



II BCA Computer networks CM

Bachelor of Computer Application (Bharathiar University)



Scan to open on Studocu

ERODE ARTS AND SCIENCE COLLEGE (AUTONOMOUS), ERODE
DEPARTMENT OF COMPUTER APPLICATIONS
ACADEMIC YEAR 2019-2020 EVEN SEMESTER
COURSE MATERIALS

SUBJECT: COMPUTER NETWORKS

CLASS: II BCA & II B.SC IT

UNIT: I

Introduction: Uses of Computer Networks - Network Hardware - Network Software - Protocol Hierarchies. Design issues for the Layers - Interfaces - Services - Connection-Oriented and Connection Less Services. Service Primitives - The Relationship of Services to Protocols. Reference Models : The OSI Reference Model - the TCP/IP Reference Model - a Comparison of the OSI and TCP Reference Models.

Introduction

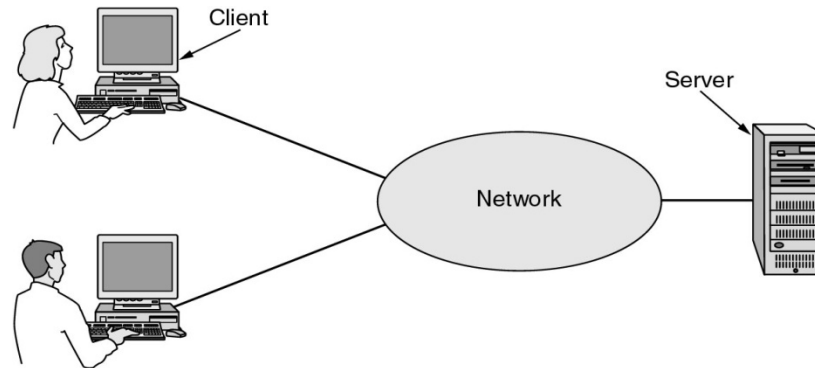
- The Historical Perspective
 - The 18th century: the great mechanical systems accompanying the Industrial Revolution.
 - The 19th century: steam engine.
 - The 20th century: information gathering, processing, and distribution.
 - The 21st century: Internet, large distributed systems (e.g., Grid), heavy reliance on computers.
- What is Computer Network?
 - We will use the term "computer network" to mean a collection of autonomous computers interconnected by a single technology.
 - Two computers are said to be interconnected if they are able to exchange information.
 - Copper wire; fiber optics, microwaves, infrared, and communication satellites, etc.

Uses of Computer Networks

- Business Applications
- Home Applications
- Mobile Users
- Social Issues

- Business Applications of Networks

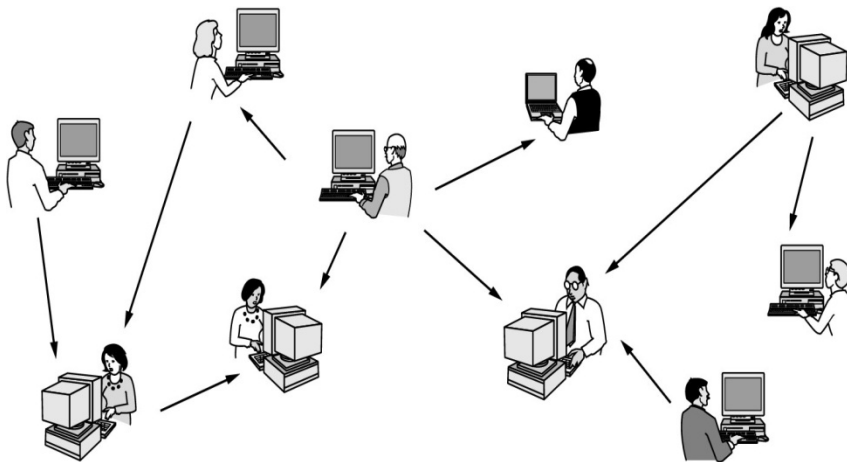
- Resource sharing (hardware, software, information, ...)
- Providing communication medium (e-mail, videoconferencing)
- Doing business electronically (B2B, B2C, e-commerce)



A network with two clients and one server

- Home Network Applications

- Access to remote information
- Person-to-person communication
- Interactive entertainment
- Electronic commerce



In peer-to-peer system there are no fixed clients and servers

- Mobile Network Users

Wireless	Mobile	Applications
No	No	Desktop computers in offices
No	Yes	A notebook computer used in a hotel room
Yes	No	Networks in older, unwired buildings
Yes	Yes	Portable office; PDA for store inventory

Combinations of wireless networks and mobile computing

- Social Issues
 - Politics
 - Religion
 - Sex
 - Privacy
 - Crime
 - junk mail, ill-information

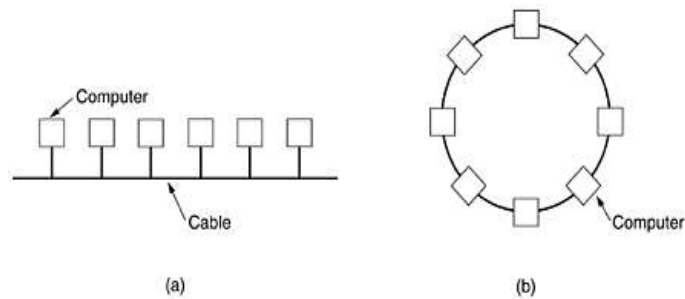
Network Hardware (Transmission Technologies)

- **Broadcast**
 - Broadcast networks have a single communication channel that is shared by all the machines on the network. Short messages, called packets in certain contexts, sent by any machine are received by all the others. (Address Checking required)
- **Point-to-point**
 - In point-to-point networks, there consist of many connections between individual pairs of machines.
- As a general rule (although there are many exceptions), smaller, geographically localized networks tend to use broadcasting, whereas larger networks usually are point-to-point.

Network Hardware

- personal area networks (1m)
- LAN: local area networks (10m ~ 1km)
- MAN: metropolitan area networks (10km)
- WAN: wide area network (100km ~ 1000km)

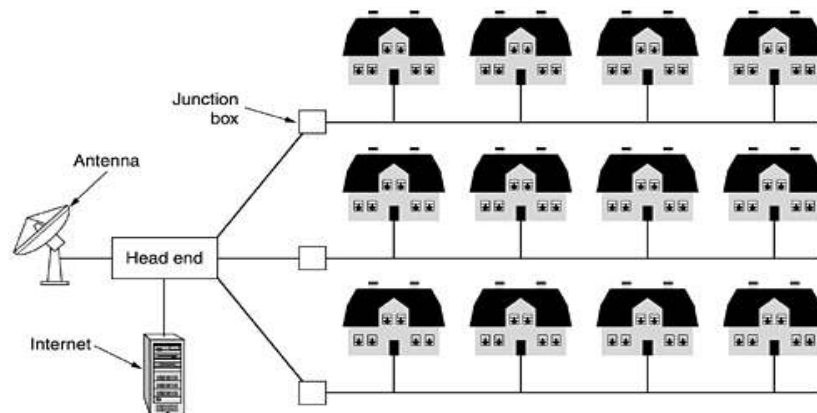
- Internet
- **LAN: local area network**
 - Local area networks, generally called LANs, are privately-owned networks within a single building or campus of up to a few kilometers in size.
 - LANs may use a transmission technology consisting of a cable to which all the machines are attached.
 - Traditional LANs run at speeds of 10 Mbps to 1000 Mbps, have low delay (microseconds or nanoseconds), and make very few errors.
 - Various topologies are possible for broadcast LANs.



(a) Bus Topology

(b) Ring Topology

- **MAN: metropolitan area network**



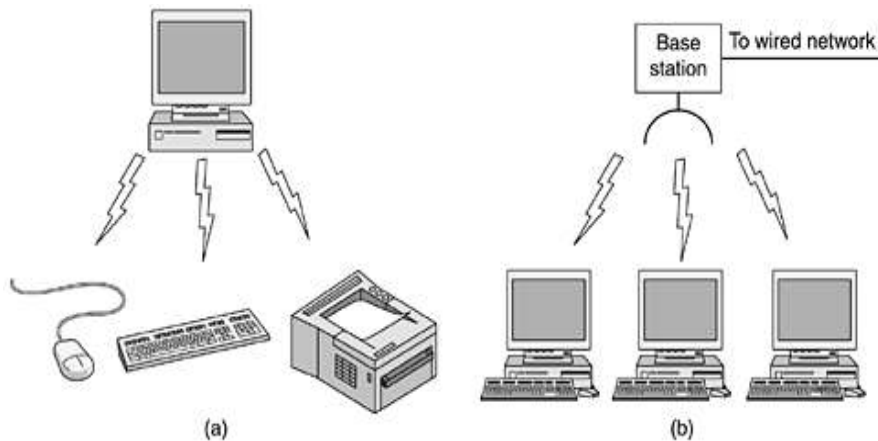
- **WAN and Internet**
 - Hosts, The hosts are owned by the customers (e.g., people's personal computers).
 - Subnet typically owned and operated by a telephone company or Internet service provider (ISP).
 - The hosts are connected by subnets.

- The subnet consists of two distinct components: transmission lines and switching elements. Transmission lines move bits between machines. Switching elements (routers) are specialized computers that connect three or more transmission lines.

Packets routing

- **Wireless Networks**

- Bluetooth: short range wireless network (<10m).
- Wireless LANs: for moderate range, becomes more and more common today (<100m).
- Network used for cellular telephones: distances involved are much greater and the bit rates much lower (above 1km).



Network Software

- Protocol Hierarchies
- Design Issues for the Layers
- Connection-Oriented and Connectionless Services
- Service Primitives

- The Relationship of Services to Protocols

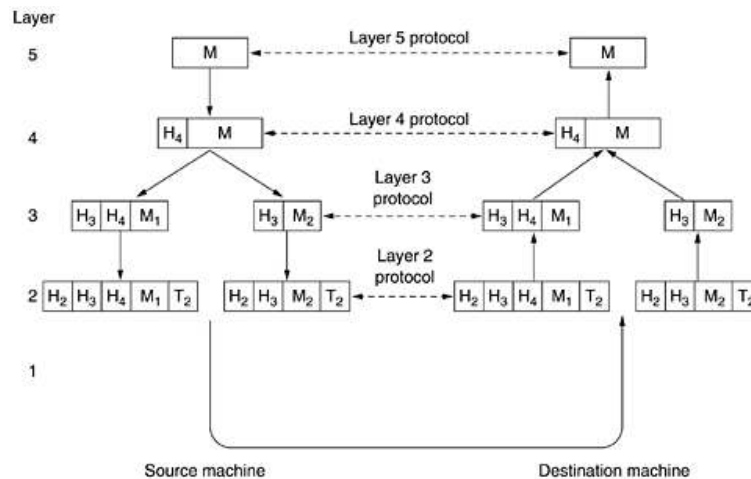
Protocol Hierarchies

- To reduce their design complexity, most networks are organized as a **stack of layers** or **levels**, each one built upon the one below it.
- The purpose of each layer is to offer certain services to the higher layers, shielding those layers from the details of how the offered services are actually implemented.
- A **protocol** is an agreement between the communicating parties on how communication is to proceed.

Physical Communication

Layers, protocols, and interfaces

- Interface: Between each pair of adjacent layers is an interface.
- Network architecture: A set of layers and protocols is called network architecture.
- Protocol stack: A list of protocols used by a certain system, one protocol per layer, is called a protocol stack.
- An example network protocol stack



Example information flow supporting virtual communication in layer 5

- Message is generated by the application of the source machine.
- Message will be sent from the source to the destination.

- Message “M” is transferred from layer 5 to layer 4, with a header containing control information, such as sequence numbers, which helps layer 4 maintain the message order.
- Layer 3 break the message from layer 4 into two pieces to fit the transmission restrictions, while adding another header to tell layer 2 where the dest. is.
- Layer 2 adds the messages from layer 3 with another header, telling the actual (physical) address of the dest, and a trailer, which is the checksum of the message for correction assertion.
- At the receiving machine the message moves upward, from layer to layer, with headers being stripped off as it progresses. Message is sent to the dest. machine.

Design Issues for the Layers

- Every layer needs a mechanism for identifying senders and receivers. (who to talk with)
- The protocol must also determine how many logical channels the connection corresponds to and what their priorities are. (simplex or duplex? single or multiple channel?)
- Error control is an important issue because physical communication circuits are not perfect.
- Message ordering is important ‘cause Not all communication channels preserve the order of messages sent on them.
- An issue that occurs at every level is how to keep a fast sender from swamping a slow receiver with data.
- Inability of all processes to accept arbitrarily long messages. (fragmentation and reassembling the messages)

Connection-Oriented and Connectionless Services

- Connection-Oriented Service: the service user first establishes a connection, uses the connection, and then releases the connection. (e.g., the telephone, tube)
- Connectionless Service: Each message carries the full destination address, and each one is routed through the system independent of all the others. (e.g., the postal system)
Usually, connectionless service cannot guarantee the order of messages.
- In order to enhance the reliability of transmission of connection-oriented service, acknowledge each received message is helpful. For example, the file transfers.
- However, some applications prefer fast speed than the reliability. For example, the digitized voice traffic, video conference.

Six different types of service

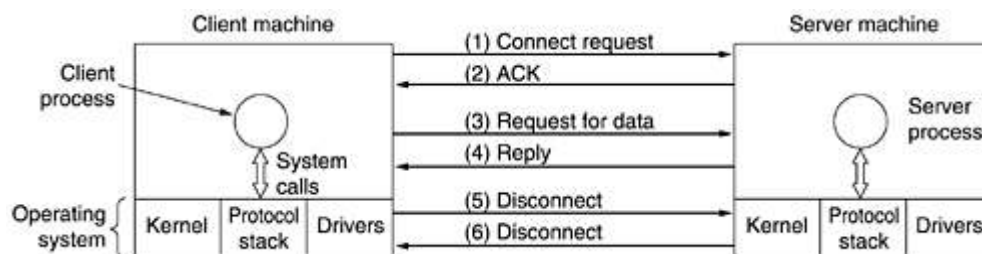
Connection-oriented	Service	Example
	Reliable message stream	Sequence of pages
	Reliable byte stream	Remote login
Connection-less	Unreliable connection	Digitized voice
	Unreliable datagram	Electronic junk mail
	Acknowledged datagram	Registered mail
	Request-reply	Database query

Both reliable and unreliable connection-oriented and connectionless communication coexist

Service Primitives

Primitive	Meaning
LISTEN	Block waiting for an incoming connection
CONNECT	Establish a connection with a waiting peer
RECEIVE	Block waiting for an incoming message
SEND	Send a message to the peer
DISCONNECT	Terminate a connection

Common Primitives



Packets sent in a simple client-server interaction on a connection-oriented network

Services to Protocols Relationship

- The service defines what operations the layer is prepared to perform on behalf of its users
- A service is a set of primitives that a layer provides to the layer above it.
- A protocol is a set of rules governing the format and meaning of the packets which are exchanged by the peer entities in the same layer.
- Services related to the interfaces between layers;

- Protocols related to the packets sent between peer entities on different machine.

Service Providers

The relationship between a service and a protocol

Reference Models

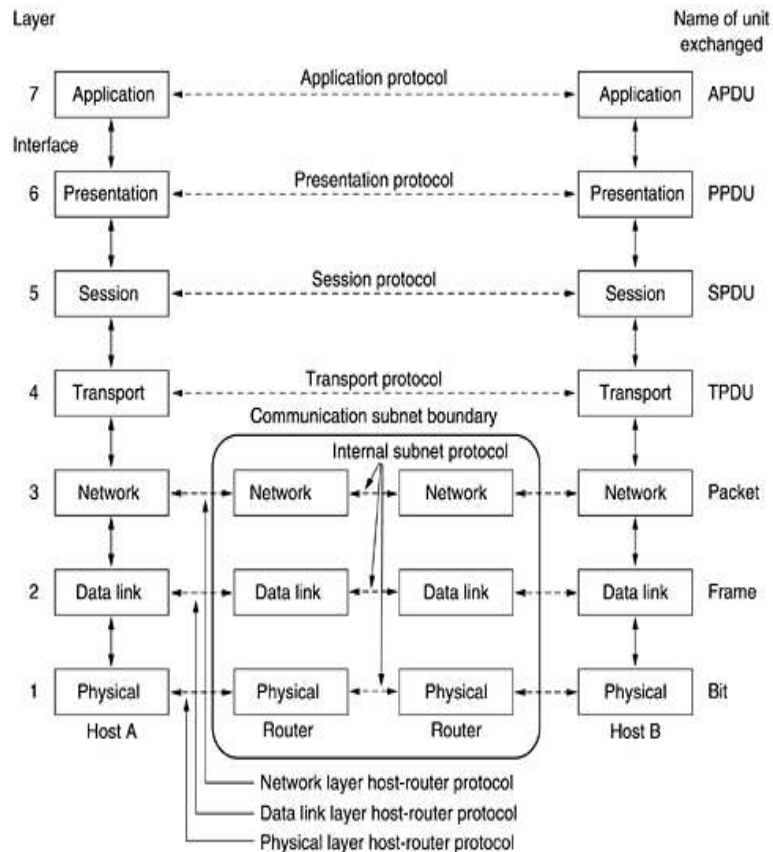
OSI reference model

- developed by the International Standards Organization (ISO)
- OSI means Open Systems Interconnection
- It is rarely used today, while it is actually quite general and still valid, and the features discussed at each layer are still very important.
- OSI model itself is not network architecture because it does not specify the exact services and protocols to be used in each layer.

TCP/IP reference model

- developed by ARPANET
- Used to construct the Internet today
- However, the model itself is not of much use

OSI reference model



OSI model

- **The Physical Layer**

- The physical layer is concerned with transmitting raw bits over a communication channel.
- Typical questions here are how many volts should be used to represent a 1 and how many for a 0, how many nanoseconds a bit lasts, whether transmission may proceed simultaneously in both directions, how the initial connection is established and how it is torn down when both sides are finished, and how many pins the network connector has and what each pin is used for.

- **The Data Link Layer**

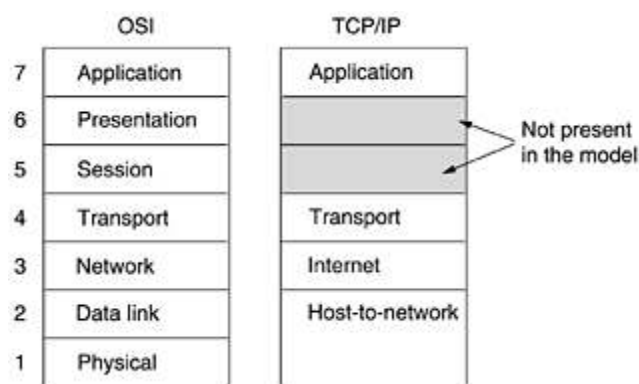
- The main task of the data link layer is to transform a raw transmission facility into a line that appears free of undetected transmission errors to the network layer.
- having the sender break up the input data into data frames and transmit the frames sequentially

- **The Network Layer**

- The network layer controls the operation of the subnet (routing).

- Congestion control, QOS (quality of service)
- **The Transport Layer**
 - The basic function of the transport layer is to accept data from above, split it up into smaller units if need be, pass these to the network layer, and ensure that the pieces all arrive correctly at the other end.
 - The transport layer is a true end-to-end layer, all the way from the source to the destination.
 - layers 1 through 3 are chained, and layers 4 through 7 are end-to-end,
- **The Session Layer**
 - The session layer allows users on different machines to establish sessions between them. Sessions offer various services, including dialog control, token management, and synchronization.
- **The Presentation Layer**
 - Concerned with the syntax and semantics of the information transmitted, to make it possible for computers with different data representations to communicate.
- **The Application Layer**
 - The application layer contains a variety of protocols that are commonly needed by users.

The TCP/IP Reference Model



The TCP/IP reference model

- **The Internet Layer**
 - Its job is to permit hosts to inject packets into any network and have them travel independently to the destination (potentially on a different network). They may even arrive in a different order than they were sent, in which case it is the job of higher layers to rearrange them, if in-order delivery is desired.

- The internet layer defines an official packet format and protocol called IP (Internet Protocol). The job of the internet layer is to deliver IP packets where they are supposed to go.
- The underlining Host-to-Network Layer is a great void. host connected to this layer can send IP packets.
- **The Transport Layer**
 - It is designed to allow peer entities on the source and destination hosts to carry on a conversation
 - Two end-to-end transport protocols have been defined here. The first one, TCP (Transmission Control Protocol), is a reliable connection-oriented protocol that allows a byte stream originating on one machine to be delivered without error on any other machine in the internet.
 - The second protocol in this layer, UDP (User Datagram Protocol), is an unreliable, connectionless protocol for applications that do not want TCP's sequencing or flow control and wish to provide their own.
- **The Application Layer**
 - The TCP/IP model does not have session or presentation layers, which are of little use to most applications.
 - It contains all the higher-level protocols. The early ones included virtual terminal (TELNET), file transfer (FTP), and electronic mail (SMTP). Many other protocols have been added to these over the years: the Domain Name System (DNS) for mapping host names onto their network addresses, NNTP, the protocol for moving USENET news articles around, and HTTP, the protocol for fetching pages on the World Wide Web, and many others.

Comparison of the OSI and TCP/IP Reference Models

- **Similarities**
 - Both are based on the concept of a stack of independent protocols.
 - Both of them have transport layer to provide an end-to-end, network-independent transport service
- **Differences**
 - In OSI, a layer's service definition tells what the layer does, and a layer's interface tells the processes above it how to access it, without explaining how the layers work inside.

- The TCP/IP model did not clearly distinguish between services, interface of the layers.
- As a consequence, the protocols in the OSI model are better hidden than in the TCP/IP model and can be replaced relatively easily as the technology changes.
- OSI model strives to describe the general model of network protocols, while TCP/IP model is specific as it is defined after the real implementations. So TCP/IP model is not useful to describe other protocols.

UNIT - II

Transmission Media: Magnetic Media - Twisted Pair -Baseband Coaxial Cable - Broadband Coaxial Cable - Fiber Optics. Wireless Transmission - the Electromagnetic Spectrum - Radio Transmission - Microwave Transmission - Infrared and Millimeter Waves -Lightwave Transmission.

Narrow Band ISDN System Architecture. Broadband ISDN and ATM Networks - Cellular Radio - Paging Systems - Cordless Telephones - Analog Cellular Telephones - Digital Cellular Telephones - Personal Communication Services.

Notes: <https://inspirit.net.in/books/networking/Introduction%20to%20Computer%20Networks.pdf>

<https://fmfi-uk.hq.sk/Informatika/Distribuvane%20Systemy/knihy/ICN/index.htm>

Transmission Media

The purpose of physical media is to transport a raw bit stream from one machine to another. These media are roughly grouped into guided media, such as copper wire and fiber optics, and unguided

media, such as radio and lasers through the air. Each one has its own characteristic in terms of bandwidth, delay, cost, and ease of installation and maintenance.

Guided Media:

Medium is more important in setting transmission parameters. Guided media includes everything that 'guides' the transmission. That usually takes the form of copper wires, optical fiber etc.

Magnetic Media:

The most useful ways to transport data from one computer to another is magnetic tape or floppy disk while this method is not as sophisticated as using a geosynchronous area. It is more cost effective. The bandwidth characteristics of magnetic tape are excellent, the delay characteristics are poor. Transmission time is measured in minutes or hours, not in milliseconds.

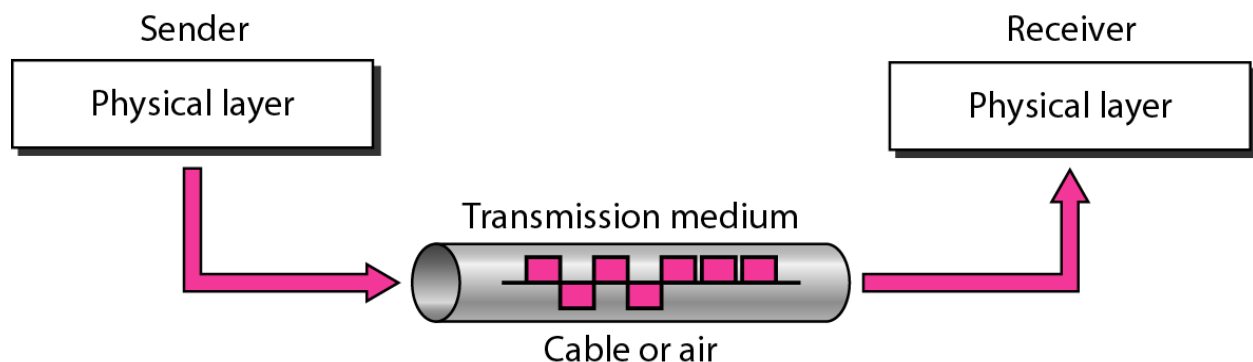
- Floppy disk
- CDs
- Magnetic tapes
- Advantage

High bandwidth (bulk of data)

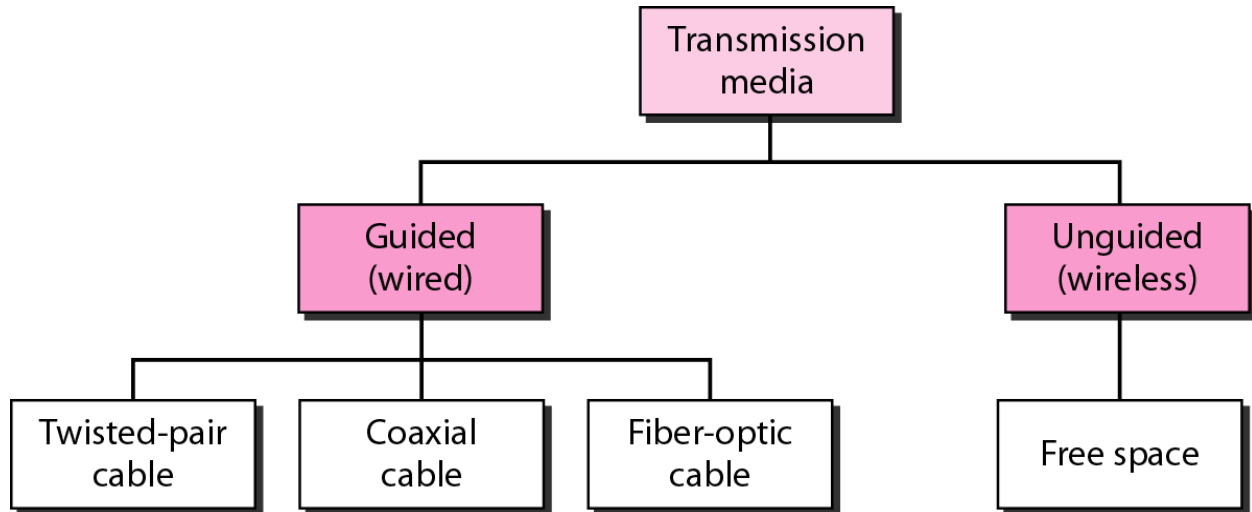
- Disadvantage

Delay is too long

All computer communication involves encoding data in a form of energy, and sending the energy across a transmission medium (i.e., the network links).



Transmission medium and physical layer



Classes of transmission media

Guided Transmission Media uses a "cabling" system that guides the data signals along a specific path. The data signals are bound by the "cabling" system. Guided Media is also known as Bound Media. Cabling is meant in a generic sense in the previous sentences and is not meant to be interpreted as copper wire cabling only.

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

There are three basic types of Guided Media:

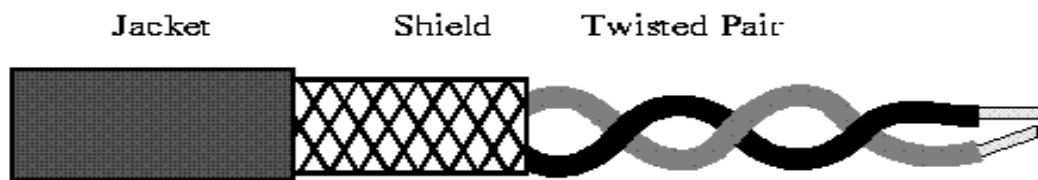
- Twisted pair
- Coaxial cable
- Optical fiber

Twisted pair

- The wires in Twisted Pair cabling are twisted together in pairs.



- Each pair would consist of a wire used for the positive data signal and a wire used for the negative data signal.
- Any noise that appears on one wire of the pair would occur on the other wire
- The noise appears on both wires; it cancels itself.
- Cables with a shield are called Shielded Twisted Pair and commonly abbreviated STP. Cables without a shield are called Unshielded Twisted Pair or UTP. Twisting the wires together results in characteristic impedance for the cable. Typical impedance for UTP is 100 ohm for Ethernet 10BaseT cable.



There are two types of twisted pair cables:

- Shielded Twisted Pair (STP) Cable
- Unshielded Twisted Pair (UTP) Cable

STP cables come with twisted wire pair covered in metal foil. This makes it more indifferent to noise and crosstalk.

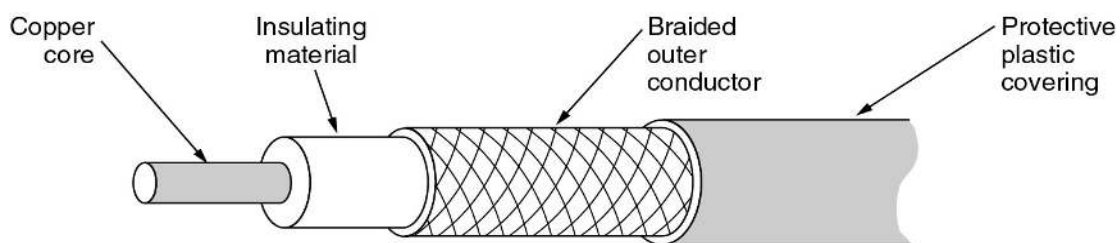
UTP has seven categories, each suitable for specific use. In computer networks, Cat-5, Cat-5e, and Cat-6 cables are mostly used. UTP cables are connected by RJ45 connectors.

Coaxial cable



- Coaxial Cable consists of two conductors. The inner conductor is held inside an insulator with the other conductor woven around it providing a shield.
- An insulating protective coating called a jacket covers the outer conductor. The outer shield protects the inner conductor from outside electrical signals.

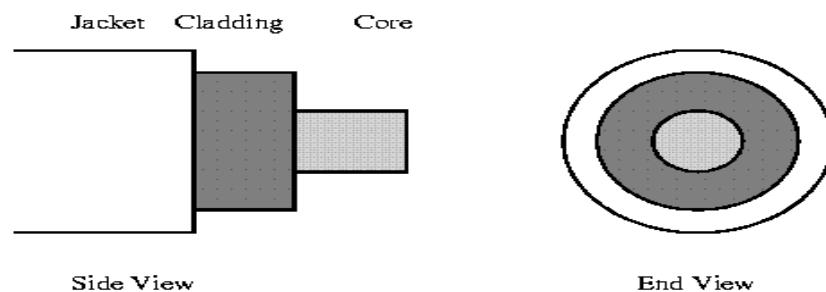
- The distance between the outer conductor (shield) and inner conductor plus the insulating material determine the cable properties or impedance.
- Typical impedances for coaxial cables are 75 ohms for Cable TV, 50 ohms for Ethernet Thinnet and Thicknet.
- A **baseband coaxial cable** transmits a single signal at a time at very high speed. A **baseband cable** is mainly used for LANs. **Baseband coaxial cable** supports frequency range of a-4kHz and are used for digital signaling.
- A **broadband coaxial cable** can transmit many simultaneous signals using different frequencies.

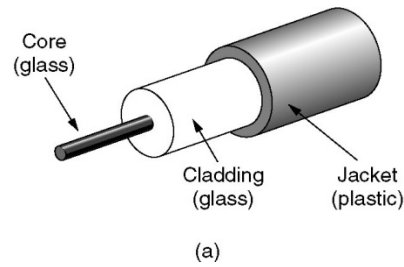


A coaxial cable

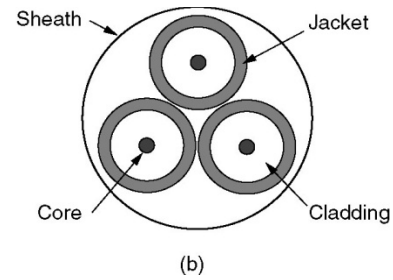
Fiber Optics

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





(a) Side view of a single fiber



(b) End view of a sheath with three fibers

Advantages of Optical Fiber:

- Noise immunity: RFI and EMI immune (RFI - Radio Frequency Interference, EMI -Electromagnetic Interference)
- Security: cannot tap into cable.
- Large Capacity due to BW (bandwidth)
- Longer distances than copper wire
- Smaller and lighter than copper wire
- Faster transmission rate

Disadvantages of Optical Fiber:

- Physical vibration will show up as signal noise
- Limited physical arc of cable. Bend it too much & it will break!
- Difficult to splice
- At higher transmission capacity, it is cheaper than copper. At lower transmission capacity, it is more expensive.

Media versus Bandwidth

- The following table shows the bandwidths for the different guided transmission media

Cable type	Bandwidth
Twisted Pair	100 MHz
Coaxial Cable	600 MHz
Optical Fiber	1 GHz

Wireless Transmission

- The Electromagnetic Spectrum
- Radio Transmission
- Microwave Transmission
- Infrared and Millimeter Waves
- Light-wave Transmission

Wireless transmission is a form of unguided media. Wireless communication involves no physical link established between two or more devices, communicating wirelessly. Wireless signals are spread over in the air and are received and interpreted by appropriate antennas.

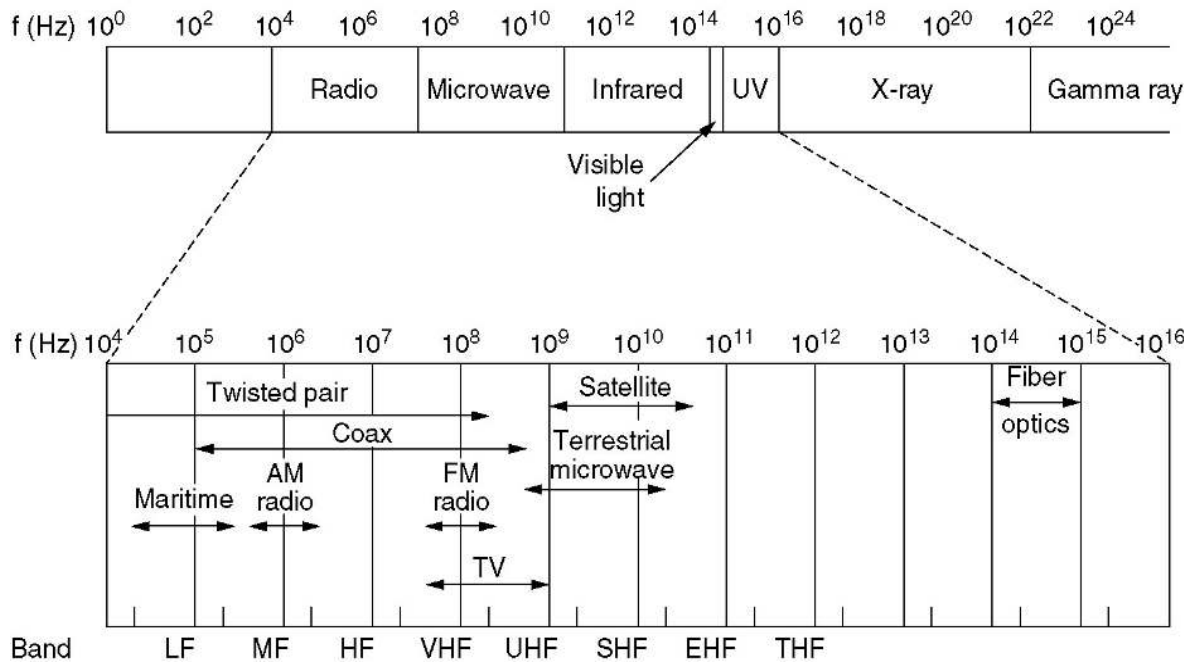
When an antenna is attached to electrical circuit of a computer or wireless device, it converts the digital data into wireless signals and spread all over within its frequency range. The receptor on the other end receives these signals and converts them back to digital data.

A little part of electromagnetic spectrum can be used for wireless transmission.



The Electromagnetic Spectrum

The **electromagnetic spectrum** is a continuum of all **electromagnetic** waves arranged according to frequency and wavelength. The sun, earth, and other bodies radiate **electromagnetic** energy of varying wavelengths. **Electromagnetic** energy passes through space at the speed of light in the form of sinusoidal waves.



The electromagnetic spectrum and its uses for communication.

Radio Transmission

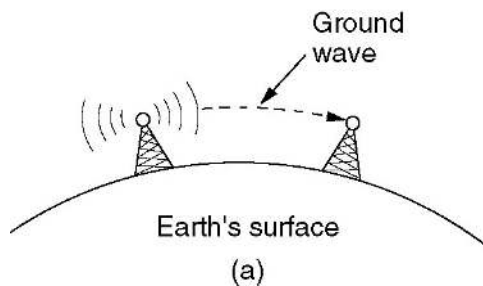
Are easy to generate, can travel long distances and can penetrate buildings easily. Are omnidirectional, I.e. they travel in all directions

- Adv : transmitter and receiver do not have to be aligned
- Disadv : interference of signals : less secure : govt license required to use particular frequency band

The properties of RW are frequency dependent

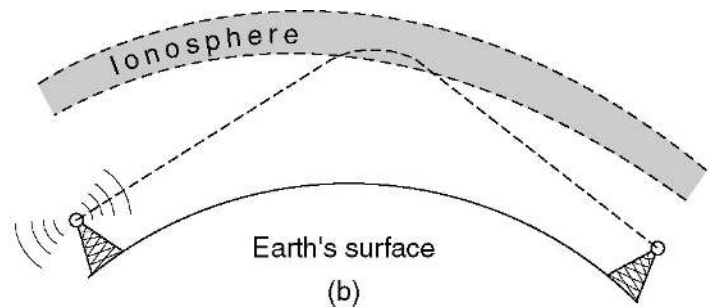
- At low frequency: pass thru obstacles well but the power falls off sharply with distance from the source.

- At high frequency : they travel in straight lines , bounce off obstacles, and absorbed by rain



(a) In the VLF, LF, and MF bands, radio

waves follow the curvature of the earth

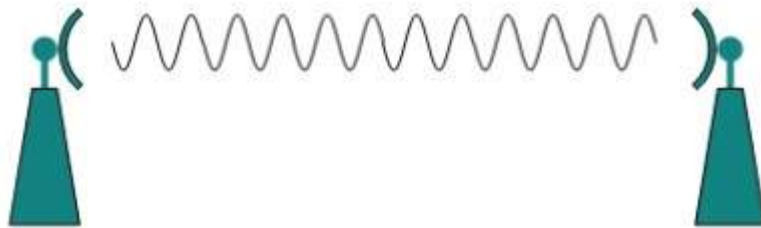


b) In the HF band, they bounce off the ionosphere

Microwave Transmission

Electromagnetic waves above 100 MHz tend to travel in a straight line and signals over them can be sent by beaming those waves towards one particular station. Because Microwaves travels in straight lines, both sender and receiver must be aligned to be strictly in line-of-sight.

Microwaves can have wavelength ranging from 1 mm – 1 meter and frequency ranging from 300 MHz to 300 GHz.



Microwave antennas concentrate the waves making a beam of it. As shown in picture above, multiple antennas can be aligned to reach farther. Microwaves have higher frequencies and do not penetrate wall like obstacles.

Microwave transmission depends highly upon the weather conditions and the frequency it is using.

Application of Microwave Transmission

- Short range Networking
- Example : WLL : Wireless Local Loop

Infrared and Millimeter wave

Infrared wave lies in between visible light spectrum and microwaves. It has wavelength of 700-nm to 1-mm and frequency ranges from 300-GHz to 430-THz. Infrared wave is used for very short range communication purposes such as television and it's remote. Infrared travels in a straight line hence it is directional by nature. Because of high frequency range, Infrared cannot cross wall-like obstacles.

- For short range
- Directional
- Do not pass thru solid objects

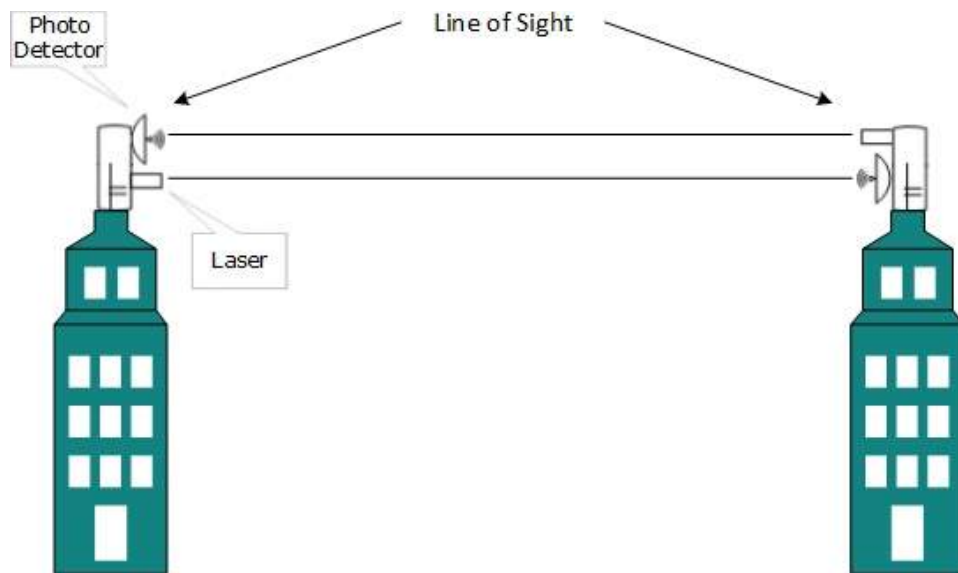
Applications of Infrared

- Applications of Infrared
 - Remote control Home- appliances
- Applications of Millimeter
 - Wireless Local Loop

Light Transmission

Highest most electromagnetic spectrum which can be used for data transmission is light or optical signaling. This is achieved by means of LASER.

Because of frequency light uses, it tends to travel strictly in straight line. Hence the sender and receiver must be in the line-of-sight. Because laser transmission is unidirectional, at both ends of communication the laser and the photo-detector needs to be installed. Laser beam is generally 1mm wide hence it is a work of precision to align two far receptors each pointing to lasers source.



Laser works as Tx (transmitter) and photo-detectors works as Rx (receiver).

Lasers cannot penetrate obstacles such as walls, rain, and thick fog. Additionally, laser beam is distorted by wind, atmosphere temperature, or variation in temperature in the path.

Laser is safe for data transmission as it is very difficult to tap 1mm wide laser without interrupting the communication channel.

Narrow Band ISDN System Architecture

ISDN is a network concept providing a integration of data, voice and video. Its based on 64Kbps digital Communication channel. ISDN is a generic term for any network which connects homes and business together with a service companies such as bank, air-lines, stock market etc using a digital network. From 1990 ISDN became a well convinced and planned area for development in the field of telecommunication. Telephone network requires two basic functions:

- (i) Signaling –To establish and realize a call.
- (ii) End-to-end transmission – To transfer the information between the users

ISDN uses these functions as separate and indeed provide different channels for these services. This is done through proper interface. The purpose is to provide access to various services that are possibly supported by different networks. The base rate interface(BRI) provides the users with two 64Kbps barrier(B) channels and one 16Kbps data(D) channels. The primary data rate interface(PRI) provides users with(23B+1D) channels in North America and Japan and (30B+1D) channels in Europe. Each B

channel is bidirectional and provides 64Kbps end –to-end digital connection that can carry PCM voice or data. The primary function of D channel is to carry information for B channels.

Types of ISDN are:

- (i) Narrow band ISDN
- (ii) Broad band ISDN

Narrow band has a smaller bandwidth and supports data rate upto 64Kbps only. The internet in HD(TV) and high speed data interconnected promoted the work on B-ISDN. Due to low bit rates the quality of service provided by the narrowband ISDN is poor. That means the quality of video Signals in the narrow band ISDN is poor. The quality of service is improved by using **broadband ISDN**. **The broadband ISDN** can support higher data rate due to use of optical fibre cables. The bandwidth of an optical fibre cable is very high. So the broadband ISDN can allow transmission of very high quality video images through it.

Merits:

- An ISDN user can establish two simultaneously independent telecom calls on existing pair of telephone wires.
- Two simultaneously calls may be of any type such as speech, data, image, video.

Narrowband ISDN. N-ISDN (**Narrowband Integrated Switch digital Network**) was an attempt to replace the analog telephone system with a digital one. ... It describes telecommunication that carries voice information in a **narrow band** of frequencies.

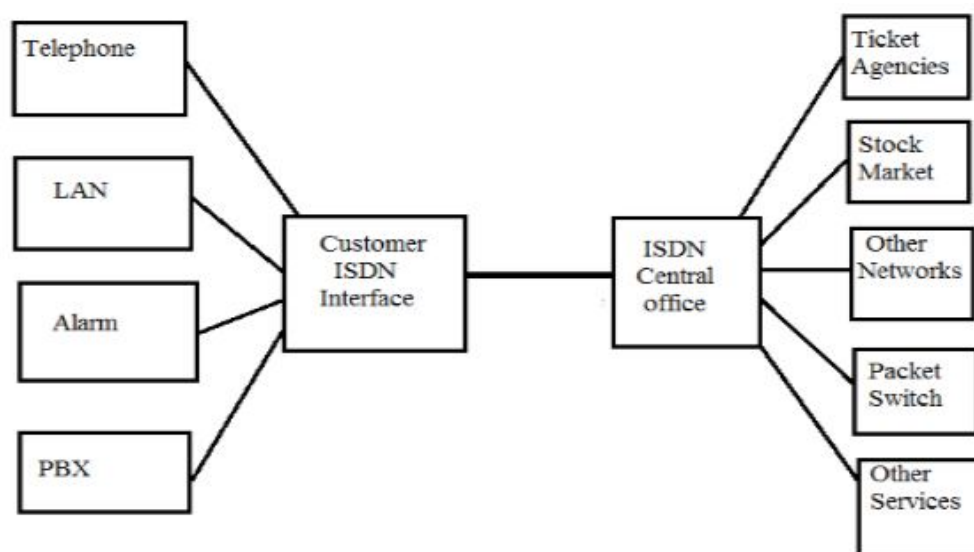


Fig no.2: Architecture of ISDN

Broadband ISDN

- B-ISDN is an all-purpose digital network that will support a wide variety of applications and services in a flexible and cost-effective manner
- Successor to Narrowband ISDN (called N-ISDN or ISDN)
- B-ISDN assumes transmission rates of 155 Mbps and up

B-ISDN Capabilities

- Interactive and distributive services
- Broadband and narrowband rates
- Bursty and continuous traffic
- Connection-oriented and connectionless
- Point-to-point or multipoint connections
- Delay-sensitive or delay-insensitive traffic

High capacity and high performance fiber-based transmission at physical layer

Flexible switching and multiplexing (i.e., “transfer modes”)

Transfer mode determines the rules for dividing up the usable capacity (bandwidth) of the network and allocating it to users (channels)

Choice: Asynchronous Transfer Mode (ATM)

Need to accommodate a wide variety of services, with diverse traffic mixes

Broadband ISDN

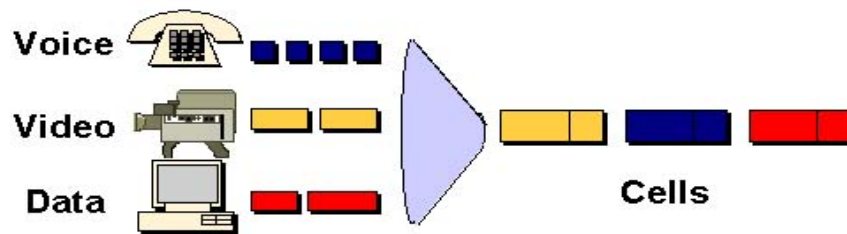
Simplified Protocol Stack

ATM Networks

Asynchronous Transfer Mode (ATM)

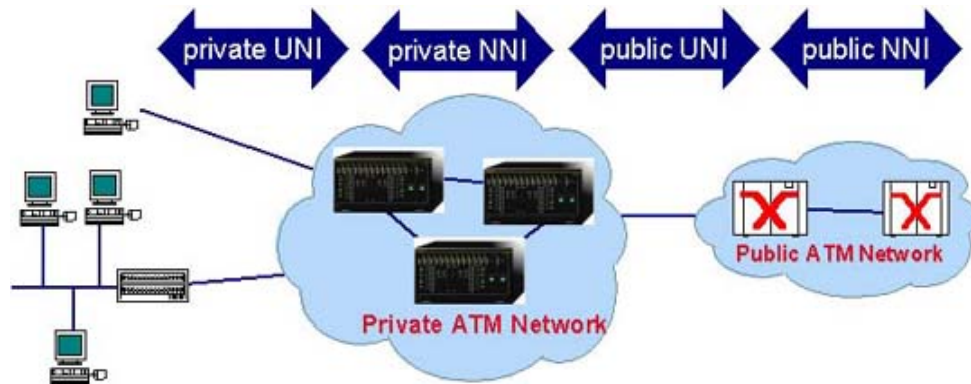
- Cell relay network protocol which encodes data traffic into small fixed sized (53 byte; 48 bytes of data and 5 bytes of header information) cells instead of variable sized *packets* as in packet-switched networks (such as the Internet Protocol or Ethernet).

Characteristics of ATM



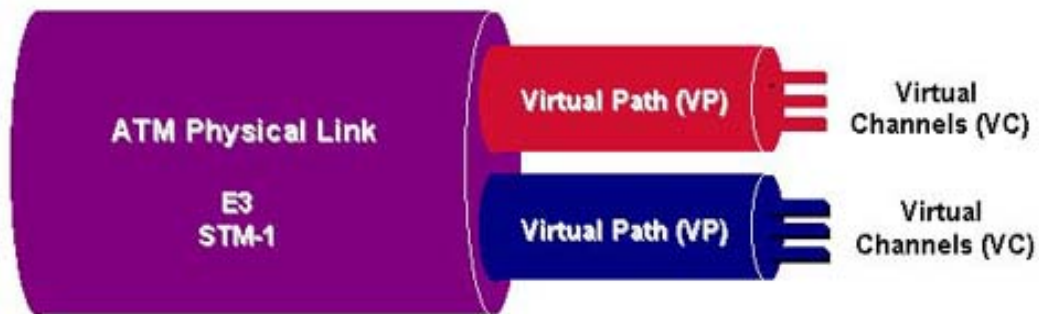
- Uses small, fixed-sized cell
- Connection oriented
- Bandwidth allocation on demand
- Supports multiple service types
- Applicable to LAN and WAN

ATM Signaling



- UNI : User to Network Interface
- NNI : Network to Network Interface
- Different protocol in the different domain

Virtual Path / Virtual Channel



- Connection Identifier : VPI/VCI
- Physical Link : Contains multiple VPs
- Virtual Path : Contains multiple VCs
- Virtual Channel : Logical path between ATM endpoint

Narrow Band ISDN System Architecture

Anticipating user demand for end-to-end digital services the world's telephone companies agreed in 1984 under the auspices of CCI TT to build a new, fully digital, circuit-switched telephone system by the early part of the 21st century. This system was called ISDN (Integrated Services Digital Network) and its primary goal was to integrate the voice and nonvoice services. It is already available in many locations and its use is growing slowly.

ISDN Services

The key ISDN service will continue to be voice but with many enhanced features. Some of them are:

- buttons for instant call setup to arbitrary telephones anywhere in the world,
- displaying the caller's telephone number, name and address while ringing,
- connecting the telephone to a computer enabling the caller's database record to be displayed on the screen as the call comes in,
- call forwarding,
- conference calls worldwide,
- on line medical, burglar, and smoke alarms giving the address to speed up response

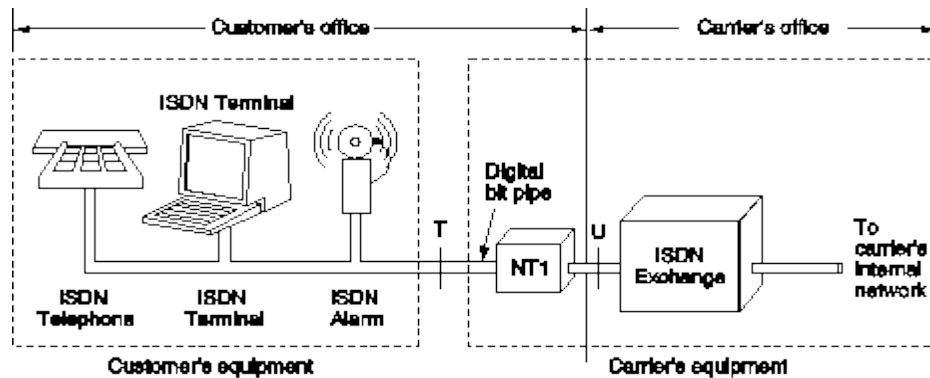
ISDN System Architecture

The key idea behind ISDN is that of the digital bit pipe between the customer and the carrier through which bits flow in both directions. Whether the bits originate from a digital telephone, a digital terminal, a digital facsimile machine, or some other device is irrelevant.

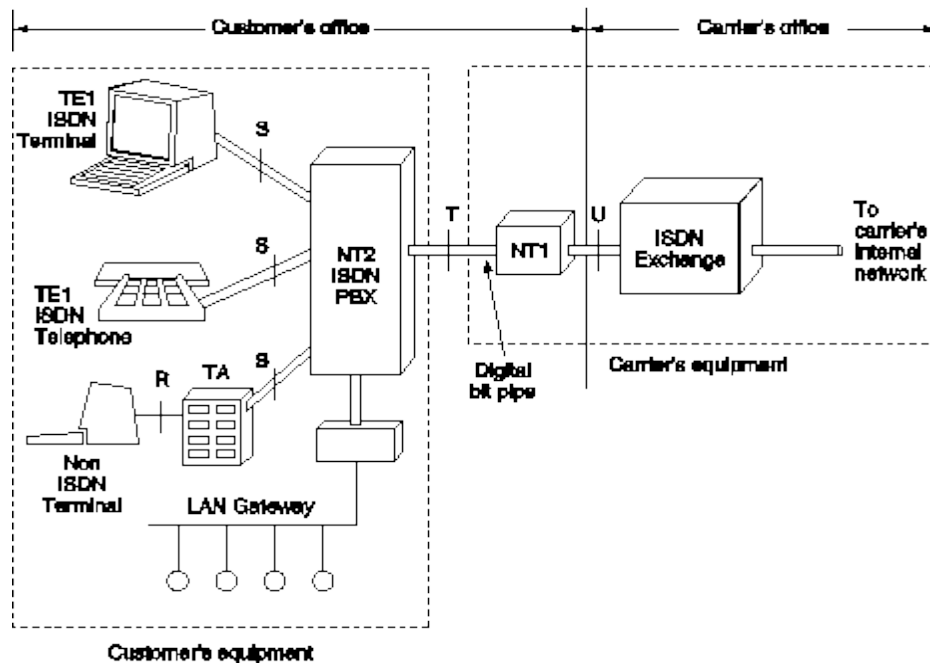
The digital bit pipe can support multiple independent channels by time division multiplexing of the bit stream. Two principal standards for the bit pipe have been developed:

- a low bandwidth standard for home use, and
- a higher bandwidth standard for business use that supports multiple channels identical to the home use channels.

Normal configuration for a home consists of a network terminating device NT1 (Fig.(a)) placed on the customer's premises and connected to the ISDN exchange in the carrier's office using the twisted pair previously used to connect the telephone. The NT1 box has a connector into which a bus cable can be inserted. Up to 8 ISDN telephones, terminals, alarms, and other devices can be connected to the cable. From the customer's point of view, the network boundary is the connector on NT1.



(a)



(b)

(a) Example ISDN system for home use

(b) Example ISDN system with a PBX for use in large businesses

For large businesses, the model of Fig. 2-41(b) is used. There is a device NT2 called PBX (Private Branch eXchange - conceptually the same as an ISDN switch) there connected to NT1 and providing the interface for ISDN devices.

CCITT defined four reference points (Fig):

- U reference point = connection between the ISDN exchange and NT1,
- T reference point = connector on NT1 to the customer,
- S reference point = interface between the ISDN PBX and the ISDN terminal,
- R reference point = the connection between the terminal adapter and non-ISDN terminal.

The ISDN Interface

The ISDN bit pipe supports multiple channels interleaved by time division multiplexing. Several channel types have been standardized:

- A - 4 kHz analog telephone channel
- B - 64 kbps digital PCM channel for voice or data
- C - 8 or 16 kbps digital channel for out-of-band signaling
- D - 16 kbps digital channel for out-of-band signaling
- E - 64 kbps digital channel for internal ISDN signaling
- H - 384, 1536, or 1920 kbps digital channel.

Perspective on N-ISDN

N-ISDN was an attempt to replace the analog telephone system with a digital one. Unfortunately, the standardization process was too long and regarding to the technology progress in this area, once the standard was finally agreed, it was obsolete.

N-ISDN basic rate is too low so for home as for business today. N-ISDN may be partly saved, but by an unexpected application: Internet access. Various companies now sell ISDN adapters that combine the 2B + D channels into a single 144 kbps digital channel. Many Internet providers also support these adapters. So the people can access Internet over a 144 kbps digital link, instead of a 28.8 kbps analog modem link and for affordable price that may be a niche for N-ISDN for the next few years.

Broadband I SDN and ATM

When CCI TT found that the N-I SDN was not going to solve the actual communication problems, it tried to think of a new service. The result was broadband I SDN (B-I SDN), basically a digital virtual circuit for moving fixed-sized packets (cells) at 155 Mbps. Broadband I SDN is based on ATM technology that is fundamentally a packet-switching technology.

Virtual Circuits versus Circuit Switching

The basic broadband I SDN service is a compromise between pure circuit switching and pure packet switching. The actual service offered is connection oriented, but it is implemented internally with packet switching. Two kinds of connections are offered:

- Permanent virtual circuits - ordered by customers at carriers, remain in place for long time.
- Switched virtual circuits - set up dynamically like telephone calls.

The advantage of permanent over a switched virtual circuit is that there is no setup time, packets along permanent circuit can move instantly. For some applications, such as credit card verification, saving a few seconds on each transaction may be worth the cost of the permanent circuit.

In a virtual circuit network, like ATM, when a circuit is established, what really happens is that route is chosen from source to destination, and all the switches (i.e., routers) along the way make table entries so that they can route any packet on that virtual circuit (Fig. 2-43). When a packet comes along, the switch inspects the packet header to find out which virtual circuit it belongs to. Then it looks up that virtual circuit in its tables to determine which communication line to send on.

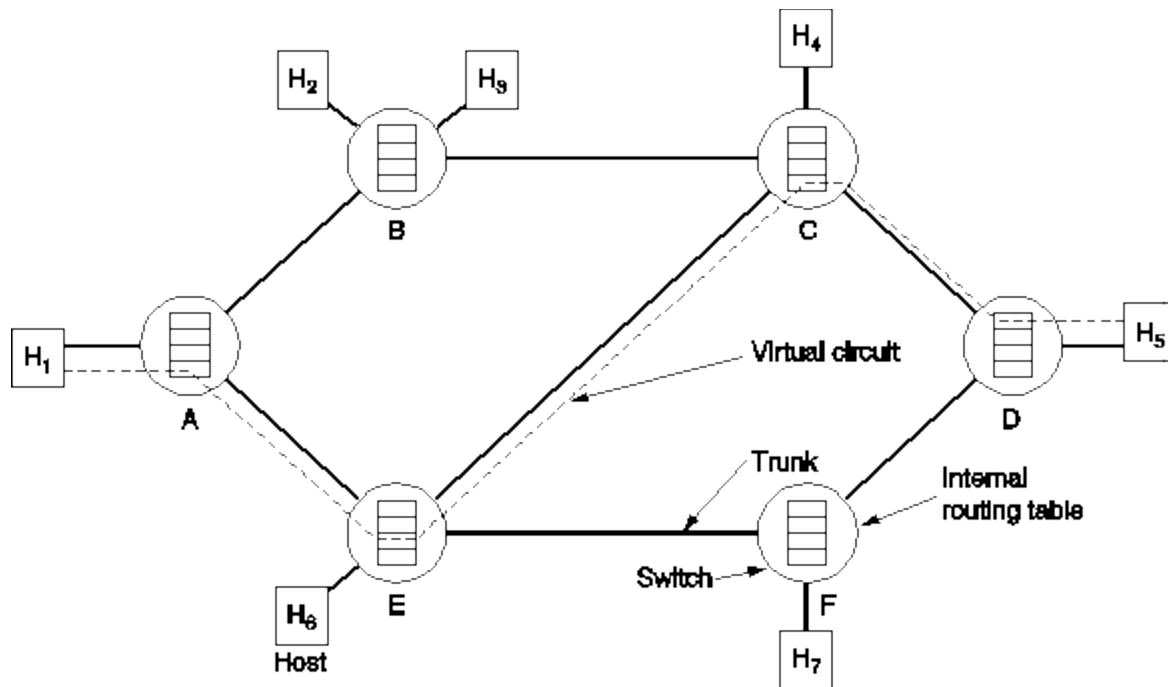


Fig. The dotted line shows a virtual circuit. It is simply defined by table entries inside the switches.

Transmission in ATM Networks

ATM stands for Asynchronous Transfer Mode. This mode can be contrasted with the synchronous T1 carrier (Fig.).

T1: frames are generated precisely every 125(sec. This rate is governed by a master clock. Slot k of each frame contains 1 byte of data from the same source.

ATM: has no requirements that cells rigidly alternate among the various sources. Cells arrive randomly from different sources. The stream of cells need not be continuous. Gaps between the data are filled by special idle cells.

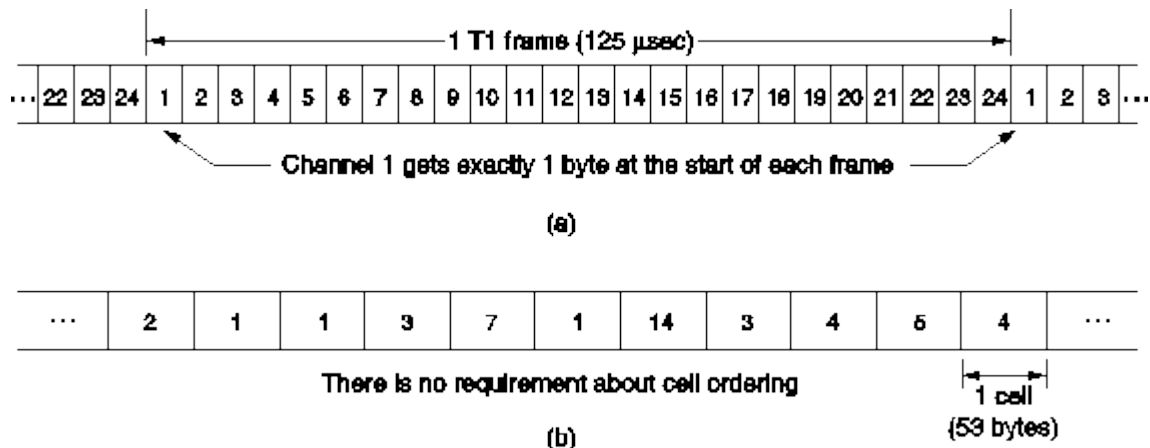


Fig. (a) Synchronous transmission mode. (b) Asynchronous transmission mode.

ATM does not standardize the format for transmitting cells. Cells are allowed to be sent individually, or they can be encased in a carrier such as T1, T3, SONET, or FDDI. For these examples, standards exist telling how cells are packed into the frames these systems provides.

In the original ATM standard, the primary rate was 155.52 Mbps, with an additional rate at four times that speed (622.08 Mbps). These rates were chosen to be compatible with SONET. ATM over T3 (44.736 Mbps) and FDDI (100 Mbps) is also foreseen.

The transmission medium for ATM is normally fiber optics, but for runs under 100 m, coax or category 5 twisted pair are also acceptable. Each link goes between a computer and an ATM switch, or between two ATM switches. So, all ATM links are point-to-point. Each link is unidirectional. For full-duplex operation, two parallel links are needed.

ATM Switches

An ATM cell switch (Fig.) has some number of input lines and some number of output lines (the both numbers are usually the same). ATM switches are generally synchronous in the sense of during a cycle, one cell is taken from each input line, passed into the internal switching fabric, and eventually transmitted on the appropriate output line.

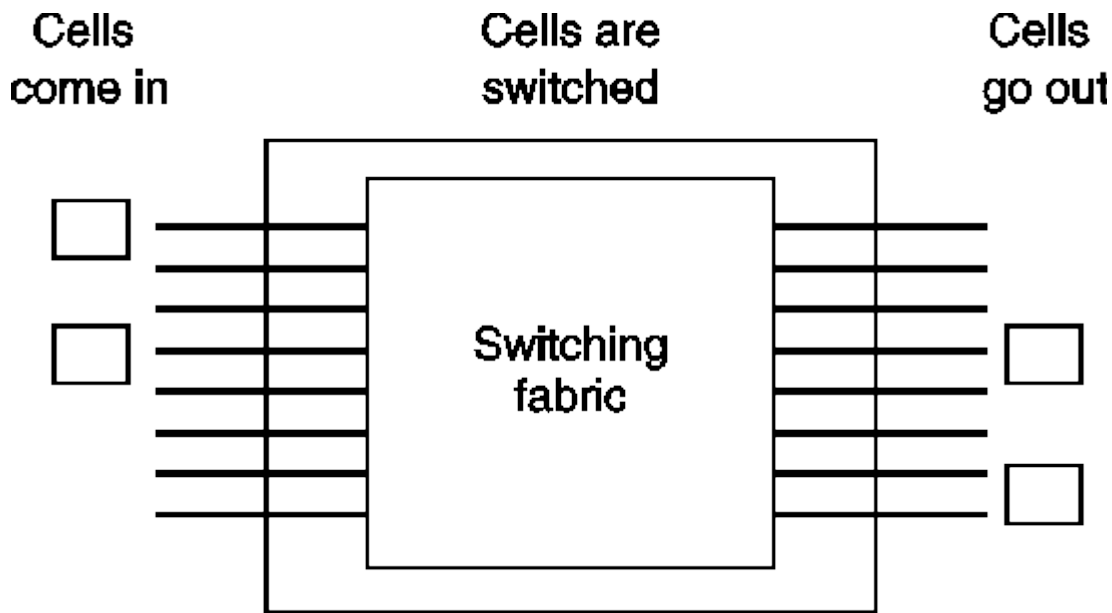


Fig. . A generic ATM switch.

Switches may be pipelined, that is, it may take several cycles before an incoming cell appears on its output line. Cells actually arrive on the input lines asynchronously, so there is a master clock that marks the beginning of a cycle. Any cell fully arrived when the clock ticks is eligible for switching during that cycle. A cell not fully arrived has to wait until the next cycle.

Cells arrive at ATM speed, normally about 150 Mbps. This works out around 360000 cells/sec, the cycle time has to be about 2.7 (sec. A commercial switch might have from 16 to 1024 input lines. At 622 Mbps the cycle time has to be about 700 nsec. The fact that the cells are fixed length and short makes it possible to build such switches. With longer variable-length packets, high speed switching would be more complex, which is why ATM uses short fixed-length cells.

All ATM switches have two common goals:

- Switch all cells with as low a discard rate as possible - cells can be dropped just in emergencies, but the loss rate should be as small as possible.
- Never reorder the cells on a virtual circuit - this constraint makes switch design considerably more difficult, but is required by the ATM standard.

A problem is if the cells arriving at more input lines want to go to the same output port in the same cycle. One solution is to provide a queue for each input line. If more cells conflict, one of them is chosen for delivery, and the rest are held for the next cycle (Fig.). The problem with input queuing is that when a cell

has to be held up, it blocks the progress of all cells behind it, even if they could otherwise be switched (head-of-line blocking).

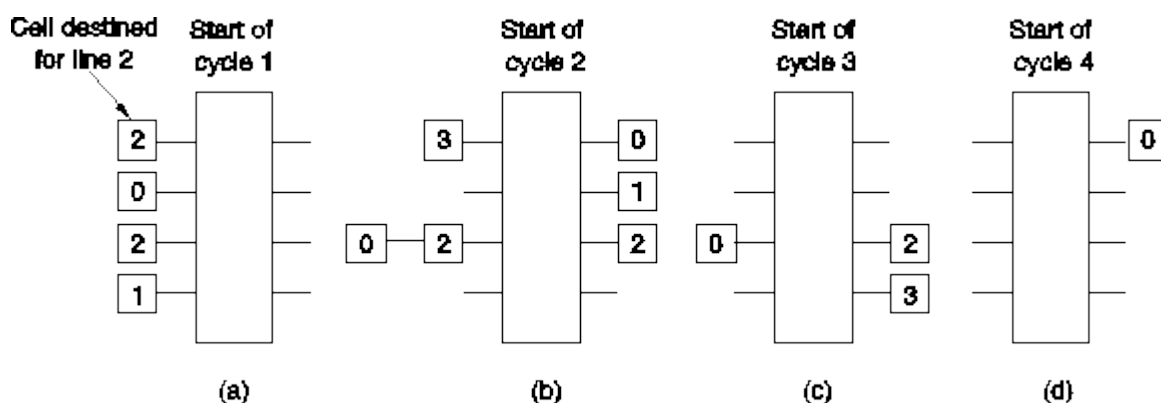


Fig.. Input queueing at an ATM switch.

An alternative design, one that does not exhibit head-of-line blocking, does the queuing on the output side (Fig.). In our example, it takes only three cycles, instead of four in the previous example, to switch all packets. Output queuing is generally more efficient than input queuing.

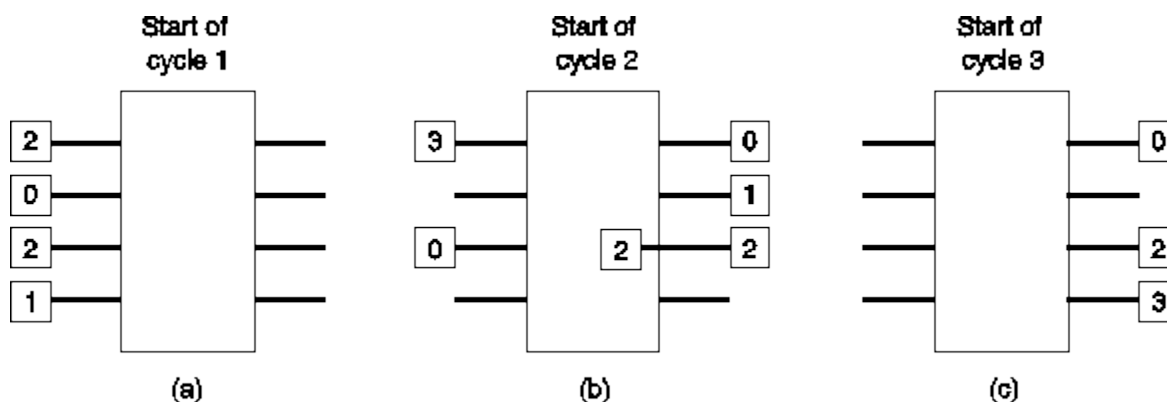


Fig.. Output queueing at an ATM switch.

Cellular Radio

The traditional telephone system (even when broadband ISDN is fully operating) will still not be able to satisfy people on the go. Consequently, there is increasing competition from systems that use wireless technologies for communication. They are already creating a huge market. Many companies in the computer, telephone, satellite, and other industries want a piece of action. The result is a chaotic market, with numerous overlapping and incompatible products and services, all rapidly changing.

Paging Systems

The first paging systems used loudspeakers within a single building. Nowadays, people who want to be paged wear small beepers, usually with tiny screens for displaying short incoming messages.

A person wanting to page a beeper wearer can call the beeper company and enter a security code, the beeper number, and the number the beeper wearer is to call (or another short message). The request is then broadcasted from a hilltop antenna (for local paging) or from a satellite (for long distance paging). When a beeper detects its unique number in the incoming radio stream, it beeps and displays the number to be called.

Most current paging systems are one-way systems, from a single computer out to a large number of receivers. There is no problem about who will speak next, and no contention among many competing users. Paging systems require little bandwidth since each message requires only a single burst of about 30 bytes. At this rate, a 1 Mbps satellite channel can handle over 240000 pages per minute. The older paging systems run in the 150 - 174 MHz band, the modern ones in the 930 - 932 MHz band. (Fig.).

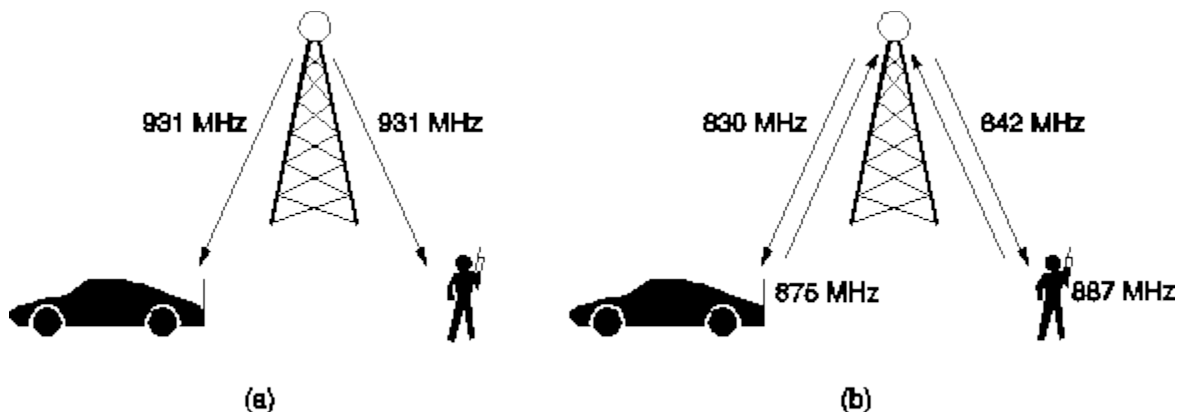


Fig. . (a) Paging systems are one way. (b) Mobile telephones are two way.

Cordless Telephones

A cordless telephone consists of two parts: a base station and a telephone. The base station has a standard phone jack and is connected by a wire to the telephone system. The telephone communicates with the base station by low-power radio. The range is typically 100 - 300 m.

Some of cheaper models of cordless telephones used a fixed frequency, selected at the factory. If, by accident, someone in the neighborhood of a user had the telephone with the same frequency, he could

listen user's calls. More expensive models avoided this problem by allowing the user to select the transmission frequency.

The generations of cordless telephones:

- CT-1 in the US and CEPT-1 in Europe - entirely analog. They could cause interference with radios and television. Poor reception and lack of security.
- CT-2 - digital standard, originated in England. Each telephone had to be within a few hundred meters of its own base station. Useful around the house or office, useless in cars when walking around the town.
- CT3 or DECT - third generation introduced in 1992. This technology is beginning to approach cellular telephones.

Analog Cellular Telephones

Mobile radiotelephones were used sporadically for maritime and military communication during the early decades of the 20th century. Push-to-talk systems (installed in several big cities in the late 1950s) had a single channel used for both sending and receiving. They used a large transmitter on top of a tall building. To talk, the user had to push a button that enabled the transmitter and disabled the receiver. The users could hear each other.

IMTS (Improved Mobile Telephone System - in the 1960s) also used a high-powered (200 watt) transmitter, on top of a hill, but it used different frequencies for sending and for receiving, so no push-to-talk button was necessary. IMTS supported 23 channels spread out from 150 MHz to 450 MHz. Due to the small number of channels, users often had to wait a long time before getting a dial tone. The adjacent systems had to be several hundred km apart. So the system was impractical due to limited capacity.

Digital Cellular Telephones

First generation cellular systems were analog. The second generation is digital. In Europe, an agreement on a common digital system, call *GSM* (Global System for Mobile communication) was achieved.

GSM operates in a new frequency band (1.8 GHz) and uses both FDM and TDM. The available spectrum is broken up into 50 200 kHz bands. With each band TDM is used to multiplex multiple users. Some GSM telephones use smart cards (credit card sized devices containing a CPU). The serial number and telephone number are contained there, not in telephone, making for better security. Encryption is also used.

Personal Communication Services

Everyone would like to have a small cordless phone that works around the house and also anywhere in the world. Such a system is currently under vigorous development. In the US it is called PCS (Personal Communication Services), everywhere else it is called PCN (Personal Communication Network). PCS will use cellular technology, but with microcells, perhaps 50 - 100 m wide. This allows very low power (1/4 watt), which makes it possible to build very small, light phones. On the other hand, it requires many more cells. The small base stations in these cells are sometimes called telepoints.

UNIT: III

Data Link Layer Design issues: Services provided to the Network Layer - Framing - Error Control - Flow Control. Error Detection and Correction. Error Correcting Codes - Error Detecting Codes.

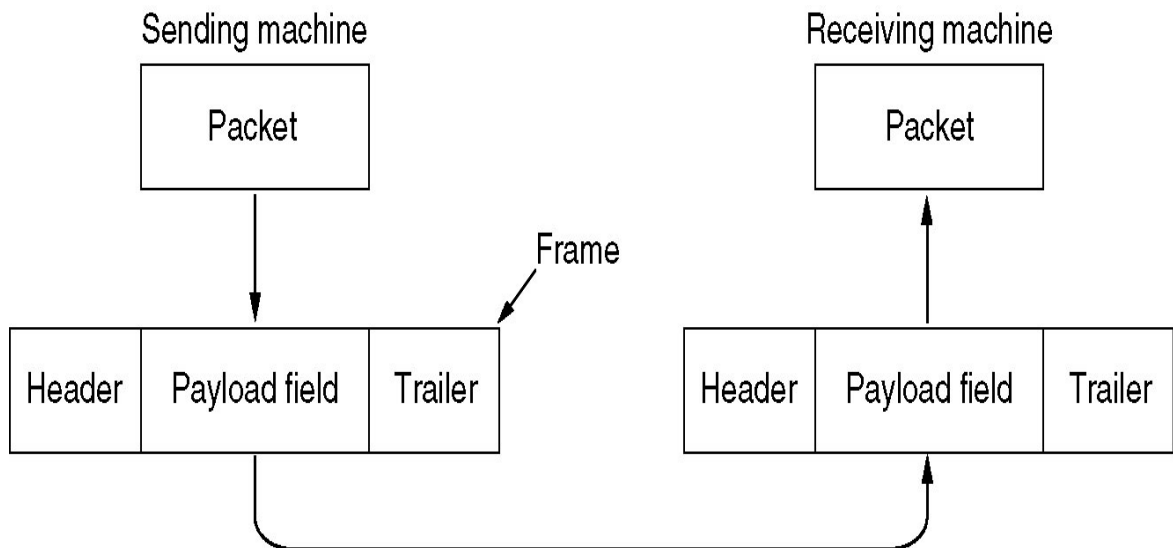
Elementary Data Link Protocols: On Unrestricted Simpler Protocol - a Simpler Stop - and - Wait Protocol -a Simpler Protocol for a Noisy Channel. a One Link Sliding Window Protocol -a Protocol Using Go Back N and Selective Repeat.

Data Link Layer Design Issues

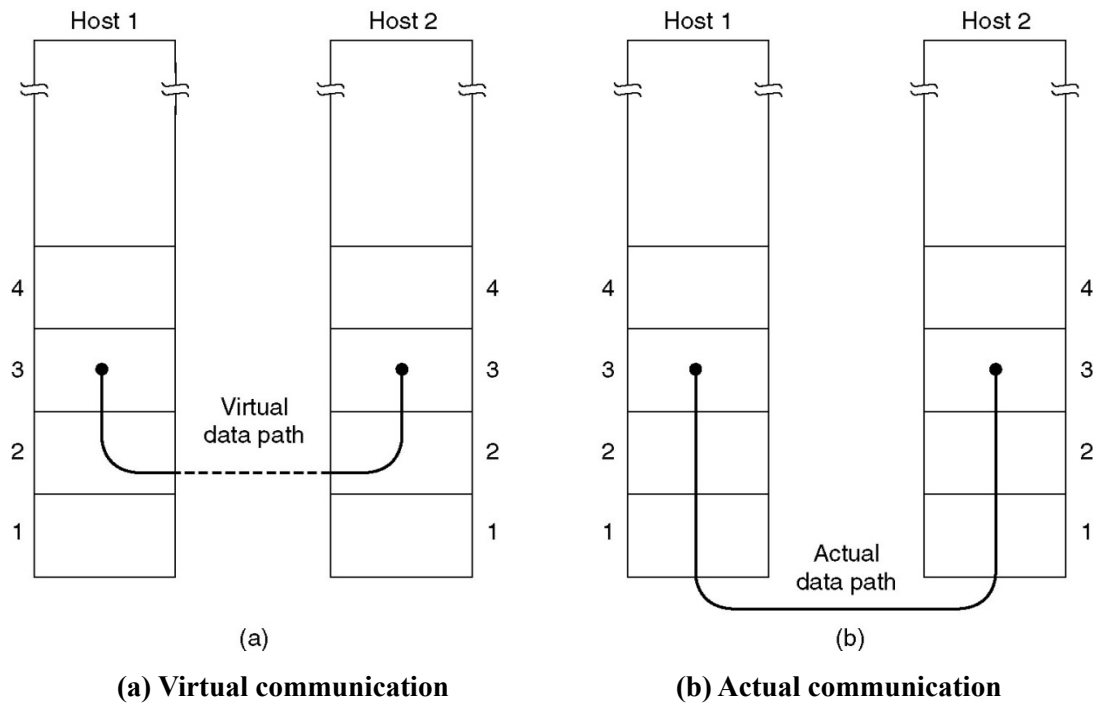
- Services Provided to the Network Layer
- Framing
- Error Control
- Flow Control

Functions of the Data Link Layer

- Provide service interface to the network layer
- Dealing with transmission errors
- Regulating data flow
- Slow receivers not swamped by fast senders



Services Provided to Network Layer



Types of services provided to the Network Layer

- Unacknowledged Connectionless service
- Acknowledged Connectionless service
- Acknowledged Connection-Oriented service

Unacknowledged Connectionless service

- Losses are taken care of at higher layers
- Used on reliable medium like coax cables or optical fiber, where the error rate is low.
- Appropriate for voice, where delay is worse than bad data.

Acknowledged Connectionless service

- Useful on unreliable medium like wireless.
- Acknowledgements add delays.
- Adding ack in the DLL rather than in the NL is just an optimization and not a requirement. Leaving it for the NL is inefficient as a large message (packet) has to be resent in that case in contrast to small frames here.

- On reliable channels, like fiber, the overhead associated with the ack is not justified.

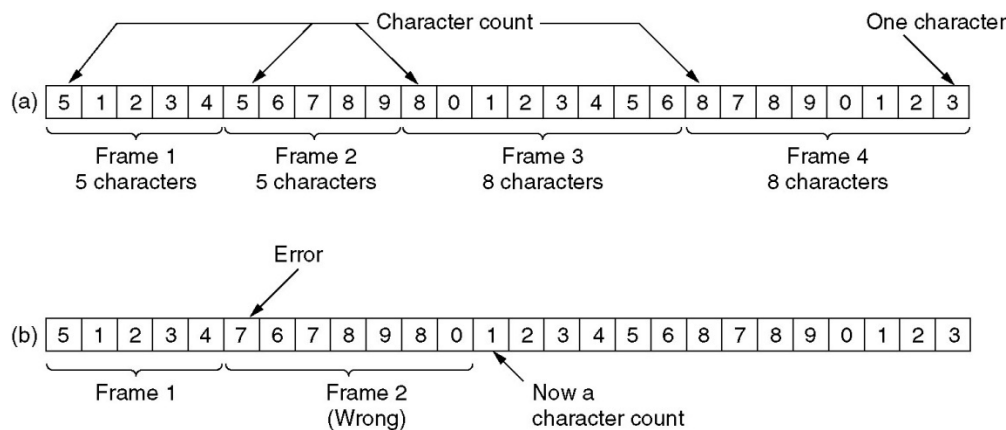
Acknowledged Connection-oriented service

- Most reliable,
- Guaranteed service –
 - Each frame sent is indeed received
 - Each frame is received exactly once
 - Frames are received in order
- Special care has to be taken to ensure this in connectionless services

Framing

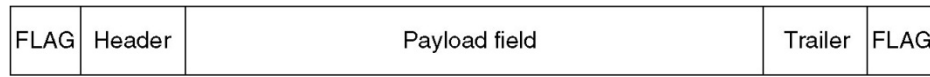
- Character Count
- Flag bytes with byte stuffing
- Flag bytes with bit stuffing

Framing with Character Count

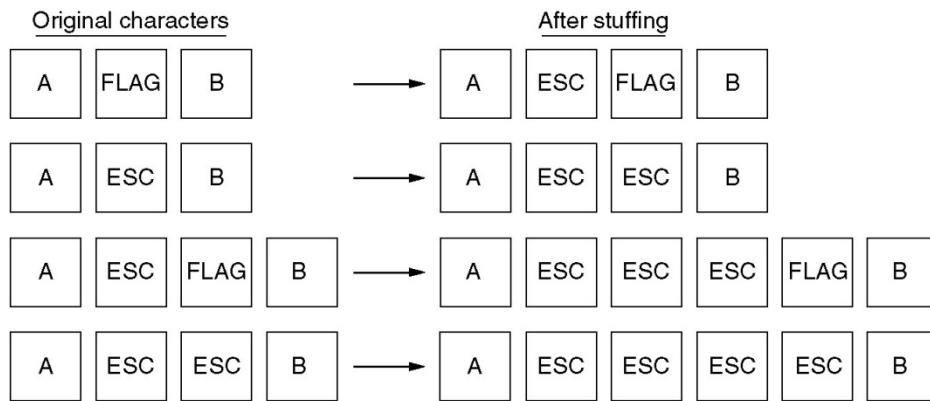


A character stream. (a) Without errors. (b) With one error.

Framing with byte stuffing



(a)



(b)

- Problem: fixed character size: assumes character size to be 8 bits : can't handle heterogeneous environment.

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 0 1 0 0 1 0

Stuffed bits

(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Bit stuffing

- The original data.
- The data as they appear on the line.
- The data as they are stored in receiver's memory after destuffing.

Error Control

- Positive and Negative feedback
- Timers : what happens when a frame completely vanishes : receiver neither sends a +ack nor –ack ... then timer comes to help.
- It may result in a frame being sent more than once and received more than once :

- solution : assign sequence numbers to frames

Flow Control

Another important design issue that occurs in the data link layer (and higher layers as well) is what to do with a sender that systematically wants to transmit frames faster than the receiver can accept them. This situation can occur when the sender is running on a fast, powerful computer and the receiver is running on a slow, low-end machine.

Two approaches are commonly used. In the first one, **feedback-based flow control**, the receiver sends back information to the sender giving it permission to send more data, or at least telling the sender how the receiver is doing. In the second one, **rate-based flow control**, the protocol has a built-in mechanism that limits the rate at which senders may transmit data, without using feedback from the receiver.

For example, hardware implementations of the link layer as **NICs (Network Interface Cards)** are sometimes said to run at “wire speed,” meaning that they can handle frames as fast as they can arrive on the link.

Error Detection and Correction

Network designers have developed two basic strategies for dealing with errors. Both add redundant information to the data that is sent. One strategy is to include enough redundant information to enable the receiver to deduce what the transmitted data must have been. The other is to include only enough redundancy to allow the receiver to deduce that an error has occurred (but not which error) and have it request a retransmission. The former strategy uses **error-correcting codes** and the latter uses **error-detecting codes**. The use of error-correcting codes is often referred to as **FEC (Forward Error Correction)**.

Error Correcting Code

Four different error-correcting codes:

1. Hamming codes.
2. Binary convolutional codes.
3. Reed-Solomon codes.
4. Low-Density Parity Check codes.

All of these codes add redundancy to the information that is sent. A frame consists of m data (i.e., message) bits and r redundant (i.e. check) bits. In a **block code**, the r check bits are computed solely as a

function of the m data bits with which they are associated, as though the m bits were looked up in a large table to find their corresponding r check bits. In a **systematic code**, the m data bits are sent directly, along with the check bits, rather than being encoded themselves before they are sent. In a **linear code**, the r check bits are computed as a linear function of the m data bits.

Error Detecting Code

Error-correcting codes are widely used on wireless links, which are notoriously noisy and error prone when compared to optical fibers. Without error-correcting codes, it would be hard to get anything through.

Three different error-detecting codes. They are all linear, systematic block codes:

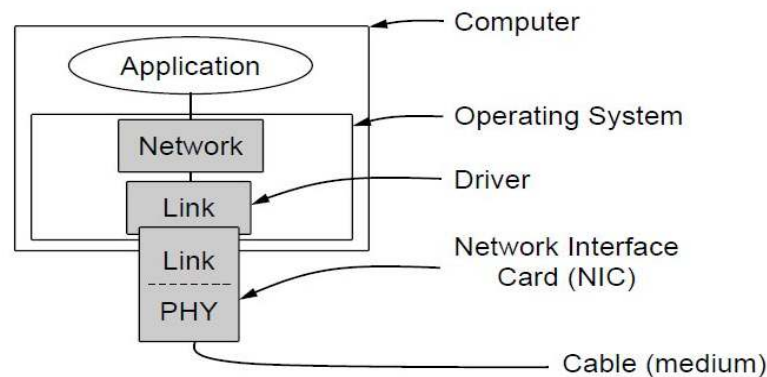
1. Parity.
2. Checksums.
3. Cyclic Redundancy Checks (CRCs).

CRC (Cyclic Redundancy Check)

- A message m : a string of bits corresponds to a polynomial denote it by $M(x)$.
- r check bits polynomial $R(x)$.
- Transmitted bits $m + r$ polynomial $T(x) = M(x) + R(x)$
- Generator polynomial $G(x)$
- r checkbits are computed so that when $G(x)$ divides $T(x)$, the remainder is zero.

Elementary Data Link Protocols

The physical layer, data link layer, and network layer are independent processes that communicate by passing messages back and forth. The physical layer process and some of the data link layer process run on dedicate hardware called a **NIC (Network Interface Card)**. The rest of the link layer process and the network layer process run on the main CPU as part of the operating system, with the software for the link layer process often taking the form of a **device driver**.



Implementation of the physical, data link, and network layers

When the data link layer accepts a packet, it encapsulates the packet in a frame by adding a data link header and trailer to it (see Fig. 3-1). Thus, a frame consists of an embedded packet, some control information (in the header), and a checksum (in the trailer). The frame is then transmitted to the data link layer on the other machine. We will assume that there exist suitable library procedures *to physical layer* to send a frame and *from physical layer* to receive a frame.

A Simpler Stop - and - Wait Protocol

The communication channel is still assumed to be error free, however, and the data traffic is still simplex. One solution is to build the receiver to be powerful enough to process a continuous stream of back-to-back frames (or, equivalently, define the link layer to be slow enough that the receiver can keep up). It must have sufficient buffering and processing abilities to run at the line rate and must be able to pass the frames that are received to the network layer quickly enough.

Protocols in which the sender sends one frame and then waits for an acknowledgement before proceeding are called **stop-and-wait**. An example of a simplex stop-and-wait protocol.

```

/* Protocol 2 (Stop-and-wait) also provides for a one-directional flow of data from sender to receiver.
The communication channel is once again assumed to be error free, as in protocol 1. However, this time
the receiver has only a finite buffer capacity and a finite processing speed, so the protocol must explicitly
prevent the sender from flooding the receiver with data faster than it can be handled. */

```

```

typedef enum {frame arrival} event type;

```

```

#include "protocol.h"

```

```

void sender2(void)
{
    frame s; /* buffer for an outbound frame */
    packet buffer; /* buffer for an outbound packet */
    event type event; /* frame arrival is the only possibility */
    while (true) {
        from network layer(&buffer); /* go get something to send */
        s.info = buffer; /* copy it into s for transmission */
        to physical layer(&s); /* bye-bye little frame */
        wait for event(&event); /* do not proceed until given the go ahead */
    }
}

void receiver2(void)
{
    frame r, s; /* buffers for frames */
    event type event; /* frame arrival is the only possibility */
    while (true) {
        wait for event(&event); /* only possibility is frame arrival */
        from physical layer(&r); /* go get the inbound frame */
        to network layer(&r.info); /* pass the data to the network layer */
        to physical layer(&s); /* send a dummy frame to awaken sender */
    }
}

```

A simplex stop-and-wait protocol

A Simpler Protocol for a Noisy Channel

The normal situation of a communication channel that makes errors. Frames may be either damaged or lost completely. We assume that if a frame is damaged in transit, the receiver hardware will detect this when it computes the checksum. If the frame is damaged in such a way that the checksum is nevertheless correct—an unlikely occurrence—this protocol (and all other protocols) can fail (i.e., deliver an incorrect packet to the network layer).

The sender could send a frame, but the receiver would only send an acknowledgement frame if the data were correctly received. If a damaged frame arrived at the receiver, it would be discarded. After a

while the sender would time out and send the frame again. This process would be repeated until the frame finally arrived intact.

The goal of the data link layer is to provide error-free, transparent communication between network layer processes. The network layer on machine *A* gives a series of packets to its data link layer, which must ensure that an identical series of packets is delivered to the network layer on machine *B* by its data link layer. In particular, the network layer on *B* has no way of knowing that a packet has been lost or duplicated, so the data link layer must guarantee that no combination of transmission errors, however unlikely, can cause a duplicate packet to be delivered to a network layer.

Consider the following scenario:

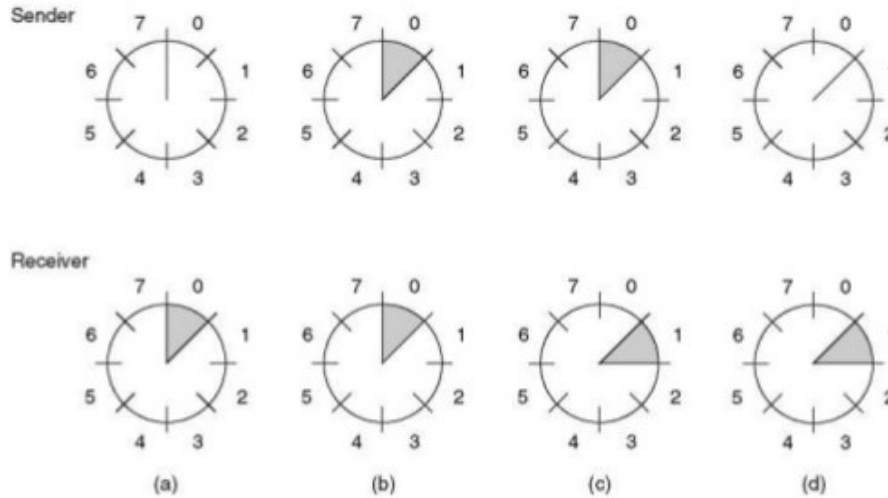
1. The network layer on *A* gives packet 1 to its data link layer. The packet is correctly received at *B* and passed to the network layer on *B*. *B* sends an acknowledgement frame back to *A*.
2. The acknowledgement frame gets lost completely. It just never arrives at all. Life would be a great deal simpler if the channel mangled and lost only data frames and not control frames, but sad to say, the channel is not very discriminating.
3. The data link layer on *A* eventually times out. Not having received an acknowledgement, it (incorrectly) assumes that its data frame was lost or damaged and sends the frame containing packet 1 again.
4. The duplicate frame also arrives intact at the data link layer on *B* and is unwittingly passed to the network layer there. If *A* is sending a file to *B*, part of the file will be duplicated (i.e., the copy of the file made by *B* will be incorrect and the error will not have been detected). In other words, the protocol will fail.

SLIDING WINDOW PROTOCOLS

A One-Bit Sliding Window Protocol

In the previous protocols, data frames were transmitted in one direction only. In most practical situations, there is a need to transmit data in both directions. One way of achieving full-duplex data transmission is to run two instances of one of the previous protocols, each using a separate link for simplex data traffic (in different directions). Each link is then comprised of a “forward” channel (for data) and a “reverse” channel (for acknowledgements). In both cases the capacity of the reverse channel is almost entirely wasted.

A ONE BIT SLIDING WINDOW PROTOCOL



A sliding window of size 1, with a 3-bit sequence number.

(a) Initially.

(b) After the first frame has been sent.

(c) After the first frame has been received.

(d) After the first acknowledgement has been received.

/* Protocol 4 (Sliding window) is bidirectional. */

```
#define MAX_SEQ 1 /* must be 1 for protocol 4 */
```

```
typedef enum {frame arrival, cksum err, timeout} event type;
```

```
#include "protocol.h"
```

```
void protocol4 (void)
```

```
{
```

```
seq nr next frame to send; /* 0 or 1 only */
```

```
seq nr frame expected; /* 0 or 1 only */
```

```
frame r, s; /* scratch variables */
```

```
packet buffer; /* current packet being sent */
```

```
event type event;
```

```
next frame to send = 0; /* next frame on the outbound stream */
```

```
frame expected = 0; /* frame expected next */
```

```
from network layer(&buffer); /* fetch a packet from the network layer */
```

```
s.info = buffer; /* prepare to send the initial frame */
```

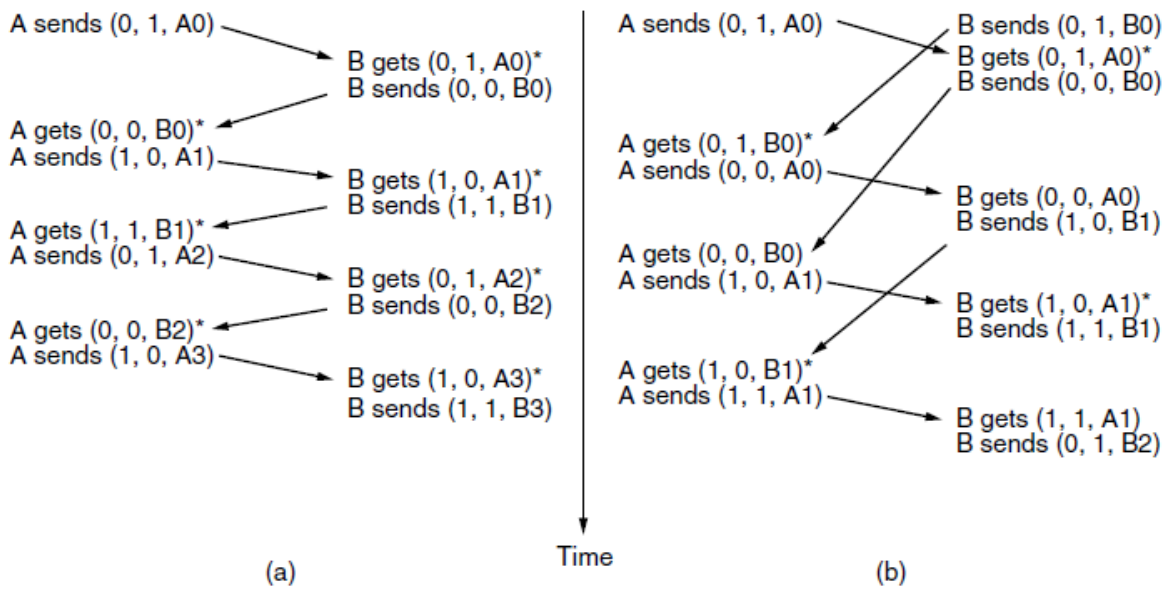
```
s.seq = next frame to send; /* insert sequence number into frame */
```

```

s.ack = 1 - frame expected; /* piggybacked ack */
to physical layer(&s); /* transmit the frame */
start timer(s.seq); /* start the timer running */
while (true) {
wait for event(&event); /* frame arrival, cksum err, or timeout */
if (event == frame arrival) { /* a frame has arrived undamaged */
from physical layer(&r); /* go get it */
if (r.seq == frame expected) { /* handle inbound frame stream */
to network layer(&r.info); /* pass packet to network layer */
inc(frame expected); /* invert seq number expected next */
}
if (r.ack == next frame to send) { /* handle outbound frame stream */
stop timer(r.ack); /* turn the timer off */
from network layer(&buffer); /* fetch new pkt from network layer */
inc(next frame to send); /* invert sender's sequence number */
}
}
s.info = buffer; /* construct outbound frame */
s.seq = next frame to send; /* insert sequence number into it */
s.ack = 1 - frame expected; /* seq number of last received frame */
to physical layer(&s); /* transmit a frame */
start timer(s.seq); /* start the timer running */
}
}

```

A 1-bit sliding window protocol



(a) Normal case (b) Abnormal case The notation is (seq, ack, packet number)

An asterisk indicates where a network layer accepts a packet

Protocol Using Go Back N

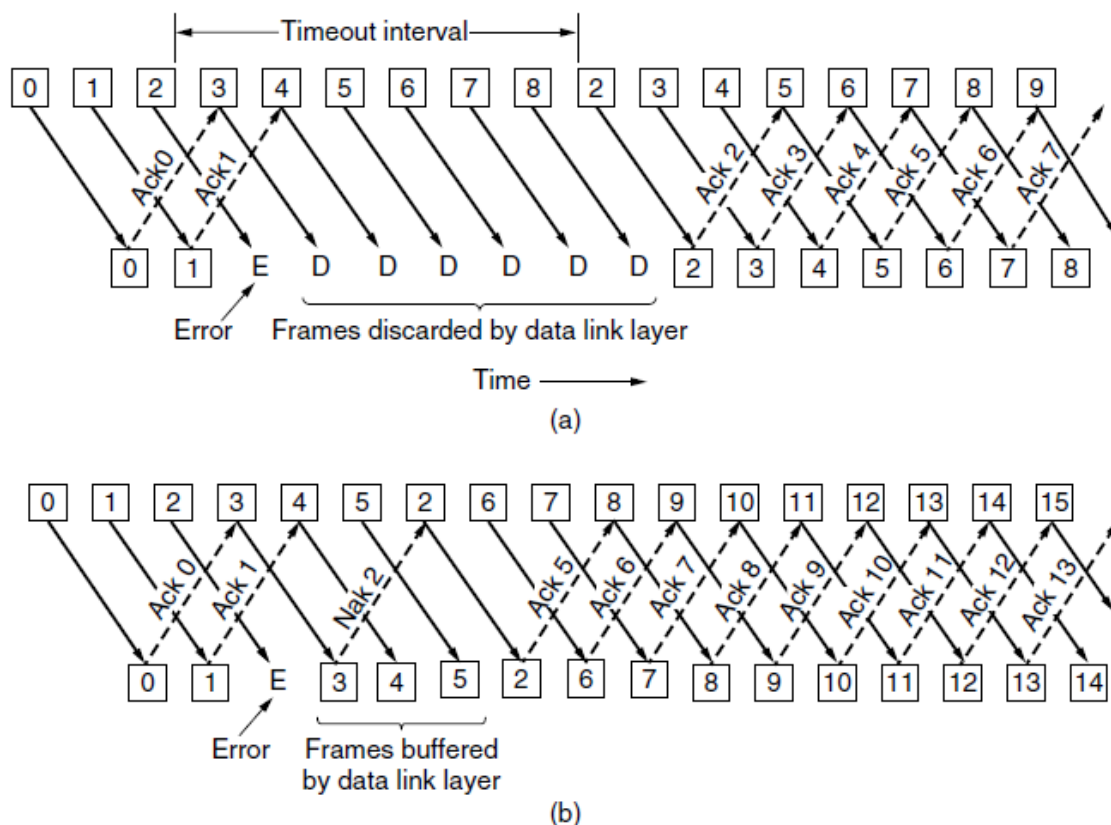
The transmission time required for a frame to arrive at the receiver plus the transmission time for the acknowledgement to come back is negligible. Sometimes this assumption is clearly false. In these situations the long round-trip time can have important implications for the efficiency of the bandwidth utilization. As an example, consider a 50-kbps satellite channel with a 500-msec round-trip propagation delay. Let us imagine trying to use protocol 4 to send 1000-bit frames via the satellite. At $t = 0$ the sender starts sending the first frame. At $t = 20$ msec the frame has been completely sent. Not until $t = 270$ msec has the frame fully arrived at the receiver, and not until $t = 520$ msec has the acknowledgement arrived back at the sender, under the best of circumstances (of no waiting in the receiver and a short acknowledgement frame). This means that the sender was blocked 500/520 or 96% of the time. In other words, only 4% of the available bandwidth was used. Clearly, the combination of a long transit time, high bandwidth, and short frame length is disastrous in terms of efficiency.

To find an appropriate value for w we need to know how many frames can fit inside the channel as they propagate from sender to receiver. This capacity is determined by the bandwidth in bits/sec multiplied by the one-way transit time, or the **bandwidth-delay product** of the link. We can divide this quantity by the number of bits in a frame to express it as a number of frames. Call this quantity BD . Then w should be set to $2BD + 1$. Twice the bandwidth-delay is the number of frames that can be outstanding if the sender continuously sends frames when the round-trip time to receive an acknowledgement is

considered. The “+1” is because an acknowledgement frame will not be sent until after a complete frame is received.

For smaller window sizes, the utilization of the link will be less than 100% since the sender will be blocked sometimes. We can write the utilization as the fraction of time that the sender is not blocked:

This $\text{link utilization} \leq \frac{w}{1 + 2BD}$ technique of keeping multiple frames in flight is an example of **pipelining**. Pipelining frames over an unreliable communication channel raises some serious issues. Two basic approaches are available for dealing with errors in the presence of pipelining,



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

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

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

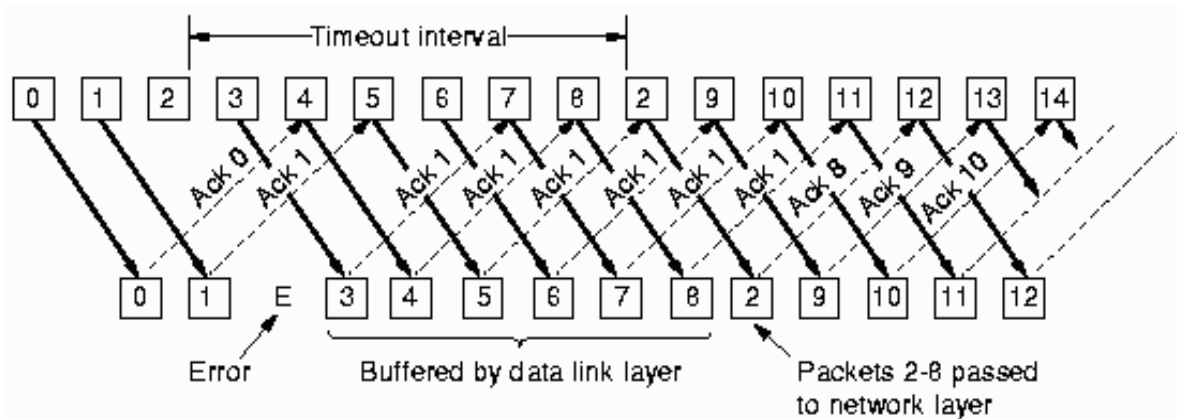
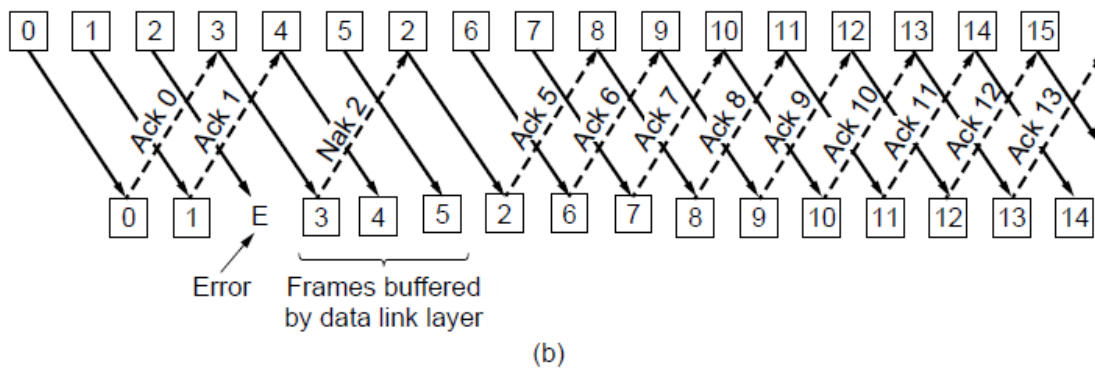
A Protocol Using Selective Repeat

The go-back-n protocol works well if errors are rare, but if the line is poor it wastes a lot of bandwidth on retransmitted frames. An alternative strategy, the selective repeat protocol, is to allow the receiver to accept and buffer the frames following a damaged or lost one.

In this protocol, both sender and receiver maintain a window of outstanding and acceptable sequence numbers, respectively. The sender's window size starts out at 0 and grows to some predefined maximum. The receiver's window, in contrast, is always fixed in size and equal to the predetermined maximum. The receiver has a buffer reserved for each sequence number within its fixed window.

NAK(Negative Acknowledgement)

- ✓ SRP is often combined with NAK
- ✓ When error is *suspected* by receiver, receiver request retransmission of a frame
 - Arrival of a damaged frame
 - Arrival of a frame other than the expected
- ✓ Does receiver keep track of NAK?
- ✓ What if NAK gets lost?
- ✓ To nak, or not to nak: that is the question



Selective Repeat with NAK

Select Repeat Protocol - Window Size

- ✓ Suppose 3-bit window is used and window size = MAX_SEQ = 7

sender

receiver

0 1 2 3 4 5 6 sent

0 1 2 3 4 5 6 accepted

0 through 6 to network layer

all acknowledgements lost

0 retransmitted

0 accepted

ack 6 received

7 sent

7 accepted

7 and 0 to network layer

- ✓ Problem is caused by new and old windows overlapped
- ✓ Solution
 - Window size = $(MAX_SEQ + 1) / 2$
 - E.g., if 4-bit window is used, $MAX_SEQ = 15$
 $\Rightarrow$ window size = $(15 + 1) / 2 = 8$
- ✓ Number of buffers needed = window size

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

UNIT –IV

Network Layer Design issues : Services Provided to the Transport Layer - Internal Organization - Comparison of Virtual Circuit and Datagram Subnets. the Transport Service: Quality of Service – Transparent Service primitives.

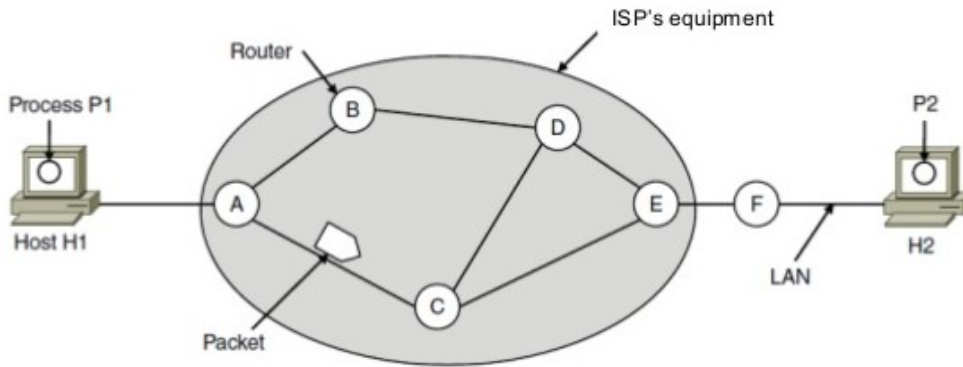
Element of Transparent protocols: Addressing, Establishing and releasing connection - the Internet Transport Protocols(TCP and UDP): the TCP Service Model - the TCP Protocol -the TCP Segment Header - TCP Connective Management - TCP Transmission Policy - TCP Congestion Control -TCP Timer Management - UDP - Wireless TCP and UDP.

<http://iips.icci.edu.iq/images/exam/Computer-Networks---A-Tanenbaum---5th-edition.pdf>
<http://www.cs.ru.nl/~ths/a3/html/h6/h6.html>

Network Layer Design issues

A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-point link to the ISP. The packet is stored there until it has fully arrived and the link has finished its processing by verifying the checksum. Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered. This mechanism is store-and-forward packet switching.

Store-and-Forward Packet Switching



The environment of the network layer protocols

Services Provided to the Transport Layer

The network layer provides services to the transport layer at the network layer/transport layer interface. An important question is precisely what kind of services the network layer provides to the transport layer. The services need to be carefully designed with the following goals in mind,

1. Services should be independent of the router technology.
2. The transport layer should be shielded from the number, type, and topology of the routers present.
3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

Given these goals, the designers of the network layer have a lot of freedom in writing detailed specifications of the services to be offered to the transport layer.

This freedom often degenerates into a raging battle between two warring factions. The discussion centers on whether the network layer should provide connection oriented service or connectionless service. One camp (represented by the Internet community) argues that the routers job is moving packets around and nothing else.

In this view (based on 40 years of experience with a real computer network), the network is inherently unreliable, no matter how it is designed. Therefore, the hosts should accept this fact and do error control (i.e., error detection and correction) and flow control themselves.

This viewpoint leads to the conclusion that the network service should be connection less, with primitive END PACKET and RECEIVE PACKET and little else. In particular, no packet ordering and flow control should be done, because the hosts are going to do that anyway and there is usually little to be gained by doing it twice. This reasoning is an example of the **end-to-end argument**, a design principle that has been very influential in shaping the Internet (Saltzer et al., 1984).

Furthermore, each packet must carry the full destination address, because each packet sent is carried independently of its predecessors, if any the other camp (represented by the telephone companies) argues that the network should provide a reliable, connection-oriented service. They claim that 100 years of successful experience with the worldwide telephone system is an excellent guide.

In this view, quality of service is the dominant factor, and without connections in the network, quality of service is very difficult to achieve, especially for real-time traffic such as voice and video.

Even after several decades, this controversy is still very much alive. Early, widely used data networks, such as X.25 in the 1970s and its successor Frame Relay in the 1980s, were connection-oriented. However, since the days of the ARPANET and the early Internet, connectionless network layers have grown tremendously in popularity.

The IP protocol is now an ever-present symbol of success. It was undeterred by a connection-oriented technology called ATM that was developed to overthrow it in the 1980s; instead, it is ATM that is now found in niche uses and IP that is taking over telephone networks. Under the covers, however, the Internet is evolving connection-oriented features as quality of service.

Comparison of Virtual Circuit and Datagram Subnets

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Several trade-offs exist between virtual circuits and datagrams. One trade-off is setup time versus address parsing time. Using virtual circuits requires a setup phase, which takes time and consumes resources. However, once this price is paid, figuring out what to do with a data packet in a virtual-circuit network is easy: the router just uses the circuit number to index into a table to find out where the packet goes. In a datagram network, no setup is needed but a more complicated lookup procedure is required to locate the entry for the destination.

A related issue is that the destination addresses used in datagram networks are longer than circuit numbers used in virtual-circuit networks because they have a global meaning. If the packets tend to be fairly short, including a full destination address in every packet may represent a significant amount of overhead and hence a waste of bandwidth.

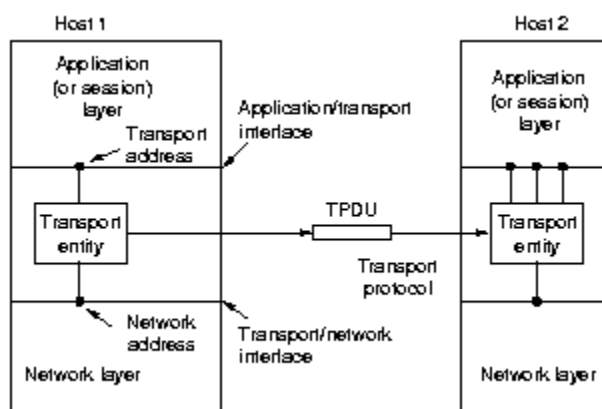
Virtual circuits have some advantages in guaranteeing quality of service and avoiding congestion within the network because resources (e.g., buffers, band-width, and CPU cycles) can be reserved in advance, when the connection is established. Once the packets start arriving, the necessary bandwidth and router capacity will be there. With a datagram network, congestion avoidance is more difficult.

The Transport Service

Services provided to the upper layers

The **transport entity** can be in the operating system kernel, in a separate user process, in a library package bound into network applications, or on a network interface card. There are 2 types of transport services, connection-oriented and connectionless, just as in the network layer, and they are very similar to them. Why are there than 2 separate layers?. The answer is subtle, but crucial. The network layer is part of the communication subnet and is run by the carrier, at least for the WAN. The users have no control over this, what happens if the offered connection-oriented service is unreliable, frequently loses packets or its routers crash from time to time? The user can not put in better routers or put more error handling in the data link layer.

In essence, the transport layer makes it possible for the transport service to be more reliable than the underlying network service, lost packets and mangled data can be detected and compensated for. Furthermore, the transport service primitives can be designed to be independent of the network service primitives, which may vary considerably from network to network (e.g. a connectionless LAN service may be quite different than a connection-oriented WAN service). Due to the transport layer, application programs can be written using a standard set of primitives so that they can work on a wide variety of networks without having to worry about dealing with different subnet interfaces and unreliable transmissions. The bottom 4 layers are therefore



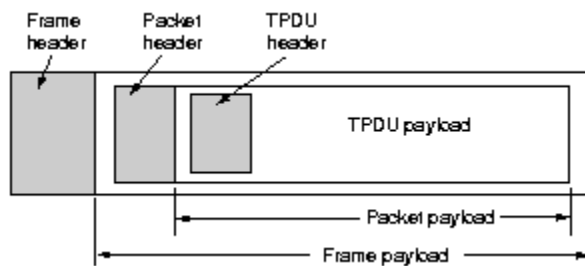
sometimes seen as the **transport service provider** and the higher layers as the **transport service user**.

Transport service primitives

Primitive	TPDU sent	Meaning
Listen	None	Block until some process tries to
Connect	Connection request	Actively attempt to establish a c
Send	Data	Send information
Receive	None	Block until a Data TPDU arrive

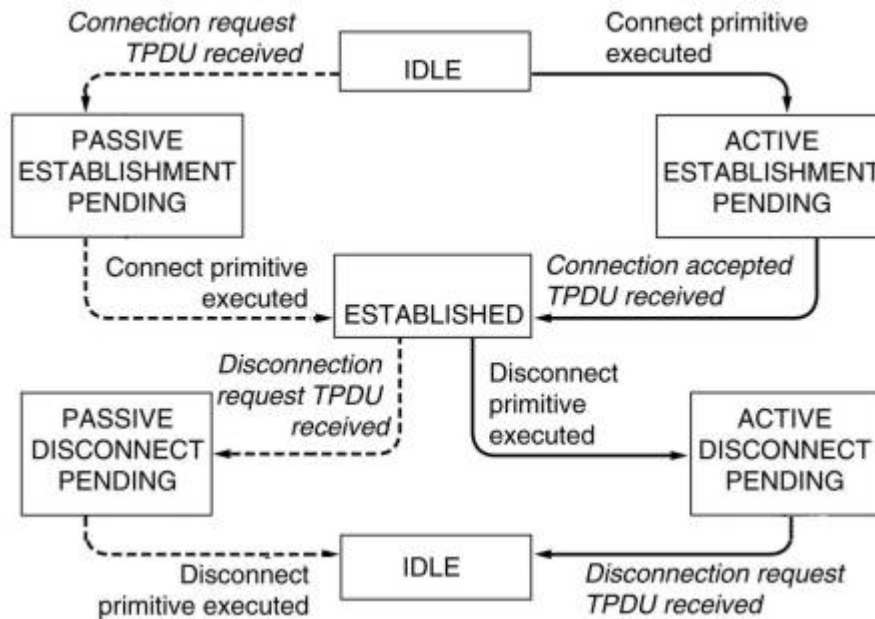
Disconnect	Disconnection req.	This side wants to release the connection
------------	--------------------	---

The primitives for a simple transport service. TPDU means Transport Protocol Data Unit. To the transport users, a connection is a reliable bit pipe: one user stuffs bits in and they magically appear at the other end.



Disconnection, meaning no more data needs to be send and the used buffer space can be released, has 2 variants. In the asymmetric variant, either transport user can issue a disconnect primitive, which results in a disconnect TPDU being sent to

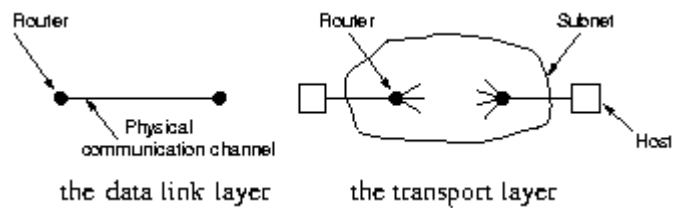
the remote transport entity. Upon arrival the connection is released. In the symmetric variant, each direction is closed separately, independent of the other one. When one side does a disconnect, it means it has no more data to send, but is still willing to accept data from its partner. A connection is released when both sides have done a disconnect.



A state diagram for connection establishment and release for these simple primitives. Each transition is triggered by some event, either a primitive executed by the local transport user or an incoming packet (labeled in italics). The solid lines

show the client's state sequence, the dashed lines the server ones.

Elements of transport protocols



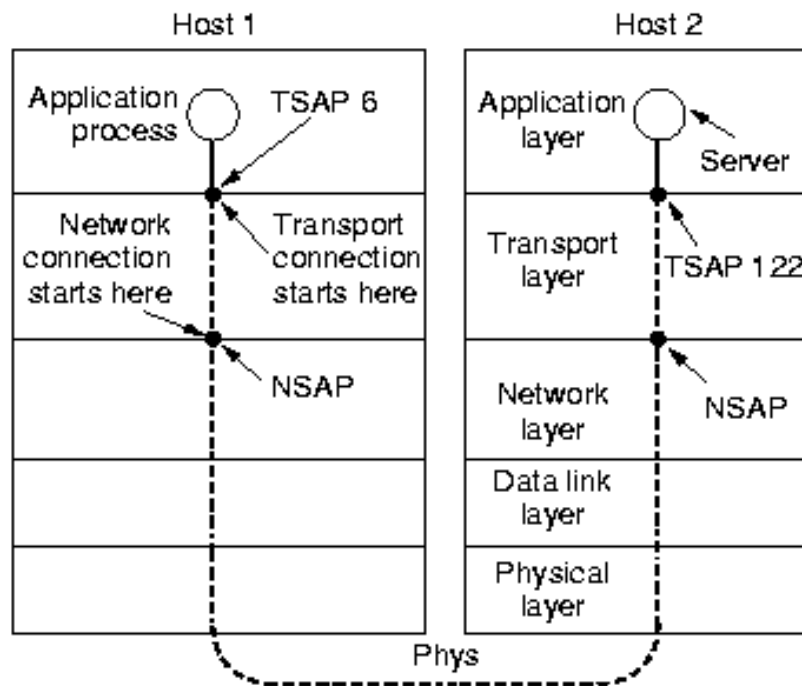
In some ways, transport protocols resemble the data link protocols, both have to deal with error control, sequencing, flow control and other issues. In the transport layer explicit addressing of destinations is needed.

The potential existence of storage capacity in the subnet has to be dealt with. If the subnet uses datagrams and adaptive routing inside, there is the possibility that a packet might be stored for a number of seconds and then delivered later. This can sometimes be disastrous, specially during connection and disconnection, and requires the use of special protocols.

Buffering and flow control are needed in both layers, but the presence of a large and dynamically varying number of connections in the transport layer, may require a different approach than is used in the data link layer. The idea of dedicating many buffers to each connection in the transport layer is not attractive.

Addressing

Transport layer addresses are called **TSAP** (Transport Service Access Point). In Internet, they consists of an (IP address, local port) pair, in ATM of an (AAL, SAP) pair. An IP address is an example of a **NSAP**. How does the user process on host 1 know that the address of the server, e.g. the time-of-day server, is attached to TSAP 122? One possibility is, that it has always been there and every one knows it. This might work for a small number of key services that never change, but, in general, user processes often want to talk to other user processes that only exist for a short time and do not have a TSAP that is known in advance. Furthermore, if there are potentially many server processes, most of which are rarely used, it is wasteful to have each of them active and listening to a stable TSAP all day long.



In the **initial connection protocol**, used by UNIX hosts, each machine that wishes to offer services to remote users, has a special **process server**, that acts as a proxy for less-heavily used servers. It listens to a set of ports at the same time, waiting for a TCP connection request. If a request comes in, the process server spawns off the requested server, allowing it to inherit the existing connection with the user. The new server does the work and the process server goes back listening for new requests.

Often a **name server** or **directory server** is used. It listens at a well know port for clients which can ask for the TSAP corresponding with a given service name, e.g. time-of-day. After receiving the TSAP, the client disconnects and establishes a connection with the service of which it now knows the TSAP. When a new service is created, it must register itself with the name server, giving it its name (typically an ASCII string) and the address of its TSAP.

Establishing a connection

The problem here is the existence of delayed duplicates. It can be attacked in various ways, none of them very satisfactory.

One way is to use throwaway transport addresses, each time a transport address is needed, a new one is generated. This makes the process server model impossible. Another way is to give each connection a connection identifier (e.g. a sequence number incremented for each connection established), chosen by the initiating party, and put that in each TPDU. After a connection is released, each transport entity could update a table listing obsolete connections as (peer transport entity, connection identifier) pairs. Whenever a connection request came in, it could be checked against the table. This scheme has a basic flaw: it requires each transport entity to maintain a certain amount of history information indefinitely and that could be lost when the machine crashes.

If we can assure that no packet lives longer than some known time, the problem becomes somewhat more manageable. This can be achieved using one of the following techniques: restricting subnet design, using a hop or time-to-live field in each packet, or timestamping each packet (times must somehow be synchronized between hosts). In practice, we will need to guarantee not only that a packet is dead, but also that all acks to it are dead, so we will now introduce T , some small multiple of the true maximum lifetime of a packet (and its acks).

With packet lifetimes bounded, it is possible to devise a foolproof way to establish connections. Each host is equipped with a time-of-day clock, a binary counter that increments itself at uniform intervals. The number of bits must equal or exceed the number of bits in the sequence numbers used in the TPDU's. The clock is assumed to continue running even if the host goes down. The clocks at different hosts need not to be synchronized. The basic idea is that two identical numbered TPDU's are never outstanding at the same time. When a connection is set up, the low-order k bits of the clock are used as the initial sequence number (also k bits). The sequence space should be so large that by the time sequence numbers wrap around, old TPDU's with the same sequence number are long gone.

•

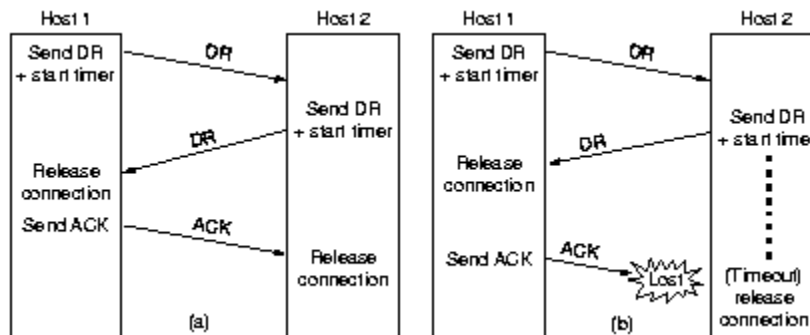
☐ CP(egg-v) ☐ ☐ CP(egg-v) ☐ ☐ ☐ ☐

sequence number can be in existence. The important thing is that no combination of old CA, CR or other TPDU can cause a connection to be set up by accident.

Releasing a connection

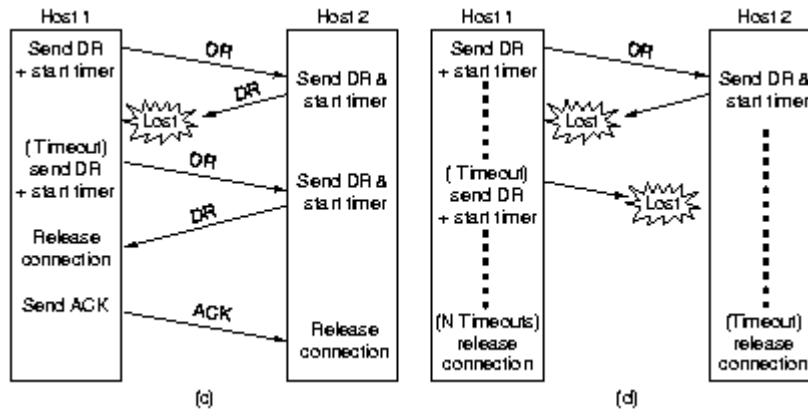
Asymmetric release is abrupt and may result in data loss, in case one site sends a disconnect request while data from the other side is still under way. Therefore usually symmetric release is used.

A three-way handshake is usually used to release a connection. Timers must be used to handle the cases that TPDU's are lost.

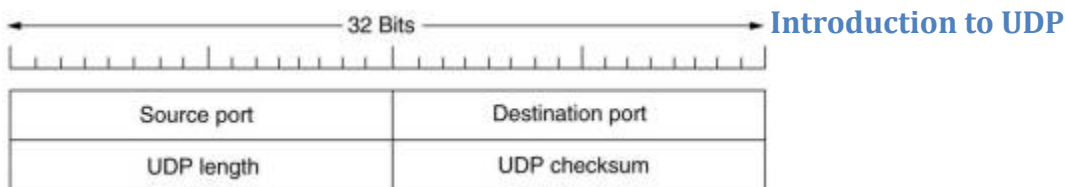


In order to handle the case that a lot of TPDU's are lost the number of retransmissions can be maximized.

Another way of assuring termination is to have a rule saying that a connection is automatically terminated if no TPDU is received for a certain number of seconds. It is then necessary for each transport entity to have a timer that is restarted after each TPDU sent. If this timer expires, a dummy TPDU must be sent, just to keep the other side from disconnecting.



The Internet transport protocol UDP



UDP (User Datagram Protocol) is just basically IP with a short header added. The port numbers indicate the sending and receiving transport end points. When a UDP packet arrives, its payload is sent to the process attached to the destination port. The checksum is optional and stored as 0 if not computed, a calculated 0 checksum is stored as all 1s. UDP does not do flow control, error control or retransmission upon receipt of a bad segment. All of that is up to the user processes.

The Internet transport protocol TCP

Introduction to TCP

TCP (Transmission Control Protocol) provides a reliable byte stream over an unreliable internetwork. Each machine supporting TCP has a TCP transport entity, either a user process or part of the kernel, that manages TCP streams (connections) and interfaces to the IP layer. A TCP entity accepts user data streams from local processes, breaks them up into pieces not exceeding 64K (usually about 1460 bytes to fit in a single Ethernet frame) and sends each piece as a separate IP datagram. The receiver side gives IP datagrams containing TCP data to its TCP entity, which reconstructs the original byte streams.

IP gives no guarantees that datagrams will be delivered properly, so it is up to TCP to time out and retransmit. Also IP datagrams might be delivered in the wrong order, it is up to TCP to rearrange them in the proper sequence.

The TCP service model

The 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 and a 16 bit number local to that host, called a **port**. A port is the TCP name for a TSAP. A connection must then explicitly be established between a socket on the sending machine and a socket on the receiving machine. Two or more connections may terminate at the same socket. Connections are identified by the socket identifiers at both ends, that is (socket1, socket2). No VC numbers or other identifiers are used. Port numbers below 1024 are called **well-known ports** and are reserved for standard services, like FTP or TELNET.

All TCP connections are full-duplex (traffic can go in both directions) and point-to-point (each connection has exactly 2 end points). Multicasting or broadcasting are not supported. A TCP connection is a byte stream, not a message stream. For example, if the sending process does four 512 byte writes to a TCP stream, these data may be delivered to the receiving process as four 512 byte chunks, two 1024 bytes chunks, one 2048 byte chunk or some other way. This is analogous to UNIX files.

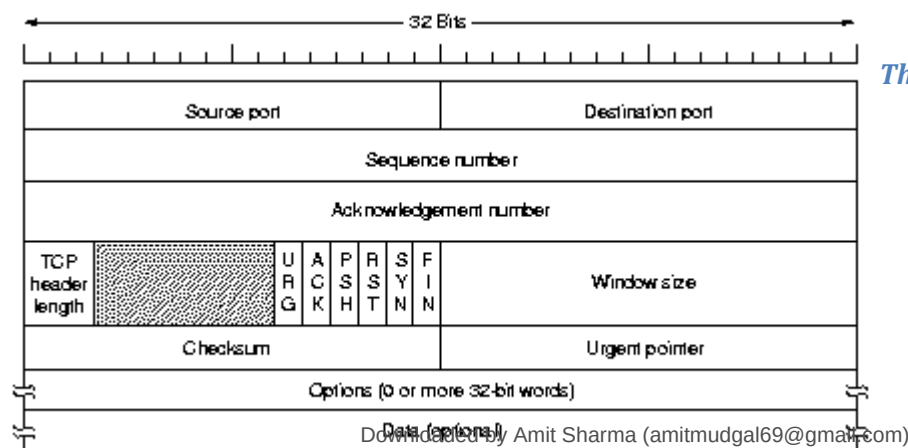
TCP may send user data immediately or buffer it, in order to send larger IP datagrams. Applications might use the **PUSH** flag to indicate that TCP should send the data immediately. The application can also send **urgent data** (e.g. when an interactive user hits the CTRL-C key), TCP then puts control information in its header and the receiving side interrupts (gives a signal in UNIX terms) to the application program using the connection. The end of the urgent data is marked, but not its beginning, the receiving application has to figure that out. This provides a crude signaling mechanism.

The TCP protocol

Every byte on a TCP connection has its own unsigned 32 bit sequence number. They are used both for acks, which uses a 32 bit header field and for the window mechanism, which uses a separate 16 bit header field.

The sending and receiving TCP entities exchange data in the form of segments. A **segment** consists of a fixed 20 byte header (plus an optional part) followed by 0 or more data bytes. The TCP software decides how big segments should be, it can accumulate bytes from several writes into one segment or split data from one write over multiple segments. Each segment (including the header) must fit into the 64K IP payload. Each network has a **MTU** (maximum transfer unit) and a segment must fit in it. In practice, the MTU is generally a few thousand bytes and thus defines the upper bound on segment size. If a segment passes through a sequence of networks and hits one whose MTU is smaller than the segment, the router at the boundary fragments the segment. Each fragment is a new segment having its own TCP and IP header, thus fragmentation increases the total overhead.

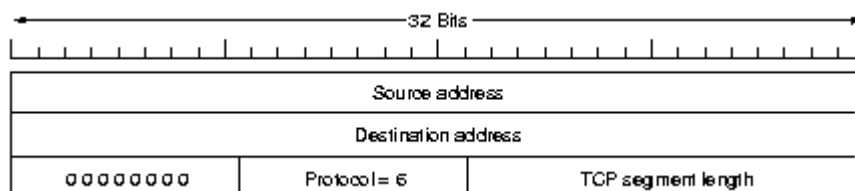
TCP basically uses a sliding window protocol. On sending a segment a timer is started, on arrival the receiver sends back a segment (with data if any exists) bearing an ack equal to the next sequence number it expects to receive. This sounds simple, but segments can be fragmented and the parts can be lost or delayed so much that a retransmission occurs. If a retransmitted segment takes a different route and is fragmented differently, bytes and pieces of both the original and the duplicate can arrive sporadically, requiring a careful administration to achieve a reliable byte stream.



The TCP header

URG is set to 1 if the urgent pointer is in use which indicates a byte offset from the current sequence number to the end of the urgent data. An ACK of 1 indicates a valid ack number. The PSH bit indicates pushed data, requesting the receiver to deliver the received data directly to its application program and not buffer it. The RST bit is used to reset a connection that has become confused due to a host crash or some other reason.

The SYN bit is used to establish a connection and the FIN bit to release one. The latter tells that the sender has no more data to send, but it may continue to receive data. Both SYN and FIN segments have sequence numbers and are thus guaranteed to be processed in the correct order. The window field tells how many bytes may be sent starting at the byte acknowledged. A window field of 0 is valid, it tells the sender to be quiet for a while. Permission to send can be granted later by sending a segment with the same ack number and a nonzero window field.



A checksum is also provided for extreme reliability. It checksums the header, the data and the conceptual pseudo header. The checksum is a simple one, just adding the 16-bits words (using padding if needed) in 1's complement and then take the 1's complement of the sum. The receiving side should thus find a checksum of 0.

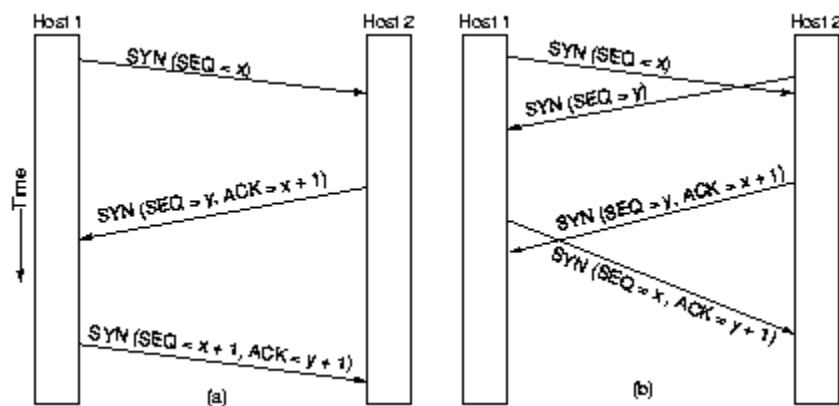
The pseudoheader contains IP addresses and thus violates the protocol hierarchy. It helps to detect misdelivered packets.

The options field was designed to provide a way to add extra facilities. An important option is to allow each host to specify the maximum TCP payload it is willing to accept (small host might not be able to accept very large segments). During connection, each side can announce its maximum and the smallest is taken. If a host does not use this option, it defaults to a 536 byte payload, which all Internet hosts are required to accept.

For lines with high bandwidth, high delay or both, the 64 KByte window is often a problem. On a T3 line (44.736 Mbps), it takes only 12 msec to output. If the round trip propagation is 50 msec (typical for a transcontinental fiber), the sender will be idle 75% of the time waiting for an ack. The window scale options allows both sides to negotiate a scale factor for the window field, allowing windows of up to 4 GBytes. Most TCP implementations now support this option.

Another option (proposed in RFC 1106 and widely implemented) is the use of selective repeat instead of go back n. It introduced NAK's, to allow the receiver to ask for specific, not (yet) received data bytes, after it has received following bytes. After it gets these, it can ack all its buffered data, thus reducing the amount of retransmitted data. This is nowadays important as memory is cheap and bandwidth is still small or expensive.

TCP connection management



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 particular other side or nobody in particular. The other side executes a connect primitive, specifying the IP and port to which it wants to connect, the maximum TCP segment size, possible other options and optionally some user data (e.g. a password). The connect primitive sends a TCP segment with the SYN bit on and the ACK bit off.

The receiving server checks to see if there is a process that has done a listen on the port given in the destination field. If not, it sends a reply with the RST bit on to reject the connection. Otherwise it gives the TCP segment to the listening process, which can accept or refuse (e.g. if it does not like the client) the connection. On acceptance a SYN is send, otherwise a RST. Note that a SYN segment occupies 1 byte of sequence space so it can be acked unambiguously. In the event that 2 host simultaneously attempt to establish a connection between the same two sockets, still just one connection is established, because connections are identified by their end points. For the initial sequence number a clock based scheme is used, with a clock tick every 4 μ sec. For additional safety, when a host crashes it may not reboot for 120 sec.

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 ack of FIN of last close

Releasing a TCP connection is symmetric. Either part can send a TCP segment with the

FIN bit set, meaning it has no more data to send. When the FIN is acked, that direction is shut

TCP transmission policy



Senders are not required to transmit data as soon as they come in from the application. Usually **Nagle's algorithm** is used. When data come into the sender one byte at a time (e.g. on a Telnet

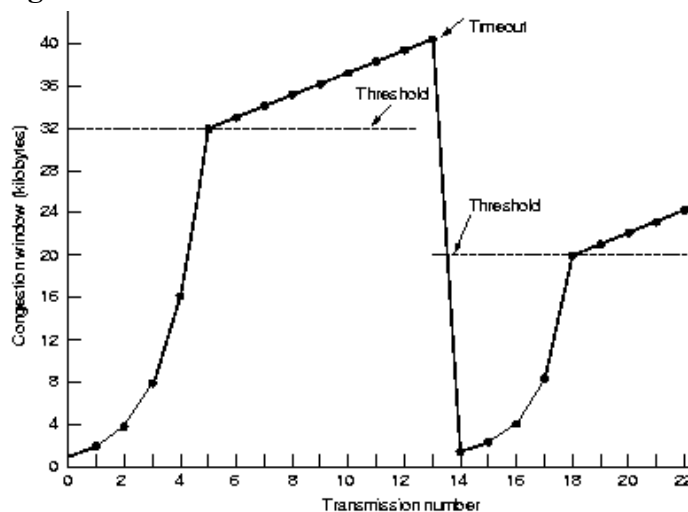
connection), just the first byte is sent and the rest is buffered until the outstanding byte is acked. Sometimes it is better to disable this, e.g. when mouse movements are sent in X-Windows applications.

Receivers are also not required to send acks and window updates immediately. Many implementations delay them for 500 msec in the hope of acquiring some data on which to hitch a free ride.

Another problem is the **silly window syndrome**, occurring when the sender transmits data in large blocks, but an interactive application on the receiver side reads data 1 byte at a time. The receiver continuously gives 1 byte window updates and the sender transmits 1 byte segments. Clark's solution is to let the receiver only send updates when it can handle the maximum segment size it advertised when the connection was established or when its buffer is half empty.

The receiver usually uses selective update, but go back n can also be used.

TCP congestion control

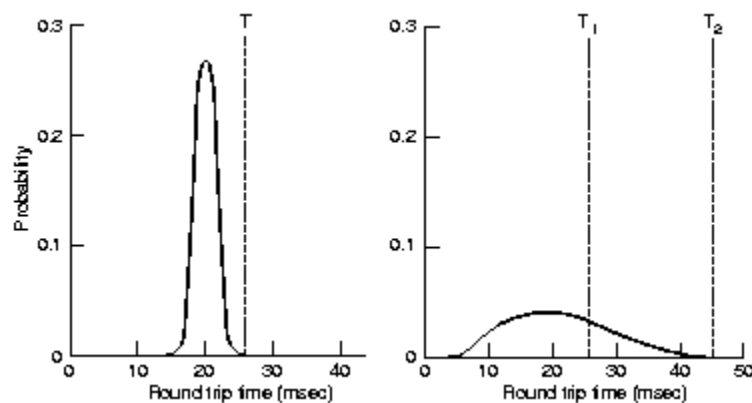


All Internet TCP algorithms assume that time-outs are caused by congestion due to network and receiver capacity as lost packets due to noise on the transmission lines are rare these days. Each sender maintains two windows: the window the receiver has granted (indicating the receiver capacity) and the **congestion window** (indicating the network capacity). The number of bytes that may be sent is the minimum of the 2 windows.

Initially the congestion window is the MTU. It is doubled on each burst successfully (an ack received before the timeout) sent. This exponential increase (called the **slow start**) continues until the **threshold** (initially 64K) is reached, after which the increase is linearly with 1 MTU.

When a timeout occurs, the threshold is set to half the current congestion window and the slow start is repeated. If an ICMP source quench packet comes in, it is treated the same way as a timeout.

TCP timer management



The **retransmission timer** has to handle the large variation in round trip time occurring in TCP. See the example right, on the left is a normal situation for the data link layer. For each segment the round trip time M is measured and the estimates of the mean and mean deviation are updated as:

$$RTT = \beta RTT + (1-\beta)M$$

$$D = \beta D + (1-\beta) |RTT - M|$$

with β a smoothing parameter, typically $7/8$. The timeout is then set to: $RTT + 4 D$

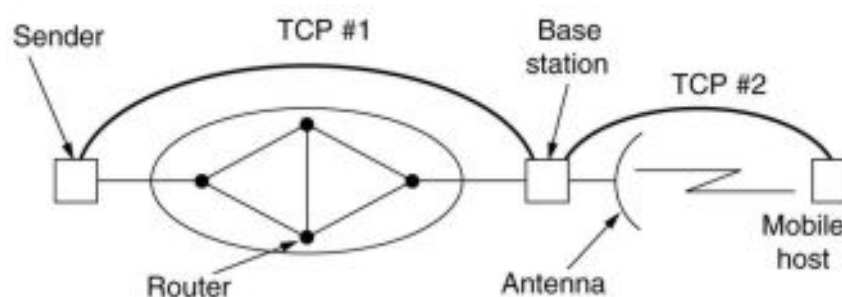
With **Karn's algorithm**, RTT and D are not updated for retransmitted segments. Instead, the timeout is doubled on each failure until the segments get through the first time. When a window size of 0 is received, the **persistence timer** is used to guard against the lost of the next window update.

Some implementations also use a **keepalive timer**. When a connection has been idle for a long time, the timeout causes a packet to be sent to see if the other side is still alive. If it fails to respond, the connection is terminated. This feature is controversial because it adds overhead and may terminate an otherwise healthy connection due to a transient network problem. The last timer is the one used in the TIMED WAIT state while closing, running for twice the maximum packet lifetime to make sure that when a connection is closed, all packets created by it have died off.

Wireless TCP and UDP

Transport protocols should be independent of the technology of the underlying network layer. In practice, most TCP implementations have been carefully optimized based on assumptions that are true for wired networks but not for wireless networks. Off course, they work correctly for wireless network, but the performance is low. On a wireless network if a packet is lost, it is usually not due to congestion, but due to "noise" on the transmission. TCP should not slow down, but retransmit as soon as possible. That can be done on a host which knows it sends over a wireless network, but what if the first part of a connection from a sender to a receiver is over a wired network, and the last part over a wireless link?

Using an **indirect TCP** solution is a possibility. But the acknowledgment, returned from the base station to the sender, does not indicate that the mobile host has received the data. Another possibility is to add a **snooping agent** on the base station. It watches the interchange between base station and mobile host and performs retransmissions (and interception of duplicated acknowledgments) on that part alone.



UNIT –V

Network Security: Traditional Cryptograph - Two Fundamental Cryptographic Principles - Secret - Key Algorithms - Public Key Algorithms.

Electronic Mail: Architecture and Services - the User Agent - Message Formats - Message Transfer - E- Mail Privacy. The World Wide Web: the Client Side - the Server Side.

Network Security

Computer Networks were primarily used by university researchers for sending email and by corporate employees for sharing printers. As millions of ordinary citizens are using networks for banking, shopping, and filing their tax returns, and weakness after weakness has been found, network security has become a problem of massive proportions. Security also deals with the problems of legitimate messages being captured and replayed, and with people later trying to deny that they sent certain messages.

Most security problems are intentionally caused by malicious people trying to gain some benefit, get attention, or harm someone. A few of the most common perpetrators are listed in Figure. It should be clear from this list that making a network secure involves a lot more than just keeping it free of programming errors. It involves outsmarting often intelligent, dedicated, and sometimes well-funded adversaries. It should also be clear that measures that will thwart casual attackers will have little impact on the serious ones. Police records show that the most damaging attacks are not perpetrated by outsiders tapping a phone line but by insiders bearing a grudge. Security systems should be designed accordingly.

Adversary	Goal
Student	To have fun snooping on people's email
Cracker	To test out someone's security system; steal data
Sales rep	To claim to represent all of Europe, not just Andorra
Corporation	To discover a competitor's strategic marketing plan
Ex-employee	To get revenge for being fired
Accountant	To embezzle money from a company
Stockbroker	To deny a promise made to a customer by email
Identity thief	To steal credit card numbers for sale
Government	To learn an enemy's military or industrial secrets
Terrorist	To steal biological warfare secrets

Fig. Some people who may cause security problems, and why

Network security problems can be divided roughly into four closely intertwined areas: secrecy, authentication, nonrepudiation, and integrity control. Secrecy, also called confidentiality, has to do with keeping information out of the grubby little hands of unauthorized users. This is what usually comes to mind when people think about network security. Authentication deals with determining whom you are talking to before revealing sensitive information or entering into a business deal. Nonrepudiation deals with signatures: how do you prove that your customer really placed an electronic order for ten million left-handed doohickies at 89 cents each when he later claims the price was 69 cents? Or maybe he claims he never placed any order. Finally, integrity control has to do with how you can be sure that a message you received was really the one sent and not something that a malicious adversary modified in transit or concocted.

In the data link layer, packets on a point-to-point line can be encrypted as they leave one machine and decrypted as they enter another. All the details can be handled in the data link layer, with higher layers oblivious to what is going on. This solution breaks down when packets have to traverse multiple routers, however, because packets have to be decrypted at each router, leaving them vulnerable to attacks from within the router. Also, it does not allow some sessions to be protected (e.g., those involving online purchases by credit card) and others not. Nevertheless, link encryption, as this method is called, can be added to any network easily and is often useful.

In the network layer, firewalls can be installed to keep good packets and bad packets out. IP security also functions in this layer.

In the transport layer, entire connections can be encrypted end to end, that is, process to process. For maximum security, end-to-end security is required.

Finally, issues such as user authentication and nonrepudiation can only be handled in the application layer.

Cryptograph

The word cryptography has come from a Greek word, which means secret writing. In the present day context it refers to the tools and techniques used to make messages secure for communication between the participants and make messages immune to attacks by hackers. For private communication through public network, cryptography plays a very crucial role.

Professionals make a distinction between ciphers and codes. A Cipher is a character-for-character or bit-for-bit transformation, without regard to the linguistic structure of the message. In contrast, a code replaces one word with another word or symbol. Codes are not used any more, although they have a glorious history. The most successful code ever devised was used by the U.S. armed forces during World War II in the Pacific.

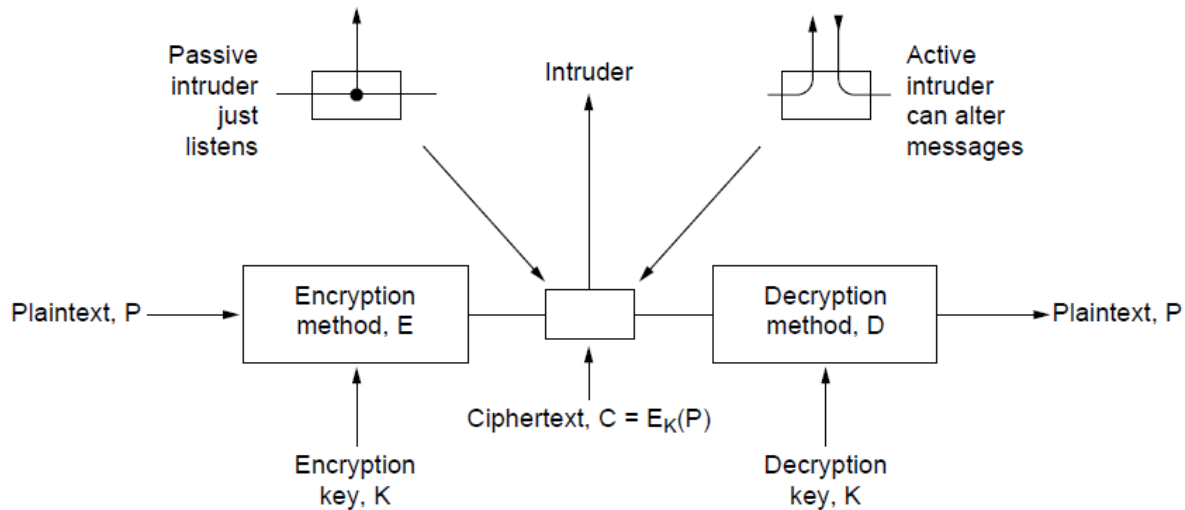


Fig. The encryption model (for a symmetric-key cipher)

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.

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). The art of breaking ciphers, known as cryptanalysis, and the art of devising them (cryptography) are collectively known as cryptology.

It will often be useful to have a notation for relating plaintext, ciphertext, and keys. We will use $C=E_K(P)$ to mean that the encryption of the plaintext P Using key K gives the ciphertext C .

Similarly, $P=D_K(C)$ represents the decryption of C to get the plaintext again. It then follows that

$$D_K(E_K(P))=P$$

This notation suggests that E and D are just mathematical functions.

Encryption methods have historically been divided into two categories: substitution ciphers and transposition ciphers.

Substitution Cipher

In a substitution cipher, each letter or group of letters is replaced by another letter or group of letters to disguise it. One of the oldest known ciphers is the Caesar cipher, attributed to Julius

Caesar. With this method, A becomes D, B becomes E, C becomes F, ..., and Z becomes C. For example, Attack becomes DWWDFN. In our examples, plaintext will be given in lowercase letters, and ciphertext in uppercase letters.

A slight generalization of the Caesar cipher allows the ciphertext alphabet to be shifted by k letters, instead of always three. In this case, k becomes a key to the general method of circularly shifted alphabets. The Caesar cipher may have fooled Pompey, but it has not fooled anyone since. The next improvement is to have each of the symbols in the plaintext, say, the 26 letters for simplicity, map onto some other letter. For example,

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

The general system of symbol-for-symbol substitution is called a monoalphabetic substitution cipher, with the key being the 26-letter string corresponding to the full alphabet.

Transposition ciphers

Substitution ciphers preserve the order of the plaintext symbols but disguise them. Transposition ciphers, in contrast, reorder the letters but do not disguise them. Figure 8-3 depicts a common transposition cipher, the columnar transposition. The cipher is keyed by a word or phrase not containing any repeated letters.

In this example, MEGABUCK is the key. The purpose of the key is to order the columns, with column 1 being under the key letter closest to the start of the alphabet, and so on. The plaintext is written horizontally, in rows, padded to fill the matrix if need be. The cipher text is read out by columns, starting with the column whose key letter is the lowest.

<u>M</u>	<u>E</u>	<u>G</u>	<u>A</u>	<u>B</u>	<u>U</u>	<u>C</u>	<u>K</u>	
<u>7</u>	<u>4</u>	<u>5</u>	<u>1</u>	<u>2</u>	<u>8</u>	<u>3</u>	<u>6</u>	
p	l	e	a	s	e	t	r	Plaintext
a	n	s	f	e	r	o	n	pleasetransferonemilliondollarsto
e	m	i	l	l	i	o	n	myswissbankaccountsixtwo
d	o	l	l	a	r	s	t	
o	m	y	s	w	i	s	s	Ciphertext
b	a	n	k	a	c	c	o	AFLLSKSOSELAWAIATOOSSCTCLNMOMANT
u	n	t	s	i	x	t	w	ESILYNTWRNNTSOWDPAEDOBUEIRICXB
o	t	w	o	a	b	c	d	

Fig. A transposition cipher

Some transposition ciphers accept a fixed-length block of input and produce a fixed-length block of output. These ciphers can be completely described by giving a list telling the order in which the characters are to be output. For example, the cipher of Fig. 8-3 can be seen as a 64 character block cipher. Its output is 4, 12, 20, 28, 36, 44, 52, 60, 5, 13, . . . , 62. In other words, the fourth input character, a, is the first to be output, followed by the twelfth, f, and so on.

Fundamental Cryptographic Principles

1. Messages must contain some redundancy
2. Some method is needed to foil replay attacks

Redundancy

The first principle is that all encrypted messages must contain some redundancy, that is, information not needed to understand the message. An example may make it clear why this is needed. Consider a mail-order company, The Couch Potato (TCP), with 60,000 products. Thinking they are being very efficient, TCP's programmers decide that ordering messages should consist of a 16-byte customer name followed by a 3-byte data field (1 byte for the quantity and 2 bytes for the product number). The last 3 bytes are to be encrypted using a very long key known only by the customer and TCP.

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, this attack no longer works because the ex-employee can no longer generate a large stream of valid messages. The moral of

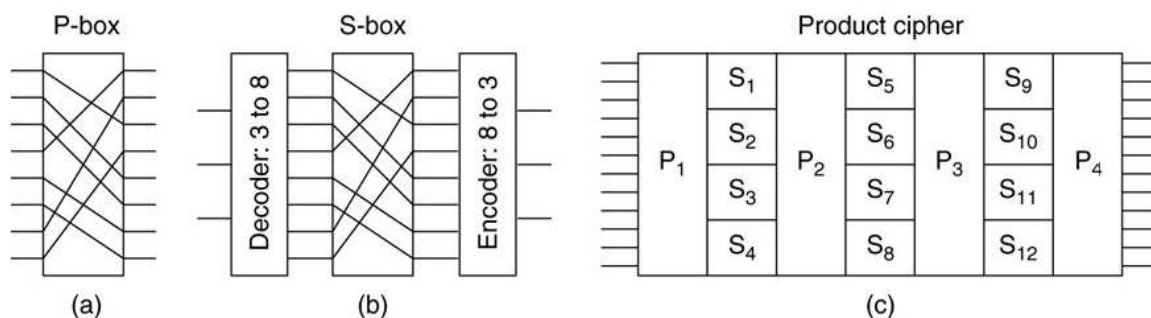
the story is that all messages must contain considerable redundancy so that active intruders cannot send random junk and have it be interpreted as a valid message.

Freshness

The second cryptographic principle is that measures must be taken to ensure that each message received can be verified as being fresh, that is, sent very recently. This measure is needed to prevent active intruders from playing back old messages. If no such measures were taken, our ex-employee could tap TCP's phone line and just keep repeating previously sent valid messages. One such measure is including in every message a timestamp valid only for, say, 10 seconds. The receiver can then just keep messages around for 10 seconds and compare newly arrived messages to previous ones to filter out duplicates. Messages older than 10 seconds can be thrown out, since any replays sent more than 10 seconds later will be rejected as too old.

Symmetric-key Algorithms

Modern cryptography uses the same basic ideas as traditional cryptography (transposition and substitution), but its emphasis is different. Traditionally, cryptographers have used simple algorithms. Nowadays, the reverse is true: the object is to make the encryption algorithm so complex and involuted that even if the cryptanalyst acquires vast mounds of enciphered text of his own choosing, he will not be able to make any sense of it at all without the key.



Basic elements of product ciphers. (a) P-box. (b) S-box. (c) Product.

Block ciphers, which take an n -bit block of plaintext as input and transform it using the key into an n -bit block of cipher text. Cryptographic algorithms can be implemented in either hardware (for speed) or software (for flexibility). Although most of our treatment concerns the algorithms and protocols, which are independent of the actual implementation, a few words about building cryptographic hardware may be of interest. Transpositions and substitutions can be implemented

with simple electrical circuits. Figure (a) shows a device, known as a P-box (P stands for permutation), used to effect a transposition on an 8-bit input. If the 8 bits are designated from top to bottom as 01234567, the output of this particular P-box is 36071245. By appropriate internal wiring, a P-box can be made to perform any transposition and do it at practically the speed of light since no computation is involved, just signal propagation. This design follows Kerckhoff's principle: the attacker knows that the general method is permuting the bits.

In this example, a 3-bit plaintext is entered and a 3-bit ciphertext is output. The 3-bit input selects one of the eight lines exiting from the first stage and sets it to 1; all the other lines are 0. The second stage is a P-box. The third stage encodes the selected input line in binary again. With the wiring shown, if the eight octal numbers 01234567 were input one after another, the output sequence would be 24506713.

DES (THE DATA ENCRYPTION STANDARD)

In January 1977, the U.S. Government adopted a product cipher developed by IBM as its official standard for unclassified information. This cipher, DES(Data Encryption Standard), was widely adopted by the industry for use in security products. It is no longer secure in its original form, but in a modified form it is still useful. Plaintext is encrypted in blocks of 64 bits, yielding 64 bits of ciphertext. The algorithm, which is parameterized by a 56-bit key, has 19 distinct stages. The first stage is a key-independent transposition on the 64-bit plaintext. The last stage is the exact inverse of this transposition. The stage prior to the last one exchanges the leftmost 32 bits with the rightmost 32 bits. The remaining 16 stages are functionally identical but are parameterized by different functions of the key. The algorithm has been designed to allow decryption to be done with the same key as encryption, a property needed in any symmetric-key algorithm. The steps are just run in the reverse order.

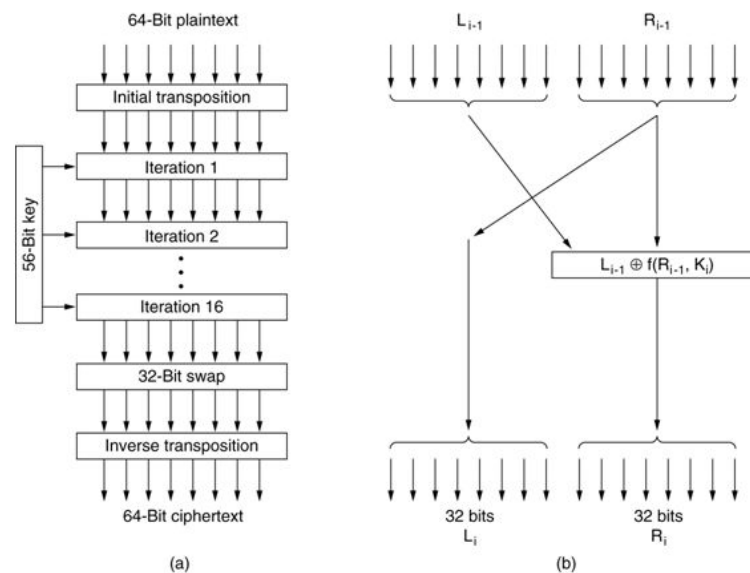
The operation of one of these intermediate stages is illustrated in Fig.(b). Each stage takes two 32-bit inputs and produces two 32-bit outputs. The left output is simply a copy of the right input. The right output is the bitwise XOR of the left input and a function of the right input and the key for this stage, K_i . Pretty much all the complexity of the algorithm lies in this function.

The function consists of four steps, carried out in sequence. First, a 48-bit number, E , is constructed by expanding the 32-bit R_{i-1} according to a fixed transposition and duplication rule. Second, E and K_i are XORed together. This output is then partitioned into eight groups of 6 bits

each, each of which is fed into a different S-box. Each of the 64 possible inputs to an S-box is mapped onto a 4 bit output. Finally, these 8×4 bits are passed through a P-box.

In each of the 16 iterations, a different key is used. Before the algorithm starts, a 56-bit transposition is applied to the key. Just before each iteration, the key is partitioned into two 28-bit units, each of which is rotated left by a number of bits dependent on the iteration number. K_i is derived from this rotated key by applying yet another 56-bit transposition to it. A different 48-bit subset of the 56 bits is extracted and permuted on each round. A technique that is sometimes used to make DES stronger is called whitening. It consists of XORing a random 64-bit key with each plaintext block before feeding it into DES and then XORing a second 64-bit key with the resulting ciphertext before transmitting it.

Data Encryption Standard



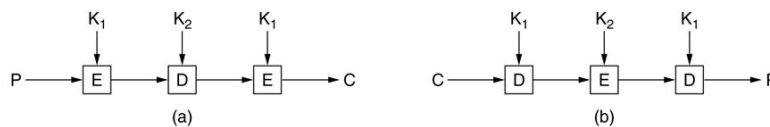
The data encryption standard. (a) General outline.
(b) Detail of one iteration. The circled + means exclusive OR.

Triple DES

As early as 1979, IBM realized that the DES key length was too short and devised a way to effectively increase it, using triple encryption (Tuchman, 1979). The method chosen, which has

since been incorporated in International Standard 8732, is illustrated in Fig. 8-8. Here, two keys and three stages are used. In the first stage, the plaintext is encrypted using DES in the usual way with K_1 . In the second stage, DES is run in decryption mode, using K_2 as the key. Finally, another DES encryption is done with K_1 .

Triple DES



(a) Triple encryption using DES. (b) Decryption.

AES (THE ADVANCED ENCRYPTION STANDARD)

As DES began approaching the end of its useful life, even with triple DES, **NIST**(National Institute of Standards and Technology), the agency of the U.S. Dept. of Commerce charged with approving standards for the U.S. Federal Government, decided that the government needed a new cryptographic standard for unclassified use.

In January 1997, researchers from all over the world were invited to submit proposals for a new standard, to be called AES(Advanced Encryption Standard). The bake-off rules were:

1. The algorithm must be a symmetric block cipher.
2. The full design must be public.
3. Key lengths of 128, 192, and 256 bits must be supported.
4. Both software and hardware implementations must be possible.
5. The algorithm must be public or licensed on nondiscriminatory terms.

In October 2000, NIST announced that it had selected Rijndael, by Joan Daemen and Vincent Rijmen. The name Rijndael, pronounced Rhine-doll (more or less), is derived from the last names of the authors: Rijmen + Daemen. In November 2001, Rijndael became the AES U.S. Government standard, published as FIPS (Federal Information Processing Standard) 197.

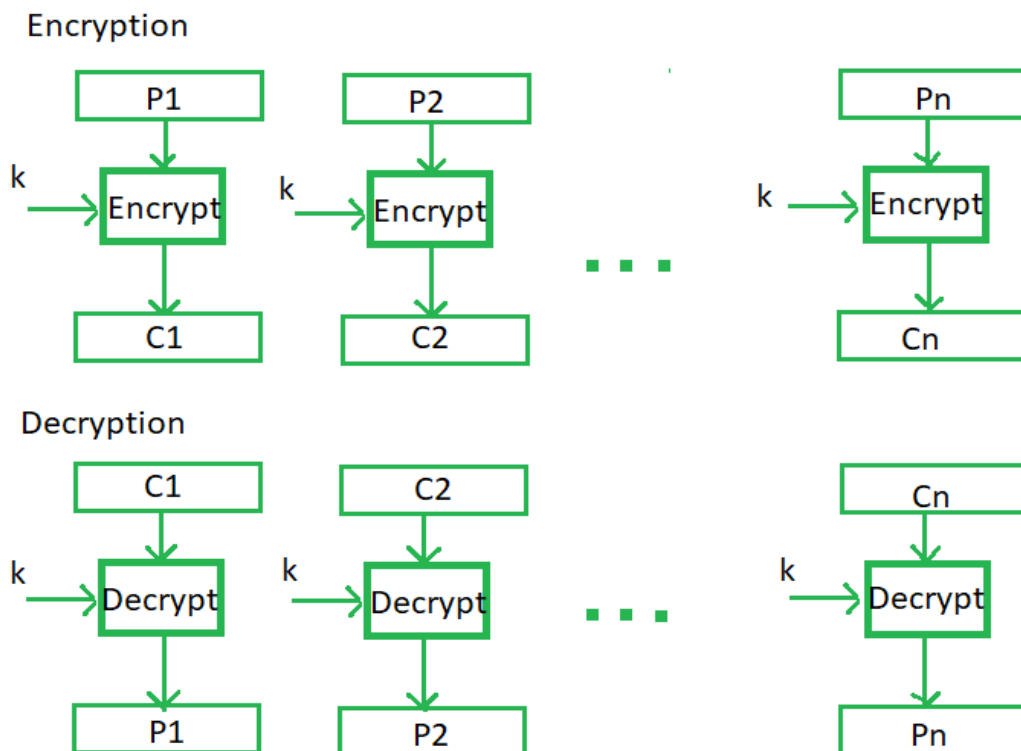
Rijndael supports key lengths and block sizes from 128 bits to 256 bits in steps of 32 bits. The key length and block length may be chosen independently. AES specifies that the block size must be 128 bits and the key length must be 128, 192, or 256 bits. It is doubtful that anyone will ever use 192-bit keys, so de facto, AES has two variants: a 128-bit block with a 128-bit key and a 128-bit block with a 256-bit key.

Cipher Modes

AES is basically a monoalphabetic substitution cipher using big characters (128-bit characters for ES and 64-bit characters for DES). Whenever the same plaintext block goes in the front end, the same ciphertext block comes out the back end. If you encrypt the plaintext abcdefgh 100 times with the same DES key, you get the same ciphertext 100 times. An intruder can exploit this property to help subvert the cipher.

Electronic Code Book (ECB) –

Electronic code book is the easiest block cipher mode of functioning. It is easier because of direct encryption of each block of input plaintext and output is in form of blocks of encrypted ciphertext. Generally, if a message is larger than b bits in size, it can be broken down into bunch of blocks and the procedure is repeated.



Advantages of using ECB

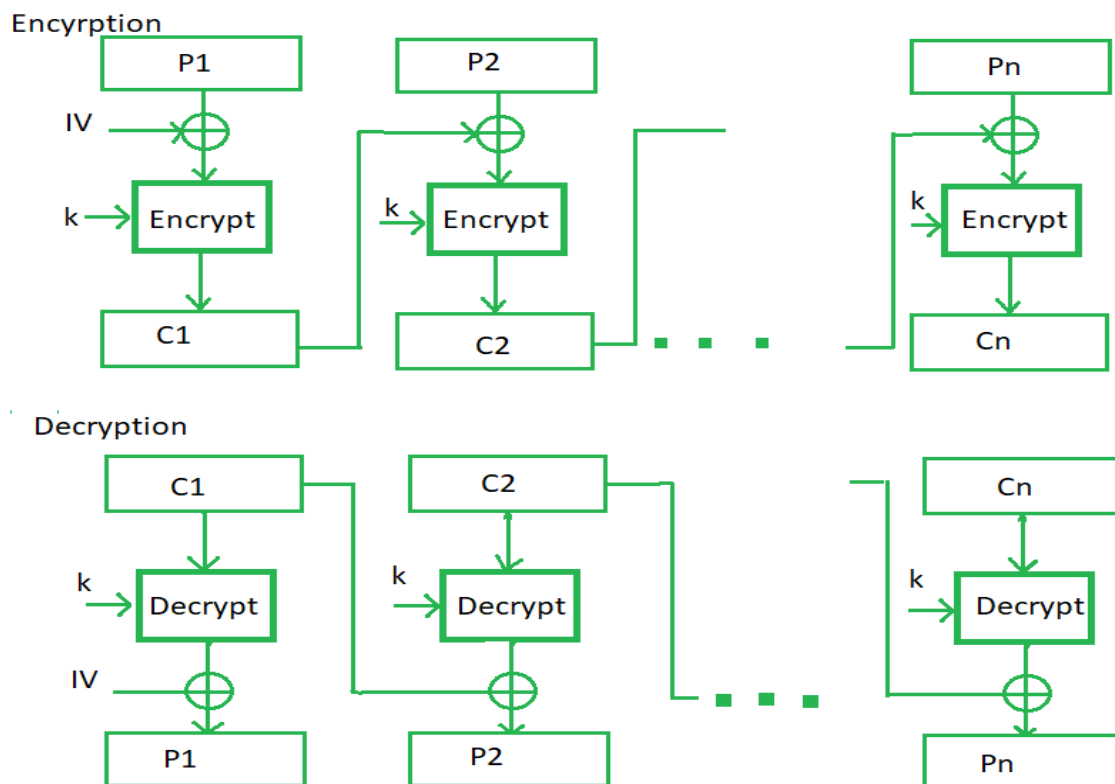
- Parallel encryption of blocks of bits is possible, thus it is a faster way of encryption.
- Simple way of block cipher.

Disadvantages of using ECB

- Prone to cryptanalysis since there is a direct relationship between plaintext and ciphertext.

Cipher Block Chaining

Cipher block chaining or CBC is an advancement made on ECB since ECB compromises some security requirements. In CBC, previous cipher block is given as input to next encryption algorithm after XOR with original plaintext block. In a nutshell here, a cipher block is produced by encrypting a XOR output of previous cipher block and present plaintext block.



Advantages of CBC –

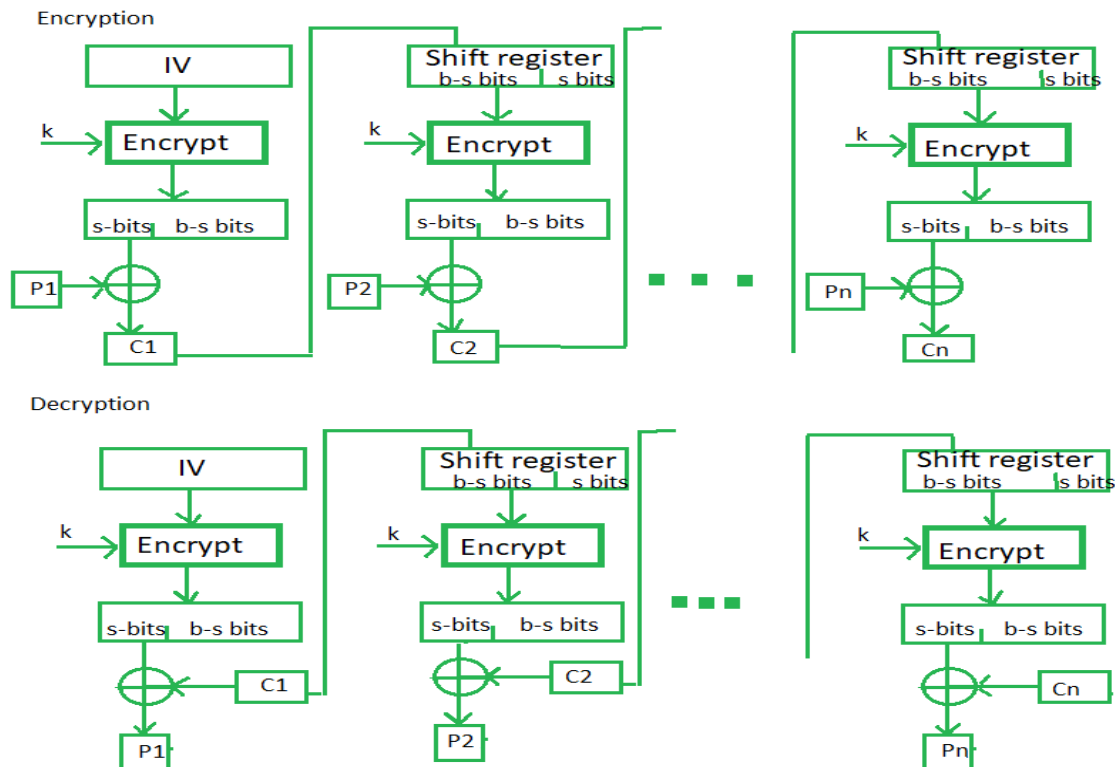
- CBC works well for input greater than b bits.
- CBC is a good authentication mechanism.
- Better resistive nature towards cryptanalysis than ECB.

Disadvantages of CBC –

- Parallel encryption is not possible since every encryption requires previous cipher.

Cipher Feedback Mode (CFB)

In this mode the cipher is given as feedback to the next block of encryption with some new specifications: first an initial vector IV is used for first encryption and output bits are divided as set of s and $b-s$ bits the left hand side s bits are selected and are applied an XOR operation with plaintext bits. The result given as input to a shift register and the process continues. The encryption and decryption process for the same is shown below, both of them use encryption algorithm.



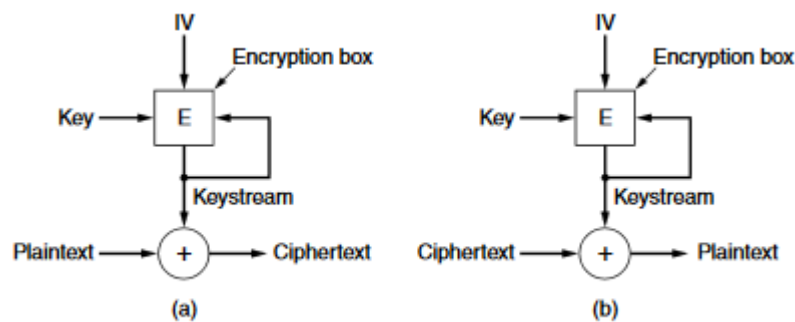
Advantages of CFB –

- Since, there is some data loss due to use of shift register, thus it is difficult for applying cryptanalysis.

Stream Cipher Mode

Applications exist in which having a 1-bit transmission error mess up 64 bits of plaintext is too large an effect. For these applications, a fourth option, stream cipher mode, exists. It works by encrypting an initialization vector, using a key to get an output block. The output block is then encrypted, using the key to get a second output block. This block is then encrypted to get a third block, and so on.

The (arbitrarily large) sequence of output blocks, called the keystream, is treated like a one-time pad and XORed with the plaintext to get the ciphertext, as shown in Fig



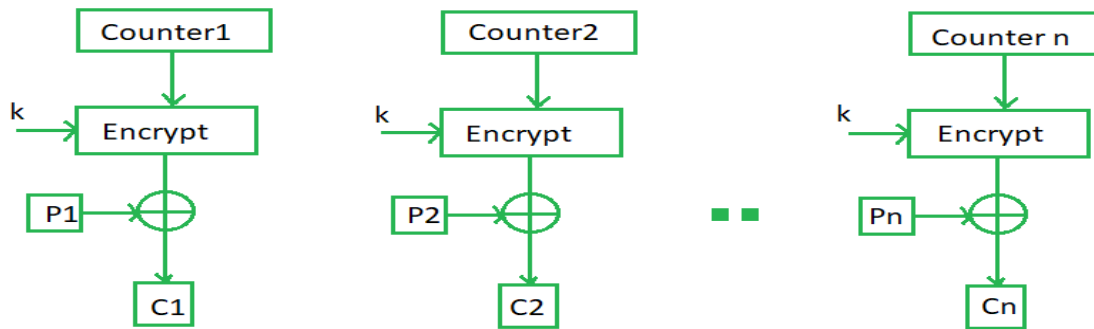
A stream cipher. (a) Encryption. (b) Decryption.

Decryption occurs by generating the same keystream at the receiving side. Since the keystream depends only on the IV and the key, it is not affected by transmission errors in the ciphertext. Thus, a 1-bit error in the transmitted ciphertext generates only a 1-bit error in the decrypted plaintext.

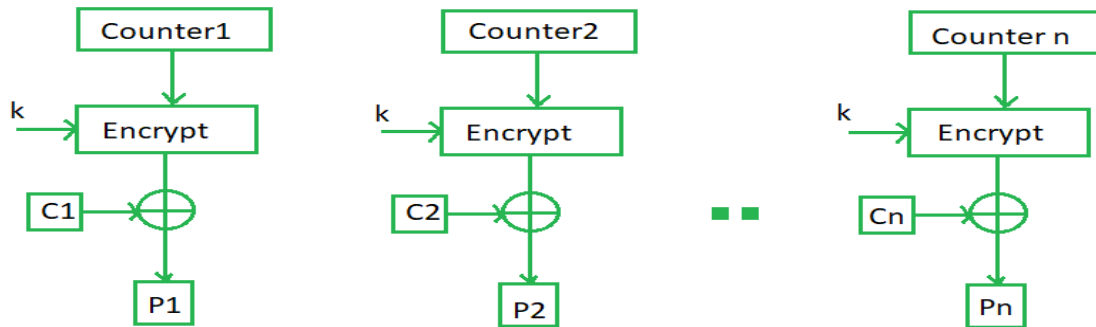
Counter Mode

The Counter Mode or CTR is a simple counter based block cipher implementation. Every time a counter initiated value is encrypted and given as input to XOR with plaintext which results in ciphertext block. The CTR mode is independent of feedback use and thus can be implemented in parallel.

Encryption



Decryption



Others Cipher

AES (Rijndael) and DES are the best-known symmetric-key cryptographic algorithms, and the standard industry choices, if only for liability reasons.

Cipher	Author	Key length	Comments
DES	IBM	56 bits	Too weak to use now
RC4	Ronald Rivest	1–2048 bits	Caution: some keys are weak
RC5	Ronald Rivest	128–256 bits	Good, but patented
AES (Rijndael)	Daemen and Rijmen	128–256 bits	Best choice
Serpent	Anderson, Biham, Knudsen	128–256 bits	Very strong
Triple DES	IBM	168 bits	Good, but getting old
Twofish	Bruce Schneier	128–256 bits	Very strong; widely used

Cryptanalysis

Before leaving the subject of symmetric-key cryptography, it is worth at least mentioning four developments in cryptanalysis. The first development is differential cryptanalysis (Biham and Shamir, 1997). This technique can be used to attack any block cipher. It works by beginning with

a pair of plaintext blocks differing in only a small number of bits and watching carefully what happens on each internal iteration as the encryption proceeds.

The second development worth noting is linear cryptanalysis (Matsui, 1994). It can break DES with only 2^{43} known plaintexts. It works by XORing certain bits in the plaintext and ciphertext together and examining the result.

The third development is using analysis of electrical power consumption to find secret keys. Computers typically use around 3 volts to represent a 1 bit and 0 volts to represent a 0 bit. Thus, processing a 1 takes more electrical energy than processing a 0.

The fourth development is timing analysis. Cryptographic algorithms are full of if statements that test bits in the round keys. If the then and else parts take different amounts of time, by slowing down the clock and seeing how long various steps take, it may also be possible to deduce the round keys.

Public-Key Algorithms

Distributing the keys has always been the weakest link in most cryptosystems. No matter how strong a cryptosystem was, if an intruder could steal the key, the system was worthless. Cryptologists always took for granted that the encryption key and decryption key were the same (or easily derived from one another). But the key had to be distributed to all users of the system. Thus, it seemed as if there was an inherent problem. Keys had to be protected from theft, but they also had to be distributed, so they could not be locked in a bank vault. In 1976, two researchers at Stanford University, Diffie and Hellman (1976), proposed a radically new kind of cryptosystem, one in which the encryption and decryption keys were so different that the decryption key could not feasibly be derived from the encryption key. In their proposal, the (keyed) encryption algorithm, E , and the (keyed) decryption algorithm, D , had to meet three requirements. These requirements can be stated simply as follows:

1. $D(E(P)) = P$.
2. It is exceedingly difficult to deduce D from E .
3. E cannot be broken by a chosen plaintext attack.

The first requirement says that if we apply D to an encrypted message, $E(P)$, we get the original plaintext message, P , back. Without this property, the legitimate receiver could not decrypt the

ciphertext. The second requirement speaks for itself. The third requirement is needed because, as we shall see in a moment, intruders may experiment with the algorithm to their hearts' content. The method works like this. A person, say, Alice, who wants to receive secret messages, first devises two algorithms meeting the above requirements. The encryption algorithm and Alice's key are then made public, hence the name public-key cryptography.

RSA

The only catch is that we need to find algorithms that indeed satisfy all three requirements. Due to the potential advantages of public-key cryptography, many researchers are hard at work, and some algorithms have already been published. One good method was discovered by a group at M.I.T. (Rivest et al., 1978). It is known by the initials of the three discoverers (Rivest, Shamir, Adleman): RSA. It has survived all attempts to break it for more than 30 years and is considered very strong. Much practical security is based on it. For this reason, Rivest, Shamir, and Adleman were given the 2002 ACM Turing Award. 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 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 $e \times d \equiv 1 \pmod{z}$

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 $0 \leq P < n$. 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 $C = P^e \pmod{n}$. To decrypt C , compute $P = C^d \pmod{n}$. 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) .

For this example, we have chosen $p=3$ and $q=11$, giving $n=33$ and $z=20$. A suitable value for d is $d=7$, since 7 and 20 have no common factors. With these choices, e can be found by solving

the equation $7e \equiv 1 \pmod{20}$, which yields $e=3$. The ciphertext, C , corresponding to a plaintext message, P , is given by $C = P^3 \pmod{33}$. The ciphertext is decrypted by the receiver by making use of the rule $P = C^7 \pmod{33}$. The figure shows the encryption of the plaintext “SUZANNE” as an example

Plaintext (P)		Ciphertext (C)			After decryption	
Symbolic	Numeric	P^3	$P^3 \pmod{33}$	C^7	$C^7 \pmod{33}$	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		

An example of the RSA algorithm

E-MAIL

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:

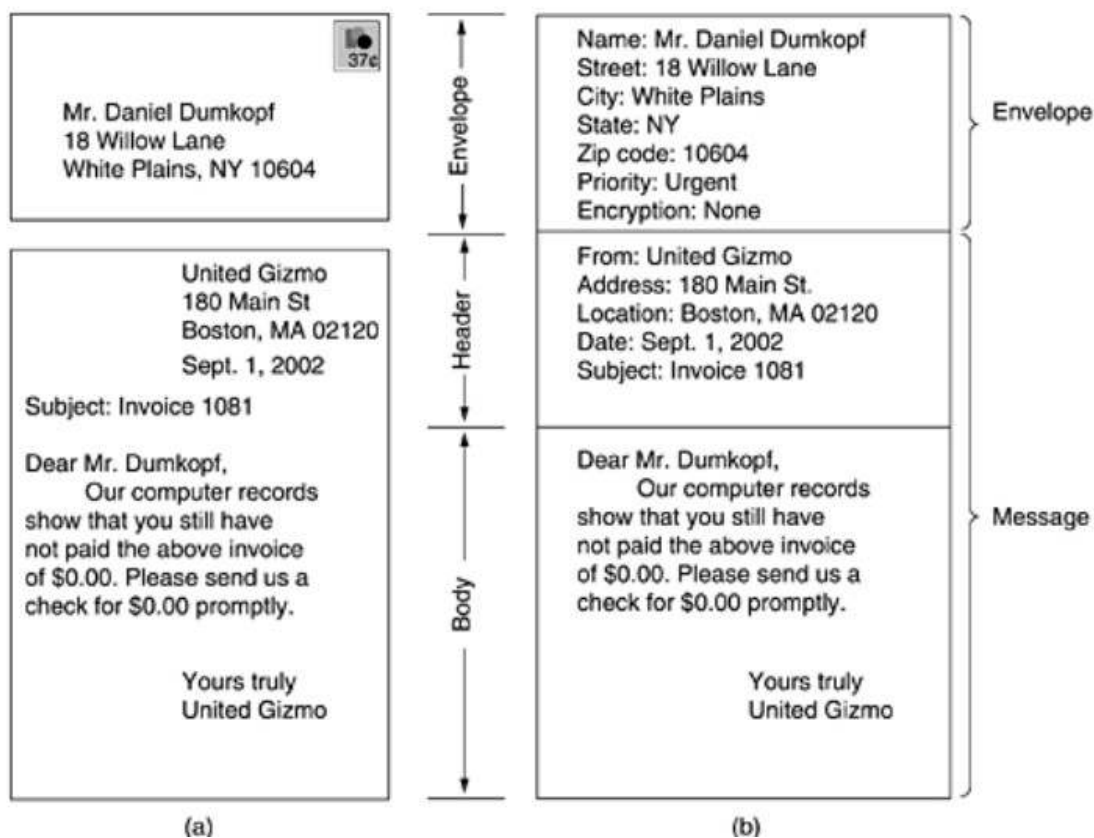
It refers to moving messages from originator to the recipient.

(d). Displaying:

Incoming messages are to be displayed so that people can read their email.

(e). Disposition:

It concerns what the recipient does 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.



Envelopes and messages. (a) Paper mail. (b) Electronic mail

THE USER AGENT

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.

SENDING E-MAIL

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`.

READING E-MAIL

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

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

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

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.

WORLD WIDE WEB (WWW)

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.

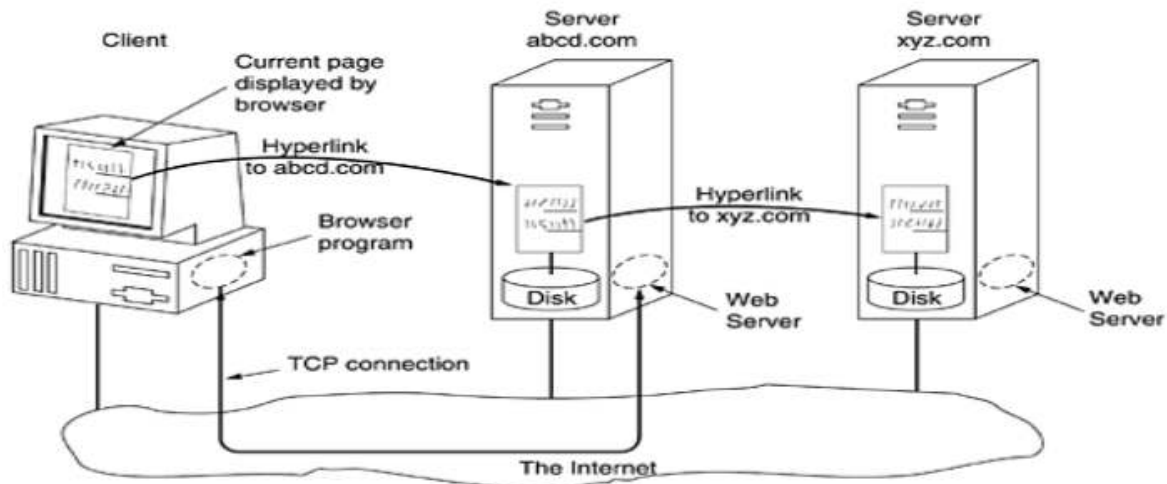
ARCHITECTURAL OVERVIEW

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).

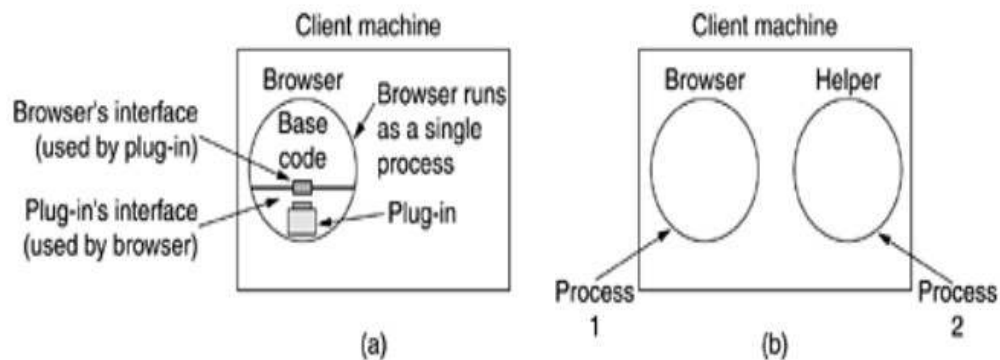
The steps that occur at the client side are:

- 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.



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

SERVER SIDE

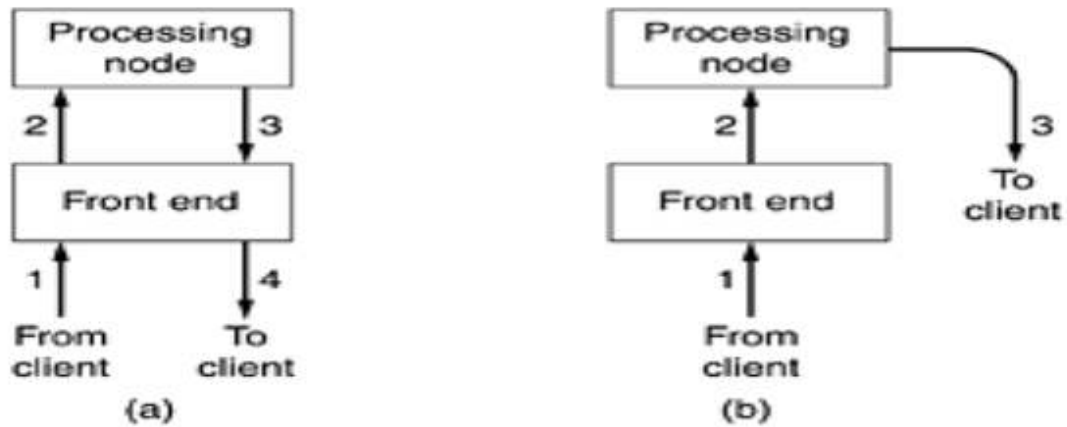
The steps to be followed by the server side are:

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.



(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.