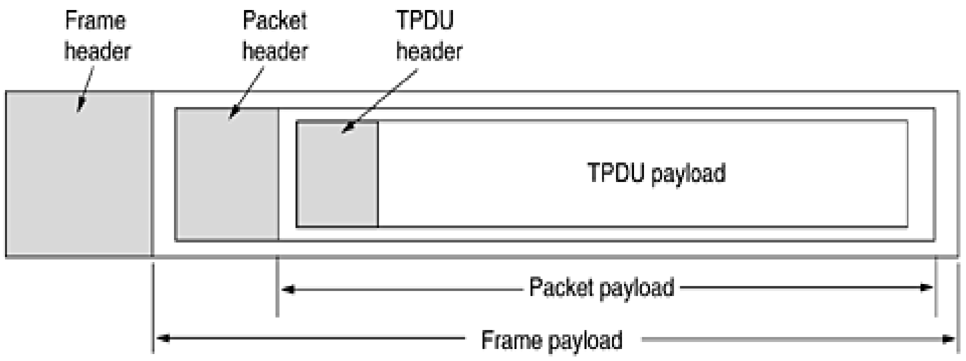


Figure 4.2 - Nesting of TPDUs, packets, and frames

- The term segment for messages sent from transport entity to transport entity.
- TCP, UDP and other Internet protocols use this term. Segments (exchanged by the transport layer) are contained in packets (exchanged by the network layer).
- These packets are contained in frames(exchanged by the data link layer).When a frame arrives, the data link layer processes the frame header and, if the destination address matches for local delivery, passes the contents of the frame payload field up to the network entity.
- The network entity similarly processes the packet header and then passes the contents of the packet payload up to the transport entity. This nesting is illustrated in Fig. 4.2.

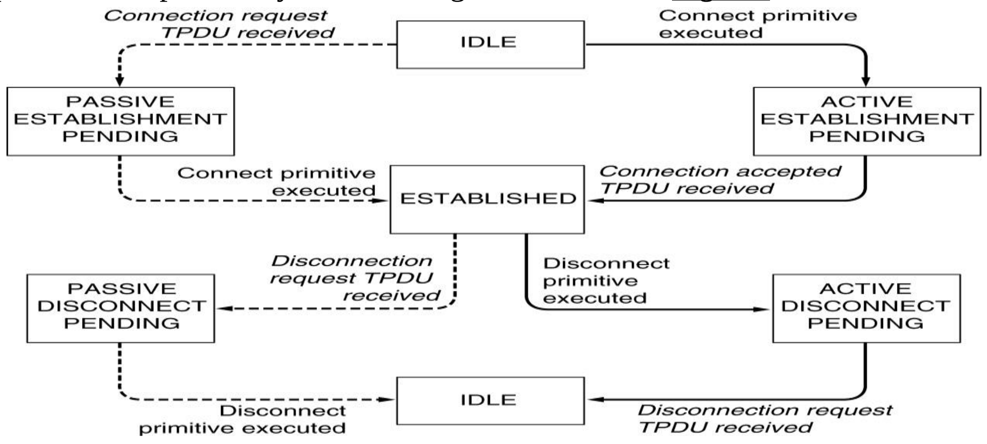

Connection request TPDU received
IDLE
Connect primitive executed
PASSIVE ESTABLISHMENT PENDING
ACTIVE ESTABLISHMENT PENDING
Connect primitive executed
ESTABLISHED
Connection accepted TPDU received
Disconnection request TPDU received
Disconnect primitive executed
PASSIVE DISCONNECT PENDING
ACTIVE DISCONNECT PENDING
IDLE
Disconnect primitive executed
Disconnection request TPDU received

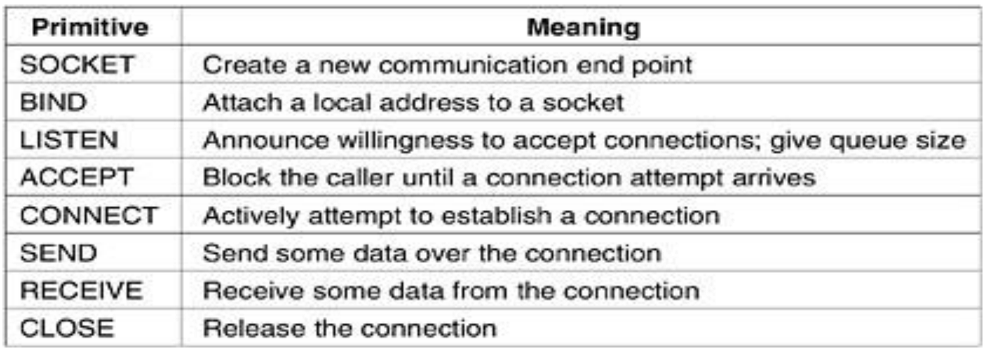
Figure 4.3 - A state diagram for a simple connection management scheme. Transitions labelled in italics are caused by packet arrivals. The solid lines show the client's state sequence. The dashed lines show the server's state sequence.

In fig. 4.3 each transition is triggered by some event, either a primitive executed by the local transport user or an incoming packet. For simplicity, we assume here that each TPDU is separately acknowledged. We also assume that a symmetric disconnection model is used, with the client going first. Please note that this model is quite unsophisticated. We will look at more realistic models later on.

BERKLEY SOCKETS

These primitives are socket primitives used in Berkley UNIX for TCP.

The socket primitives are mainly used for TCP. These sockets were first released as part of the Berkeley UNIX 4.2BSD software distribution in 1983. They quickly became popular. The primitives are now widely used for Internet programming on many operating systems, especially UNIX -based systems, and there is a socket-style API for Windows called ‘‘**winsock**.’’



Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Figure 4.4 - The socket primitives for TCP.

The first four primitives in the list are executed in that order by servers.

The **SOCKET** primitive creates a new endpoint and allocates table space for it within the transport entity. The parameter includes the addressing format to be used, the type of service desired and the protocol. Newly created sockets do not have network addresses.

- The **BIND** primitive is used to connect the newly created sockets to an address. Once a server has bound an address to a socket, remote clients can connect to it.
- The **LISTEN** call, which allocates space to queue incoming calls for the case that several clients try to connect at the same time.
- The server executes an **ACCEPT** primitive to block waiting for an incoming connection.

Some of the client side primitives are. Here, too, a socket must first be created

- The **CONNECT** primitive blocks the caller and actively starts the connection process. When it completes, the client process is unblocked and the connection is established.
- Both sides can now use **SEND** and **RECEIVE** to transmit and receive data over the full-duplex connection.
- Connection release with sockets is symmetric. When both sides have executed a **CLOSE** primitive, the connection is released.

ELEMENTS OF TRANSPORT PROTOCOLS

The transport service is implemented by a transport protocol used between the two transport entities. The transport protocols resemble the data link protocols. Both have to deal with error control, sequencing, and flow control, among other issues. The difference transport protocol and data link protocol depends upon the environment in which they are operated.

These differences are due to major dissimilarities between the environments in which the two protocols operate, as shown in Fig.

At the data link layer, two routers communicate directly via a physical channel, whether wired or wireless, whereas at the transport layer, this physical channel is replaced by the entire network. This difference has many important implications for the protocols.

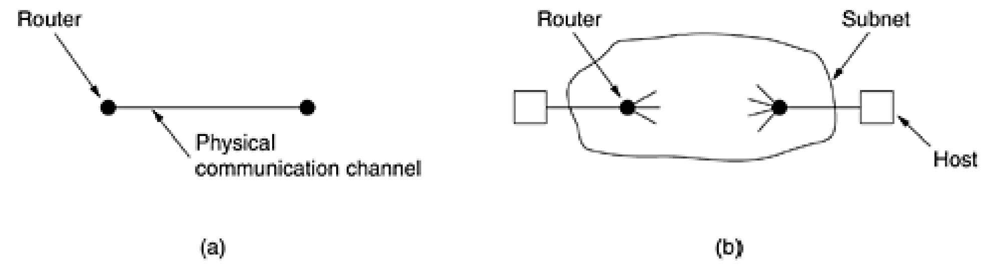


Figure (a) Environment of the data link layer. (b) Environment of the transport layer.

In the data link layer, it is not necessary for a router to specify which router it wants to talk to. In the transport layer, explicit addressing of destinations is required.

In the transport layer, initial connection establishment is more complicated, as we will see. Difference between the data link layer and the transport layer is the potential existence of storage capacity in the subnet

Buffering and flow control are needed in both layers, but the presence of a large and dynamically varying number of connections in the transport layer may require a different approach than we used in the data link layer.

The transport service is implemented by a transport protocol between the 2 transport entities.

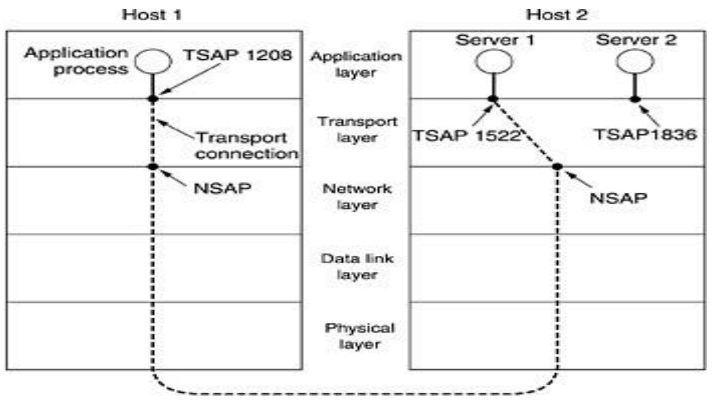


Figure 4.5 illustrates the relationship between the NSAP, TSAP and transport connection. Application processes, both clients and servers, can attach themselves to a TSAP to establish a connection to a remote TSAP.

These connections run through NSAPs on each host, as shown. The purpose of having TSAPs is that in some networks, each computer has a single NSAP, so some way is needed to distinguish multiple transport end points that share that NSAP.

The elements of transport protocols are:

1. ADDRESSING
2. Connection Establishment.
3. Connection Release.
4. Error control and flow control
5. Multiplexing.

1. ADDRESSING

When an application (e.g., a user) process wishes to set up a connection to a remote application process, it must specify which one to connect to. The method normally used is to define transport addresses to which processes can listen for connection requests. In the Internet, these endpoints are called **ports**.

There are two types of access points.

TSAP (Transport Service Access Point) to mean a specific endpoint in the transport layer.

The analogous endpoints in the network layer (i.e., network layer addresses) are not surprisingly called **NSAPs (Network Service Access Points).** IP addresses are examples of NSAPs.

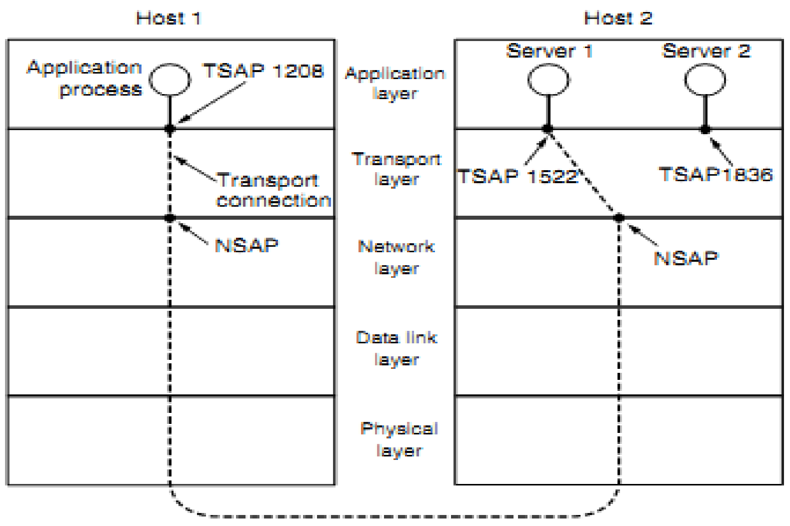


Fig 4.5: TSAP and NSAP network connections

Application processes, both clients and servers, can attach themselves to a local TSAP to establish a connection to a remote TSAP. These connections run through NSAPs on each host. The purpose of having TSAPs is that in some networks, each computer has a single NSAP, so some way is needed to distinguish multiple transport endpoints that share that NSAP.

A possible scenario for a transport connection is as follows:

1. A mail server process attaches itself to TSAP 1522 on host 2 to wait for an incoming call. How a process attaches itself to a TSAP is outside the networking model and depends entirely on the local operating system. A call such as our LISTEN might be used, for example.
2. An application process on host 1 wants to send an email message, so it attaches itself to TSAP 1208 and issues a CONNECT request. The request specifies TSAP 1208 on host 1 as the source and TSAP 1522 on host 2 as the destination. This action ultimately results in a transport connection being established between the application process and the server.
3. The application process sends over the mail message.
4. The mail server responds to say that it will deliver the message.
5. The transport connection is released.

2. CONNECTION ESTABLISHMENT:

With packet lifetimes bounded, it is possible to devise a fool proof way to establish connections safely. Packet lifetime can be bounded to a known maximum using one of the following techniques:

- Restricted subnet design
- Putting a hop counter in each packet
- Time stamping in each packet

Using a 3-way hand shake, a connection can be established. This establishment protocol doesn't require both sides to begin sending with the same sequence number.

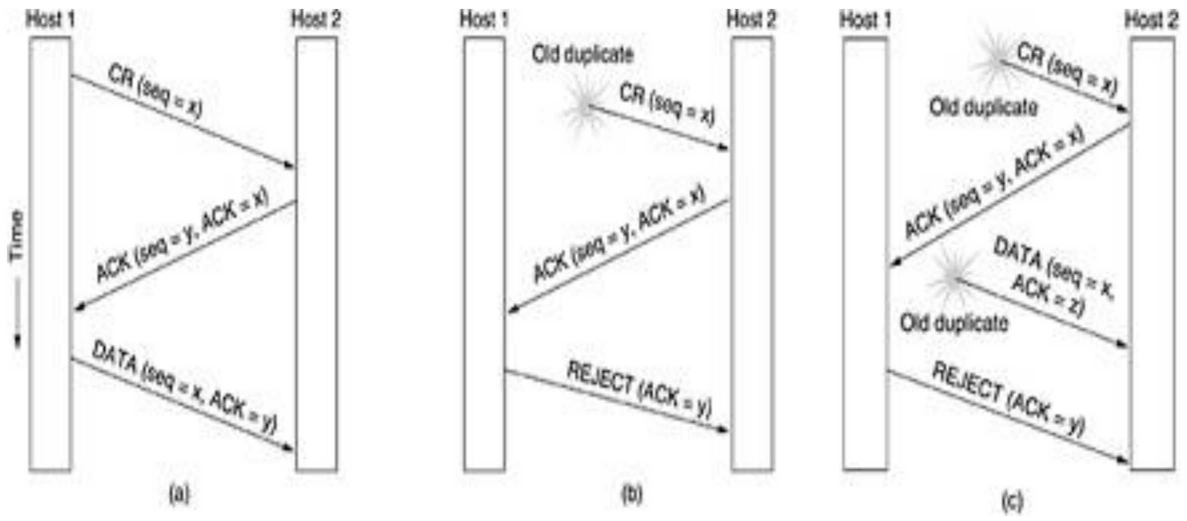


Fig 4.6: Three protocol scenarios for establishing a connection using a three-way handshake. CR denotes CONNEC TION REQUEST (a) Normal operation. (b) Old duplicate CONNECTION REQUEST appearing out of nowhere. (c) Duplicate CONNECTION REQUEST and duplicate ACK .

- The **first technique** includes any method that prevents packets from looping, combined with some way of bounding delay including congestion over the longest possible path. It is difficult, given that internets may range from a single city to international in scope.

- The **second method** consists of having the hop count initialized to some appropriate value and decremented each time the packet is forwarded. The network protocol simply discards any packet whose hop counter becomes zero.
- The **third method** requires each packet to bear the time it was created, with the routers agreeing to discard any packet older than some agreed-upon time.

In **fig (A)** Tomlinson (1975) introduced the **three-way handshake**.

- This establishment protocol involves one peer checking with the other that the connection request is indeed current. Host 1 chooses a sequence number, x , and sends a CONNECTION REQUEST segment containing it to host 2. Host 2replies with an ACK segment acknowledging x and announcing its own initial sequence number, y.

- Finally, host 1 acknowledges host 2's choice of an initial sequence number in the first data segment that it sends

In **fig (B)** the first segment is a delayed duplicate CONNECTION REQUEST from an old connection.

- This segment arrives at host 2 without host 1's knowledge. Host 2 reacts to this segment by sending host1an ACK segment, in effect asking for verification that host 1 was indeed trying to set up a new connection.
- When host 1 rejects host 2's attempt to establish a connection, host 2 realizes that it was tricked by a delayed duplicate and abandons the connection. In this way, a delayed duplicate does no damage.

- The worst case is when both a delayed CONNECTION REQUEST and an ACK are floating around in the subnet.

In **fig (C)** previous example, host 2 gets a delayed CONNECTION REQUEST and replies to it.

- At this point, it is crucial to realize that host 2 has proposed using y as the initial sequence number for host 2 to host 1 traffic, knowing full well that no segments containing sequence number y or acknowledgements to y are still in existence.
- When the second delayed segment arrives at host 2, the fact that z has been acknowledged rather than y tells host 2 that this, too, is an old duplicate.

- The important thing to realize here is that there is no combination of old segments that can cause the protocol to fail and have a connection set up by accident when no one wants it.

3.CONNECTION RELEASE:

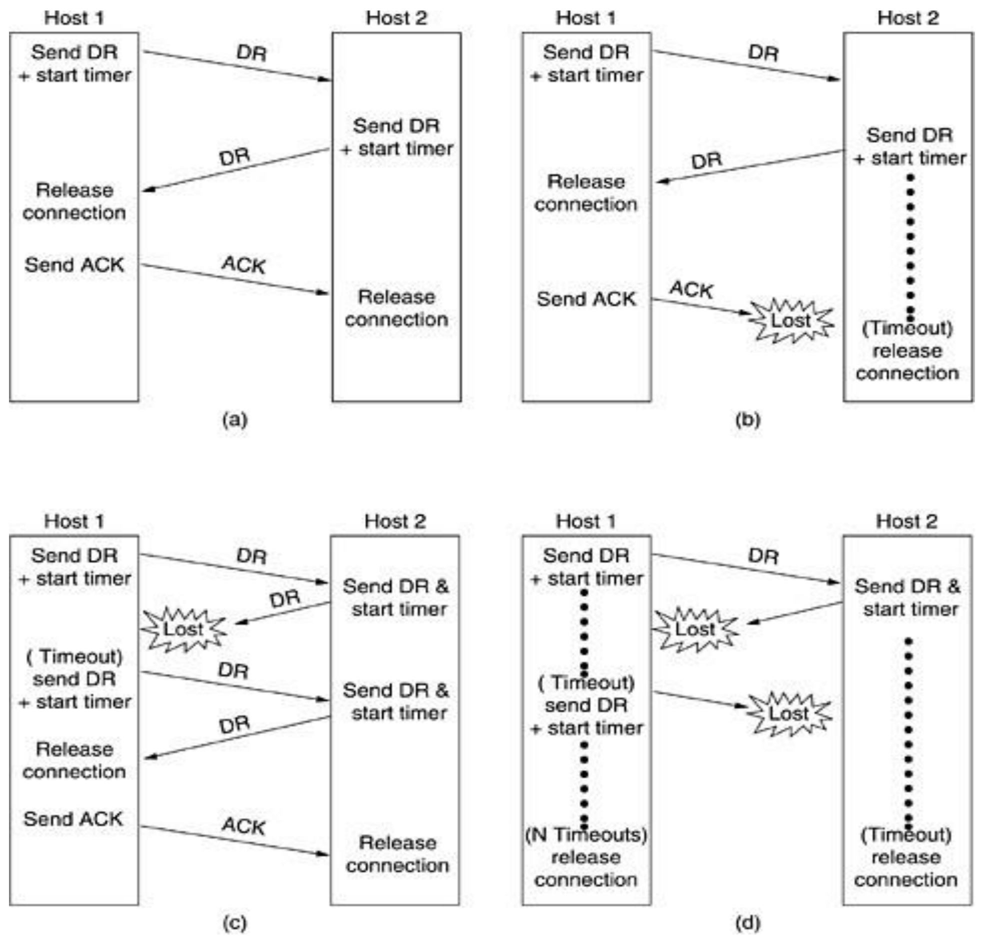
A connection is released using either asymmetric or symmetric variant. But, the improved protocol for releasing a connection is a 3-way handshake protocol.

There are two styles of terminating a connection:

1) Asymmetric release and
2) Symmetric release.

Asymmetric release is the way the telephone system works: when one party hangs up, the connection is broken. **Symmetric release** treats the connection as two separate unidirectional connections and requires each one to be released separately.

Fig-(a)	Fig-(b)	Fig-(c)	Fig-(d)
One of the user sends a DISCONNECTION REQUEST TPDU in order to initiate connection release. When it arrives, the recipient sends back a DR-TPDU, too, and starts a timer. When this DR arrives, the original sender sends back an ACK-TPDU and releases the connection. Finally, when the ACK-TPDU arrives, the receiver also releases the connection.	Initial process is done in the same way as in fig-(a). If the final ACK-TPDU is lost, the situation is saved by the timer. When the timer is expired, the connection is released.	If the second DR is lost, the user initiating the disconnection will not receive the expected response, and will timeout and starts all over again.	Same as in fig-(c) except that all repeated attempts to retransmit the DR is assumed to be failed due to lost TPDUs. After 'N' entries, the sender just gives up and releases the connection.



4.FLOW CONTROL AND BUFFERING:

Flow control is done by having a sliding window on each connection to keep a fast transmitter from over running a slow receiver. Buffering must be done by the sender, if the network service is unreliable. The sender buffers all the TPDUs sent to the receiver. The buffer size varies for different TPDUs.

They are:

a) Chained Fixed-size Buffers
b) Chained Variable-size Buffers
c) One large Circular Buffer per Connection

(a). Chained Fixed-size Buffers:

If most TPDUs are nearly the same size, the buffers are organized as a pool of identical size buffers, with one TPDU per buffer.

(b). Chained Variable-size Buffers:

This is an approach to the buffer-size problem. i.e., if there is wide variation in TPDU size, from a few characters typed at a terminal to thousands of characters from file transfers, some problems may occur:

- If the buffer size is chosen equal to the largest possible TPDU, space will be wasted whenever a short TPDU arrives.
- If the buffer size is chosen less than the maximum TPDU size, multiple buffers will be needed for long TPDUs.

To overcome these problems, we employ variable-size buffers.

(c). One large Circular Buffer per Connection:

A single large circular buffer per connection is dedicated when all connections are heavily loaded.

1. Source Buffering is used for low band width bursty traffic
2. Destination Buffering is used for high band width smooth traffic.
3. Dynamic Buffering is used if the traffic pattern changes randomly.

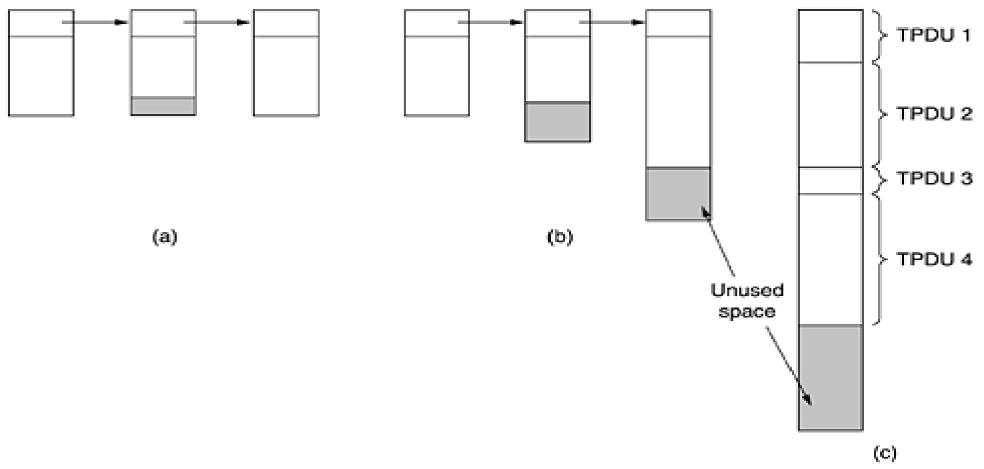


Figure 4.7. (a) Chained fixed-size buffers. (b) Chained variable-sized buffers. (c) One large circular buffer per connection.

<u>5.MULTIPLEXING:</u>

In networks that use virtual circuits within the subnet, each open connection consumes some table space in the routers for the entire duration of the connection. If buffers are dedicated to the virtual circuit in each router as well, a user who left a terminal logged into a remote machine, there is need for multiplexing. There are 2 kinds of multiplexing:

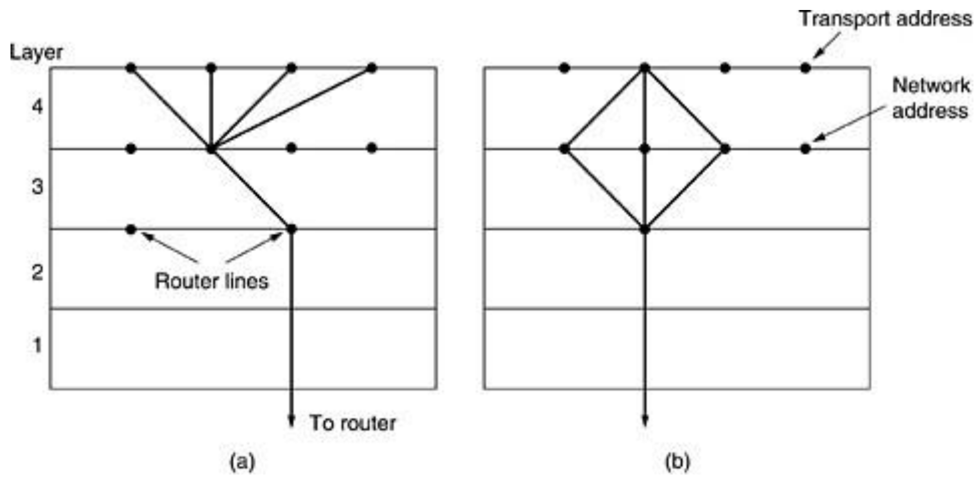


Figure 4.8. (a) Upward multiplexing. (b) Downward multiplexing

(a). <u>UP-WARD MULTIPLEXING:</u>

In the below figure, all the 4 distinct transport connections use the same network connection to the remote host. When connect time forms the major component of the carrier's bill, it is up to the transport layer to group port connections according to their destination and map each group onto the minimum number of port connections.

(b). <u>DOWN-WARD MULTIPLEXING:</u>

- If too many transport connections are mapped onto the one network connection, the performance will be poor.
- If too few transport connections are mapped onto one network connection, the service will be expensive.

The possible solution is to have the transport layer open multiple connections and distribute the traffic among them on round-robin basis, as indicated in the below figure:
With 'k' network connections open, the effective band width is increased by a factor of 'k'.

TRANSPORT PROTOCOLS - UDP

The Internet has two main protocols in the transport layer, **a connectionless protocol** and a **connection-oriented** one. The protocols complement each other. The connectionless protocol is **UDP.** It does almost nothing beyond sending packets between applications, letting applications build their own protocols on top as needed.

The connection-oriented protocol is **TCP.** It does almost everything. It makes connections and adds reliability with retransmissions, along with flow control and congestion control, all on behalf of the applications that use it. Since UDP is a transport layer protocol that typically runs in the operating system and protocols that use UDP typically run in user s pace, these uses might be considered applications.

INTROUCTION TO UDP

- The Internet protocol suite supports a connectionless transport protocol called UDP (User Datagram Protocol). UDP provides a way for applications to send encapsulated IP datagrams without having to establish a connection.
- UDP transmits segments consisting of an 8-byte header followed by the pay-load. The two ports serve to identify the end-points within the source and destination machines.
- When a UDP packet arrives, its payload is handed to the process attached to the destination port. This attachment occurs when the BIND primitive. Without the port fields, the transport layer would not know what to do with each incoming packet. With them, it delivers the embedded segment to the correct application.

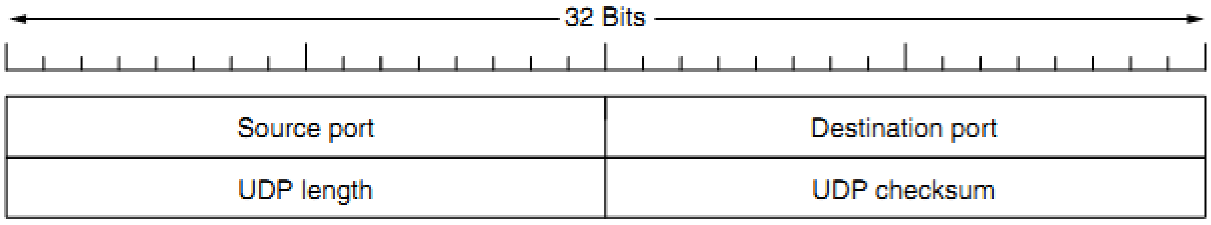


Fig 4.9: The UDP header

Source port, destination port: Identifies the end points within the source and destination machines.
UDP length: Includes 8-byte header and the data
UDP checksum: Includes the UDP header, the UDP data padded out to an even number of bytes if need be. It is an optional field

REMOTE PROCEDURE CALL

- In a certain sense, sending a message to a remote host and getting a reply back is like making a function call in a programming language. This is to arrange request-reply interactions on networks to be cast in the form of procedure calls.
- For example, just imagine a procedure named *get IP address* (*host name*) that works by sending a UDP packet to a DNS server and waiting or the reply, timing out and trying again if one is not forthcoming quickly enough. In this way, all the details of networking can be hidden from the programmer.

- RPC is used to call remote programs using the procedural call. When a process on machine 1 calls a procedure on machine 2, the calling process on 1 is suspended and execution of the called procedure takes place on 2.

- Information can be transported from the caller to the callee in the parameters and can come back in the procedure result. No message passing is visible to the application programmer. This technique is known as **RPC** (**Remote Procedure Call**) and has become the basis for many networking applications.

Traditionally, the calling procedure is known as the **client** and the called procedure is known as the **server.**

- In the simplest form, to call a remote procedure, the client program must be bound with a small library procedure, called the **client stub**,that represents the server procedure in the client's address space. Similarly, the server is bound with a procedure called the **server stub**. These procedures hide the fact that the procedure call from the client to the server is not local.

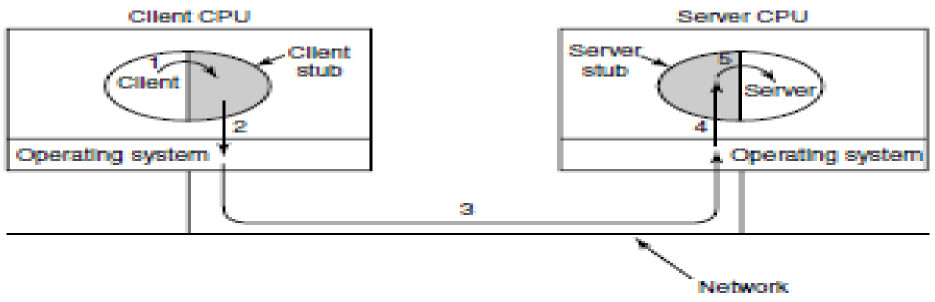


Fig 4.10: Steps in making a RPC

Step 1 is the client calling the client stub. This call is a local procedure call, with the parameters pushed onto the stack in the normal way.
Step 2 is the client stub packing the parameters into a message and making a system call to send the message. Packing the parameters is called **marshaling**.
Step 3 is the operating system sending the message from the client machine to the server machine.
Step 4 is the operating system passing the incoming packet to the server stub.
Step 5 is the server stub calling the server procedure with the **unmarshaled** parameters. The reply traces the same path in the other direction.

The key item to note here is that the client procedure, written by the user, just makes a normal (i.e., local) procedure call to the client stub, which has the same name as the server procedure. Since the client procedure and client stub are in the same address space, the parameters are passed in the usual way.

Similarly, the server procedure is called by a procedure in its address space with the parameters it expects. To the server procedure, nothing is unusual. In this way, instead of I/O being done on sockets, network communication is done by faking a normal procedure call. With RPC, passing pointers is impossible because the client and server are in different address spaces.

TCP (TRANSMISSION CONTROL PROTOCOL)

It was specifically designed to provide a reliable end-to end byte stream over an unreliable network. It was designed to adapt dynamically to properties of the inter network and to be robust in the face of many kinds of failures.

Each machine supporting TCP has a TCP transport entity, which accepts user data streams from local processes, breaks them up into pieces not exceeding 64kbytes and sends each piece as a separate IP datagram. When these datagrams arrive at a machine, they are given to TCP entity, which reconstructs the original byte streams. It is up to TCP to time out and retransmits them as needed, also to reassemble datagrams into messages in proper sequence.

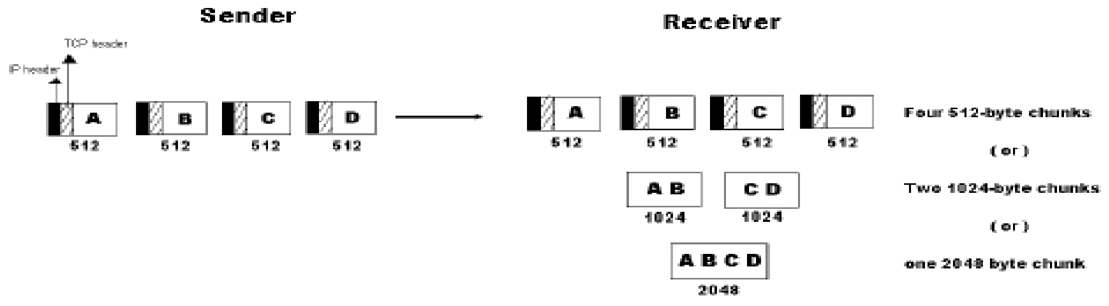
The different issues to be considered are:

1. The TCP Service Model
2. The TCP Protocol
3. The TCP Segment Header
4. The Connection Management
5. TCP Transmission Policy
6. TCP Congestion Control
7. TCP Timer Management.

The TCP Service Model

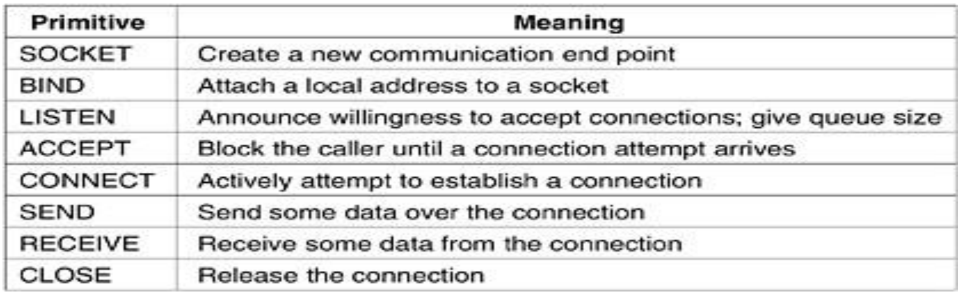
- TCP service is obtained by having both the sender and receiver create end points called **SOCKETS**
- Each socket has a socket number(address)consisting of the IP address of the host, called a **“PORT”** (= TSAP)
- To obtain TCP service a connection must be explicitly established between a socket on the sending machine and a socket on the receiving machine
- All TCP connections are full duplex and point to point i.e., multicasting or broadcasting is not supported.
- A TCP connection is a byte stream, not a message stream i.e., the data is delivered as chunks

E.g.: 4 * 512 bytes of data is to be transmitted.



Sockets:

A socket may be used for multiple connections at the same time. In other words, 2 or more connections may terminate at same socket. Connections are identified by socket identifiers at same socket. Connections are identified by socket identifiers at both ends. Some of the sockets are listed below:



Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Ports: Port numbers below 256 are called Well- known ports and are reserved for standard services.
***Eg*:**

PORT-21	To establish a connection to a host to transfer a file using FTP
PORT-23	To establish a remote login session using TELNET

<u>The TCP Protocol</u>

- A key feature of TCP, and one which dominates the protocol design, is that every byte on a TCP connection has its own 32-bit sequence number.
- When the Internet began, the lines between routers were mostly 56-kbps leased lines, so a host blasting away at full speed took over 1 week to cycle through the sequence numbers.
- The basic protocol used by TCP entities is the **<u>sliding window protocol</u>**.
- When a sender transmits a segment, it also starts a timer.
- When the segment arrives at the destination, the receiving TCP entity sends back a segment (with data if any exist, otherwise without data) bearing an acknowledgement number equal to the next sequence number it expects to receive.
- If the sender's timer goes off before the acknowledgement is received, the sender transmits the segment again.

<u>The TCP Segment Header</u>

Every segment begins with a fixed-format, 20-byte header. The fixed header may be followed by header options. After the options, if any, up to 65,535 - 20 - 20 = 65,495 data bytes may follow, where the first 20 refer to the IP header and the second to the TCP header. Segments without any data are legal and are commonly used for acknowledgements and control messages.

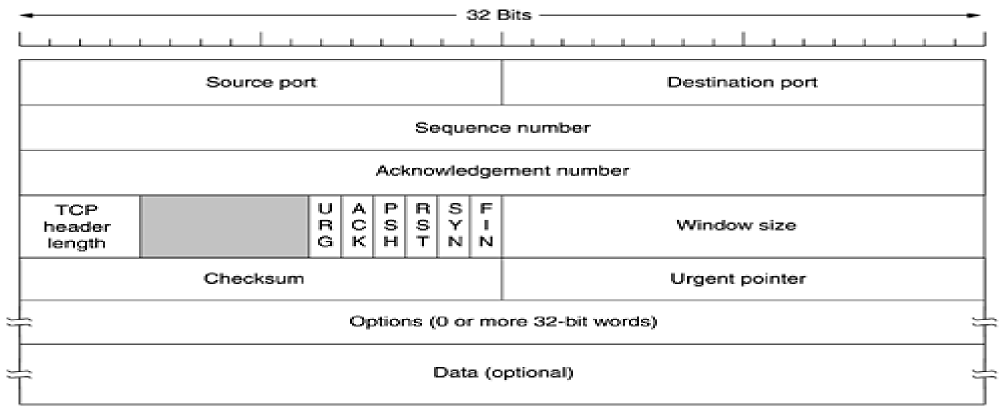


Fig 4.11: The TCP Header

Source Port, Destination Port : Identify local end points of the connections

Sequence number: Specifies the sequence number of the segment

Acknowledgement Number: Specifies the next byte expected.

TCP header length: Tells how many 32-bit words are contained in TCP header

URG: It is set to 1 if URGENT pointer is in use, which indicates start of urgent data.

ACK: It is set to 1 to indicate that the acknowledgement number is valid.

PSH: Indicates pushed data

RST: It is used to reset a connection that has become confused due to reject an invalid segment or refuse an attempt to open a connection.

FIN: Used to release a connection.

SYN: Used to establish connections.

<u>TCP Connection Establishment</u>

To establish a connection, one side, say, the server, passively waits for an incoming connection by executing the LISTEN and ACCEPT primitives, either specifying a specific source or nobody in particular.

The other side, say, the client, executes a CONNECT primitive, specifying the IP address and port to which it wants to connect, the maximum TCP segment size it is willing to accept, and optionally some user data (e.g., a password).

The CONNECT primitive sends a TCP segment with the *SYN* bit on and *ACK* bit off and waits for a response.

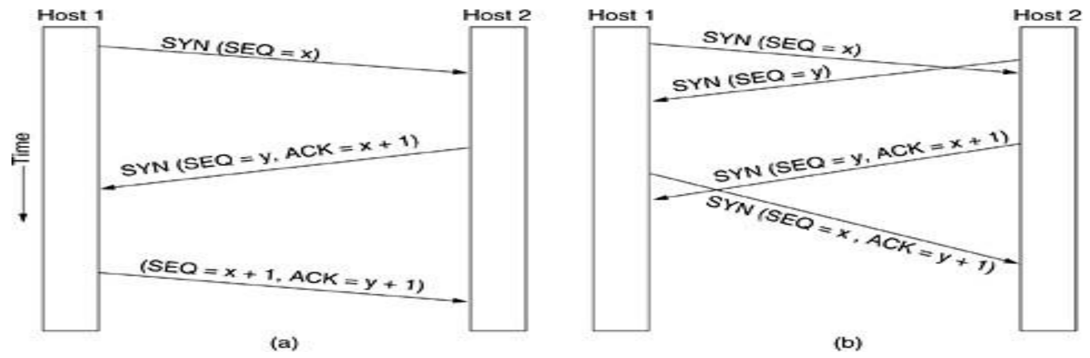

Host 1
Host 2
SYN (SEQ = x)
SYN (SEQ = y, ACK = x + 1)
(SEQ = x + 1, ACK = y + 1)
Time
(a)
Host 1
Host 2
SYN (SEQ = x)
SYN (SEQ = y)
SYN (SEQ = y, ACK = x + 1)
SYN (SEQ = x, ACK = y + 1)
(b)

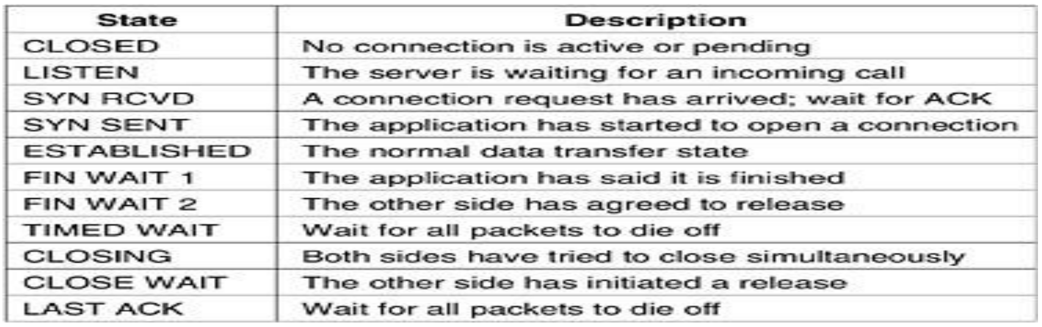
Fig 4.12: a) TCP Connection establishment in the normal case b) Call Collision

TCP Connection Release

- Although TCP connections are full duplex, to understand how connections are released it is best to think of them as a pair of simplex connections.
- Each simplex connection is released independently of its sibling. To release a connection, either party can send a TCP segment with the *FIN* bit set, which means that it has no more data to transmit.
- When the *FIN* is acknowledged, that direction is shut down for new data. Data may continue to flow indefinitely in the other direction, however.
- When both directions have been shut down, the connection is released.
- Normally, four TCP segments are needed to release a connection, one *FIN* and one *ACK* for each direction. However, it is possible for the first *ACK* and the second *FIN* to be contained in the same segment, reducing the total count to three.

TCP Connection Management Modeling

The steps required establishing and release connections can be represented in a finite state machine with the 11 states listed in Fig. 4.13. In each state, certain events are legal. When a legal event happens, some action may be taken. If some other event happens, an error is reported.



State	Description
CLOSED	No connection is active or pending
LISTEN	The server is waiting for an incoming call
SYN RCVD	A connection request has arrived; wait for ACK
SYN SENT	The application has started to open a connection
ESTABLISHED	The normal data transfer state
FIN WAIT 1	The application has said it is finished
FIN WAIT 2	The other side has agreed to release
TIMED WAIT	Wait for all packets to die off
CLOSING	Both sides have tried to close simultaneously
CLOSE WAIT	The other side has initiated a release
LAST ACK	Wait for all packets to die off

Figure 4.13. The states used in the TCP connection management finite state machine.

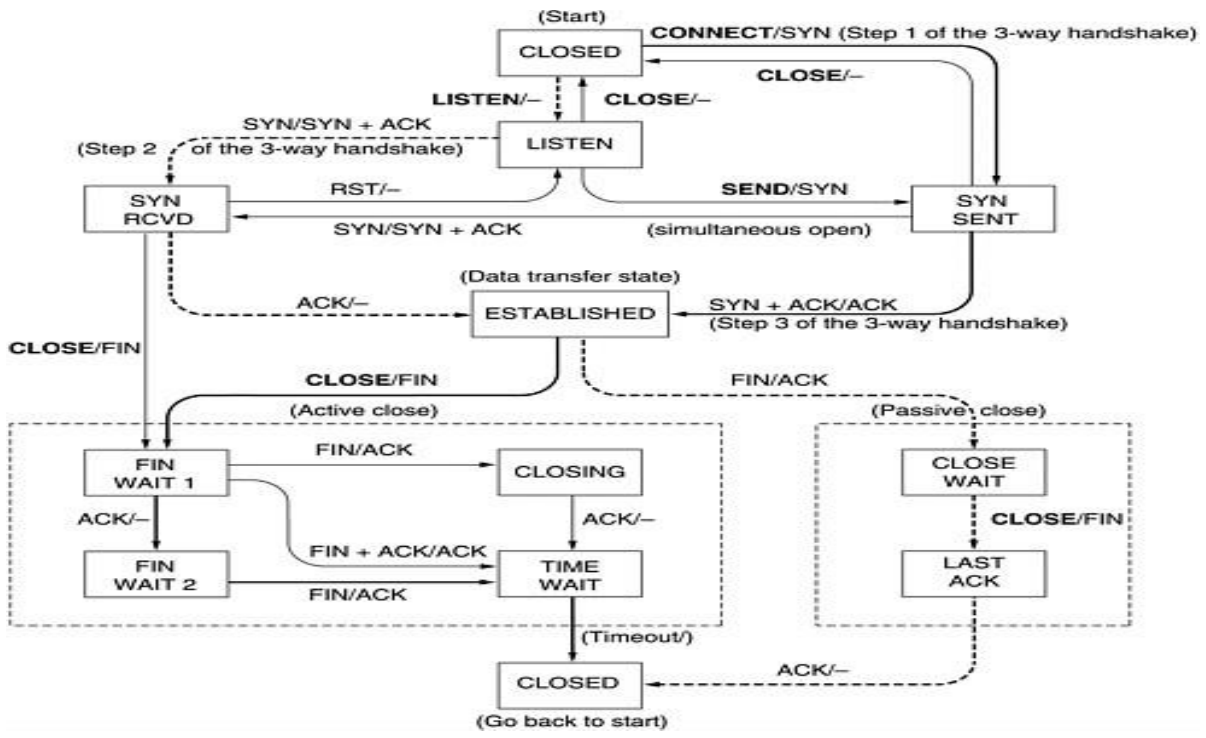


Figure 4.14 - TCP connection management finite state machine.

TCP Connection management from server's point of view:

1. The server does a **LISTEN** and settles down to see who turns up.
2. When a **SYN** comes in, the server acknowledges it and goes to the **SYNRCVD** state
3. When the servers **SYN** is itself acknowledged the 3-way handshake is complete and server goes to the **ESTABLISHED** state. Data transfer can now occur.
4. When the client has had enough, it does a close, which causes a **FIN** to arrive at the server [dashed box marked passive close].
5. The server is then signaled.
6. When it too, does a **CLOSE**, a **FIN** is sent to the client.
7. When the client's acknowledgement shows up, the server releases the connection and deletes the connection record.

TCP CONGESTION CONTROL:

TCP does to try to prevent the congestion from occurring in the first place in the following way:

When a connection is established, a suitable window size is chosen and the receiver specifies a window based on its buffer size. If the sender sticks to this window size, problems will not occur due to buffer overflow at the receiving end. But they may still occur due to internal congestion within the network. Let's see this problem occurs.

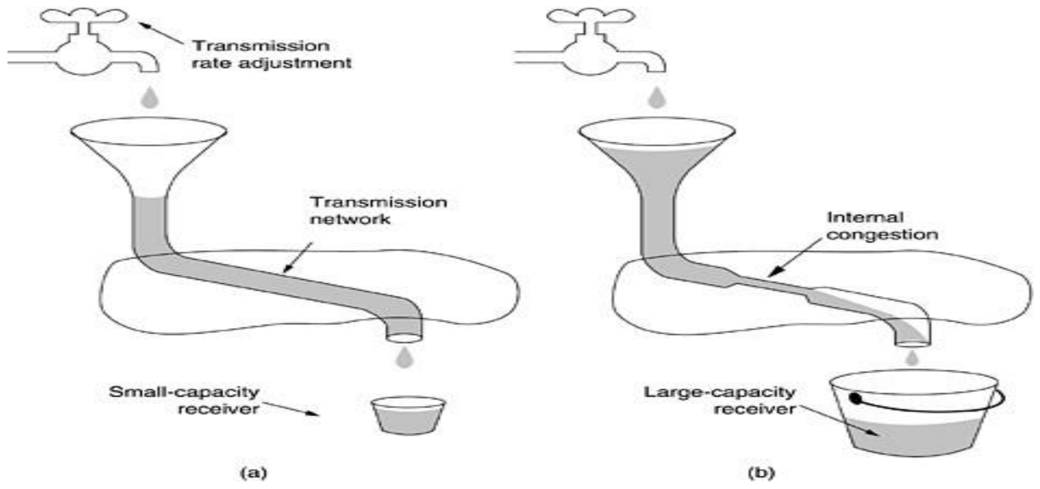


Figure 4.16. (a) A fast network feeding a low-capacity receiver. (b) A slow network feeding a high-capacity receiver.

In fig (a): We see a thick pipe leading to a small- capacity receiver. As long as the sender does not send more water than the bucket can contain, no water will be lost.

In fig (b): The limiting factor is not the bucket capacity, but the internal carrying capacity of the n/w. if too much water comes in too fast, it will backup and some will be lost.

- When a connection is established, the sender initializes the congestion window to the size of the max segment in use our connection.
- It then sends one max segment .if this max segment is acknowledged before the timer goes off, it adds one segment s worth of bytes to the congestion window to make it two maximum size segments and sends 2 segments.
- As each of these segments is acknowledged, the congestion window is increased by one max segment size.
- When the congestion window is ‘n’ segments, if all ‘n’ are acknowledged on time, the congestion window is increased by the byte count corresponding to ‘n’ segments.
- The congestion window keeps growing exponentially until either a time out occurs or the receiver’s window is reached.
- The internet congestion control algorithm uses a third parameter, the **“threshold”** in addition to receiver and congestion windows.

Different congestion control algorithms used by TCP are:

- RTT variance Estimation.
- Exponential RTO back-off
- Karn’s Algorithm

} Re-transmission Timer Management

- Slow Start
- Dynamic window sizing on congestion
- Fast Retransmit
- Fast Recovery

} Window Management

UNIT – V
APPLICATION LAYER

The Application Layer- DNS - Name Space - Resource Records - Name Servers- E-Mail - Architecture And Services – The User Agent –Message Format –Message Transfer –Final Delivery – WWW –Architecture – Static Web Pages – Dynamic Web Pages And Web Applications –HTTP –Network Security –Introduction To Cryptography –Substitution Ciphers-Transposition Ciphers-Public Key Algorithms – RSA – Authentication Protocols –Authentication Using Kerberos.

DOMAIN NAME SYSTEM

This is primarily used for mapping host and e-mail destinations to IP addresses but can also be used other purposes. DNS is defined in RFCs 1034 and 1035.

Working:-

- To map a name onto an IP address, an application program calls a library procedure called Resolver, passing it the name as a parameter.
- The resolver sends a UDP packet to a local DNS server, which then looks up the name and returns the IP address to the resolver, which then returns it to the caller.
- Armed with the IP address, the program can then establish a TCP connection with the destination, or send it UDP packets.

1. **The DNS name space.**
2. **Resource Records.**
3. **Name Servers.**

1. THE DNS NAME SPACE:

The Internet is divided into several hundred top level domains, where each domain covers many hosts. Each domain is partitioned into sub domains, and these are further partitioned as so on. All these domains can be represented by a tree, in which the leaves represent domains that have no sub domains. A leaf domain may contain a single host, or it may represent a company and contains thousands of hosts. Each domain is named by the path upward from it to the root. The components are separated by periods (pronounced “dot”)
Eg: Sun Microsystems Engg. Department = eng.sun.com.

The top domain comes in 2 flavours:-

- **Generic:** com(commercial), edu(educational instructions), mil(the U.S armed forces, government), int (certain international organizations), net(network providers), org (non profit organizations).

- **Country:** include 1 entry for every country. Domain names can be either absolute (ends with a period e.g. eng.sum.com) or relative (doesn't end with a period). Domain names are case sensitive and the component names can be up to 63 characters long and full path names must not exceed 255 characters.

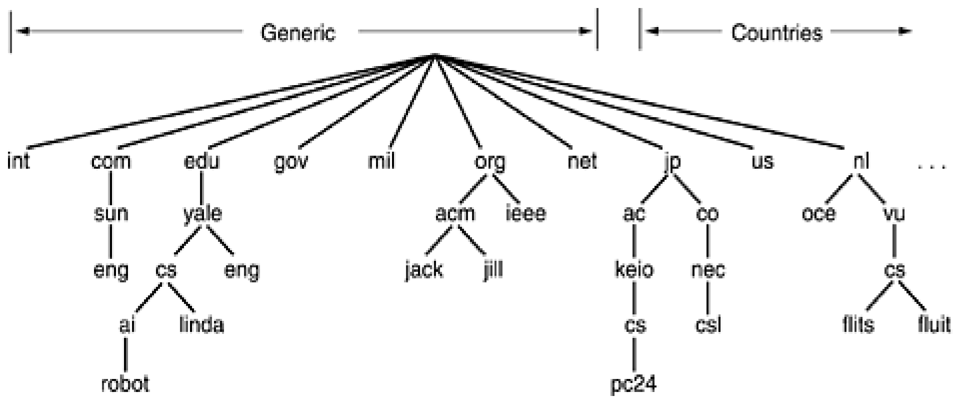


Figure 5-1. A portion of the Internet domain name space.

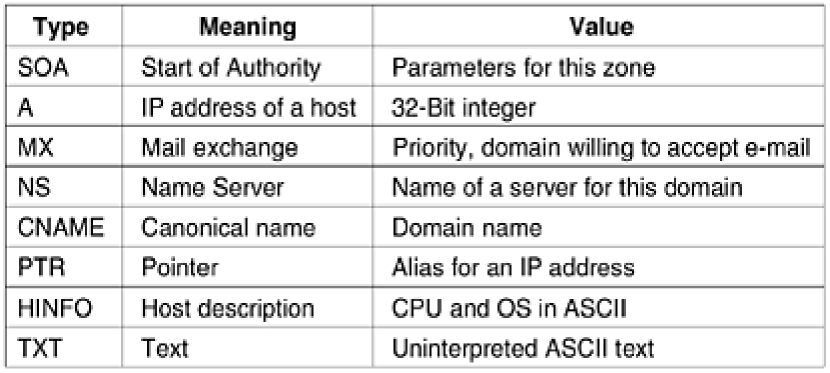
Insertions of a domain into the tree can be done in 2 days:-
• Under a generic domain (Eg: cs.yale.edu)
• Under the domain of their country (E.g: cs.yale.ct.us)

2. <u>RESOURCE RECORDS:</u>

Every domain can have a sent of resource records associated with it. For a single host, the most common resource record is just its IP address. When a resolver gives a domain name to DNS, it gets both the resource records associated with that name i.e., the real function of DNS is to map domain names into resource records.
A resource record is a 5-tuple and its format is as follows:

Domain	Name	Time to live	Type	Class	Value

Domain _name : Tells the domain to which this record applies.
Time- to- live : Gives an identification of how stable the record is (High Stable = 86400 i.e. no. of seconds /day) (High Volatile = 1 min)
Type: Tells what kind of record this is.
Class: It is IN for the internet information and codes for non internet information
Value: This field can be a number a domain name or an ASCII string



Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

3. **<u>NAME SERVERS:</u>**

It contains the entire database and responds to all queries about it. DNS name space is divided up into non-overlapping zones, in which each zone contains some part of the tree and also contains name servers holding the authoritative information about that zone.

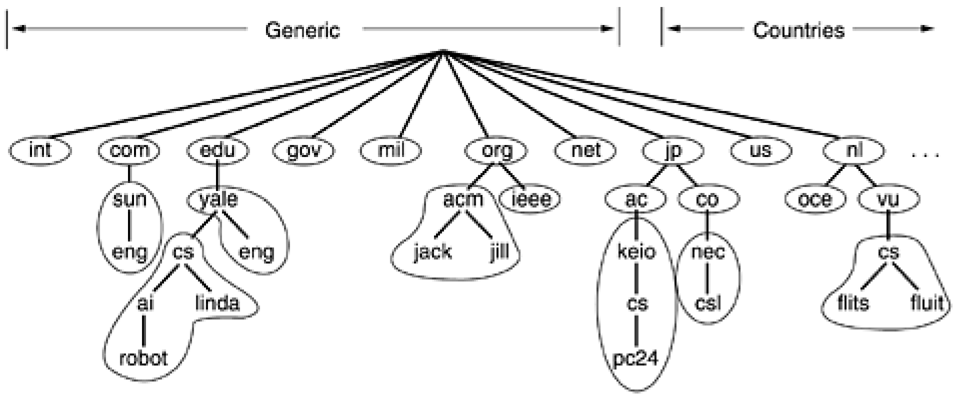


Figure 5-2. Part of the DNS name space showing the division into zones.

When a resolver has a query about a domain name, it passes the query to one of the local name servers:

1. If the domain being sought falls under the jurisdiction of name server, it returns the authoritative resource records (that comes from the authority that manages the record,and is always correct).
2. If the domain is remote and no information about the requested domain is available locally the name server sends a query message to the top level name server for the domain requested.

E.g.: A resolver of flits.cs.vle.nl wants to know the IP address of the host Linda.cs.yale.edu

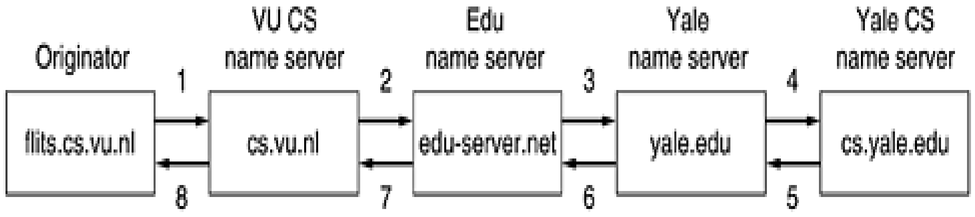


Figure 5-3. How a resolver looks up a remote name in eight steps.

Step 1: Resolver sends a query containing domain name sought the type and the class to local name server, cs.vu.nl.

Step 2: Suppose local name server knows nothing about it, it asks few others nearby name servers. If none of them know, it sends a UDP packet to the server for edu-server.net.

Step 3: This server knows nothing about Linda.cs.yale.edu or cs.yale.edu and so it forwards the request to the name server for yale.edu.

Step 4: This one forwards the request to cs.yale.edu which must have authoritative resource records.

Step 5 to 8: The resource record requested works its way back in steps 5-8 This query method is known as **Recursive Query**

3. When a query cannot be satisfied locally, the query fails but the name of the next server along the line to try is returned.

ELECTRONIC MAIL

1. ARCHITECTURE AND SERVICES:

E-mail systems consist of two subsystems. They are:-

(1). User Agents, which allow people to read and send e-mail
(2). Message Transfer Agents, which move messages from source to destination

E-mail systems support 5 basic functions:-

a. Composition
b. Transfer
c. Reporting
d. Displaying
e. Disposition

(a). Composition: It refers to the process of creating messages and answers. Any text editor is used for body of the message. While the system itself can provide assistance with addressing and numerous header fields attached to each message.

(b). Reporting: It has to do with telling the originator what happened to the message that is, whether it was delivered, rejected (or) lost.

(c). Transfer: It refers to moving messages from originator to the recipient.

(d). Displaying: Incoming messages are to be displayed so that people can read their email.

(e). Disposition: It concerns what the recipient dose with the message after receiving it. Possibilities include throwing it away before reading (or) after reading, saving it and so on.

Most systems allow users to create **mailboxes** to store incoming e-mail. Commands are needed to create and destroy mailboxes, inspect the contents of mailboxes, insert and delete messages from mailboxes, and so on.

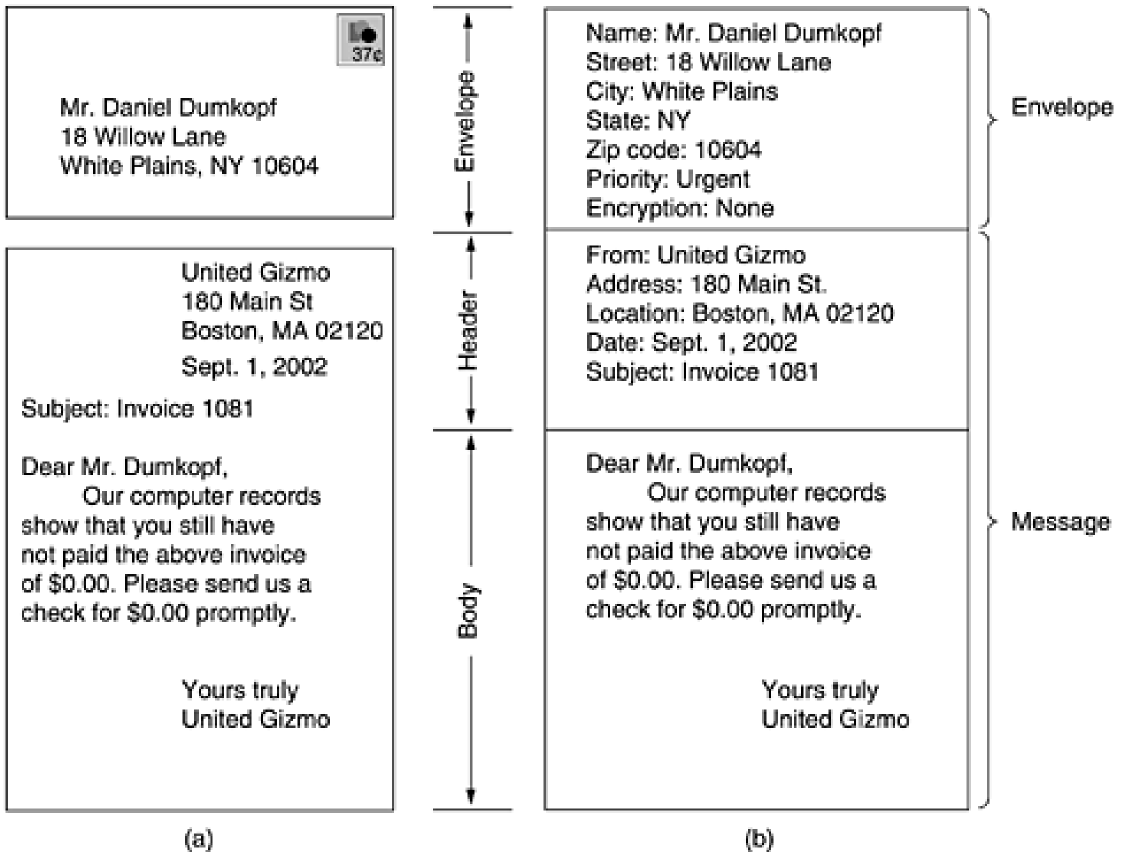


Figure 5-4: Envelopes and messages. (a) Paper mail. (b) Electronic mail.

(1) <u>THE USER AGENT</u>

A user agent is normally a program (sometimes called a mail reader) that accepts a variety of commands for composing, receiving, and replying to messages, as well as for manipulating mailboxes.

<u>SENDING E-MAIL</u>

To send an e-mail message, a user must provide the message, the destination address, and possibly some other parameters. The message can be produced with a free-standing text editor, a word processing program, or possibly with a specialized text editor built into the user agent. The destination address must be in a format that the user agent can deal with. Many user agents expect addresses of the form *user@dns-address*.

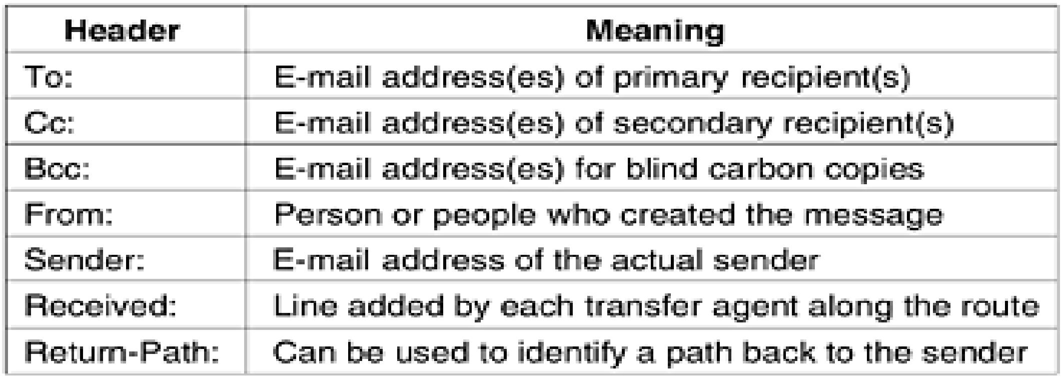
<u>READING E-MAIL</u>

When a user agent is started up, it looks at the user's mailbox for incoming e-mail before displaying anything on the screen. Then it may announce the number of messages in the mailbox or display a one-line summary of each one and wait for a command.

(2) MESSAGE FORMATS

RFC 822

Messages consist of a primitive envelope (described in RFC 821), some number of header fields, a blank line, and then the message body. Each header field (logically) consists of a single line of ASCII text containing the field name, a colon, and, for most fields, a value.



Header	Meaning
To:	E-mail address(es) of primary recipient(s)
Cc:	E-mail address(es) of secondary recipient(s)
Bcc:	E-mail address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	E-mail address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

Figure 5-5: RFC 822 header fields related to message transport

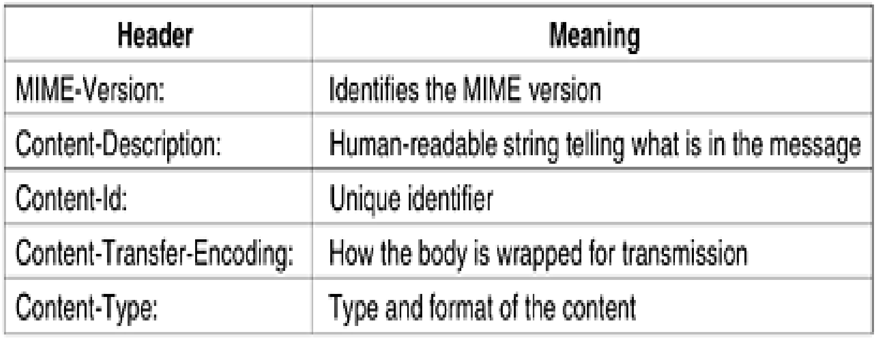
MIME — The Multipurpose Internet Mail Extensions

RFC 822 specified the headers but left the content entirely up to the users. Nowadays, on the worldwide Internet, this approach is no longer adequate. The problems include sending and receiving

1. Messages in languages with accents (e.g., French and German).
2. Messages in non-Latin alphabets (e.g., Hebrew and Russian).
3. Messages in languages without alphabets (e.g., Chinese and Japanese).
4. Messages not containing text at all (e.g., audio or images).

A solution was proposed in RFC 1341 called **MIME** (**Multipurpose Internet Mail Extensions**)

The basic idea of MIME is to continue to use the RFC 822 format, but to add structure to the message body and define encoding rules for non-ASCII messages. By not deviating from RFC 822, MIME messages can be sent using the existing mail programs and protocols. All that has to be changed are the sending and receiving programs, which users can do for themselves.



Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

Figure 5-6: RFC 822 headers added by MIME

MESSAGE TRANSFER

The message transfer system is concerned with relaying messages from the originator to the recipient. The simplest way to do this is to establish a transport connection from the source machine to the destination machine and then just transfer the message.

SMTP—THE SIMPLE MAIL TRANSFER PROTOCOL

SMTP is a simple ASCII protocol. After establishing the TCP connection to port 25, the sending machine, operating as the client, waits for the receiving machine, operating as the server, to talk first. The server starts by sending a line of text giving its identity and telling whether it is prepared to receive mail. If it is not, the client releases the connection and tries again later.

Even though the SMTP protocol is completely well defined, a **few problems** can still arise.

One problem relates to message length. Some older implementations cannot handle messages exceeding 64 KB.

Another problem relates to timeouts. If the client and server have different timeouts, one of them may give up while the other is still busy, unexpectedly terminating the connection.

Finally, in rare situations, infinite mailstorms can be triggered.

For example, if host 1 holds mailing list *A* and host 2 holds mailing list *B* and each list contains an entry for the other one, then a message sent to either list could generate a never-ending amount of e-mail traffic unless somebody checks for it.

FINAL DELIVERY

With the advent of people who access the Internet by calling their ISP over a modem, it breaks down.

One solution is to have a message transfer agent on an ISP machine accept e-mail for its customers and store it in their mailboxes on an ISP machine. Since this agent can be on-line all the time, e-mail can be sent to it 24 hours a day.

POP3

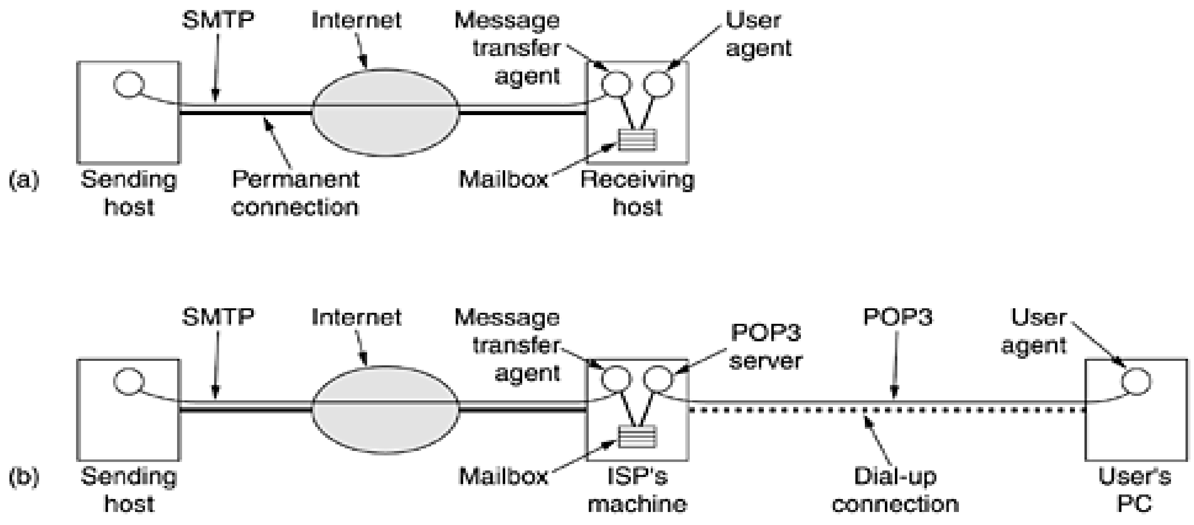


Figure:5-7

(a) Sending and reading mail when the receiver has a permanent Internet connection and the user agent runs on the same machine as the message transfer agent.

(b) Reading e-mail when the receiver has a dial-up connection to an ISP

POP3 begins when the user starts the mail reader. The mail reader calls up the ISP (unless there is already a connection) and establishes a TCP connection with the message transfer agent at port 110. Once the connection has been established, the POP3 protocol goes through three states in sequence:

1. Authorization.
2. Transactions.
3. Update.

The authorization state deals with having the user log in.

The transaction state deals with the user collecting the e-mails and marking them for deletion from the mailbox.

The update state actually causes the e-mails to be deleted.

IMAP (**Internet Message Access Protocol**).

POP3 normally downloads all stored messages at each contact, the result is that the user's e-mail quickly gets spread over multiple machines, more or less at random; some of them not even the user's.

This disadvantage gave rise to an alternative final delivery protocol, **IMAP** (**Internet Message Access Protocol**).

IMAP assumes that all the e-mail will remain on the server indefinitely in multiple mailboxes. IMAP provides extensive mechanisms for reading messages or even parts of messages, a feature useful when using a slow modem to read the text part of a multipart message with large audio and video attachments.

WORLD WIDE WEB

WORLD WIDE WEB

The World Wide Web is an architectural framework for accessing linked documents spread out over millions of machines all over the Internet. The initial proposal for a web of linked documents came from CERN physicist Tim Berners-Lee in 1989.

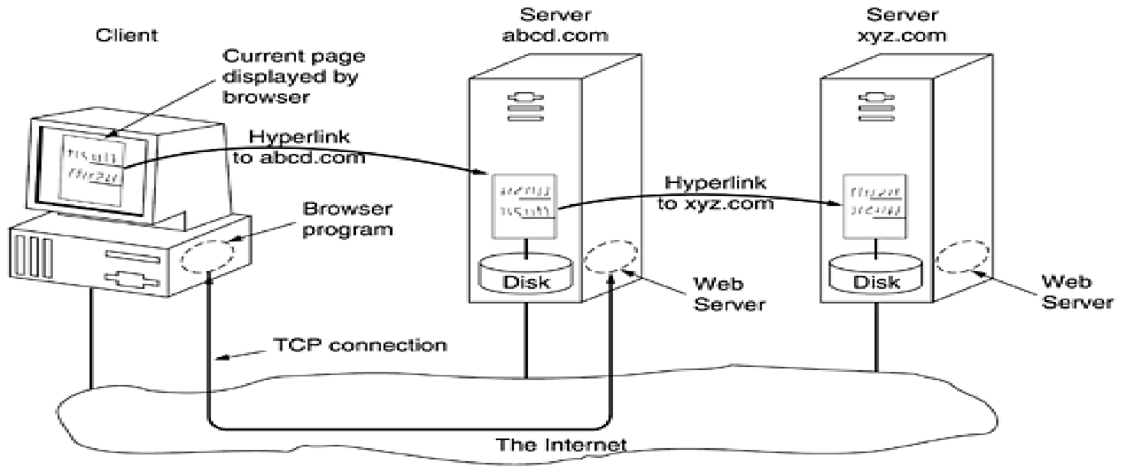
ARCHITECTURAL OVERVIEW

From the users' point of view, the Web consists of a vast, worldwide collection of documents or **Web pages.** Each page may contain links to other pages anywhere in the world. Users can follow a link by clicking on it, which then takes them to the page pointed to. This process can be repeated indefinitely.

Pages are viewed with a program called a **browser**, of which Internet Explorer and Netscape Navigator are two popular ones. The browser fetches the page requested, interprets the text and formatting commands on it, and displays the page, properly formatted, on the screen.

Strings of text that are links to other pages, called **hyperlinks**, are often highlighted, by underlining, displaying them in a special color, or both.

THE PARTS OF THE WEB MODEL



Here the browser is displaying a Web page on the client machine. When the user clicks on a line of text that is linked to a page on the *abcd.com* server, the browser follows the hyperlink by sending a message to the *abcd.com* server asking it for the page. When the page arrives, it is displayed. If this page contains a hyperlink to a page on the *xyz.com* server that is clicked on, the browser then sends a request to that machine for the page.

CLIENT SIDE

When an item is selected, the browser follows the hyperlink and fetches the page selected. Therefore, the embedded hyperlink needs a way to name any other page on the Web. Pages are named using **URLs** (**Uniform Resource Locators**).

The steps that occur at the client side are:

- The browser determines the URL
- The browser asks DNS for the IP address
- DNS replies with the IP address
- The browser makes a TCP connection to port 80 on the IP address
- It sends a request asking for file
- The *site* server sends the file
- The TCP connection is released.
- The browser fetches and displays all the text and images in the file.
- Web pages are written in standard HTML language to make it understandable by all browsers.

There are two possibilities: plug-ins and helper applications. A plug-in is a code module that the browser fetches from a special directory on the disk and installs as an extension to itself.

The other way to extend a browser is to use a helper application. This is a complete program, running as a separate process.

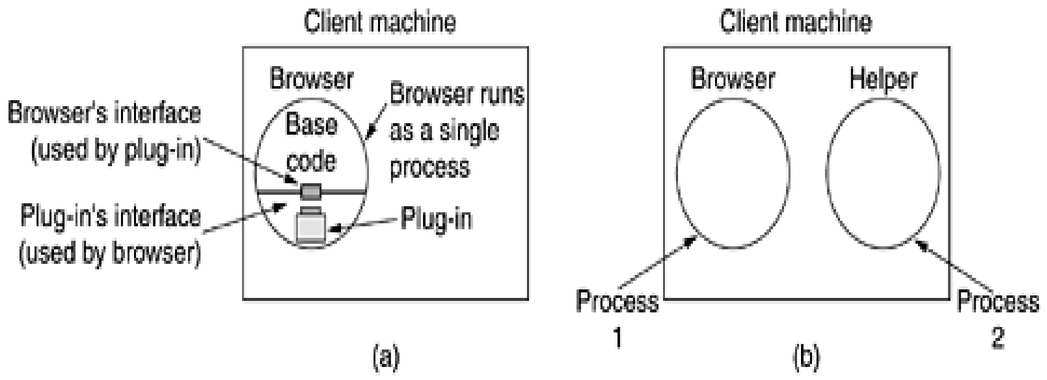

Client machine
Browser
Base code
Browser runs as a single process
Browser's interface (used by plug-in)
Plug-in's interface (used by browser)
Plug-in
(a)
Client machine
Browser
Helper
Process 1
Process 2
(b)

Figure 5-8. (a) A browser plug-in. (b) A helper application.

SERVER SIDE

The steps to be followed by the server side are:

1. Accept a TCP connection from a client (a browser).
2. Get the name of the file requested.
3. Get the file (from disk).
4. Return the file to the client.
5. Release the TCP connection.

PROCESSING OF REQUEST

The processing of request on the web is as follows:

1. Resolve the name of the Web page requested.
2. Authenticate the client.
3. Perform access control on the client.
4. Perform access control on the Web page.
5. Check the cache.
6. Fetch the requested page from disk.
7. Determine the MIME type to include in the response.
8. Take care of miscellaneous odds and ends.
9. Return the reply to the client.
10. Make an entry in the server log.

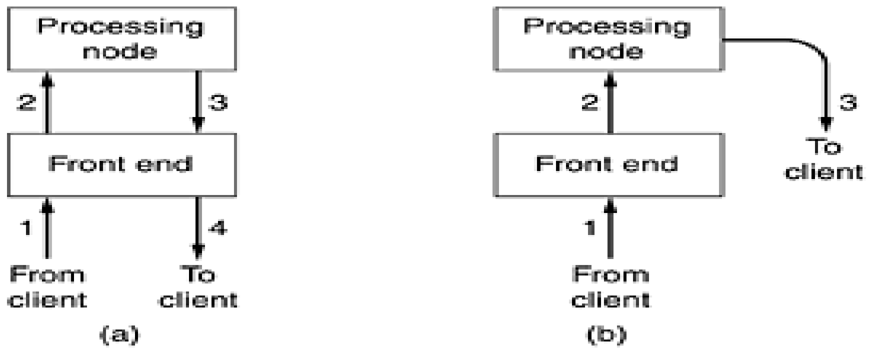


Figure 5-9. (a) Normal request-reply message sequence. (b) Sequence when TCP handoff is used

Sometimes a trick, called TCP handoff, is used to get around this problem. With this trick, the TCP end point is passed to the processing node so it can reply directly to the client.

URLs— UNIFORM RESOURCE LOCATORS

When the Web was first created, it was immediately apparent that having one page point to another Web page required mechanisms for naming and locating pages. In particular, three questions had to be answered before a selected page could be displayed:

1. What is the page called?
2. Where is the page located?
3. How can the page be accessed?

If every page were somehow assigned a unique name, there would not be any ambiguity in identifying pages. Nevertheless, the problem would not be solved.

Consider a parallel between people and pages. In the United States, almost everyone has a social security number, which is a unique identifier, as no two people are supposed to have the same one. Nevertheless, if you are armed only with a social security number, there is no way to find the owner's address, and certainly no way to tell whether you should write to the person in English, Spanish, or Chinese. The Web has basically the same problems.

The solution chosen identifies pages in a way that solves all three problems at once. Each page is assigned a **URL** (**Uniform Resource Locator**) that effectively serves as the page's worldwide name.

URLs have three parts: the protocol (also known as the **scheme**), the DNS name of the machine on which the page is located, and a local name uniquely indicating the specific page (usually just a file name on the machine where it resides). As an example, the Web site for the author's department contains several videos about the university and the city of Amsterdam. The URL for the video page is

http://www.cs.vu.nl/video/index-en.html

This URL consists of three parts: the protocol (http), the DNS name of the host (www.cs.vu.nl), and the file name (video/index-en.html), with certain punctuation separating the pieces. The file name is a path relative to the default Web directory at cs.vu.nl.

STATIC WEB DOCUMENTS

- The basis of the Web is transferring Web pages from server to client. In the simplest form, Web pages are static. They are just files sitting on some server waiting to be retrived.
- In this context, even a video is a static web page because it is just a file.
- In this section we will look at static web page in details. In the next one, we will examine dynamic content.

HTML—The HyperText Markup Language:

- HTML allows users to produce Web pages that include text, graphics, video, pointers toother Web pages, and more.
- HTML is a markup language, or language for describing how documents are to be formatted.
- Markup languages thus contain explicit commands for formatting. For example, in HTML, <b>means start boldface mode, and </b> means leave boldface mode.
- Writing a browser is then straightforward: the browser simply has to understand the markup commands.
- Embedding all the markup commands within each HTML file and standardizing them makes it possible for any Web browser to read and reformat any Web page.
- While it is certainly possible to write documents like this with any plain text editor, and many people do, it is also possible to use word processors or special HTML editors that do most of the work.
- A Web page consists of a head and a body, each enclosed by<html>and </html>tags.
- The head is bracketed by the <head>and </head>tags and the body is bracketed by the<body>and </body>tags. The strings inside the tags are called directives.
- Most, but not all, HTML tags have this format. That is, they use <something>to mark the beginning of something and </something>to mark its end.
- Tags can be in either lowercase or uppercase. Thus, <head>and <HEAD>mean the same thing, but lower case is best for compatibility.
- Some tags have (named) parameters, called attributes. For example, the<img>tag is used for including an image inline with the text. It has two attributes, src and alt. The first attribute gives the URL for the image.
- The list of special characters is given in the standard. All of them begin with an ampersand and end with a semicolon.
    - For example, &nbsp; produces a space, &egrave; produces è and &eacute; producesé. Since <, >, and & have special meanings, they can be expressed onlywith their escape sequences, &lt;, &gt;, and &amp;.
- The main item in the head is the title, delimited by <title>and </title>.The title itself is not displayed on the page. Some browsers use it to label the page's window.
- Several headings are used in each heading is generated by an <h*n*>tag, where *n* is a digit in the range 1 to 6. Thus, <h1>is the most important heading ;<h6>is the least important one.

- <h1>headings are large and boldface with at least one blank line above and below. In contrast, <h2>headings are in a smaller font with less space above and below.
- The tags <b> and <i> are used to enter boldface and italics mode. The <p> tag starts a paragraph. the </p> tag that exists to mark the end of a paragraph

Tag	**Description**
<html>…</html>	Declares the web page to be written in HTML.
<head>…</head>	Delimits the page's head.
<title>…</title>	Defines the title.
<body>…</body>	Delimits the page's body.
<h n>…</h n>	Delimits a level n heading.
<b> … </b>	Set … in boldface.
<i> … </i>	Set … in italic.
<center>…</center>	Center … on the page horizontally.
<ul> … </ul>	Brackets an unordered list.
<ol> … </ol>	Brackets a numbered list.
<li> … </li>	Brackets an item in an ordered or numbered list.
<br>	Forces a line break here.
<p>	Starts a paragraph.
<hr>	Inserts a horizontal rule.
<img src="">	Displays an image here.
<a href="…">….</a>	Defines a hyperlink.

XML and XSL:

- XML and XSL is (eXtensible Markup Language) and (eXtensible Style Language).
- HTML, with or without forms, does not provide any structure to web pages.
- It also mixes the content with the formatting, as e-commerce and other applications become more common, there is an increasing need for structuring pages and separating the content from the formatting.
- The W#C has developed an enhancement to HTML to allow web pages to be structured for automated processing.
- It defines a structure called book_list, which is a list of books. Each book has three fields, the title, author, and year of publication.
- In this example, each of the three fields is an indivisible entity, but it is also permitted to futher subdivide the fields.
- The author fields could have been done as follows to give a finer-grained control over searching and formatting.

- Example:
  <author>
  <first_name>Andrew</first_name>
  <last_name>Tanenbaum</last_name>
  </author>
- Each field can be subdivided into subfields and sub subfields arbitrarily deep.
- The file is a style sheet that tells how to display the page, it is design view in the xml file.

XHTML: (The eXtended HyperText Markup Language)

HTML keeps evolving to meet new demands. Many people in the industry feel that in the future, the majority of web-enabled device will not be PCs, but wireless, handheld PDA-type device.

- These devices have limited memory for large browser full of heuristics that try to somehow deal with syntactically incorrect web pages.
- There are 6 major difference in HTML and XHML:

  - ➢ XHTML pages and browser must strictly conform to the standard. No more shodly web pages.
  - ➢ All tags and attributes must be in lower case, tags like <HTML> are not valid in XHTML.
  - ➢ Closing tags are required, even for </p>. for tags that have no natural closing tag. Such as <br>,<hr> and <img>,a slash must preced the closing">".

    Eg:<img src="pic0001.jpg"/>
  - ➢ Attribute must be contained within quotation marks.

    Eg: <img src="pic001.jpg" height=500/>

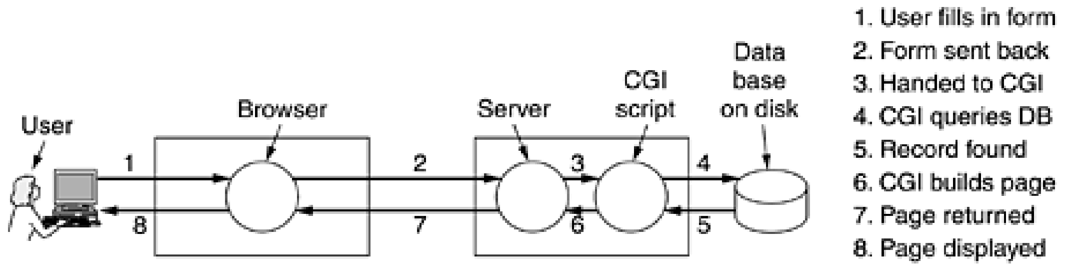
    The 500 has to be enclosed in quotation marks, just like the name of the JPEG file, even though 500 is just a number.
  - ➢ Tags must be nest properly. In the past, proper nesting was not required as long as final state achieved was correct. Tags closed in the inverse order that they were opened.

    Eg: <center><b>vacation pictures</center></b>.
  - ➢ Every document must specify its document type. For a discussion of all the changes, major and minor, see www.w3.org.

DYNAMIC WEB DOCUMENTS

Dynamic web documents are created at both client and server sides.

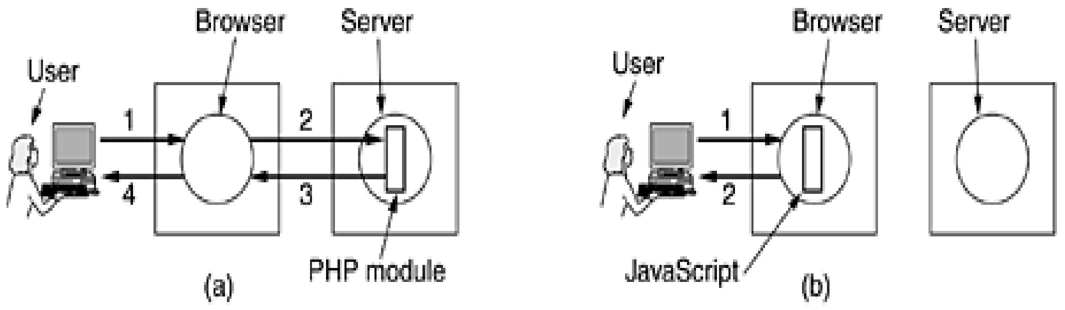
SERVER-SIDE GENERATION



The server side generation involves the following steps:

- User fills in form.
- Form sent back
- Handed to CGI
- CGI queries database.
- Record found
- CGI builds page
- Page returned
- Page displayed

CLIENT-SIDE GENERATION



CGI, PHP, JSP, and ASP scripts solve the problem of handling forms and interactions with databases on the server. They can all accept incoming information from forms, look up information in one or more databases, and generate HTML pages with the results.

Usually the server side scripting is done with PHP and client side scripting is javascript. Complete web pages can be generated on the fly by various scripts on the server machine. Once they are received by the browser, they are treated as normal HTML pages and displayed.

Dynamic content generation is also possible on the client side. Web pages can be written in XML and then converted to HTML according to XSL file. Javascript programs can perform arbitrary computations.

Finally plugins and helper applications can be used to display content in a variety of formats.

NETWORK SECURITY

- The requirements of information security within an organization have undergone two major changes in the last decades.
- The generic name for the collection of tools designed **to protect data and to prevent hackers is computer security.**
- Another important thing is that affected security is the introduction of distributed systems and the use of networks and communication facilities for carrying data between terminal user and computer and between computer and compiler.
- Network security measures are needed to protect data during their transmission and to guarantee that data transmissions are authentic.

SECURITY ATTACKS:

Any action that compromises the security of information owned by an organization.

- Two types of security attacks are

  1)Passive Attack-Just listens the message.

  2)Active Attack-Alter message.

SECURITY MECHANISMS:

- A process that designed to detect ,prevent,recover from security attack.

SECURITY SERVICE:

- Use of one or mor e security mechanisms to provide service.

IMPORTANT FEATURES OF SECURITY:

1)CONFIDENTIALITY-It is the protection of transmitted data from passive attacks.

2)AUTHENTICATION-The receiptient that the message is from source that it claim to be from.

3)NON REPUDIATION-It prevent either sender or receiver.

- When a message is sent ,the receiver can prove that the alleged sender in fact sent themessage similarly when a message is received,the sender can prove that alleged receiver in fact received the message.

SECURITY REQUIREMENTS AND ATTACKS

Computer and network security address three requirements.

- Secrecy
- Integrity
- Availability

SECRECY

Secrecy requires that the information in a computer system only be accessible for reading by authorized parties. This type of access includes printing, displaying and other forms of disclosure, including simply revealing the existence of an object.

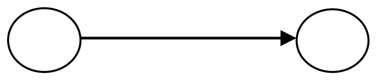
INTEGRITY

Integrity requires that computer system can be modified only by authorized parties. Modification includes writing, changing status, deleting and creating.

AVAILABILITY

- Availability requires that computer systems are available to authorized parties.
- The types of attacks on the security of a computer system or network can be viewed by the function of the computer system.

In general, there is a flow of information from a source to destination. This is called normal flow.



Information source destination source

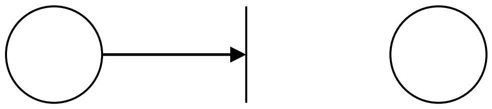
The following are the four categories of attack.

They are:

- Interruption
- Interception
- Modification
- Fabrication

INTERRUPTION

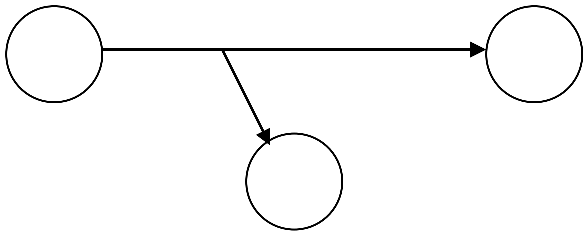
An asset of the system is destroyed or it becomes unavailable or unusable. This is an attack on availability. Examples: destruction of a piece of hardware such as hard disk, the cutting of a communication line, or the disabling of the file management system.



INTERCEPTION

An authorized party gains access to an asset. This is an attack on confidentiality. The authorized party could be a person or program or a computer.

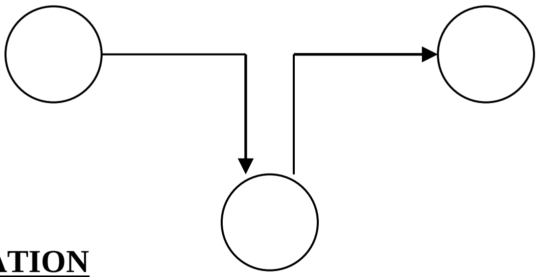
E.g.: wiretapping to capture data in a network and the illicit copying of files or programs.



MODIFICATION

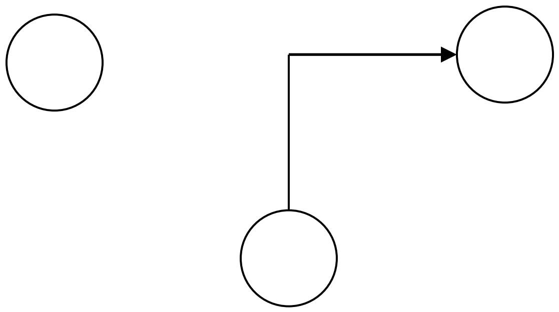
An authorized party gains access to an asset. This is an attack on confidentiality. This is an attack on integrity.

E.g: changing the values in a data file, altering a program so that it performs differently and modifying the content of messages being transmitted in a network.



FABRICATION

An authorized party inserts counterfeit objects into the system. This is an attack on authenticity. Examples include the insertion of spurious messages in a network or the addition of records to a file.



PASSIVE ATTACKS

Passive attacks means the eavesdropping on or monitoring of, transmissions. The goal of the component is to obtain information that is being transmitted. Two types of attacks are involved here.

- Release of message contents
- Traffic analysis

RELEASE OF MESSAGE CONTENTS

In this, a telephone conversation, an e-mail message, a transferred file may contain sensitive or confidential information. This helps to prevent the opponent from learning the content of these transmissions.

TRAFFIC ANALYSIS

In this encryption is used for masking the contents which helps to observe the pattern of the messages. The opponent could determine the location and identity of communicating hosts and could observe the frequency and length of messages being exchanged.

Passive attacks are difficult to detect because they do not involve any alteration of data.

ACTIVE ATTACKS

In this, the attacks involve some modification of the data stream or creation of false stream. This is divided into four categories.

They are:

- Masquerade
- Replay
- Modification of messages
- Denial of service

- **Masquerade**

A masquerade takes place when one entity pretends to be a different entity. A masquerade attack usually includes one of the other forms of active attacks.

- **Replay**

Replay involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect.

- **Modification**

This means that some portion of a legitimate message is altered or that messages are delayed or reordered to produce an unauthorized effect.

- **Denial of service**

The denial of service prevents or inhibits the normal size or management of communication facilities.

Active attacks present the opposite characteristics of passive attacks.

It is quite difficult to prevent active attacks absolutely and would require physical protections of all communications facilities and paths at all times.

CRYPTOGRAPHY:

- Cryptography is the study of secret(crypto)writing(graphy)
- The art of science encompassing the principles and methods of transforming an intelligible message into one that is intelligible,and then retransforming that message back to its original form.

Some of the encryption scheme are.

PLAINTEXT:This is the original intelligible message.

CIPHERTEXT:Transformed message.

ENCRYPTION ALGORITHMS:The encryption algorithm performs various substitutions and transformations on the plaintext.

SECRET KEY: Some critical information used by the cipher knows only to sender and receiver.

ENCIPHER(ENCODE): The process of converting plaintext to ciphertext.

DECIPHER(DECODE): The process of converting ciphertext back into plaintext.

CRYPTOGRAPHY SYSTEMS ARE CHARACTERISED INTO THREE:

1)The type of operation used for transforming plaintext to ciphertext.

Two general principles

- Substitution.
- Tranposition.

SUBSTITUTION: In which each element in the plaintext(bit,letter,group or letter is mapped into another element.

TRANSPOSITION:In which elements in plaintext are rearranged.

2)The numbers of keys used.

Both sender and receiver use same key as

- Symmetric
- Single key
- Secret key or conventional encryption.

Both sender and receiver use different key

- Asymmetric ,two key or public key encryption.

3)The way in which plaintext is processed.

- **A Block cipher** processes the input one block of elements at a time ,producing an output block for each input block.(ie.Encrypt one bit/character at a time)

  Eg:THIS IS EASY KEY(3)

  WKLV LV HDVB.

- **A stream cipher** processs the input elements continuosly,producing output one element at a time.(ie. Break plaintext message in equal size blocks and encrypt each block as a unit).
  Eg:THIS IS EASY

  THI ISI SEA SY….. KEY(135)
  UKN JWN……

INTRODUCTION TO CRYPTOGRAPHY

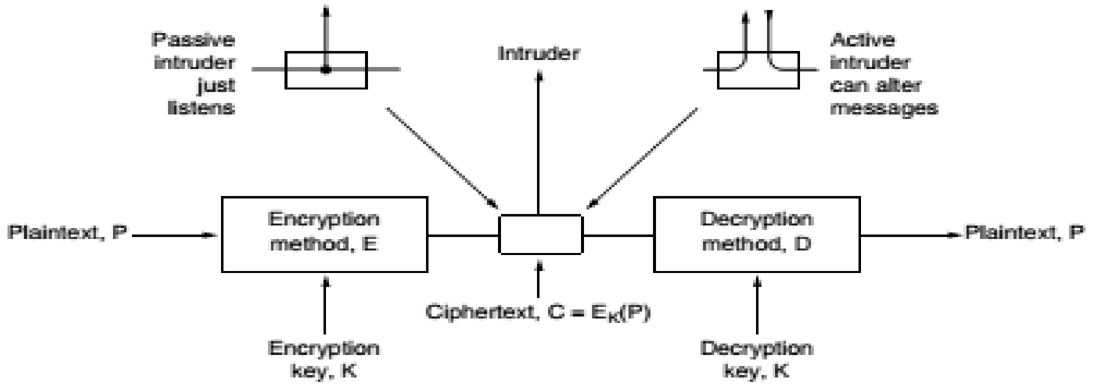


Figure 8-2. The encryption model (for a symmetric-key cipher).

- The messages to be encrypted, known as the plaintext, are transformed by a function that is parameterized by a key.
- The output of the encryption process, known as the ciphertext, is then transmitted, often by messenger or radio.
- We assume that the enemy, or intruder, hears and accurately copies down the complete ciphertext.
- However, unlike the intended recipient, he does not know what the decryption key is and so cannot decrypt the ciphertext easily.
- Sometimes the intruder can not only listen to the communication channel (passive intruder) but can also record messages and play them back later, inject his own messages, or modify legitimate messages before they get to the receiver(active intruder)

<u>SUBSTITUTION TECHNIQUES:</u>

- The two basic building block of all encryption techniques are substitution and transposition.
- A substitution techniques is one in which the letters of plaintext are replaced by other letters or by number or symbols.

<u>CAESAR CIPHER</u>:

- The earliest known use of a substitution cipher and the simplest,was by Julius Caesar.
- The Caesar cipher involves replacing each letter of alphabets with the letter standing three places further down the alphabets.

Eg:plaintext: meet me after the toga party.

Ciphertext:PHHW PH DIWHU WKH WRJD SDUWB.

-The letter following Z to A.

Plaintext: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z.

Ciphertext: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

<u>MONOALPHABETIC CIPHER:</u>

- The general system of symbol for symbol substitution cipher.
- With the key being 26 letter string corresponding to the full alphabet.

Eg:A->X,B->Y,C->Z,D->A…….Z->W

Eg) starbucks at three

PQXOYRZHP XQ QEOBB

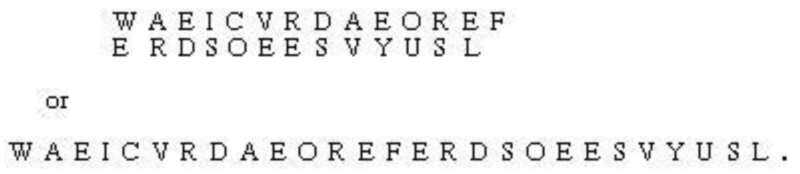
- The basic attack takes advantage of the statistical properties of natural languages.
- In English, for example ,e is the most common letter, followed by t, o, a, n, i, etc.

- The most common two-letter combinations, or digrams, are th, in, er, re, and an.
- The most common three-letter combinations, or trigrams,are the, ing, and, and ion.
- A cryptanalyst trying to break a monoalphabetic cipher would start out by
- counting the relative frequencies of all letters in the ciphertext.
- He would then look at trigrams to find a common one of the form tXe, which strongly suggests that
- X is h. Similarly, if the pattern thYt occurs frequently, they probably stands for a.

TRANSPOSITION TECHNIQUES:

- A very different kind of mapping is achieved by performing some sort of permutation on plaintext letters.
- This technique is referred as transposition cipher.
- In which letters of the plaintext are written alternating between rows and the rows and then read sequentially to give cipher.

WE ARE DISCOVERED SAVE YOURSELF would be written

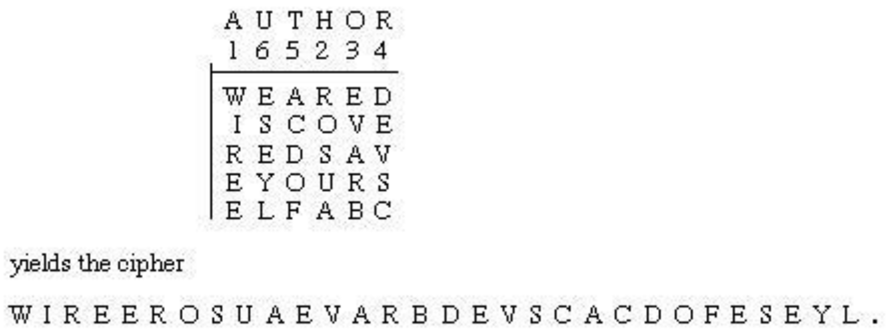


```
W A E I C V R D A E O R E F
E R D S O E E S V Y U S L
```

or

```
W A E I C V R D A E O R E F E R D S O E E S V Y U S L .
```

- To write the message in rectangle row by row and read the message off,column by column but permute the order of column.
- The order of column then becomes keyword AUTHOR and order the column by lexicographic order of the letters in the keyword.



A	U	T	H	O	R
1	6	5	2	3	4
W	E	A	R	E	D
I	S	C	O	V	E
R	E	D	S	A	V
E	Y	O	U	R	S
E	L	F	A	B	C

yields the cipher

```
W I R E E R O S U A E V A R B D E V S C A C D O F E S E Y L .
```

One-Time pad:

Introduction:

- First described by Frank Miller in 1882.
- The one-time pad was re-invented in 1917 and patented a couple of years later.

- It is derived from the *Vernam cipher*, named after Gilbert Vernam, one of its inventors.
- Vernam's system was a cipher that combined a message with a key read from a punched tape.
- In its original form, Vernam's system was vulnerable because the key tape was a loop, which was reused whenever the loop made a full cycle.
- One-time use came later, when Joseph Mauborgne recognized that if the key tape were totally random, then cryptanalysis would be impossible.

Or

- The One-Time Pad is an evolution of the Vernham cipher, which was invented by Gilbert Vernham in 1918, and used a long tape of random letters to encrypt the message.
- An Army Signal Corp officer, Joseph Mauborgne, proposed an improvement using a random key that was truly as long as the message, with no repetitions, which thus totally obscures the original message.
- Since any plaintext can be mapped to any ciphertext given some key, there is simply no way to determine which plaintext corresponds to a specific instance of ciphertext.

Define:

- Each new message requires a new key of the same length as the new message. Such a scheme, known as a one-time pad.
- It is unbreakable.
- It produces a random output that bears no statistical relationship to the plaintext.
- Because the ciphertext contains no information whatsoever about the plaintext, there is simply no way to break the code.

Example:

Suppose Alice wishes to send the message "HELLO" to Bob. Assume two pads of paper containing identical random sequences of letters were somehow previously produced and securely issued to both. Alice chooses the appropriate unused page from the pad.

The way to do this is normally arranged for in advance, as for instance 'use the 12th sheet on 1 May', or 'use the next available sheet for the next message'. The material on the selected sheet is the *key* for this message. Each letter from the pad will be combined in a predetermined way with one letter of the message. It is common, but not required, to assign each letter a numerical value, e.g., "A" is 0, "B" is 1, and so on. In this example, the technique is to combine the key and the message using modular addition. The numerical values of corresponding message and key letters are added together, modulo 26. If key material begins with "XMCKL" and the message is "HELLO", then the coding would be done as follows:

```
      H       E       L       L       O  message
   7 (H)   4 (E)  11 (L)  11 (L)  14 (O) message
+ 23 (X)  12 (M)   2 (C)  10 (K)  11 (L) key
= 30      16      13      21      25     message + key
=  4 (E)  16 (Q)  13 (N)  21 (V)  25 (Z) message + key (mod 26)
```

```
    E     Q     N     V     Z  → ciphertext
```

If a number is larger than 25, then the remainder after subtraction of 26 is taken in modular arithmetic fashion. This simply means that if the computations "go past" Z, the sequence starts again at A.

The ciphertext to be sent to Bob is thus "EQNVZ". Bob uses the matching key page and the same process, but in reverse, to obtain the plaintext. Here the key is *subtracted* from the ciphertext, again using modular arithmetic:

```
       E       Q       N       V       Z  ciphertext
    4 (E)  16 (Q)  13 (N)  21 (V)  25 (Z) ciphertext
-  23 (X)  12 (M)   2 (C)  10 (K)  11 (L) key
= -19       4      11      11      14     ciphertext – key
=   7 (H)   4 (E)  11 (L)  11 (L)  14 (O) ciphertext – key (mod 26)
       H       E       L       L       O  → message
```

Similar to the above, if a number is negative then 26 is added to make the number zero or higher.

- The Security of the one-time pas is entirely due to the randomness of the key.
- If the Stream of characters that constitute the key is truly random, then the stream of characters that constitute the ciphertext will be truly random.
- Thus, there are no patterns or regularities that a cryptanalyst can use to attack the ciphertext.

The one-time pad offers complete security but, in practice, has two fundamental difficulties:

1. There is the practical problem of making large quantitied of random keys. Any heavily used system might requires millions of random character on a regular basis. Supply truly random characters in this volume is a significant task.
2. Even more daunting is the problem of key distribution and protection. For every message to be sent, a key of equal length is needed by both sender and receiver. Thus, a mammoth key distribution problem exists.

Cryptographic Principles:

- Redundancy
  - All encrypted messages must contain some redundancy, that is, information not needed to understand the message.
- Freshness
  - Some measures must be taken to ensure that each message received can be verified as being fresh, that is, sent very recently.

Redundancy Motivation:

- Consider a mail-order company, The Couch Potato (TCP), with 60,000 products.
- Ordering messages consist of a 16-byte customer name followed by a 3-byte data field.

- The last 3 bytes are to be encrypted using a very long key known only by the customer and TCP.
- This might seem secure since passive intruders cannot decrypt the messages.
- Suppose that a recently-fired employee wants to punish TCP.
- Just before leaving, he takes the customer list with him.
- He writes a program to generate fictitious orders using real customer names.
- Since he does not have the list of keys, he just puts random numbers in the last 3 bytes, and sends hundreds of orders.
- When these messages arrive, TCP's computer uses the customer's name to locate the key and decrypt the message.
- Unfortunately for TCP, almost every 3-byte message is valid, so the computer begins printing out shipping instructions.
- In this way an active intruder can cause a massive amount of trouble, even though he cannot understand the messages his computer is generating.
- This problem can be solved by the addition of redundancy to all messages.
- For example, if order messages are extended to 12 bytes, the first 9 of which must be zeros, then this attack no longer works because the ex-employee can no longer generate a large stream of valid messages.

All messages must contain considerable redundancy so that active intruders cannot send random junk and have it be interpreted as a valid message

Freshness

- This measure is needed to prevent active intruders from playing back old messages.
- If no such measures were taken, our ex-employee could keep repeating previously sent valid messages.
- Some method is needed to foil replay attacks
- A solution is to include in every message a timestamp valid only for, say, 10 seconds.
- The receiver can then just keep messages around for 10 seconds. Messages older than 10 seconds can be thrown out.

<u>RSA</u>

- RSA is a public key algorithm.
- It is the first algorithm known to be suitable for signing as well as encryption, and was one of the first great advances in public key cryptography.
- RSA is widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and the use of up-to-date implementations.

Much practical security is based on it. Its major disadvantage is that it requires keys of at least 1024 bits for good security (versus 128 bits for symmetric-key algorithms), which makes it quite slow.

The RSA algorithm involves three steps:

- key generation
- encryption
- decryption

RSA involves a **public** key and a **private key.** The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted using the private key.

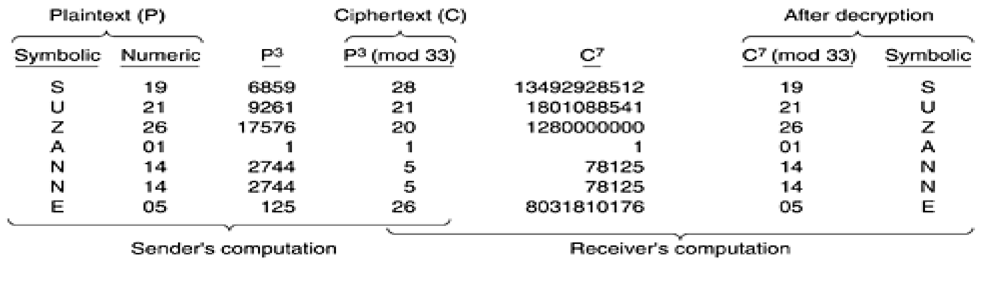
The RSA method is based on some principles from number theory. We will now summarize how to use the method.

1. Choose two large primes, *p* and *q* (typically 1024 bits).
2. Compute $n = p \times q$ and $z = (p - 1) \times (q - 1)$.
3. Choose a number relatively prime to *z* and call it *d*.
4. Find *e* such that $e \times d = 1 \; mod \; z$.

The security of the method is based on the difficulty of factoring large numbers. If the cryptanalyst could factor the (publicly known) *n*, he could then find *p* and *q*, and from these *z*.

With these parameters computed in advance, we are ready to begin encryption. Divide the plaintext (regarded as a bit string) into blocks, so that each plaintext message, *P*, falls in the interval $0 \; P < n$. Do that by grouping the plaintext into blocks of *k* bits, where *k* is the largest integer for which $2^k < n$is true.

To encrypt a message, *P*, compute $C = P^e$ (mod *n*). To decrypt *C*, compute $P = C^d$ (mod *n*). It can be proven that for all *P* in the specified range, the encryption and decryption functions are inverses. To perform the encryption, you need *e* and *n*. To perform the decryption, you need *d* and *n*. Therefore, the public key consists of the pair (*e*, *n*), and the private key consists of (*d*, *n*).



Plaintext (P)			Ciphertext (C)		After decryption	
Symbolic	Numeric	P^3	P^3 (mod 33)	C^7	C^7 (mod 33)	Symbolic
S	19	6859	28	13492928512	19	S
U	21	9261	21	1801088541	21	U
Z	26	17576	20	1280000000	26	Z
A	01	1	1	1	01	A
N	14	2744	5	78125	14	N
N	14	2744	5	78125	14	N
E	05	125	26	8031810176	05	E

Sender's computation

Receiver's computation

In the above example, the encryption of the plain text "SUZANNE" is shown:

p = 3, q = 11, n = 33, z = 20

- d = 7 (since 7 and 20 have no common factors)
- 7e = 1 (mod 20)
- e = 3
- C = P3 (mod 33)
- P = C7 (mod 33)

Encryption:

C = Me mod n
= 887 mod 187
= [(884 mod 187) * (882 mod 187) * (881 mod 187] mod 187
= [(59,969,536 mod 187)(7744 mod 187)(88 mod 187)] mod187
= (132 * 77 * 88) mod 187
= 894,432 mod 187
= 11

Decryption:

M =Cd mod n
=1123 mod 187
=[(111 mod 187)* (112 mod 187)* (114 mod 187)*(118 mod 187)* (118mod 187)] mod 187
=[(111mod187)*(121mod187)*(14,641mod187)*(214,358,881mod187)*(214,358,881mod187)]mod187
=(11 * 121 * 55 * 33 * 33) mod 187
=79,720,245 mod 187
=88.

If p = 3, q = 11, n = 33, Φ(n) = 20,
d = 7 (because 7, 20 have no common factors)
=>7e = 1 mod 20
=3

Some form of chaining is needed for data encryption. However, in practice, most RSA-based systems use public-key cryptography primarily for distributing one-time session keys for use with some symmetric-key algorithm such as AES or triple DES. RSA is too slow for actually encrypting large volumes of data but is widely used for key distribution.

8.7.4 Authentication Using Kerberos

An authentication protocol used in many real systems (including Windows 2000 and later versions) is **Kerberos**, which is based on a variant of Needham-Schroeder. It is named for a multiheaded dog in Greek mythology that used to guard the entrance to Hades (presumably to keep undesirables out). Kerberos was designed at M.I.T. to allow workstation users to access network resources in a secure way. Its biggest difference from Needham-Schroeder is its assumption that all clocks are fairly well synchronized. The protocol has gone through several iterations. V5 is the one that is widely used in industry and defined in RFC 4120. The earlier version, V4, was finally retired after serious flaws were found (Yu et al., 2004). V5 improves on V4 with many small changes to the protocol and some improved features, such as the fact that it no longer relies on the now-dated DES. For more information, see Neuman and Ts'o (1994).

Kerberos involves three servers in addition to Alice (a client workstation):

1. Authentication Server (AS): Verifies users during login.
2. Ticket-Granting Server (TGS): Issues "proof of identity tickets."
3. Bob the server: Actually does the work Alice wants performed.

AS is similar to a KDC in that it shares a secret password with every user. The TGS's job is to issue tickets that can convince the real servers that the bearer of a TGS ticket really is who he or she claims to be.

To start a session, Alice sits down at an arbitrary public workstation and types her name. The workstation sends her name and the name of the TGS to the AS in plaintext, as shown in message 1 of Fig. 8-42. What comes back is a session key and a ticket, $K_{TGS}(A, K_S, t)$, intended for the TGS. The session key is encrypted using Alice's secret key, so that only Alice can decrypt it. Only when message 2 arrives does the workstation ask for Alice's password—not before then. The password is then used to generate K_A in order to decrypt message 2 and obtain the session key.

At this point, the workstation overwrites Alice's password to make sure that it is only inside the workstation for a few milliseconds at most. If Trudy tries logging in as Alice, the password she types will be wrong and the workstation will detect this because the standard part of message 2 will be incorrect.

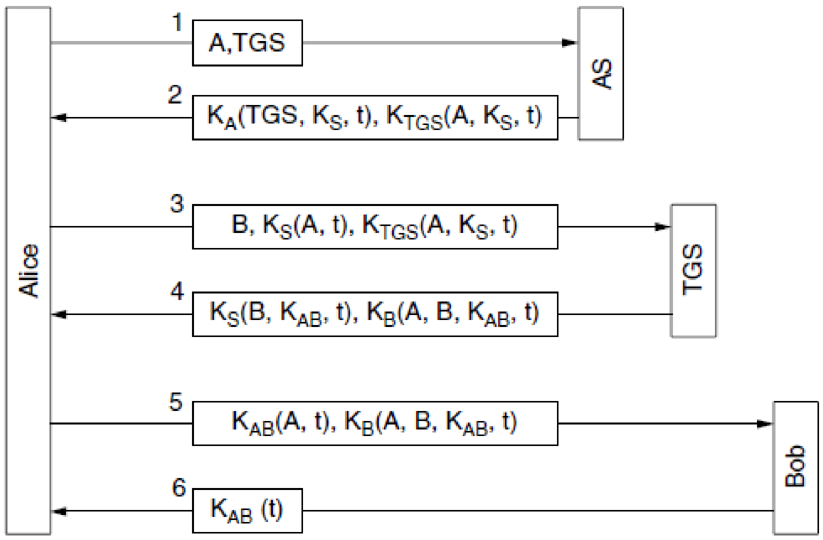

Alice
AS
TGS
Bob
1
A,TGS
2
K_A(TGS, K_S, t), K_TGS(A, K_S, t)
3
B, K_S(A, t), K_TGS(A, K_S, t)
4
K_S(B, K_AB, t), K_B(A, B, K_AB, t)
5
K_AB(A, t), K_B(A, B, K_AB, t)
6
K_AB (t)

Figure 8-42. The operation of Kerberos V5.

After she logs in, Alice may tell the workstation that she wants to contact Bob the file server. The workstation then sends message 3 to the TGS asking for a ticket to use with Bob. The key element in this request is the ticket $K_{TGS}(A, K_{S,} t)$, which is encrypted with the TGS's secret key and used as proof that the sender really is Alice. The TGS responds in message 4 by creating a session key, K_{AB}, for Alice to use with Bob. Two versions of it are sent back. The first is encrypted with only K_S, so Alice can read it. The second is another ticket, encrypted with Bob's key, K_B, so Bob can read it.

Trudy can copy message 3 and try to use it again, but she will be foiled by the encrypted timestamp, t, sent along with it. Trudy cannot replace the timestamp with a more recent one, because she does not know K_S, the session key Alice uses to talk to the TGS. Even if Trudy replays message 3 quickly, all she will get is another copy of message 4, which she could not decrypt the first time and will not be able to decrypt the second time either.

Now Alice can send K_{AB} to Bob via the new ticket to establish a session with him (message 5). This exchange is also timestamped. The optional response (message 6) is proof to Alice that she is actually talking to Bob, not to Trudy.

After this series of exchanges, Alice can communicate with Bob under cover of K_{AB}. If she later decides she needs to talk to another server, Carol, she just repeats message 3 to the TGS, only now specifying C instead of B. The TGS will promptly respond with a ticket encrypted with K_C that Alice can send to Carol and that Carol will accept as proof that it came from Alice.

The point of all this work is that now Alice can access servers all over the network in a secure way and her password never has to go over the network. In fact, it only had to be in her own workstation for a few milliseconds. However, note that each server does its own authorization. When Alice presents her ticket to Bob, this merely proves to Bob who sent it. Precisely what Alice is allowed to do is up to Bob.

Since the Kerberos designers did not expect the entire world to trust a single authentication server, they made provision for having multiple **realms**, each with its own AS and TGS. To get a ticket for a server in a distant realm, Alice would ask her own TGS for a ticket accepted by the TGS in the distant realm. If the distant TGS has registered with the local TGS (the same way local servers do), the local TGS will give Alice a ticket valid at the distant TGS. She can then do business over there, such as getting tickets for servers in that realm. Note, however, that for parties in two realms to do business, each one must trust the other's TGS. Otherwise, they cannot do business.