

# Computer Networks (IT 3001)

Prof. Amit Jha

School of Electronics Engineering  
KIIT University, Bhubaneswar



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Syllabus

- **Go through the attachments uploaded on the website.**

# Course Materials

- ***Text Book:***

1. COMPUTER NETWORKS: A Top-Down Approach by Behrouz A Forouzan, Firouz Mosharraf

- ***Reference Book:***

1. Computer Networks 5th edition by Tannenbaum
2. Computer networks An Open source approach by Ying-Dar Lin
3. Computer Networks A systems approach by Peterson and Daive
4. Computer Networking: A top-down approach by Kurose and Ross

# Marking Schemes

2 Quizzes, 4 Assignments and 1 Activity Based Learning	Mid-Sem	End-Sem
30	20	50

- If attendance <75% **Debar**
- **No assignments will be accepted beyond the due date**
- ***Notebook maintenance is mandatory.***



# Motivation for the Course

- Three main reasons for studying computer networking
- **1. You don't have to be a math wiz:** Networking starts with basic logic and connections, **only needs interests!!!**
- **2. Every workplace needs a few friendly geeks:**  
Networking skills give you an edge and an opportunity to make a career in almost any sector you can imagine: financial services, education, transportation, manufacturing, oil and gas, mining and minerals, technology, government, hospitality, health care, retail... you name it.
- **3. Opportunities abound:** Some well known companies

# Well known Firms

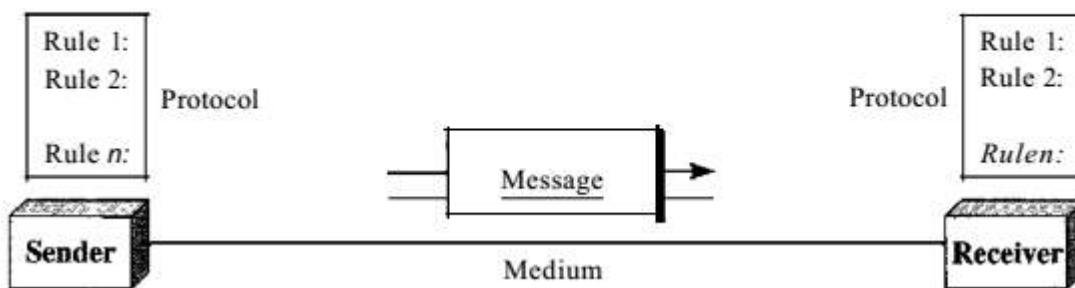


# Why to take this course seriously?

- Cisco Certified Network Associate CCNA
- Cisco Certified Network Professional CCNP
- Juniper Networks Certified Internet Associate JNCIA
- Comptia Network+

# What is CN and why needed?

- **Computer Networking:** The process by which two computers from same or different manufacturers are communicated is called Computer Networking.



## 5 Components of Networking

**Understanding of CN is needed** in order to connect multiple computers together so that data can be exchanged among themselves reliably.

**What is protocol and why it is needed?** Protocol is set of rules necessary to communicate the data between two devices from different manufacturers.

*E.g.* A Sony Vio laptop can use processor from intel, Qualcomm, AMD, etc.

## Data flow types

**Simplex:** Unidirectional communication like one way street.

**Half-Duplex:** Both directional but not at the same time.

**Full-duplex:** Both directional and at the same time like two way road.

# Challenges in Computer Networking

- Data format and size, and from what type of sensors or equipment;
- How often the data needs to be transmitted;
- The transmission environment and overall distance it has to travel
- The type(s) of terrain and foliage the signal will encounter;
- The data communication protocol used;
- Available radio frequency choices available for each site and the related signal strength;
- Selecting the optimal combination of radios and antennas;
- The position and mounting of the antennas.

- Do we cover all these in this course?



- Do we cover all these in this course?
- NO.....



- We deal with only “**the networking parts, protocols and standards**”



# What is Internet?

Internet is a collection of networks or network of networks. Various networks such as LAN and WAN connected through suitable hardware and software to work in a **seamless manner**. It allows various applications such as e-mail, file transfer, remote login, World Wide Web, Multimedia, etc run across the internet.

# The Internet Today

- Born in 1960 and has come a long way since then.
- Today run by private companies, not the government.
- Internet service provider (ISP) is backbone which provides the internet to end user.
- Number of people using it approx. 42.4% of total world population (world population more than 7 billion till march 2012)

# Advantages of the Internet

Voice over IP, Video on demand, Network for the people, Commercial, Scientific and technical computing, Teleconferencing , Electronic mail, Electronic Data Interchange, Information Services, Marketing & Sales, Financial Services, Manufacturing , Directory services



It has everything for everyone and everywhere !!!!!!

# Some bad results of the Internet

*The Day That Albert Einstein Feared Has Arrived!*



*Having coffee with frens*



*A day in a beach*



*Cheering your team*



*Out on an intimate date*



*Enjoying the sights*



*Having dinner*



*"I fear the day that technology will surpass our human interaction. The world will have a generation of idiots"*

*Albert Einstein*

# Communication System

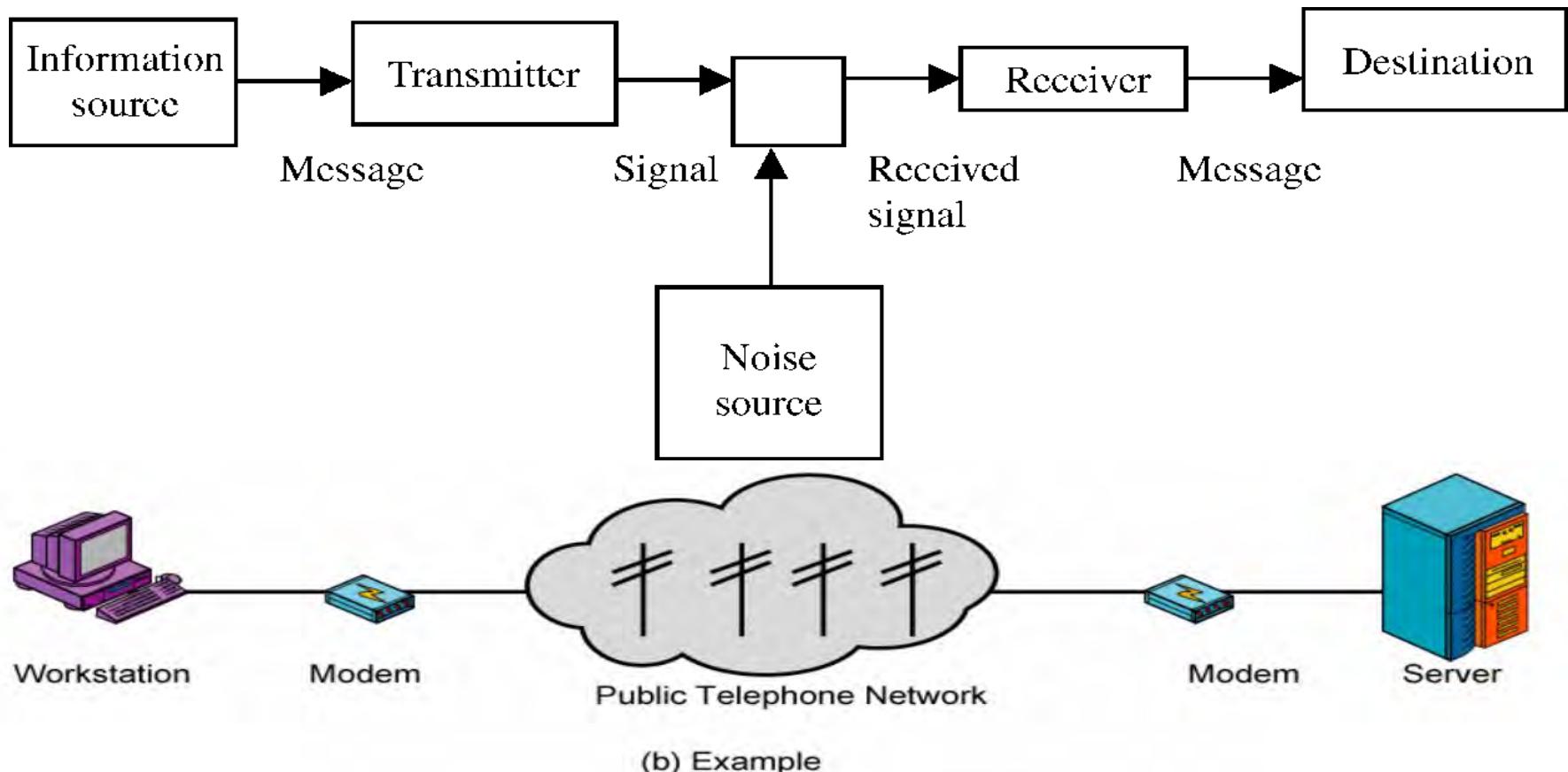
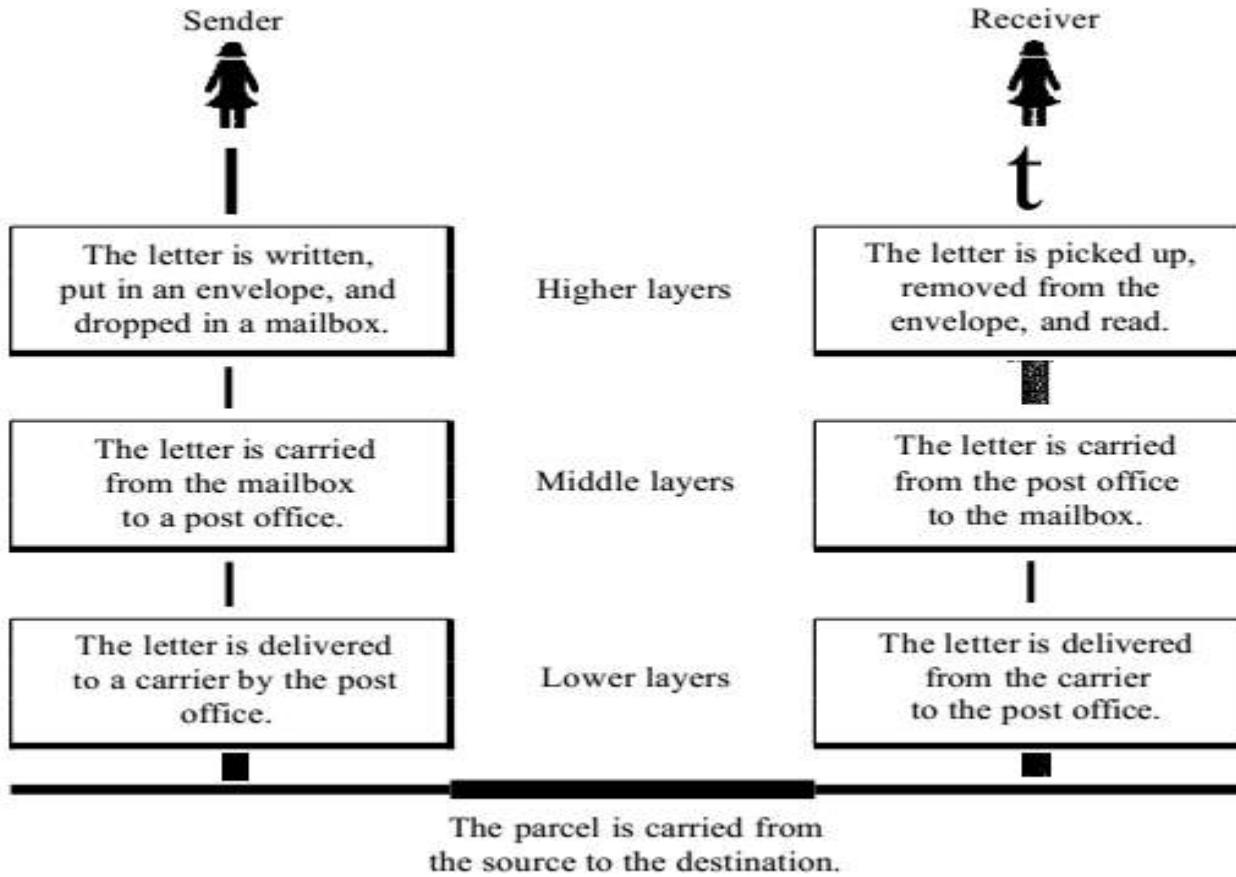


Figure 1.1 Simplified Communications Model

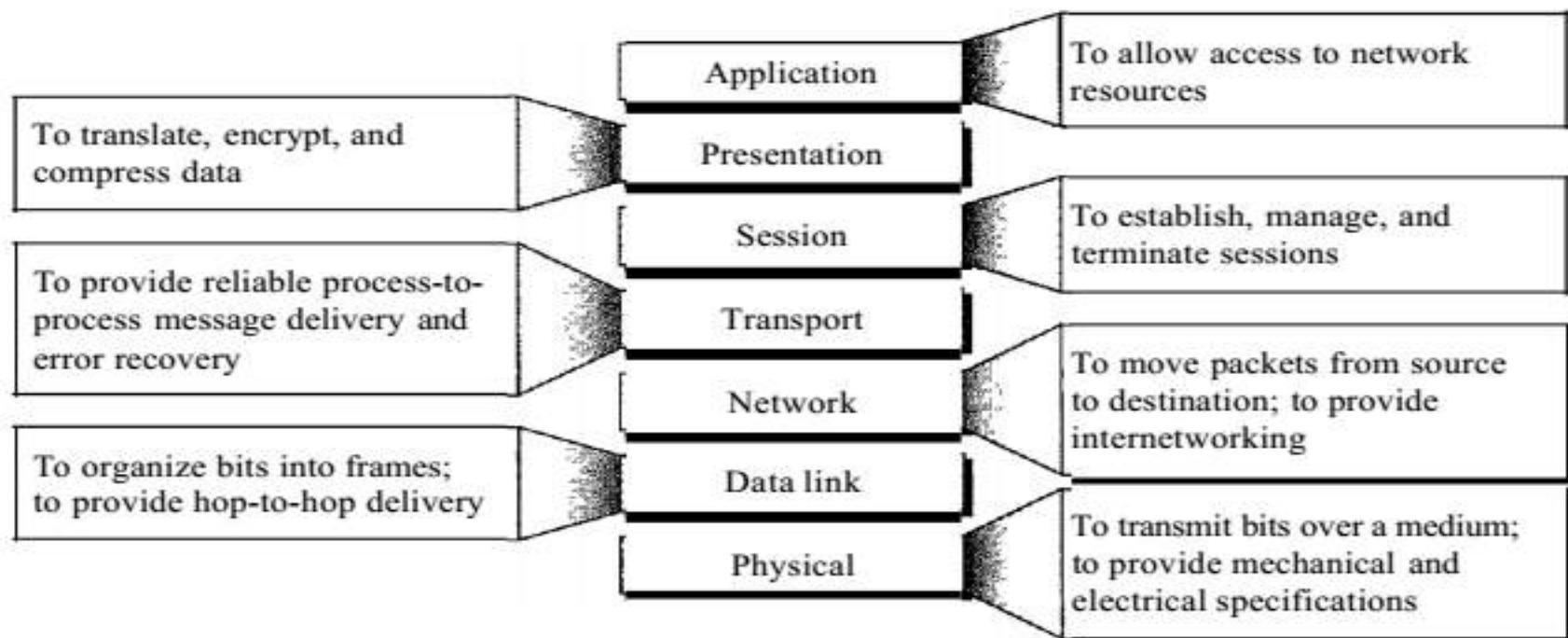
# Layered Approach: What & Why?



# OSI model

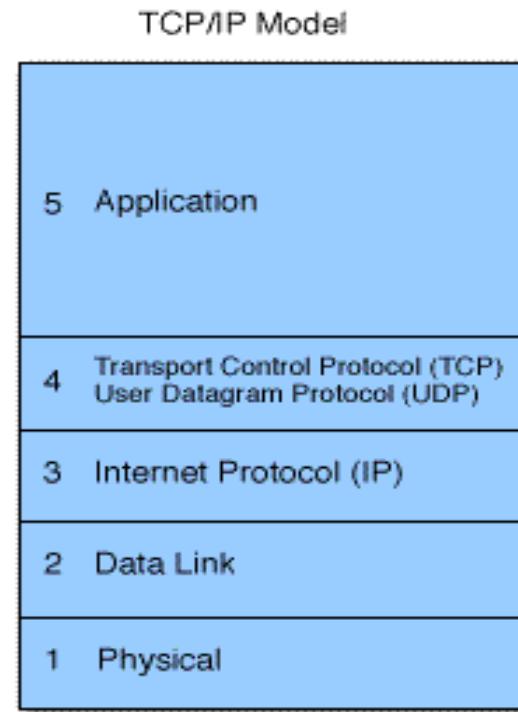
- OSI stands for open system for interconnection. It is a proposed model.
- It is not a protocol; it is a model for understanding and designing a network architecture that is flexible, robust, and interoperable.
- It has 7 layers.

## Functions of each layer



# TCP/IP Protocol Suite

- Stands for Transmission Control Protocol/Internet Protocol. It is implemented model and designed prior to OSI.



# CN (IT-3001)

## Introduction

Prof. Amit Jha  
School of Electronics Engineering (SOEE)  
KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Content

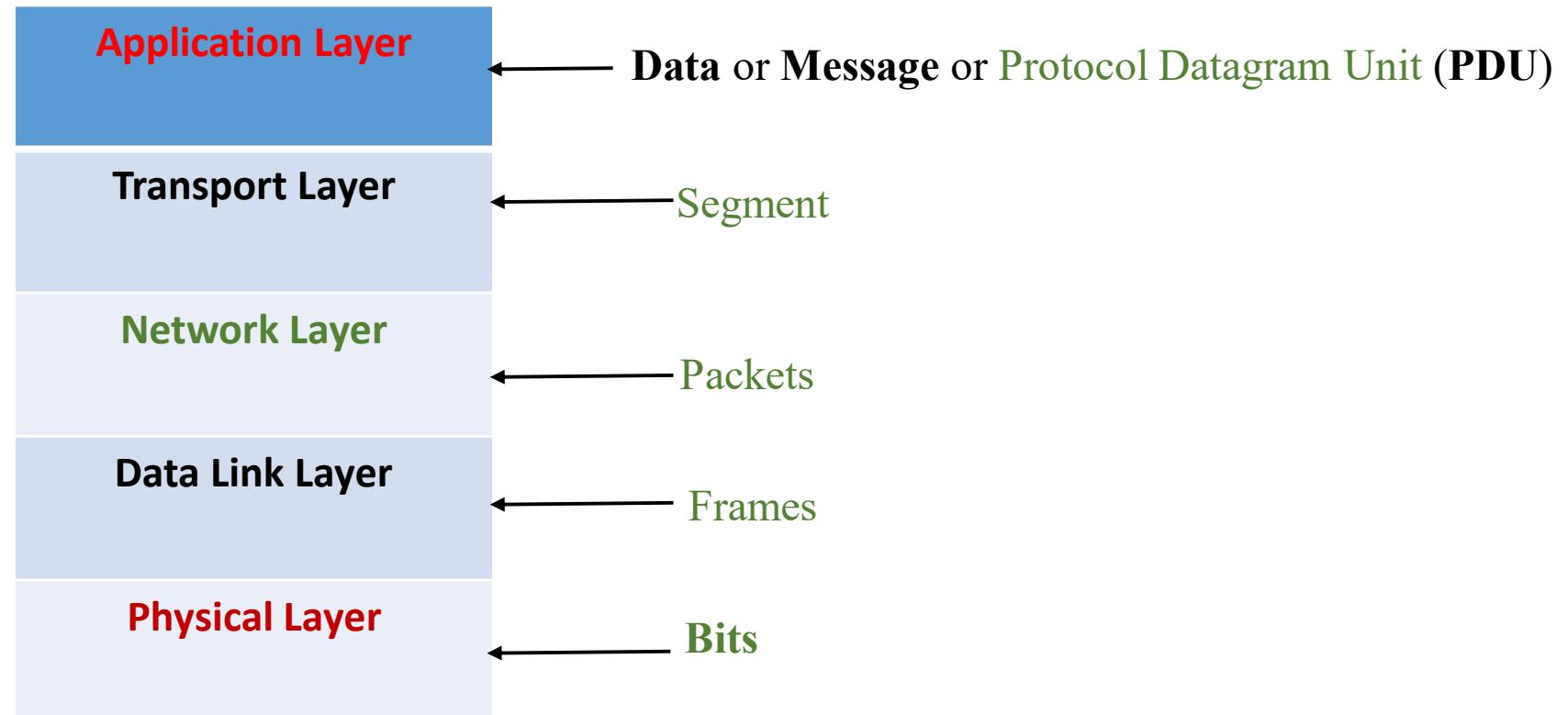
- Basics of TCP/IP Suite
  - Different naming to message and its reason
  - Different addressing Mechanism and its reason
  - Different protocols at different layers
  - Logical connection between different layers
  - Encapsulation and Decapsulation

# TCP/IP Protocol Suite

- It is a protocol suite used in the Internet today.
- It is a hierarchical protocol made up of interactive modules, each of which provides a specific functionality.
- The term hierarchical means that each upper level protocol is supported by the services provided by one or more lower level protocols.
- The original TCP/IP protocol suite was defined as four software layers built upon the hardware.
- Today, however, TCP/IP is thought of as a five-layer model.
- Protocols: A network protocol defines rules and conventions for communication between network devices. Network protocols include mechanisms for devices to identify and make connections with each other, as well as formatting rules that specify how data is packaged into sent and received messages.
- A set of cooperating network protocols is called a protocol suite. The TCP/IP suite includes numerous protocols across layers -- such as the data, network, transport and application layers -- working together to enable internet connectivity

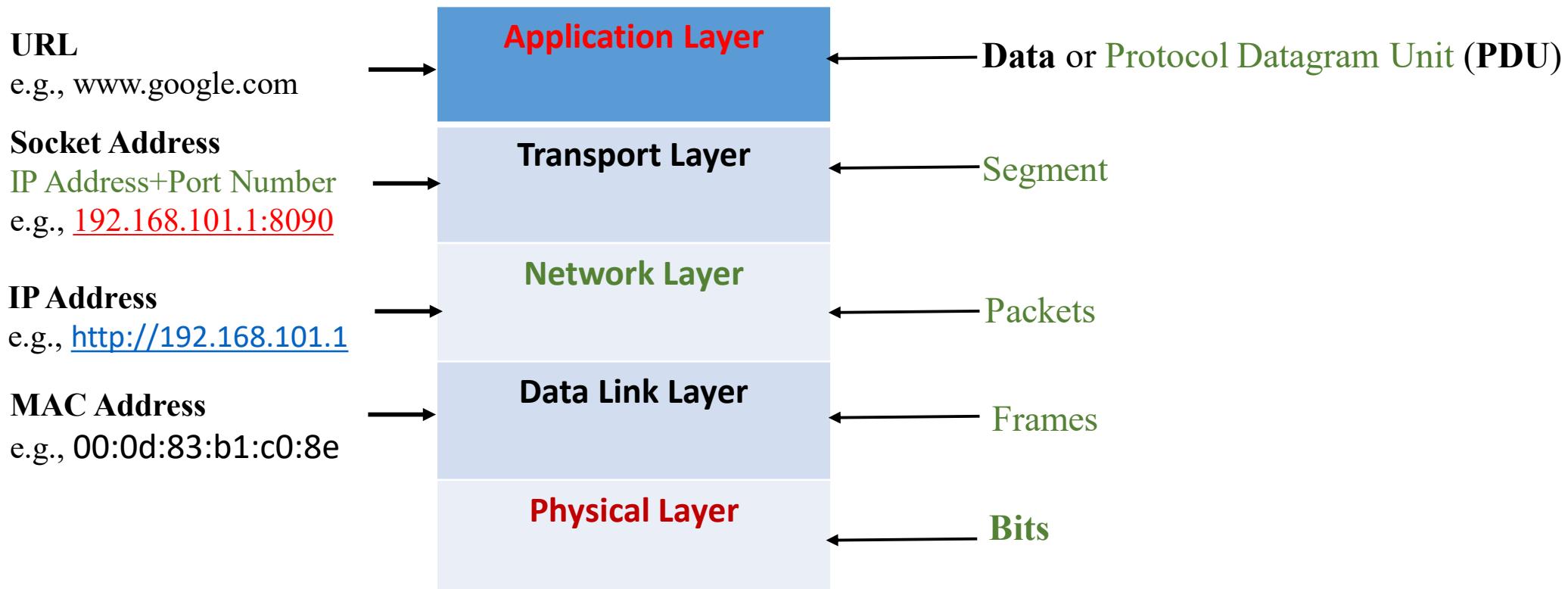
# TCP/IP: Nomenclature of the Data

- We use different names for the data or information to each layers of TCP/IP Stack.



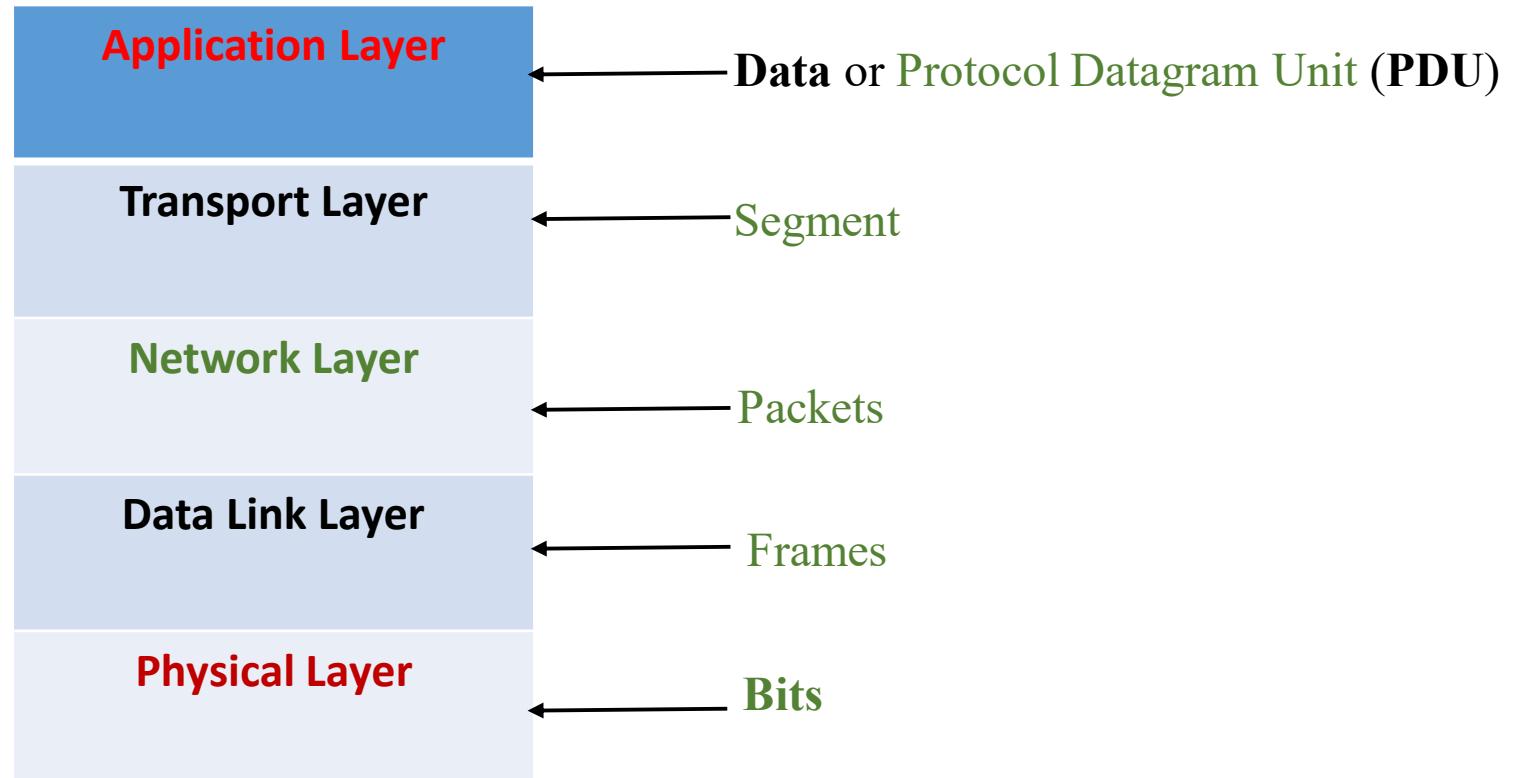
# TCP/IP: Addressing at Different Layers

- We use different names for the data or information to each layers of TCP/IP Stack.

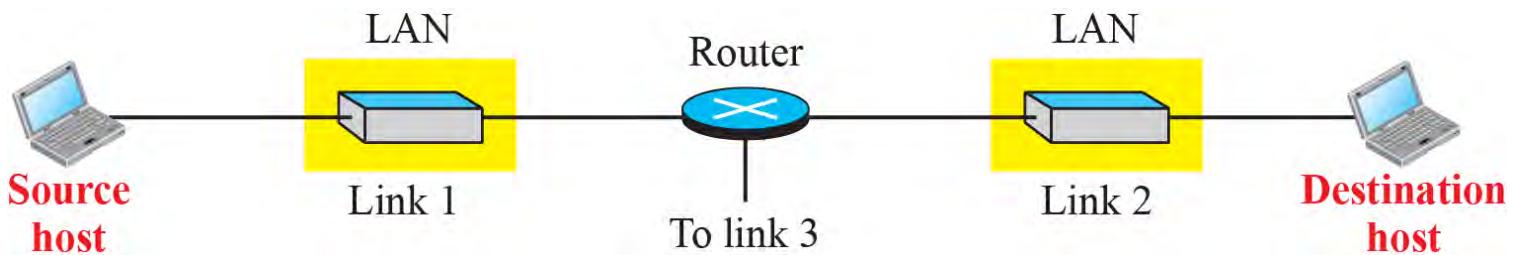
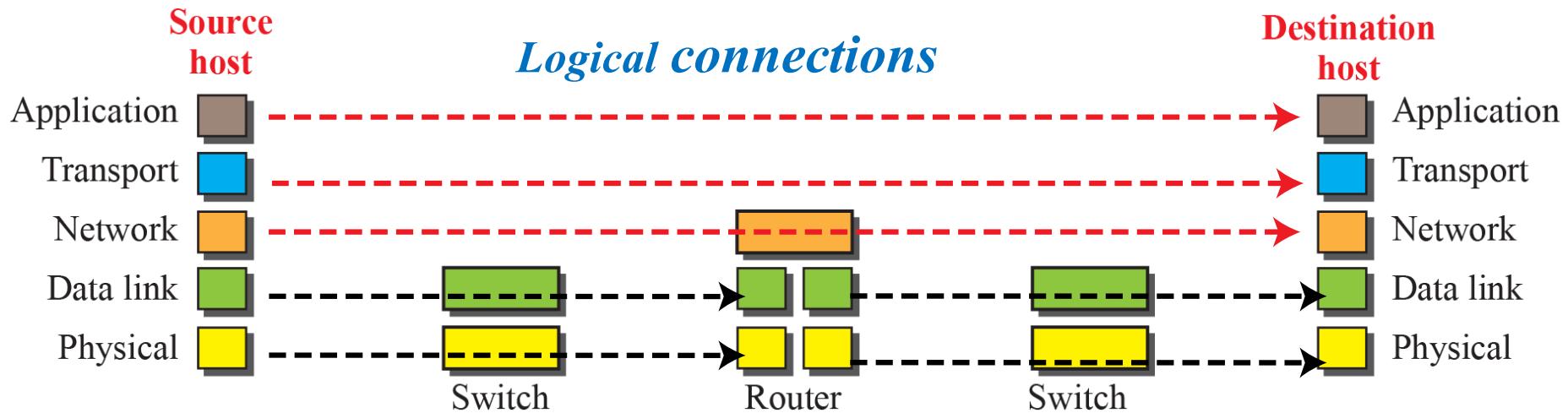


# TCP/IP: Protocols

- We use different names for the data or information to each layers of TCP/IP Stack.



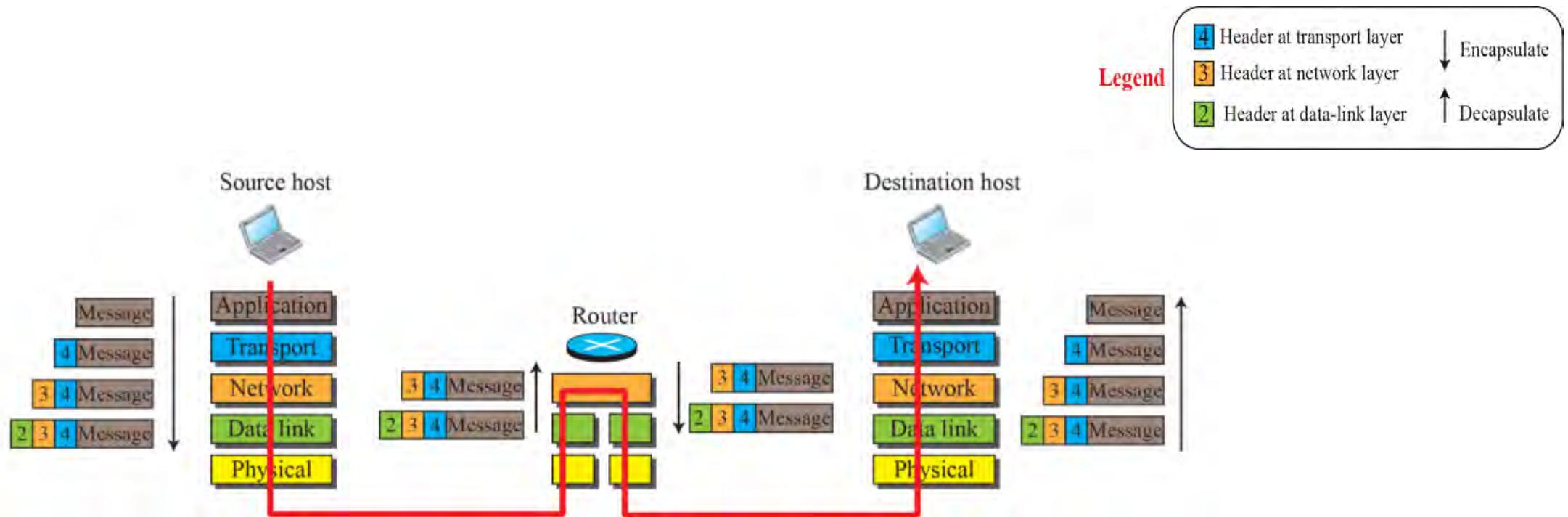
# TCP/IP: Logical Connections Between Different Layers



# Description

- Logical connections makes it easier for us to think about the duty of each layer.
- From logical connections, we can observe that
  - Duty of application, transport and network layer is *end-to-end*.
  - However, duty of data link layer and physical layer is *hop-to-hop*.

# TCP/IP: Encapsulation and Decapsulation



# TCP/IP: Encapsulation and Decapsulation

- **Encapsulation:**

- The term encapsulation is used to describe a process of adding headers and trailers around some data.
- Encapsulation is the process of taking data from one protocol and translating it into another protocol, so the data can continue across a network.
- The lower layer encapsulates the higher layer's data between a header (Data Link protocols also add a trailer).
- For example, a TCP/IP packet contained within an ATM frame is a form of encapsulation.

- **Decapsulation:**

- Decapsulation is the process of opening up encapsulated data that are usually sent in the form of packets over a communication network.
- Data decapsulation is simply the reverse of encapsulation.
- As the data moves up from the lower layer to the upper layer of TCP/IP protocol stack (incoming transmission), each layer unpacks the corresponding header and uses the information contained in the header to deliver the packet to the exact network application waiting for the data.

# CN (IT-3001)

## Introduction

Prof. Amit Jha  
School of Electronics Engineering (SOEE)  
KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Content

- Types of connections
  - Point-to-Point
  - Multipoint
  - Broadcast
- Network Topologies:
  - Mesh
  - Star
  - Bus
  - Ring
  - Tree
- Categorization of Network based on Area
  - PAN
  - LAN
  - MAN
  - WAN

# Types of Connections

- There are two possible types of connections: **Point-to-Point** and **Multipoint**.

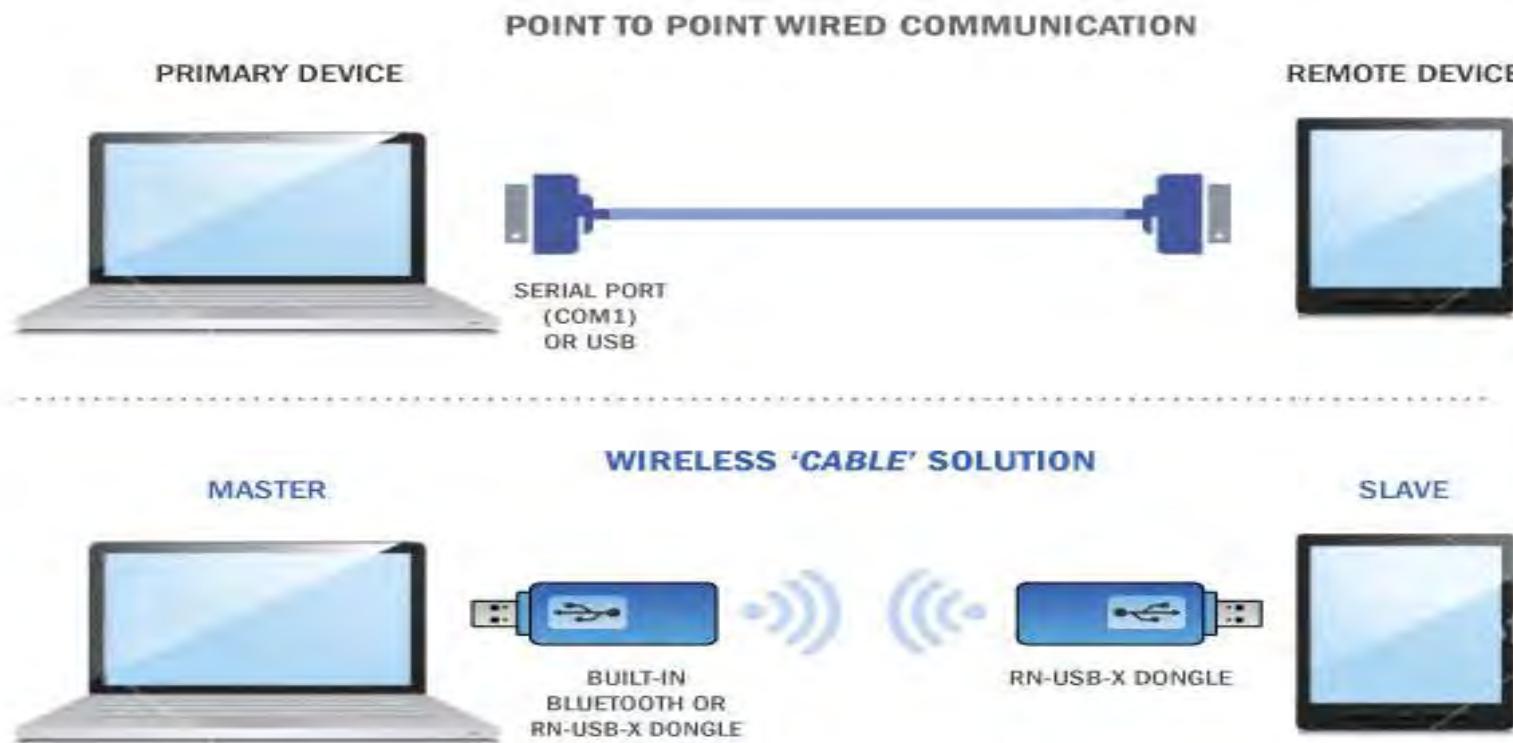
## 1) Point-to-Point:

- A point-to-point connection provides a **dedicated** link between two devices.
- The **entire capacity** of the link is **reserved** for transmission between those two devices.

**Note:** *Most point-to-point connections use an actual length of wire or cable to connect the two ends, but other options, such as microwave or satellite links, are also possible.*

**Example:** When we change television channels by infrared remote control, we are establishing a point-to-point connection between the **remote control** and the **television's** control system.

# Point-to-point connection



## 2) Multipoint

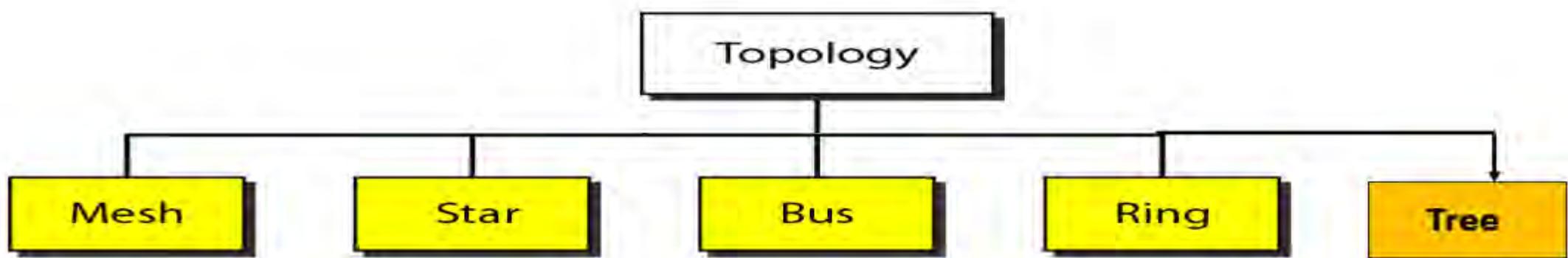
- A multipoint (also called multidrop) connection is one in which more than two specific devices share a single link.
- In a multipoint environment, the capacity of the channel is shared, either **spatially or temporally**.
- If several devices can use the link simultaneously, it is a ***spatially shared*** connection.
- If users must take turns, it is a ***timeshared*** connection.



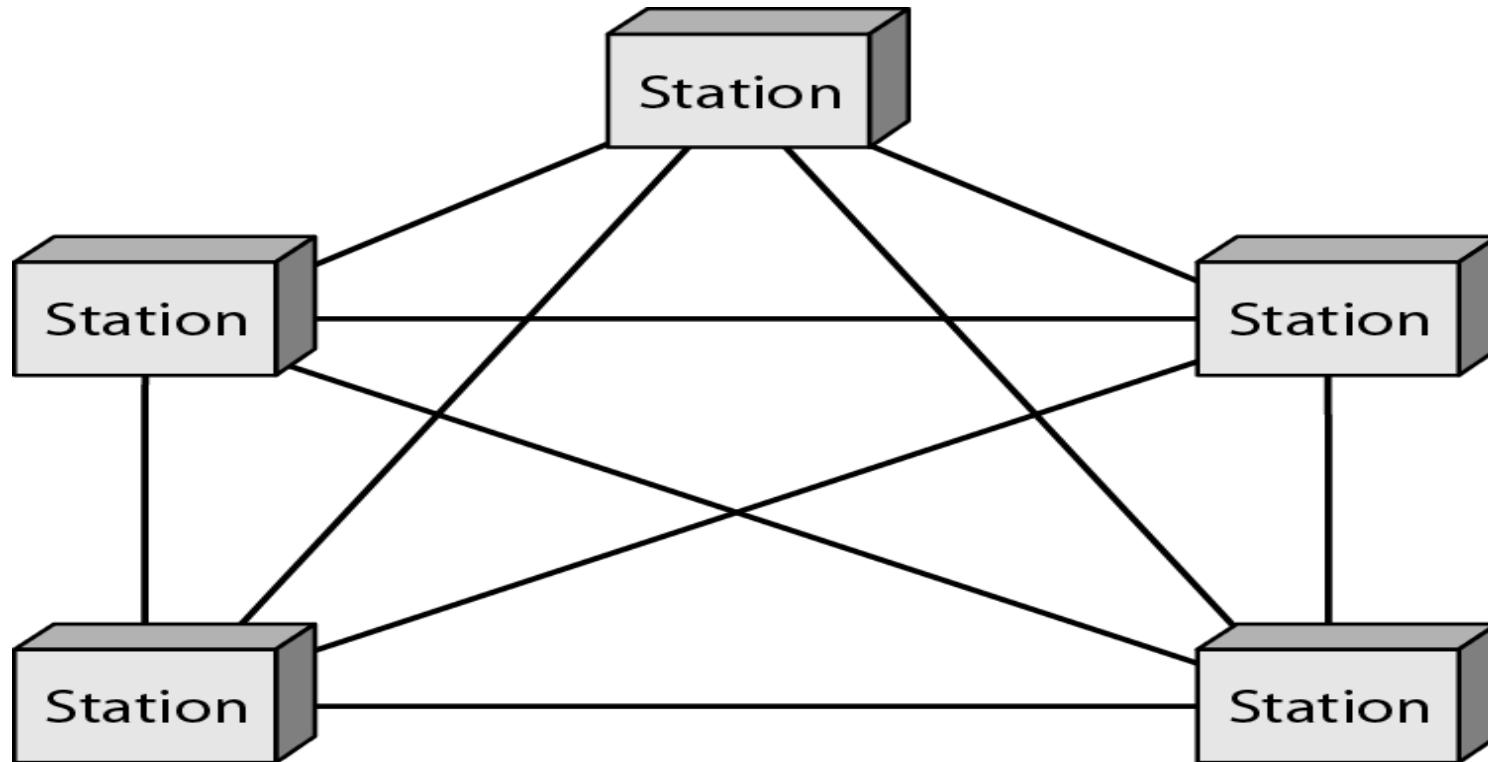
**In what fashion devices should  
be connected to a network ?**

# Network Topology

- A Network Topology is the arrangement with which computer systems or network devices are connected to each other.
- Topologies may define both physical and logical aspect of the network. Both logical and physical topologies could be same or different in a same network.
- There are four basic and one hybrid topologies possible: mesh, star, bus, ring and Tree.



# Network Topology: Mesh



In a mesh topology, every device has a **dedicated** point-to-point link to every other device. In a **fully connected** mesh network, total number of physical links required for  $n$  devices are:  $n(n-1)$

And if we assume each link is **duplex** then, total number of physical links required will be  $n(n-1)/2$ .

- **Advantage:** A mesh offers several advantages over other network topologies like
  - The use of **dedicated links** guarantees that each connection can carry its own data load, thus eliminating the **traffic problems** that can occur when links must be shared by multiple devices.
  - A mesh topology is **robust**. If one link becomes unusable, it does not incapacitate the **entire system**.
  - There is the advantage of **privacy or security**. When every message travels along a dedicated line, only the intended recipient sees it.
  - Finally, **point-to-point links** make fault identification and fault isolation **easy**. So, traffic can be routed to avoid links with suspected problems.
- **Disadvantage:** The main reason for disadvantage is **the amount of cabling** and the number of I/O ports required.
  - First, because every device must be connected to every other device, installation and reconnection are difficult.
  - Second, the sheer **bulk of the wiring** can be greater than the available space to accommodate.
  - Finally, the hardware required to connect each link (I/O ports and cable) can be prohibitively **expensive**.
- **Application:** For the reasons of all its disadvantages, a mesh topology is usually implemented in a limited fashion, for example, as a backbone connecting the main computers of a **hybrid network** that can include several other topologies.

One **practical example** of a mesh topology is the connection of **telephone regional offices** in which each regional office needs to be connected to every other regional office.

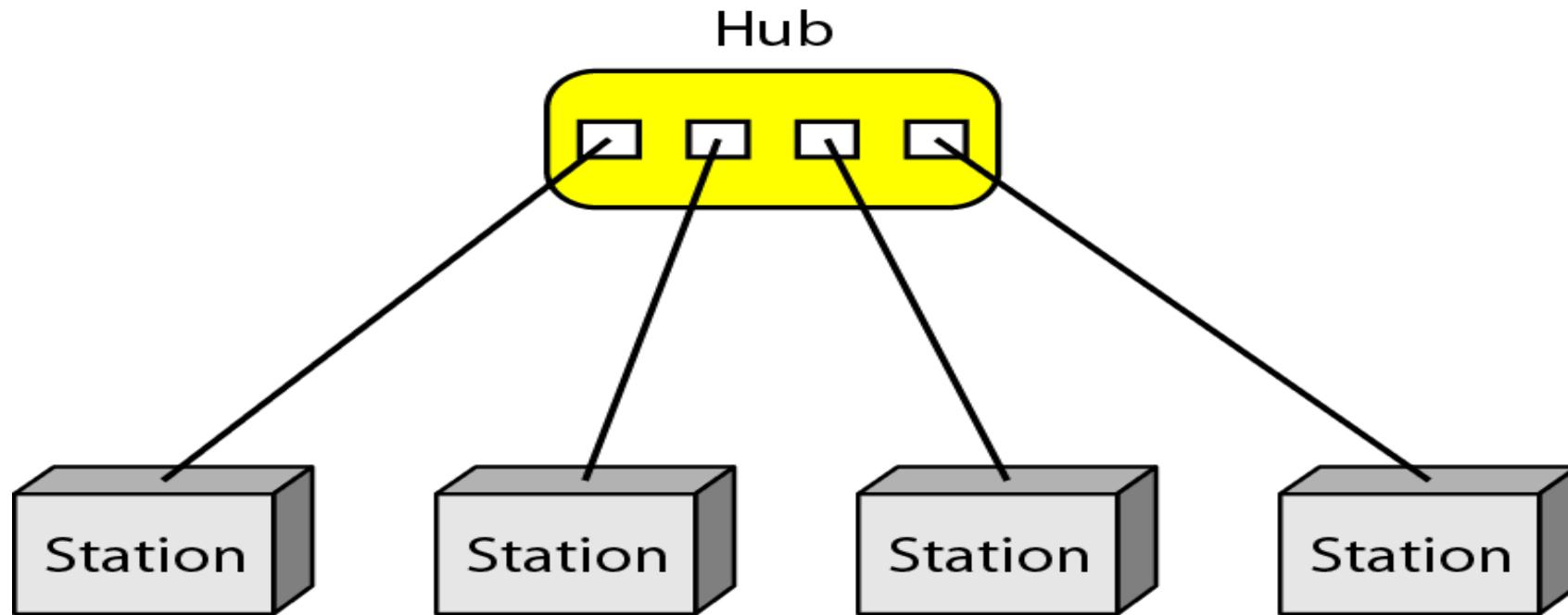
**Can we go for mesh topology for our campus LAN connection?**

**Can we go for mesh topology for our campus LAN connection?**



No, otherwise this may be the scenario...

# Network Topology: Star



In a star topology, each device has a **dedicated point-to-point link** only to a central controller, usually called a **hub**.

Unlike a mesh topology, a star topology does not allow direct traffic between devices. The **controller** acts as an exchange: If one device wants to send data to another, it sends the data to the controller, which then relays the data to the other connected device.

- **Advantage:**
  - Since, less # of dedicated links, it is less expensive than a star topology.
  - Each device needs only one link and one I/O port to connect it to any number of others. This factor also makes it easy to install and reconfigure.
  - Since less cabling, addition, deletion, movement involve only one connection: between that device and the hub.
  - Robustness: i.e. if one link fails, only that link is affected. This factor also lends itself to easy fault identification and isolation.
- **Disadvantage:** If the hub goes down, the whole system is dead. This is because the whole system depends upon a central controller, called ***Hub***.
- **Application:** High-speed LANs often use a star topology with a central hub.

**Question:** Mesh topology and star topology belong to which one of the following types of connections?

a) Point-to-point

b) Multipoint



**Question:** Mesh topology and star topology belong to which one of the following types of connections?

a) Point-to-point

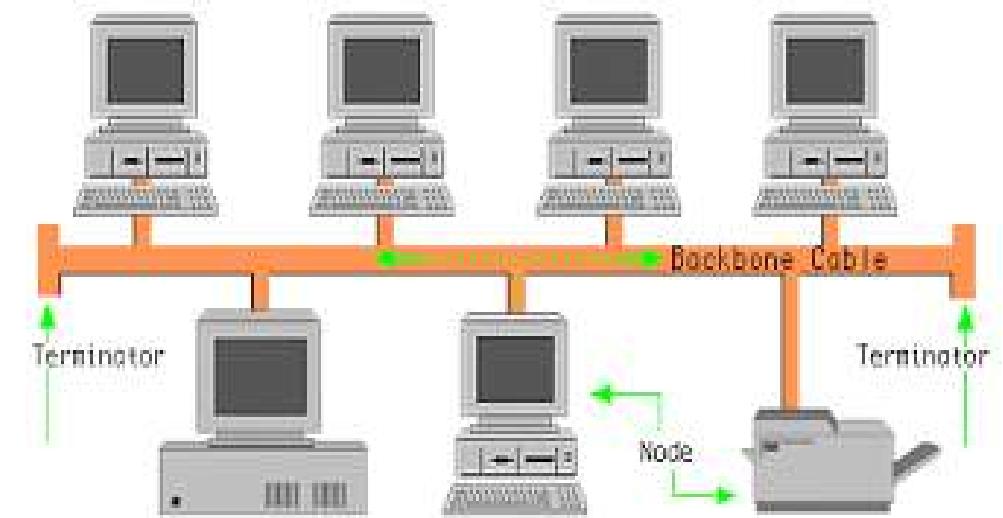
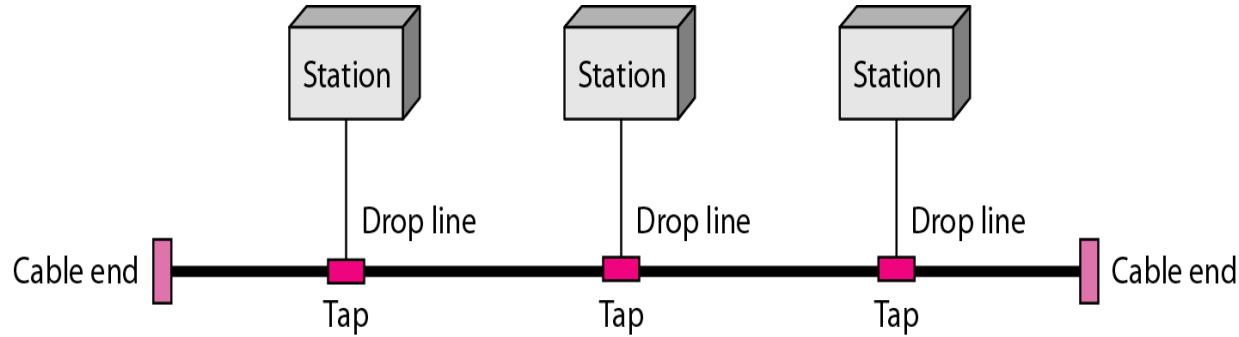
b) Multipoint



Both belongs to point-to-point connection.

# Network Topology: Bus

- A **bus topology** is an example of multipoint connection.



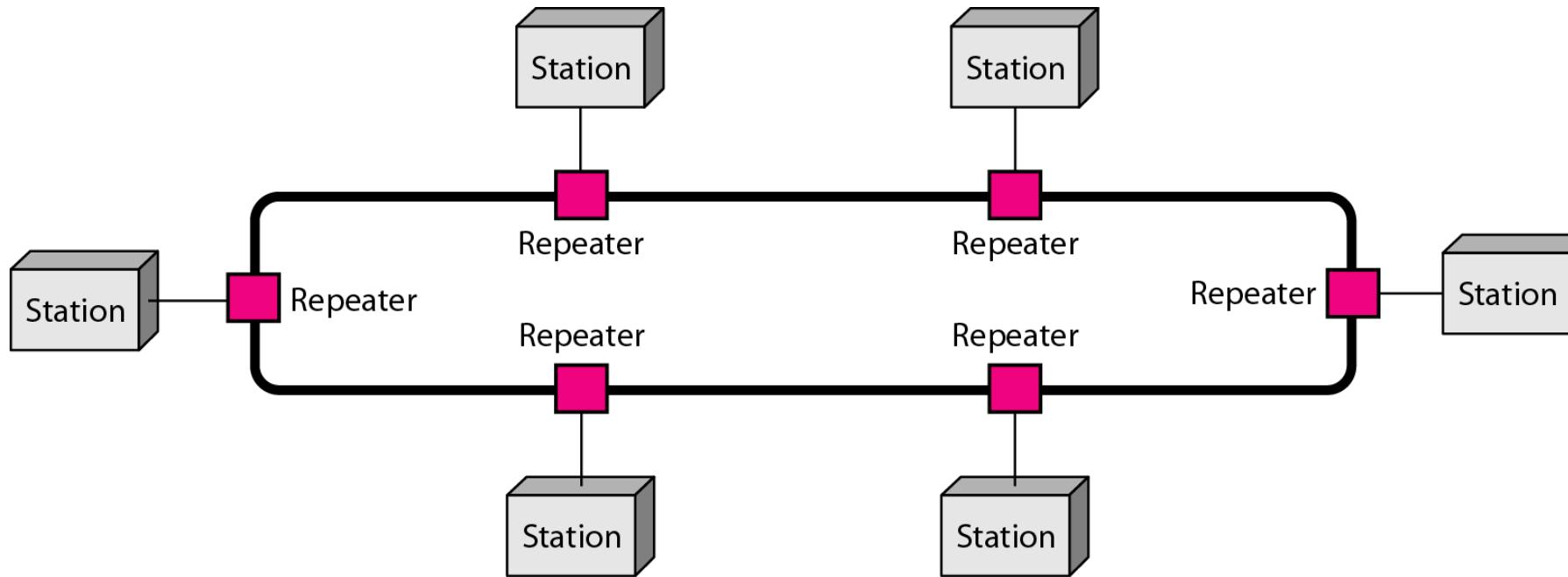
One long cable acts as a **backbone** to link all the devices in a network.

Nodes are connected to the bus cable by **drop lines** and **taps**.

A drop line is a connection running between the device and the main cable, whereas tap is a connector.

- **Advantages:**
  - Easy installation.
  - A bus uses **less cabling** than mesh or star topologies.
  - For e.g., in a star topology with 100 devices, we need 100 cable, on the other hand, in case of bus topology we need **only one backbone** cable to which 100 devices are connected by **small length drop line**.
- **Disadvantage:**
  - Reconnection and fault isolation are difficult.
  - **Adding new device is difficult.** This is because of a bus is usually designed to be optimally efficient at installation.
  - The **signal reflection at tap** degrades the quality of the signal, thus backbone is designed by limiting the number and spacing of devices connected to it.
  - A fault or break in the bus cable stops all transmission.
- **Application:** Ethernet LANs

# Network Topology: Ring



- In a ring topology, each device has a **dedicated point-to-point** connection with only the **two devices on either side of it**.
- A signal is passed along the ring in **one direction**, from device to device, until it reaches its destination.
- Each device in the ring incorporates a **repeater**. When a device receives a signal intended for another device, its repeater regenerates the bits and passes them along.

- **Advantage:**

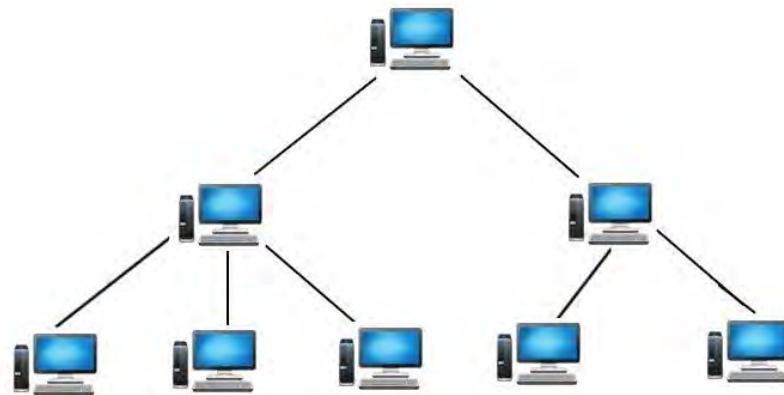
- A ring is relatively easy to install and reconfigure.
- Since, each device is linked to only its immediate neighbours. To add or delete a device requires changing only two connections.
- Easily problem identification and solution, this is because: Generally in a ring, a signal is circulating at all times. If one device does not receive a signal within a specified period, it can issue an alarm. The alarm alerts the network operator to the problem and its location.

- **Disadvantage:**

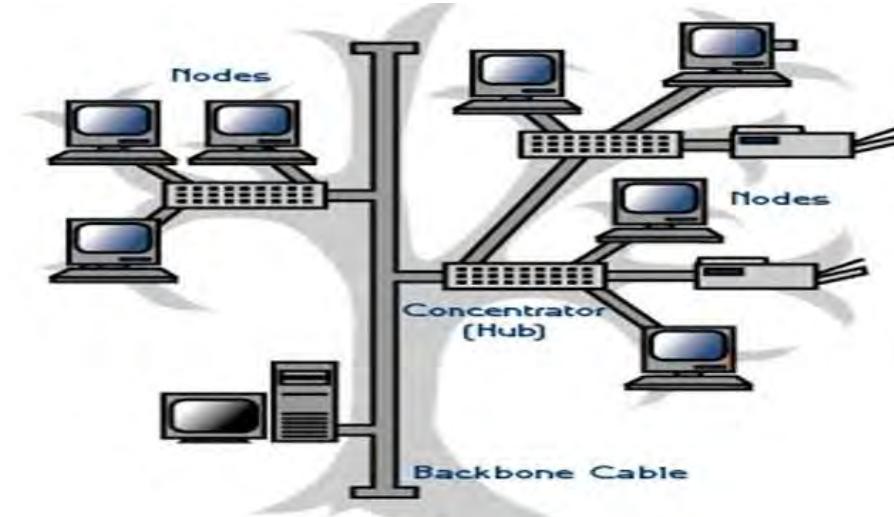
- Since traffic is unidirectional, In a simple ring, a break in the ring (such as a disabled station) can disable the entire network. This weakness can be solved by using a dual ring or a switch capable of closing off the break.
- **Application:** Used in LAN introduced by IBM, but now its less popular because of the need of high-speed LAN.

# Network Topology: Tree

- A **tree topology** is a special type of structure in which many connected elements are arranged like the branches of a tree.
- Tree topologies form a natural parent and child hierarchy as shown in Fig.1
- In computer networks, a tree topology is also known as a **star bus topology**. It incorporates elements of both a bus topology and a star topology.



**Fig 1:** Tree Topology, In general



**Fig:** Tree Topology, as star-Bus topology

### **Advantages of tree topology:**

- Scalable as leaf nodes can accommodate more nodes in the hierarchical chain.
- A point to point wiring to the central hub at each intermediate node of a tree topology represents a node in the bus topology
- Other hierarchical networks are not affected if one of them gets damaged
- Easier maintenance and fault finding

### **Disadvantages of tree topology:**

- Huge cabling is needed
- A lot of maintenance is needed
- backbone forms the point of failure.

# What network topologies to choose and when to choose



What network topologies to choose and when to choose



For point-to-point connection star topology is better  
And in case of multipoint connection bus topology is better.

But, there are some more conditions for choosing the **best** topology.

## Try by yourself

Consider you want to start your own small company consisting of you as owner, a business manager, an administrator, and four agents. Assume that your company will grow and after two years, you have to hire two new agents for the same.

Everyone in the company has a computer, but the business manager has the only printer. These computers are not connected by any form of networking. When agents need to print a document, they must first copy the file to a USB, then carry it to the business manager's computer, where they are finally able to print it. Similarly, when staff members want to share data, the only means available is to copy the data on one computer to a USB and insert the it in another computer.

Recently, problems have arisen. The business manager is spending too much time printing other people's documents; and it is frequently unclear which copy of a given document is the current and authoritative version.

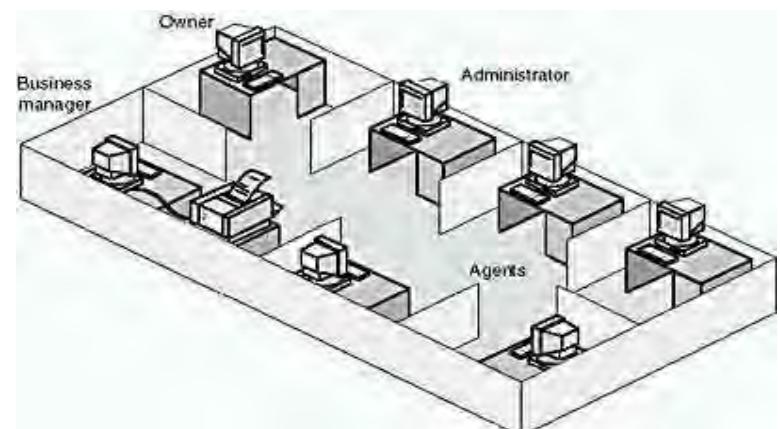
Your task is to design a network for this company by answering following questions.

*Which type of network would you suggest for this company?*

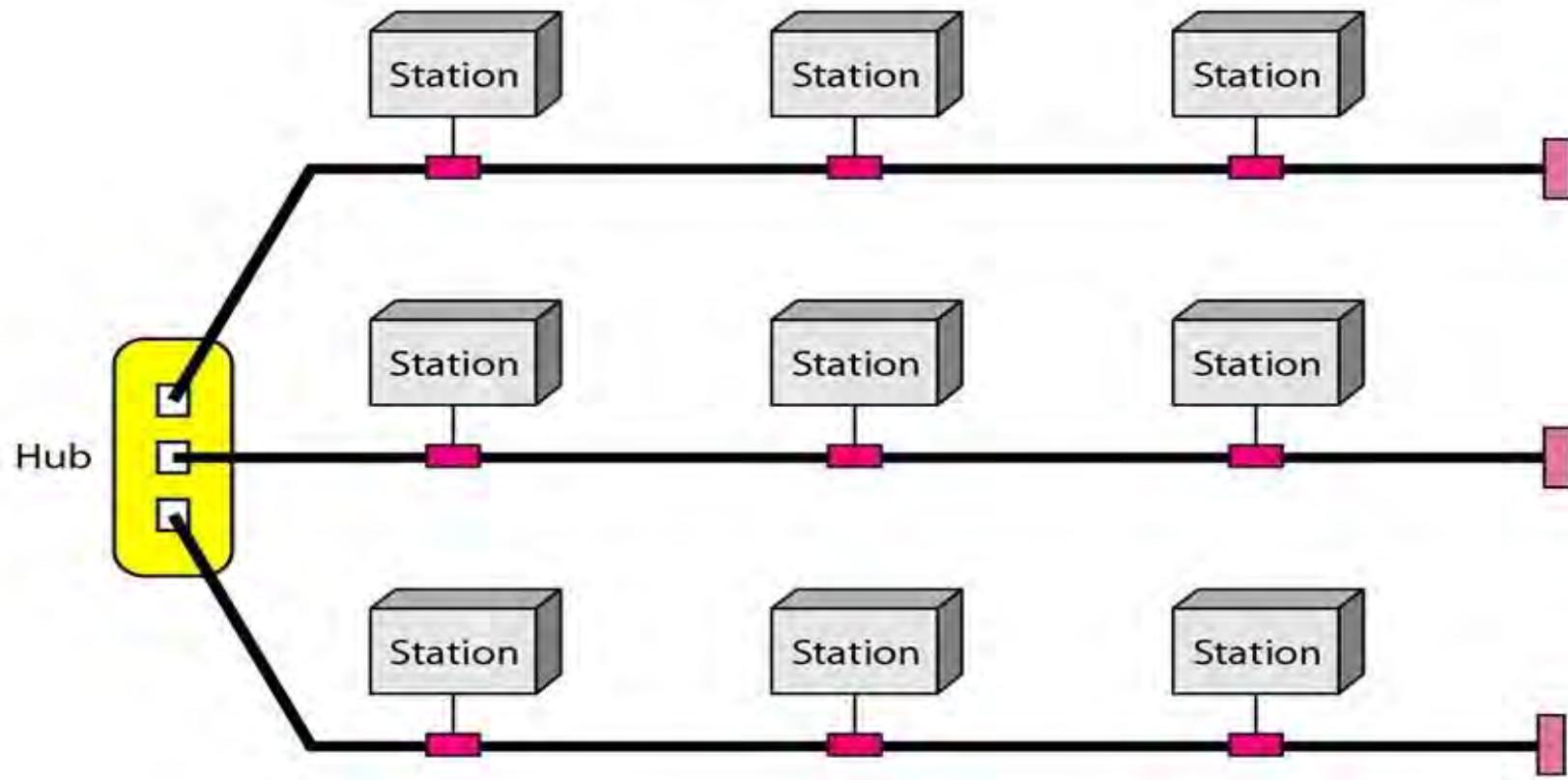
Peer-to-peer or Server-based

*Which network topology would be most appropriate in this situation?*

Bus,	Ring	Star
Mesh	Star bus	Star ring



- **Hybrid Topology:** A network can be hybrid. For example, we can have a main star topology with each branch connecting several stations in a bus topology as shown below.
- **Note:** star-bus topology is also known as tree topology.



# Categories of Networks

- It is of three types based on geographical area:
  1. PAN
  2. LAN
  3. MAN
  4. WAN

**What is PAN, LAN, MAN & WAN ?**

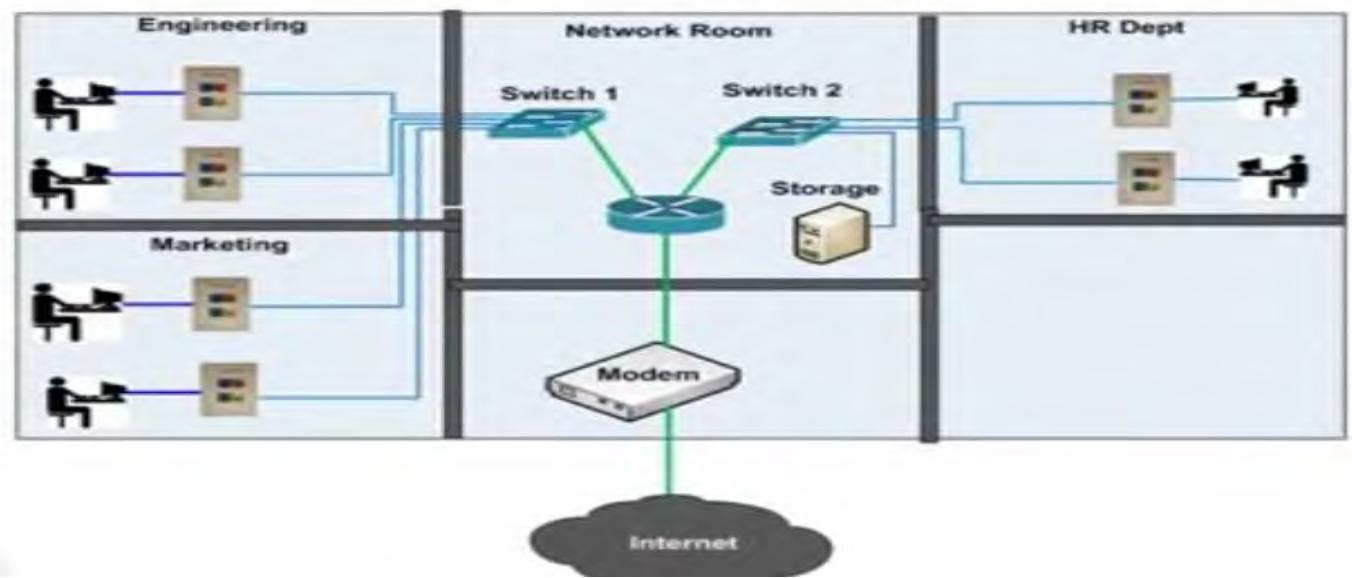
- **Personal Area Network (PAN):**

- It is a network with a size covering **few meters** of area.
- It normally covers the area inside a room.
- It is designed to communicate the devices nearby a person.
- PAN can be wired, such as USB or FireWire, or it can be wireless- such as infrared, Bluetooth, ZigBee, Hotspot, etc.
- Examples: FireStick, Hotspot, wireless mouse, wireless keyboard, gaming consoles, etc.



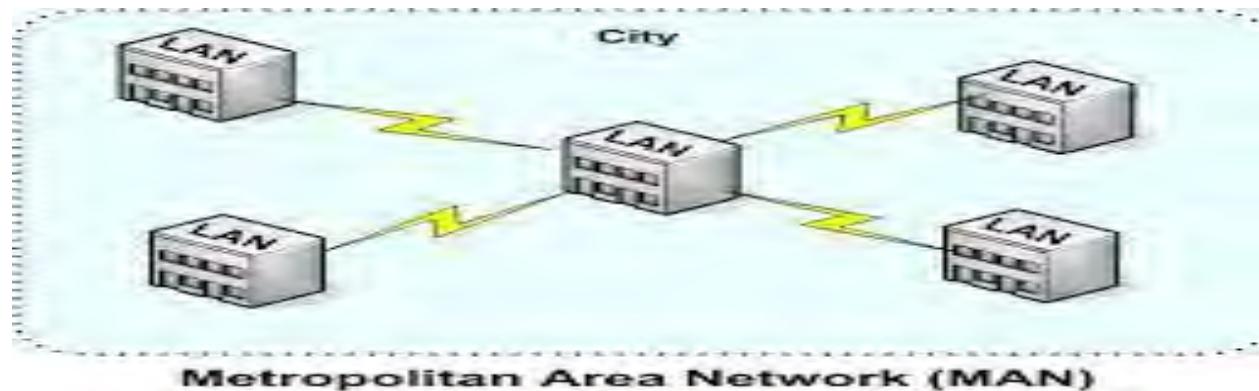
- **Local Area Network (LAN):**

- Generally covers **few kilometres**.
- A local area network (LAN) is usually privately owned and links the devices in a single office, building, or campus.
- Early LANs had data rates in the 4 to 16 megabits per second (Mbps) range. Today, however, speeds are normally 100 or 1000 Mbps.
- It may be wired or wireless.



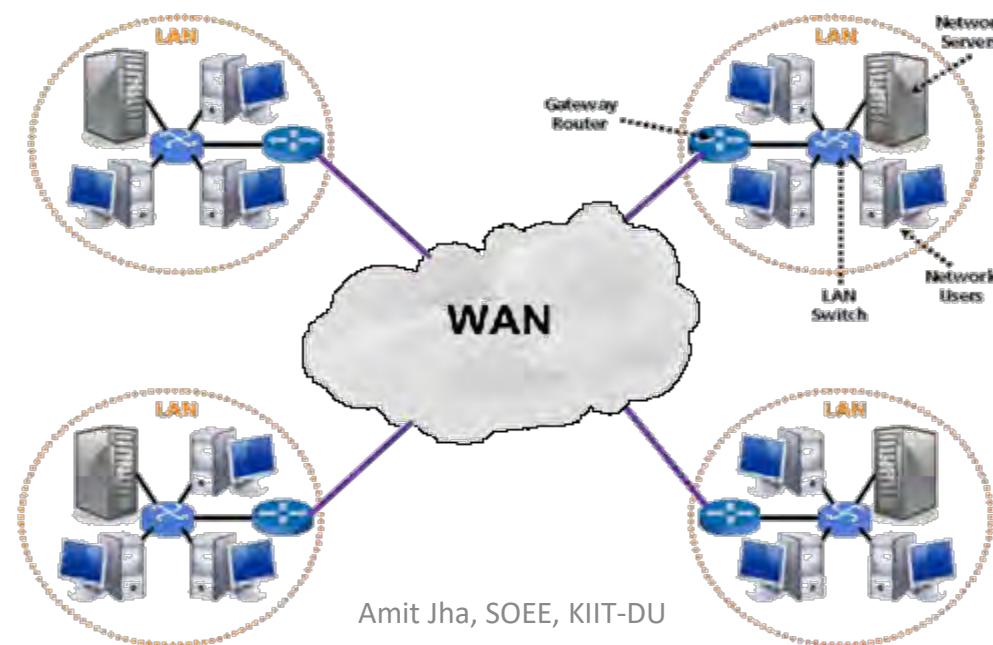
- **Metropolitan Area Network (MAN):**

- It is a network with a size between a LAN and a WAN.
- It normally covers the area inside a town or a city.
- It is designed for customers who need a high-speed connectivity, normally to the Internet, and have endpoints spread over a city or part of city.
- A good example of a MAN is the part of the telephone company network that can provide a high-speed DSL line to the customer.
- Another example is the cable TV network that originally was designed for cable TV, but today can also be used for high-speed data connection to the Internet.



- **Wide Area Network (WAN):**

- A wide area network (WAN) provides long-distance transmission of data, image, audio, and video information over large geographic areas.
- Area may comprise a country, a continent, or even the whole world.
- A good example of WAN is the asynchronous transfer mode (ATM) network.



# CN (IT-3001)

## Networking Devices, Switching

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Content

- Networking Devices
  - Hub, switches, routers, gateway, etc.
- Different Switching techniques:
  1. Circuit Switching
  2. Message Switching
  3. Packet Switching
    - Virtual Circuit Switching
    - Datagram switching

# Types of connecting devices in a network

- Basically, we have five connecting devices: **Repeater, Hubs, Switch, Routers, Bridges, and Gateways.**

# Repeater/Hub

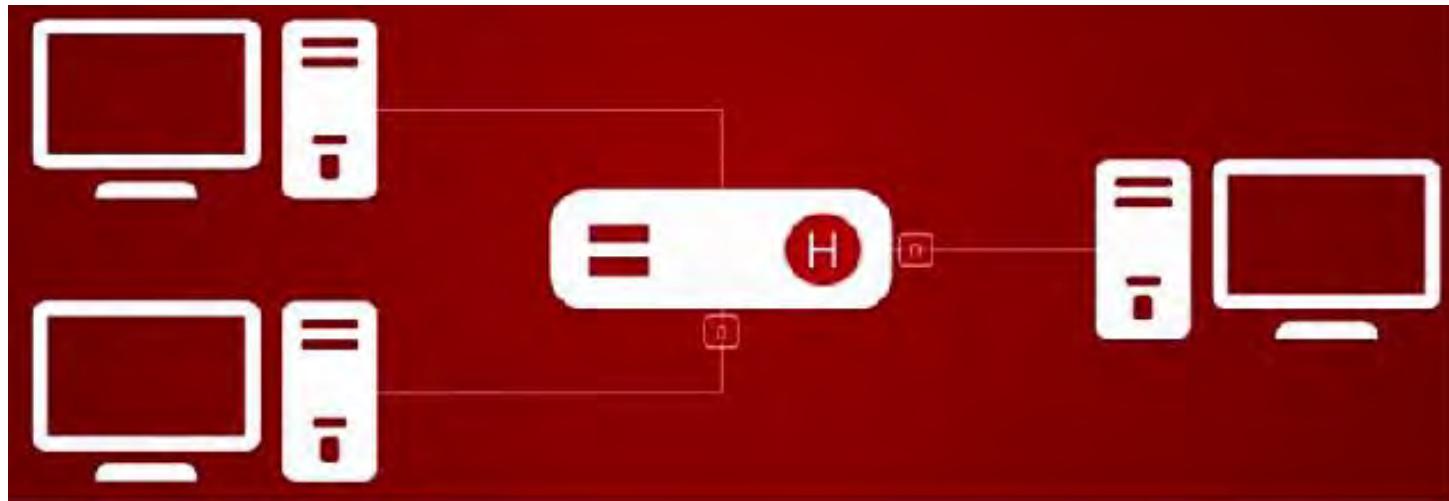


Fig. Network with a hub



# How a repeater/hub behaves

- It does not know anything about the packets(from network layer) i.e., to whom **does it belong to.**
- It sends packets to **each device** connected to it.
- It is the simplest and cheapest way to create a network.
- But it generates **lots of unnecessary traffic.**
- Thus, **wastage in Bandwidth.**
- **Security** is an issue.

# Switch/Bridge

- It is smarter brother of hub.
- It sends packets to the exact destination without spanning the entire network.
- Each device has NIC card with a unique MAC address.

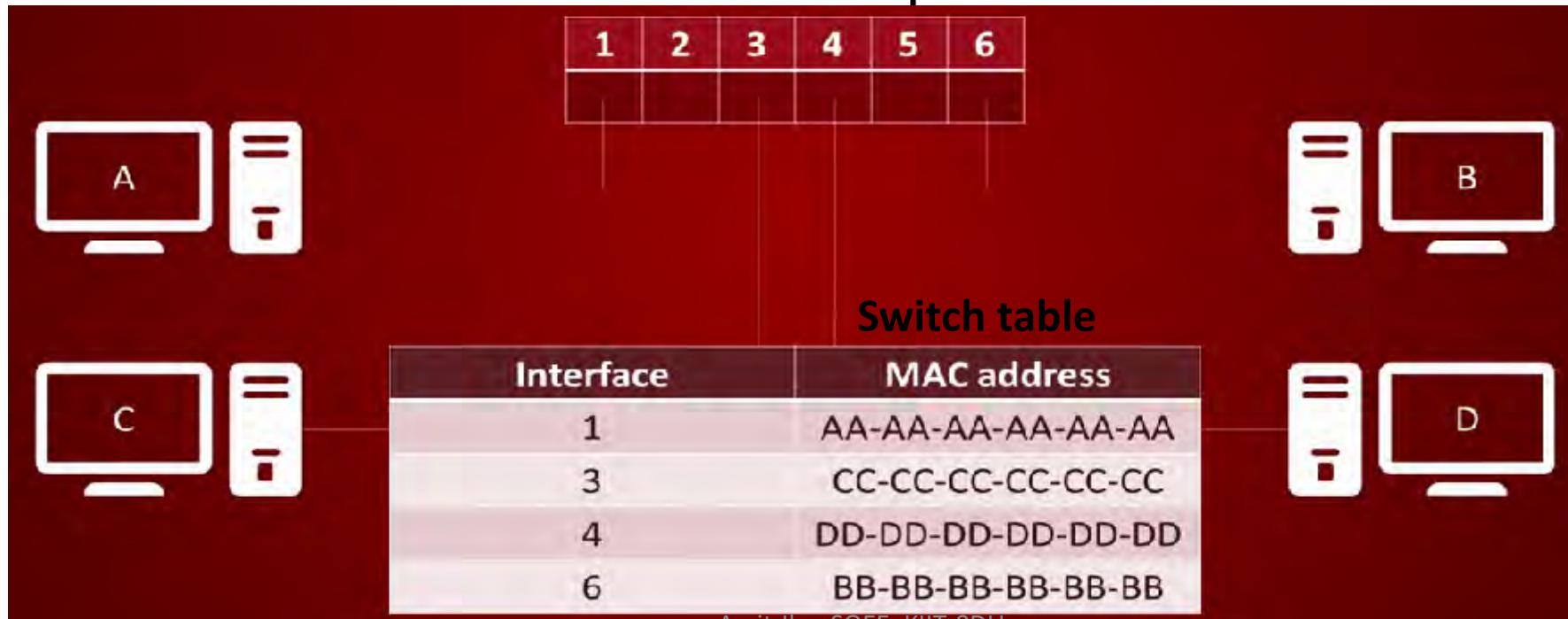


Fig. Network with a switch

# How a Switch behaves?

- Every device has **a unique MAC** address printed on NIC card.
- Every packet is sent to the specific destination only.
- Thus, **No flooding** of packets.
- No traffic (B.W) problem.
- Not only the **Ethernet** connections but **also Wi-Fi** uses it.



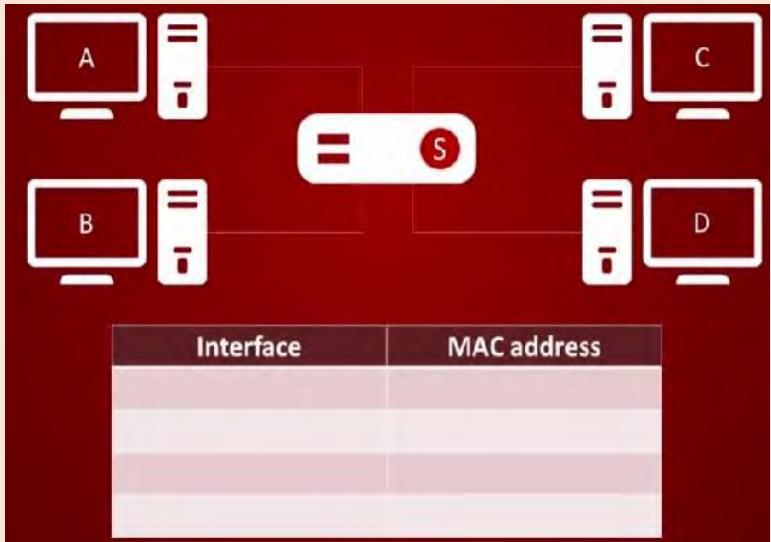
- *How does a switch know the MAC address of every devices?*



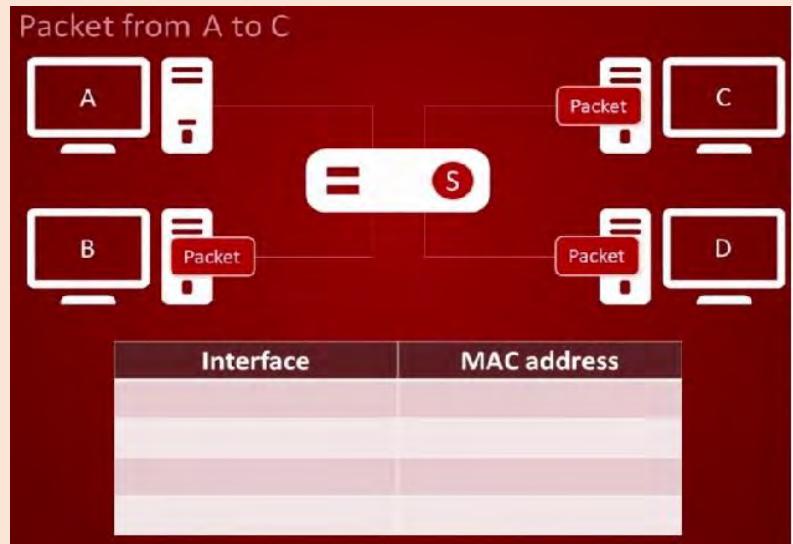
- *How does a switch know the MAC address of every devices?*

*Ans: The switch table.*

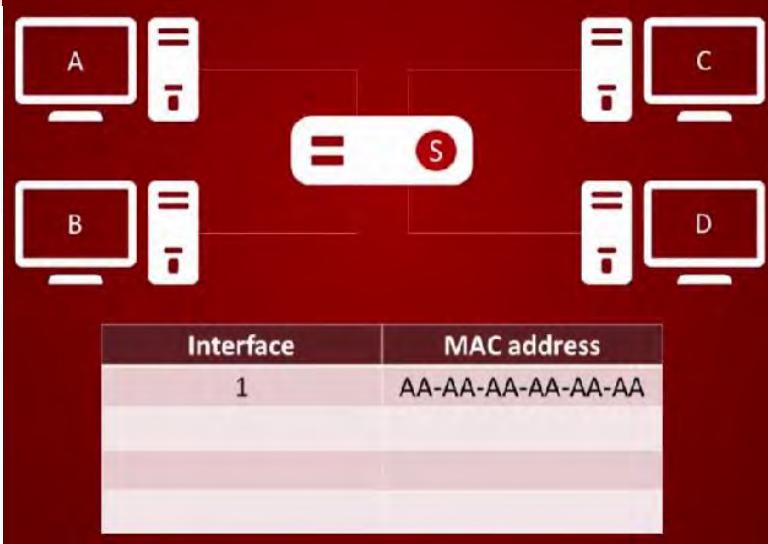




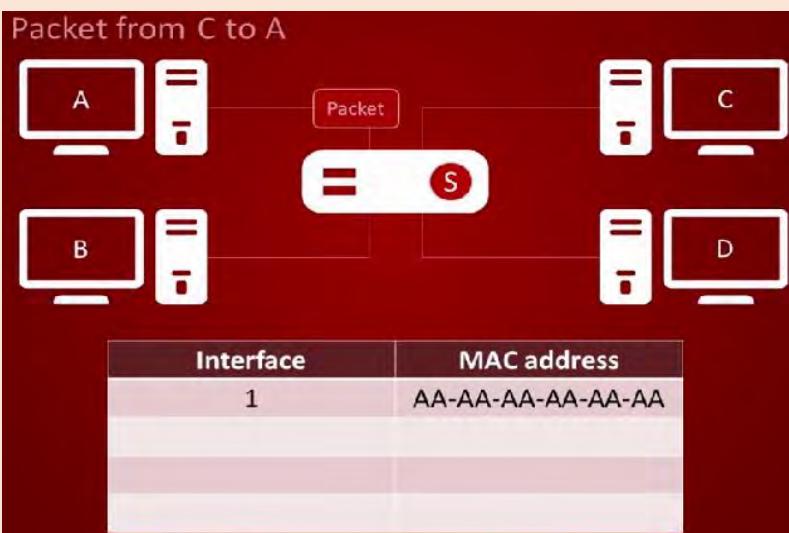
Step 1: Initially switch is empty



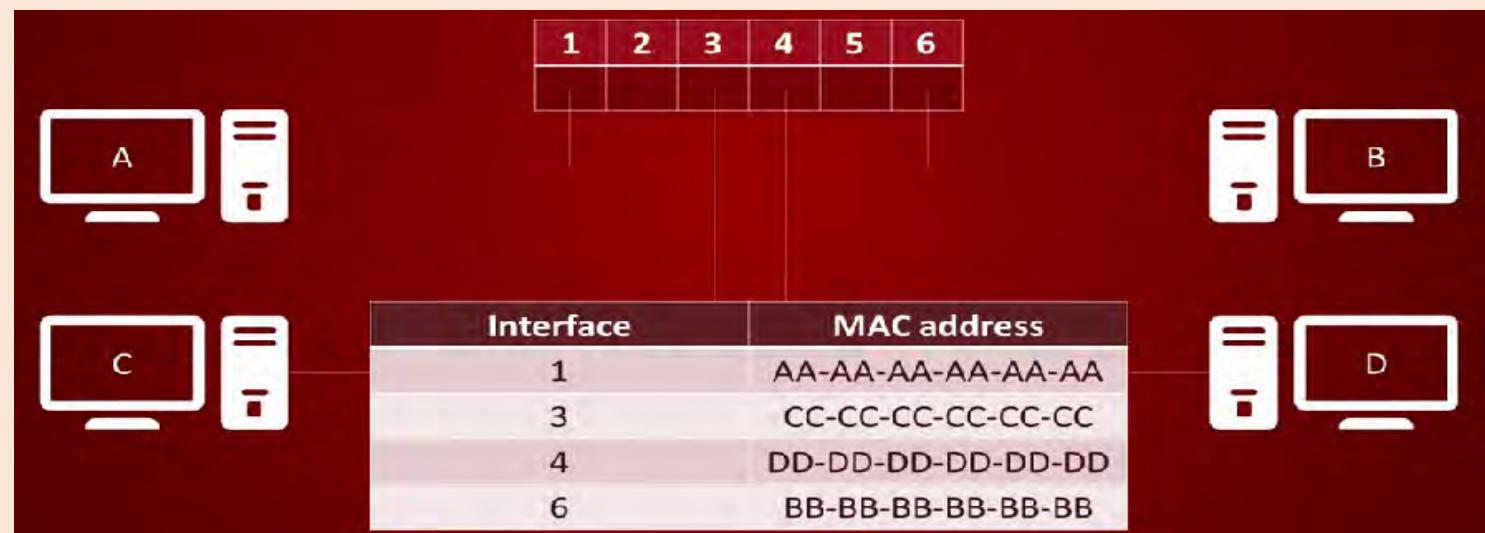
Step 2: Flooding of packets



Step 3: Previously used MAC address is now known



Step 4: No flooding



Step 5: Finally, the switch table is created with the same process

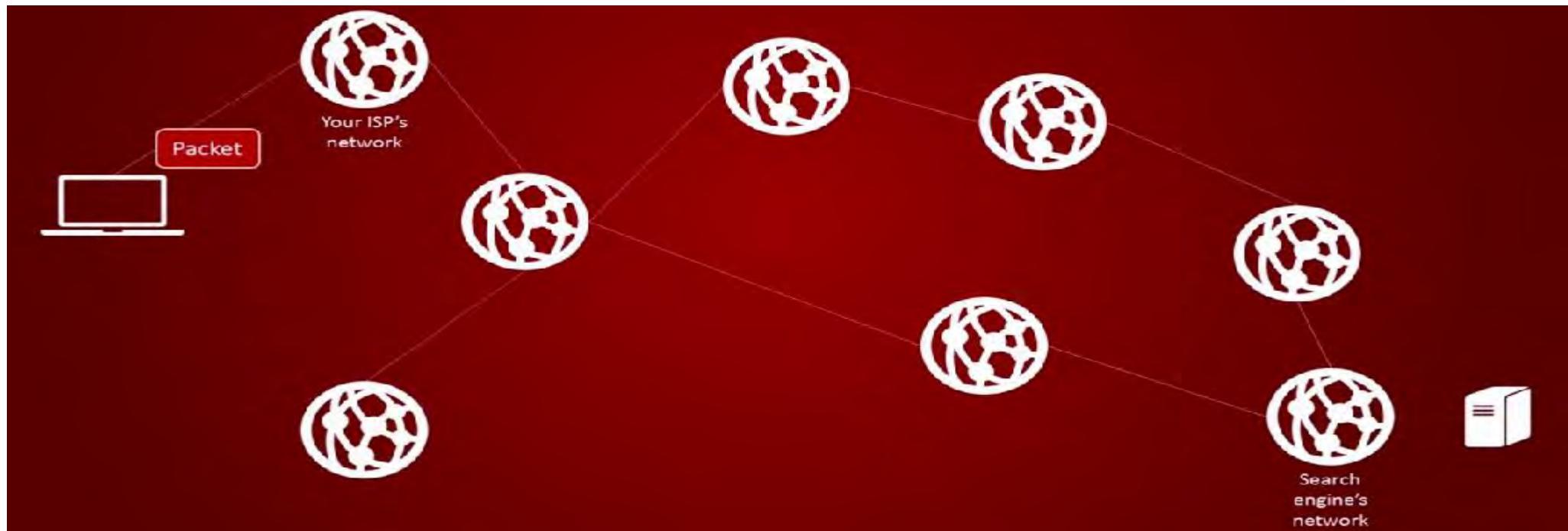
# Router

- It is the **glue** that ties networks together.
- **Hubs** and **switches** are the devices used to create networks.
- But if we want to send packets between those networks then routers come into the picture.
- We need routers to send packets from our laptop to search engine server over the Internet.
- Once, we send packets to our **ISP**, routers make sure that packets are passed on from network to network to reach search engine server.
- We can use routers at home as well.
- Some other functions of a router include
  - Performing network address translation
  - Assigning IP address to hosts using DHCP
  - Broadcasting a WiFi signal (i.e., acts as an access point for WiFi)

# How does a router behaves?



Fig. Why do we need routers?



Amit Jha, SOEE, KIIT-ODU  
Fig. Routers on the Internet



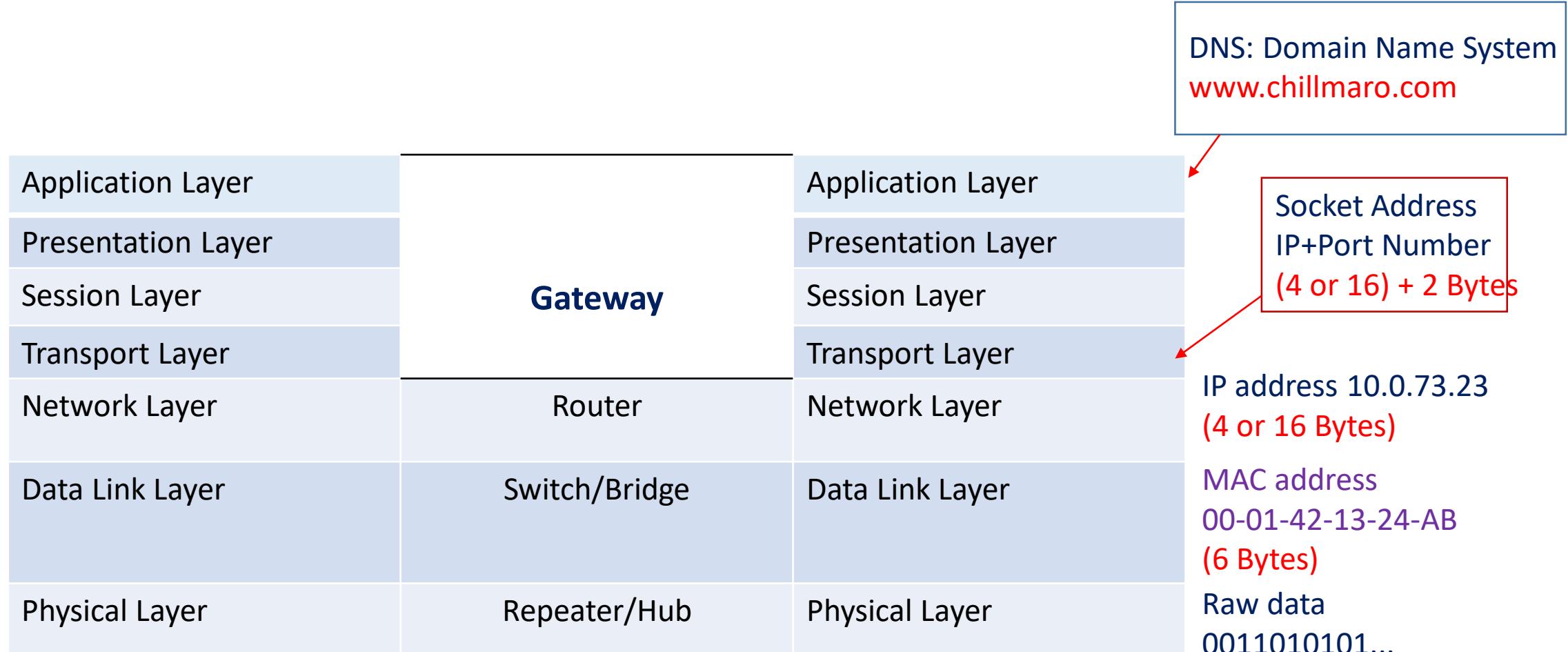
**Can we use any connecting devices at any layer?**



**Can we use any connecting devices at any layer?**

**No...**

# Use of different connecting devices in a network



# What should you buy?

- Don't buy a hub because
  - They are bandwidth wasters.
  - Switches are not expensive than hub. So buy a switch instead of a hub,
- Switch
  - is cheaper than a router.
  - We can still connect the switch to a router
  - Quicker than routers for internal communications
  - Many routers have the Ethernet switch built in
- Router
  - It **connects** a network with the other networks.

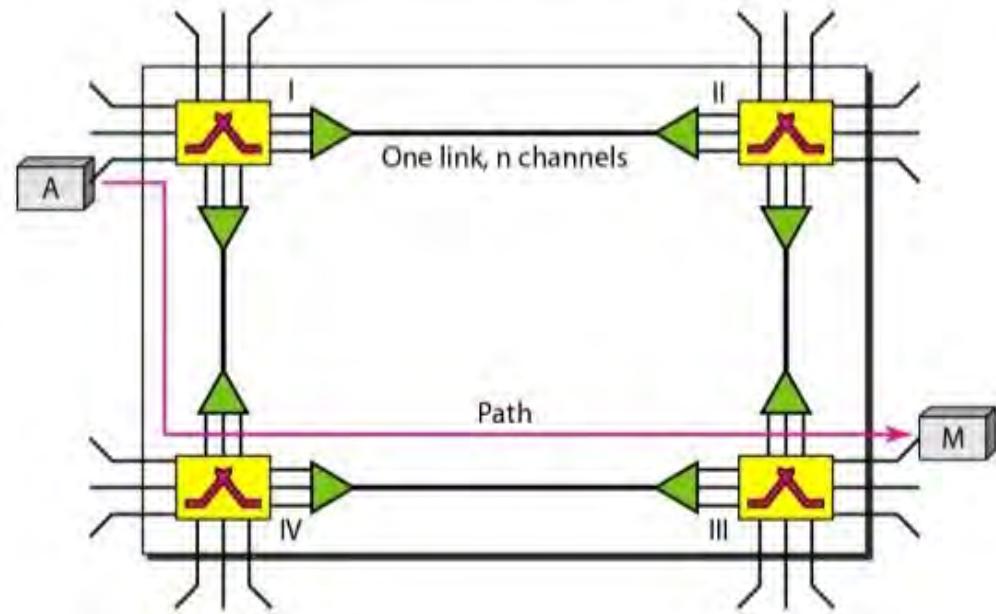
# Switching

- Traditionally, three methods of switching have been important: *circuit switching*, *message switching*, and *packet switching*.
- The first and last are commonly used today, the middle one has been phased out in general communications but still has networking applications.
- We can then divide today's networks into three broad categories: *circuit-switched networks*, *message-switched networks*, and *packet-switched networks*.
- Packet-switched networks can further be divided into two subcategories datagram networks and virtual-circuit networks.

# Circuit-Switched Networks

- It is made of a set of switches connected by physical links, in which each link is divided into *n* channels.
- In the **circuit-switched network**, there are 4 switches and 4 links.
- Each link is divided into *n* (*n*=3, here) channels by using FDM or TDM.
- In circuit switching, the resources need to be reserved during the setup phase; the resources remain dedicated for the entire duration of data transfer until the teardown phase.

A trivial circuit-switched network



# Circuit switching: Key points

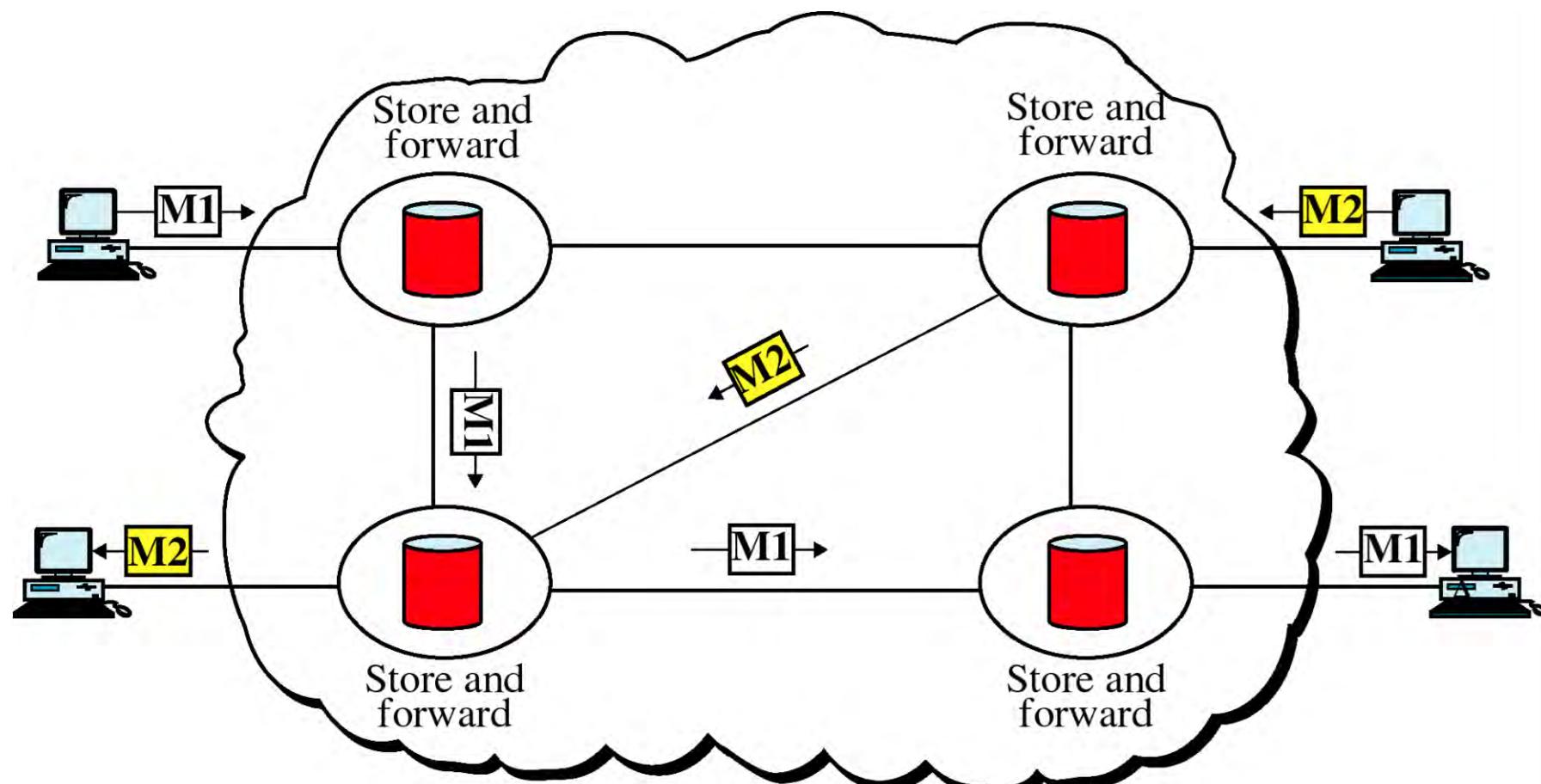
- *Circuit switching* is a technique that directly connects the sender and the receiver in an unbroken path.
- Telephone switching equipment, for example, establishes a path that connects the caller's telephone to the receiver's telephone by making a physical connection.
- With this type of switching technique, once a connection is established, a dedicated path exists between both ends until the connection is terminated.
- Three phases in circuit switching are Establish, Transfer, Disconnect.
- Routing decisions must be made when the circuit is first established, but there are no decisions made after that time.
- Ex: It is used in telephone networks.

- ***Advantages:*** The communication channel (once established) is dedicated.
- ***Disadvantages:***
  - Possible **long wait** to establish a connection, during which no data can be transmitted.
  - **More expensive** than any other switching techniques, because a dedicated path is required for each connection.
  - **Inefficient use** of the communication channel, because the channel is not used when the connected systems are not using it.

# Message Switching

- With message switching there is no need to establish a dedicated path between two stations.
- When a station sends a message, the destination address is appended to the message.
- The message is then transmitted through the network, in its entirety, from node to node.
- Each node receives the entire message, stores it in its entirety on disk, and then transmits the message to the next node.
- This type of network is called a store-and-forward network.

# A Message Switched Network



# Message Switching: Key points

- A message-switching node is typically a general-purpose computer.
- The device needs **sufficient secondary-storage** capacity to store the incoming messages, which could be long.
- A **time delay** is introduced using this type of scheme due to store-and-forward time, plus the time required to find the next node in the transmission path.
- Example application: Although, we do not see its application at lower layer, it is still used in some applications like **electronic mail** (e-mail).

- ***Advantages:***

- Channel efficiency can be greater compared to circuit-switched systems, because more devices are sharing the channel.
- Traffic congestion can be reduced, because messages may be temporarily stored in route.
- Message priorities can be established due to store-and-forward technique.

- ***Disadvantages:***

- Message switching is not compatible with **interactive applications**.
- **Store-and-forward** devices are expensive, because they must have large disks to hold potentially long messages.

*Q.) Which are the factors on which we need to work on?*



***Q.) Which are the factors on which we need to work on?***

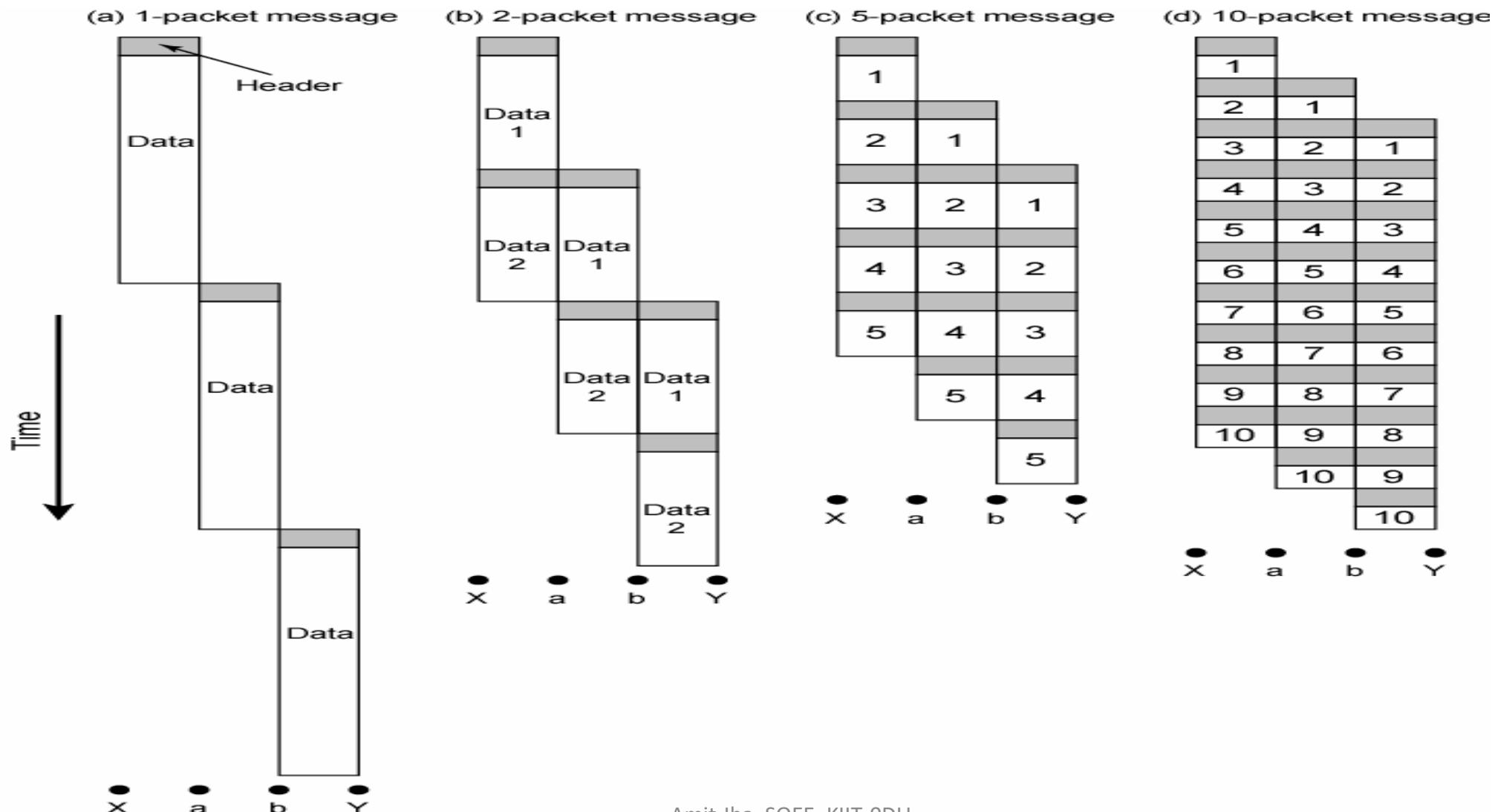
**Ans)**

- The first and foremost, we don't want dedicated channels.
- It will be better to divide message into **packets** bcz
  - Long message has more chances to **incur error** than short packets.
  - Long message is not suitable for interactive application because it causes very **long waiting delay** on other messages.
  - Moreover, the delay for one message is generally longer than the **total delay of packets** the message is divided into.

**Confused ??????????????????**



# Packet Size and its importance



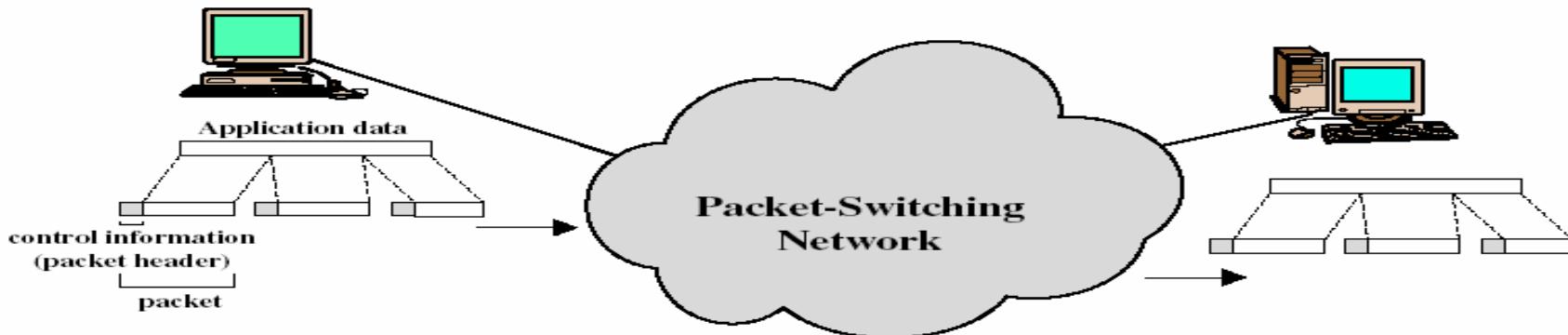
# Packet Switching

- **Packet switching** can be seen as a solution that tries to combine the advantages of message and circuit switching and to minimize the disadvantages of both.
- There are two methods of packet switching: **Datagram** and **virtual circuit**.
- In both packet switching methods, **a message is broken** into small parts(fixed or variable), called packets.
- Each packet is tagged with appropriate source and destination addresses.
- Since packets have a strictly defined maximum length, they can be stored in **main memory instead of disk**, therefore access delay and cost are minimized.
- Also the transmission speeds, between nodes, are optimized.
- With current technology, packets are generally accepted onto the network on a **first-come, first-served basis**. If the network becomes overloaded, packets are delayed or discarded ("dropped").

# Packet Switching: Key points

- In a packet switched, message needs to be divided into packets of fixed or variable size.
- The **size** of the packet is determined by the network and the governing protocol.
- Resources are allocated **on demand** on a first-come, first-served basis.
- Packet switching is further classified as:
  - Datagram Switching and
  - Virtual Circuit Switching

# Using Packet Switching: A Quick Look



packet header  
contains routing  
information

Intermediate  
nodes(routers) store the  
packet, decide the route,  
and forward the packet:  
processing delay

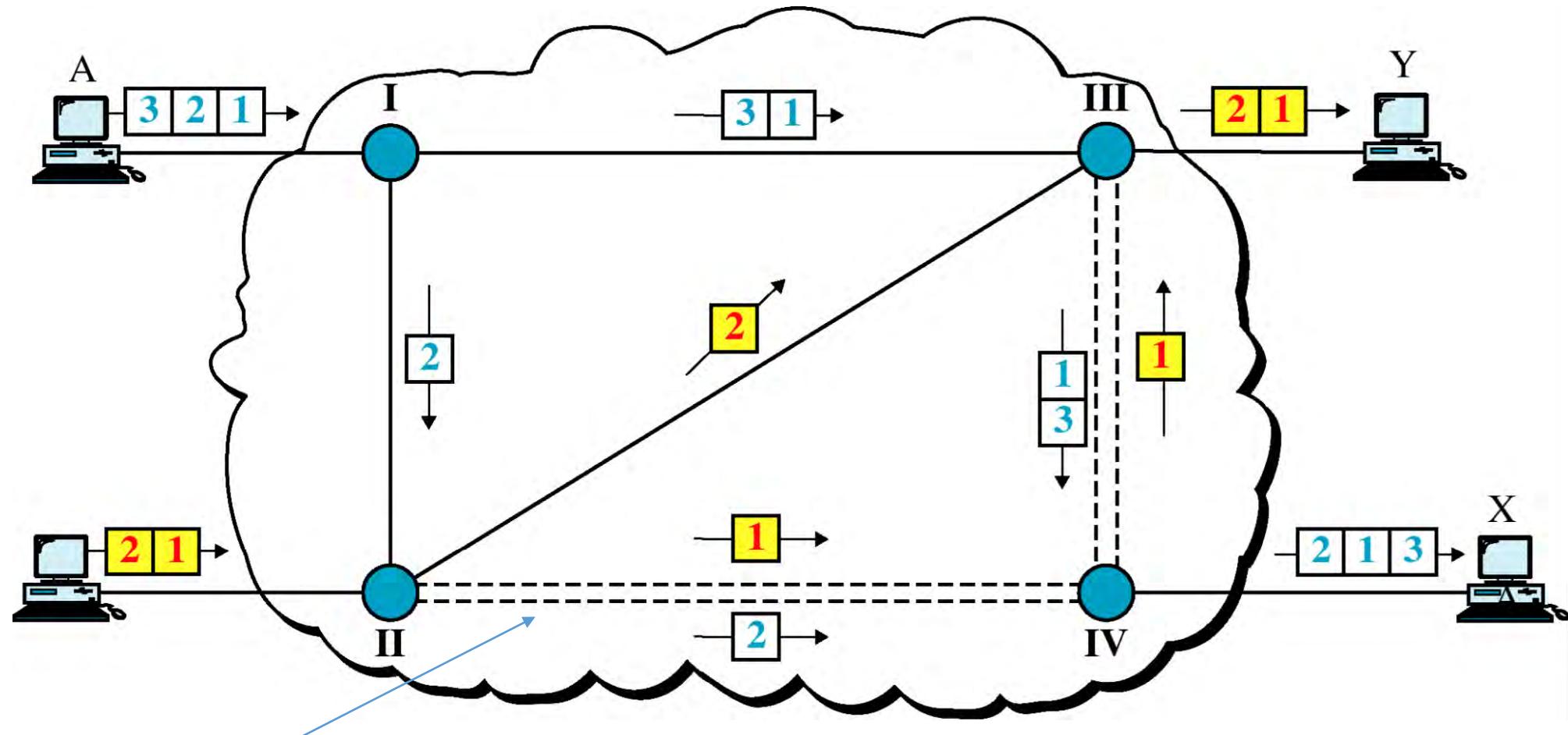
# Packet Switching Implementation

- Station breaks long message into packets
- Packets sent one at a time to the network
- Packets are handled in two different approaches
  - Datagram Service (Connection less)
  - Virtual circuit (Connection oriented)

# Packet Switching: Datagram Service

- In a datagram network, each packet is treated independently of all others, even if they belong to the same message.
- It is generally done at network layer.
- For each packet, each node makes its own decision as to how to forward it so that it eventually reaches its destination.
- Different packets may take different paths to reach to the same destination.
- Thus, reordering may required but at the destination only.
- It is possible for a packet to be destroyed if one of the nodes on its way is crashed momentarily. Thus all its queued packets may be lost.
- It is called ***connection less*** bcz setup or teardown phases are not required

# A datagram network with four switches(routers)



Duplex Channel

Amit Jha, SOEE, KIIT-ODU

# Datagram Switching: Key points

- Each packet contains a full destination address.
- Packets can take **any practical route**.
- Each packet is treated **independently**.
- Packets may arrive **out of order** (so receiver may require re-ordering)
- Packets may **go missing** (end node handles recovery of missing packets)
- *It is best effort network.*
- Store and forward operation required at each node for each packet
- Connection less protocol: Ethernet, IP, UDP
- Ex: Internet using IP, voice and video communication and notifying message to alert a user that he/she has received a new e-mail(using UDP)

*Q) If there are no setup or teardown phases, how are the packets routed to their destination in a datagram network?*

*Q) If there are no setup or teardown phases, how are the packets routed to their destination in a datagram network?*

**Ans:** In this type of network, each switch has a **routing table** based on destination address. The routing tables are dynamic and updated periodically.

# Packet Switching: Virtual-Circuit

- In this, a pre-planned route is established before any data packets are sent.
- A **logical(virtual)** connection is established when
  - A sender sends a “call request packet” to the receiver.
  - The receiver sends back an acknowledgement packet “call accepted packet” to the sender the receiver agrees on conversational parameters.
- The conversational parameter can be maximum packet size, path to be taken, and other variables necessary to establish and maintain the conversation.
- In virtual circuit, the route between stations does not mean that this is a dedicated path, as in circuit switching.

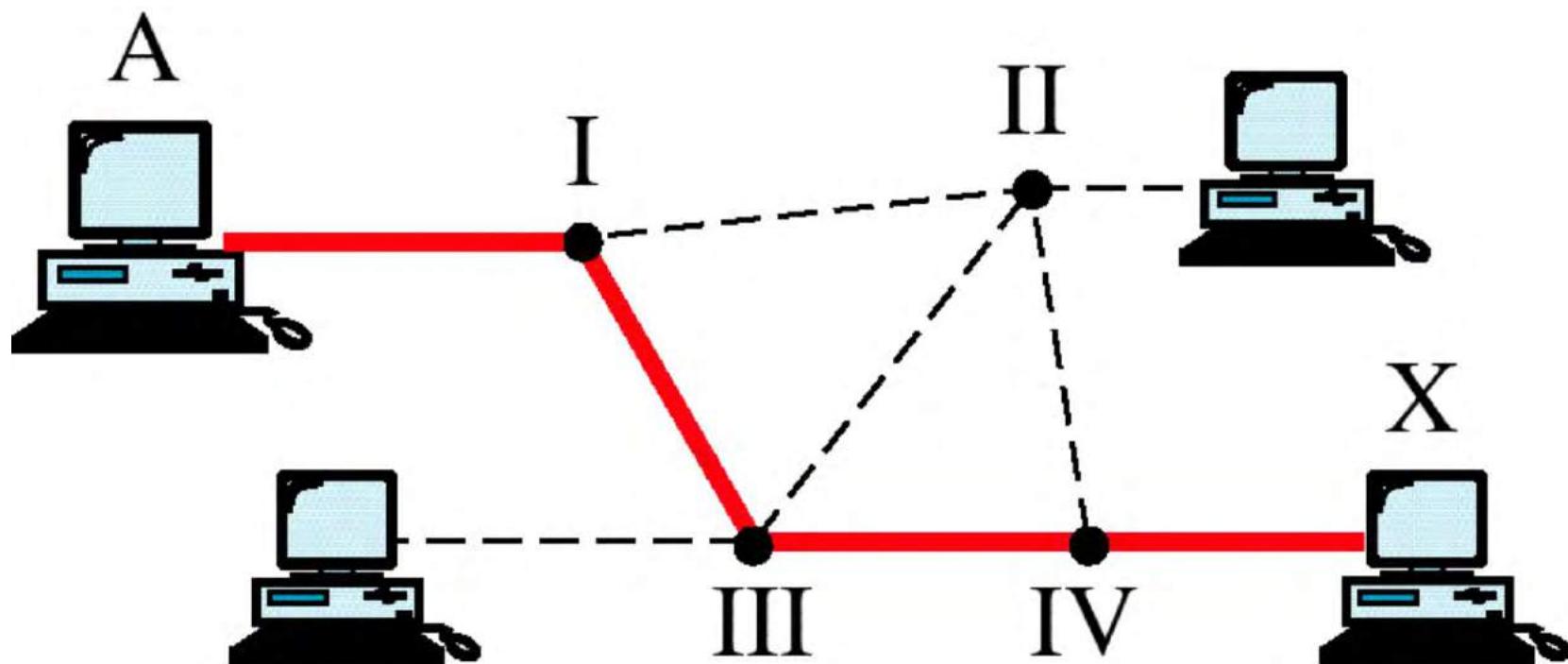
- In virtual circuit approach, routing decision is not made every time for all packets.

**Routing decision is made only once** for all packets using that virtual circuit.

Packets transfer involves basically three steps:

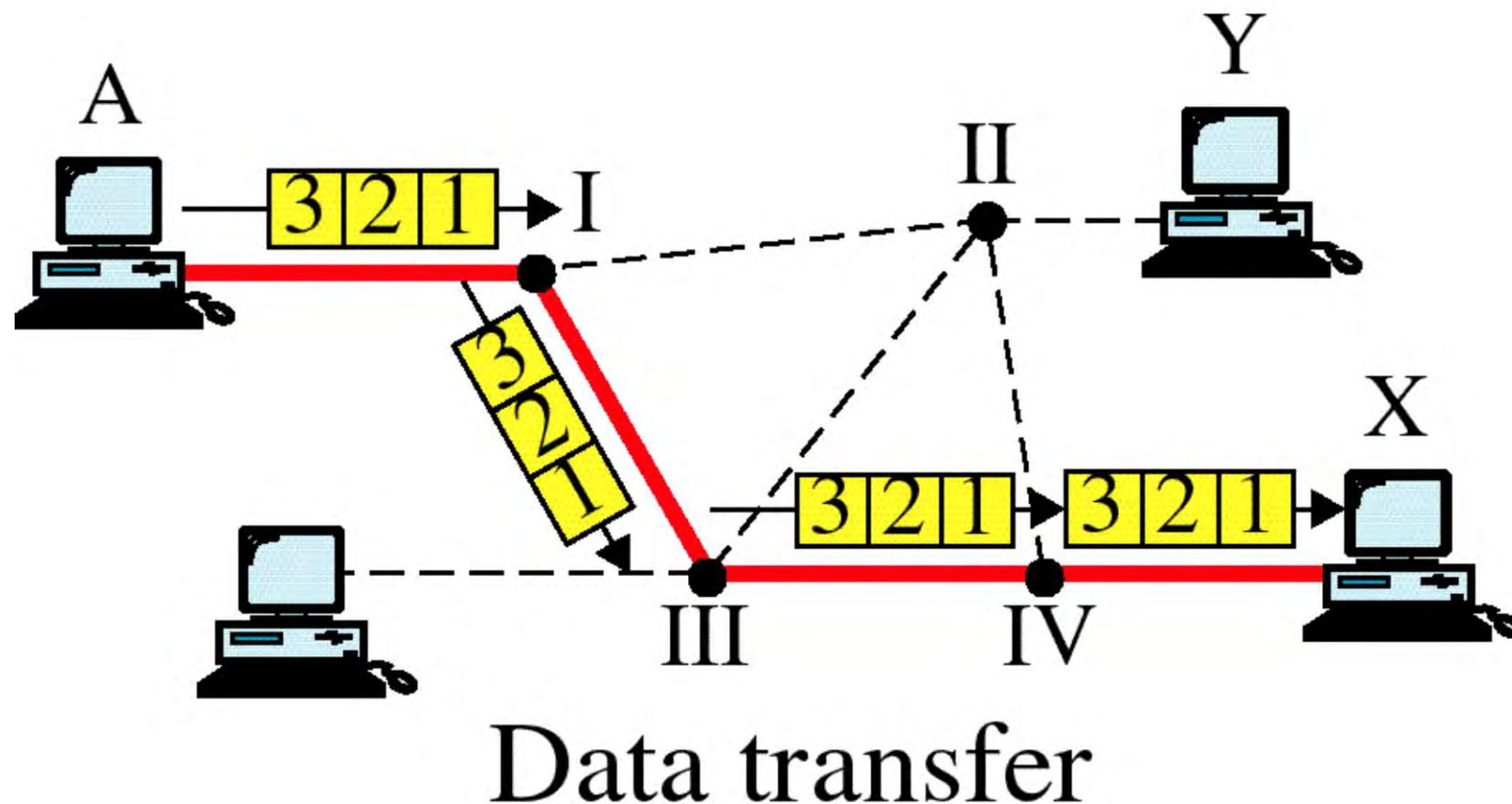
- Connection establishment
- Data transfer
- Connection release

# Step 1 of VC: Connection establishment

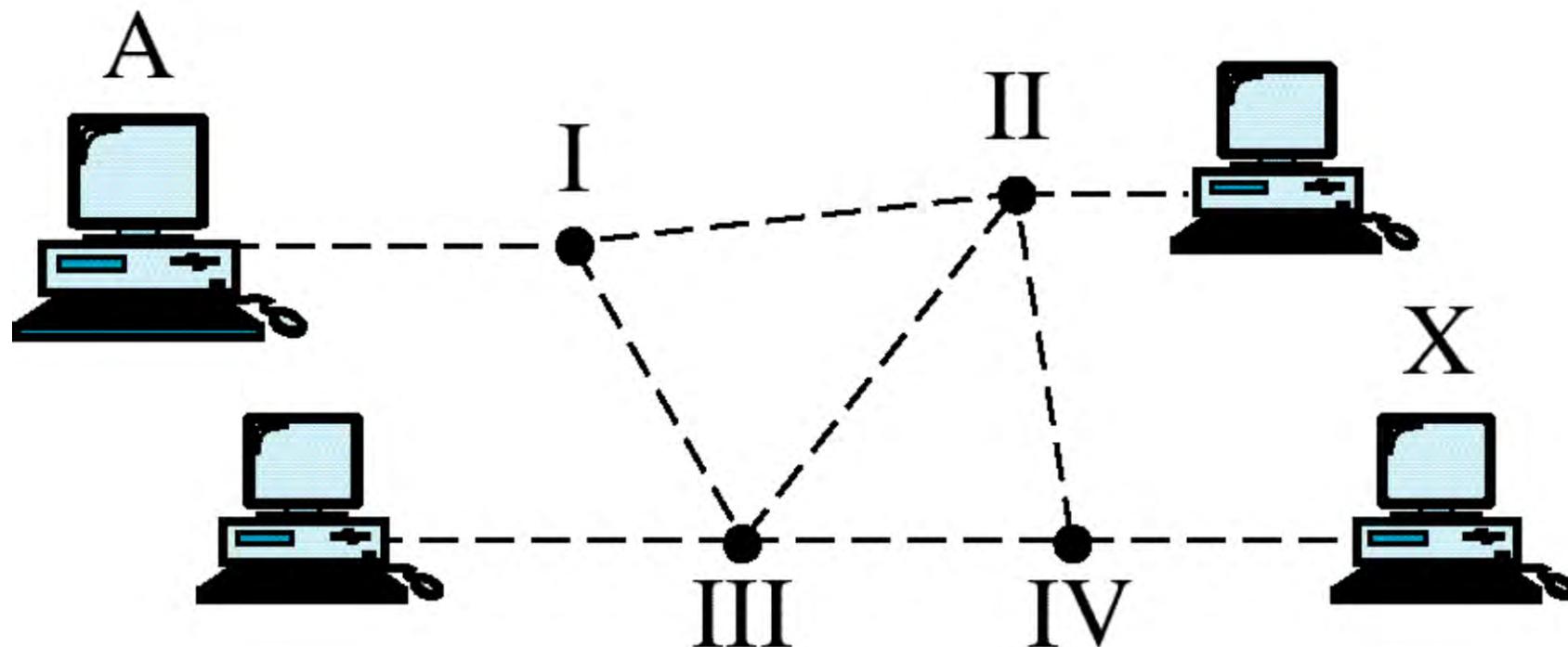


Connection establishment

## Step 2 of VC: Data Transfer



# Step 3 of VC: Connection Release



Connection release

# Virtual-Circuit: Key Points

- A packet is still buffered at each node and queued for output over line
- VC's offer guarantee that
  - The packets sent arrive in order at the destinations
  - No duplication of the packets or omission
  - No error
- **Routing decision is made only once** for all packets using that virtual circuit.
- Packets are forwarded more quickly as no routing decision to make every time.
- Intermediate nodes need to do store and forward in both the cases.

- **Advantages of Packet Switching**

- **Line efficiency:** maximize link efficiency by making optimal use of link bandwidth
- **Robust** against link and node failure
- Data rate conversion: Nodes buffer the data if required to equalize rates
- Packets are accepted even when network is busy
- Priority can be used
- Cost effective

- **Disadvantages of Packet Switching**

- Protocols for packet switching are typically **more complex**
- Packet switched system **still can't deliver the same quality** as delivered by circuit switched system.
- If packet is lost, sender needs to **retransmit** the data.
- It can add some initial cost in implementation.

# A comparison of circuit switched and packet switched(connection less) system

Item	Circuit-switched	Packet-switched
Call setup	Required	Not needed
Dedicated physical path	Yes	No
Each packet follows the same route	Yes	No
Packets arrive in order	Yes	No
Is a switch crash fatal	Yes	No
Bandwidth available	Fixed	Dynamic
When can congestion occur	At setup time	On every packet
Potentially wasted bandwidth	Yes	No
Store-and-forward transmission	No	Yes
Transparency	Yes	No
Charging	Per minute	Per packet

Circuit Switching	Virtual Circuit Packet Switching
Dedicated path is established before communication	No dedicated path
Message is sent as it is	Message is broken into small parts called <b>packets</b>
No Store-and-forward transmission	store-and-forward transmission
Processing delay is incurred at each node During call setup; but it is incurred during call accept signal as connection already has been setup	Processing delay is incurred during call setup; also during call accept because of store and forward technique
Constant data rate	Constant data rate is not guaranteed because of store-and-forward of packets at all nodes
Money is charged as per minute	Money is charged as per packets
No overhead bits after call setup	Overhead is present in each packet

# Performance Metrics for Packet Switched Network

- We are going to see only two performance metrics used to analyse the performance of a packet switched networks.
  1. Throughput
  2. Latency

**1. Throughput:** The throughput is a measure of how fast we can actually send data through a network.

**Can we say that bandwidth in bits per second and throughput are same?**

No, they are not same even they sound like.

- A link may have a bandwidth of  $B$  bps, but we can only send  $T$  bps through this link with  $T$  always less than  $B$ . In other words, the bandwidth is a potential measurement of a link; the throughput is an actual measurement of how fast we can send data.

**Confused ?**  
Amit Jha, SOEE, KIIT-DU



- For example, we may have a link with a bandwidth of 1 Mbps, but the devices connected to the end of the link may handle only 200 kbps. This means that we cannot send more than 200 kbps through this link.
- Imagine a highway designed to transmit 1000 cars per minute from one point to another. However, if there is congestion on the road, this figure may be reduced to 100 cars per minute. The bandwidth is 1000 cars per minute; the throughput is 100 cars per minute.

- **Ex.** A network with bandwidth of 10 Mbps can pass only an average of 12,000 frames per minute with each frame carrying an average of 10,000 bits. What is the throughput of this network?

- **Ex.** A network with bandwidth of 10 Mbps can pass only an average of 12,000 frames per minute with each frame carrying an average of 10,000 bits. What is the throughput of this network?
- **Sol:** Throughput=  $(12000 * 10000) / 60 = 2 \text{ Mbps}$

**Observation:** The throughput is almost **one-fifth** of the bandwidth in this case.

## 2. Latency (Delay)

- The latency or delay defines how long it takes for an entire message to completely arrive at the destination from the time the first bit is sent out from the source.
- Latency is made up of four components: *propagation time*, *transmission time*, *queuing time* and *processing delay*.

SO,

$$\text{Latency} = \text{propagation time} + \text{transmission time} + \text{queuing time} + \text{processing delay}$$

**1) Propagation Time:** Propagation time measures the time required for a bit to travel from the source to the destination.

$$\text{Propagation Time} = \frac{\text{Distance}}{\text{Velocity}}$$

**2) Transmission Time:** This is the time required to send a complete message.

$$\text{Transmission Time} = \frac{\text{Message size}}{\text{Bandwidth}}$$

**3) Queuing Time:** the time needed for each intermediate or end device to hold the message before it can be processed.

The queuing time is not a fixed factor; it changes with the load imposed on the network.

**4) Processing time:** Time required to process the message like error detection, acknowledgement, correction, etc).

# Practice Session to Chapter-01

**Ex 1:** What are the propagation time and the transmission time for a 2.5 Kbyte message (an e-mail) if the bandwidth of the network is 1 Gbps? Assume that the distance between the sender and the receiver is 12,000 km and that light travels at  $2.4 \times 10^8 \text{ m/s}$ .

# Practice Session to Chapter-01

**Ex 2:** A device is sending out the data at the rate of 1000 bps.

- a. *How long does it take to send out 10 bits?*
- b. *How long does it take to send out a character (8 bits)?*
- c. *How long does it take to send a file of 100,000 characters?*

# Practice Session to Chapter-01

**Ex 3:** A device is sending out the data at the rate of 1000 bps.

- a. *How long does it take to send out 10 bits?*
- b. *How long does it take to send out a character (8 bits)?*
- c. *How long does it take to send a file of 100,000 characters?*

# Practice Session to Chapter-01

**Ex 4:** A file contains 2 million bytes. How long does it take to download this file using a 56 Kbps channel (dial-up modem)?

# Practice Session to Chapter-01

**Ex 5:** What is the transmission time of a packet sent by a station if the length of the packet is 1 million bytes and bandwidth of the channel is 200 Kbps.

# Practice Session to Chapter-01

**Ex 6:** What is the length of a bit in a channel if the propagation speed in the medium is  $2 \times 10^8$ m/s and the channel bandwidth is

- a.  $1 \text{ Mbps}$  ?
- b.  $10 \text{ Mbps}$  ?

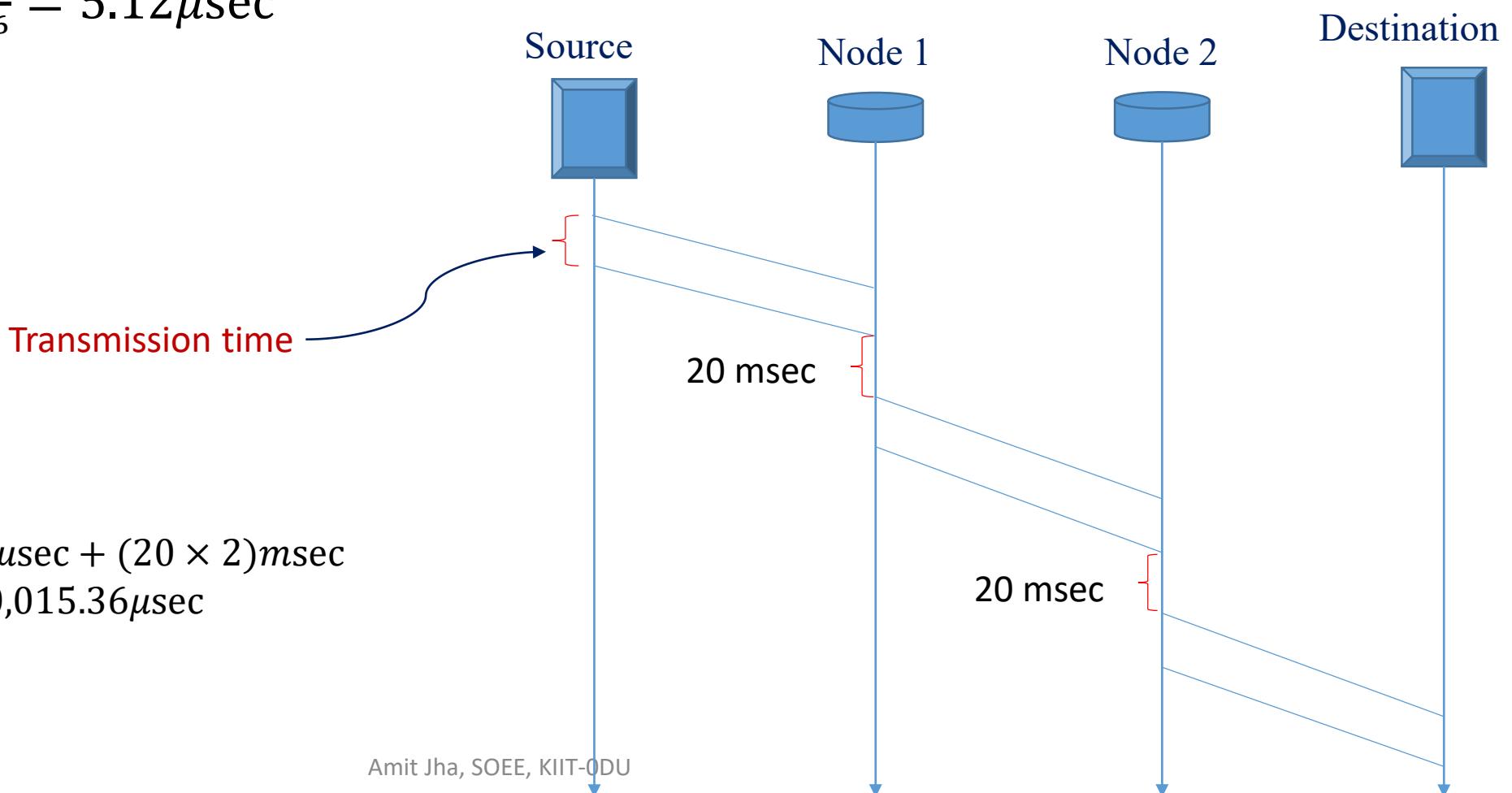
# Practice Session to Chapter-01

**Ex 7:** Consider a packet switched network. A 64 bytes Packet is sent over a 100 Mbps link. There are two intermediate nodes and the waiting time at each node is 20 msec. Calculate the total time required by the packets to reach the destination.

## Solution to Ex.7:

Propagation delay is not given, so assume it as negligible.

$$\text{Transmission Time} = \frac{64 \times 8}{100 \times 10^6} = 5.12 \mu\text{sec}$$



So, total time required is

$$\begin{aligned} T &= 3 \times 5.12 \mu\text{sec} + (20 \times 2) \text{msec} \\ &= 40,015.36 \mu\text{sec} \end{aligned}$$

# CN (IT-3001)

## Application Layer

Prof. Amit Jha  
School of Electronics Engineering (SOEE)  
KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

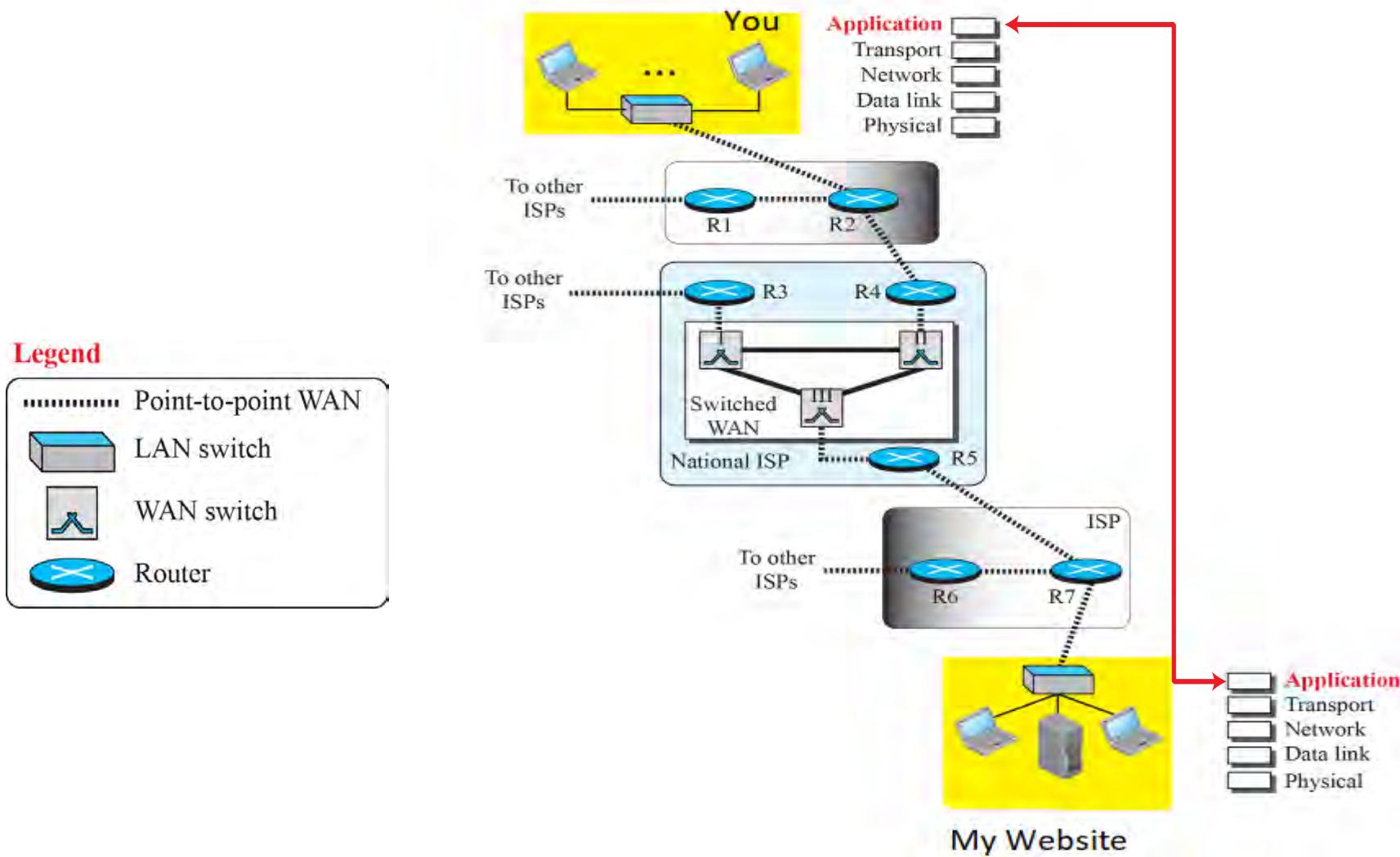
# Content

- Responsibility of Application Layer
- Logical Connections at the Application layer
- Different Application Layer Paradigms
  1. Client-Server Paradigm
  2. Peer-to-Peer Paradigm
  3. Hybrid
- Application Programming Interface (API)
- Socket address and its significance
- Transport Layer Keypoints

# Responsibility of Application Layer

- Application layer provides **services** to users.
- Communication is provided using a **logical connection**, which means that the two application layers assume that there is an imaginary connection through which they communicate the data.
- The communication at application layer is logical and **not physical**.
- However, actual communication takes place through several devices and several physical channels as shown in the next slide.
- Protocols at this layer **do not provide services** to any other protocols in the suite; they only receive services from the protocols in the transport layer. This Means
  - Protocols can be **removed** from this layer easily.
  - Also, a new protocol can be **added** to this layer easily as long as the new protocol can utilize the service provided by any of the transport layer protocols → TCP, UDP and STCP.

# Logical Connection at the Application Layer



# Application Layer Paradigm

- In the Internet, **two application programs should interact** with each other in order to communicate the data:
  - One running on a computer somewhere in the world;
  - The other running on another computer somewhere else in the world.
- Now, the question arises:
  - Should **both** application programs to be able to **request** services?
  - Should **both** application programs to be able to **receive** services?
  - Should **one** application programs to be able to request services whereas other should provide service or vice-versa?

# Application Layer Paradigm

- In the Internet, **two application programs should interact** with each other in order to communicate the data:
  - One running on a computer somewhere in the world;
  - The other running on another computer somewhere else in the world.

Now, the question arises:

- Should **both** application programs to be able to **request** services?
- Should **both** application programs to be able to **receive** services?
- Should **one** application programs to be able to request services whereas other should provide service or vice-versa?
- To answer this question, following three application layer paradigm have been developed:
  1. Traditional Paradigm: Client-Server
  2. New Paradigm: Peer-to-Peer
  3. Hybrid

# Client-Server Paradigm

- Most popular until few years ago.
- In this we have two systems:

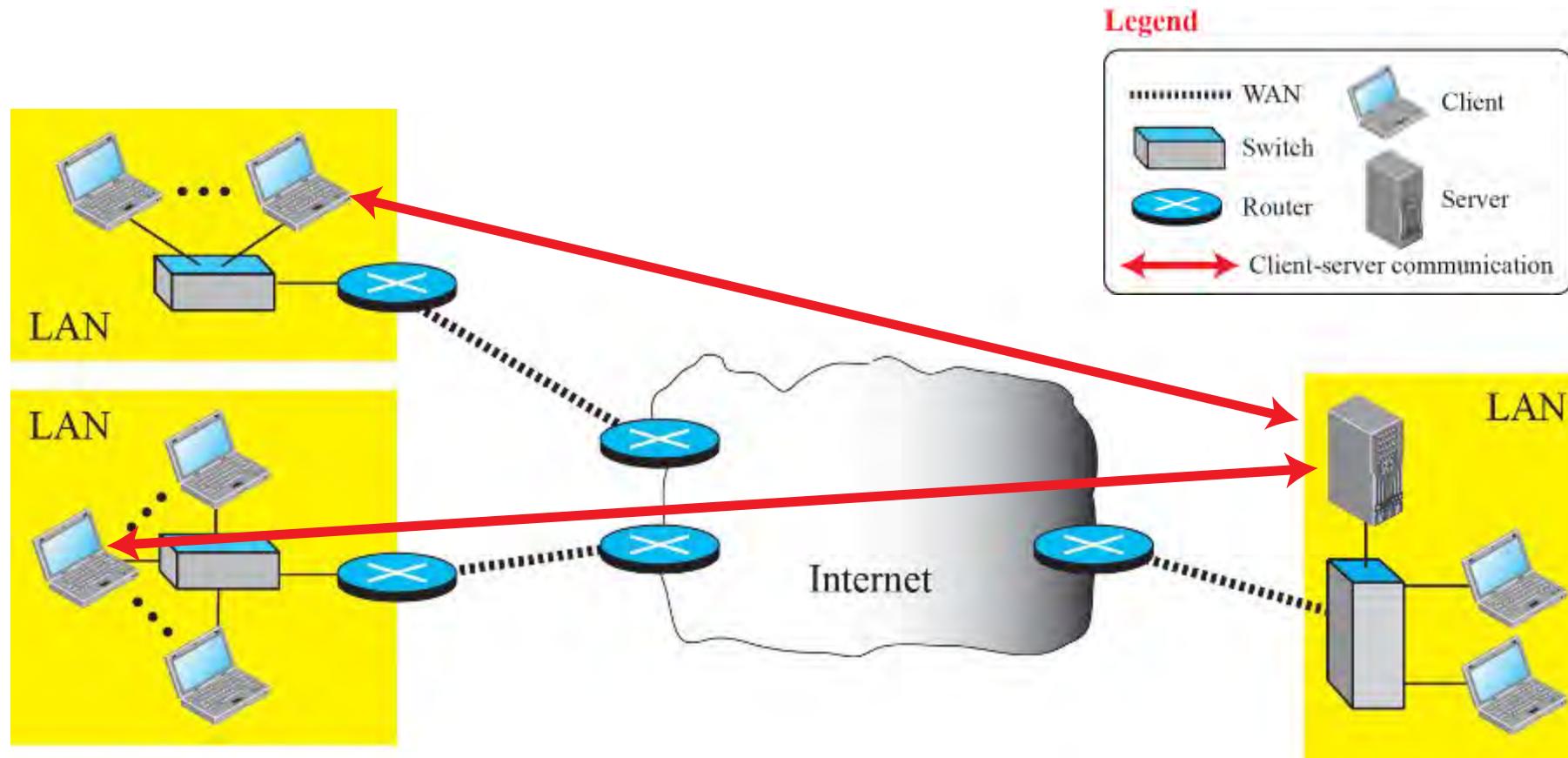
## 1. Client:

- It request the server for the connection and communication.
- It does not run all the time, runs only when it needs to receive service.

## 2. Server

- It runs continuously, waiting for another application program, called client.
- It runs all the time.
- **Examples:** World Wide Web (www), Hyper Text Transfer Protocol (HTTP), File Transfer Protocol (FTP), Secure Shell (SSH), e-mail, etc.

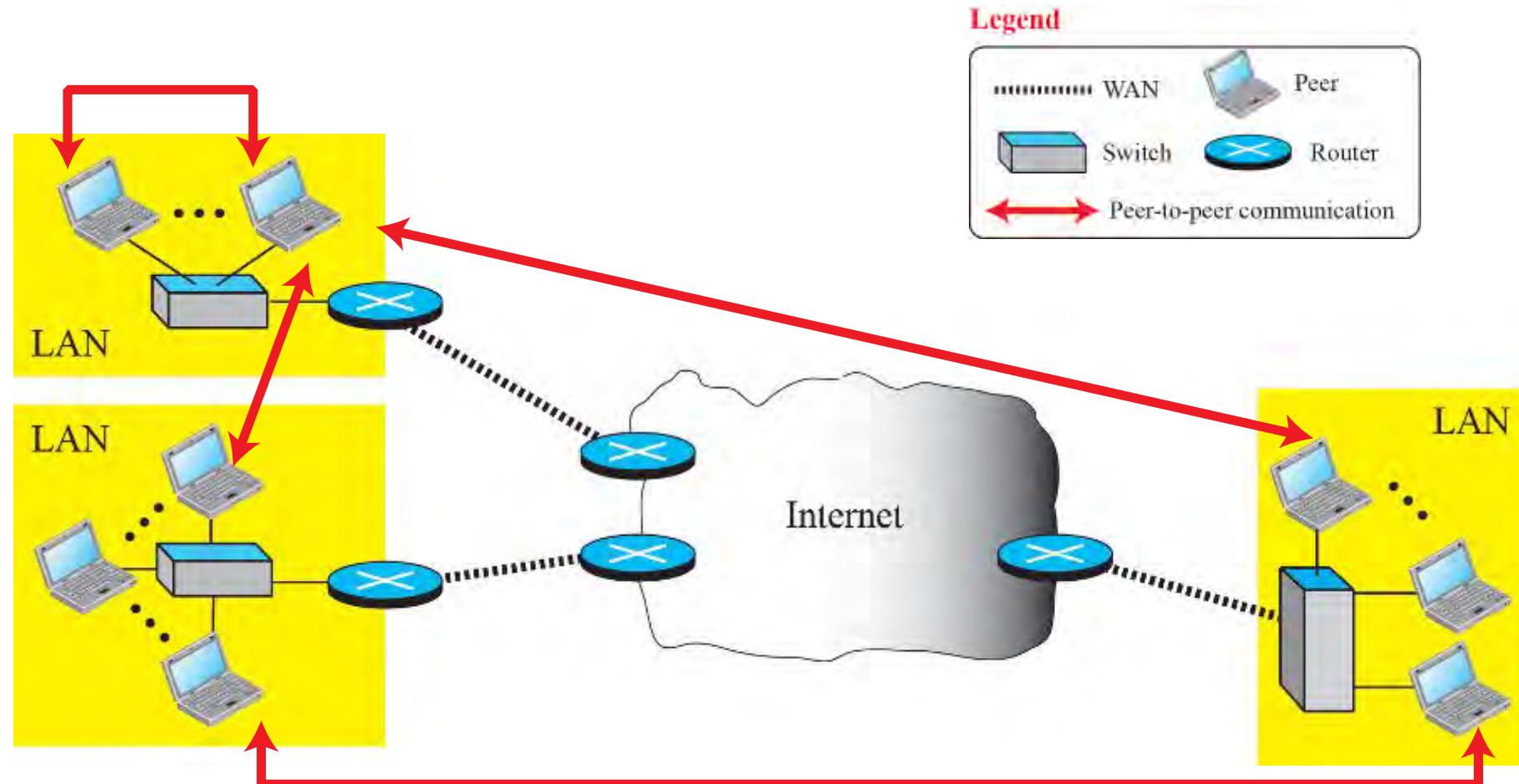
# Client-Server Paradigm: Illustration



# Peer-to-Peer Paradigm (P2P)

- Most popular now a days.
- **In this,**
- there is no need for a server process running all the time and waiting a client process to connect.
- The responsibility is shared between peers.
- A computer connected to the Internet can either
  - Provide service at one time and receive at another time **or**
  - Can provide and receive the service at the same time.
- **Examples:** Internet telephony, BitTorrent, Skype, IPTV, etc.

# Peer-to-Peer Paradigm (P2P): Illustration

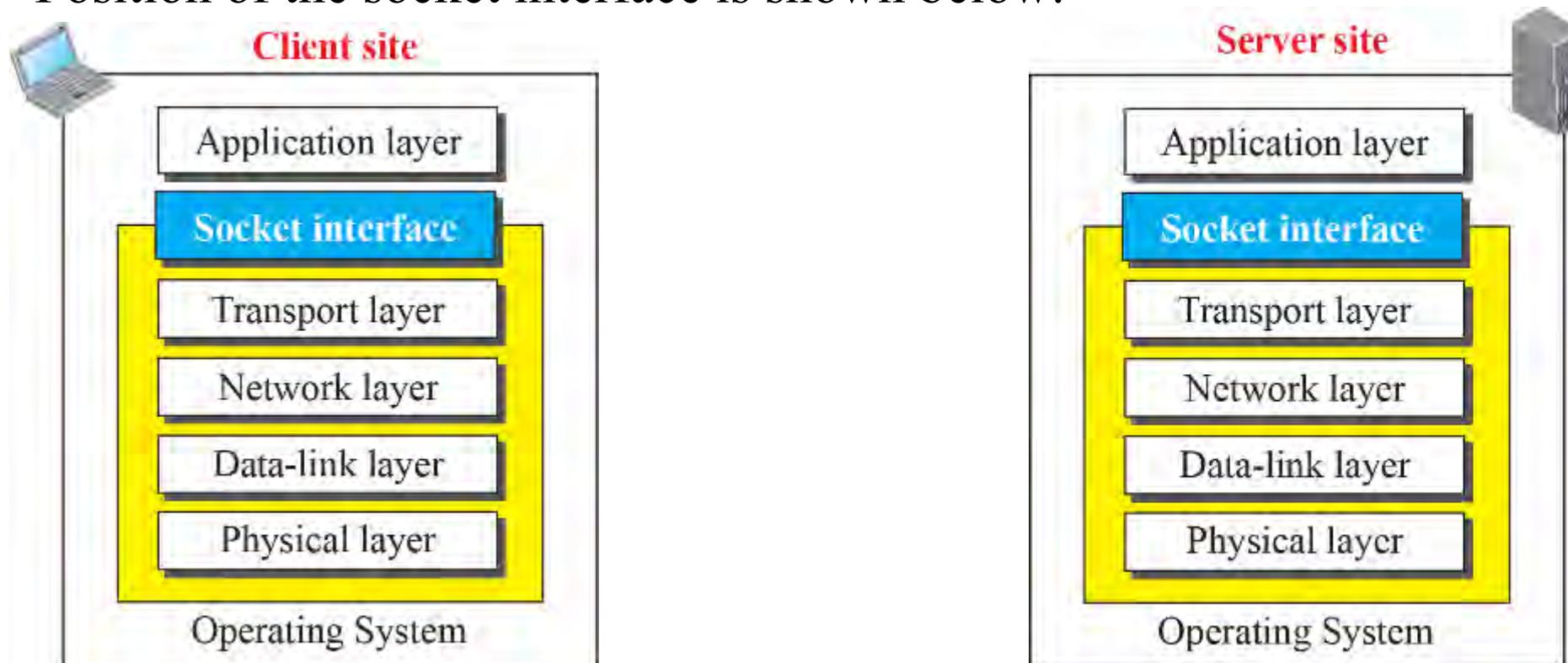


- How can a client process communicate with a server process ?

- How can a client process communicate with a server process ?
- Using **Application Programming Interface** (API).
  - It is used to represent a set of instructions to tell the lowest four layers of TCP/IP suite to open the connection, send and receive the data from the other end, and close the connection.
  - **Interface:** An interface in programming is a set of instructions between two entities.
  - In this case, one of the entities is the
    - **process at the application** layer and other is the
    - **operating system** that encapsulates the first four layers of the TCP/IP suite.
  - In other words, a computer manufacturer needs to build the first four layers of the suite in the operating system and include an API.
  - In this way, the processes running at the application layer are able to communicate with the operating system when sending or receiving message through the Internet.
  - **Few examples:** socket interface, Transport Layer Interface (TLI) and STREAM.

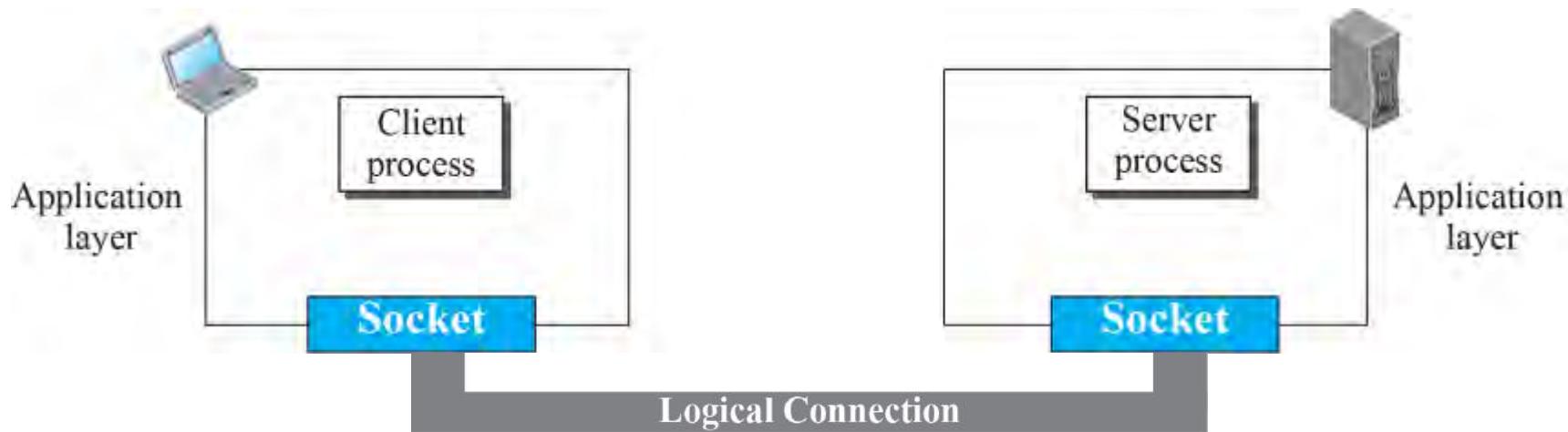
# Sockets

- It is a ***data structure*** created and used by the application program.
- Position of the socket interface is shown below:



# Sockets

- From the application layer point-of-view, the communication between client process and server process takes place through the sockets, created at two ends as shown below.



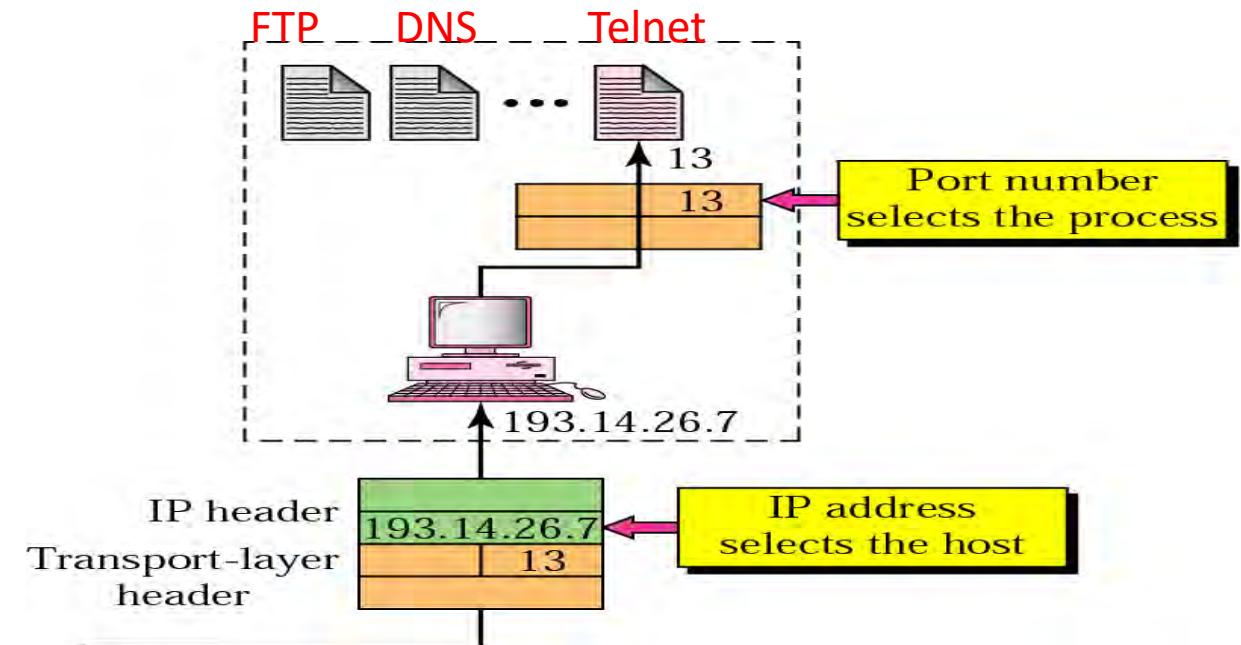
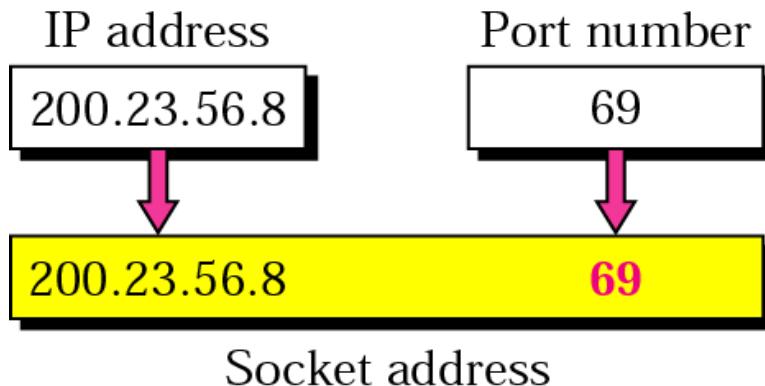
# Sockets Address

- In two-way communication, we need a pair of addresses:
  - Local (sender) address
  - Remote (receiver) address
- Note: local address in one direction is remote address in other direction and vice-versa.
- Socket address is combination of **32-bit IP address** (to uniquely identify the device) and **16-bit port number** (to identify the process) as shown below.



# Sockets Address

- In two-way communication, we need a pair of addresses:
  - Local (sender) address
  - Remote (receiver) address
- Note: local address in one direction is remote address in other direction and vice-versa.
- Socket address is combination of **32-bit IP address** (to uniquely identify the device) and **16-bit port number** (to identify the process) as shown below.



# Sockets Address: Key Points

- ***Port number:*** It is 16-bit integers between 0 to 65,535.
- **Client port number:** Client chooses its port number randomly from 0 to 65,535 using the transport layer software running on the client host. This is called ***ephemeral(temporary)*** port number.
- **Server port number:** Server process is also defined by a port number. But, its not randomly chosen. If it is random, then client will not know the port number in order to access that server. These are called ***well-known port number.***

# Transport Layer Protocols: Only Key Points

- Three protocols are defined at the transport layer:
  1. TCP
    - Connection-oriented
    - Reliable → supports flow and error control mechanism
    - Supports full-duplex
    - It is a byte-stream oriented protocol
  2. UDP
    - Connection less
    - unreliable → no flow and error control mechanism
  3. STCP
    - Provides service which is a combination of TCP and UDP.
    - Connection-oriented
    - Reliable
    - But it is not byte-stream oriented protocol. It is message oriented protocol like UDP.

# CN (IT-3001)

## Application Layer: Client-Server Paradigm

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Content

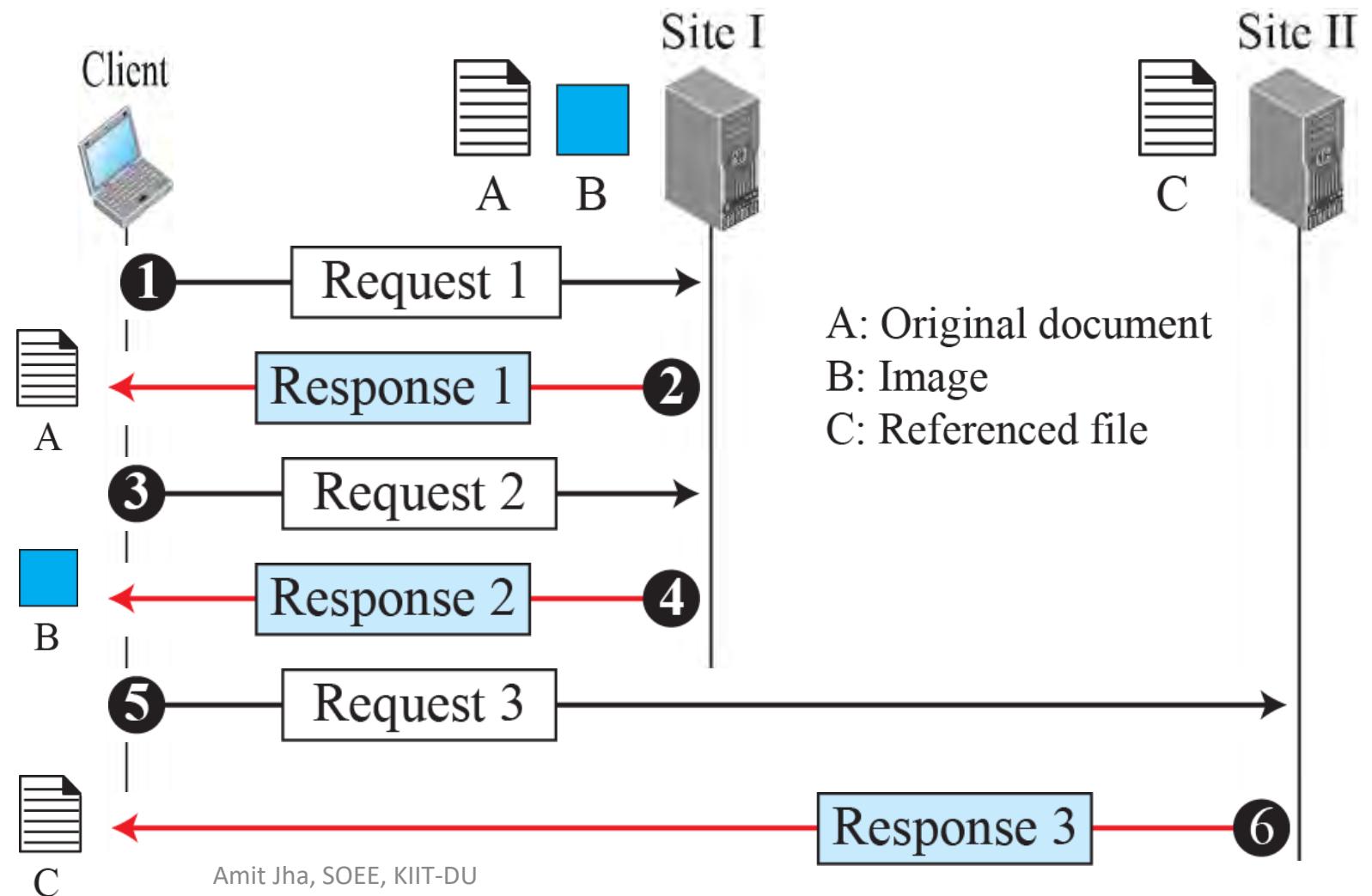
- Different application of standard **client-server paradigm** such as
  1. Web
  2. HTTP
  3. FTP
  4. E-mail
  5. DNS
- Different application of standard **Peer-to-Peer paradigm** such as
  1. P2P network
  2. P2P Application: BitTorrent

# Client-Server Application: WWW

- It is a distributed client-server service, in which a client using a browser can access a service using a server.
- However, the service provided is distributed over many locations called *sites*.
- Each site holds one or more documents, referred to as web pages.
- A web page can be simple or composite:
- Simple web page → No link to other web pages
- Composite web page → link to other web pages

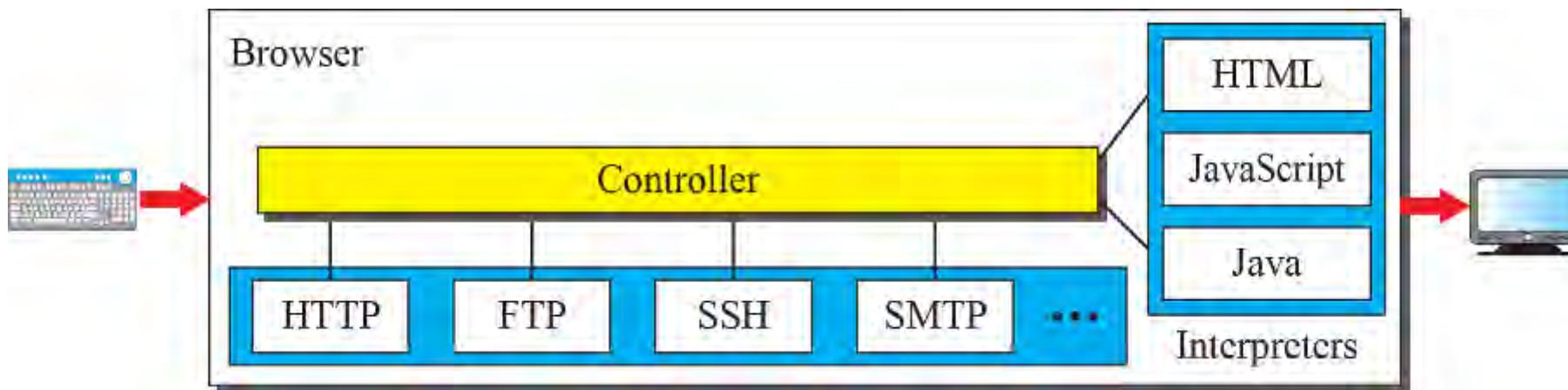
# WWW: Example

- File A, B and C are **independent Web pages**.
- **Links** for web page B and C are included in A.
- However, B and C can also be reached **independently**.



# WWW: key elements

- **Web Client (Browser):** Its responsibility is to interpret and display a web page.
- Each browser usually consists of three parts:
  1. **Controller:** It receives input from the keyboard and mouse and uses the client program to access the documents. After the documents are accessed, the controller uses one of the interpreters to display the document on the screen.
  2. **Client Protocol:** Client protocols are like HTTP, FTP, SSH, etc.
  3. **Interpreter:** It can be HTML, Java, or JavaScript depending upon the type of documents.



# WWW: key elements

- **Web Server:** The web pages are stored at the servers which are expected to run for all the time. To improve the efficiency, a cache memory is used generally. Ex. **Apache** and **Microsoft Information Server**.
  - **Uniform Resource Locator (URL):** It is an identifier used to distinguish a web page from another.
    - It has three fields: *host, port and path*.
    - However, before defining the web page, we need to tell the browser what client-server application we want to use.
    - Thus, we now have **four identifiers** (instead of three) to uniquely identify a web page.
1. **Protocol:** Its like a **vehicle** to be used to access the web page. E.g., HTTP, FTP, etc.
  2. **Host:** It is IP address or unique name of the server.
  3. **Port:** It is 16-bit integer which is normally pre-defined for client-server based application.  
For e.g., if HTTP is used, then port no is 80 (well-known). However, if different port is used, it should be clearly written while accessing the web page.
  4. **Path:** It is location of the file in underlying operating system.

Example:

- **protocols://host/path** → used for most of the time
- **protocols://host:port/path** → used when port number is needed

Practical Example

<https://sites.google.com/view/amitkumarvjha/home>  
<https://192.168.101.1:8090/httpclient.html>

# WWW: key elements

- **Web Documents:** It can be grouped into three broad categories: static, dynamic and active.

## 1. Static Documents:

- It is fixed-content documents that are created and stored in a server.
- The content of the file is decided when it is created.
- Static documents are prepared using: **HTML, XML, XSL, XHTML**

## 2. Dynamic Documents:

- It is created by a web server whenever a browser requests the documents.
- Dynamic documents are prepared using: **Java Server Page (JSP), Active Server Page (ASP), ColdFusion.**

## 3. Active Documents:

- For many applications, we need a program or a script to be run at the client site. These are called active documents.
- **Java applets**, is used to create such documents. **JavaScript** can also be used, but it needs to download and then run.

# Client-Server Application: HTTP

- The HTTP is a protocol that is used to define how the client-server programs can be written to retrieve web pages from the web.
- The **server uses port no 80**; whereas client uses a temporary port number.
- HTTP uses TCP service of the transport layer which is reliable, and thus;
  - Client and server do not need to worry for the error in message exchanged or loss of the message as it will be taken care by the TCP.
- I have a doubt...
  - **How a file on the web page will be accessed?**



- **How a file on the web page will be accessed?**
- **Answer:**
  - It is accessed by **establishing** a connection (TCP Connection) between client and server.
  - The data is retrieved based on request and response method. Where,
    - Client sends request and
    - Server responds to the client's request
  - As it is TCP, so once data transmission is over, the connection has to be **released**.

Can you answer these doubts!

1. If object is located on the different servers, do we need to establish a TCP connection to each server ?
2. If objects are located on the same server, do we need to establish a TCP connection to the server
  1. Each time to access the object?
  2. Only once to access all the objects on the server?

Can you answer these doubts!

1. If object is located on the different servers, do we need to establish a TCP connection to each server? → Yes, we have to!
2. If objects are located on the same server, do we need to establish a TCP connection to the server
  1. Each time to access the object? → yes, this way is also possible  
→ It is known as **nonpersistent method**
  2. Only once to access all the objects on the server? → yes, this way is also possible  
→ It is known as **persistent method**

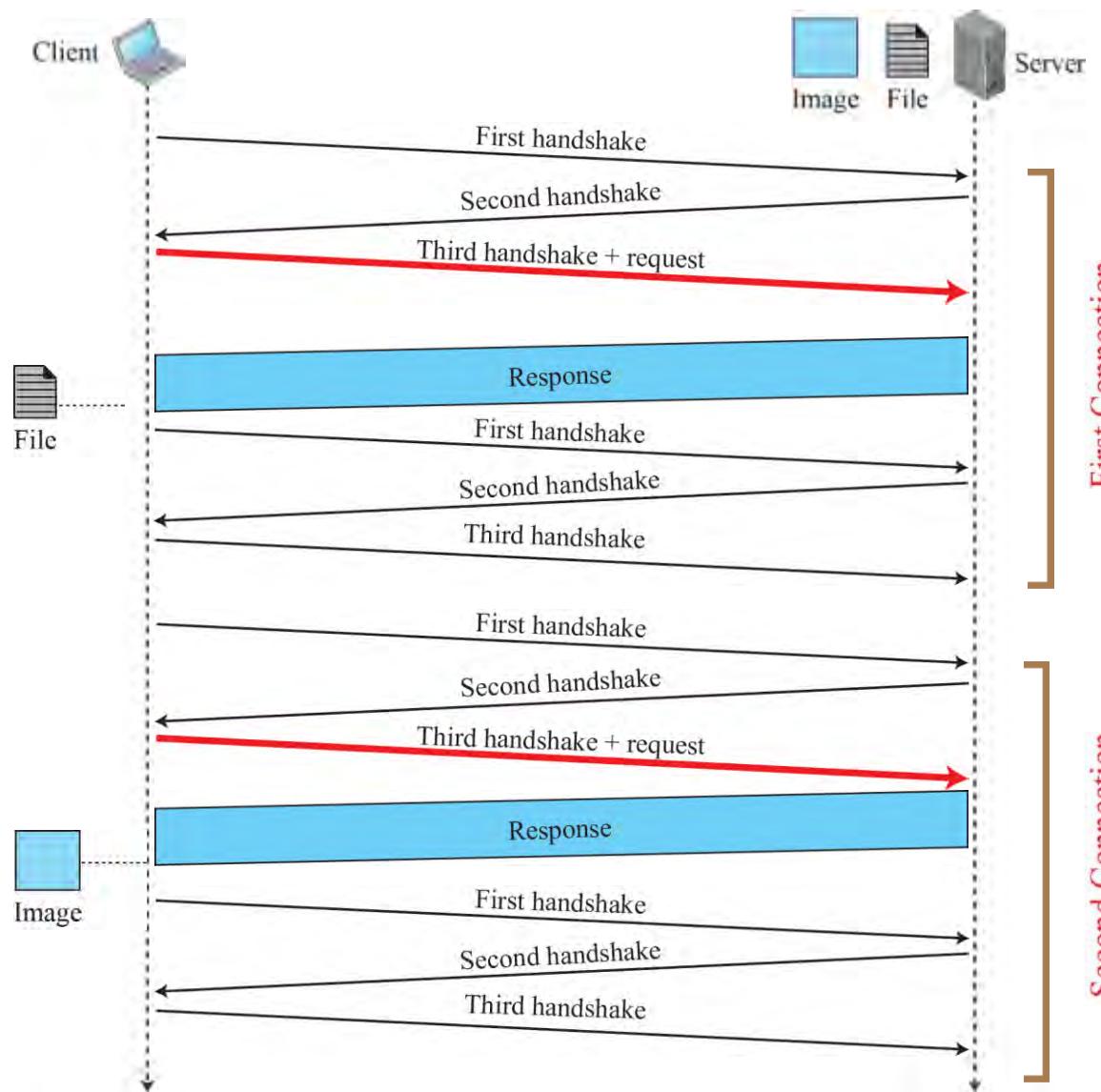
- **Nonpersistent connection:**

- In this one TCP connection is made for each request/response in following stages.
  - Client opens a TCP connection and sends a request.
  - The server sends the response and closes the connection.
  - The client then reads the data until it encounters and end-of-file marker, it then closes the connection.

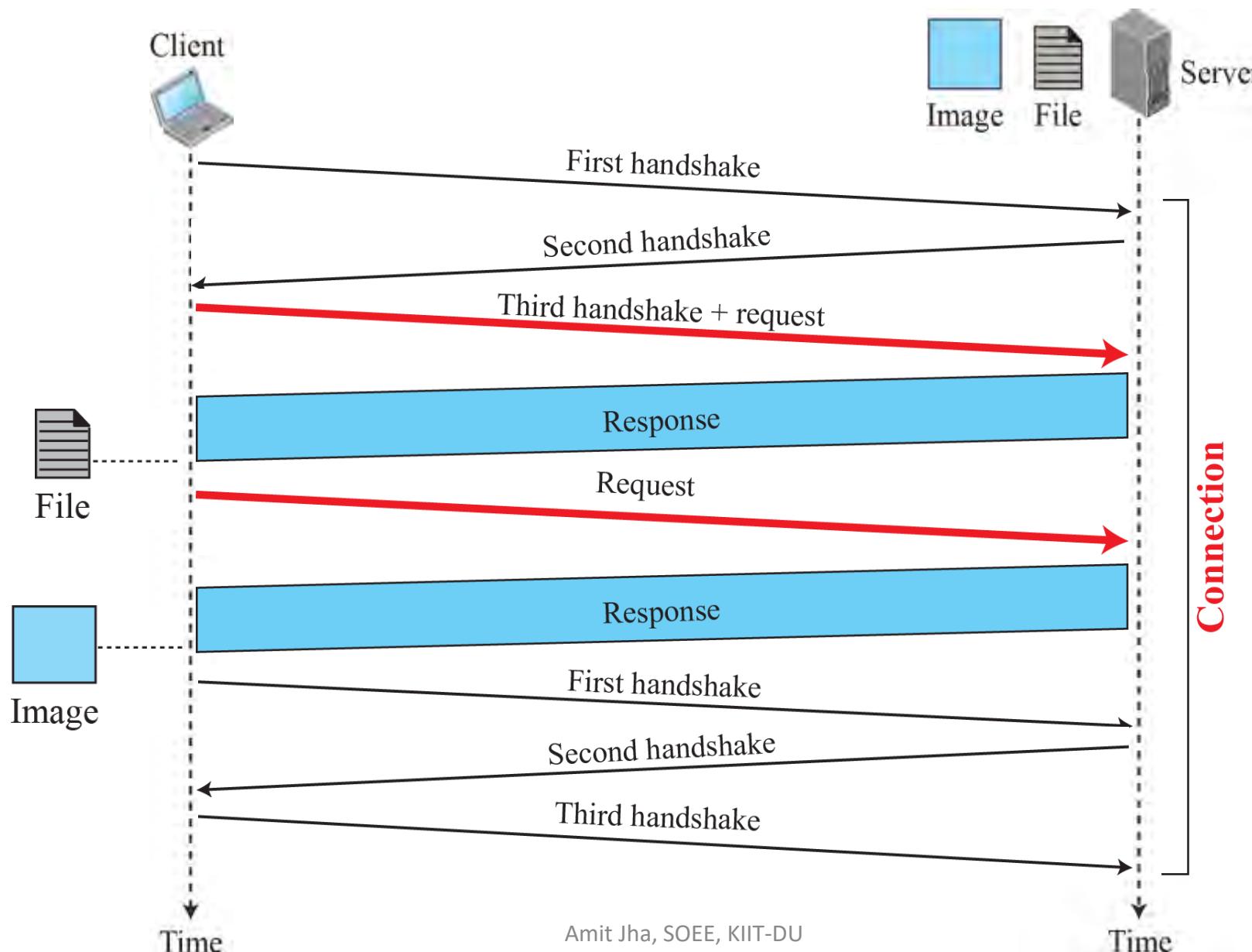
- **Persistent connection:**

- In persistent connection, the server leaves the connection open for more requests after sending a response.
- The server can close the connection either
  - at the request of a client or
  - If a time-out has been reached

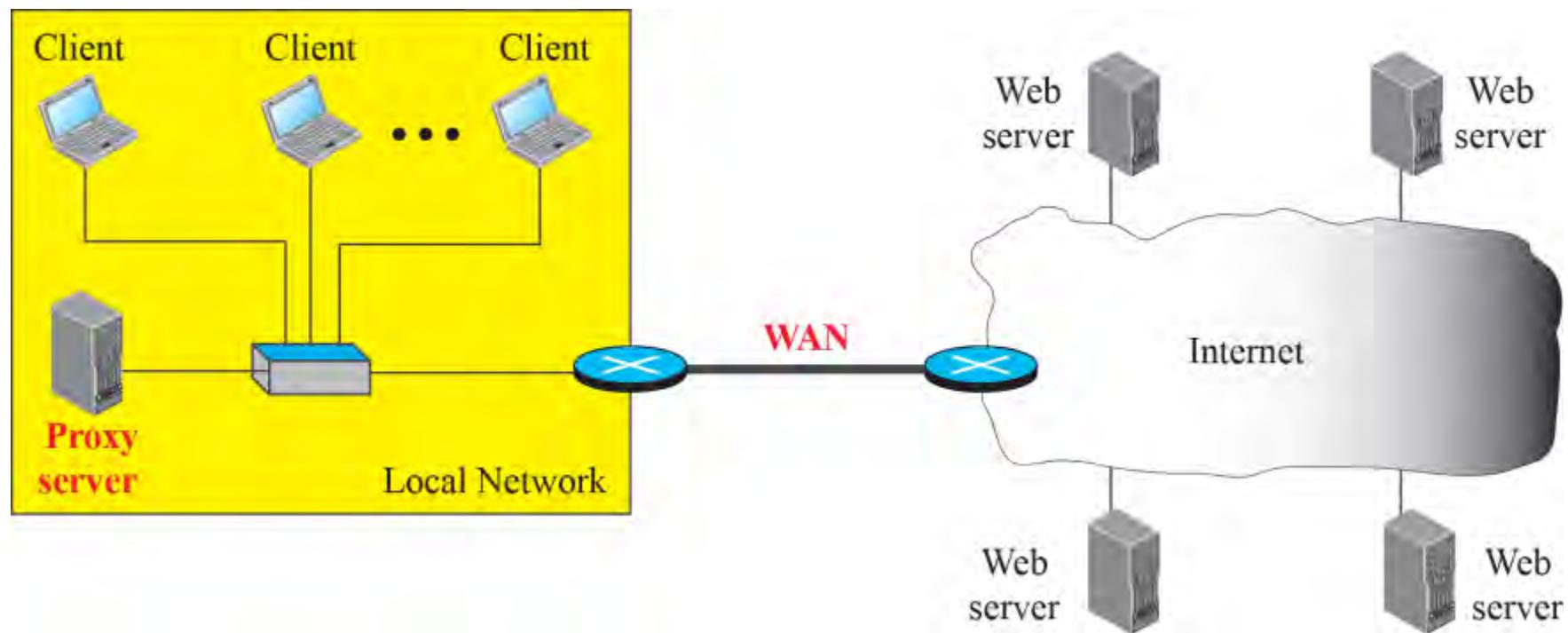
## Scenario of Nonpersistent communication to get a file and an image from the same server



## Scenario of persistent communication to get a file and an image from the same server



# Proxy Server



- The proxy server is installed in the local network.
- When an HTTP request is created by any of the clients (browsers), the request is first directed to the proxy server.
- If the proxy server already has the corresponding web page, it sends the response to the client.
- Otherwise, the proxy server acts as a client and sends the request to the web server in the Internet.
- When the response is returned, the proxy server makes a copy and stores it in its cache before sending it to the requesting client.

# Hone Your Understanding (HYU)

1. What is cache and cookies?
2. Why do we need it?
3. What are the advantages?
4. How does it work?
5. Are they same like proxy server?
6. How cookies are maintained at both client and server to behave the HTTP as a state-full protocol?

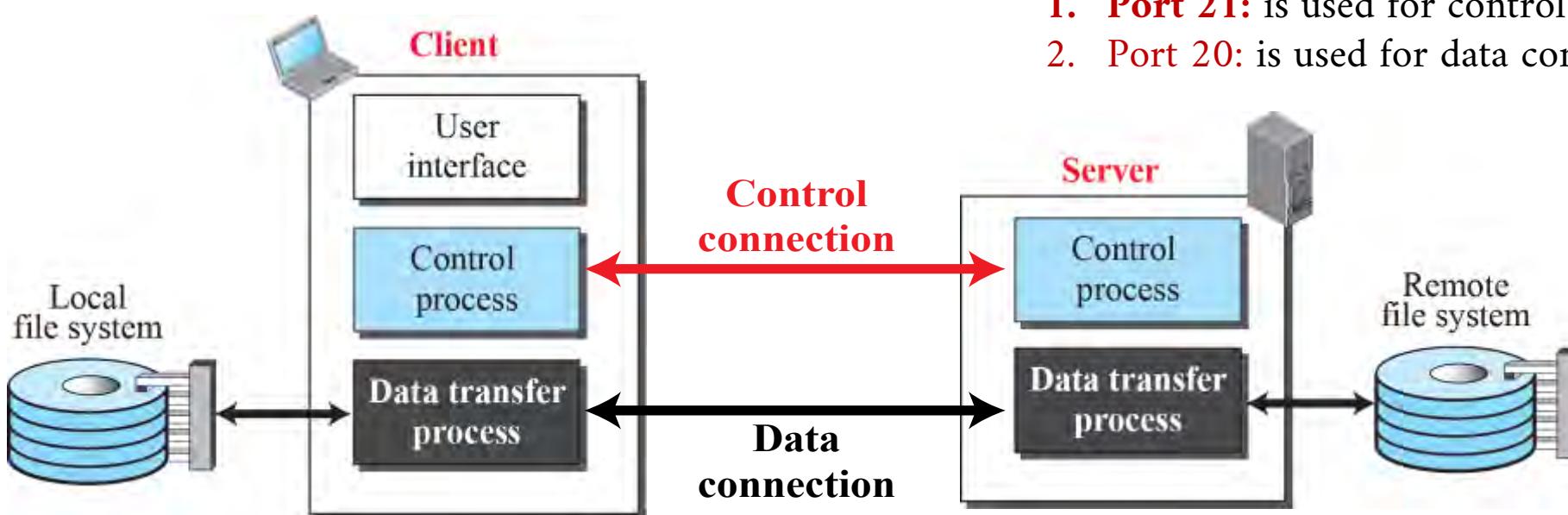
# Client-Server Application: **FTP**

- It is standard protocol provided by TCP/IP to copy a file from one host to another.
- Although HTTP can be used to transfer the file; FTP is better choice for large file or file with different formats.
- FTP is not secure as data transfer takes place in the form of plain text, which is insecure.
- Although, FTP requires a password, the password is sent in plaintext (**unencrypted**), which means it can be intercepted.
- For security, one can add Secure Socket Layer (SSL) between the FTP application layer and TCP layer. Here this is called **SSL-FTP**.
- FTP can transfer → ASCII file, EBCDIC file, image file, etc.

## Basic Model of FTP

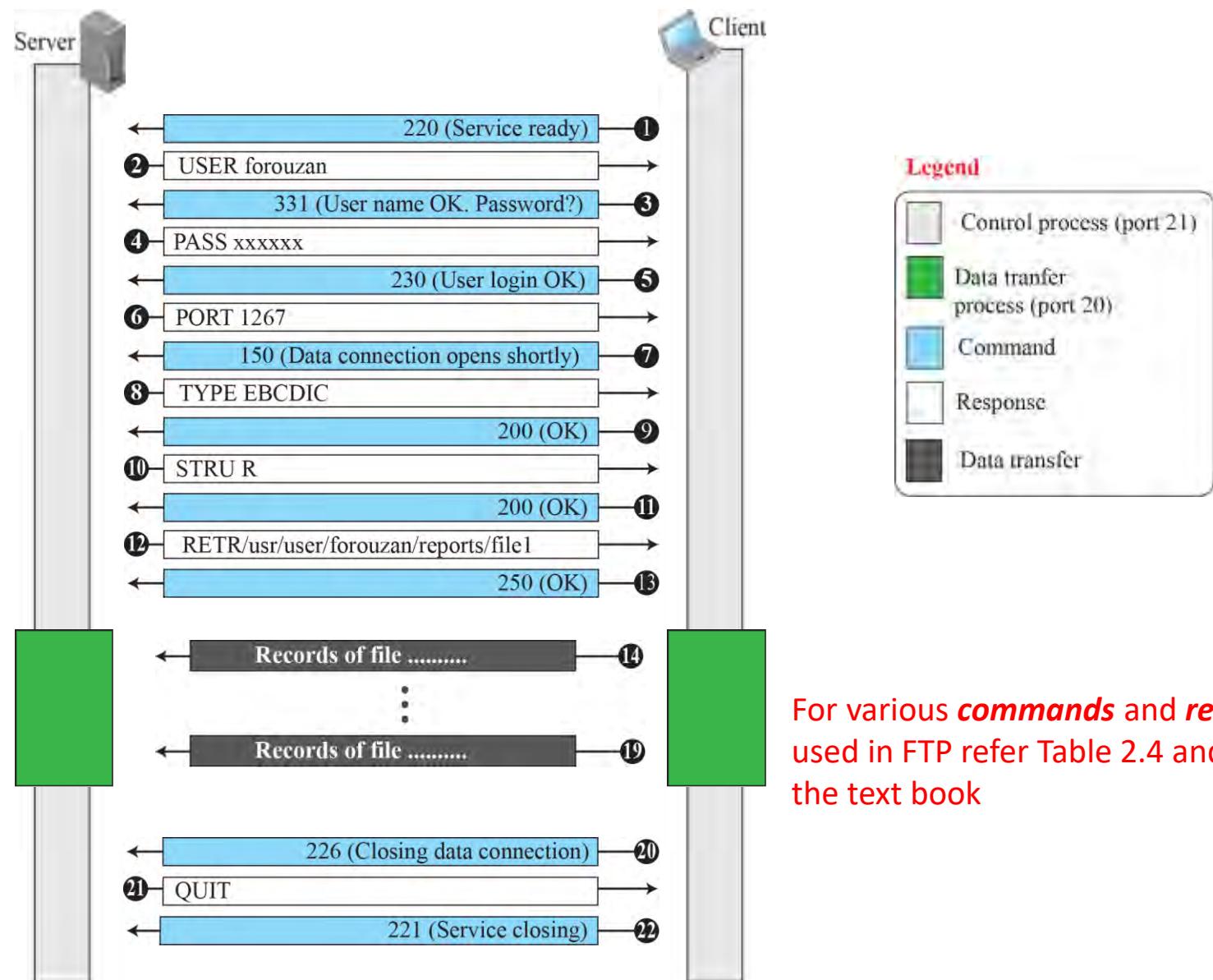
- The client has three components:
  1. User interface
  2. Client control process → remains connected for entire FTP session
  3. Client Data transfer process → it is opened and then closed for each file transfer activity
- The server has two components:
  1. Server control process
  2. Server Data transfer process

**Note:** FTP uses two well-known ports of TCP,  
1. **Port 21:** is used for control connection  
2. **Port 20:** is used for data connection



*An example of using FTP for retrieving a file (server to client).*

- *# of files to be transferred:* 1
- *Control connection:* remains open all the time
- *Data connection:* is opened and closed repeatedly.
- *File transfer:* We assume the file is transferred in six sections.
- After all records have been transferred, the server control process announces that the file transfer is done.
- Since the client control process has no file to retrieve, it issues the QUIT command, which causes the service connection to be closed.



For various **commands** and **responses** used in FTP refer Table 2.4 and 2.5 of the text book

# Client-Server Application: Email

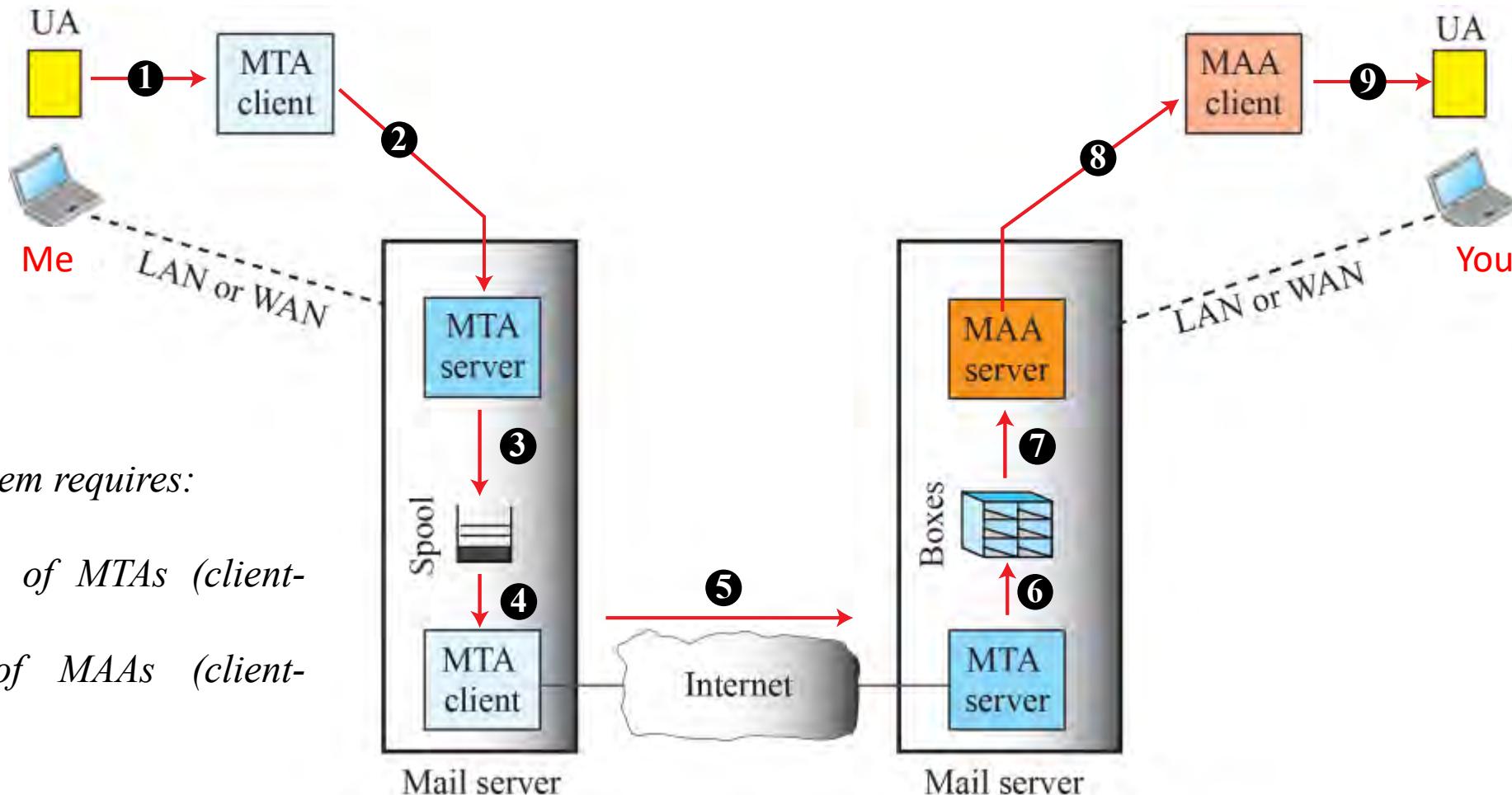
- In HTTP or FTP, the server program is running all the time. When a request comes to the server, server provides service to the client → Typical Client-server paradigm.
- Does e-mail work in the same fashion?
  - For e.g., if I (**client**) send you an email expecting a reply from you (**server**), then is it mandatory for you to reply me back?

# Client-Server Application: Email

- In HTTP or FTP, the server program is running all the time. When a request comes to the server, server provides service to the client → Typical Client-server paradigm.
- Does e-mail work in the same fashion?
  - For e.g., if I (**client**) send you an email expecting a reply from you (**server**), then is it mandatory for you to reply me back?
  - **Answer** → is no, it is not mandatory for you to reply me back. i.e., your system should not be running all the time like a typical server.
- So, in email, the idea of client-server programming is implemented by different way: → using some **intermediate computers (servers)**.
- Hence, the user runs only client programs when they want and the intermediate servers apply the client-server paradigm.

## Communication Architecture of email application

UA: user agent  
MTA: message transfer agent  
MAA: message access agent



Note:

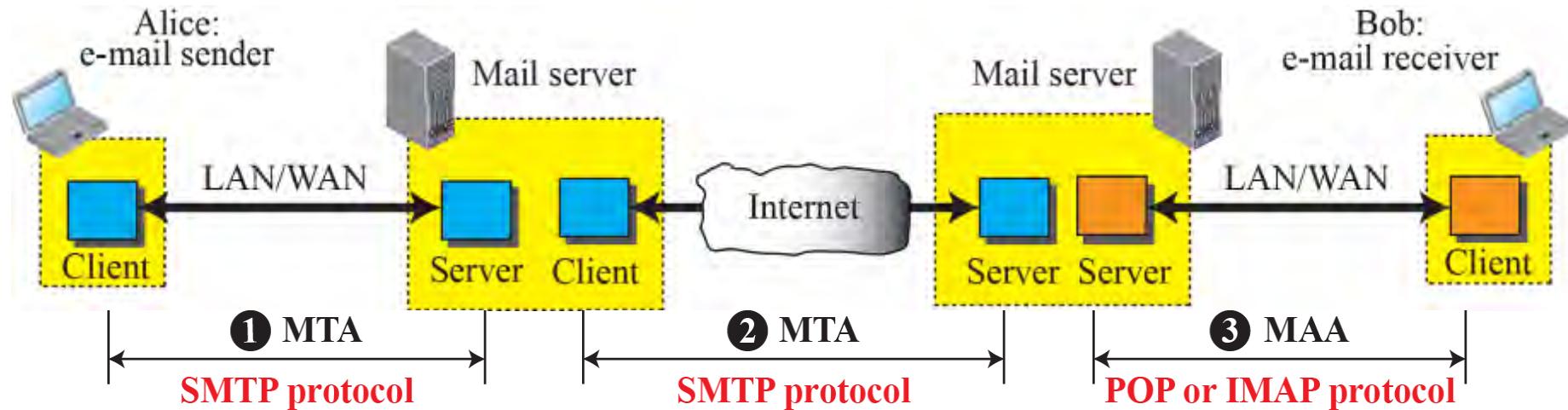
The email system requires:

1. Two UAs
2. Two pairs of MTAs (client-server)
3. A pair of MAAs (client-server)

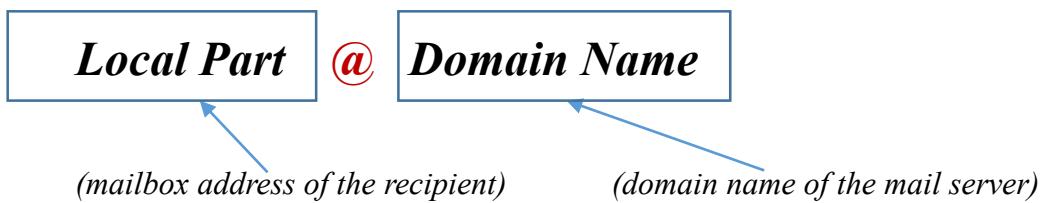
- I have some doubts !!
  1. Can you bypass the mail server and receive my email directly?
  2. Why I need MTA as client server program; whereas you need MAA client-server program?

- *SMTP*: used two times,
  1. Between sender and sender's mail server
  2. Between two mail servers
- *POP and IMAP*: used only one time

## Protocols used in email



*Format of email address*



## Protocols used in email

### 1. Simple Mail Transfer Protocol (SMTP):

- SMTP is message transfer agent.
- It is a push protocol, which pushes the message from client to server.
- It simply defines how commands and responses must be sent back and forth.
- There are several standard commands and responses → refer textbook, page no 67 and 68 for details.

### 2. Post Office Protocol (POP): currently POP3 → 3<sup>rd</sup> version

- POP is message access agent
- It is a pop protocol, which pulls the message from server to the client.
- The client POP3 software is installed on recipient computer, whereas, the server POP3 software is installed on mail the server.

### 3. Internet Mail Access Protocol (IMAP): currently IMAP4 → 4<sup>th</sup> version

- It is similar to POP3, but more powerful and complex than POP3.

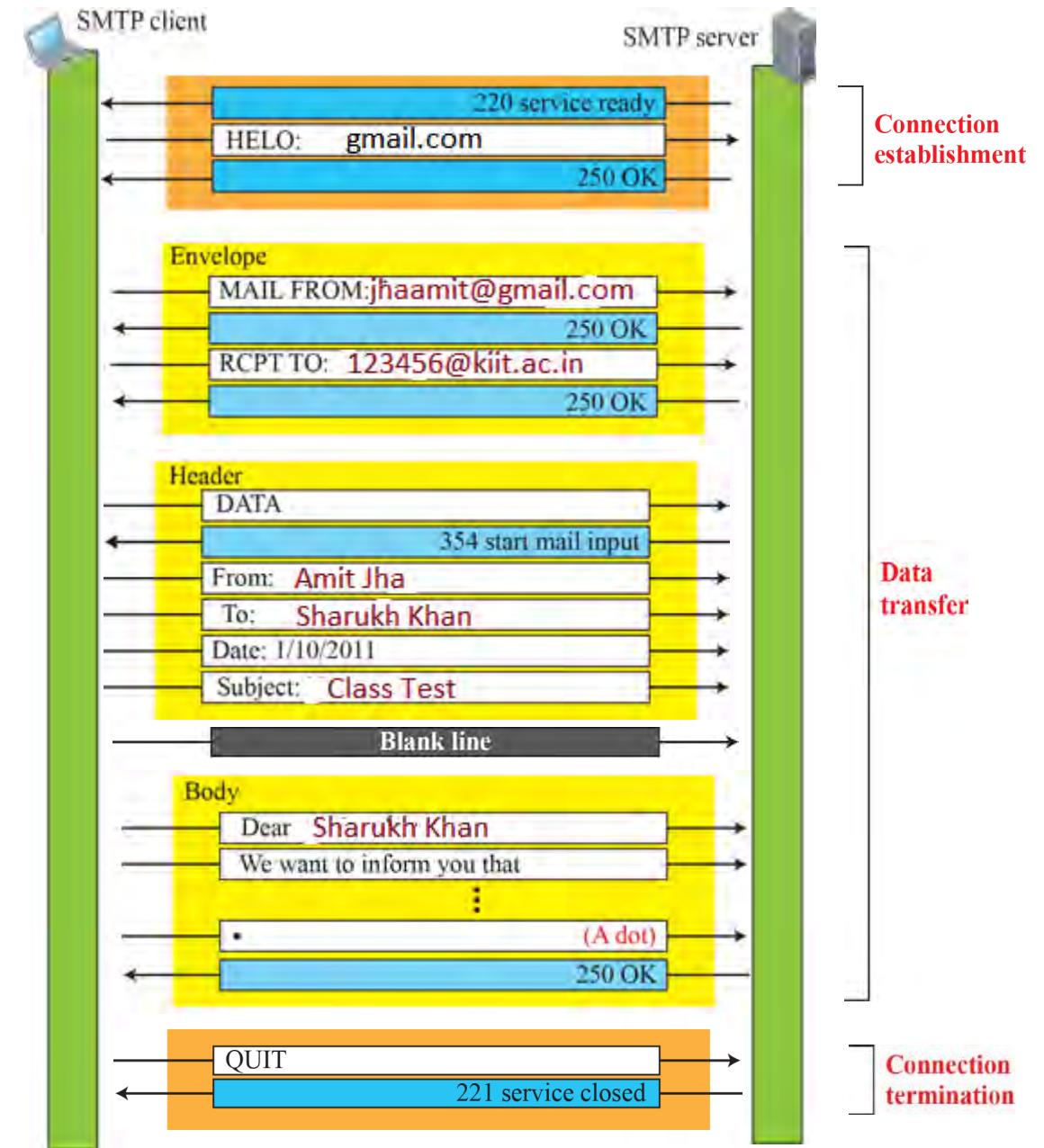
POP3	IMAP4
User can not have different mail folder on the mail server	User can create, delete, rename mailboxes on the mail server
Does not allow user to partially check the content of the mail	Allows user to <b>check email header, search content of email prior to downloading, partial downloading email.</b>
Cannot create hierarchy of mailboxes in a folder for email storage	

# Mail Transfer Phase

- The process of transferring a mail occurs in following three phases:
  1. *Connection Establishment*: first client makes a TCP connection to well-known **port 25**, then SMTP server starts the connection phase.
    1. The Server sends **code 220 (service ready)**.
    2. The client sends **HELO** message to identify itself using a domain name address.
    3. The server responds with **code 250 (request command completed)** or some other code depending upon the situation.
  2. *Message Transfer*: It takes total 8 number of steps for message transfer
    1. Refer the next diagram or page 69 of the text book.
  3. *Connection Termination*: happens in two phases
    1. The Client sends the **QUIT** command
    2. The Server responds with code **221** or some other appropriate code.

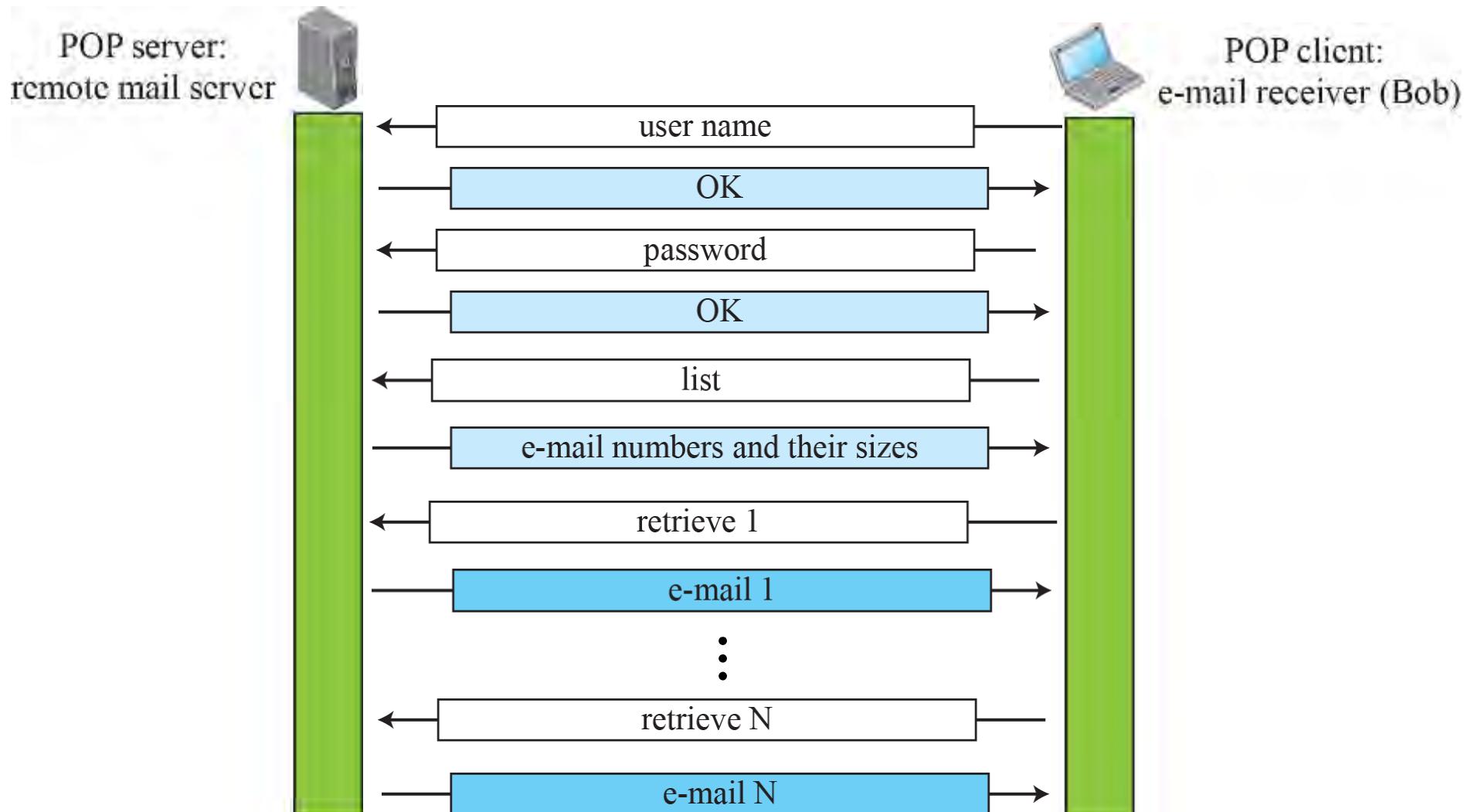
## *An example of using SMTP for mail transfer.*

- Note: This process will be happened two times:
  - 1<sup>st</sup>, to transfer email sender to local mail server.
  - 2<sup>nd</sup> , from local mail server to remote mail server
- Note: different commands and responses can be seen from Table 2.6 and 2.7 of the text book.



## Use of POP3 to receive the mail at the receiver

Messages are pulled



- I have some doubts !!
  1. Can we send email in language other than English?
  2. Can we send binary files, audio files, or video files using email?

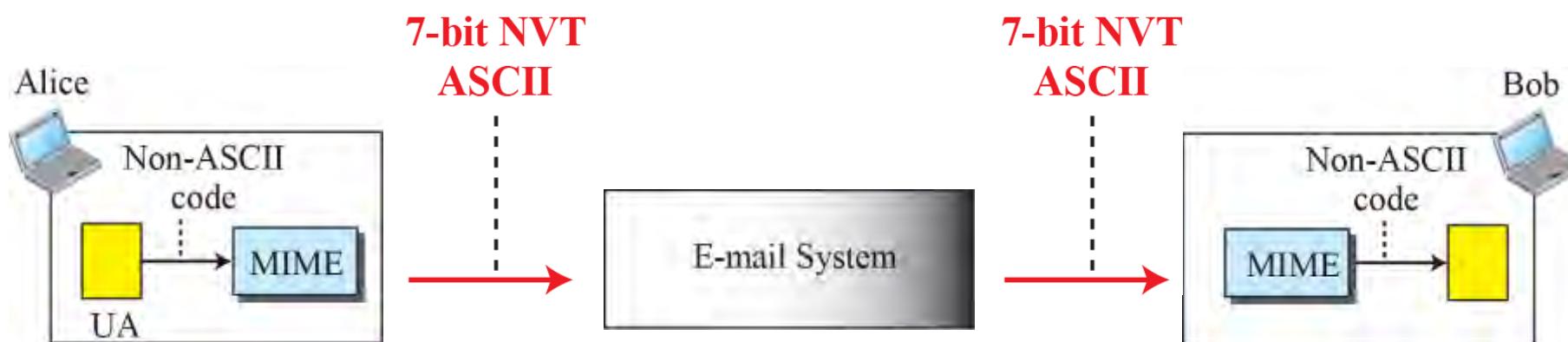
- I have some doubts !!
  1. Can we send email in language other than English? → No
  2. Can we send binary files, audio files, or video files using email? → No

Answer: This is because, email can send message only in NVT 7-bit ASCII format.

So, to achieve the above target, we need a supplementary protocol.

**Multipurpose Internet Mail Extension (MIME)**: it is supplementary protocol that allows non-ASCII data to be sent through e-mail.

- MIME transforms **non-ASCII** data at the sender site to **NVT ASCII** data and delivers it to the client MTA to be sent through the Internet.
- The message at the receiving site is transformed back to the original data.



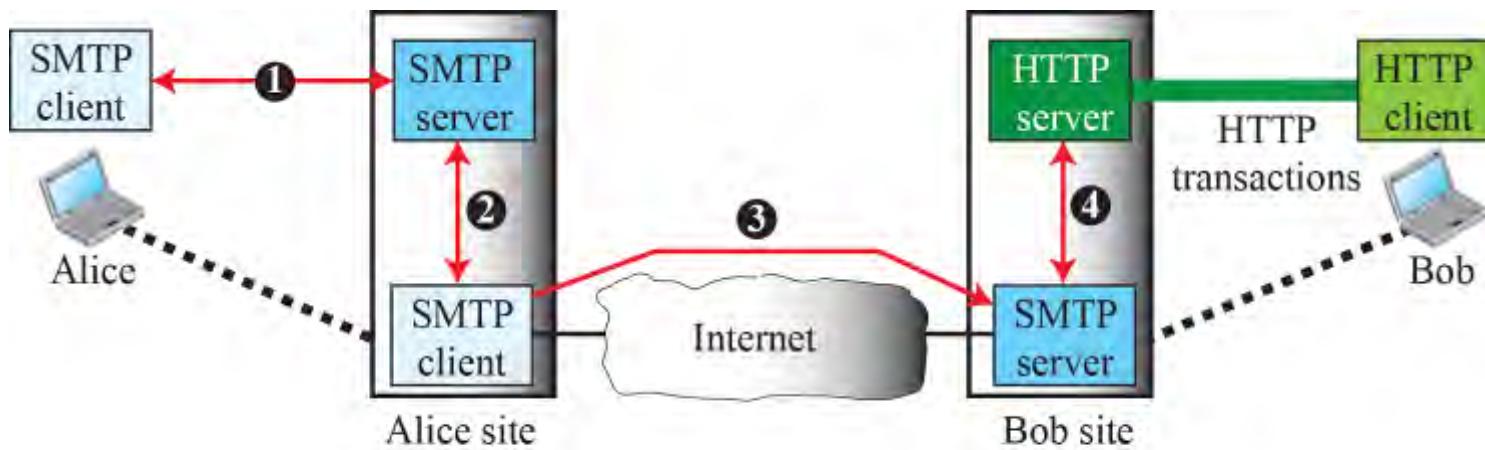
## MIME headers

- **MIME Header:** It defines five headers, as shown below.

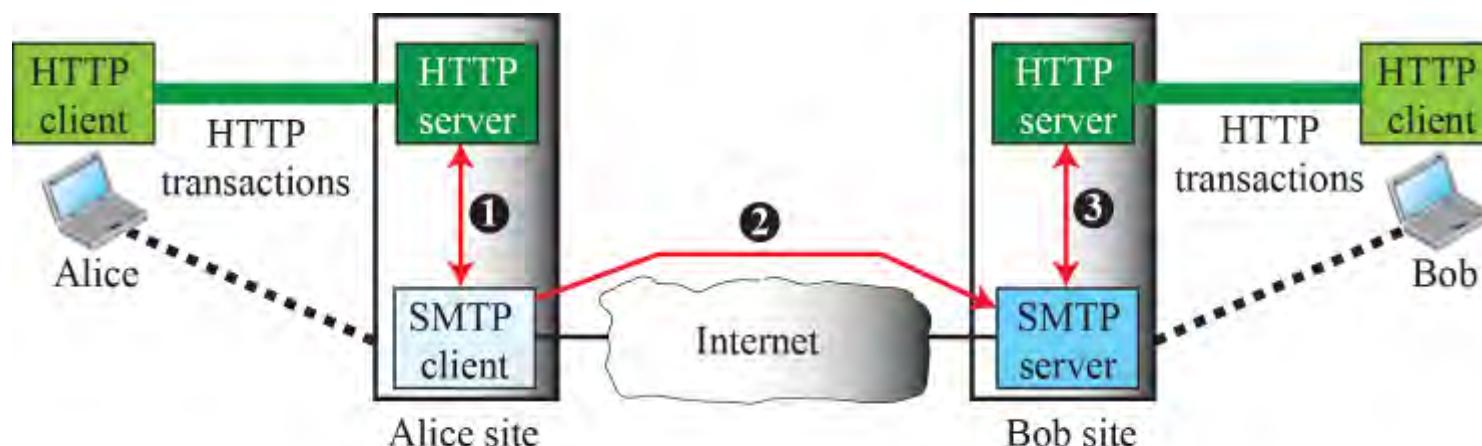
E-mail header
MIME-Version: 1.1
Content-Type: type/subtype
Content-Transfer-Encoding: encoding type
Content-ID: message ID
Content-Description: textual explanation of non-technical contents
E-mail body

Description	
Version	Current version is 1.1
Content Type/subtype	Text, multipart, message, image, video, audio, application
Content-Transfer-Encoding	7bit → NVT ASCII 8-bit → Non ASCII Binary, Base64, Quoted-printable
Content ID	To uniquely identify whole message in multi message environment
Content Description	Defines whether the body is image, , audio or video

# Web-Based Email: Case-I and Case-II



Case 1: Only receiver uses HTTP

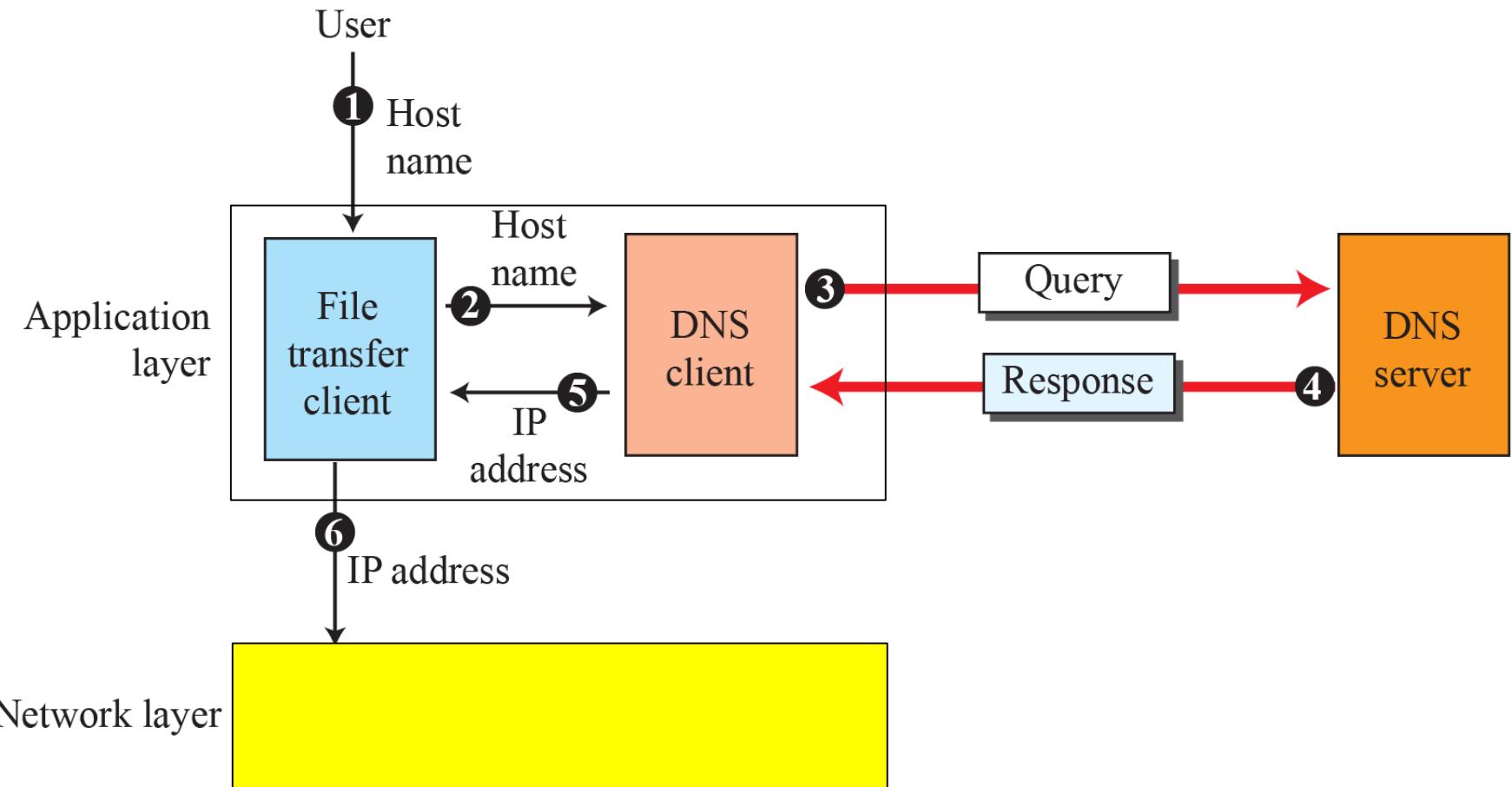


Case 2: Both sender and receiver use HTTP

# Client-Server Application: **DNS**

- **Motivation:** To identify an entity, TCP/IP protocols use the IP address, which uniquely identifies the connection of a host to the Internet. However, people prefer to use names instead of numeric addresses. Therefore, we need a system that can map a name to an address or an address to a name. It is done by DNS.

# Operation of DNS Server



# Domain Name Space

- **Name Space:** Like IP Address is unique, each machine has unique name from the available name space for unambiguous operation.

## Classification of Name Space:

1. Flat name space

2. Hierarchical name space

**Flat name Space:** A name in this is a sequence of characters without structures. So, it must be centrally controlled to avoid the duplication of ambiguity.

**Hierarchical name Space:** It is in the form of structures. Name consists of many parts, each having significance. So, it is not centrally controlled to avoid the duplication of ambiguity, thus decentralized from the administrative point-of-view.

- Think Now !
  - Which structure we follow for domain name space?
  - Flat or hierarchical method ?

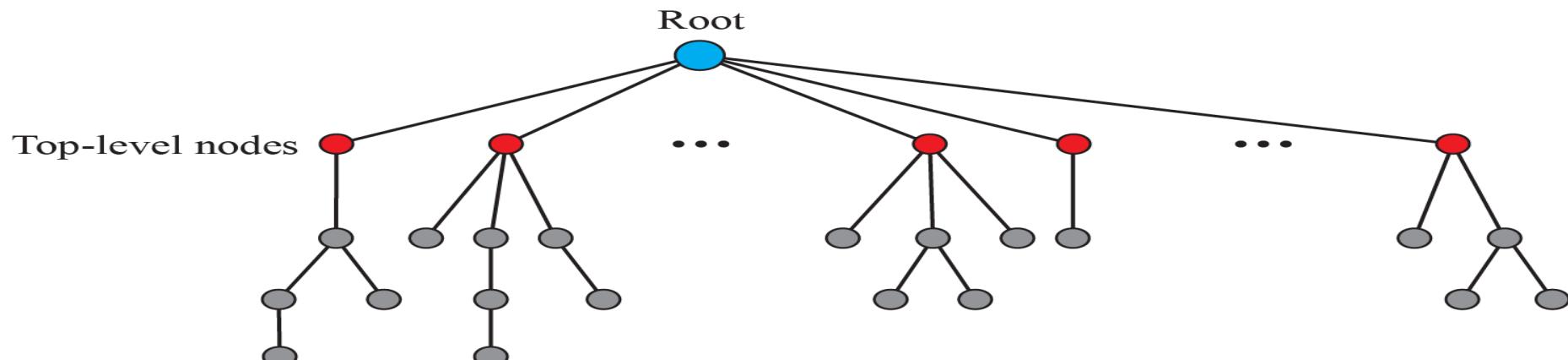


**Answer:** we use hierarchical domain name space.

- Where, a name is made of several parts and each has some significance.

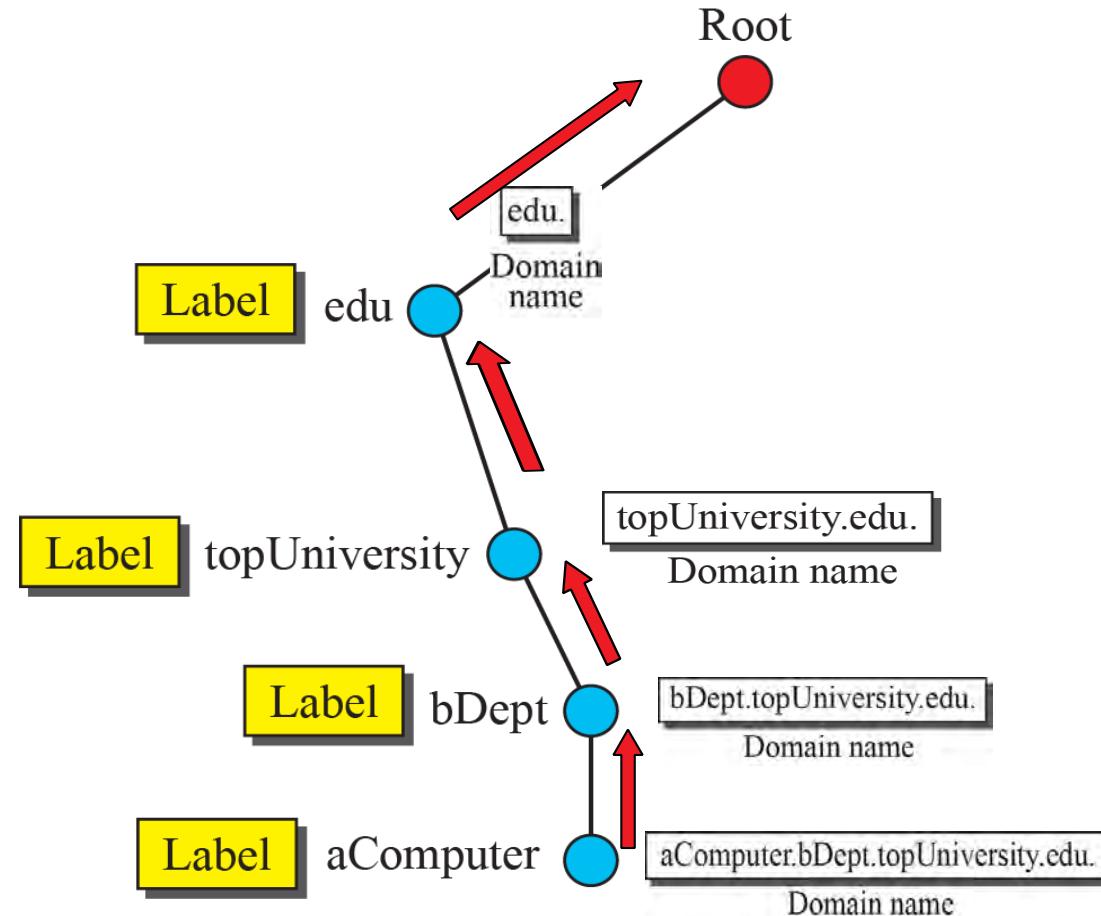
**Domain Name space:** To have hierarchical name space, domain name space was designed.

- In this design, names are defined in inverted-tree structure with the root at the top.
- The maximum possible levels can be up to 128; from level 0 → *root* to level 127 as shown figure below.



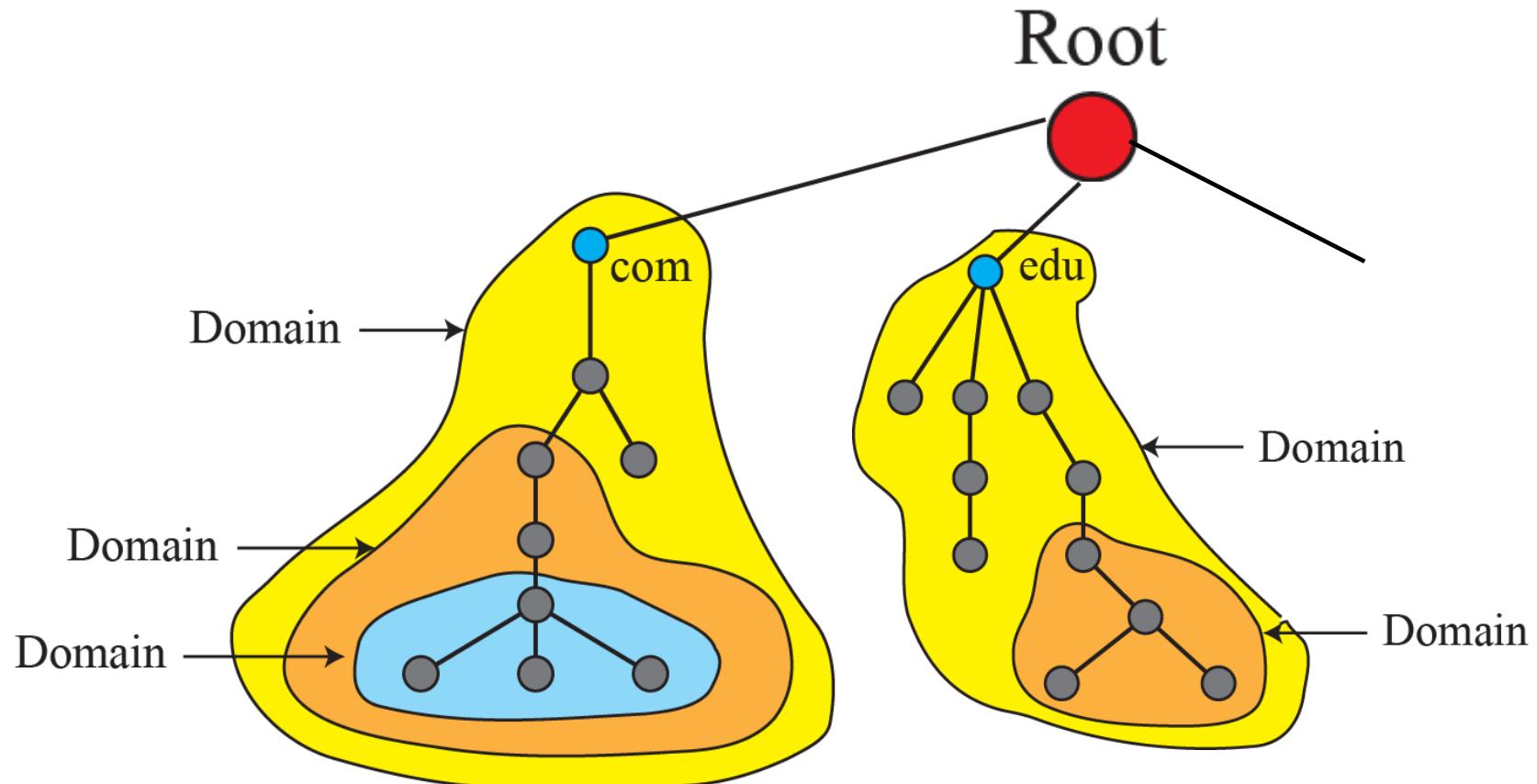
# DNS: Domain Names and Labels

- **Label:**
  - Each node in the tree has a **label**; which is a string with a maximum of **63 characters**.
  - The root label is null string (empty string).
  - The DNS requires the children of a node should have different labels.
- **Domain Name:**
  - Each node in a tree has a domain name.
  - A full domain name is a sequence of labels separated by **dots (.)**.
  - The domain names are always read from the node up to the root.
- **Fully Qualified Domain Name (FQDN):** is ended with null string, i.e., dot **(.)**. Otherwise, it is called **PQDN**



**Domain:** A **domain** is a subtree of the domain name space.

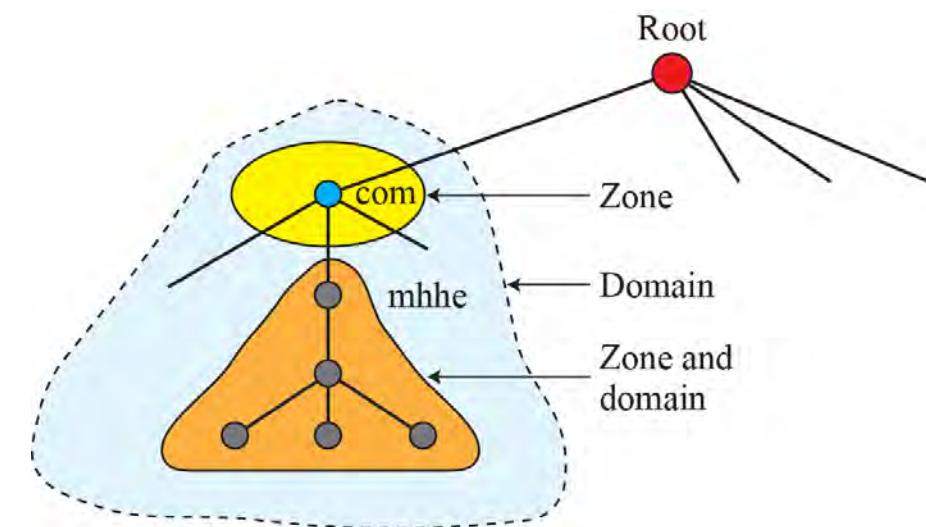
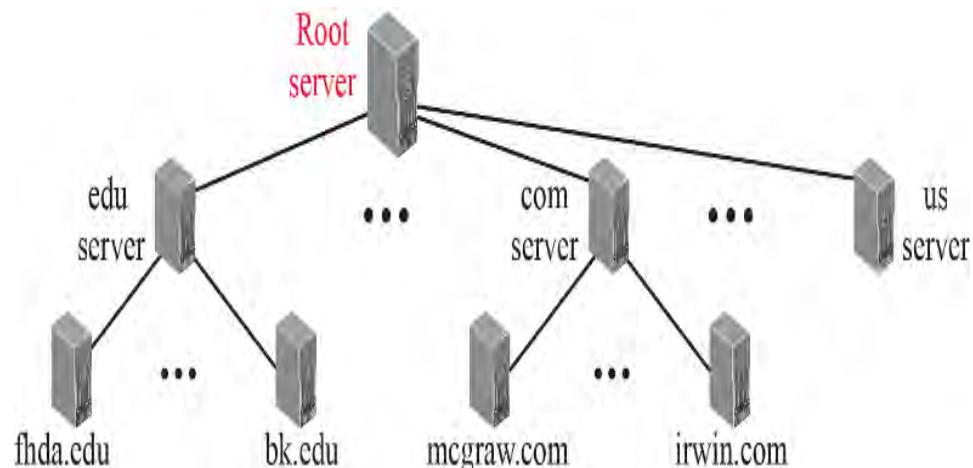
- **Name of the domain** is the **name of the node** at the top of the tree.



**Distribution of Name Space:** Name space is distributed in hierarchy manner. The name space is distributed among many computers called **DNS servers**.

DNS allows to be divided further into smaller domains (subdomains).

**Zone:** Since the complete domain name hierarchy cannot be stored on a single server, it is divided among many servers. What a server is responsible for or has authority over is called a **zone**.



- **Root Server:** It is a server whose zone consists of the whole tree. The root server does not store any information about domain but delegates its authority to other servers, keeping references to those servers.

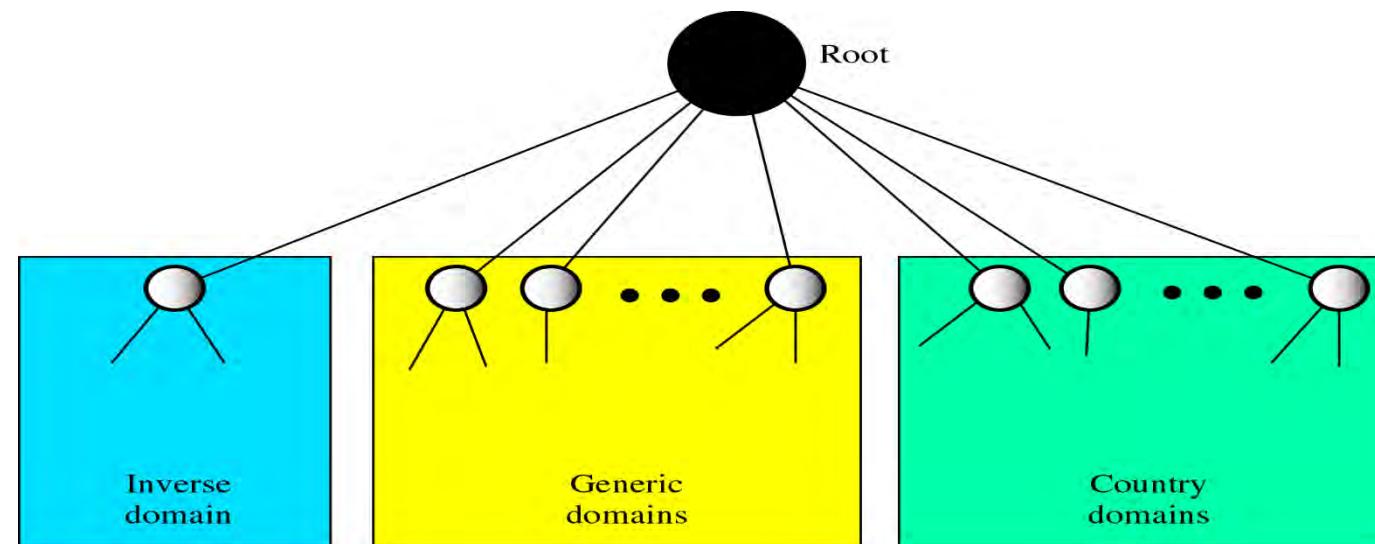
## **DNS defines two types of servers:**

1. **Primary Server:** It is server that stores a file about the zone for which it has an authority. It is responsible for creating, maintaining, and updating the zone file. It stores the zone file on local disk.
2. **Secondary Server:** It is a server that transfers the complete information about a zone from another server (primary or secondary) and stores the file on its local disk. **It neither creates nor updates the zone files.** If update is required, it must be done by the primary server, which sends the updated version to secondary.

**Note:** A primary server loads all information from the disk files; the secondary server loads all information from the primary server.

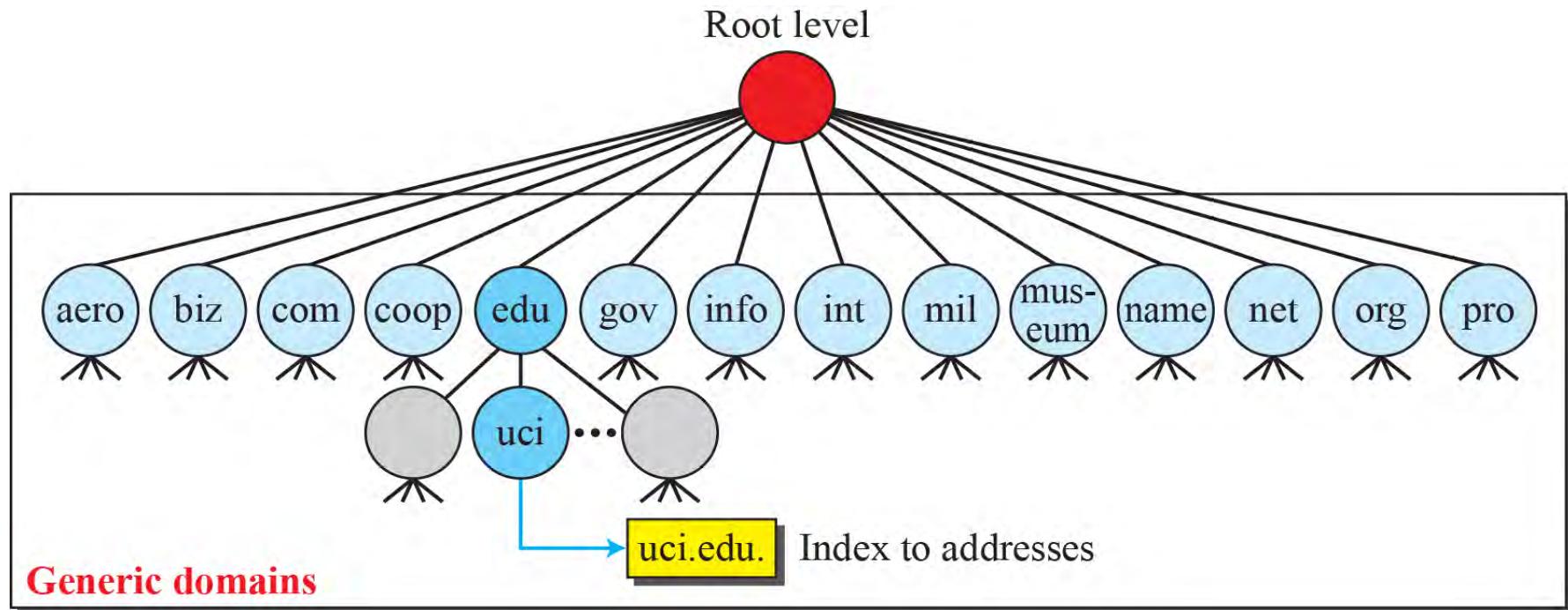
# DNS in the Internet

- DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) was originally divided into three different sections:
  1. Generic domains
  2. Country domains
  3. Inverse domains.



# Generic Domains

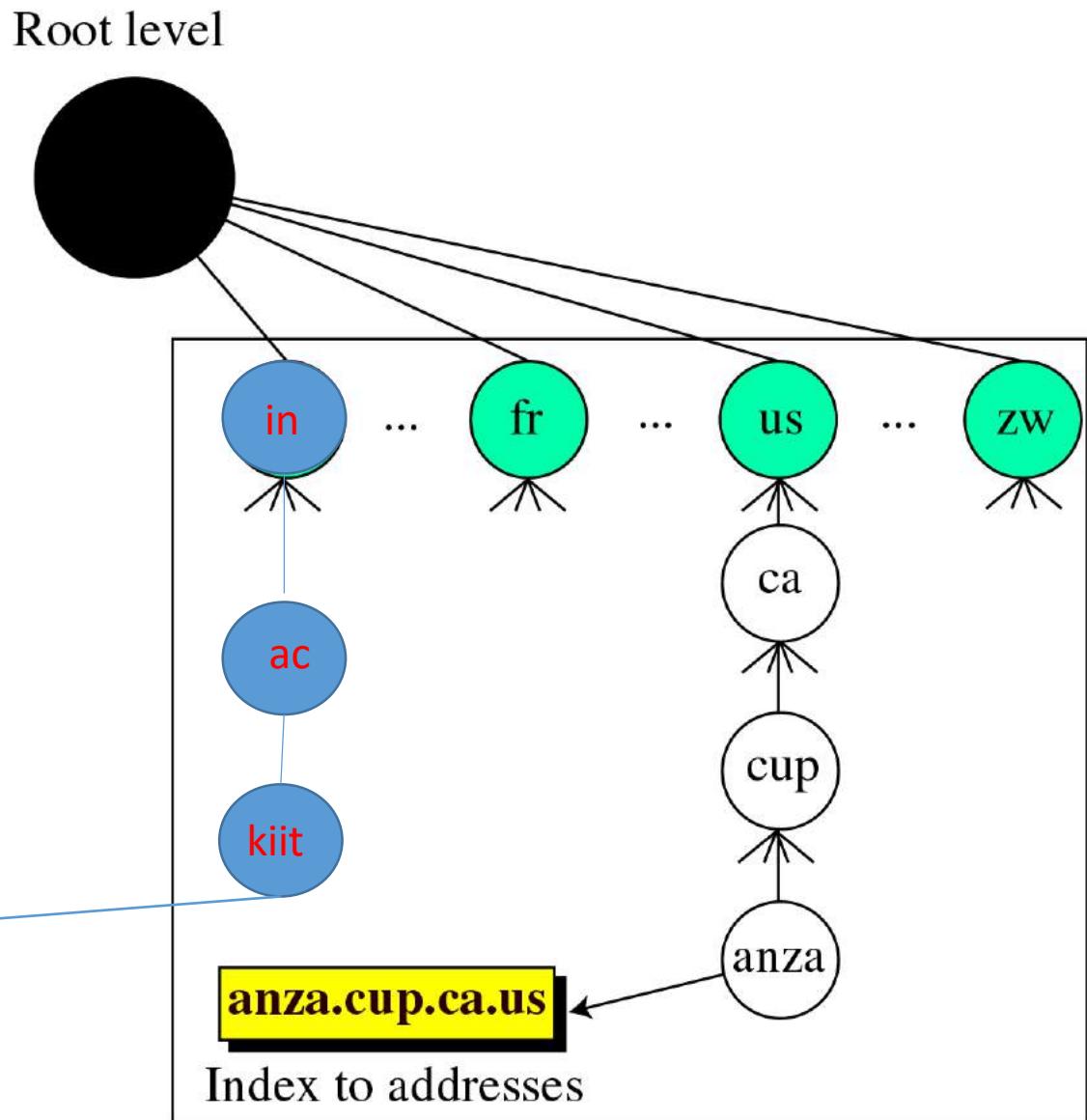
- The generic domains define registered hosts according to their generic behavior. Each node in the tree defines a domain, which is an index to the domain name space database.



# Country Domains

- The country domains section uses two-character country abbreviations (for e.g., India→ **in**). The second labels can be organizational, or they can be more specific, like academic→ **ac**

**kiit.ac.in**



- Do we need to study 3<sup>rd</sup> domain name?  
→ Inverse domain name



**Answer → NO,**

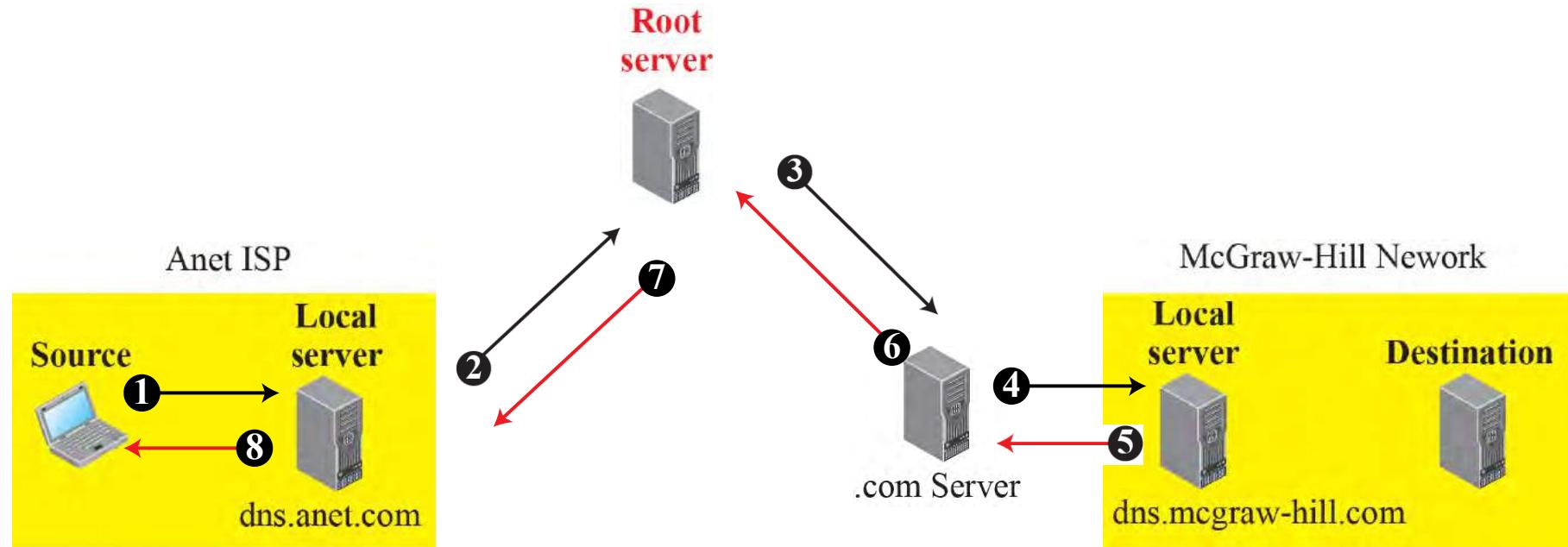
**Explanation:**

However, due to rapid growth of the Internet, it becomes extremely difficult to keep track of the inverse domains, which could be used to find out name of the host when IP address is known.

# DNS Resolution:

- **Resolution:** Mapping a name to an address is called *name-address resolution*.
- **Resolver:** A host that needs to map an address to a name or a name to an address calls a **DNS client** known as **resolver**.
- A resolution can be of two types:
  1. **Recursive Resolution:** In this, each server that does not know the mapping sends the **query** to
    - the **root DNS server** as root DNS server does not keep the mapping information
    - It sends the query to this top-level-domain server.
    - process goes on as shown in the diagram, until mapping is done.
  2. **Iterative Resolution:** In this, each server that does not know the mapping sends the **IP address of the next server** back to one that requested it.

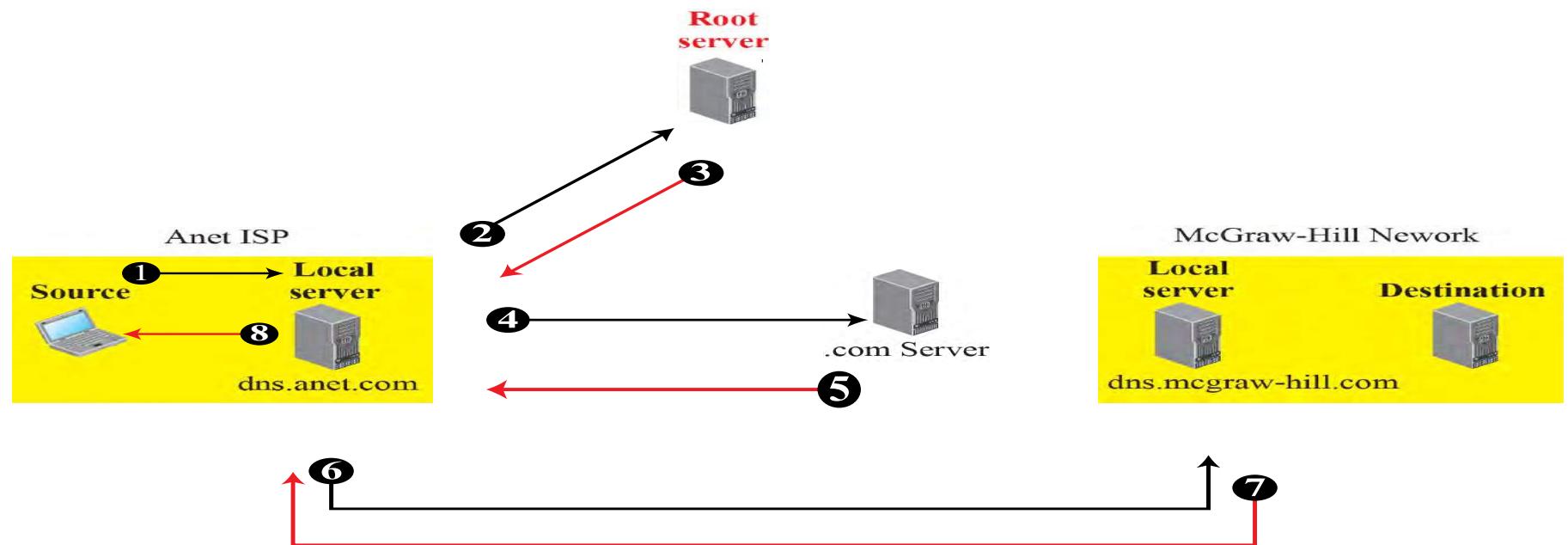
# Architecture of DNS: Using Recursive Resolution



**Source:** some.anet.com

**Destination:** engineering.mcgraw-hill.com

# Architecture of DNS: Using Iterative Resolution



**Source:** some.anet.com  
**Destination:** engineering.mcgraw-hill.com

Amit Jha, SOEE, KIIT-DU

# HYU.....

- Can we use *caching* for the domain name as well?



# Resource Records

- The zone information associated with a server is implemented as a set of *resource records*. A *resource record* is a 5-tuple structure as shown below → **Domain Name, Type, Class, TTL, Value**
- **Domain Name:** It identifies the resource records.
- **Type:** defines how the value should be interpreted.
- **Class:** defines the type of network. For Internet, it is **IN**.
- **TTL:** defines the number of seconds for which the information is valid.
- **Value:** The information kept about the domain name.

Type	Interpretation of value
A	A 32-bit IPv4 address (see Chapter 4)
NS	Identifies the authoritative servers for a zone
CNAME	Defines an alias for the official name of a host
SOA	Marks the beginning of a zone
MX	Redirects mail to a mail server
AAAA	An IPv6 address (see Chapter 4)

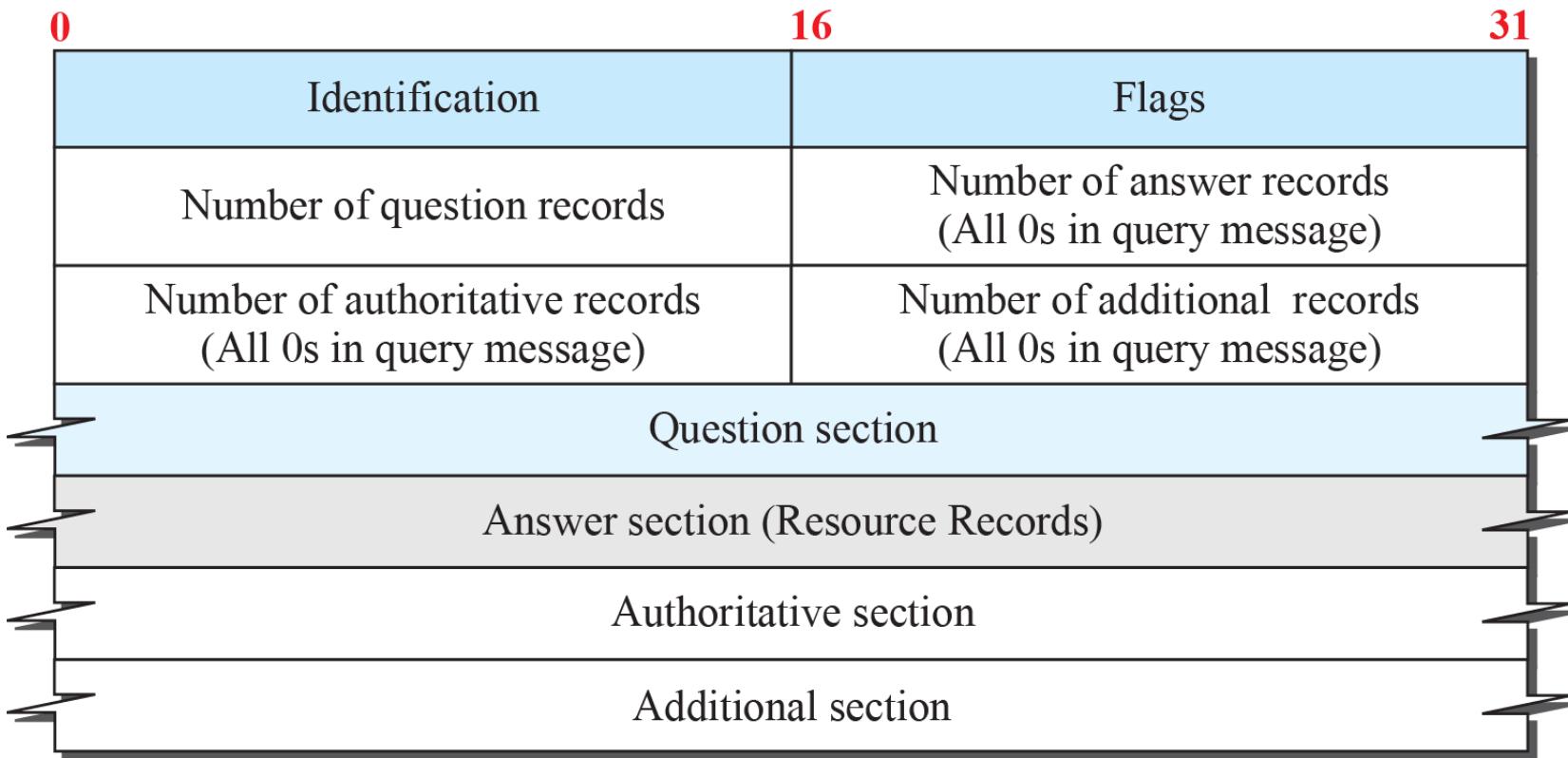
# Various Types of Records Maintained by DNS

- **Host A:** It is used to translate human friendly domain names such as "www.example.com" into IP-addresses such as 23.211.43.53 (machine friendly numbers). For IPv4 it is A, whereas, for IPv6, it is AAAA record.
- **NS:** NS-records identify the **DNS servers responsible (authoritative) for a zone**. An NS-record identifies the name of a DNS server - not the IP-address.
- **MX:** MX-records are used to specify the e-mail server(s) responsible for a domain name. It redirects mail to mail server.
  - If a domain name is handled by multiple e-mail servers (for backup/redundancy), a separate MX-record is used for each e-mail server, and the preference numbers then determine in which order (lower numbers first) these servers should be used by other e-mail servers.
  - If a domain name is handled by a single e-mail server, only one MX-record is needed and the preference number does not matter.
  - When sending an e-mail to "[user@example.com](mailto:user@example.com)", your e-mail server must first look up any MX-records for "[example.com](http://example.com)" to see which e-mail servers handles incoming e-mail for "example.com".
- **CNAME:** CNAME-records are **domain name aliases**.
  - Computers on the Internet often performs multiple roles such as web-server, ftp-server, chat-server etc.
  - To mask this, CNAME-records can be used to give a single computer multiple names (aliases).
  - For example, the computer "computer1.xyz.com" may be both a web-server and an ftp-server, so two CNAME-records are defined:
    - "[www.xyz.com](http://www.xyz.com)" = "[computer1.xyz.com](http://computer1.xyz.com)" and "[ftp.xyz.com](http://ftp.xyz.com)" = "[computer1.xyz.com](http://computer1.xyz.com)".
  - The most common use of the CNAME-record type is to provide access to a web-server using both the standard "[www.domain.com](http://www.domain.com)" and "[domain.com](http://domain.com)" (with and without the www prefix).
    - This is usually done by creating an [A-record](#) for the short name (without www), and a CNAME-record for the www name pointing to the short name.

# Which service DNS uses of Transport Layer?

- It can use either UPD or TCP.
- **UDP**
  - It is used when the size of the response packet is less than 512 bytes because most of the UDP packets has limitation **of 512** bytes packet size.
- **TCP**
  - If size of the response packet is more than 512 bytes, it uses TCP service of the Transport Layer.
- **Note:** *In both the cases (UDP and TCP), the well-known port used by the server is 53.*

# DNS Message Format



**Note:**

The query message contains only the question section.  
The response message includes the question section,  
the answer section, and possibly two other sections.

# P2P Networks

- **Peer:** In this, internet users which are ready to share their information becomes peers.
- **P2P Network:** Different peers are connected over the Internet to create a network.
- **How it works:** When a peer in the network has a file to share, it makes it available to the rest of the peers. An interested peer can connect itself to the computer where the file is stored and download it. After a peer downloads a file, it can make it available for other peers to download. As more peers join and download that file, more copies of the file become available to the group.
- Think..
  - Since, the list peers may grow and shrink, then how the P2P paradigm keeps tracks of the loyal peers and the location of the file?



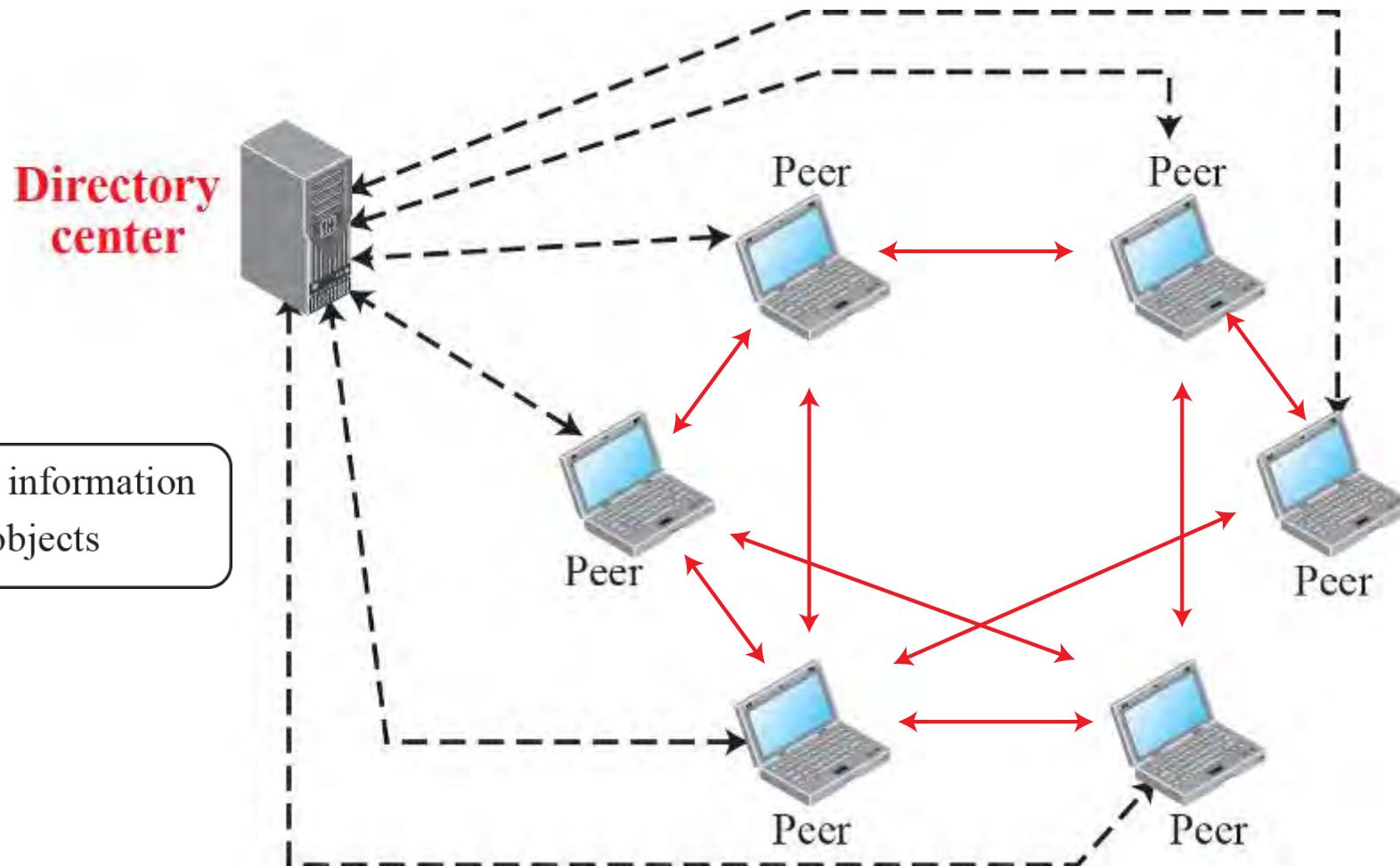
To answer the last question, the P2P network is divided into two categories: **Centralized** and **Decentralized**

- **Centralized P2P Networks:**

- In this a directory system is used which works on **client-server paradigm**.
- It has list of all the peers and the information about (files) what they can offer.
- Since, listing of the peers and the file information is done by directory system which uses a client-server paradigm, however, the files download and upload is achieved using peer-to-peer paradigm, this architecture is also referred to as **hybrid P2P networks**.
- **How one can register as a peer?**
  - In this, a peer first register itself with the directory system (central server) by providing its IP address and the files which it has to share.
- **How file is downloaded?**
  - A peer, looking for a particular file, sends a query to a central server. The server searches its directory and responds with the IP address of the node (peer) that have a copy of the file. The peer contacts on of the node and downloads the file.

**Note:** *The directory is constantly updated as nodes join or leave the network.*

## Working of centralized P2P Networks

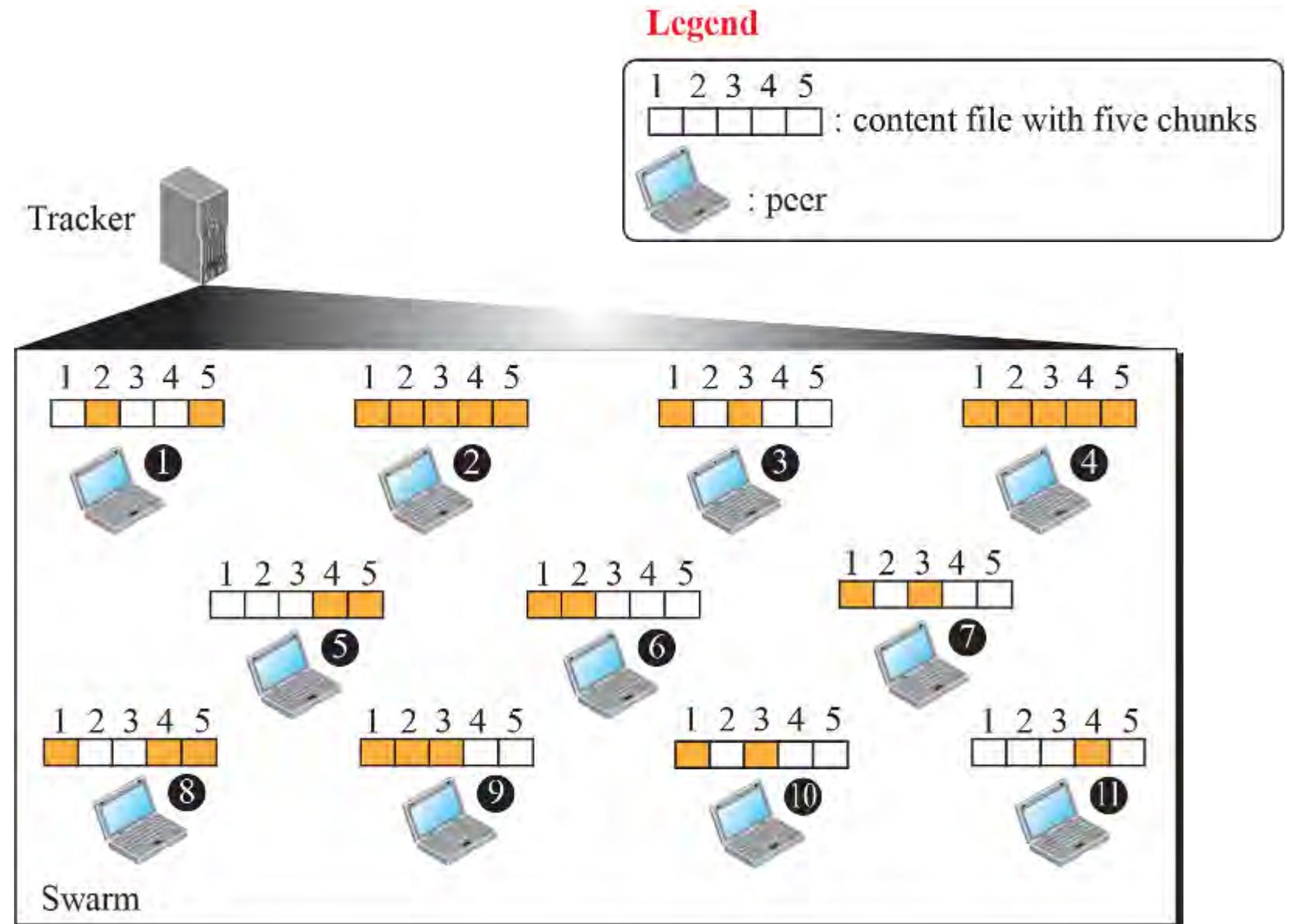


- **Decentralized P2P Networks:**
  - In this, there is no need of a centralized directory system.
  - Here, peers arrange themselves into an overlay network, which is a logical network made on top of the physical network.
  - Depending upon how the nodes in the overlay network are linked, a decentralized P2P network is classified as either unstructured or structured.
- **Unstructured P2P Networks:** In this, nodes are linked randomly. A search in this network is not very efficient because a query to find a file must be flooded through the network, which produces significant traffic and still the query may not be resolved.
- **Structured P2P Networks:** In this, nodes are linked through a predefined set of rules. Thus, a search in this network is very efficient because a query to find a file must not be flooded through the network, which produces less traffic to resolve the query. The most common technique used for creating structured networks is using Distributed Hash Table (DHT).

# P2P Application: BitTorrent

- In original BitTorrent, there is another entity in a torrent, called the tracker, which as the name implies, tracks the operation of the swarm. Swarm is the group of peers in the torrent.
- **Let us assume:**
  - The file is to be shared, is divided into 5 pieces (chunks).
  - Peer 2 and peer 4 already have all the pieces; other peers have some pieces.
  - The pieces that each peer has are shaded.
  - Some peers may leave the torrent; some may join the torrent.
- **Think...w.r.t given statement as above. Explain the process that how does a peer can download the same file?**

- The new peer accesses the BitTorrent server with the name of the content file (or simply file).
- It receives a metafile (called torrent file), that contains the information about the pieces in the content file and address of the tracker that handles that specific torrent.
- The new peer now accesses the tracker and receives the addresses of some peers in the torrent.
- The new peer is now part of the torrent and can download and upload the pieces of the content file.



**Note:** Peers 2 and 4 are seeds; others are leeches.

# Remember.....

- *Also, there exist trackerless BitTorrent.*
- Importantly, BitTorrent clients never actually download files from the tracker itself. The tracker participates in the torrent only by keeping track of the BitTorrent clients connected to the swarm, not actually by downloading or uploading data.
- **Note:** *BitTorrent may be primarily used for piracy at the moment, as its decentralized and peer-to-peer nature are a direct response to efforts to crack down on Napster and other peer-to-peer networks with central points of failure. However, BitTorrent is a tool with legitimate uses in the present – and many other potential uses in the future.*

## **End of Module-2**

You are advised to go through the text book as mentioned in the class for more clarity and details.

# CN (IT-3001)

## Transport Layer: Introduction

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University

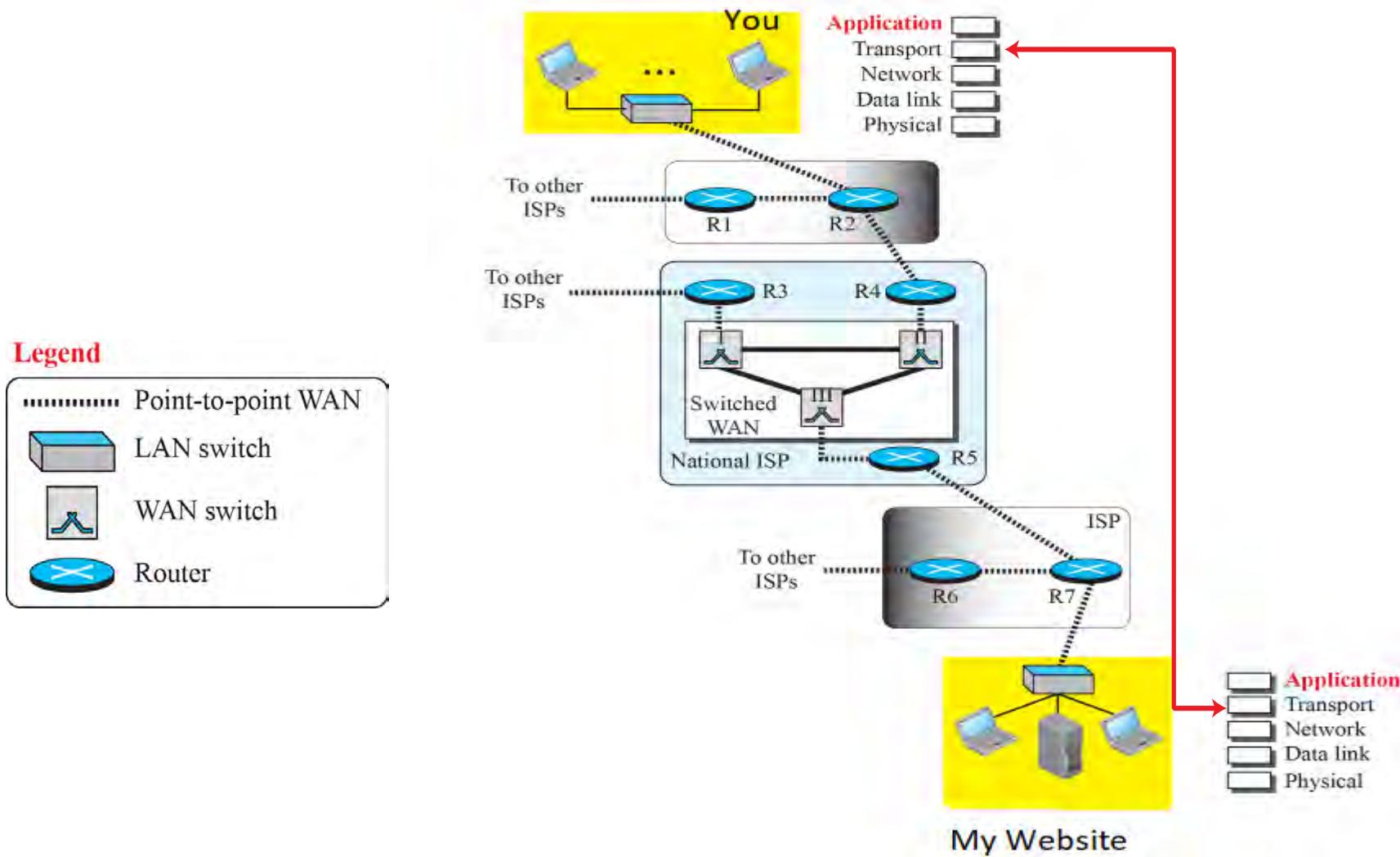


**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Content

- Transport Layer Services
- Flow Control in Transport Layer
  - Stop-and-wait
  - Go-back- $N$
  - Selective Repeat
- UDP: services and applications
- TCP:
  - Services, features and applications
  - TCP connection
  - State transition diagram
  - Windows in TCP
  - Flow Control
  - Congestion control

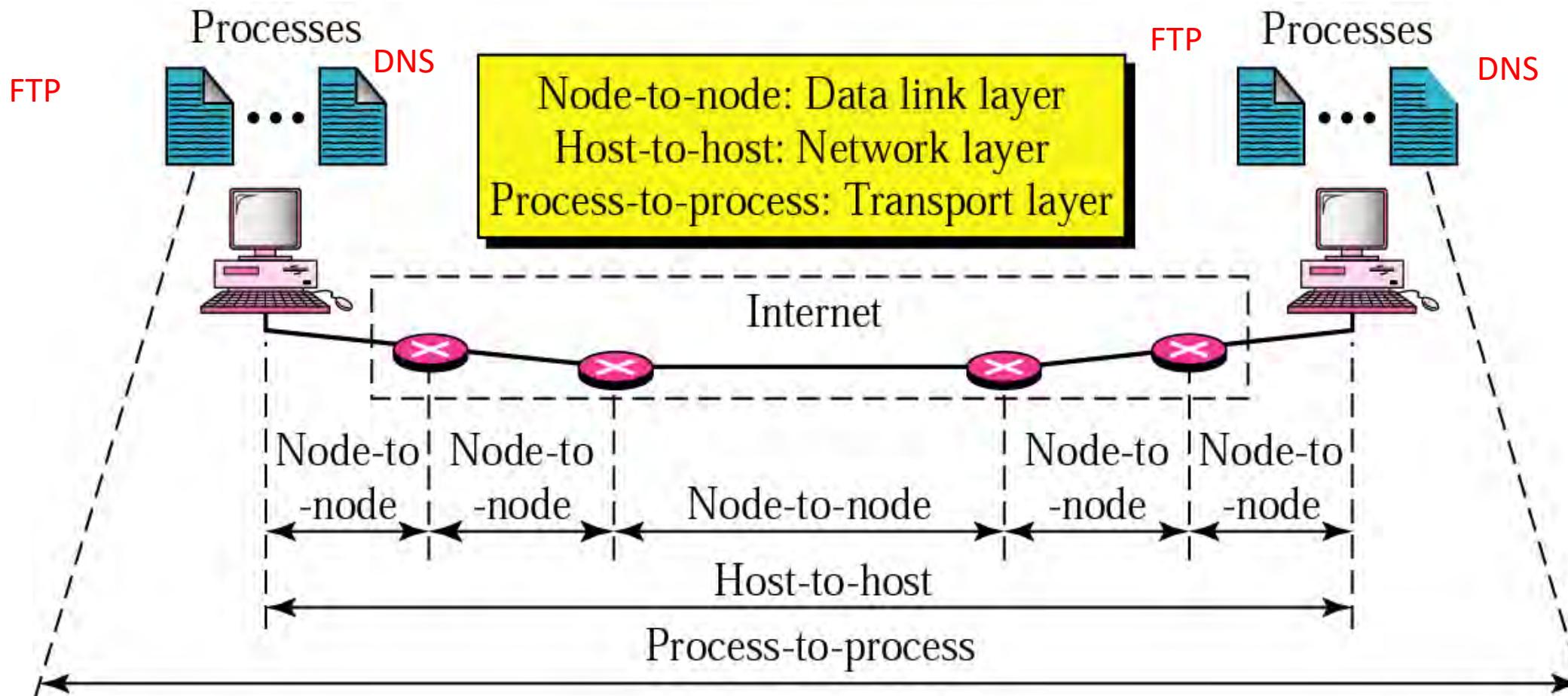
# Logical Connection at the Transport Layer



# Transport Layer Services

- The transport layer is responsible for
  - Providing services to the application layer and receives the services from the network layer.
  - Providing *process-to-process* communication between two application layers.
  - Providing communication using a logical connection.
  - Providing multiplexing at the source; and demultiplexing at the destination host.
  - Providing flow and error control services.
  - Providing congestion control

# Types of data deliveries



**Ques:** How can two process of different hosts communicate?

Although there are several ways to achieve process-to-process communication, the most common one is through the client/server paradigm.

### **Client/Server Paradigm:**

**Client:** The process on the local host is called client.

**Server:** The process on the remote host is called server.

**Note:** Both processes (client and server) have the same name. For example, to get the day and time from a remote machine, we need a Daytime client process running on the local host and a Daytime server process running on a remote machine.

# Client/Server Paradigm

- For communication, we must define; local host, local process, remote host, remote process.
- **Addressing:** Like MAC and IP addresses at DLL and Network Layer; we need transport layer address, called a *port number*, to choose among the multiple processes running on the destination host.
- The destination port number is needed for delivery; the source port number is needed for the reply.
- **Port number:** It is 16-bit integers between 0 to 65,535.
- **Client port number:** Client chooses its port number randomly from 0 to 65,535 using the transport layer software running on the client host. This is called *ephemeral(temporary)* port number. However, an ephemeral port number is recommended to be greater than 1023 for some client-server program to work properly.

...cont

- **Server port number:** Server process is also defined by a port number. But, its **not randomly chosen**. If it is random, then client will not know the port number in order to access that server. These are called **well-known port number**.

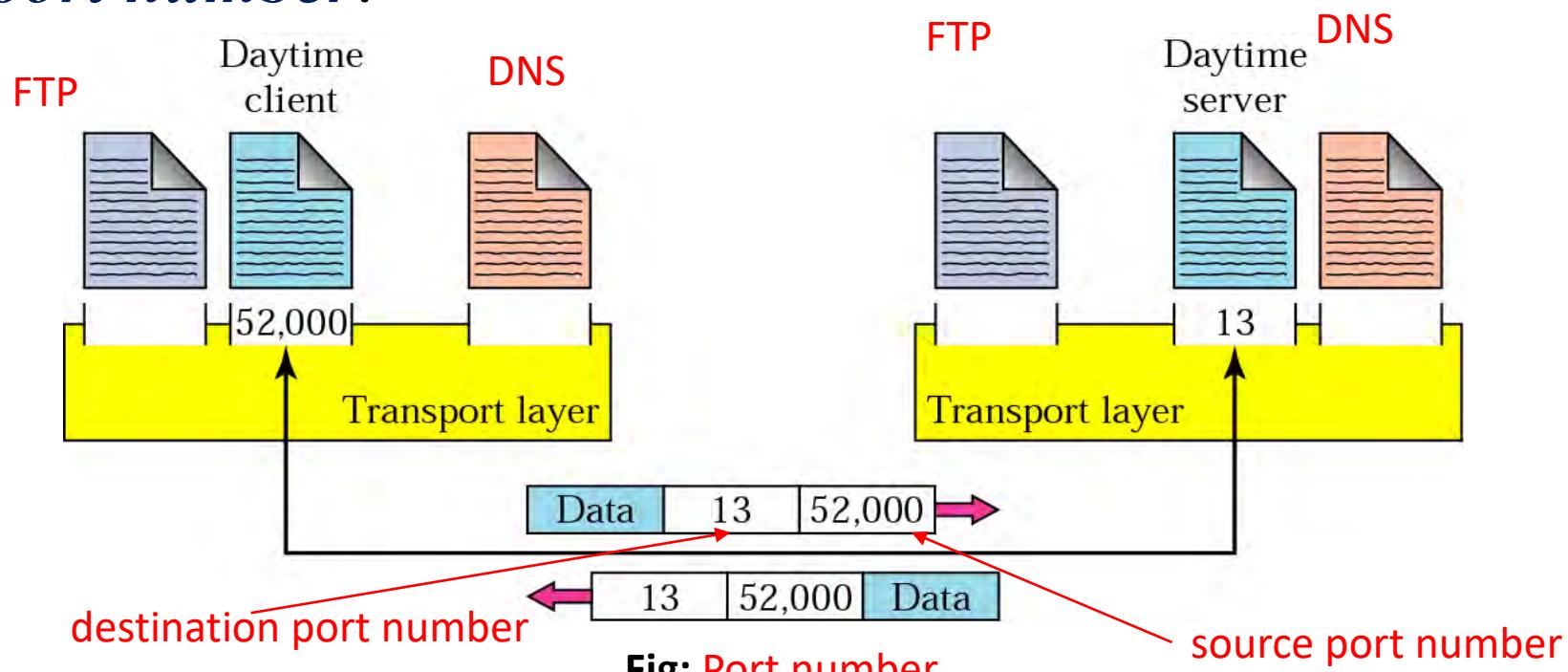
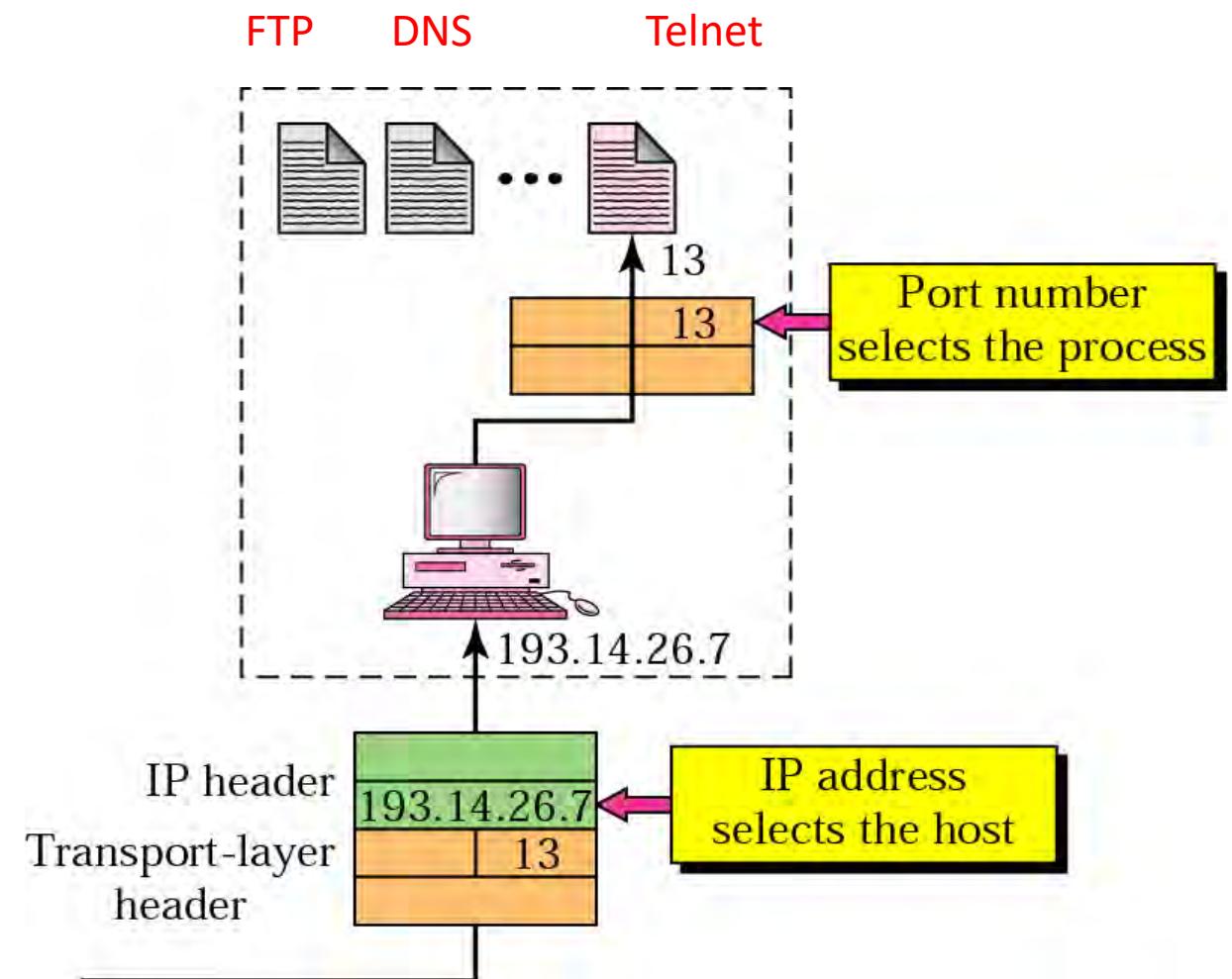
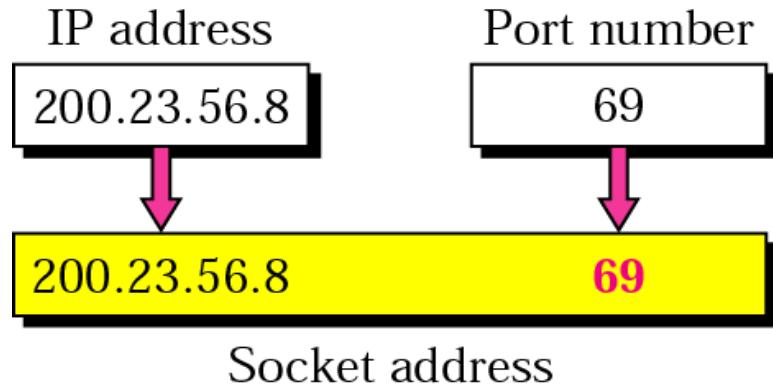


Fig: Port number

# IP addresses versus port number



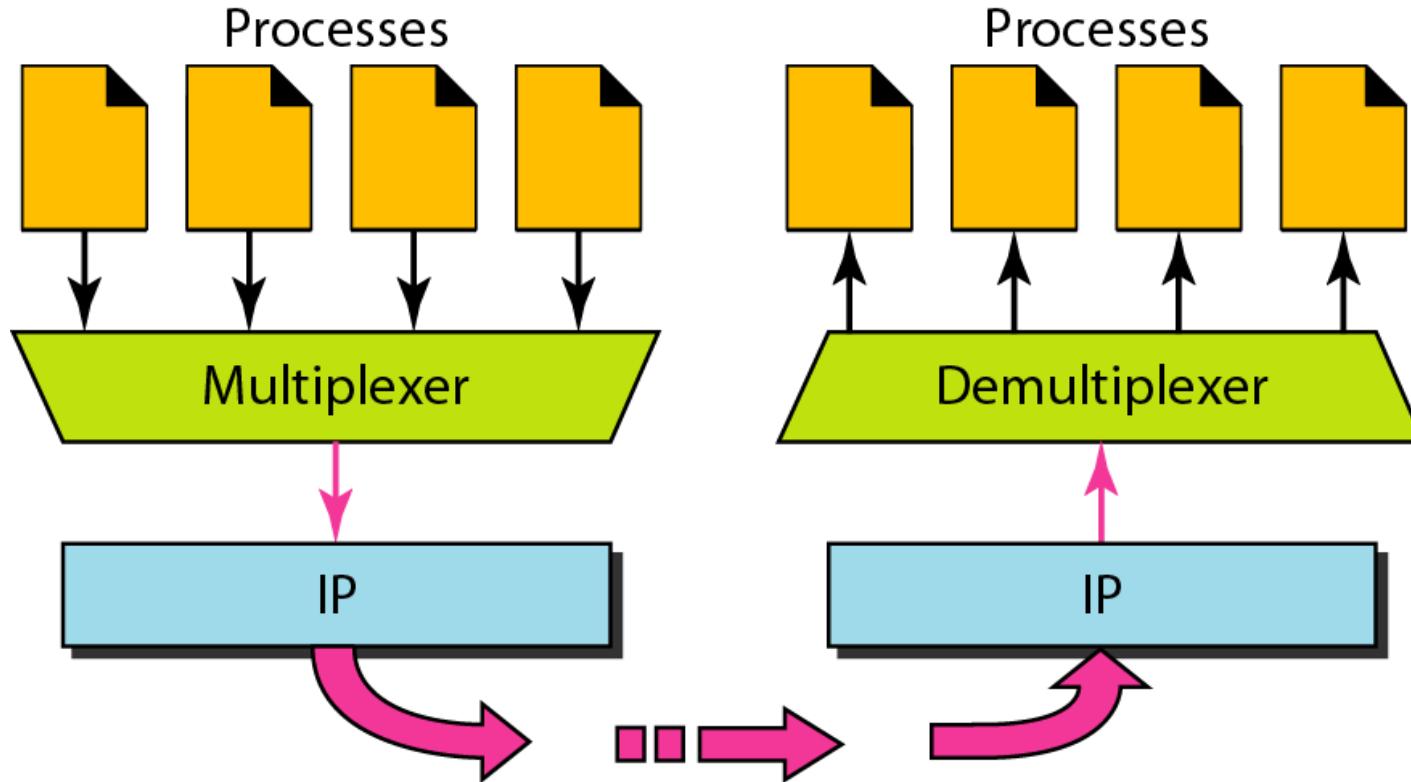
# Questions

1. What do you understand by socket address?
2. What is the socket address if you want to access email having port number 1400 from google server having IP address 192.10.10.142? Assume your IP address is 172.10.10.191.

**Ans:** Socket Address is **192.10.10.142 1400**

- **ICANN Ranges:** It has categorized the port numbers into three parts: **well-known, registered and dynamic (private).**
  1. **Well-known port:** It is assigned and controlled by the ICANN. Range is from **0-1023**.
  2. **Registered port:** These ports are neither assign nor controlled by ICANN.  
They are only registered by ICANN so that the duplication of the port can be avoided. Range → **1024-49151**.
  3. **Dynamic Port:** these are temporary port numbers. They are neither assigned nor registered by ICANN.  
Range is → **49152-65,535**.

# Multiplexing & Demultiplexing at Transport Layer



# Connectionless Vs. Connection oriented Service

- Transport layer supports two kind of services: connectionless and connection oriented
- ***Connectionless Service:*** In this, the packets are sent from one party to another with no need for connection establishment or connection release. The packets are not numbered; they may be delayed or lost or may arrive out of sequence. There is no acknowledgment either. E.g., UDP (User Datagram Protocol)
- ***Connection Oriented Service:*** In a connection oriented service, a connection is first established between the sender and the receiver. Data are transferred. At the end, the connection is released. For e.g., TCP (Transmission Control Protocol) and SCTP (Stream Control Transmission Protocol) are connection-oriented protocols.

# Reliable Vs. Unreliable Service

- The transport layer service can be reliable or unreliable.
- ***Reliable:*** If the application layer program needs reliability, we use a reliable transport layer protocol by implementing flow and error control at the transport layer. This means a slower and more complex service.
- ***Unreliable:*** On the other hand, if the application program does not need reliability because it uses its own flow and error control mechanism or it needs fast service or the nature of the service does not demand flow and error control (real-time applications), then an unreliable protocol can be used.
- **Note:** UDP is connectionless and thus, unreliable; TCP and SCTP are connection oriented and thus, reliable.

# CN (IT-3001)

## Transport Layer: Error and flow control Protocols

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Transport Layer Protocols

- These protocols are implemented in the transport layer to provide basically two main functionality:
  1. Flow control and
  2. Error Control

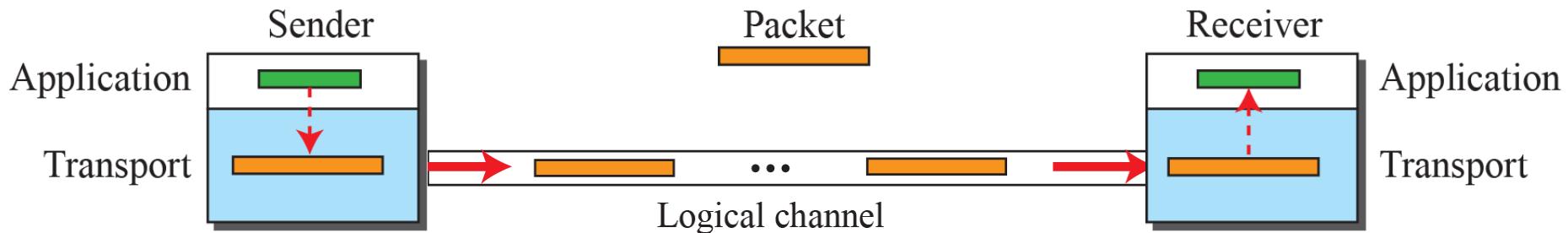
We discuss following protocols in the subsequent slides.

1. Simple Protocol
2. Stop-and-Wait Protocol
3. Go-Back-N Protocol
4. Selective-Repeat Protocol

# Simple Protocol → Connection Less Protocol

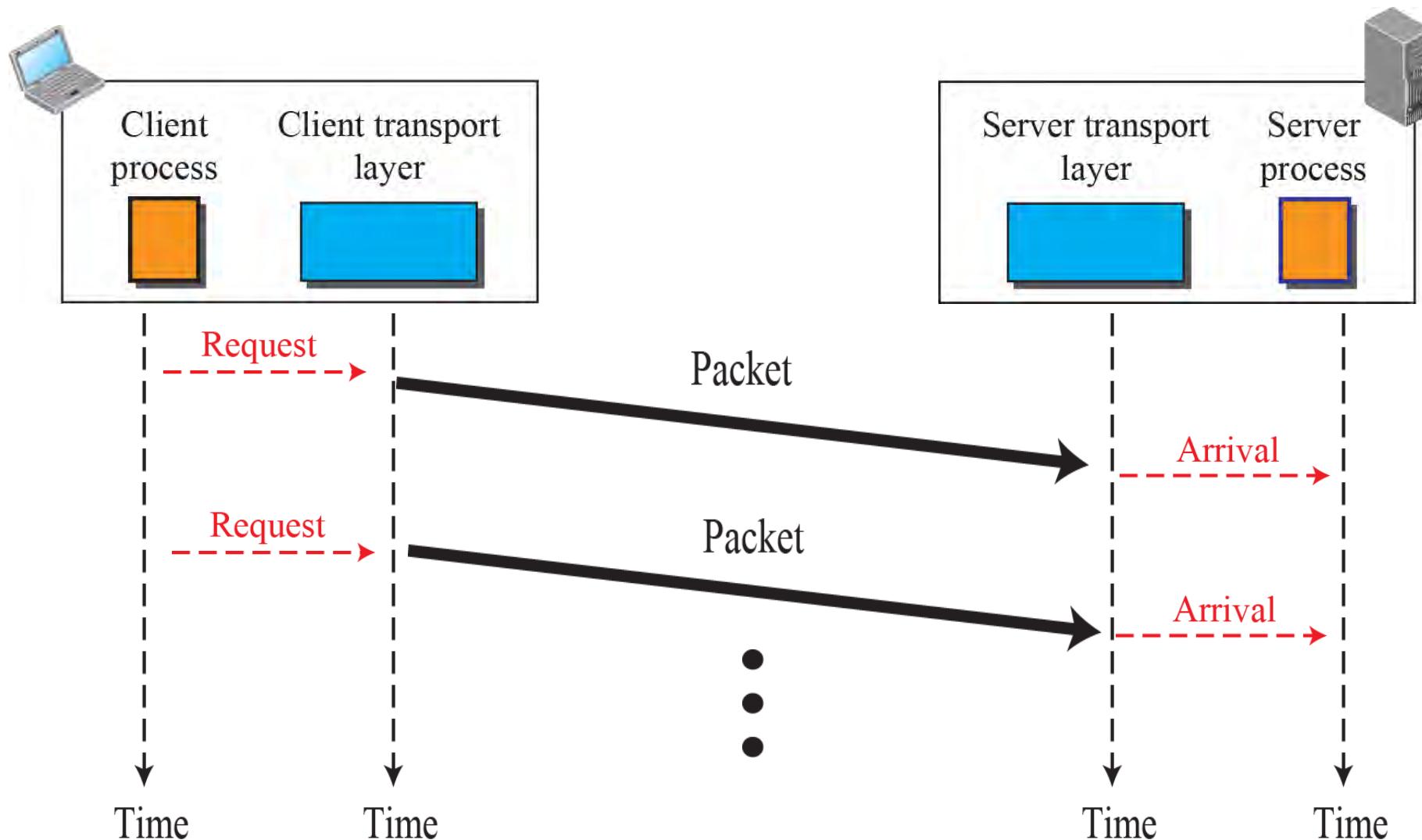
The channel is assumed to be ideal i.e., no error. The Sender and receiver are assumed to work at the same speed thus, no overwhelming of the receiver.

Thus, this protocol is designed for neither flow nor error control.



- **The Transport layer at the sender side**
  - gets a message from its application layer.
  - Then it makes a packet out of it, and sends the packet.
- **The Transport layer at the receiver side**
  - Receives a packet from its network layer
  - Extracts the message from the packet and delivers the message to its application layer.

**Example 3.1:** The sender sends packets one after another without even thinking about the receiver.



- Practically this can never be true, and thus the simple protocol is not used at all in practice.
- Now, let us assume that the sender and the receiver do not operate at the same speed.

*So, we need to design a protocol for flow control*

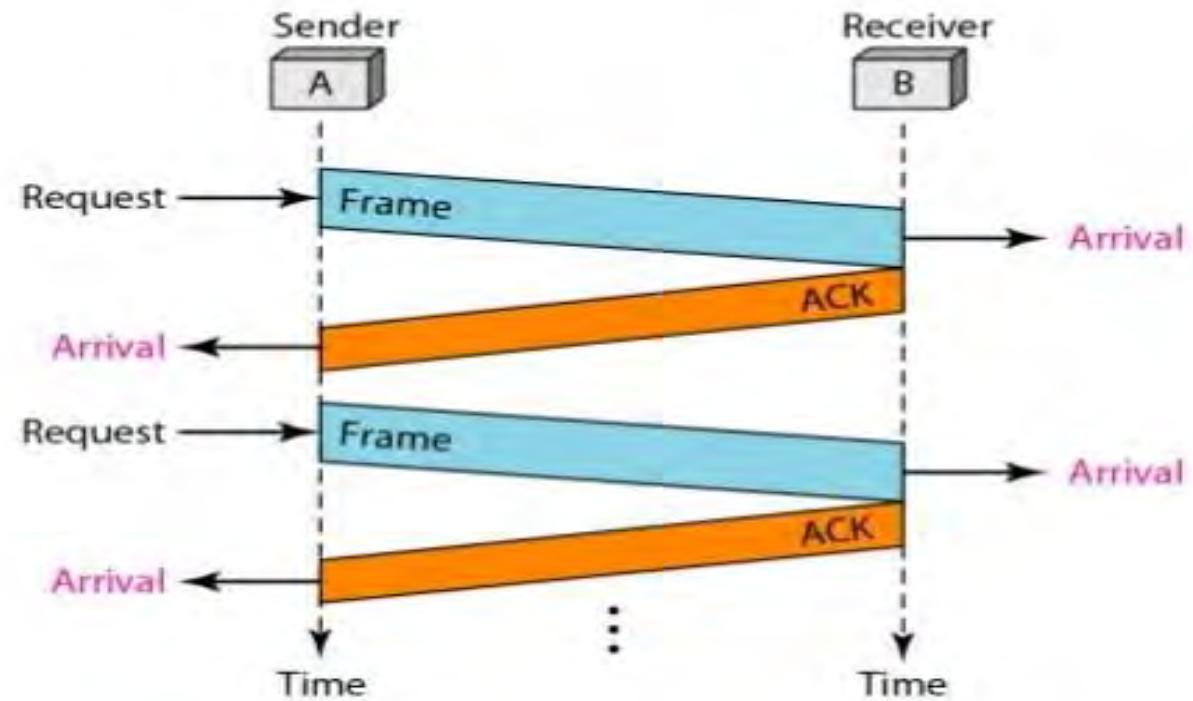
**Note:** *Here also, we assume that the channel is ideal i.e., there is no error.*

# Stop-and-Wait Protocol

- This is connection oriented protocol.
- Here, we implement only flow control.
- The flow control is implemented using following strategy:
  - The sender must wait until it receives the **ACK** packet before sending the next packet.

**Note:** *A small processing delay may be introduced between reception of the last byte of a Data PDU and generation of the corresponding ACK.*

**Example 3.2:** The sender sends one frame and waits for feedback from the receiver. **If and only if** the ACK arrives, the sender sends the next frame.



**Note:** The sending two frames in the protocol involves the sender in four events and the receiver in two events.

- Practically, noiseless channel does not exist.
- So, we need to design a protocol for noisy channel.
- The Stop-and-Wait protocol can also be used for flow control in noisy channel with some modification for error control as well. It can use the checksum for detecting the corrupted packets.
- So, for noisy channel, we have following three protocols implementing flow as well as error control.
  1. *Stop-and-Wait Protocol*
  2. *Go-Back-N Protocol*
  3. *Selective-Repeat Protocol*

**For non-ideal channel, in general, the following two cases may occur....**

1. If Packet is

- **Corrupted:** To detect corrupted packets, add checksum to data packet.
- **Lost:** resend the packet
- **Out of order:** Give numbering to each packet, called sequence number

2. If ACK is

- **Corrupted:** add redundant bits i.e., checksum.
- **Lost:** give numbering to each ACK, called ACK number
- **Out of order:** give numbering

# Stop-and-Wait Protocol

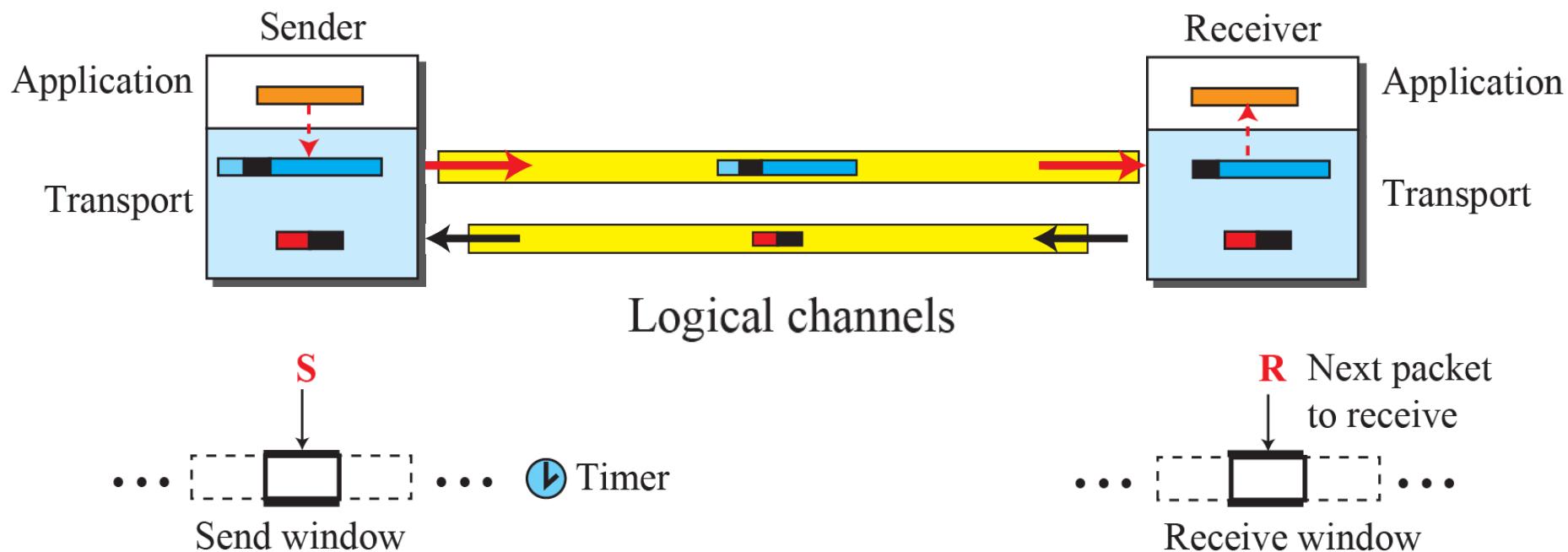
- It is connection oriented protocol that implements both error and flow control.
- Error control is implemented by adding checksum to each packet.
- When the packet arrives at the receiver site, it is checked; and if it is **corrupted**, it is silently **discarded**.
- Lost packets are more complex to handle than corrupted one.
- Unlike previous protocols, each packet is numbered for identification.
- The **corrupted and lost packet are resent as follows.**
  - The sender keeps a **copy** of each sent packet.
  - At the same time, it **starts a timer**.
  - If the timer expires and there is no ACK for the sent packet, **the packet is resent, the copy is held, and the timer is restarted.**

# Stop-and-Wait Protocol

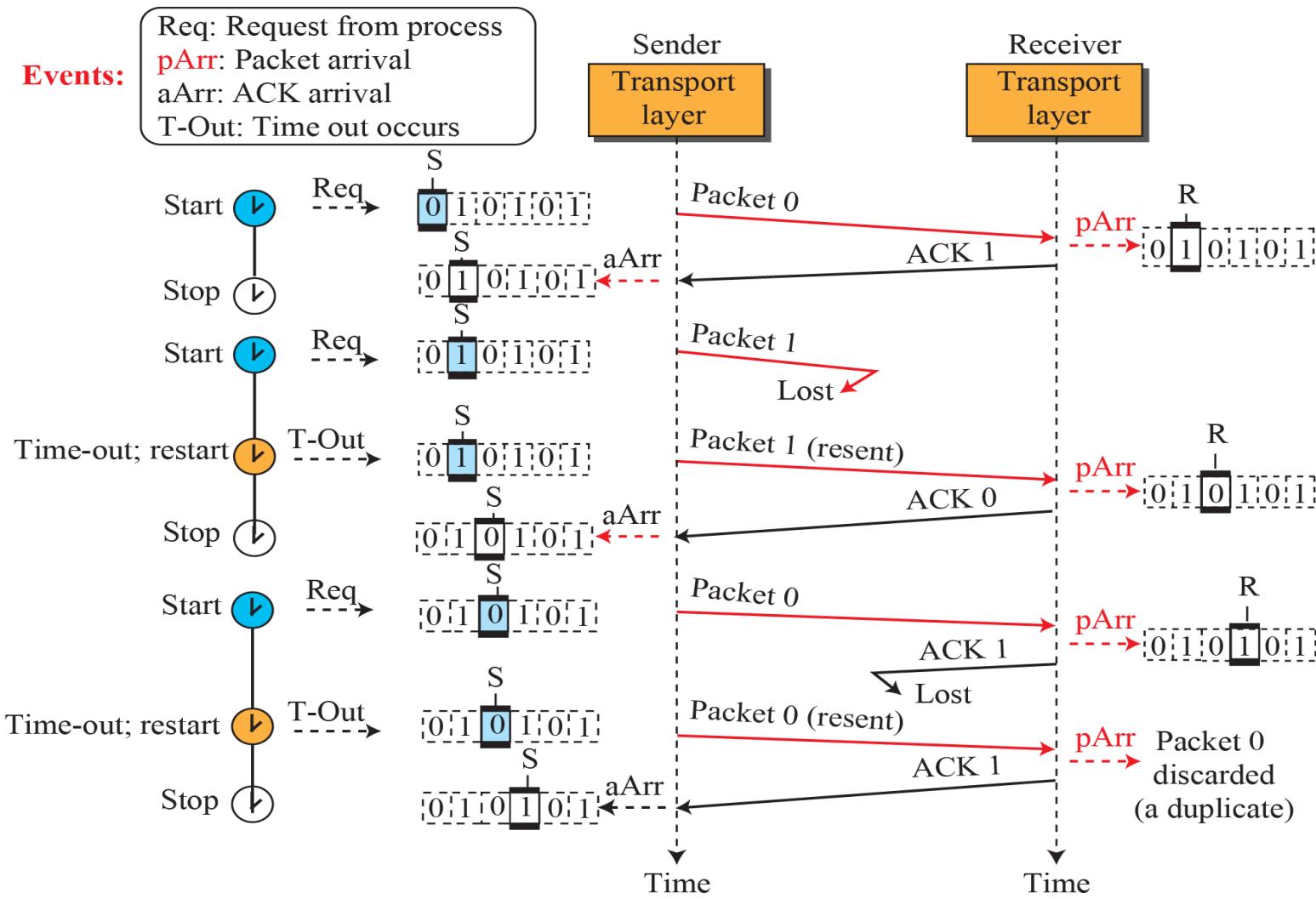
- **Sequence number (seqNo):** It is the packet being sent. Only 1-bit sequence number will suffice, i.e., either 1 or 0 can be the sequence number.
- This is bcz, only three things can happen when sender sends packet x.
  1. If packet arrives **safe and sound** at the Rx then receiver sends ACK x+1.
  2. If packet arrives safe and sound at the Rx, but **ACK is lost or corrupted** then sender resends packet x after timeout. Rx can recognize that this is a duplicate frame because it was expecting packet x+1 instead of x.
  3. The **packet never reaches** at the receiver site, sender resends the packet x.
- **Acknowledgement number (ackNo):** It is the sequence number of the next expected packet.

## For Stop-and-Wait Protocol, Illustration of

- Packets → seqNo and checksum
- ACK → ackNo and checksum
- Send and receive window size



**Example 3.3:** Packet 0 is sent and acknowledged. Packet 1 is lost and resent after the time-out. The resent packet 1 is acknowledged and the timer stops. Packet 0 is sent and acknowledged, but the acknowledgment is lost. **The sender has no idea if the packet 0 or the acknowledgment 1 is lost**, so after the time-out, it resends packet 0, which is acknowledged.



# Stop-and-Wait ARQ: Key points

- **Advantage:**
  - Easy to implement.
  - Requires minimum buffer size comparatively as the send and receive window size is same and equal to 1.
- **Disadvantage:**
  - It makes highly inefficient use of communication links.
  - Particularly, when the channel is thick (more bandwidth) and long (needs long time for ACK to reach to sender).

# Efficiency of Stop-and-Wait ARQ

- **Example 3.3:**

- Assume that, in a Stop-and-Wait system, the bandwidth of the line is 1Mbps, and 1 bit takes 20ms to make a round trip. What is the bandwidth delay product? If the system data packets are 1000 bits in length, what is the utilization percentage of the link?

**Sol:** *The bandwidth delay product is*

$$= (1 \times 10^6) \times (20 \times 10^{-3}) = 20,000 \text{ bits}$$

*The system can send 20,000 bits but it is actually sending only 1000 bits (packet size). Thus , the link utilization factor is only 1000/20000, or 5%.*



- **Example 3.4:**

- What is the utilization percentage of the link with bandwidth 1Mbps, and 1 bit takes 20ms to make a round trip, if we have a protocol that can send up to 15 packets before stopping and worrying about the acknowledgments?
- **Sol:** *The bandwidth delay product is same i.e., 20,000 bits.*

*But now a packet consists of 1000 bits and at a time we can send up to 15 packets. So, total number of bits during a round trip is....*

$$15,000/20,000, \text{ or } 75\%$$

**Conclusion:** The efficiency of the system can be increased if instead of sending only one packet at a time and waiting for acknowledgement, we can send more than one packet without waiting for the acknowledgement.

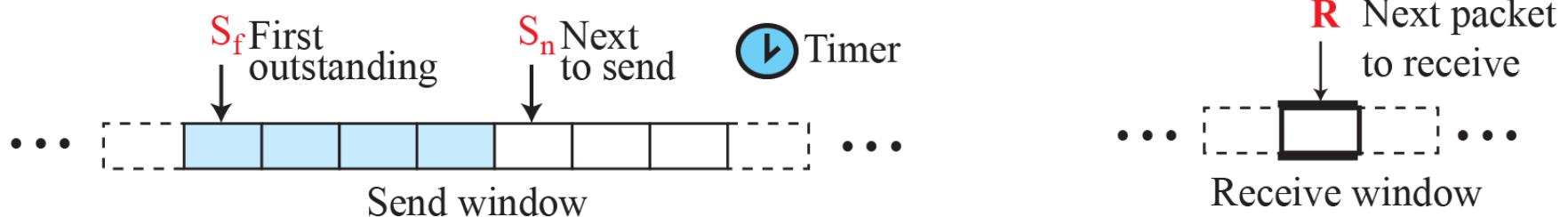
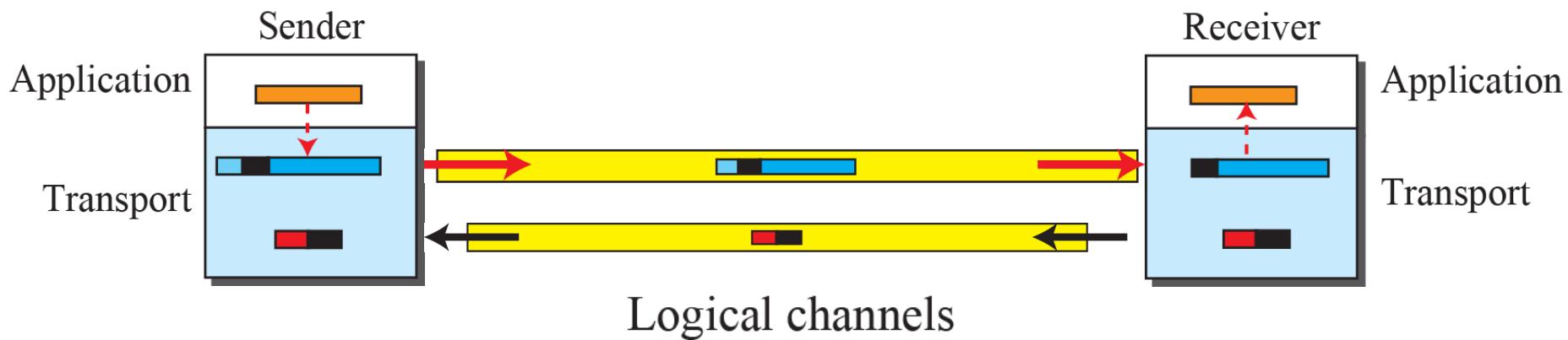
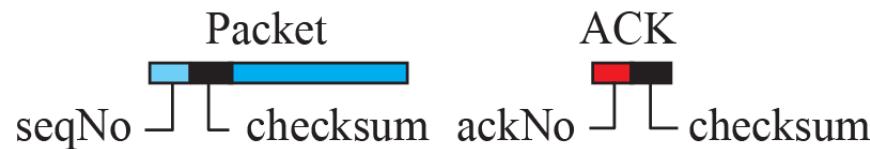


# Go-Back-N (GBN)

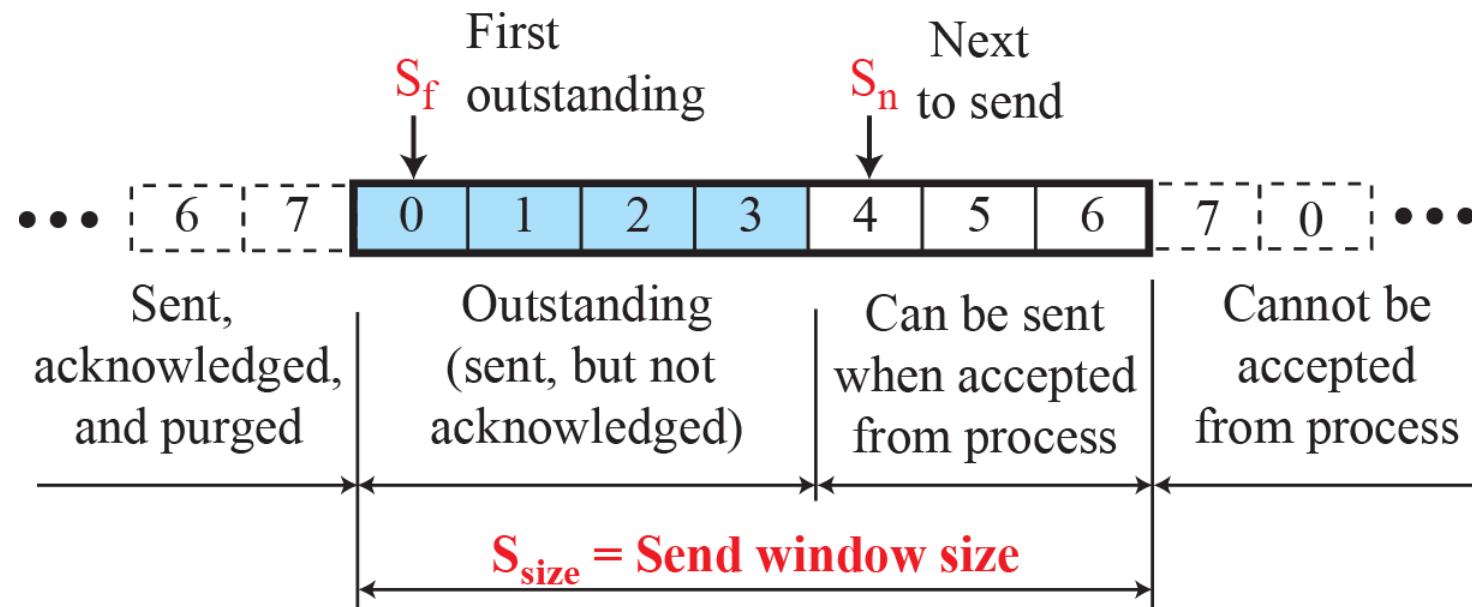
- **Key idea:** To improve link efficiency (filling the pipe) of Stop-and-Wait, multiple packets must be in transition while waiting for acknowledgement.
- In Go-Back-N protocol, several packets are sent before receiving acknowledgement.
- **Sequence number:** Because of multiple packets, one bit sequence number will not suffice. Generally if header allows  $m$ -bits for the sequence number, the sequence number ranges from 0 to  $2^m - 1$ , i.e. sequence numbers are modulo  $2^m$ . Sequence number can be repeated.
- **Acknowledgement number:** ackNo in this protocol is cumulative and defines the sequence number of the next packet expected.

## For Go-Back-N Protocol, Illustration of

- Packets → seqNo and checksum
- ACK → ackNo and checksum
- Send and receive window size

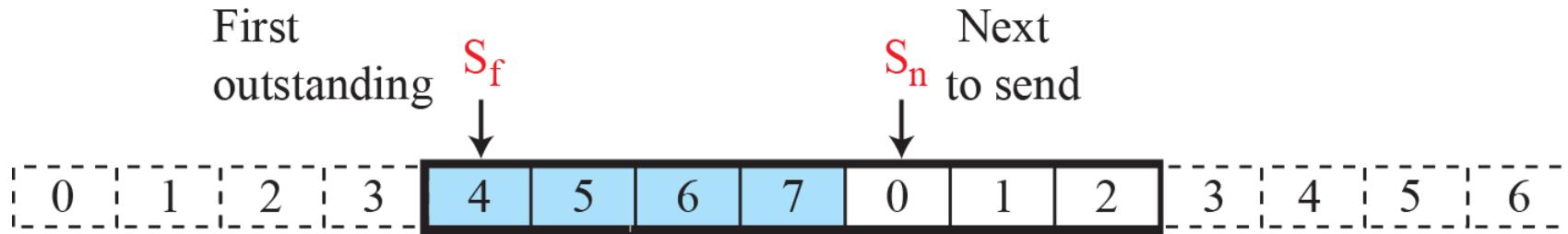


- **Sender Sliding Window:** It is an abstract concept defining an imaginary box of size  $2^m - 1$  with three variables  $S_f$ ,  $S_n$ , and  $S_{size}$ .



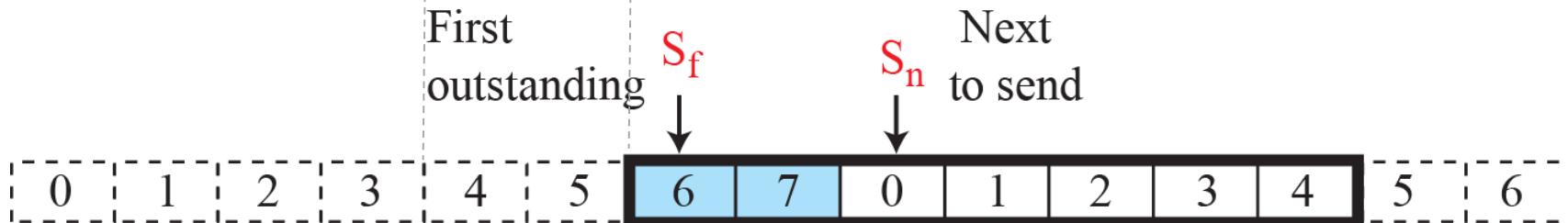
**Note:** The value of  $m$  is 4 in aforementioned example.

# Sliding of sending window



a. Window before sliding

*Sliding direction*



b. Window after sliding (an ACK with ackNo = 6 has arrived)

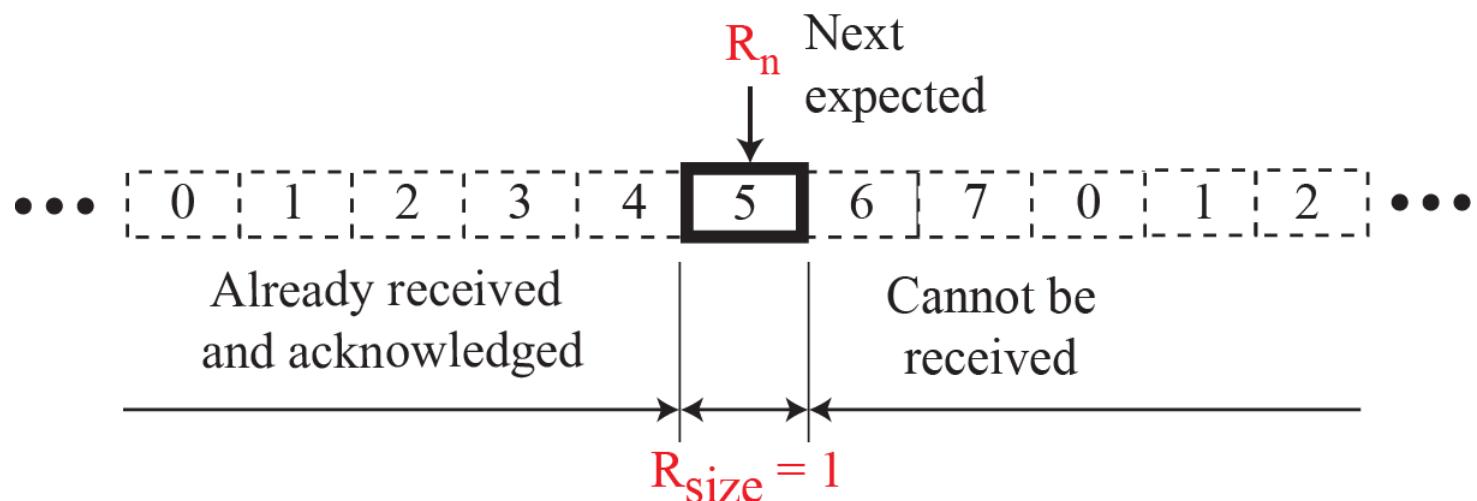
**Note:-** Send window can slide more than one slot

# Please try out.....

- Repeat the last example, i.e., draw the receiver window by explicitly mentioning sequence number for ‘m=3’. Also, repeat for ‘m=5’.



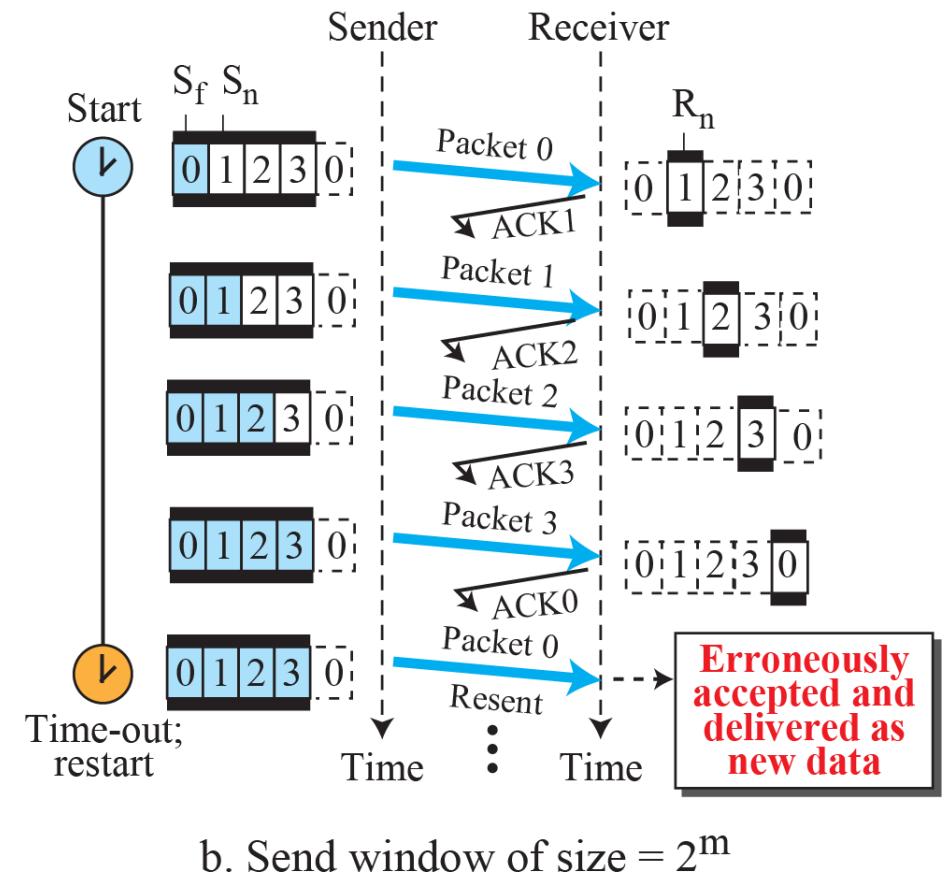
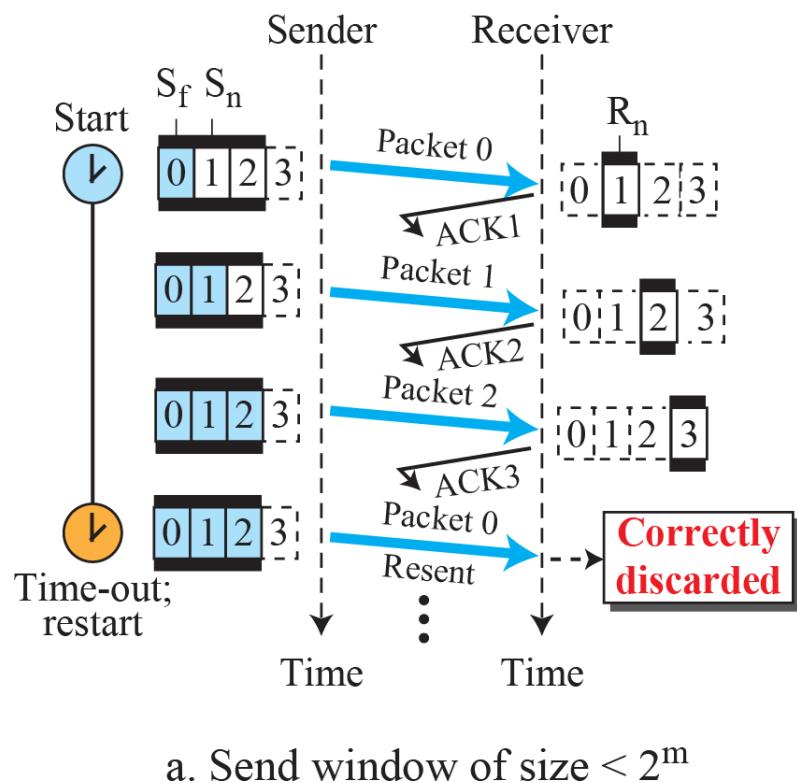
- **Receiver Sliding Window:** The receive window is an abstract concept defining an imaginary box of **size 1** with one single variable  $R_n$  (*receive window, next packet expected*). The window slides when a correct packet has arrived; sliding occurs one slot at a time.
- Only a packet with a sequence number matching the value of  $R_n$  is accepted and acknowledged.



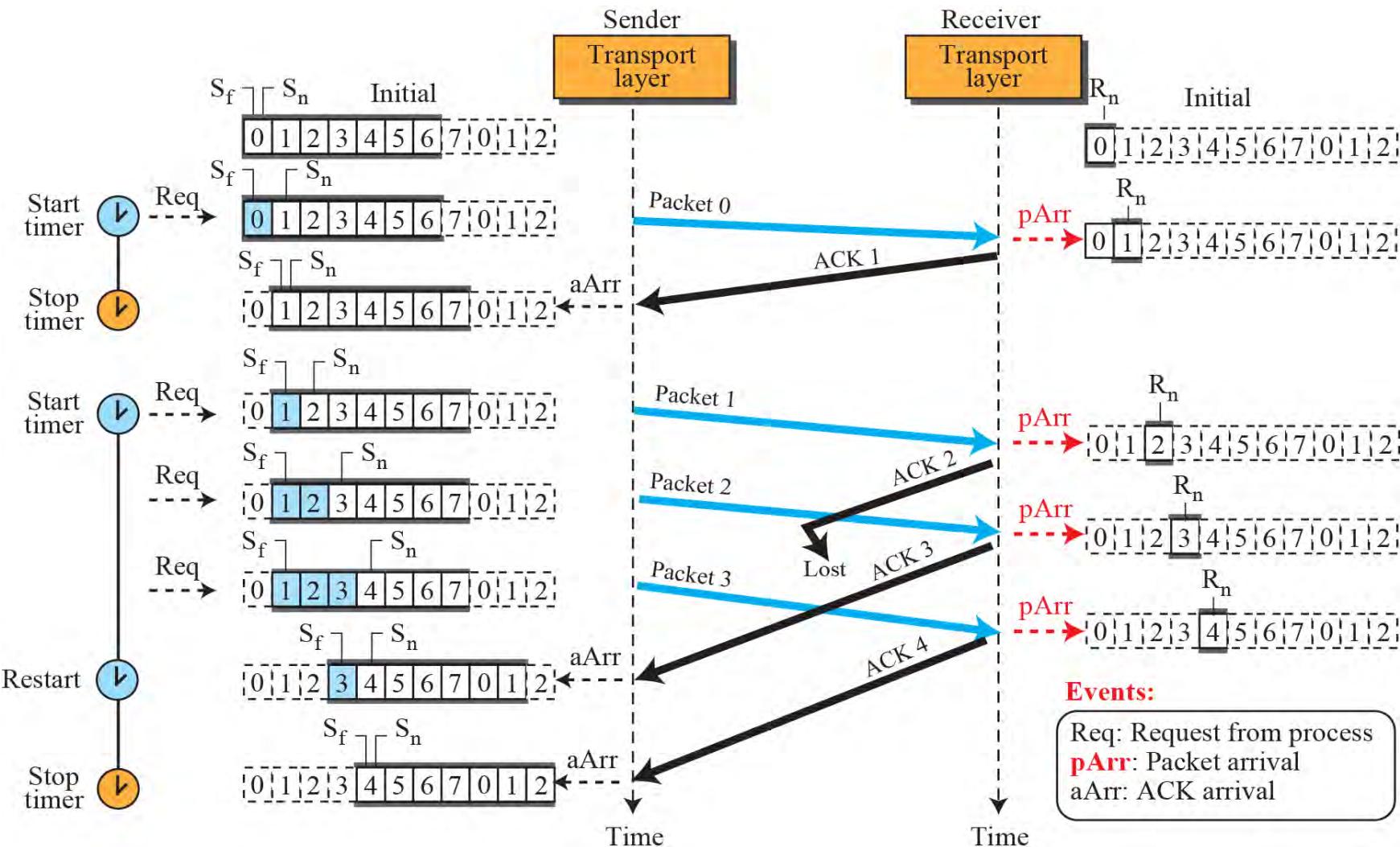
# Go-Back-N: Key points

- **Timers:** instead of having timer for each frame, we always keep timer **for the first outstanding frame**. This is because timer for the first outstanding frame always **expires first**. We send all outstanding frames when this timer expires.
- **Acknowledgement:** The receiver sends a positive acknowledgement if and only if a frame has arrived **safe, sound, and in order**. If a frame is damaged or is received out of order, the receiver is silent and will discard all subsequent frames until it receives the one it is expecting.
- **Resending frame:** Whenever the **timer expires**, the sender resends **all outstanding frame**. This is why this protocol is known as Go-Back-N ARQ

# Why to keep sender window size $2^m - 1$ ?



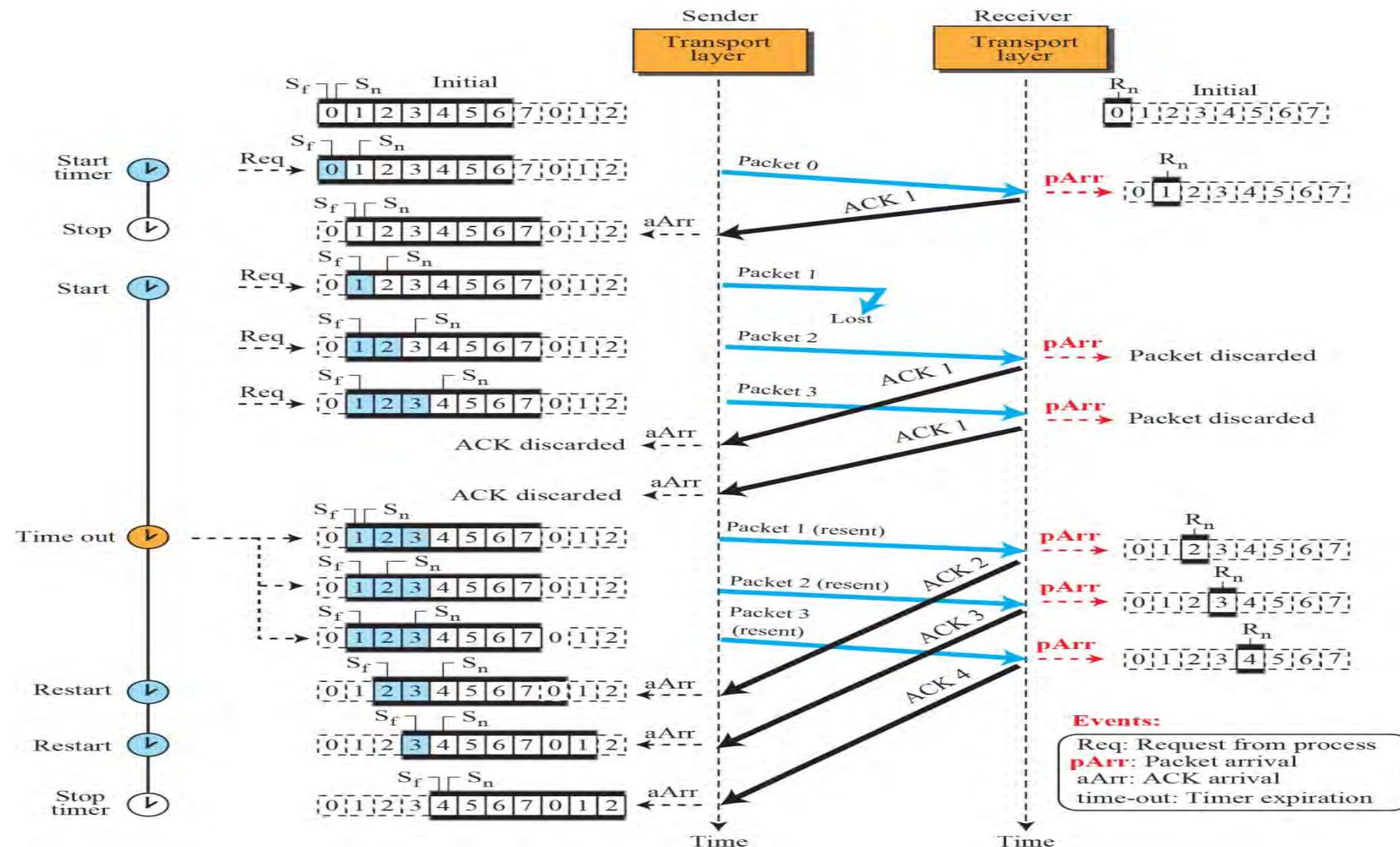
**Example 3.5:** Figure below shows an e.g. of Go-Back-N ARQ where the forward channel is reliable, but the reverse is not. No packets are lost, but some ACKs are delayed and one is lost. This example also shows how cumulative acknowledgements can help if acknowledgements are delayed or lost.



# Go-Back-N Vs. Stop-and-Wait

Stop-and-Wait	Go-Back-N
Sender window size is 1	Sender window size is $2^m - 1$
Receiver window size is 1	Receiver window size is 1
1 bit sequence number i.e., either 0 or 1	$m$ bit sequence number i.e., from 0 to $2^m - 1$

**Example 3.6:** For the example shown below, write all the events with description in your own words.



- **Hint for Example 3.6:**

- Packets 0, 1, 2 and 3 are sent.
- However, packet 1 is lost. The receiver receives packet 2 and 3, but they are discarded because they are received out of order.
- When the receiver receives packet 2 and 3, it sends ACK 1 to show that it expects to receive packet 1.
- However, these ACKs are not useful for the sender because the ackNo is equal to  $S_f$ , and not greater than  $S_f$ . So the sender discards them.
- When the timeout occurs, the sender resends packets 1, 2 and 3, which are acknowledged.



**Is this an efficient method to resend all N-packets even though there is error in any single packet out of N-packets?**



**Is this an efficient method to resend all N-packets even though there is error in any single packet out of N-packets?**

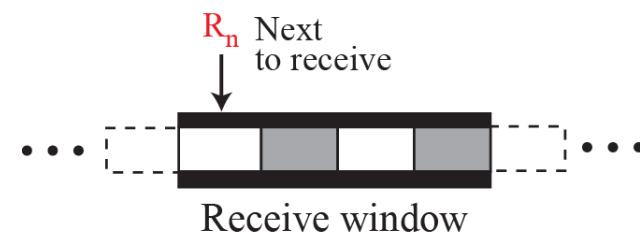
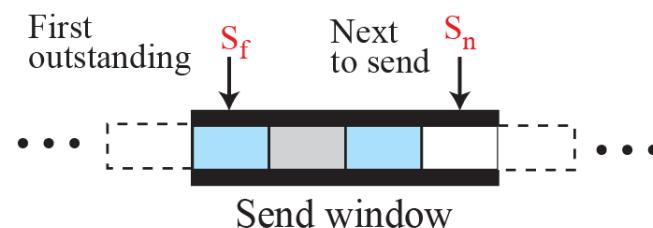
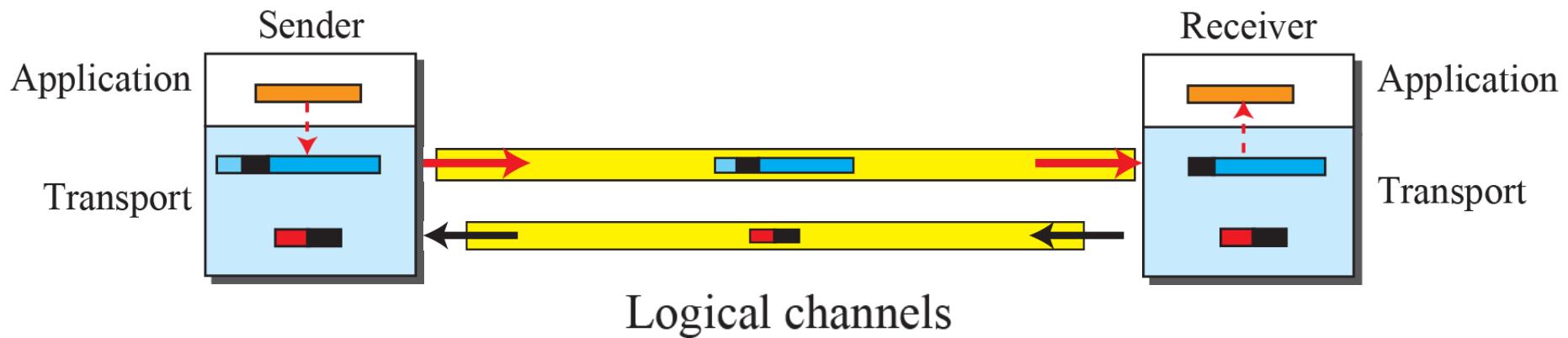
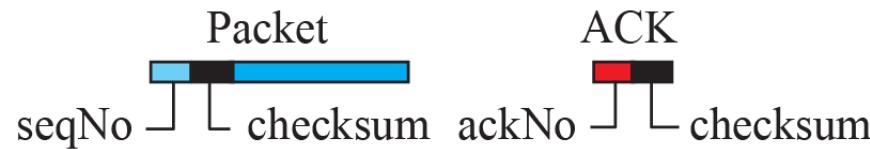
**Ans:** Obviously no...*otherwise we may waste some link bandwidth.*

# Selective-Repeat Protocol

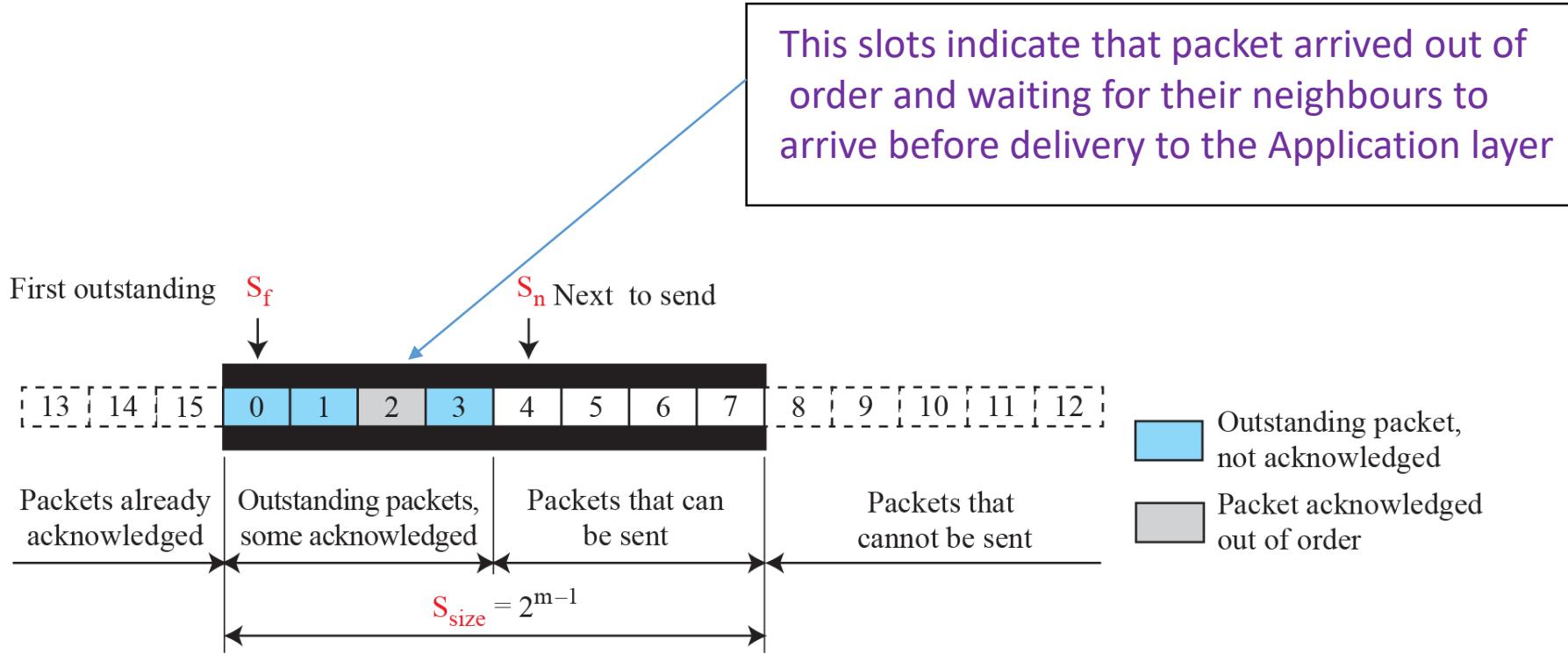
- In this protocol, receive window and send window are of same size.
- The maximum window size is much smaller than one used in Go-Back-N. Here it is  $2^{m-1}$ .
- E.g., for m=4 bit sequence number, sequence number go from 0 to 15 but the window size is only 8.
- The smaller window size means less efficiency in filling the pipe, but the fact that there are fewer duplicate packets can compensate for this.

## For Selective Repeat Protocol, Illustration of

- Packets → seqNo and checksum
- ACK → ackNo and checksum
- Send and receive window size

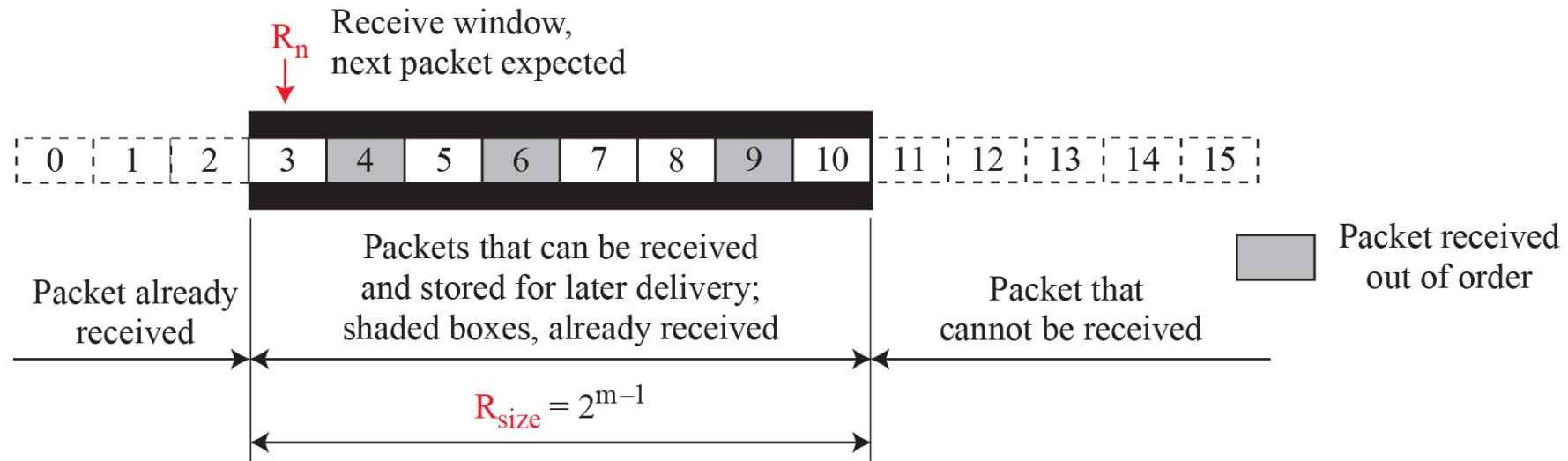


- **Sender Sliding Window:** It is an abstract concept defining an imaginary box of size  $2^{m-1}$  with three variables  $S_f$ ,  $S_n$ , and  $S_{size}$ .



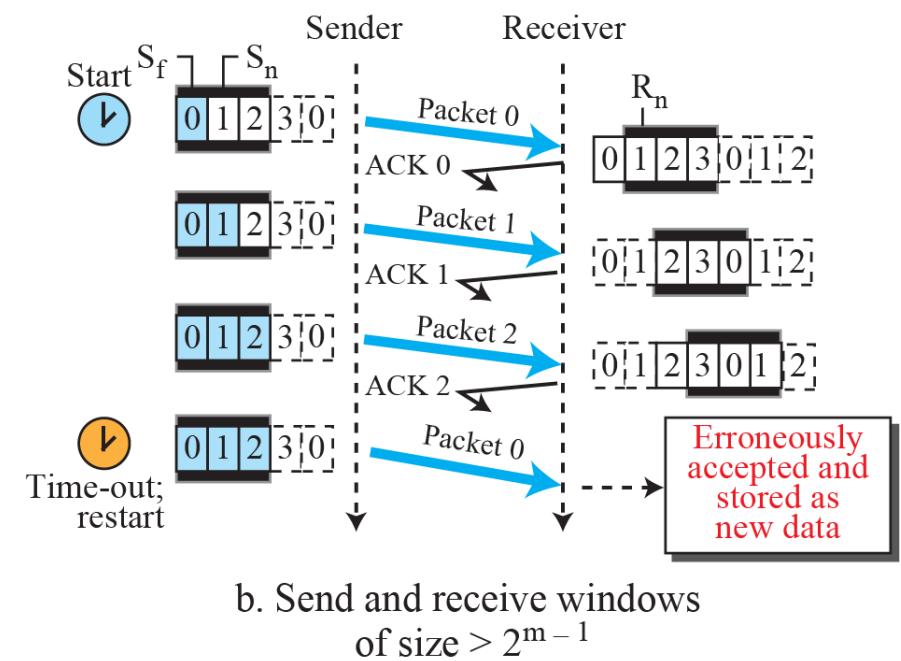
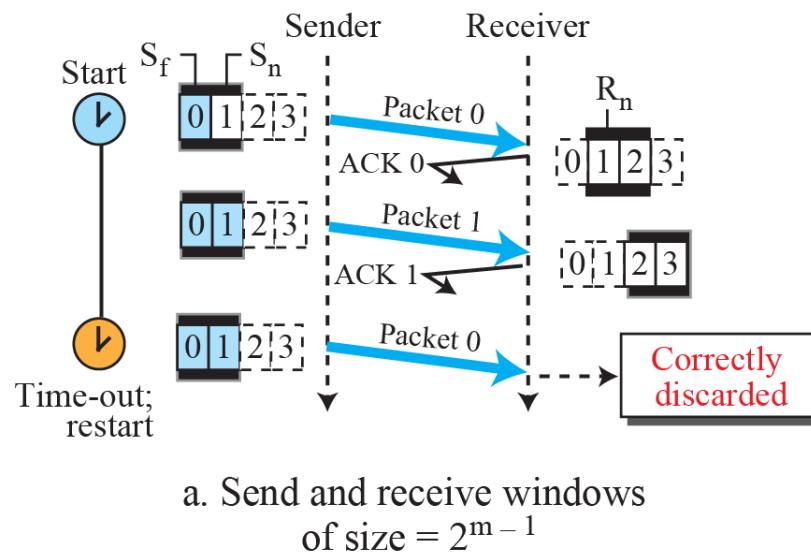
**Note:** The value of m is 4 in aforementioned example.

## Illustration of Receiving Window for Selective-Repeat

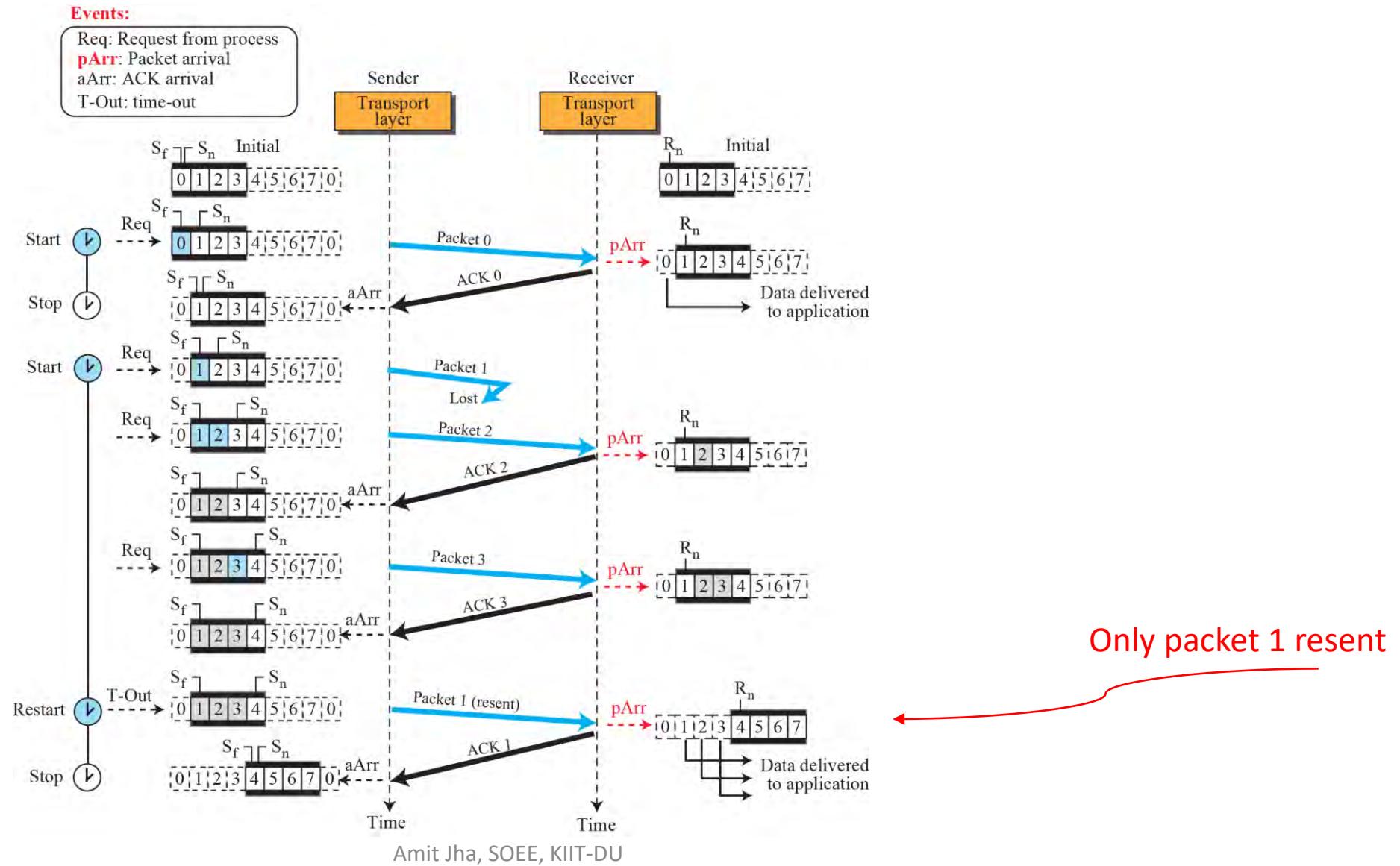


# Why window size is $2^{m-1}$ only?

- Let  $m=2$ , i.e., total four sequence numbers are 0, 1, 2, 3, Now, we choose window size 2 and 3 for analysis.



**Example 3.7:** It is similar to Example 3.6. Write description of all events in your own words.



**Example 3.8:** Assume a sender sends 6 packets: packets 0, 1, 2, 3, 4, and 5. The sender receives an ACK with ackNo = 3. What is the interpretation if

1. The system uses Go-Back-N protocol
2. The system uses Selective-Repeat protocol?

### Solution

- If the system uses Go-Back-N protocol, it means that packets 0, 1, and 2 have been received uncorrupted and the receiver is expecting packet 3.
- If the system uses Selective Repeat protocol, it means that packet 3 has been received uncorrupted; the ACK does not say anything about other packets.

### Now, Think....

- *An acknowledgement number in the Go-Back-N protocol defines the next packet expected, but an acknowledgement number in the Selective-Repeat protocol defines the sequence number of the packet to be acknowledged. Can you explain the reason?*



# Key Points

Stop-and-Wait	Go-Back-N	Selective-Repeat
Sender window size is 1	Sender window size is $2^m - 1$	Sender window size is $2^{m-1}$
Receiver window size is 1	Receiver window size is 1	Receiver window size is also $2^{m-1}$
1 bit sequence number i.e., either 0 or 1	$m$ bit sequence number i.e., from 0 to $2^m - 1$	$m$ bit sequence number i.e., from 0 to $2^m - 1$

# Please try out.....

- Calculate sequence number, sender window size, receiver window size, and draw the sender and receiver window explicitly for 'm=3'. Also, repeat for 'm=4' and 'm=5'.

**Note:** While explaining any of the three protocol discussed for noisy channel, please keep in mind the following questions.....

- 1) What is the range of sequence number and ACK number?
- 2) What is the significance of ACK number?
- 3) What is the size of sender window and receiver window?
- 4) What should be the position of  $S_f$ ,  $S_n$ ,  $R_n$  after every operation (sending frames and receiving ACK or NAK)?
- 5) Show that how does sender or receiver or both window slide explicitly.
- 6) Show sender and receiver window explicitly by drawing overlapping rectangle.
- 7) Show timer time out conditions as and when requires.

# *What is piggybacking?*



# Piggybacking



**Disclaimer:-** The images are taken from the Google for explaining the concept to students. Its intention is not to hurt sentiments of any students or person portrayed in this.

Amit Jha, SOEE, KIIT-DU

# Piggybacking



- The three protocols we discussed in this section are all unidirectional: data packets flow in only one direction although control information such as ACK and NAK packets can travel in the other direction.
- In real life, data packets are normally flowing in both directions. This means that the control information also needs to flow in both directions.
- A technique called **piggybacking** is used to improve the efficiency of the bidirectional protocols.
- When a packet is carrying data from A to B, it can also carry control information about arrived (or lost) packets from B; when a packet is carrying data from B to A, it can also carry control information about the arrived (or lost) packet from A.

**Disclaimer:-** The images are taken from the Google for explaining the concept to students. Its intention is not to hurt sentiments of any students or person portrayed in this.

# CN (IT-3001)

## Transport Layer: Transport Protocols

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University

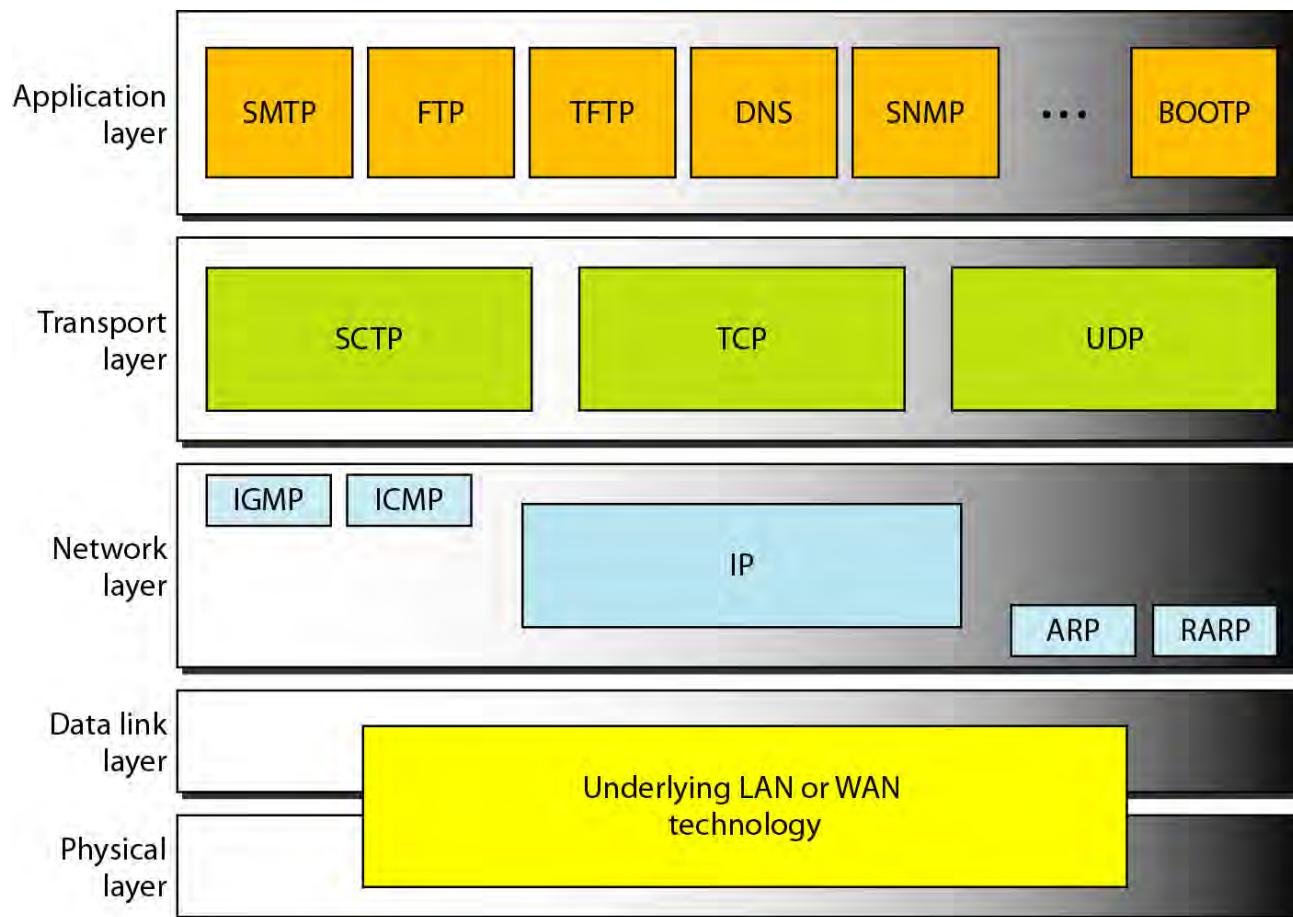


**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Transport Protocols

- We discuss following protocols in the subsequent slides.
  1. User Datagram Protocol (UDP)
  2. Transmission Control Protocol (TCP)

# Position of UDP, TCP and SCTP in TCP/IP Suite



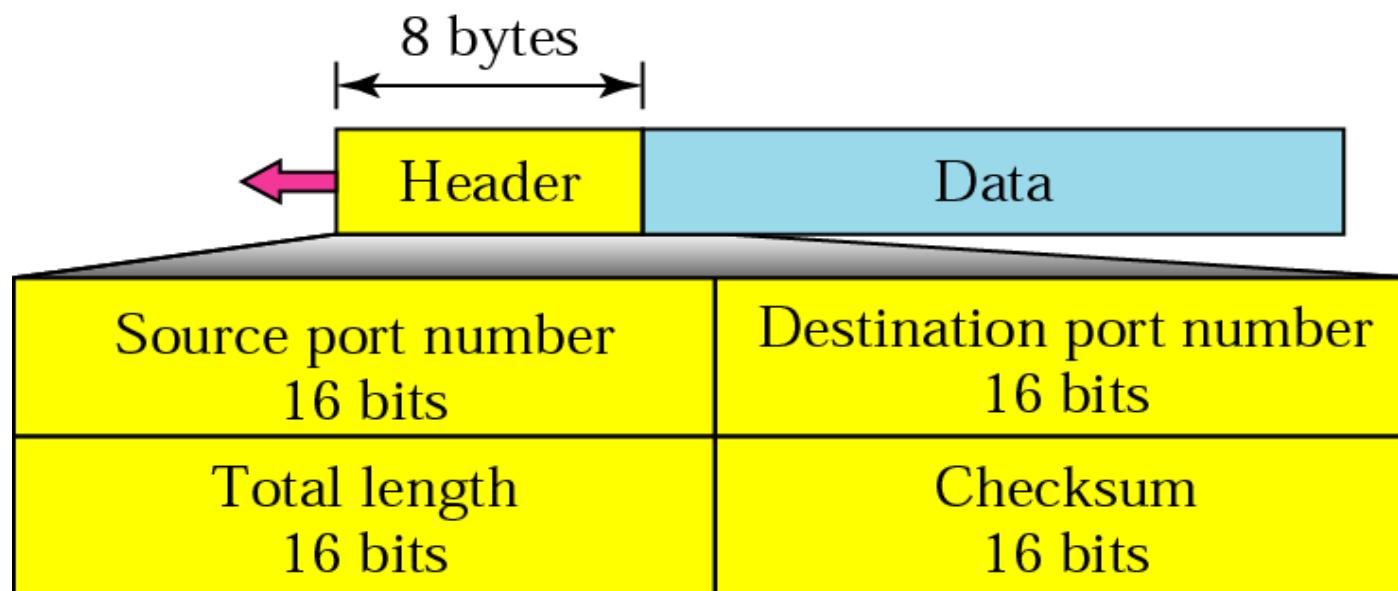
# User Datagram Protocol (UDP)

- It is a **connectionless** and thus, **unreliable** transport protocol.
- It does not add anything to the service of IP except to provide process-to-process communication instead of host-to-host.
- It **does not support** flow control thus, no window mechanism.
- There is **no error control** mechanism except for the checksum.
- To handle multiple processes there is provision of a *queue*.

**Q) If UDP is so powerless, why would a process want to use it?**

**Ans:** UDP is very simple protocol using a minimum of overhead. If a process wants to send a small message and does not care much about reliability, it can use UDP.

# UPD header format



# Application of UDP:

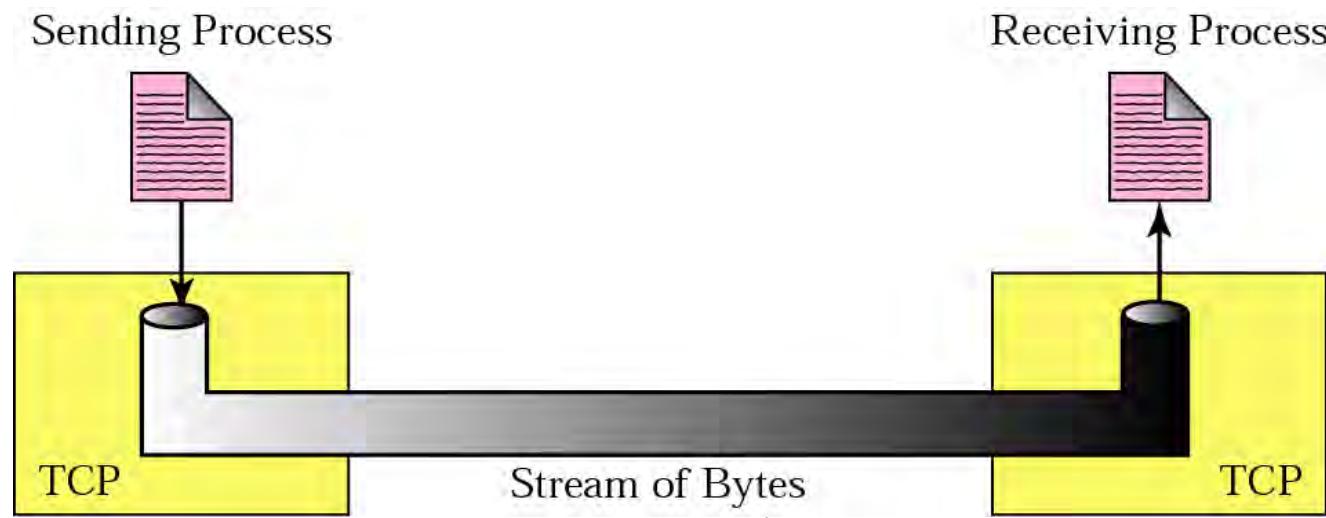
- UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control. It is not usually used for a process such as **FTP that needs to send bulk data**.
- UDP is suitable for a process with internal flow and error control mechanisms. For example, the **Trivial File Transfer Protocol(TFTP)** process includes flow and error control. It **can easily use UDP**.
- UDP is a suitable transport protocol for **multicasting**. Multicasting capability is embedded in the UDP software but not in the TCP software.
- UDP is used for management processes such as **SNMP, DNS**.
- UDP is used for some route updating protocols such as **Routing Information Protocol(RIP)**.

# Transmission Control Protocol (TCP)

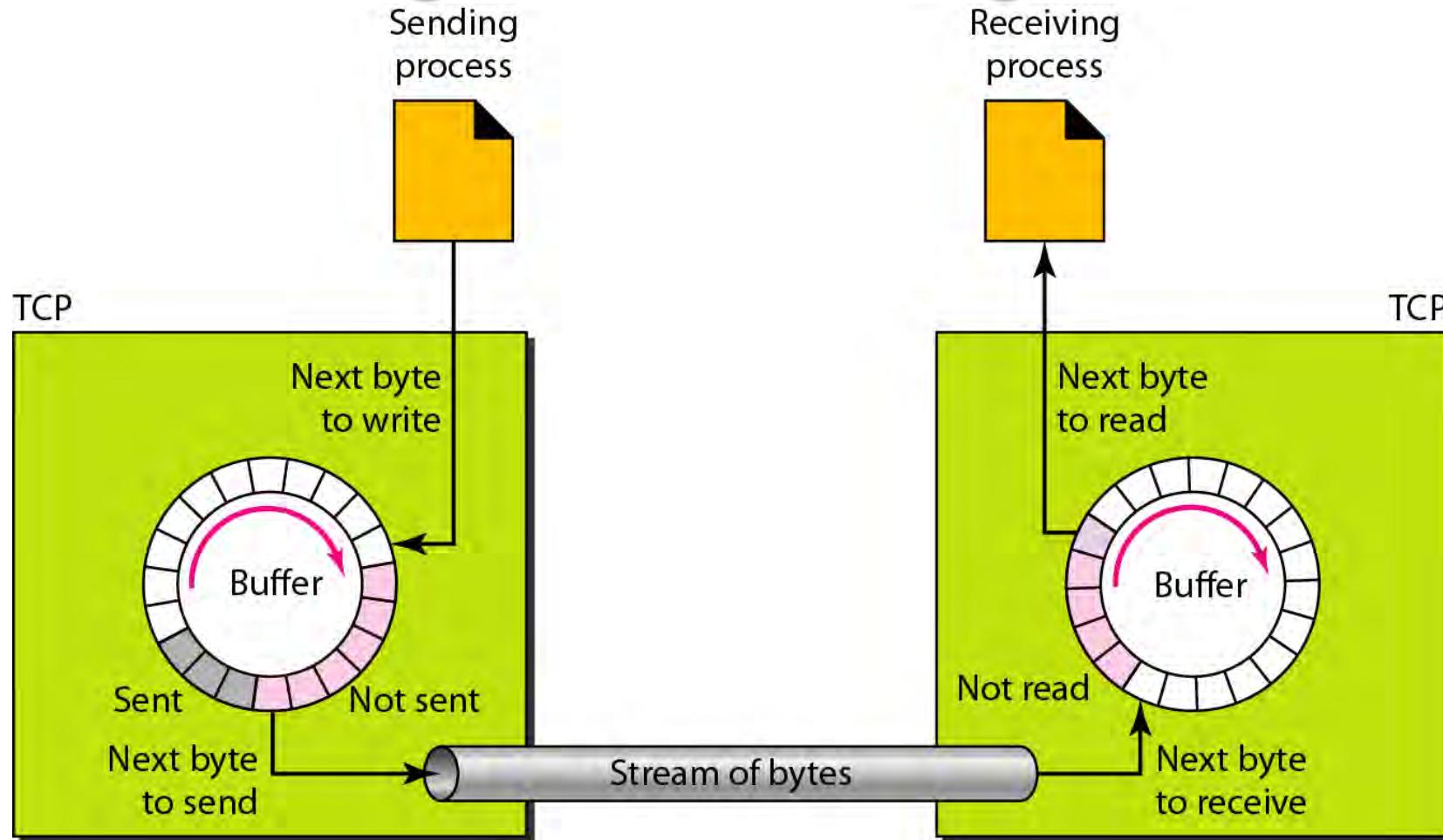
- It is a connection oriented protocol.
- It provides *process-to-process* communication.
- It is reliable and provides flow and error control mechanism.
- It is a stream oriented protocol.
- Provides full duplex communication.
- It supports multiplexing and demultiplexing.
- It also implements the congestion control mechanism.

# Transmission Control Protocol (TCP)

- Like UDP, it provides process-to-process communication using port number.
- It is a *connection-oriented*, thus *reliable* transport protocol.
- Unlike UDP, it is **a stream oriented protocol**.
- Supports **full-duplex service**.

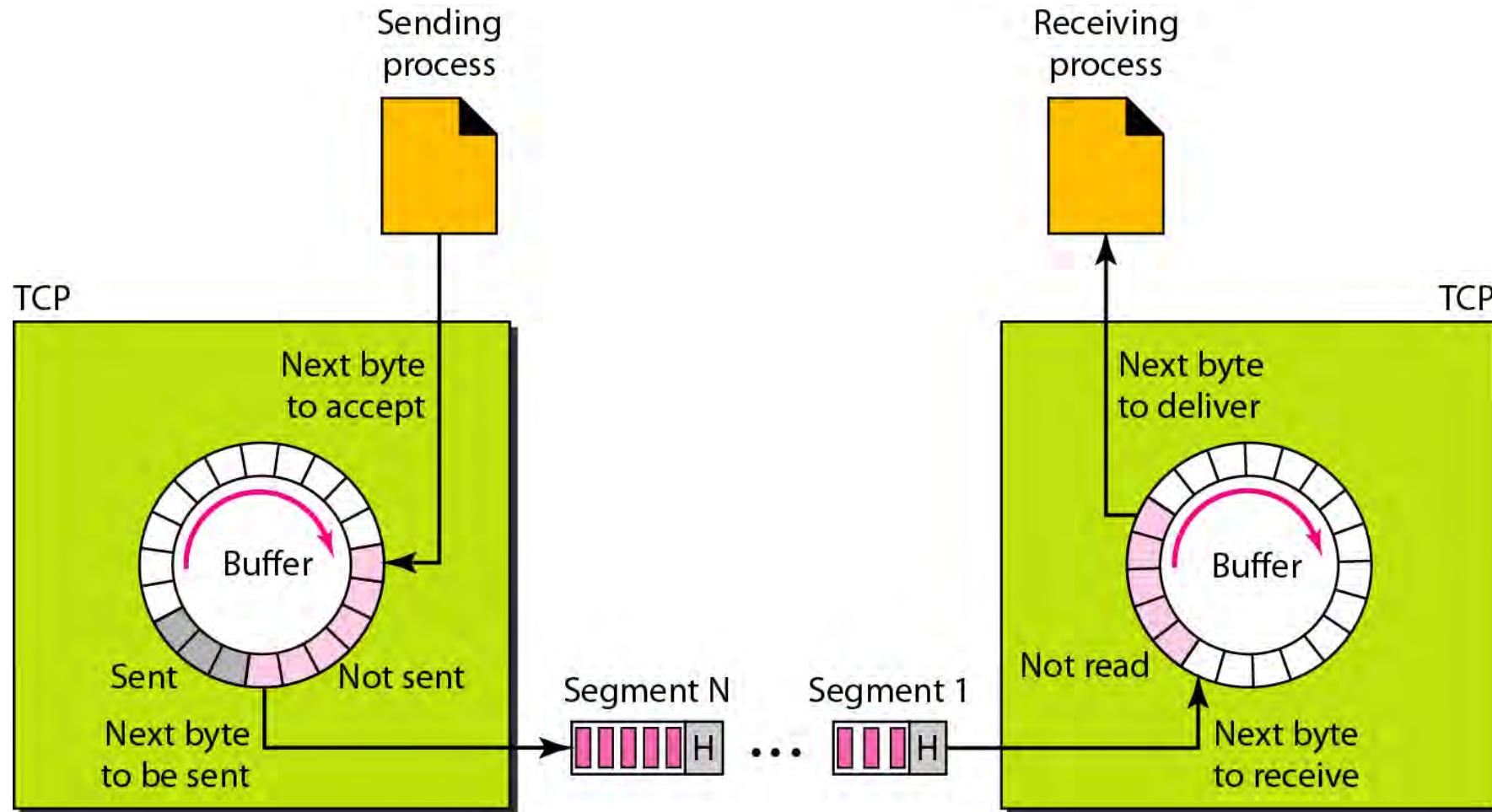


# TCP: Sending and Receiving buffer



**Note:** It is needed when sender and receiver do not operate at the same speed.

# TCP Segments



The IP layer, as a service provider for TCP, needs to send data in packets, not as a stream of bytes. At the transport layer, TCP groups a number of bytes together into a packet called a segment. TCP adds header to each segment.

# TCP Features

Several features of TCP to provide various services as discussed before, are described below:

- **Numbering System:** The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number between 0 to  $2^{32} - 1$ .
- **Sequence Number:** After the bytes have been numbered, TCP assigns a sequence number to each segment that is being sent. **The sequence number for each segment is the number of the first byte carried in that segment.**
- **Acknowledgement Number:** The acknowledgment number defines the number of the next byte that the party expects to receive. It is cumulative in nature.

**Example 3.9:** Imagine a TCP connection is transferring a file of 6000 bytes. The first byte is numbered 10,001. What are the sequence numbers for each segment if data are sent in five segments with the first four segments carrying 1000 bytes and the last segment carrying 2000 bytes?

**Solution:**

The following shows the sequence number for each segment:

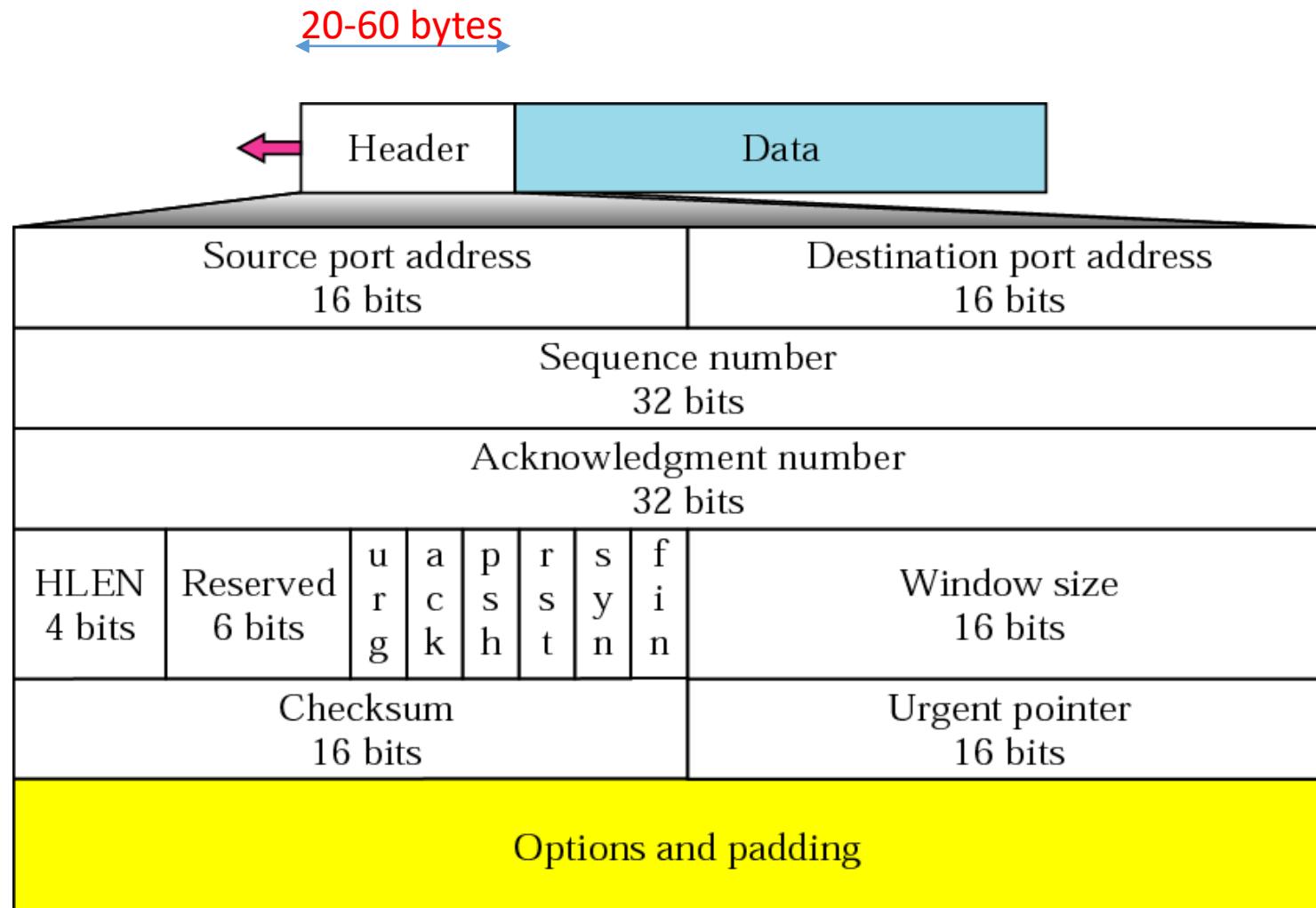
- Segment 1 ==> sequence number: 10,001 (range: 10,001 to 11,000)
- Segment 2 ==> sequence number: 11,001 (range: 11,001 to 12,000)
- Segment 3 ==> sequence number: 12,001 (range: 12,001 to 13,000)
- Segment 4 ==> sequence number: 13,001 (range: 13,001 to 14,000)
- Segment 5 ==> sequence number: 14,001 (range: 14,001 to 16,000)

# TCP Features....contd

- **Flow Control:** Unlike UDP, TCP provides flow control. The numbering system allows TCP to use a byte-oriented flow control.
- **Error Control:** To provide reliable service, TCP implements a byte-oriented error control mechanism.
- **Congestion Control:** TCP, unlike UDP, takes into account congestion in the network.

**Note:** A packet in TCP is called segment.

# TCP Header Format



# TCP Header...contd

- **Header length (HLEN):** This 4-bit field indicates the length of the header in the units of 4-byte words.
- **Reserved:** This is a 6-bit field reserved for future use.
- **Control:** This field defines 6 different control bits or flags. One or more of these bits can be set at a time.

URG: Urgent pointer is valid	RST: Reset the connection
ACK: Acknowledgment is valid	SYN: Synchronize sequence numbers
PSH: Request for push	FIN: Terminate the connection



# TCP Header...contd

- **Window size:** This field defines the size of the window, in bytes, that the other party must maintain. As the length of this field is 16 bits, the maximum size of the window is 65,535 bytes.
- **Checksum:** The calculation of the checksum for TCP follows the same procedure as the one described for UDP. However, the inclusion of the checksum in the UDP datagram is optional, whereas the inclusion of the checksum for TCP is mandatory.
- **Urgent pointer:** This 16-bit field, which is valid only if the urgent flag is set, is used when the segment contains urgent data. It defines the number that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment.

# TCP Connection

- TCP is a *connection-oriented* protocol.
- When a process at site A wants to send and receive data from another process at site B, the following events occur:
  1. ***Connection Establishment:*** The two TCPs establish a connection between them.
  2. ***Data Transfer:*** Data are exchanged in both directions.
  3. ***Connection Termination:*** The connection is terminated.
- **Note:** This is a virtual connection, not a physical connection.

# ***Connection Establishment: A 3-Way Handshake***

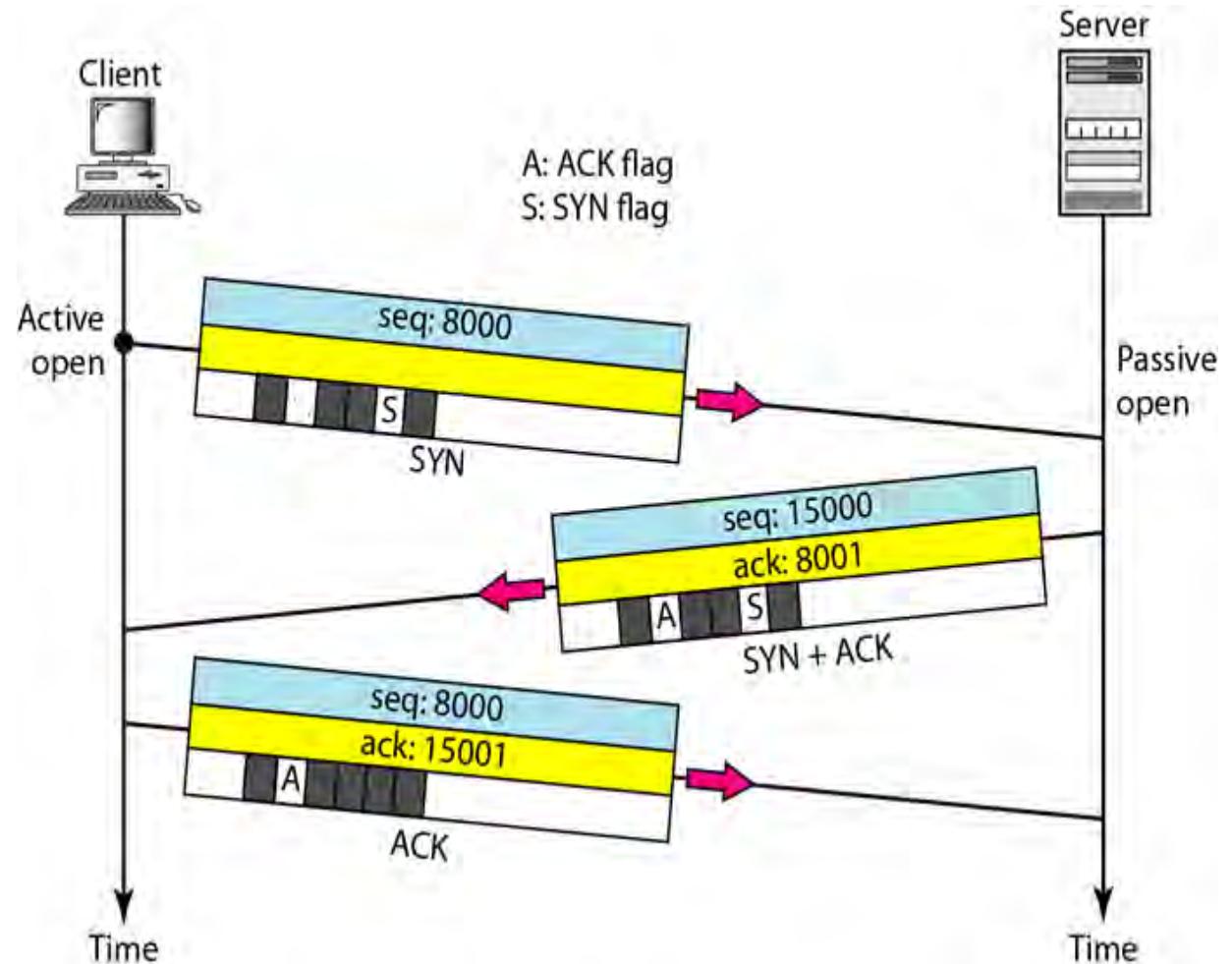
- The connection establishment in TCP is called three-way handshaking.
- **Explanation:** Consider an application program, called the client, wants to make a connection with another application program, called the server, using TCP as the transport layer protocol.
- The process starts with the server. The server program tells its TCP that it is ready to accept a connection. This is called a request for a ***passive open***.
- The client program issues a request for an ***active open***. A client that wishes to connect to an open server tells its TCP that it needs to be connected to that particular server. TCP can now start the three-way handshaking process.

# Illustration of a 3-Way Handshaking for *Connection Establishment*



## Questions

- Why does SYN carry no data?
- Why sequence no is required?
- In case of sequence no, what should be the value of sequence no?



# ...contd

1. **First:** The client sends the first segment, a **SYN** segment, in which only the SYN flag is set. This segment is for synchronization of sequence numbers.
  - Note: **A SYN segment can not carry data, but it consumes one sequence number.**
2. **Second:** The server sends the second segment, a **SYN+ACK** segment, with 2 flag bits set: SYN and ACK. This segment has a dual purpose. It is a SYN segment for communication in the other direction and serves as the acknowledgment for the SYN segment.
  - Note: **A SYN+ACK segment cannot carry data, but does consume one sequence number.**
3. **Third:** The client sends the third segment. This is just an **ACK** segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. Note that the sequence number in this segment is the **same as** the one in the SYN segment.
  - Note: **An ACK segment, if carrying no data, consumes no sequence number.**

# Data Transfer in TCP

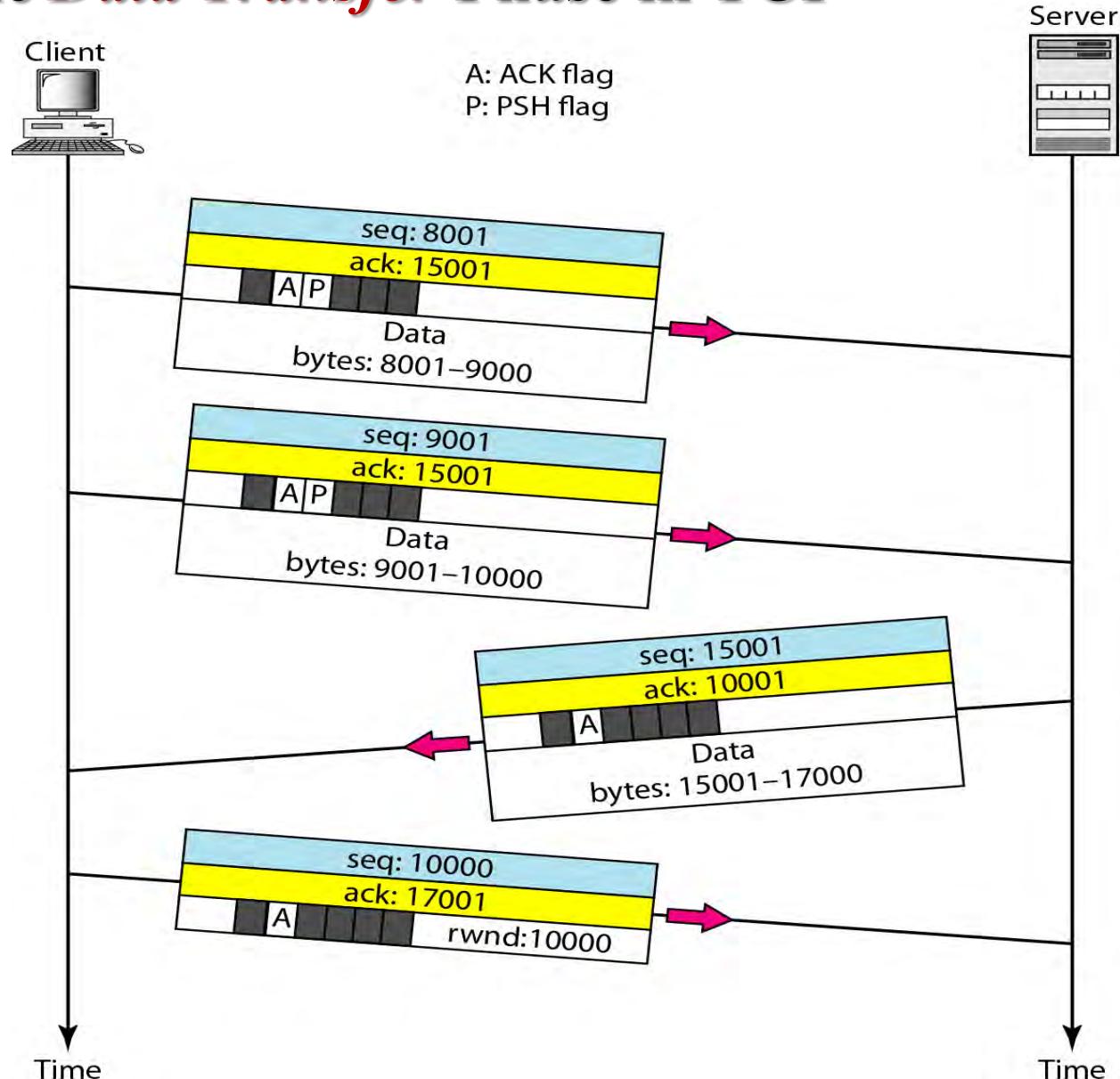
- After connection is established, the client and server can both send data and acknowledgments.

**Explanation:** After connection is established, the client sends 2000 bytes of data in two segments. The server then sends 2000 bytes in one segment. The client sends one more segment. The first three segments carry both data and acknowledgment, but the last segment carries only an acknowledgment because there are no more data to be sent.

**Note:** The data segments sent by the client have the PSH(push) flag set so that the server TCP knows to deliver data to the server process as soon as they are received.

# Illustration of the *Data Transfer* Phase in TCP

**Note:** The acknowledgment is piggybacked with the data.

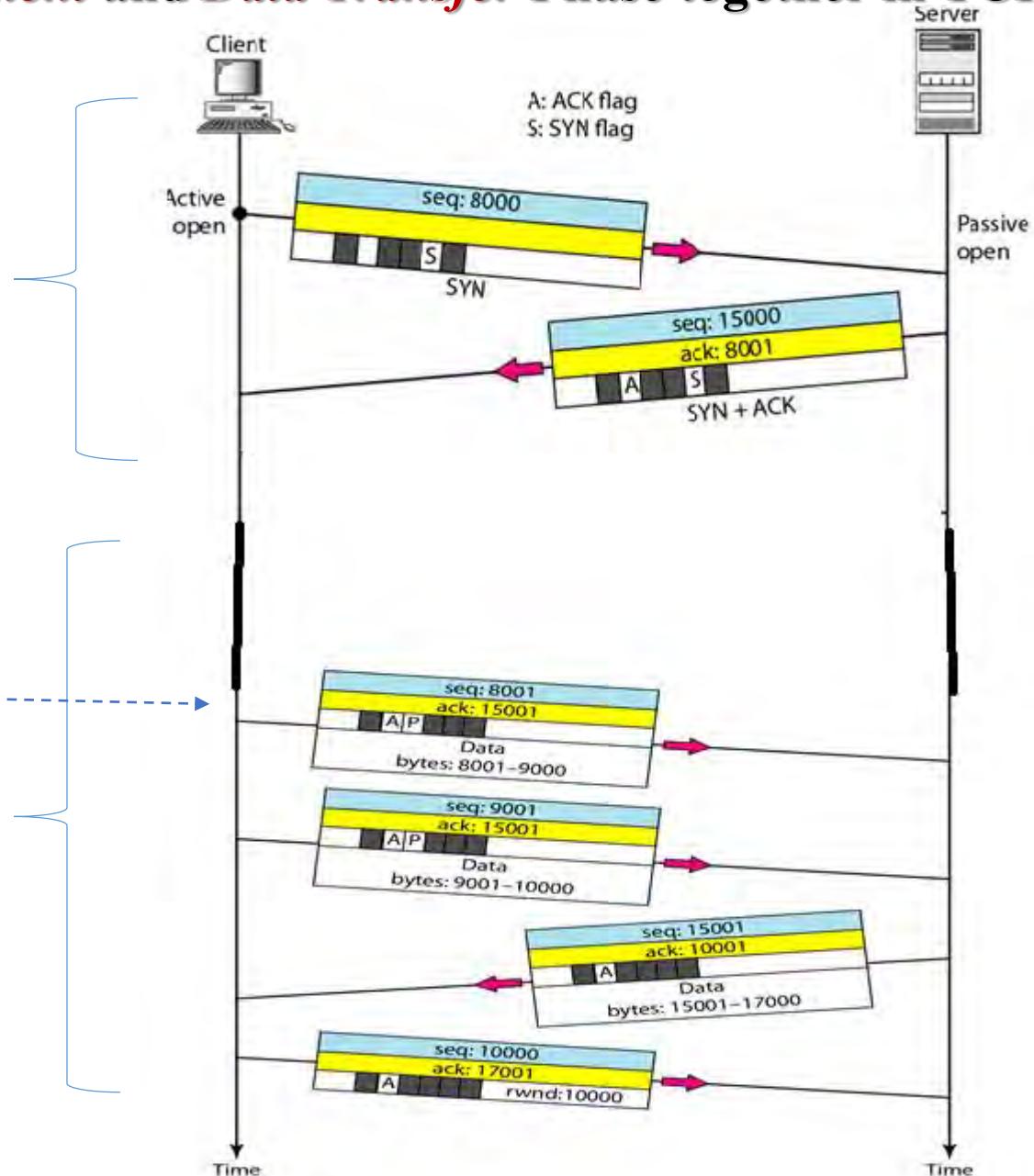


# Illustration of the *connection establishment* and *Data Transfer* Phase together in TCP

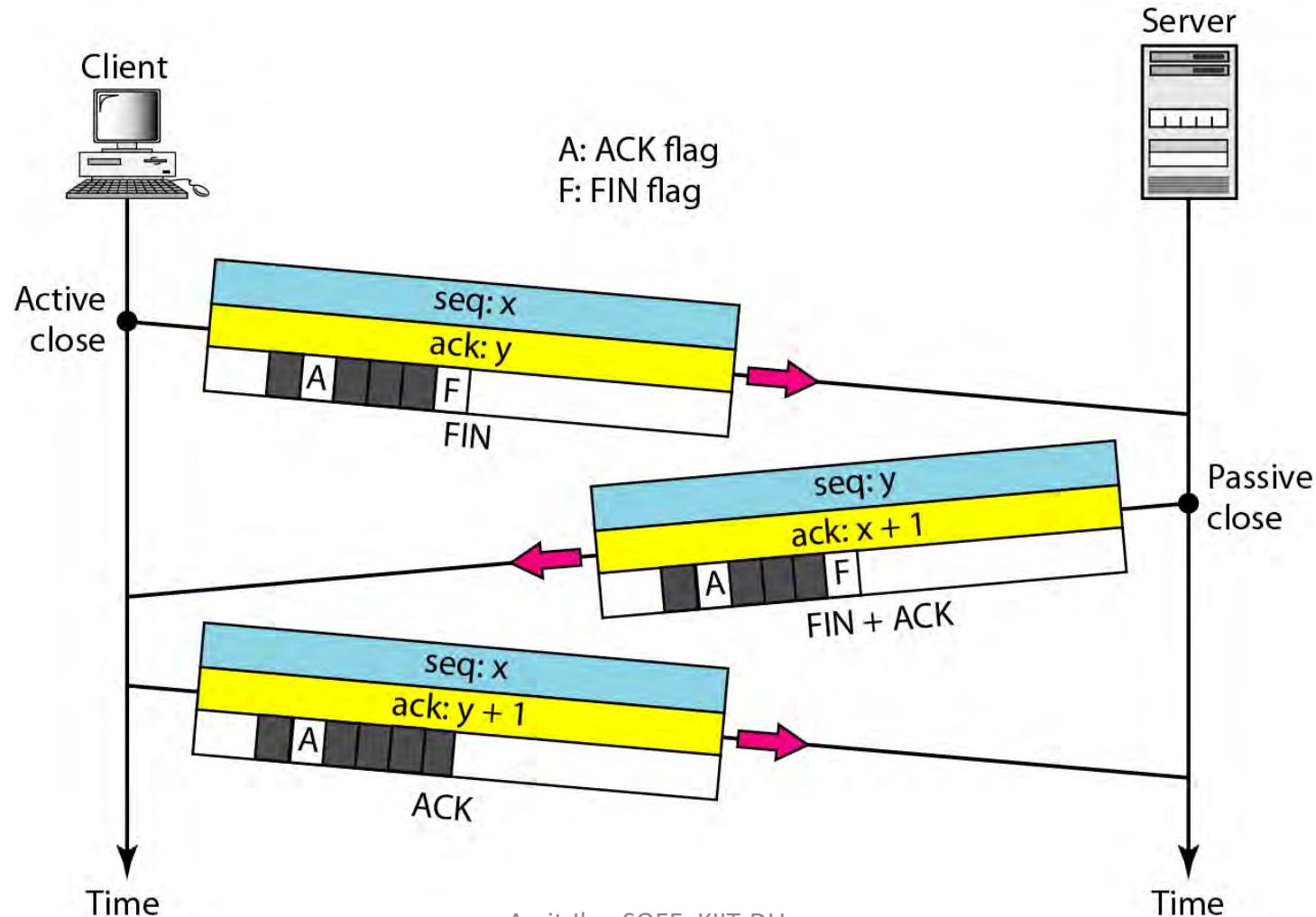
Three-way handshaking for connection establishment

Piggybacking

Data Transfer



# A 3-Way Handshaking for *Connection Termination*



## ...contd

- Generally initiated by client.
1. **First:** In a normal situation, the client TCP, after receiving a close command from the client process, sends the first segment, a **FIN segment** in which the FIN flag is set. Note that a **FIN segment can include the last chunk of data sent by the client.**
    - **Note:** The **FIN** segment consumes one sequence number if it does not carry data.
  2. **Second:** The server TCP, after receiving the FIN segment, informs its process of the situation and sends the second segment, a **FIN+ACK segment**, to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the connection in the other direction. **This segment can also contain the last chunk of data from the server.**
    - **Note:** The **FIN+ACK** segment consumes one sequence number if it does not carry data.
  3. **Third:** The client TCP sends the last segment, an **ACK segment**, to confirm the receipt of the FIN segment from the TCP server.
    - **Note:** This segment can not carry data and consumes no sequence numbers.

# Half Close

- Is half close possible?
  - If yes, then elaborate.
- 
- HYU:
    - With the help of **state transition diagram**, explain the process of *connection establishment, data transfer* and *connection termination*.



# Homework

- Draw the client/server diagram for all three phases of TCP connection like; connection establishment, data transfer, and connection termination together.

# Homework

1. The clients and server wants to communicate with each other using TCP. The client wants to send 2000 bytes of data in two segments of 1000 bytes each. Then server responds by sending 2000 bytes of data in one segment only. Explain the process explicitly by showing: 3-way handshaking for connection establishment, data transfer, and 3-way handshaking for connection termination. Show only the relevant fields of TCP header.
2. In relevant to TCP, explain the process of 3-way handshaking for connection establishment, data transfer and 3-way handshaking for connection termination for the following conditions:
  - a. Client sends two segments namely, Seg1.1 and Seg1.2 of 1000 bytes each.
  - b. Server sends two segments namely, Seg2.1 and Seg2.2 of 1000 and 1500 bytes resp.
  - c. Client sends Seg1.3 of size 2000 bytes.
  - d. Finally, Server replies with Seg2.3 of 500 bytes.

# TCP Key- Points

- Features:
  - Connection-oriented
  - Byte-stream service
- Advantages:
  - Reliable, Full duplex
  - implements error control, flow control and congestion control
- Disadvantages:
  - More Complex Transmitter and receiver
  - more overhead as compared to UDP
  - Higher delay than UDP
- Applications: Most of the applications use TCP. To name a few:
  - HTTP, SMTP, FTP, TELNET, POP3, DHCP,.....many more

# CN (IT-3001)

## Transport Layer: Congestion Control

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

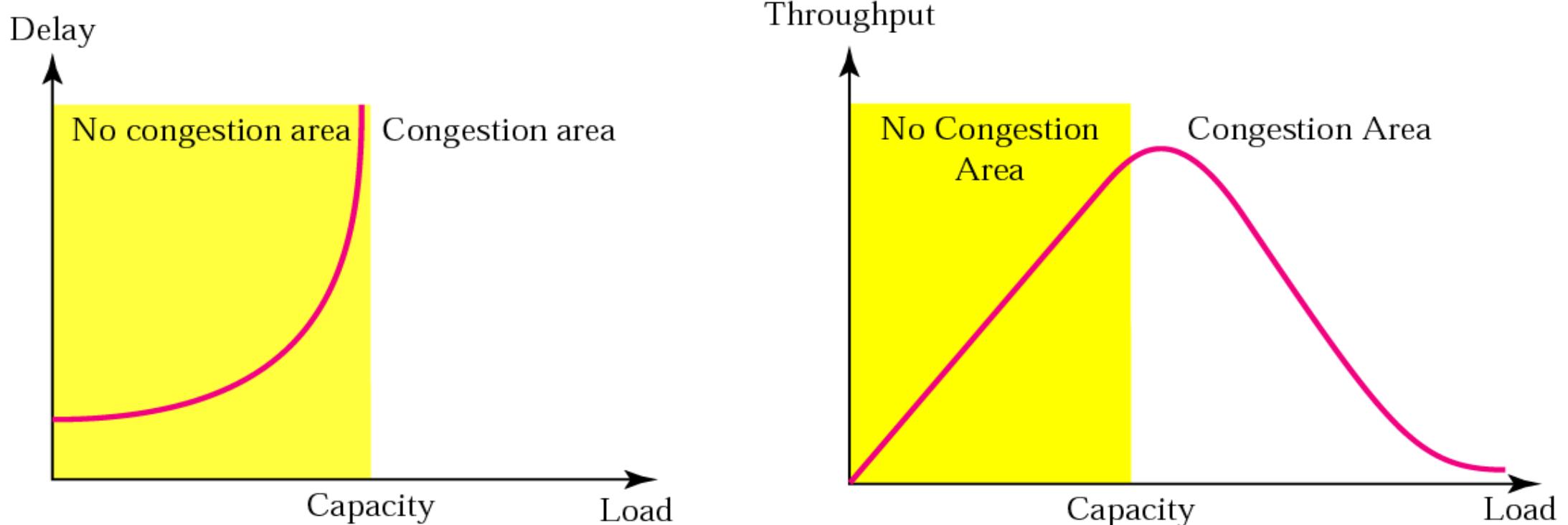
# Congestion Control at the Transport Layers

- We discuss following protocols in the subsequent slides.
  1. User Datagram Protocol (UDP)
  2. Transmission Control Protocol (TCP)

# Congestion

- An important issue in a packet-switched network is congestion.
- Congestion in a network may occur if;
  - the load on the network is greater than the *capacity* of the network.
- **Reason for Congestion:**
  - An accident during rush hour.
  - Routers and switches have queues buffers that hold the packets before and after processing.

# Congestion vs Network Performance



**Fig:** Packet delay and throughput as functions of load

# Congestion Control

- Congestion control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened.
- In other words, it refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.

# Windows method/policy of congestion control in TCP

- ***Congestion Window:***

- While discussing flow control, we said that the sender window size is determined by the available buffer space in the receiver ( $rwnd$ ).
- However, We totally ignored another entity here-the network. If the network can not deliver the data as fast as they are created by the sender, it must tell the sender to slow down.
- Today, the sender's window size is determined not only by the receiver but also by congestion in the network.
- The sender has two pieces of information: the receiver-advertised window size ( $rnwd$ ) and the congestion window size ( $cwnd$ ).
- The actual size of the window is the minimum of these two.

Actual window Size = minimum ( $rwnd, cwnd$ )

# Congestion Detection

- In TCP, congestion is detected on the occurrence of following two events:
  1. Time-out
  2. Three duplicate ACKs received
- The congestion due to time-out is more severe as compared the congestion due to three duplicate ACKs.
- The reception of three duplicate ACKs implies either network is less congested or has recovered from congestion.

# *Congestion Handling Policy*

- TCP's general policy for handling congestion is based on following three algorithms:
  1. Slow-start (SS)
  2. Congestion Avoidance (CA), and
  3. Fast Recovery (FR)
- In the slow-start phase,
  - the sender starts with a very slow rate of transmission, but increases the rate rapidly to reach a threshold.
- When the threshold is reached,
  - the data rate is reduced to avoid congestion.
- Finally if congestion is detected, the sender goes back to the slow-start or congestion avoidance phase based on how the congestion is detected.

# **1. Slow Start: Exponential Increase (SS:EI)**

- It starts with congestion window size ( $cwnd$ ) = one maximum segment size (MSS) i.e.,  $cwnd = 1 \text{ MSS}$ .
- The size of  $cwnd$  increases by one after every ACK received until it reaches a threshold, called *slow-start threshold (ssthresh)*.
- In this, the growth is exponentially after every round.

The size of the  $cwnd$  and ACK are function of each other as shown below:

- If an ACK arrives then,  $cwnd = cwnd + 1$

## Illustration of Slow Start: Exponential Growth

**Note:** Slow Start:  $cwnd$  increases by 1 MSS

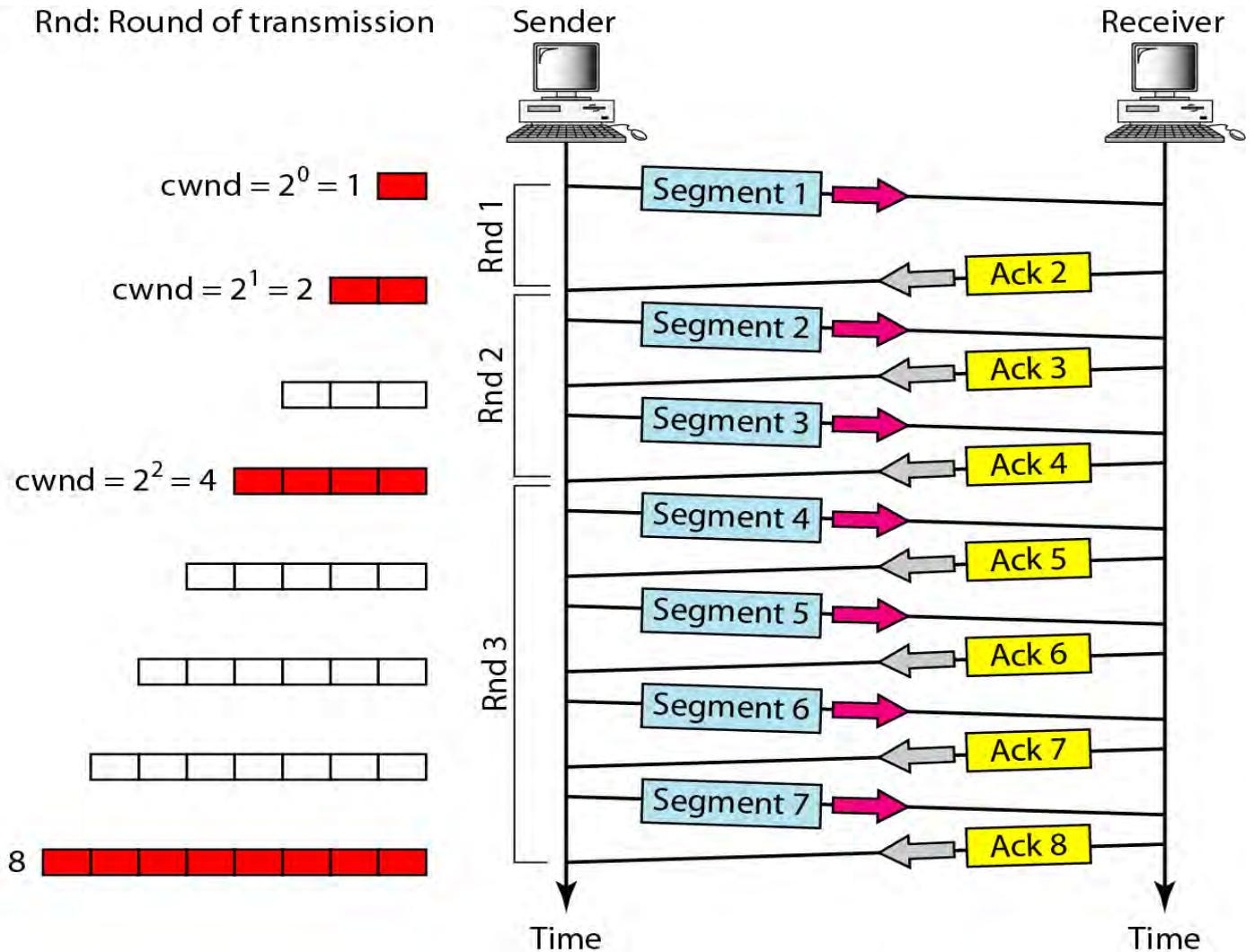
**Note:** Growth is exponential after every round, (red colour box)

Start:  $cwnd = 2^0 = 1$

After 1<sup>st</sup> round:  $cwnd = 2^1 = 2$

After 2<sup>nd</sup> round:  $cwnd = 2^2 = 4$

After 3<sup>rd</sup> round:  $cwnd = 2^3 = 8$



**Fig:** Slow-Start and Exponential growth

## **2. Congestion Avoidance: Additive Increase (CA:AI)**

- When the size of the congestion window reaches the slow-start threshold, the slow-start phase stops and the additive phase i.e., CA phase begins.
- In this, each time the whole window of segments is acknowledged (**one round**), the size of the congestion window is increased by 1.

The size of the *cwnd* and ACK are function of each other as shown below:

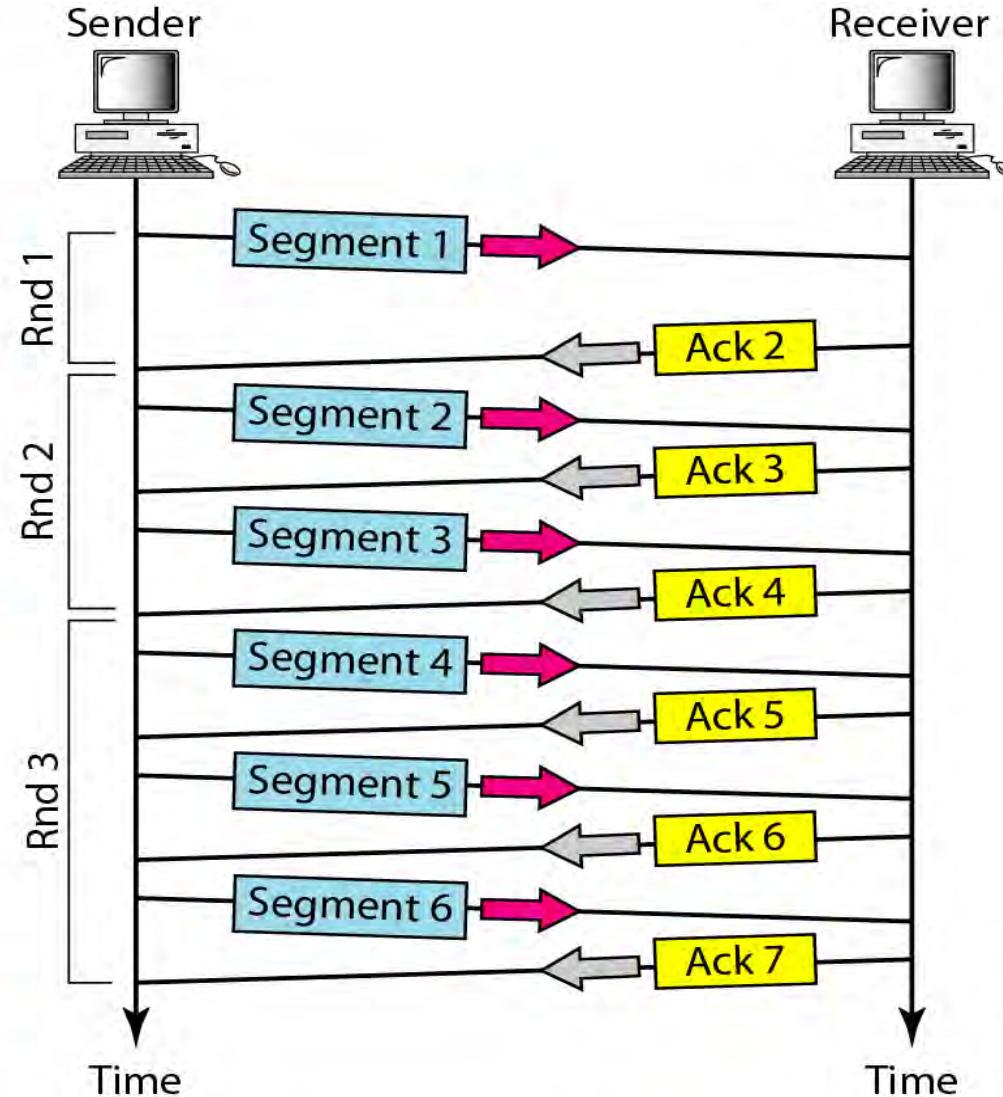
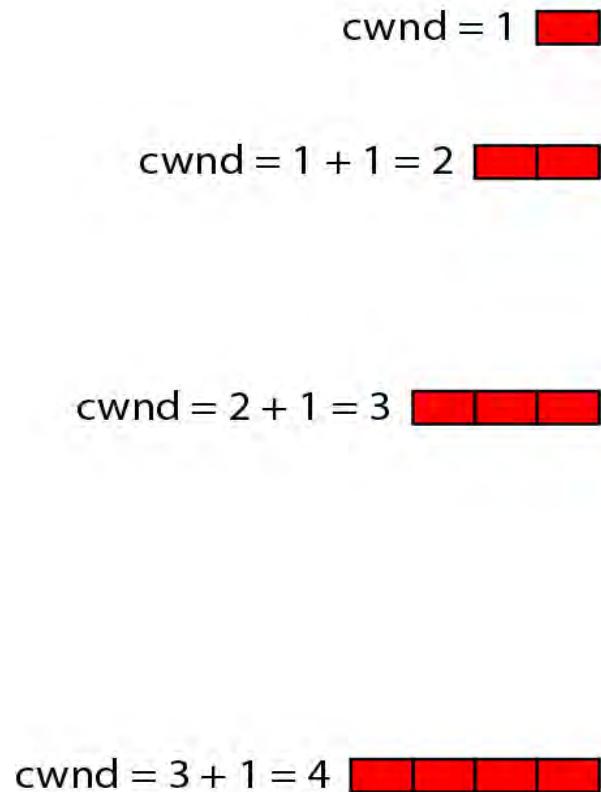
- If an ACK arrives then,  $cwnd = cwnd + \frac{1}{cwnd}$

## Illustration of Congestion Avoidance: Additive Increase

Rnd: Round of transmission

**Note:** Congestion avoidance algorithm usually starts when the size of the window is much greater than 1

Start:  $cwnd = 1$   
 After 1<sup>st</sup> round:  $cwnd = 2$   
 After 2<sup>nd</sup> round:  $cwnd = 3$   
 After 3<sup>rd</sup> round:  $cwnd = 4$



**Fig:** Congestion avoidance, additive increase

### **3. Fast Recovery (FR)**

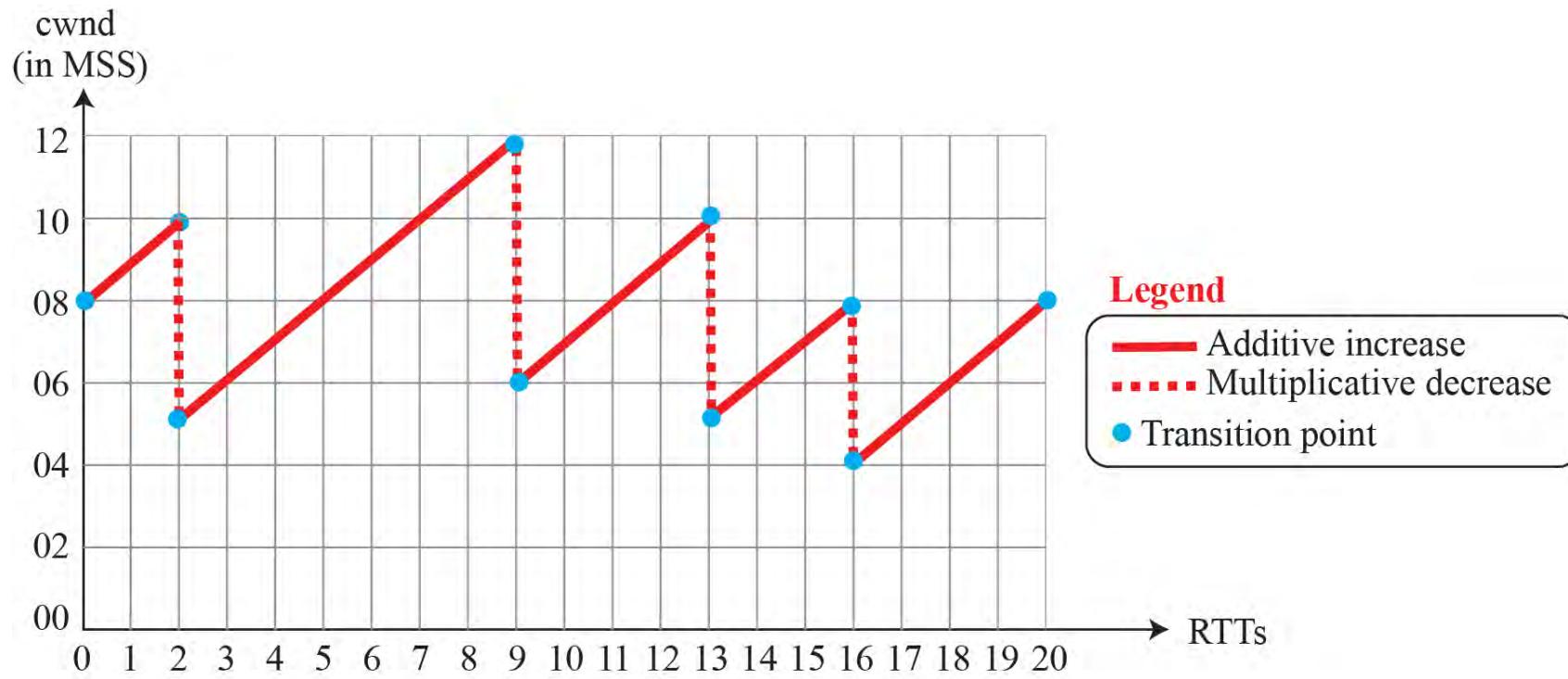
- The fast recovery algorithm is optional in TCP.
- It **starts when three duplicate ACKs arrives** → which is the interpretation of the light congestion in the network.
- It sets  $ssthresh = \frac{cwnd}{2}$ . And  $cwnd = ssthresh + 3$
- Like congestion avoidance, this algorithm is also an additive increase, but it increases the size of the congestion window when **a duplicate** ACK arrives (after three duplicate ACKs that trigger the use of this algorithm).
- When TCP is in FR state, and if
  - **Time-out occurs** → it moves to SS phase and follows the rule of SS phase
  - **A new ACK arrives** → it moves to CA phase.
  - **A duplicate ACK arrives** → it remains in FR phase

The size of the  $cwnd$  and ACK are function of each other as shown below:

- If a **duplicate** ACK arrives then,  $cwnd = cwnd + \frac{1}{cwnd}$

# Additive Increase and Multiplicative Decrease (AIMD)

- In long TCP connection, if we ignore the slow start phase, then
  - When a congestion is detected, the *cwnd* becomes **half** of the current *ssthresh*. → This is called **Multiplicative Decrease (MD)**.
  - And congestion avoidance phase starts. → this is called **Additive Increase (AI)**.
- The AIMD phase is shown in the diagram below which is observed like a **saw tooth** pattern.



- Now, the question comes?
  1. When each of these policies will be used?
  2. When TCP moves from one policy to another?

**Answers:**

- The use of different algorithm and the way to move from one policy to another is implemented differently in following three versions of TCP.
- This is also referred to as **Policy Transition** in TCP.
- Three version of TCP are:
  1. Taho TCP
  2. Reno TCP
  3. New Reno TCP

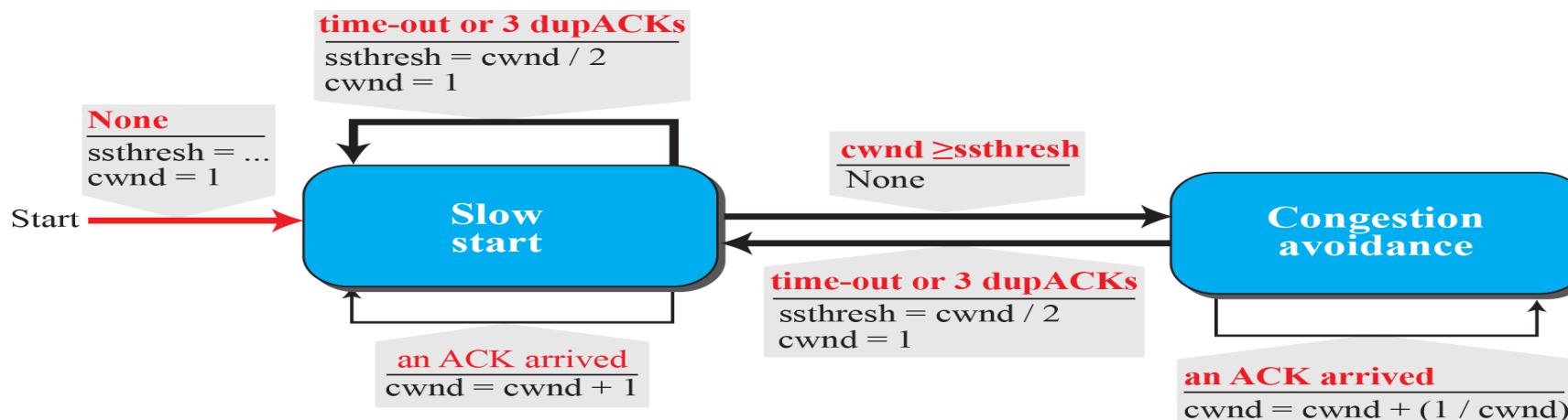


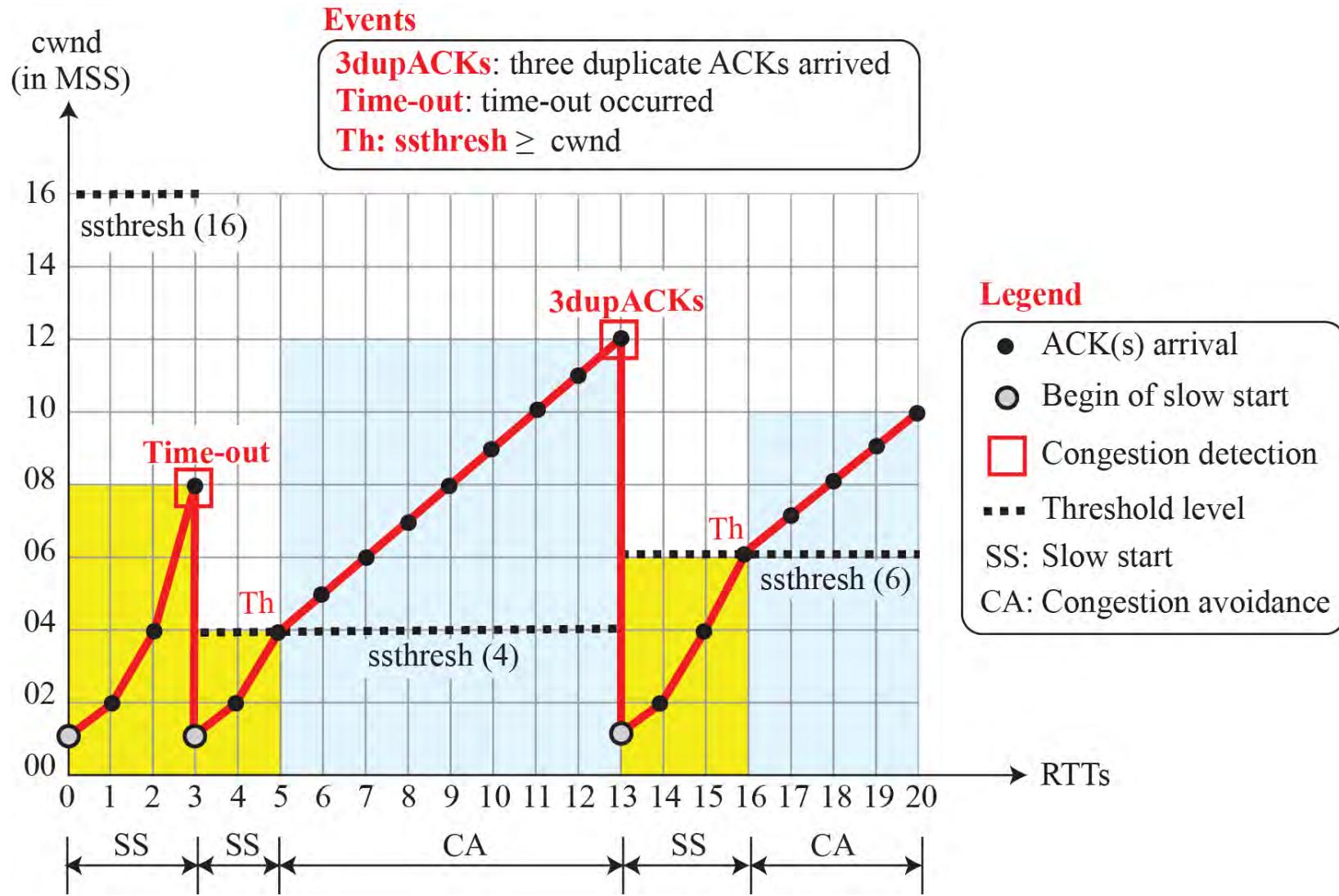
# Taho TCP

- This is the first way to implementation of congestion control mechanism in TCP.
- It uses only two algorithms (policies)
  1. *Slow-Start* and
  2. *Congestion Avoidance*
- If the congestion is detected due to
  - time-out → *Slow-Start* algorithm starts and TCP follows the rule of SS
  - three duplicate ACKs received → *Slow-Start* algorithm starts and TCP follows the rule of SS

Note:- It treats both the congestion in the same way.

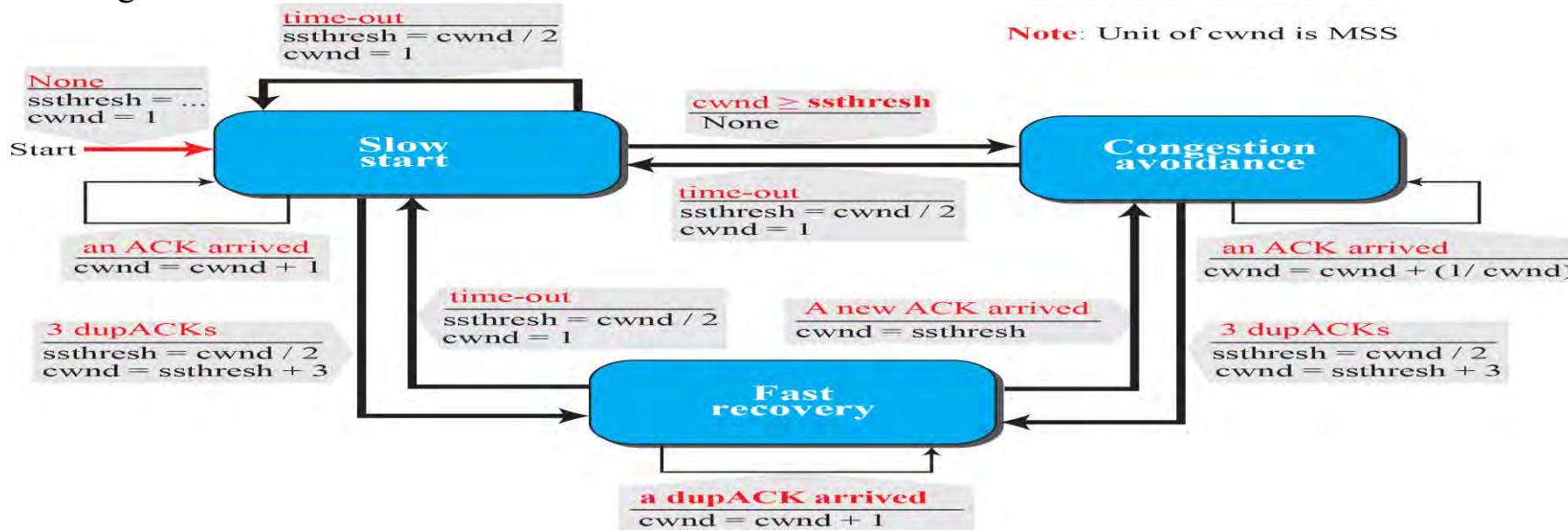
- The FSM diagram for Taho TCP is shown below.

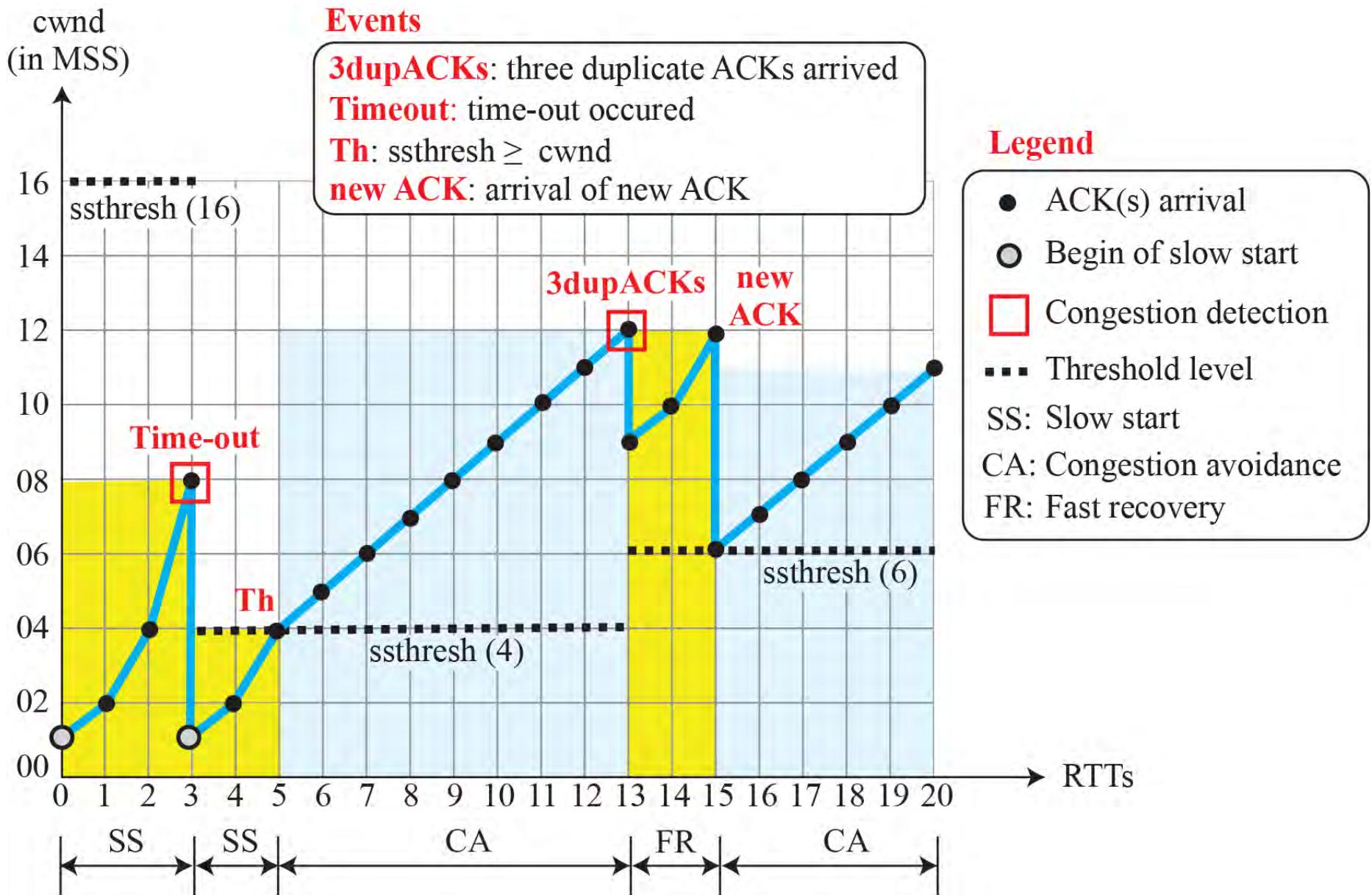




# Reno TCP

- This version of TCP handles the two signals of congestion differently.
- It uses all three algorithms (policies)
  1. *Slow-Start*
  2. *Congestion Avoidance*
  3. *Fast Recovery*
- If the congestion is detected due to
  - time-out → *Slow-Start* algorithm starts and TCP follows the rule of SS phase
  - three duplicate ACKs received → *Fast Recovery* algorithm starts and TCP follows the rule of FR phase
- The FSM diagram for Reno TCP is shown below.





# ***New Reno TCP***

- This version made an extra optimization of existing Reno TCP. For more detail, refer Page no -221 of the text book.

To understand Slow start, CA, and Congestion Detection and its effect through the graph..

- Imagine the example covered in the class of **Mummy** and **You..**



**Disclaimer:-** The images are taken from the Google for explaining the concept to students. Its intention is not to hurt sentiments of any students or person portrayed in this.

## Summary Table for different algorithms

Condition	Size of the <i>ssthresh</i> when the corresponding phase starts	Size of the <i>cwnd</i> when the corresponding phase starts	Growth of <i>Cwnd</i>
Slow Start	$ssthres = \frac{cwnd}{2}$	$cwnd = 1$	$cwnd = cwnd + 1$ When an ACK arrives
Congestion Avoidance	$ssthres = \frac{cwnd}{2}$	$cwnd = ssthresh$	$cwnd = cwnd + \frac{1}{cwnd}$ When an ACK arrives
Fast Recovery	$ssthres = \frac{cwnd}{2}$	$cwnd = ssthresh + 3$	$cwnd = cwnd + \frac{1}{cwnd}$ When a Duplicate ACK arrives

## Summary Table for different policy transition methods

Policy transition Method	Reason of Congestion	Algorithm to be Adopted
Taho TCP	Time – Out	SS
	Three Duplicate ACK	SS
Reno TCP	Time – Out	SS
	Three Duplicate ACK	Fast Recovery

# Homework

- 1) Explain the window methods of congestion control implementation in Tahoe TCP with the help of graph plotted for '*rounds*' on x axis against '*cwnd*' on y axis. Assume, the maximum window size is 32 segments. Show the regions like Slow-Start (SS), Additive Increase (AI), Multiplicative Decrease (MD) explicitly in the graph for the following situations.
  - A congestion is detected due to time out when *cwnd* reaches to 20.
  - Further, the congestion is detected due to reception of three ACKs when *cwnd* reaches to 12.
- 2) Repeat above homework using Reno TCP.

## **End of Module-3**

You are advised to go through the chapter 3 of the text book as mentioned in the class for more clarity and details.

# CN (IT-3001)

## Data Link Layer: MAC Protocol

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



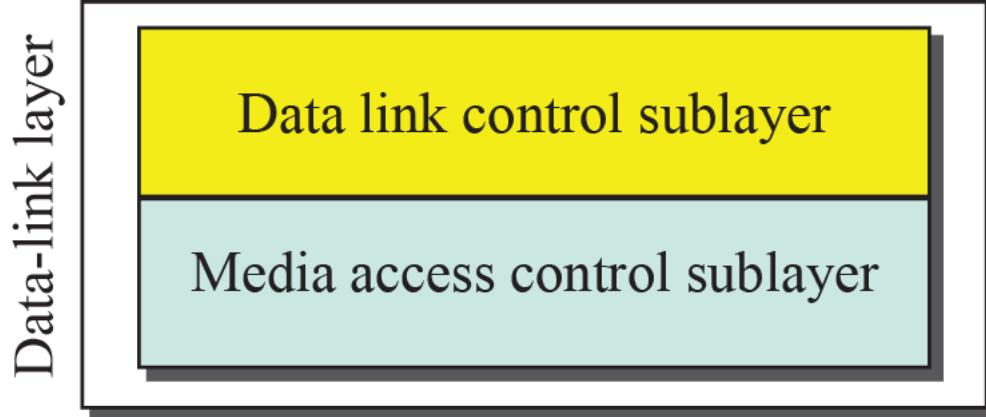
**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose. Images portrayed (if any) are not to hurt the sentiments of any person.

# Objective

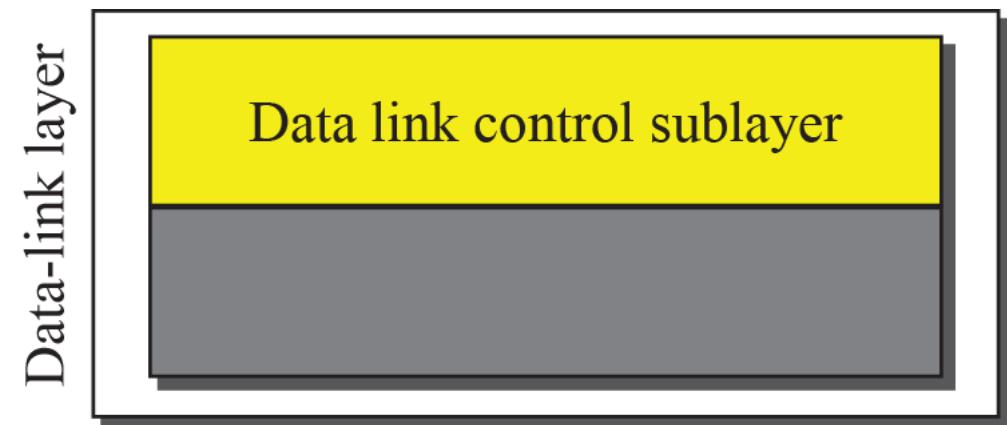
- Media Access/Multiple Access
  - Random Access
    - ALOHA
    - CSMA
    - CSMA/CD
    - CSMA/CA
  - Controlled Access
    - Reservation
    - Polling
    - Token passing
  - Channelization
    - FDMA
    - TDMA
    - CDMA

# Two Sublayers of the Data-Link-Layer

- The data link layer is divided into two sublayers as shown below.
  1. Data Link Control (DLC) sublayer
  2. Media Access Control (MAC) Sublayer



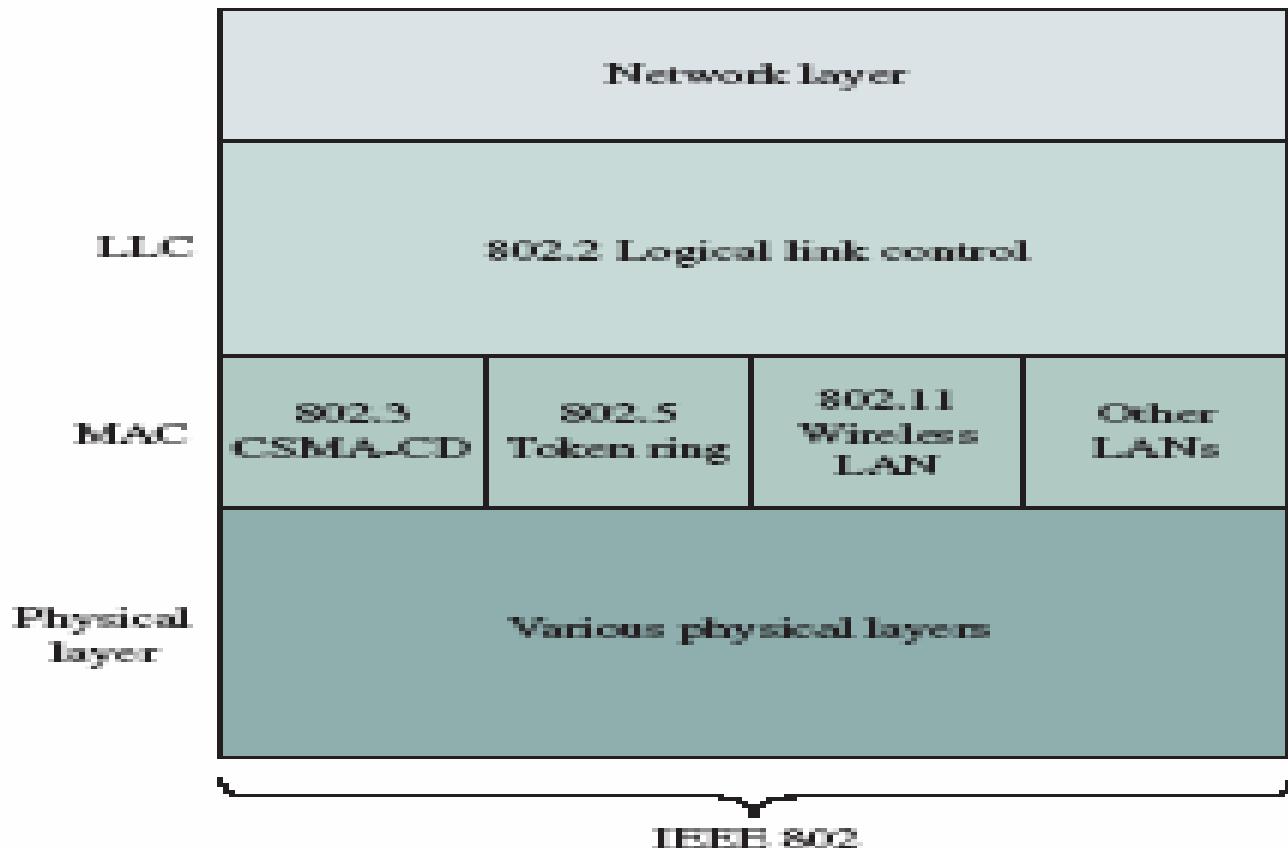
a. Data-link layer of a broadcast link



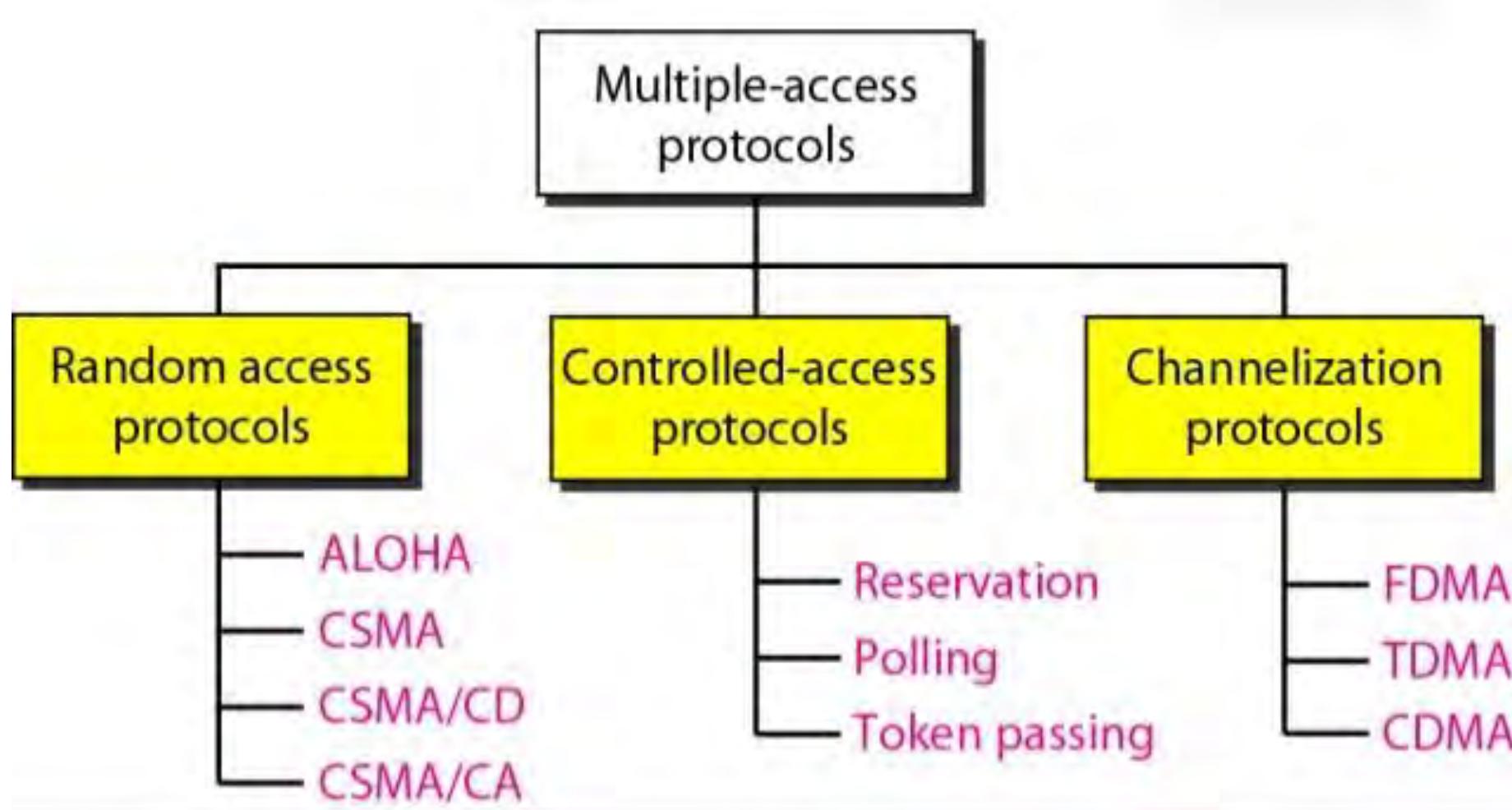
b. Data-link layer of a point-to-point link

- The upper sublayer that is responsible for flow and error control is called the *logical link control* (LLC) layer.
- The lower sublayer that is mostly responsible for multiple access resolution is called the *media access control* (MAC) layer.
- **Why do we need multiple-access protocol?**

**Ans:** In a broadcast or multipoint, nodes use a common link. To use this common link efficiently, we need a multiple-access protocol to coordinate access to the link.



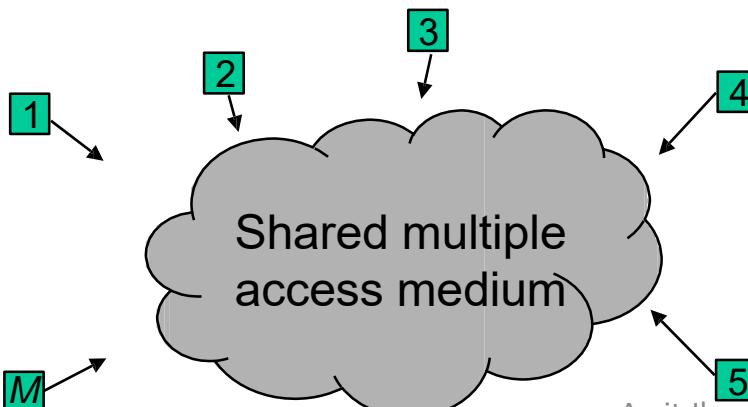
# Types of Multiple-Access Protocol/ MAC Protocol



# Random Access Protocol

- Why the name *random access*?
  - There is no scheduled time for a station to transmit.
  - Transmission is random among the stations.
  - No station is superior to another station and none is assigned the controlled over another.
- In a random access method, each station has the right to the medium without being controlled by any other station. However, if more than one station tries to send, there is an access conflict-collision-and the frames will be either destroyed or modified.

- To avoid access conflict or to resolve it when it happens, each station follows a procedure that answers the following questions:
  - When can the station access the medium?
  - What can the station do if the medium is busy?
  - How can the station determine the success or failure of the transmission?
  - What can the station do if there is an access conflict?

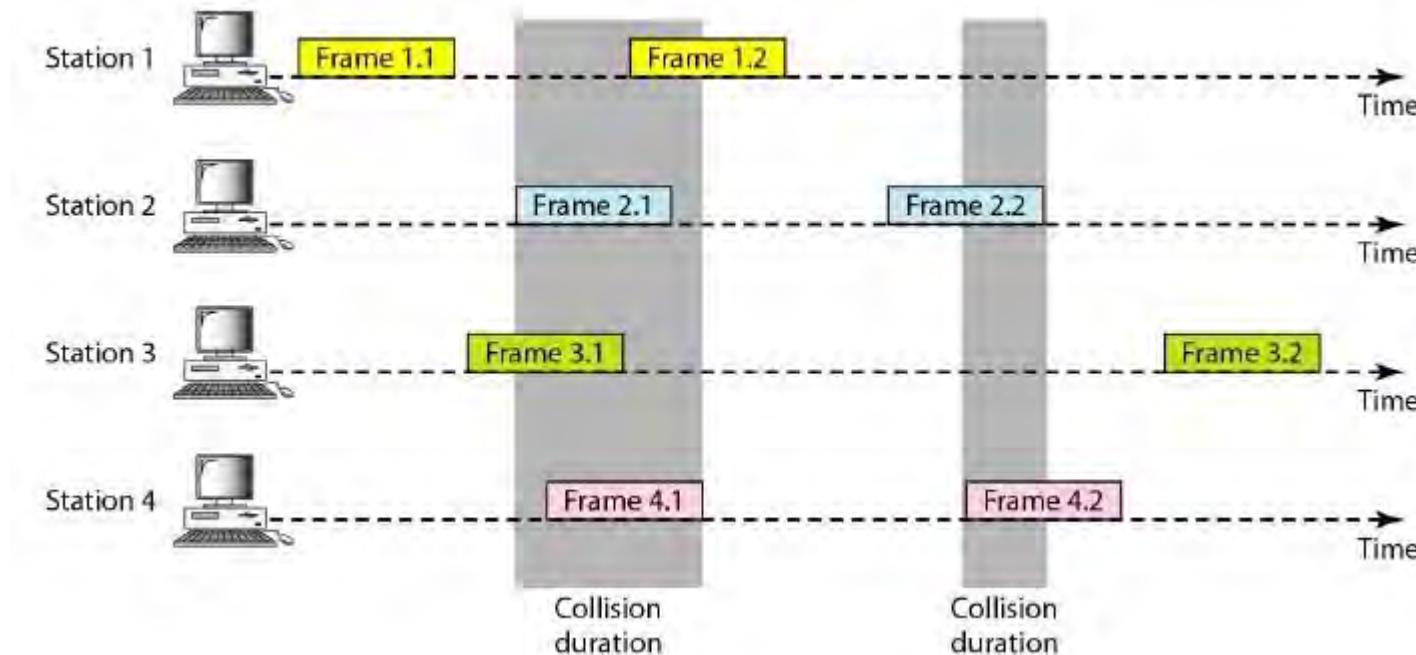


# ALOHA: Pure ALOHA

- Based upon the simplest solution: **just do it**
  - A station transmits whenever it has data to transmit.
  - If more than one frames are transmitted, they interfere with each other (collide) and are lost.
  - If ACK not received within timeout, then a station picks **random back-off time** (to avoid repeated collision).
  - Station retransmits frame after back-off time denoted as  $T_B$
- **Note:** A collision involves two or more stations. If all these stations try to resend their frames after the time-out, the frames will collide again.
- After a maximum number of retransmission attempts  $K_{max}$  a station must give up and try later.

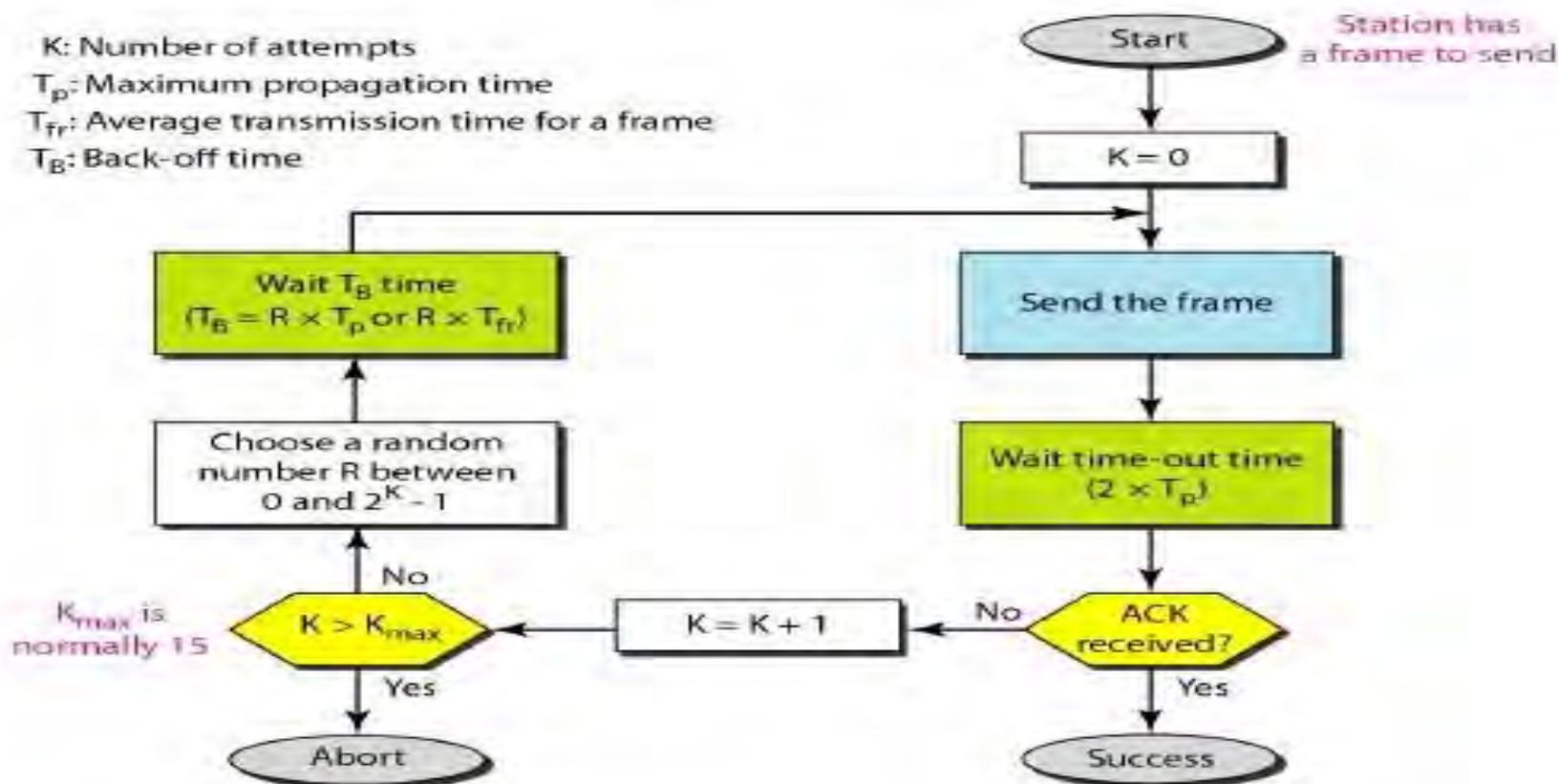
*Four stations transmitting 2 frames each.*

*Out of all the frames, only two frames survive: frame 1.1 and frame 3.2*



**Fig.** Example of frame collisions in pure ALOHA

# Procedure for pure ALOHA protocol



**Note:** R is a random number chosen from the range 0 to  $2^k - 1$ , and value of the random number increases after each collision.

**Example 1:** The stations on a wireless ALOHA network are a maximum of 600 km apart. If we assume that signals propagate at  $3 \times 10^8 \text{ m/s}$ , we find  $T_P = (600 \times 10^3) / (3 \times 10^8) = 2 \text{ ms}$ . Now we can find the value of  $T_B$  for different values of  $K$ .

- a) For  $K=1$ , the range of  $R$  is  $\{0, 1\}$ . The station needs to generate a random number with value 0 or 1. So,  $T_B$  is either 0 or 2ms, based on outcome of the random variable.
- b) For  $K=2$ , the range of  $R$  is  $\{0, 1, 2, 3\}$ . So,  $T_B$  can be 0, 2, 4 or 6ms, based on outcome of the random variable.
- c) For  $K=3$ , the range of  $R$  is  $\{0, 1, 2, 3, \dots, 7\}$ . So,  $T_B$  is can be 0, 2, 4, 6, 8, 10, 12, or 14ms, based on outcome of the random variable.
- d) So on.....
- e) We need to mention that if  $k > 10$ , it is normally set to 10.

**Vulnerable time:** It is the time duration , in which there is a possibility of collision. **Vulnerable time in pure ALOHA =  $2 \times T_{fr}$**

$T_{fr}$  = frame transmission time

B already sent a frame

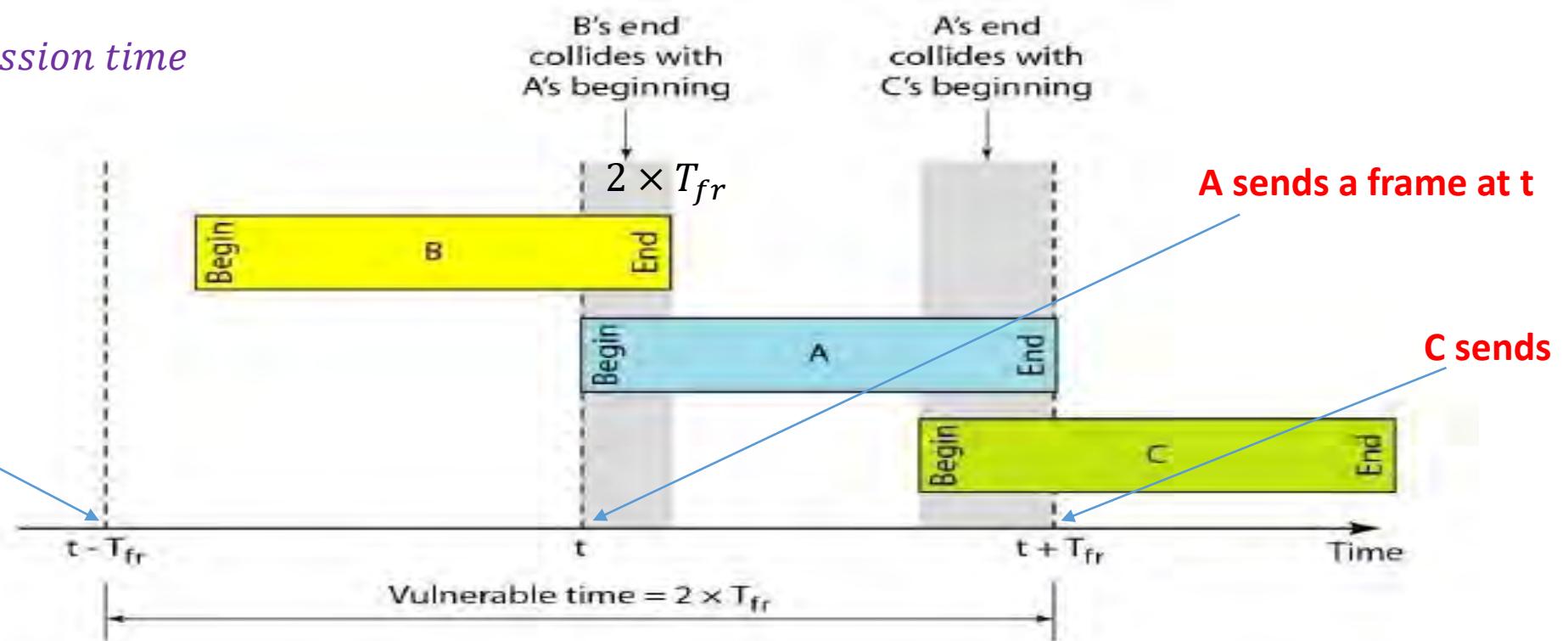


Fig: Vulnerable time for pure ALOHA

**Example 2:** A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the requirement to make this frame collision-free?

**Sol:**

*Average frame transmission time  $T_{fr}$  is 200 bits/200 kbps or 1 ms. The vulnerable time is  $2 \times 1 \text{ ms} = 2 \text{ ms}$ .*

*This means no station should send later than 1 ms before this station starts transmission and no station should start sending during the one 1-ms period that this station is sending.*

# Pure ALOHA Model

- Definitions and assumptions
  - $T_{fr}$  frame transmission time (assume constant)
  - $S$ : throughput (average # successful frame transmissions per  $T_{fr}$  seconds)
  - $G$ : load (average # transmission attempts per  $T_{fr}$  sec.)
  - $P_{success}$  : probability a frame transmission is successful

**Note:** Any transmission that begins during vulnerable period leads to collision. Success if and only if no arrivals during  $2 T_{fr}$  seconds.

Throughput is given by,

$$S = GP_{success}$$

## Abramson's assumption for calculation of $P_{Success}$

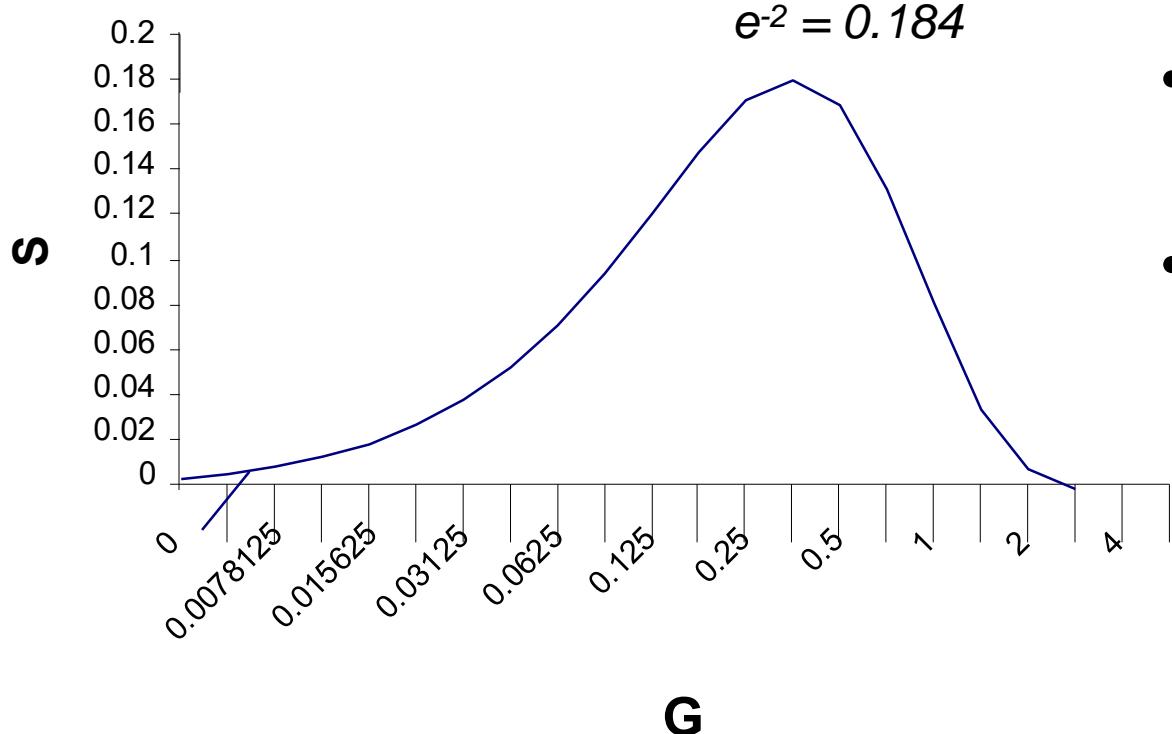
- *What is probability of no arrivals in vulnerable period?*
- **Abramson's assumption:** Effect of back-off algorithm is that frame arrivals are equally likely to occur at any time interval.
- $G$  is avg. # arrivals per  $T_{fr}$  seconds
- Divide  $T_{fr}$  into  $n$  intervals of duration  $\Delta = T_{fr}/n$
- $p$  = probability of arrival in  $\Delta$  interval, then

$$G = n p \quad \text{since there are } n \text{ intervals in } T_{fr} \text{ seconds}$$

$$\begin{aligned} P_{success} &= P[0 \text{ arrivals in } 2T_{fr} \text{ seconds}] = \\ &= P[0 \text{ arrivals in } 2n \text{ intervals}] \quad \dots \dots \dots \text{Abramson's assumption:} \\ &= (1 - p)^{2n} = \left(1 - \frac{G}{n}\right)^{2n} \rightarrow e^{-2G} \quad \text{as } n \rightarrow \infty \end{aligned}$$

# Throughput of ALOHA

$$S = GP_{success} = Ge^{-2G}$$



- Collisions are means for coordinating access
- Max throughput is  $\rho_{max} = 1/2e (18.4\%)$
- Bimodal behavior:
  - Small  $G$ ,  $S \approx G$
  - Large  $G$ ,  $S \downarrow 0$

Use basic maths

**Example 3:** A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces

- a. 1000 frames per second
- b. 500 frames per second
- c. 250 frames per second

**Sol:** Here,  $T_{fr}$  is 200 bits/200 kbps or 1 ms.

**a.** If the system creates 1000 frames per second, this is 1 frame per millisecond. The load is 1. In this case  $S = G \times e^{-2G}$  or  $S = 0.135$  (13.5 percent). This means that the throughput is  $1000 \times 0.135 = 135$  frames. Only 135 frames out of 1000 will probably survive.

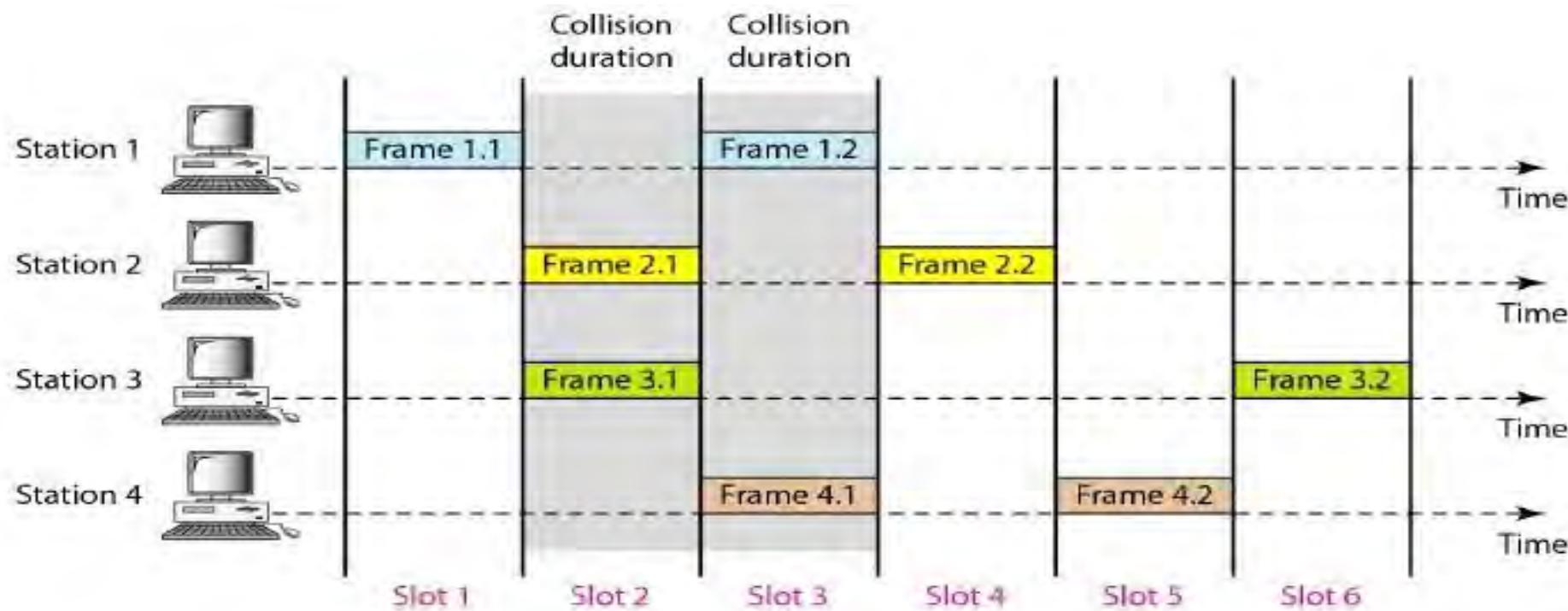
**b.** If the system creates 500 frames per second, this is  $(1/2)$  frame per millisecond. The load is  $(1/2)$ . In this case  $S = G \times e^{-2G}$  or  $S = 0.184$  (18.4 percent). This means that the throughput is  $500 \times 0.184 = 92$  and that only 92 frames out of 500 will probably survive.

Note that this is the maximum throughput case, percentagewise.

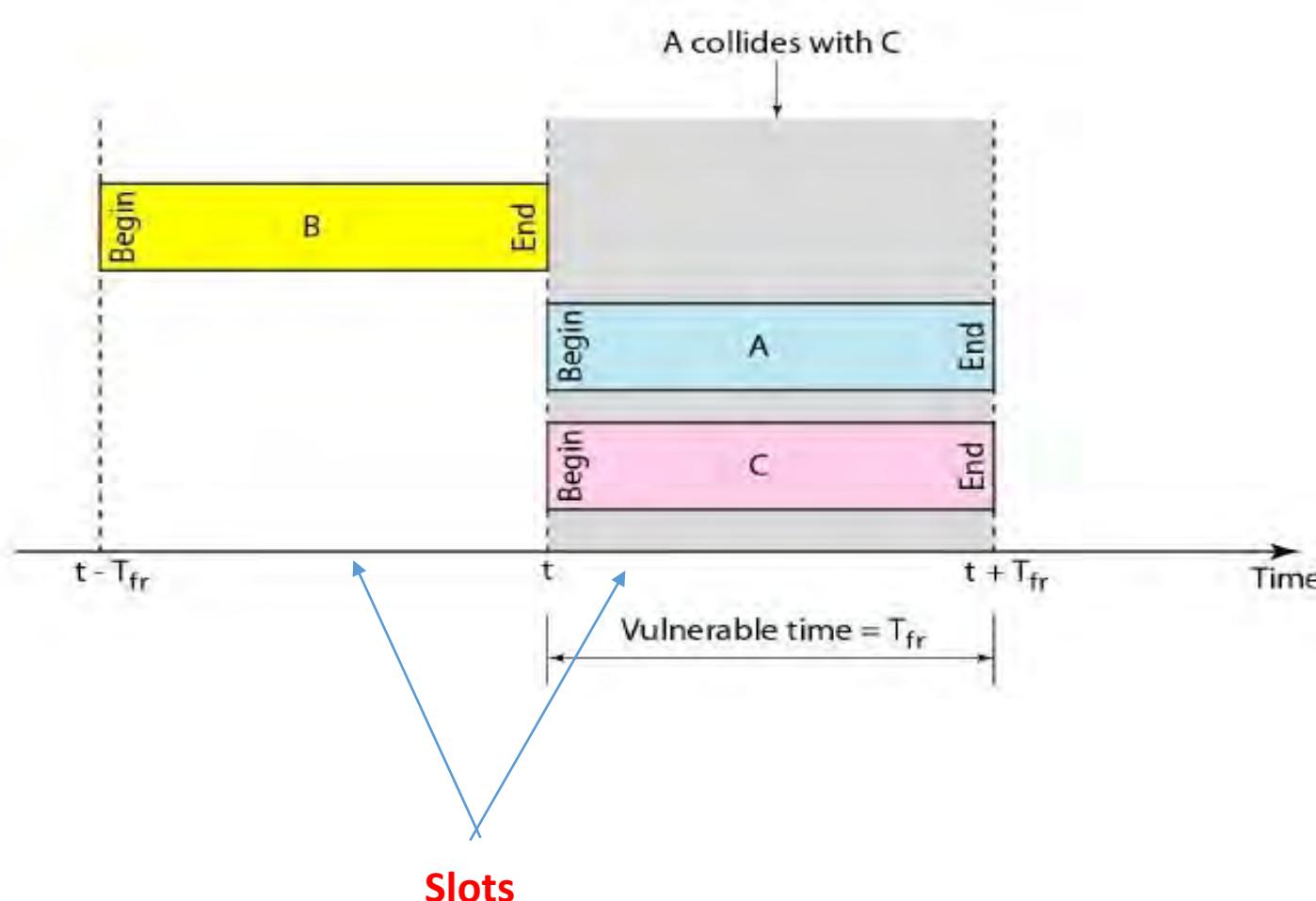
**c.** If the system creates 250 frames per second, this is  $(1/4)$  frame per millisecond. The load is  $(1/4)$ . In this case  $S = G \times e^{-2G}$  or  $S = 0.152$  (15.2 percent). This means that the throughput is  $250 \times 0.152 = 38$ . Only 38 frames out of 250 will probably survive.

# Slotted ALOHA

Time is slotted in  $T_{fr}$  seconds slots  
Stations synchronized to frame times  
Stations transmit frames in first slot after frame arrival  
Backoff intervals are in multiples of slots



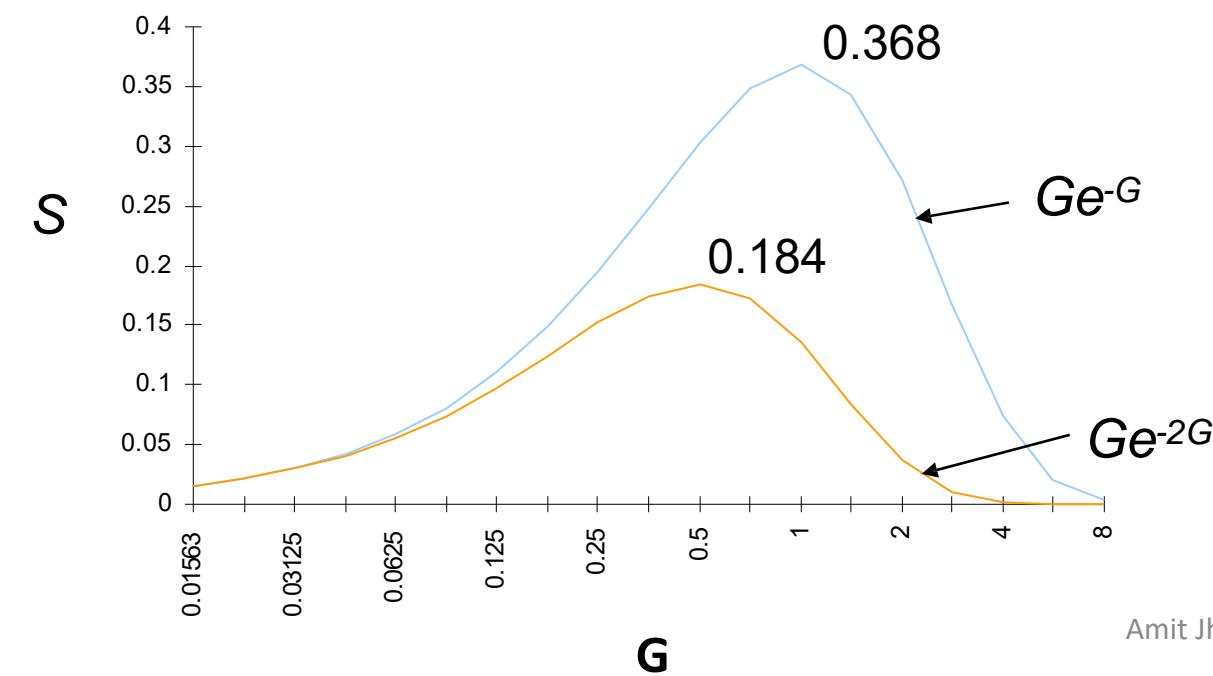
# Vulnerable time for slotted aloha



# Throughput of Slotted ALOHA

$$\begin{aligned} P_{success} &= P[0 \text{ arrivals in } T_{fr} \text{ seconds}] \\ &= P[0 \text{ arrivals in } n \text{ intervals}] \quad \dots \text{Abramson's assumption} \\ &= (1 - P)^n = \left(1 - \frac{G}{n}\right)^n \\ &\rightarrow e^{-G} \quad \dots \dots \dots \text{as } n \rightarrow \infty \end{aligned}$$

$$\therefore S = GP_{Success} = Ge^{-G}$$



## Limitations of ALOHA

The throughput for pure ALOHA is  $S = G \times e - 2G$ .

The maximum throughput  $S_{max} = 0.184$  when  $G = (1/2)$ .

The throughput for slotted ALOHA is  $S = G \times e - G$ .

The maximum throughput  $S_{max} = 0.368$  when  $G = 1$ .

**Homework:** 1) Repeat Example 3 for slotted ALOHA, and observe the conclusion.  
2) Derive the formulae for the maximum throughput for Pure and Slotted ALOHA.

# CN (IT-3001)

## Network Layer: Introduction & basics of IPv4 Datagram

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University

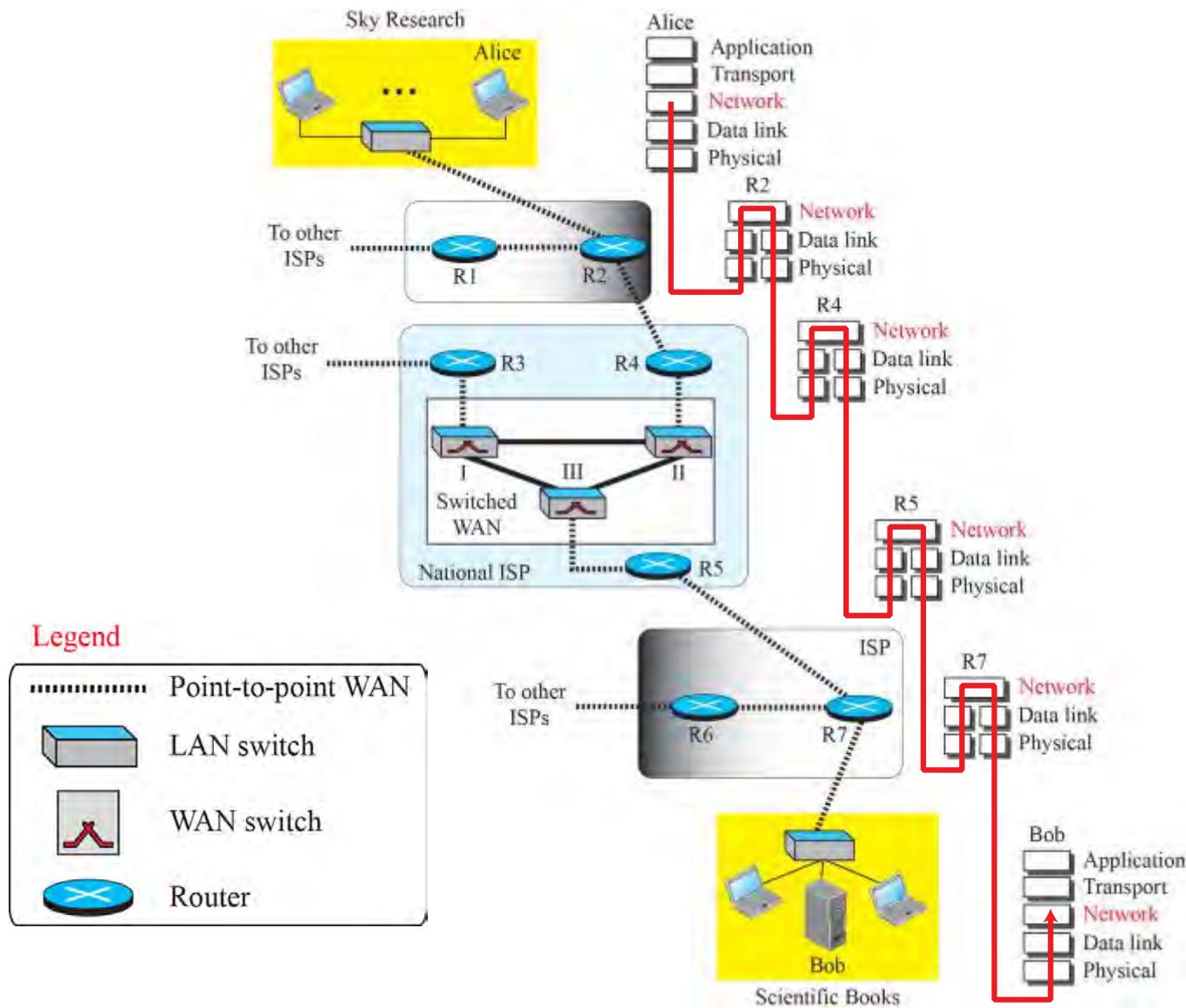


**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.

# Objectives

- The objective of this module is to discuss following concepts...
  1. Position of Network Layer
  2. Services and Responsibilities of Network Layer
  3. Performance parameter at the Network Layer
  4. Network Layer Protocol
    1. IPv4 Datagram Format
  5. Fragmentation

# Communication at the Network Layer

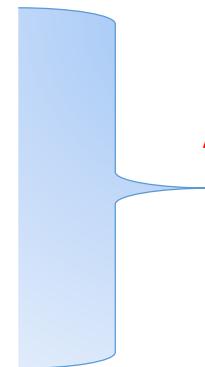


# Responsibilities of the Network Layer

- **Packetizing:** the first responsibility of the network layer is encapsulating the payload in a network-layer packet at the source and decapsulating the payload from the network layer packet at the destination.
- **Addressing:** It adds a header that includes the logical addresses of source and destination to the packet coming from the upper layer.
- **Routing:** The most vital responsibility of the network layer is routing mechanism.
- **Delivery:** The network layer is responsible for the delivery of individual packets from the source to the destination host.

# Network Layer Performance

- The performance of the network can be measured in terms of *delay*, *throughput*, and *packet loss*.
- **Delay** includes:
  1. Transmission Delay
  2. Propagation Delay
  3. Processing Delay
  4. Queuing Delay
- **Throughput**
- **Packet Loss**

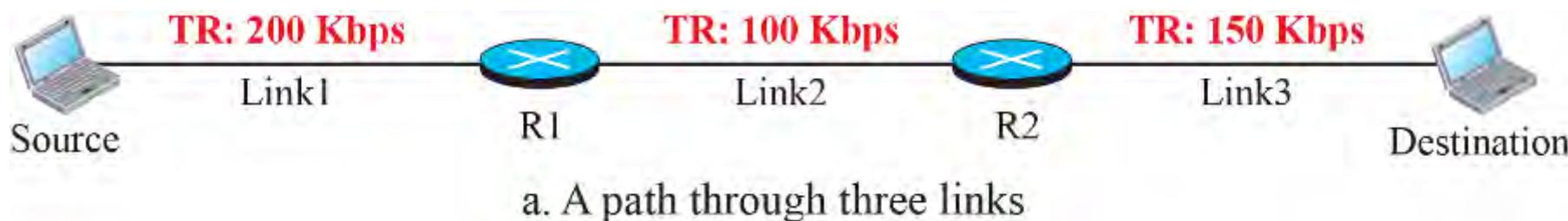


All these are already discussed in Module-1

- If we assume equal delays for the sender, routers, and receiver, then total delay (source-to-destination) a packet encounters can be calculated if we know the number of routers,  $n$ , in the whole path as follows.

$$\text{Total Delay} = (n + 1)\{T_{tr} + T_{prop} + T_{proc}\} + (n)\{T_{queu}\}$$

- Throughput:** Throughput at any point in the network is defined as the number of bits passing through the point in a second successfully, which is actually the transmission rate of the data at that point.



**Example 4.1:** Calculate the throughput for the diagram shown above.

Answer- 100 Kbps

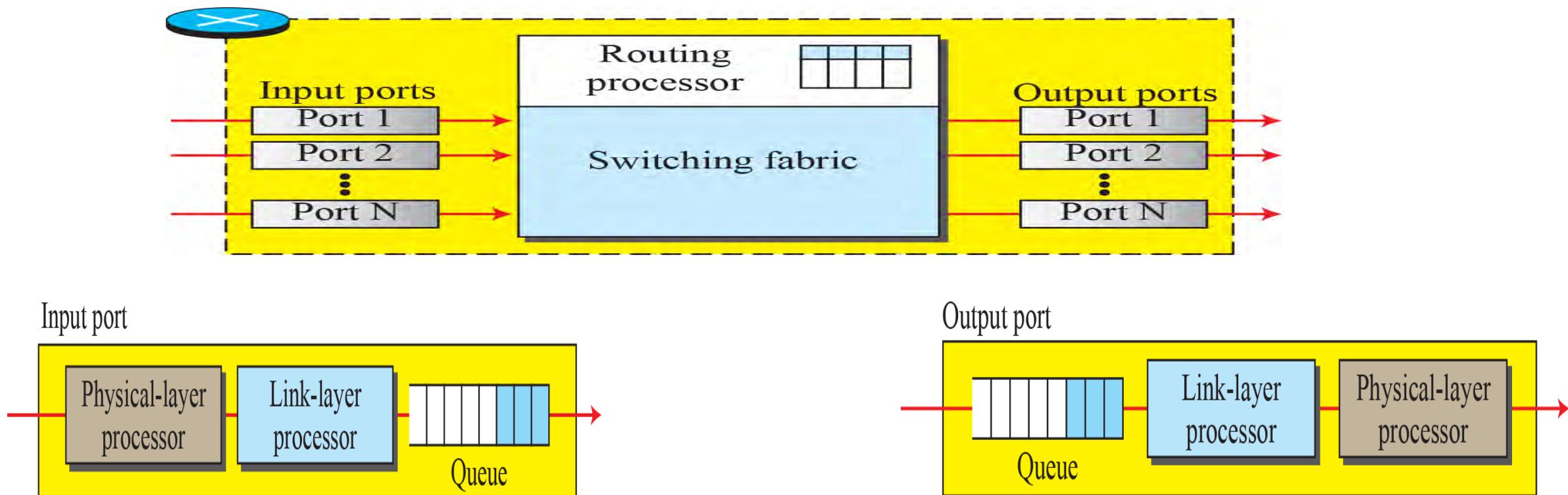
**Explanation:-**  $\text{Throughput} = \min\{TR_1, TR_2, \dots, TR_n\}$



- **Packet Loss:** When a router receives a packet while processing another packet, the received packet needs to be stored in the input buffer waiting for its turn. A router, however, has an input buffer with a limited size. A time may come when the buffer is full and the next packet needs to be dropped. This results in packet loss. The effect of packet loss on the Internet network layer is that the packet needs to be resent, which in turn may create overflow and cause more packet loss.

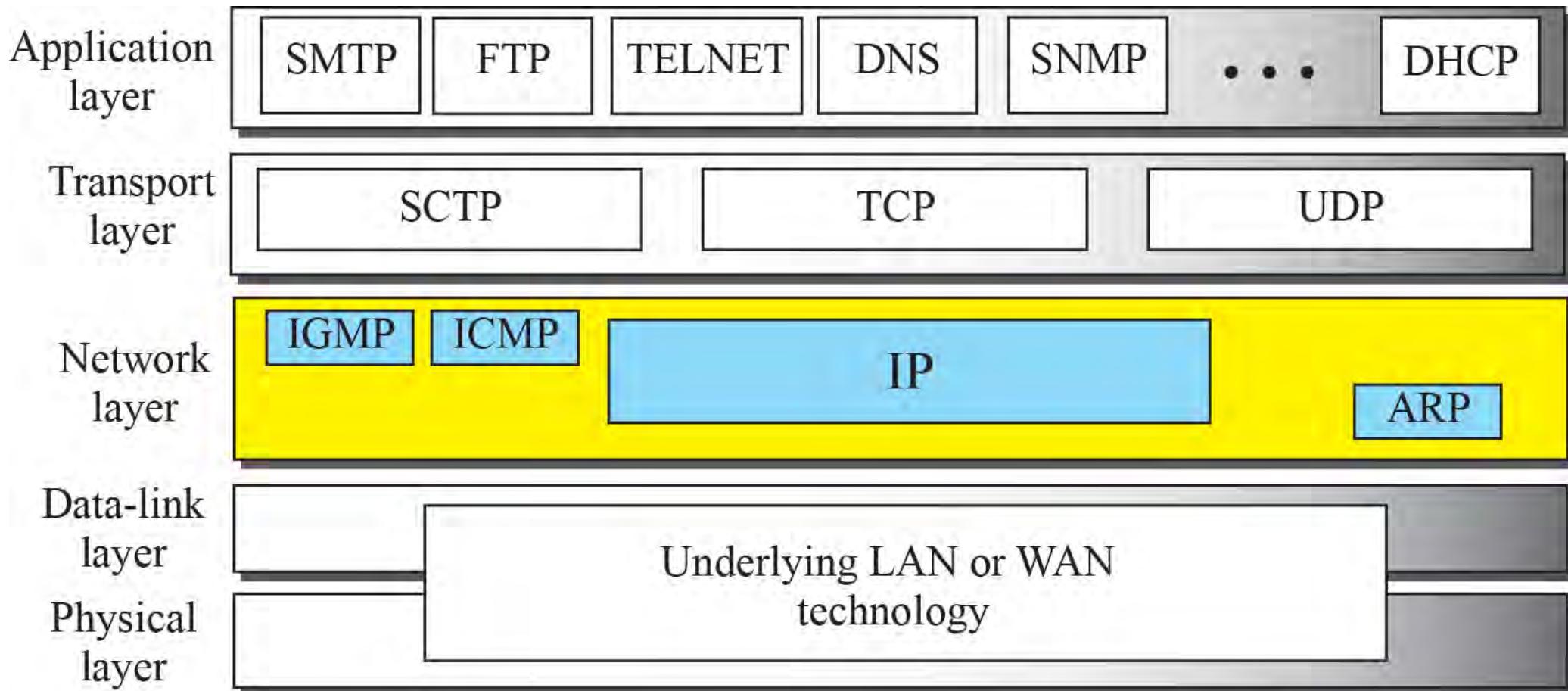
# Structure of a Router

- The overview of a router is shown below briefly.



- **Input port:**
  - An input port performs the physical and link-layer functions of the router.
  - The bits are constructed from the received signal. The bits are decapsulated from the frame, checked for errors, and discarded if corrupted.
  - The packet is then ready to be processed by the network layer.
  - In addition to a physical-layer processor and a link-layer processor, the input port has buffers (queues) to hold the packets before they are directed to the switching fabric.
- **Output Port:**
  - An output port performs the same functions as the input port, but in the reverse order.
  - First the outgoing packets are queued, then each packet is encapsulated in a frame, and finally the physical layer function is applied to the frame to create the signals to be sent on the line.
- **Routing Processor:**
  - Routing processor searches the forwarding table.
  - Routing processor uses the destination address to find the address of the next hop and, at the same time, the output port number from which the packet is sent out.
- **Switching Fabrics:**
  - Switching fabrics is used to move the packets from the input queue to the output queue.
  - Some of the switching fabrics include:
    - Crossbar Switch
    - Banyan Switch
    - Batcher-Banyan Switch

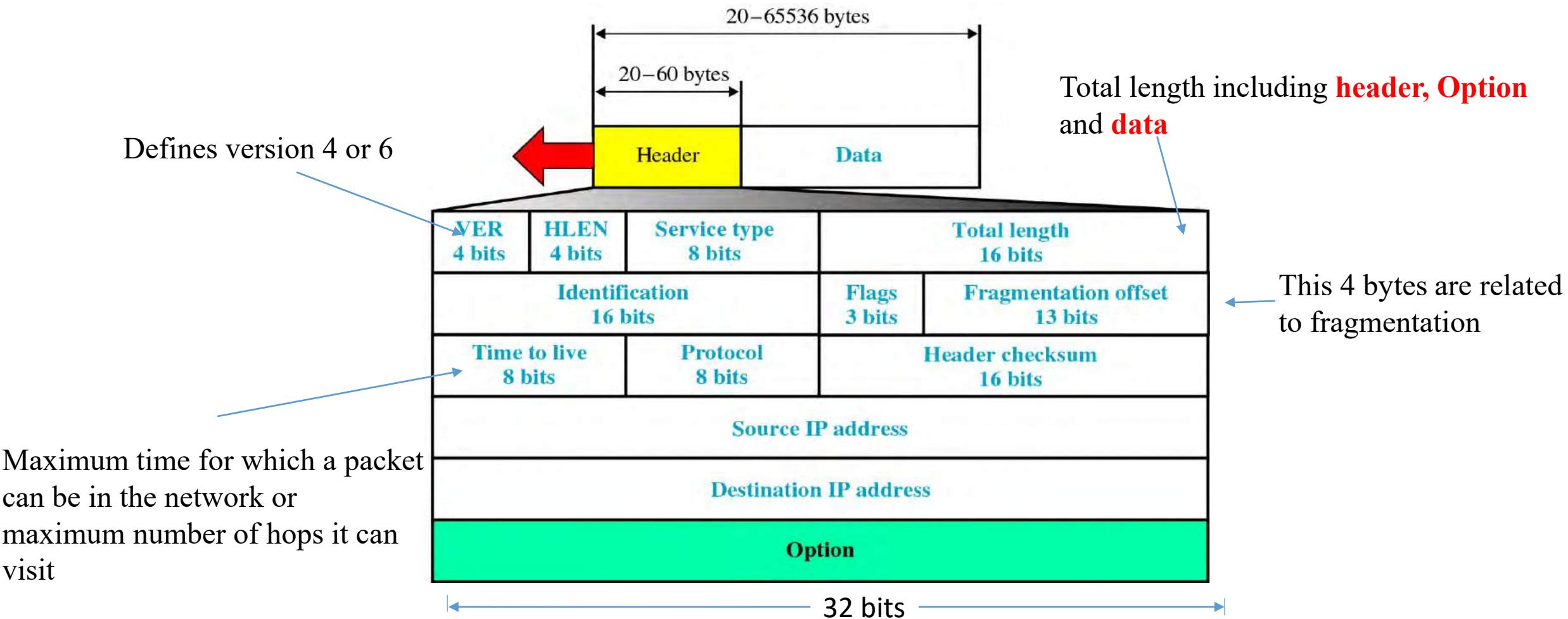
# Network Layer Protocols



# Internet as a Datagram Network

- Out of three switching techniques: **circuit, message and packet switching**; the Internet has chosen packet switching.
- Out of two available approaches: **virtual circuit and datagram**; switching at the network layer in the Internet uses the **datagram approach** to packet switching.
- Thus, communication at the network layer in the Internet is connectionless.
- Packets at the IP layer are called ***datagrams***.

# IPv4 Datagram Format



**Note:** It is customary in TCP/IP to show the header in 4-byte sections.

- **HLEN:**
  - It defines the total length of the datagram header in 4-bytewords.
  - When there are no options, the header length is 20 bytes, and the value of this field is 5 ( $5 \times 4=20$ ). When the option field is at its maximum size, the value of this field is 15 ( $15 \times 4=60$ ).
- **Service Type:** In this interpretation, the first 3-bits are called precedence bits. The next 4-bits are called type of service (TOS) bits, and the last bit is not used.
  - a. **Precedence:** It is a 3-bit sub field ranging from 0 to 7. It defines the priority of the datagram in issues such as congestion. If a router is congested and needs to discard some datagrams, those datagrams with lowest precedence are discarded first.
  - b. **TOS:** Only one bit can be set at a time.

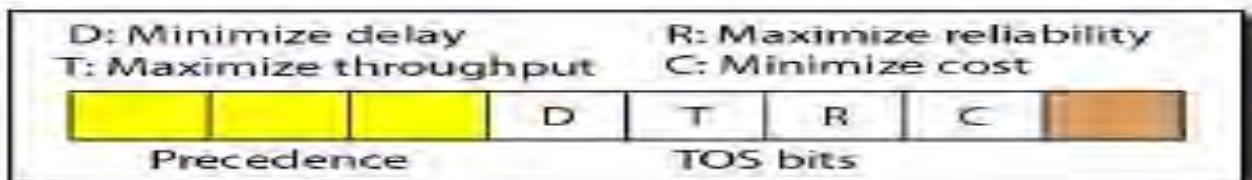


Fig: Service type

<i>TOS Bits</i>	<i>Description</i>
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

- **TTL:** Maximum time for which a packet can be in the network or maximum number of hops it can visit. A router which processes a datagram decrements the value of TTL field by 1.
- **Protocol:** This 8-bit field defines the higher-level protocol that uses the services of the IP layer.

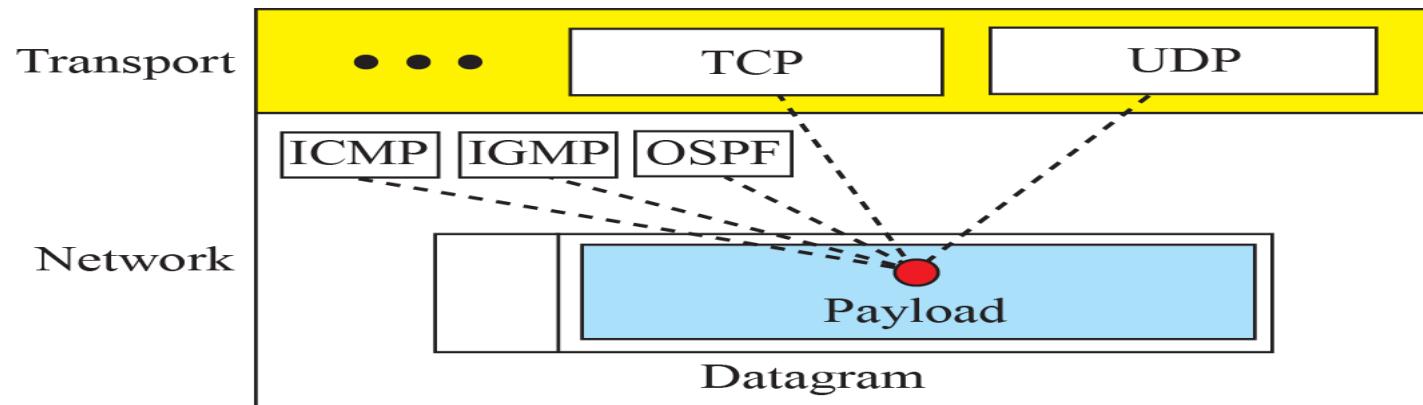
**Question:** After processing a packet, if TTL becomes zero, then comment whether this packet will be sent to the next router or not?

**Question:** Can a router receive a packet with TTL=0?



ICMP: 01	UDP: 17
IGMP: 02	OSPF: 89
TCP: 06	

**Some protocol values**



**Fig:** Protocol field and encapsulated data

**SCTP:** Stream control transmission protocol

**TCP:** Transmission control protocol

**UDP:** User Datagram Protocol

**ICMP:** Internet Control Message Protocol

**IGMP:** Internet Group Management Protocol

**OSPF:** Open Shortest Path First

- **Checksum:** Used for error detection of the header parts only but not the data.

This is because;

- First, **all higher-level protocols** that encapsulate data in the IPv4 datagram have a **checksum field** that covers the whole packet.
- Second, the header of the IPv4 packet changes with each visited router, but the data do not.

**Note:** First, the value in the checksum field is **kept 0**. Then the entire header is divided into **16-bit sections** and added together. The result (sum) is **complemented** and inserted into the checksum field.

- **Option:** It is not compulsory part of the IPv4 header. Its maximum size is of 40 bytes and this field is used for **network testing and debugging**.

# Fragmentation

## What is fragmentation and why it is needed?

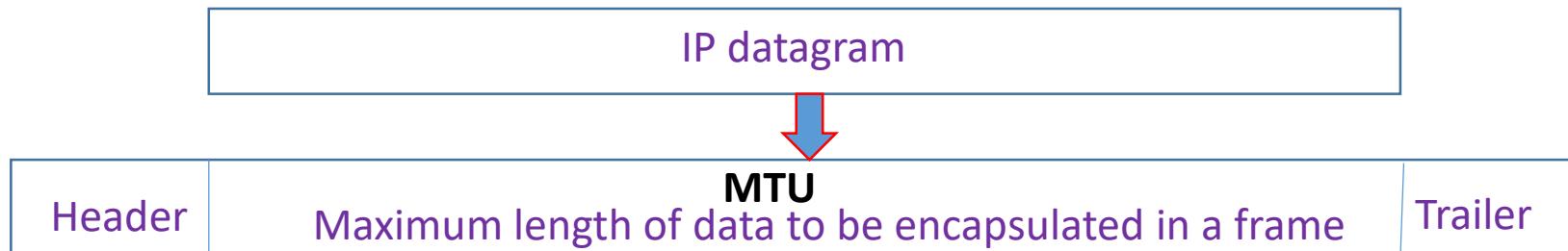


- **Need:** The value of maximum transfer unit (MTU) depends upon the physical layer protocol. For e.g., the maximum and minimum value of MTU is **65,535** in **case of Hyperchannel** and **296** in **case of PPP**. If size of IPv4 datagram equals to 65,535 then transmission becomes more efficient. But at the same time, it is not possible to send this size of datagram over other than Hyperchannel network.

Hence, we need to **divide a datagram into smaller datagrams** such that it can pass over any physical networks. This is called **fragmentation**.

**Maximum Transfer Unit (MTU):** It is the maximum size of the data field.

In other words, when a datagram is encapsulated in a frame, the total size of the datagram must be less than this maximum size.



# Fields Related to Fragmentation

- **Identification:** The identification number helps the destination in reassembling the datagram. It knows that all fragments having the same identification value must be assembled into one datagram.
- **Flags:** The first bit is reserved. If **D=1**, then datagram can not be fragmented. If **M=1**, there are more fragments after this one. If **M=0**, it means this is **the last or only fragment**.

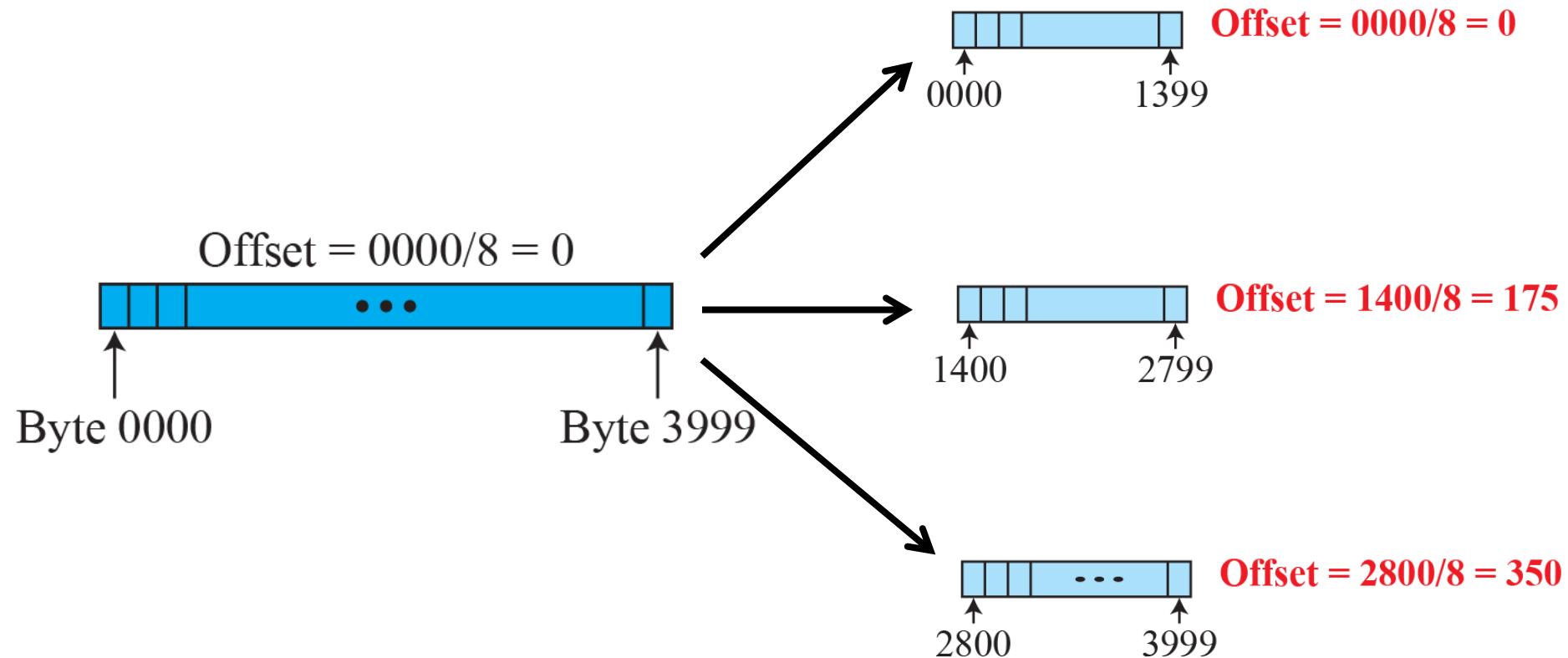


**Fig: Flags used in fragment**

**D:** Do not fragment  
**M:** More fragment

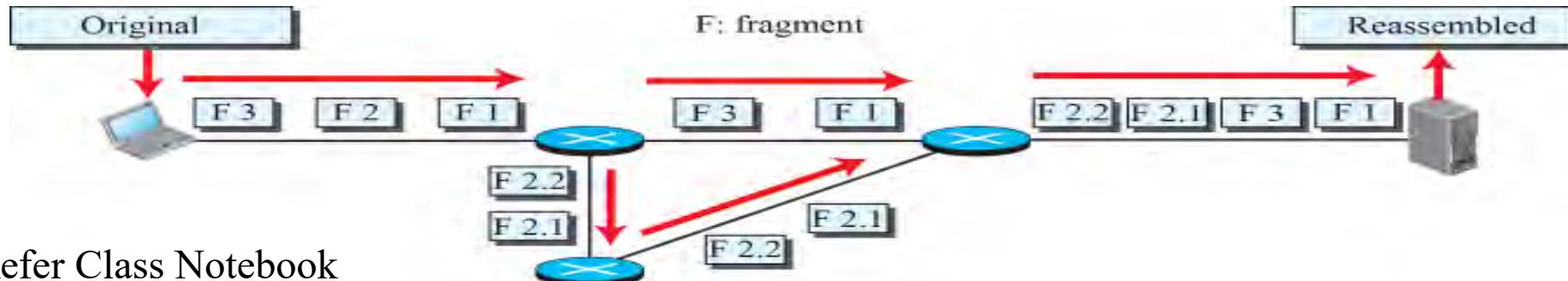
- **Fragment Offset:** This 13-bit field shows the relative position of this fragment with respect to the whole datagram. It is measured in units of 8 bytes.

**Example 4.2:** Fragmentation in a datagram with a data size of 4000 bytes if MTU allowed is 1420 bytes only.

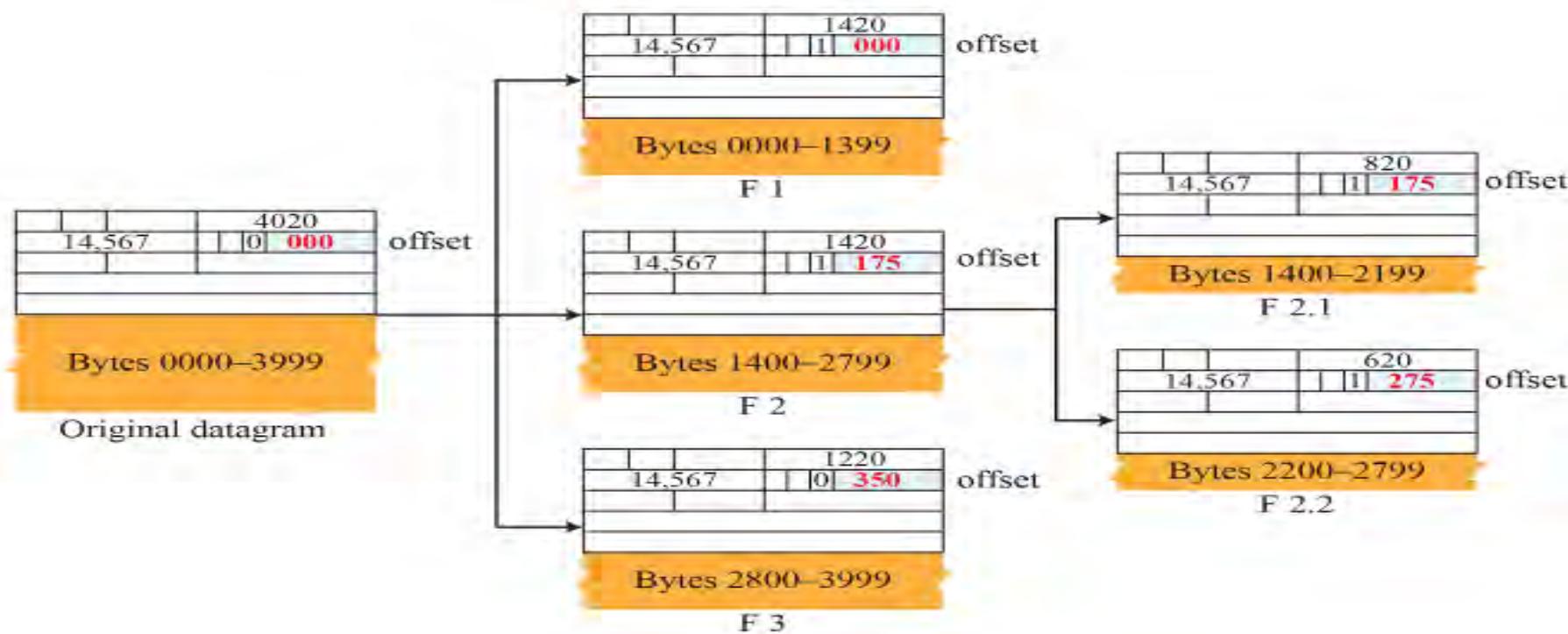


**Fig:** Fragmentation example

# Do we need reassembly? If yes, then how?



Example 4.3: Refer Class Notebook





**Question:** Do we need reassembly?

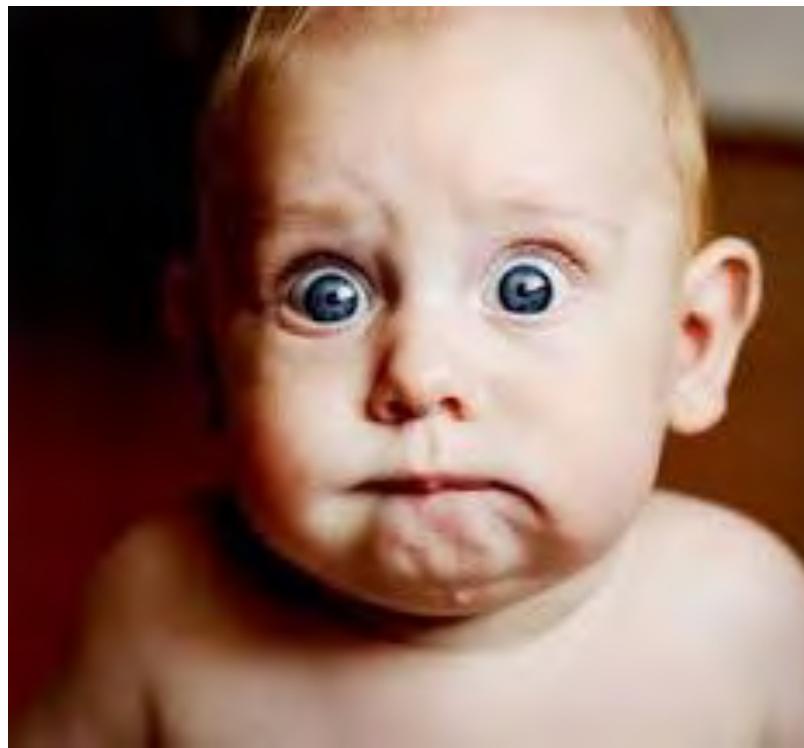
**Yes, we need reassembly of the packets at the receiver side:** It is obvious that even if each fragment follows a different path and arrives out of order, the **final destination host needs** to arrange the datagram in proper order.

**Question:** How the different fragmentations of a datagram will be rearranged?

**Answer:** The original datagram from the fragments received (if none of them is lost) can be rearranged by using the following strategy:

1. The first fragment has an offset field value of zero.
2. Divide the length of the first fragment by 8. The second fragment has an offset value equal to that result.
3. Divide the total length of the first and second fragments by 8. The third fragment has an offset value equal to that result.
4. Continue the process. The last fragment has a more fragment bit value of 0.

Confused ????



**Blind Rule to decide fragment offset.**

**Blind Rule:** Divide the first data of the fragment by 8 to get the offset value of that fragment.

## **HYU 4.1:**

A 1480 byte IPv4 datagram is fragmented into 800 byte, 400 byte and 280 byte datagram. The second fragment is again fragmented into two fragments of 200 byte each. Neglect the optional header. Explain the process by mention the following fields for each fragmented datagram.

- i) Total length
- ii) Identification
- iii) Flag
- iv) Offset

Assume identification field value for original datagram is 14,657.

**Example 4.4:** A packet has arrived with an  $M$  bit value 0. Is this the first fragment, the last fragment, or a middle fragment? Do you know if the packet was fragmented?

**Answer:** It is the last fragment. However, we can't say whether original packet is fragmented or not. A non-fragmented packet is considered the last fragmented.

**Example 4.5:** Repeat example 1 if value of  $M$  bit is 1.

# All in One concept !!!!!!!

**Example 4.6:** An IPv4 packet has arrived with the first few hexadecimal digits as “0x4802002800210000020001.....”.

Answer the following questions.

- a) This packet belongs to version 4 or 6?
- b) How many bytes of options are carried by this packet?
- c) Which type of the service is associated with this packet viz. delay, throughput, reliability and cost?
- d) How many bytes of data are being carried by this packet?
- e) What is the sequence number of this packet?
- f) Is it first or last fragment?
- g) How many hops can it pass through?
- h) Is the packet correctly received?

## Answers:



- a) Version 4 since first byte is 4.
- b) Second byte is 8. so HLEN is  $8*4=32$  bytes. So  $32-20= 12$  bytes of options are being carried.
- c) 0x02 in binary = 00000010 thus minimize cost.
- d) 0x0028 in binary = 0000 0000 0010 1000 = 40 in decimal. Thus  $(40-32)= 8$  bytes of data.
- e) 0x0021 in binary = 0000 0000 0010 0001 = 33 in decimal. Thus sequence number is 33
- f) First fragment since offset value is 0. (**Note:** If offset would not have been 0, it would have been last fragment.)
- g) 9<sup>th</sup> byte is 02 which says 2 number of hops.
- h) It cant be said since we don't know the value of rest of the fields.

# Practice Questions as HYU

2. A 64 bytes of packet is sent over a 100 Mbps link. If the value of TTL field is 61 at the destination (assume TTL= 63 at source) then find the total time required to reach the packet to destination. Assume that the waiting time at each intermediate router is 20 msec.
3. What is the content of HLEN field in IPv4 datagram when header is followed by two optional header of 16 bytes each?
4. An IPv4 packet has arrived with the first 8 bits as “01000010”. Will packet be discarded or accepted?
5. In an IPv4 packet, the value of HLEN is 1000 in binary. How many bytes options are being carried by this packet?

# CN (IT-3001)

## Network Layer: IP Addressing

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose. Images portrayed (if any) are not to hurt the sentiments of any person.

# Objectives

- The objective of this module is to discuss following concepts...
  1. IP Addressing
  2. Subnetting
  3. Supernetting

# Logical addressing/IP addressing

- **Motivation:** The MAC address does not work if packet to be transmitted from one network to other network.

Thus, we need other addressing method, IP address, which can be used to remove limitations of MAC address.

## Recent Development: **IPTV**



# IPv4 Addressing

- It is a **32-bit address** that *uniquely* and *universally* defines the connection of a device to the Internet.
- Two devices on the Internet can never have the same address **at the same time**.
- On the other hand, if a device operating at the network layer **has  $m$  connections to the Internet**, it needs to have  $m$  addresses, e.g., Router.
- **Address Space:**
  - The address space of IPv4 is  $2^{32} = 4,294,967,296$ .
  - So, if there were no restrictions, we can have **more than 4 billion** devices on the Internet theoretically. However, this number is much less practically.

# IPv4 Addressing: Notations

- Two notations: Binary notation and dotted-decimal notation.
- **Binary Notation:** In this, address is displayed as 32 bits. So, IPv4 address in this case referred to as a 32-bit address or a 4-bytes address.

**00001010 00000000 01001001 00010111**

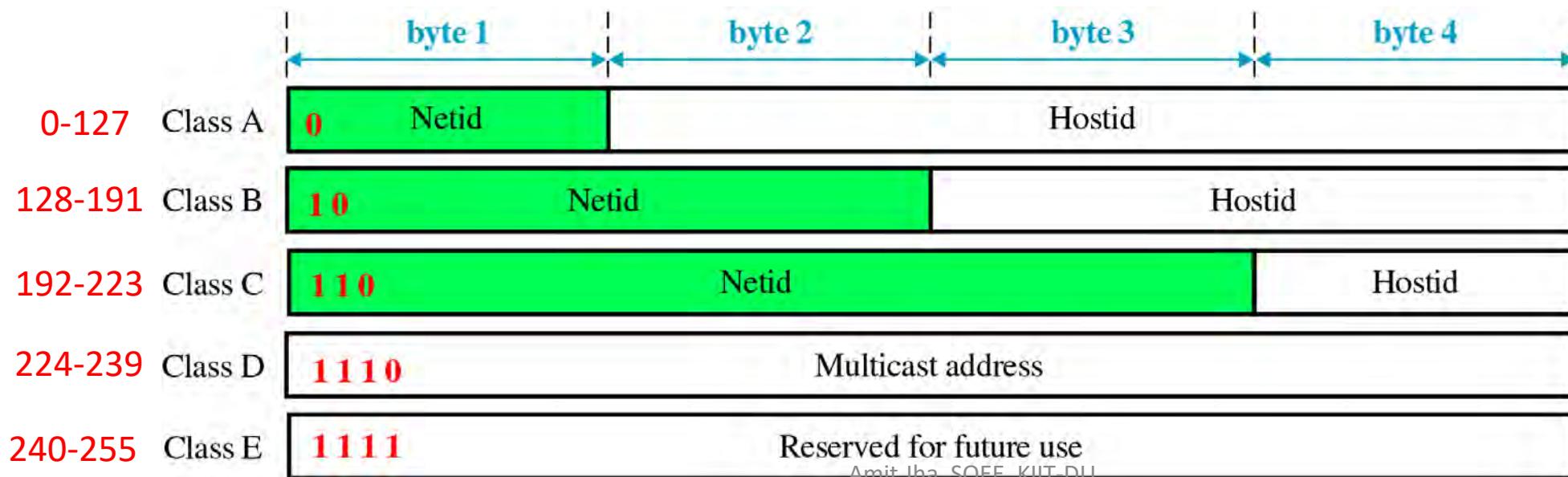
- **Dotted-Decimal Notation:** It is compact and easy to read.

**00001010 00000000 01001001 00010111**

**10.0.73.23**

# IPv4 Addressing: Classful Addressing

- Total address space is divided into five classes, A, B, C, D, E.
- In binary notation, the **first few bits** can immediately tell us the class of the address.
- In dotted-decimal notation, **the first byte** defines the class.



# IPv4 Addressing: Class Ranges

	From	To
<b>Class A</b>	<b>0.0.0.0</b>	<b>127.255.255.255</b>
	Netid Hostid	Netid Hostid
<b>Class B</b>	<b>128.0.0.0</b>	<b>191.255.255.255</b>
	Netid Hostid	Netid Hostid
<b>Class C</b>	<b>192.0.0.0</b>	<b>223.255.255.255</b>
	Netid Hostid	Netid Hostid
<b>Class D</b>	<b>224.0.0.0</b>	<b>239.255.255.255</b>
	Group address	Group address
<b>Class E</b>	<b>240.0.0.0</b>	<b>255.255.255.255</b>
	Undefined	Undefined

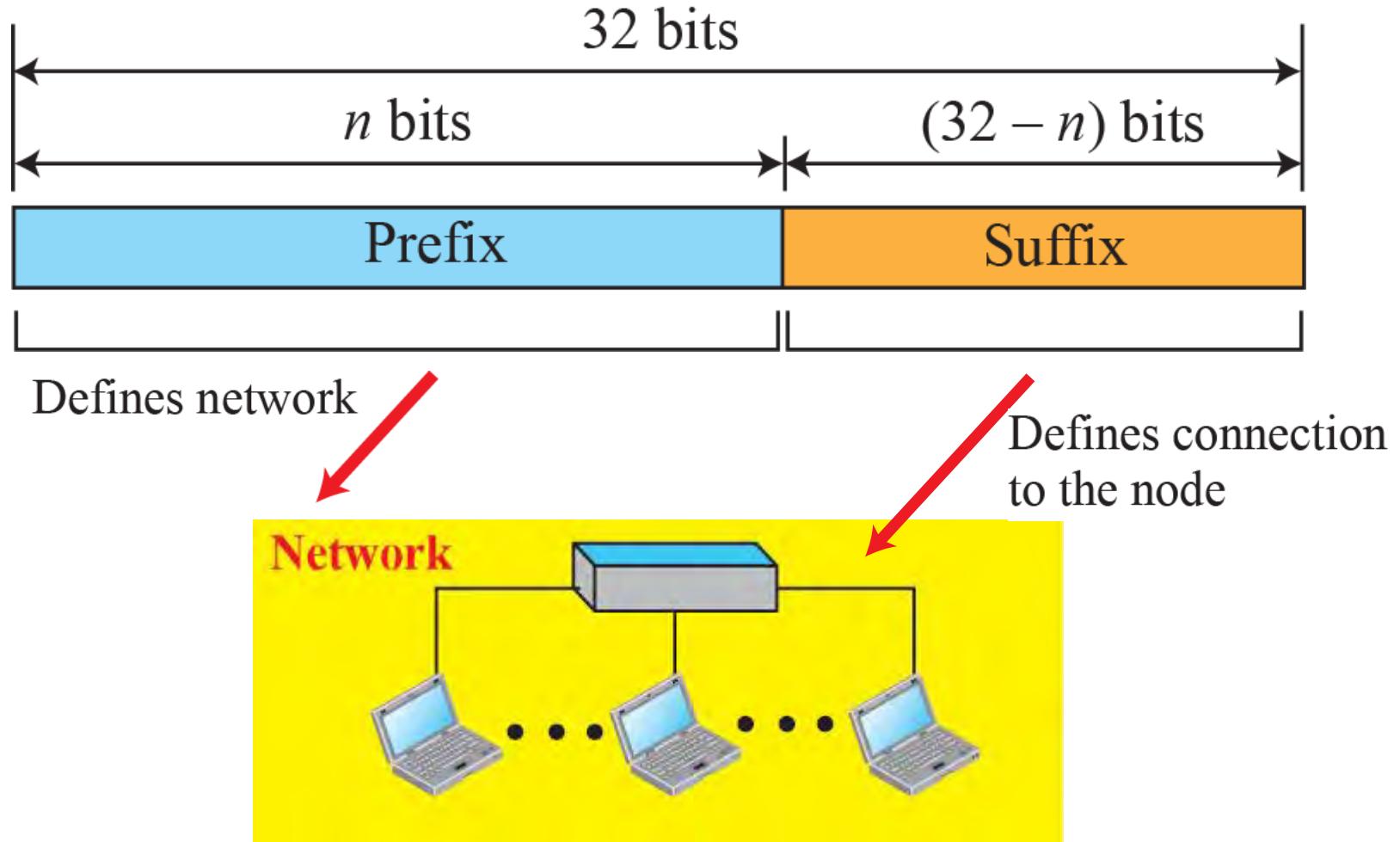
Total # Networks Possible	Total # Hosts Possible	Application
$2^7 = 128$	$2^{24} = 16,777,216$	Unicast
$2^{14} = 16,384$	$2^{16} = 65,536$	Unicast
$2^{21} = 2,097,152$	$2^8 = 256$	Unicast
$2^0 = 1$	$2^{28} = 268,435,456$	Multicast
$2^0 = 1$	$2^{28} = 268,435,456$	Reserved

- **Netid and Hostid:** Netid and Hostid stand for **network address** and **host address** respectively. Netid is called *prefix* and Hostid is called *suffix*.
- **Mask:** It is a 32-bit numbers made of contiguous 1s followed by contiguous 0s. In classful addressing, mask is predetermined number, thus referred to as a *default mask*.
- **Classless Interdomain Routing (CIDR):** It gives the **number of bits used as a mask**. In the address ***a.b.c.d /n***, *n* represents number of masking bits.

### ***Default masks for classful addressing***

<i>Class</i>	<i>Binary</i>	<i>Dotted-Decimal</i>	<i>CIDR</i>
A	1111111 00000000 00000000 00000000	255.0.0.0	/8
B	1111111 1111111 00000000 00000000	255.255.0.0	/16
C	1111111 1111111 1111111 00000000	255.255.255.0	/24

# Hierarchy in addressing



# **Classful addressing is not an efficient method of IPv4 addressing.....**

Class A addresses were designed for large organizations with a large number of attached hosts or routers. Class B addresses were designed for mid size organizations, and class C for small organizations.



Let us say, when KIIT has established, it was given a class C address. So, in the starting, KIIT could have used this **class C** address to accommodate **256 hosts**, approximately all it could have. Due to the exponential growth of KIIT, it now need to have more than 10000 hosts. But, as per the class C, it can handle at max 256 hosts only if it could have used classful addressing.

**Note: In Classful addressing, either a large part of the available addresses are wasted or we are unable to accommodate all hosts.**

# Classless Addressing & Block Allocation

- In this scheme, there are no classes, but the addresses are still granted in blocks. This is done by the ICANN.
- However, ICANN does not normally allocate addresses to individual Internet users. It assigns a large block of addresses to an IP.
- In classless addressing, when an entity, small or large, needs to be connected to the Internet, it is granted a block (range) of addresses by ISP.
- The size of the block depends upon the size of entity.
- For proper operation of the CIDR, three restrictions need to be applied to the allocated block. They are as follows:-
  - **Restriction:**
    1. The addresses in a block must be contiguous, one after another.
    2. The number of addresses in a block N must be a power of 2.
    3. The first address must be evenly divisible by the number of addresses.

**HYU 4.6:** Why all three restrictions need to be followed to allocate the block of IP addresses?

## Example 4.7: An Example of classless address with Block Allocation

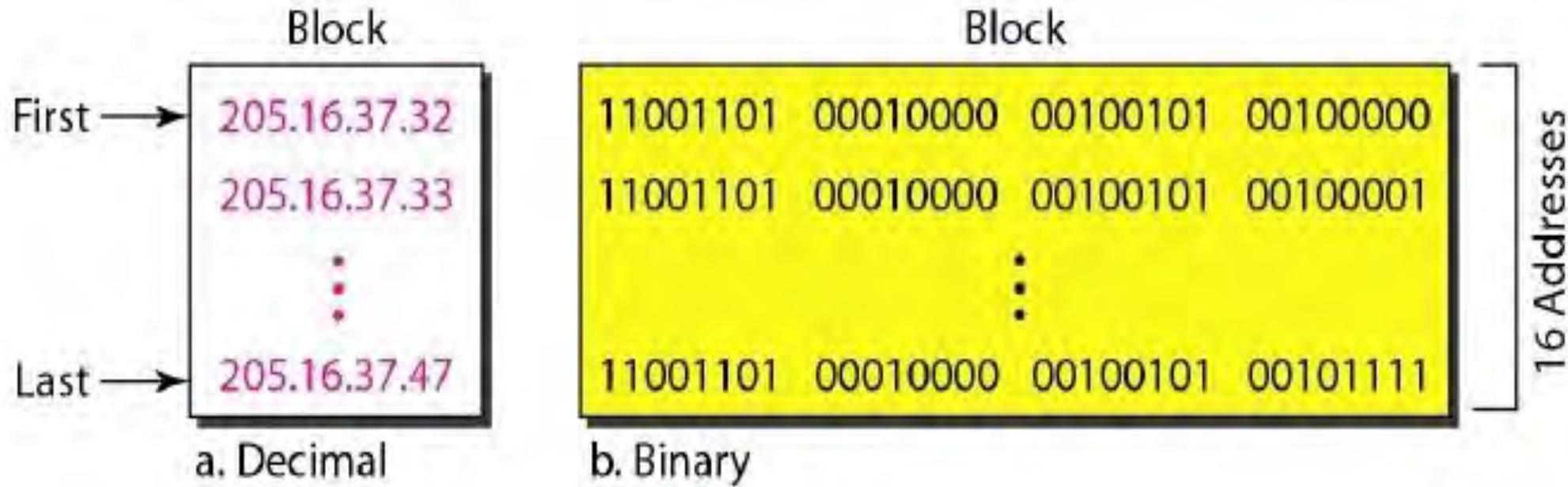


Fig: *A block of 16 addresses granted to a small organization*

**Note:** The above addressing obeys all three restrictions.

# Classless Addressing: CIDR

- The address and the  $/n$  notation completely define the whole block (the first address, the last address, the number of addresses, and the network address).
- **Number of Address:** It can be found out by using the formula  $2^{32-n}$ .
- **First Address:** The first address in the block can be found by setting the right most  $32-n$  bits to 0s.
- **First address** can also be found out by doing ‘AND’ operation of any one of the given/known address with Mask.
- **Last Address:** The last address in the block can be found by setting the right most  $32-n$  bits to 1s.
- **Network Address:** The first address in the class, however, is normally (not always) treated as a network address.
- **Mask:** It is obtained by making leftmost  $n$  bits as 1s.

**Example 4.8:** If in an organization, one of the address is **205.16.37.39/28** then find out, its mask, network address, the first address, and the last address.

**Sol:** Here, n=28.      205.16.37.39 **in binary** 11001101 00010000 00100101 00100111

**Mask** = 11111111 11111111 11111111 11110000 or **255.255.255.240**

**First address:**  $32-n=32-28=4$ , so make rightmost 4 bits as 0s.

so the first address is 11001101 00010000 00100101 00100000 or **205.16.37.32/28**

**Last address:** make rightmost 4 bits 1s.

11001101 00010000 00100101 00101111 or **205.16.37.47/28**

**Network address:** 205.16.37.32/28      First address

The network address can also be found out as (**Given address AND Mask**)

e.g.,      11001101 00010000 00100101 00100111      } AND Operation

11111111 11111111 11111111 11110000

11001101 00010000 00100101 00100000 = 205.16.37.32/28

**Example 4.9:** If in an organization, one of the address is **167.199.170.82/27** then find out, its mask, network address, the first address, and the last address.

**Sol:** Here, n=27.      167.199.170.82 **in binary** 10100111 11000111 10101010 01010010

**Mask** = 11111111 11111111 11111111 11100000 or **255.255.255.224**

**First address:** 32-n=32-27=5, so make rightmost 5 bits as 0s.

so the first address is 10100111 11000111 10101010 01000000 or **167.199.170.64/27**

**Last address:** make rightmost 5 bits 1s.

10100111 11000111 10101010 01011111 or **167.199.170.95/27**

**Network address:** **167.199.170.64/27** ← **First address**

The network address can also be found out as (**Given address AND Mask**)

e.g.,

$$\begin{array}{cccccc} 10100111 & 11000111 & 10101010 & 01010010 & \text{AND Operation} \\ 11111111 & 11111111 & 11111111 & 11100000 \\ \hline 10100111 & 11000111 & 10101010 & 01000000 = & \text{167.199.170.64/27} \end{array}$$

**Example 4.10:** An ISP has requested a block of 1000 addresses. Determine the prefix length and verify whether the first address is divisible by total no of addresses or not. Also, note whether it follows all three restrictions of block allocation or not.

**Answer:-** Since 1000 is not a power of 2, 1024 addresses are granted. The prefix length is calculated as  $n = 32 - \log_2 1024 = 22$ . Assume that an available block, 18.14.12.0/22, is granted to the ISP. It can be seen that the first address in decimal is 302,910,464, which is divisible by 1024.

**Note:**  $18.14.12.0 = 00010010. 00001110. 00001100. 00000000 \rightarrow 302910464$  (*combining all bytes together*)

# What is the concept of *subnet*?

The process of dividing a *larger network* into *smaller number* of sub networks is known as *subnet*.

**Blind Rule:** Borrow from the Host ID, thus subnetting increases the number of 1s in the mask (prefix).

# Subnetting

- More level of hierarchy can be created using subnetting.
- An organization (or an ISP) that is granted a range of addresses may divide the range into several subranges and assign each subrange to a subnetwork (or subnet).
- Note that nothing stops an organization from creating more levels, i.e.,
  - A subnetwork can be divided into several sub-subnetworks.
  - A sub-subnetwork can be divided into several sub-sub-subnetworks. And so on..
-

# Designing of Subnets

Subnet should be designed properly to facilitate routing of packets.

- Let total number of addresses granted to the organization is  $N$ , the prefix length is  $n$ , the assigned number of addresses to each subnetwork is  $N_{sub}$  and the prefix length for each subnetwork is  $n_{sub}$ . Then following rules is applied.
  - The number of addresses in each subnetwork should be power of 2.
  - The prefix length of each subnetwork is found using  $n_{sub} = 32 - \log_2 (N_{sub})$ .
  - The starting address in each subnetwork should be divisible by the number of addresses in the subnetwork. This can be achieved if we first assign addresses to the larger subnetwork.

# Few Key points to Remember for Subnet

- **For a subnet,**
  - A subnet with **subnet id (prefix id) bits as all zeros** is not used → because this is not allowed by most of the routers, however, some of the routers like Cisco Systems devices allow the use of these subnets when the **id subnet zero** command is configured.
  - Similarly, a **subnet with subnet id bits as all ones** is not allowed → however, some of the devices may allow to use these subnetworks.
- **In a subnet,**
  - Host id with **all host id (suffix id) bits set to zero** is not allowed as host IP address → this is because, this results in first IP address in that subnet which is generally a network (subnetwork here) address.
  - Host id (suffix id) with **all host id bits set to one** is not allowed as host IP address → this is because, this results in the broadcast address in that network (subnetwork here).

**Example 4.11:-** Given the Class C network of 204.15.5.0/24, subnet the network in order to create the network as shown in figure 3 with the host requirements depicted therein.

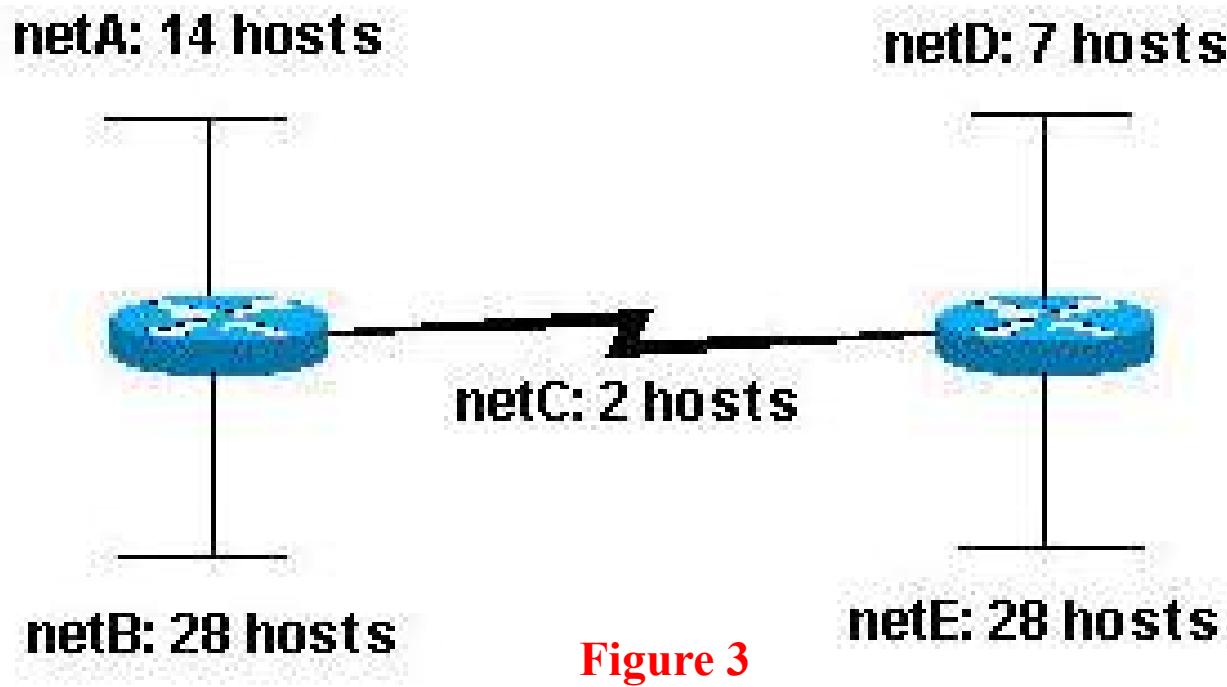


Figure 3

**Solution:-** you can see that you are required to create five subnets. The largest subnet must support 28 host addresses.

*Is this possible with a Class C network? and if so, then how?*

Lets design the subnet by first answering the following points:-

How many subnets we need?

**Ans:- five**

How many bits we need to borrow at least from the host id ?

**Ans:- three bits**

Note:- Two bits would only allow you four subnets ( $2^2$ ).

How many hosts does this support?

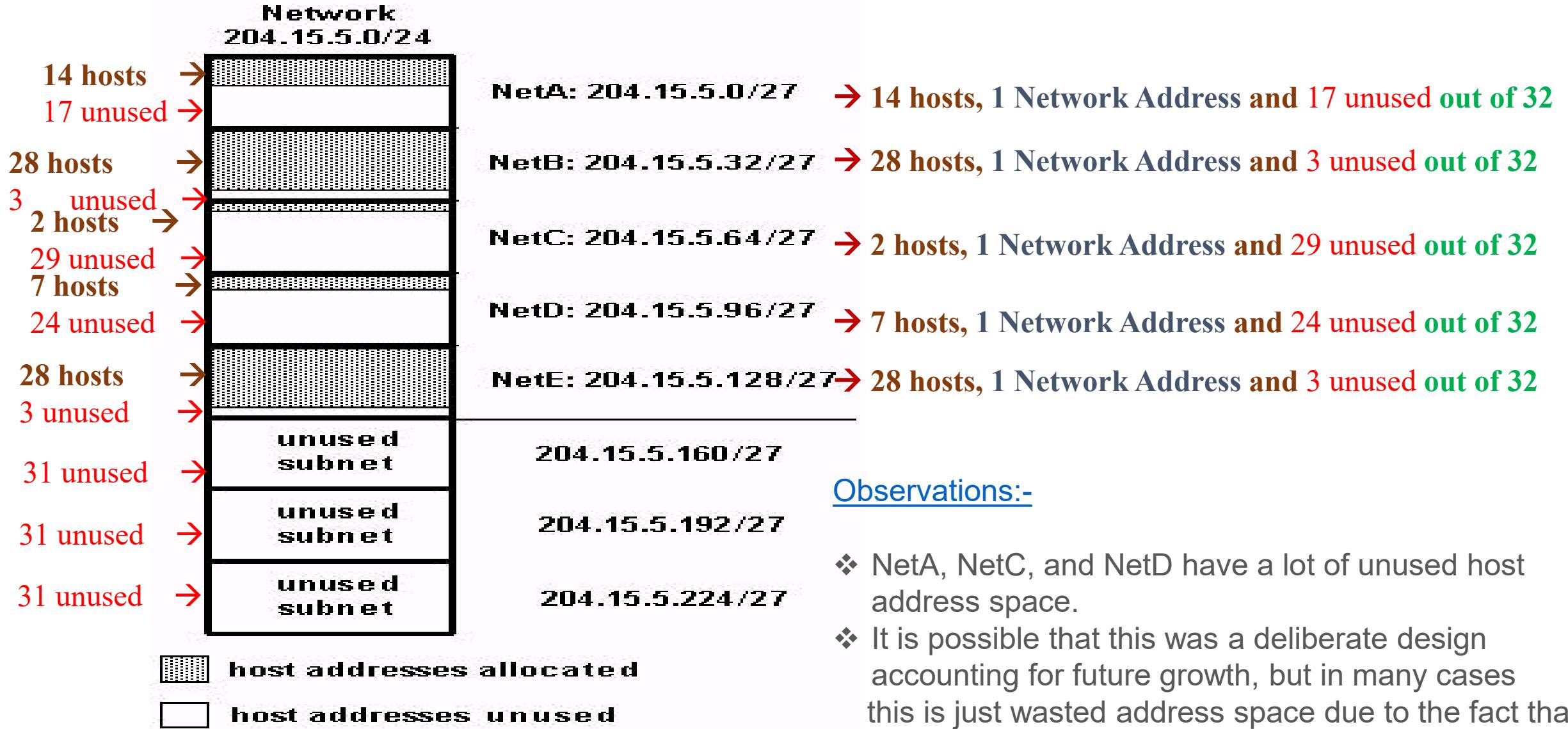
**Ans:-  $2^5 = 32$  (30 usable).** since host ids of all zeros or all ones are not allowed (it is very important to remember this).

*Let us now create the subnetworks.....*

Name of Subnet	Mask	Network Address	Range of IP Address	Remark
netA	255.255.255.224	204.15.5.0/27	204.15.5. <b>00000000</b> / <b>27</b> = 204.15.5. <b>0</b> / <b>27</b> 204.15.5. <b>00000001</b> / <b>27</b> = 204.15.5. <b>1</b> / <b>27</b> 204.15.5. <b>00000010</b> / <b>27</b> = 204.15.5. <b>2</b> / <b>27</b> : : 204.15.5. <b>00011110</b> / <b>27</b> = 204.15.5. <b>30</b> / <b>27</b> 204.15.5. <b>00011111</b> / <b>27</b> = 204.15.5. <b>31</b> / <b>27</b>	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used
netB	255.255.255.224	204.15.5.32/27	204.15.5. <b>00100000</b> / <b>27</b> = 204.15.5. <b>32</b> / <b>27</b> 204.15.5. <b>00100001</b> / <b>27</b> = 204.15.5. <b>33</b> / <b>27</b> 204.15.5. <b>00100010</b> / <b>27</b> = 204.15.5. <b>34</b> / <b>27</b> : : 204.15.5. <b>00111110</b> / <b>27</b> = 204.15.5. <b>62</b> / <b>27</b> 204.15.5. <b>00111111</b> / <b>27</b> = 204.15.5. <b>63</b> / <b>27</b>	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used
netC	255.255.255.224	204.15.5.64/27	204.15.5. <b>01000000</b> / <b>27</b> = 204.15.5. <b>64</b> / <b>27</b> 204.15.5. <b>01000001</b> / <b>27</b> = 204.15.5. <b>65</b> / <b>27</b> 204.15.5. <b>01000010</b> / <b>27</b> = 204.15.5. <b>66</b> / <b>27</b> : : 204.15.5. <b>01011110</b> / <b>27</b> = 204.15.5. <b>94</b> / <b>27</b> 204.15.5. <b>01011111</b> / <b>27</b> = 204.15.5. <b>95</b> / <b>27</b>	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used

Name of Subnet	Mask	Network Address	Range of IP Address	Remark
netD	255.255.255.224	204.15.5. <b>96</b> /27	204.15.5. <b>01100000</b> / <b>27</b> = 204.15.5. <b>96</b> /27 204.15.5. <b>01100001</b> / <b>27</b> = 204.15.5. <b>97</b> /27 204.15.5. <b>01100010</b> / <b>27</b> = 204.15.5. <b>98</b> /27 : : 204.15.5. <b>01111110</b> / <b>27</b> = 204.15.5. <b>126</b> /27 204.15.5. <b>01111111</b> / <b>27</b> = 204.15.5. <b>127</b> /27	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used
netE	255.255.255.224	204.15.5. <b>128</b> /27	204.15.5. <b>10000000</b> / <b>27</b> = 204.15.5. <b>128</b> /27 204.15.5. <b>10000001</b> / <b>27</b> = 204.15.5. <b>129</b> /27 204.15.5. <b>10000010</b> / <b>27</b> = 204.15.5. <b>130</b> /27 : : 204.15.5. <b>10011110</b> / <b>27</b> = 204.15.5. <b>158</b> /27 204.15.5. <b>10011111</b> / <b>27</b> = 204.15.5. <b>159</b> /27	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used

# Used Vs Unused IP Address



**Question:-** How can we avoid unnecessary wastage of IP addresses in designing subnetworks?

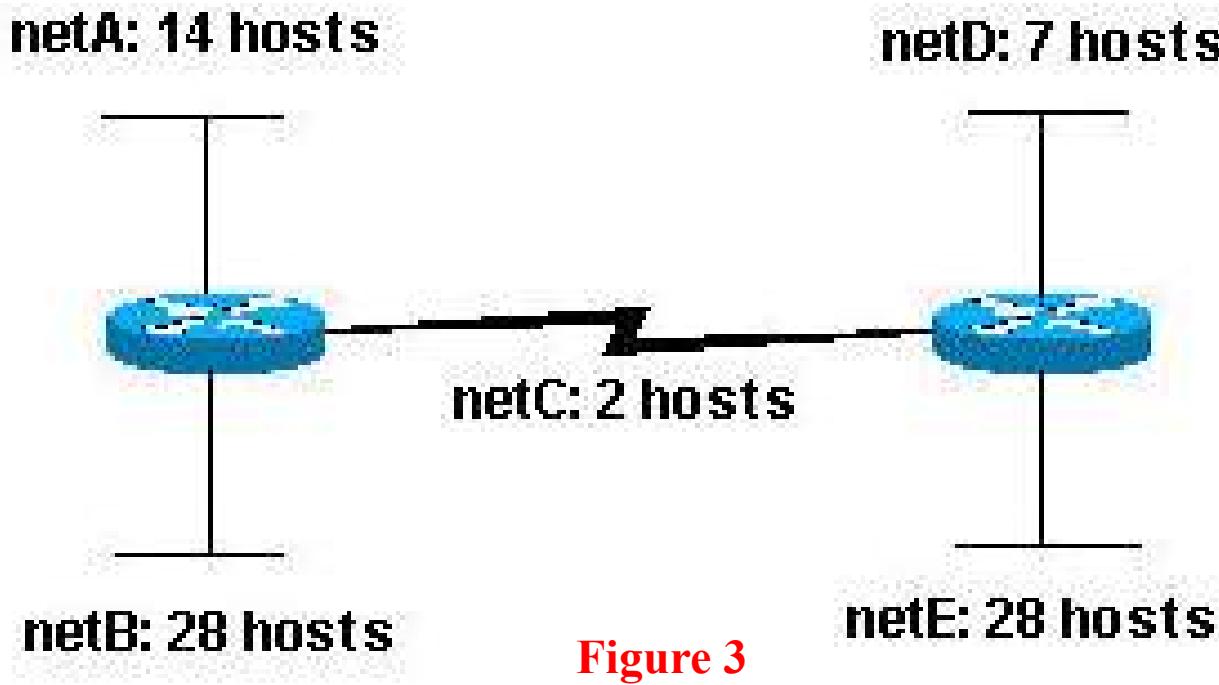
**Ans:-**

By using Variable Length Subnet Masks (VLSM), we can avoid the unnecessary wastage of IP addresses. VLSM allows us to use different masks for each subnet, thereby using address space efficiently.

# VLSM (Variable Length Subnet Mask)

- Here, instead of allocating the same subnet mask for all the subnets, **different subnet mask** will be applicable for different network depending upon the requirement of the subnetworks.
- To Understand, let us solve the same example using VLSM as discussed in the next slide.
- ***Note:- In previous example, the mask was same for all the subnets.***
- **Summary:-**
- **In VLSM**→ Masking bits will be decided as per the requirements of the subnets (i.e., number of host addresses in subnet).
- **Without VLSM**→ The masking bit will be decided as per the number of subnets and not on the # host.

**Example 4.12:-** Given the Class C network of 204.15.5.0/24, subnet the network in order to create the network as shown in figure 3 with the host requirements depicted therein. Use VLSM.



# Solution

Determine what mask allows the required number of hosts.

- **netA:** requires a /28 (255.255.255.240) mask to support 14 hosts
- **netB:** requires a /27 (255.255.255.224) mask to support 28 hosts
- **netC:** requires a /30 (255.255.255.252) mask to support 2 hosts
- **netD\*:** requires a /28 (255.255.255.240) mask to support 7 hosts
  - Note:- \* a /29 (255.255.255.248) would only allow 6 usable host addresses therefore netD requires a /28 mask.
- **netE:** requires a /27 (255.255.255.224) mask to support 28 hosts

Address Assignment:- The easiest way to assign the subnets is to assign **the largest first**. For example, you can assign in this manner:

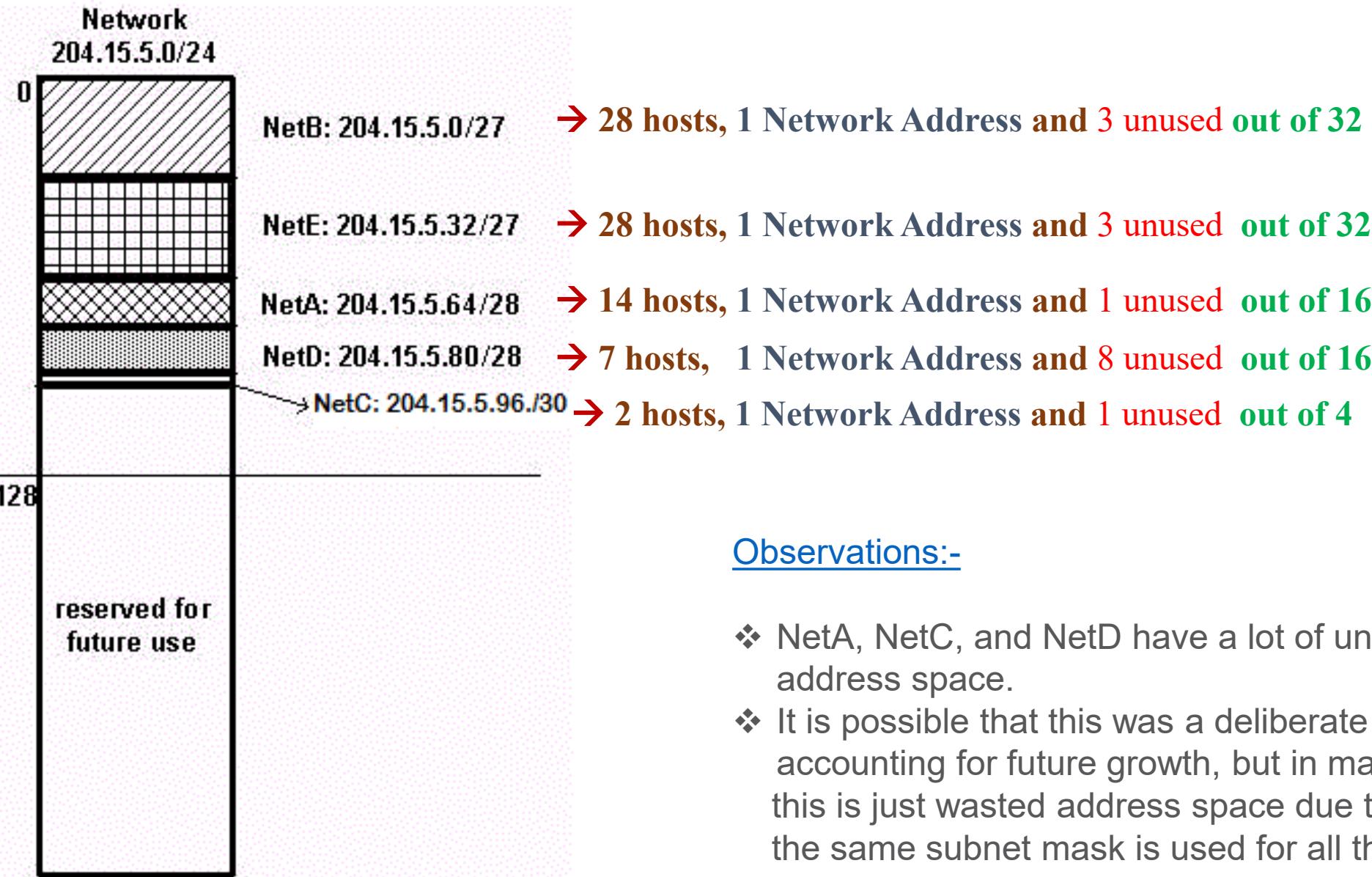
- ✓ netB: 204.15.5.**0**/27 host address range 1 to 30,
- ✓ **netE:** 204.15.5.**32**/27 host address range 33 to 62,
- ✓ **netA:** 204.15.5.**64**/28 host address range 65 to 78,
- ✓ **netD:** 204.15.5.**80**/28 host address range 81 to 94,
- ✓ **netC:** 204.15.5.**96**/30 host address range 97 to 98.

A complete network design is shown in the next slide.

Name of Subnet	Mask	Network Address	Range of IP Address	Remark
netB (28 Hosts)	255.255.255.224	204.15.5.0/27	204.15.5.00000000/27 = 204.15.5.0/27 204.15.5.00000001/27 = 204.15.5.1/27 204.15.5.00000010/27 = 204.15.5.2/27 : : 204.15.5.00011110/27 = 204.15.5.30/27 204.15.5.00011111/27 = 204.15.5.31/27	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used
netE (28 Hosts)	255.255.255.224	204.15.5.32/27  <i>Note: 0+2^5= 32</i>	204.15.5.00100000/27 = 204.15.5.32/27 204.15.5.00100001/27 = 204.15.5.33/27 204.15.5.00100010/27 = 204.15.5.34/27 : : 204.15.5.0011110/27 = 204.15.5.62/27 204.15.5.0011111/27 = 204.15.5.63/27	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used
netA (14 Hosts)	255.255.255.240	204.15.5.64/28  <i>Note: 32+2^5= 64</i>	204.15.5.01000000/28 = 204.15.5.64/28 204.15.5.01000001/28 = 204.15.5.65/28 204.15.5.01000010/28 = 204.15.5.66/28 : : 204.15.5.01001110/28 = 204.15.5.78/27 204.15.5.01001111/28 = 204.15.5.79/27	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used

Name of Subnet	Mask	Network Address	Range of IP Address	Remark
netD (7 Hosts)	255.255.255.240	204.15.5. <b>80</b> /28 <i>Note: 64+2^4= 80</i>	204.15.5. <b>01010000</b> / <b>28</b> = 204.15.5. <b>80</b> /28 204.15.5. <b>01010001</b> / <b>28</b> = 204.15.5. <b>81</b> / <b>28</b> 204.15.5. <b>01010010</b> / <b>28</b> = 204.15.5. <b>82</b> / <b>28</b> : : 204.15.5. <b>01011110</b> / <b>28</b> = 204.15.5. <b>94</b> / <b>28</b> 204.15.5. <b>01011111</b> / <b>28</b> = 204.15.5. <b>95</b> / <b>28</b>	→ Can not be used → First host address → 2 <sup>nd</sup> Host address : : → Last host address → Can not be used
netC (2 Hosts)	255.255.255.252	204.15.5. <b>96</b> /30 <i>Note: 80+2^4= 96</i>	204.15.5. <b>01100000</b> / <b>30</b> = 204.15.5. <b>96</b> /30 204.15.5. <b>01100001</b> / <b>30</b> = 204.15.5. <b>97</b> /30 204.15.5. <b>01100010</b> / <b>30</b> = 204.15.5. <b>98</b> /30 204.15.5. <b>01100011</b> / <b>30</b> = 204.15.5. <b>99</b> /30	→ Can not be used → First host address → 2 <sup>nd</sup> Host address → Can not be used

# Used Vs Unused IP Address

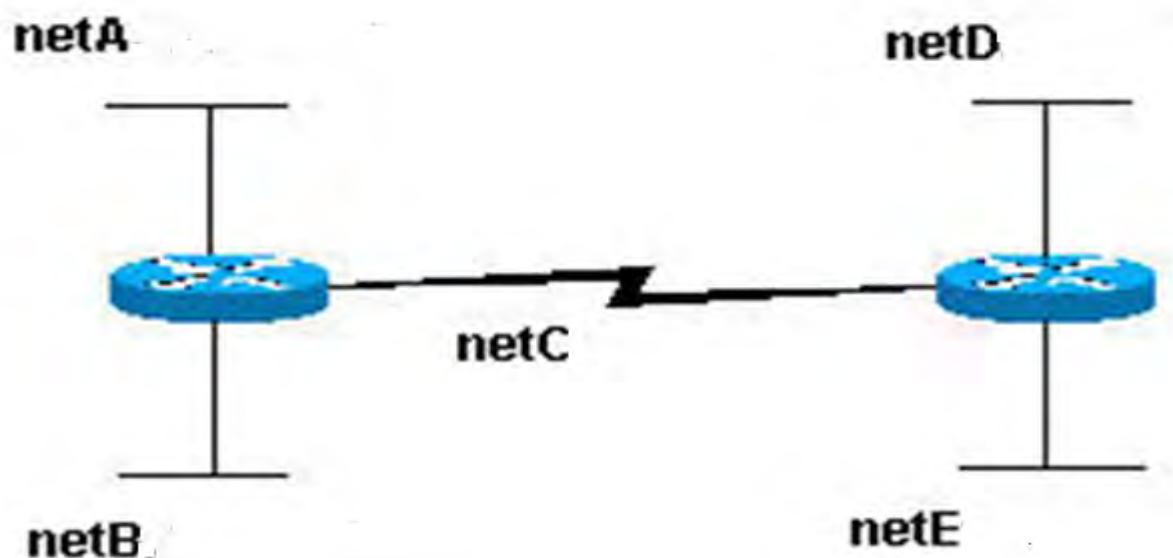


## Observations:-

- ❖ NetA, NetC, and NetD have a lot of unused host address space.
- ❖ It is possible that this was a deliberate design accounting for future growth, but in many cases this is just wasted address space due to the fact that the same subnet mask is used for all the subnets.

- **HYU 4.7:-** Consider the ISP has provided you a Class C IP address 204.15.5.0/24. Now you want to create five subnetworks with name and number of hosts as follows: **Network A** with 126 hosts **Network B** with 62 hosts, **Network C** with 30 hosts, **Network D** with 14 hosts, and **Network E** with 14 hosts as shown in the diagram below. For the scenario elaborated above, design the network by focusing on the following points:

1. Network Address and subnet mask of each sub network.
2. Distribution of IP addresses for the host in each sub network.
3. The total unused IP addresses.
4. % of wastage of IP addresses.
5. Clearly state assumptions (if any).



**HYU 4.8:-** An organization is granted a block of address with the beginning address 14.24.74.0/24. The organization needs to have three subblocks of addresses to use in its three subnets:

- One subblock of 10 addresses
- One subblock of 60 addresses and
- One subblock of 120 addresses

Design the subnetworks. Also, calculate the total unused IP addresses. Textbook solution

**Example 4.13:-** Consider the IP addresses of two devices given below. Identify, whether these devices are connected on same or different network.

DeviceA: 172.16.17.30/20

DeviceB: 172.16.28.15/20

**Solution:-**

**Determine the Subnet for DeviceA:**

172.16.17.30 → 10101100.00010000.00010001.00011110

Subnet Mask → 11111111.11111111.11110000.00000000 = 255.255.240.0

Subnet = 10101100.00010000.00010000.00000000 = 172.16.16.0

In this case, Device A belongs to subnet 172.16.16.0.

**Determine the Subnet for DeviceB:**

172.16.28.15 → 10101100.00010000.00011100.00001111

Subnet Mask → 11111111.11111111.11110000.00000000 = 255.255.240.0

Subnet = 10101100.00010000.00010000.00000000 = 172.16.16.0

From these determinations, DeviceA and DeviceB have addresses that are part of the same subnet.

## Example 4.14:-

- Consider an organization is given the block **17.12.14.0/26**, which contains 64 addresses. The organization has three offices and needs to divide the addresses into three sub-blocks of 32, 16, and 16 addresses. How can this be done?



# Solution

The first office can use a subnet mask  $n_1$ , s.t  $2^{32-n_1} = 32$ . So,  $n_1=27$ .

So, first office **subnet mask** becomes 255.255.255.224

**1<sup>st</sup> address:** 17.12.14.0/27; **last address:** 17.12.14.31/27

Thus, **subnet 1 address** = 17.12.14.0/27

The second office can use a subnet mask  $n_2$ , s.t  $2^{32-n_2} = 16$ . So,  $n_2=28$ .

So, second office **subnet mask** becomes 255.255.255.240

**Any address in subnet 2**, say **17.12.14.35/28**, can give us its address;

**1<sup>st</sup> address:** 17.12.14.32/28; **last address:** 17.12.14.47/28

Thus, **subnet 2 address:** 17.12.14.32/28

Similarly, for the third office, subnet mask  $n_3 = 28$ .

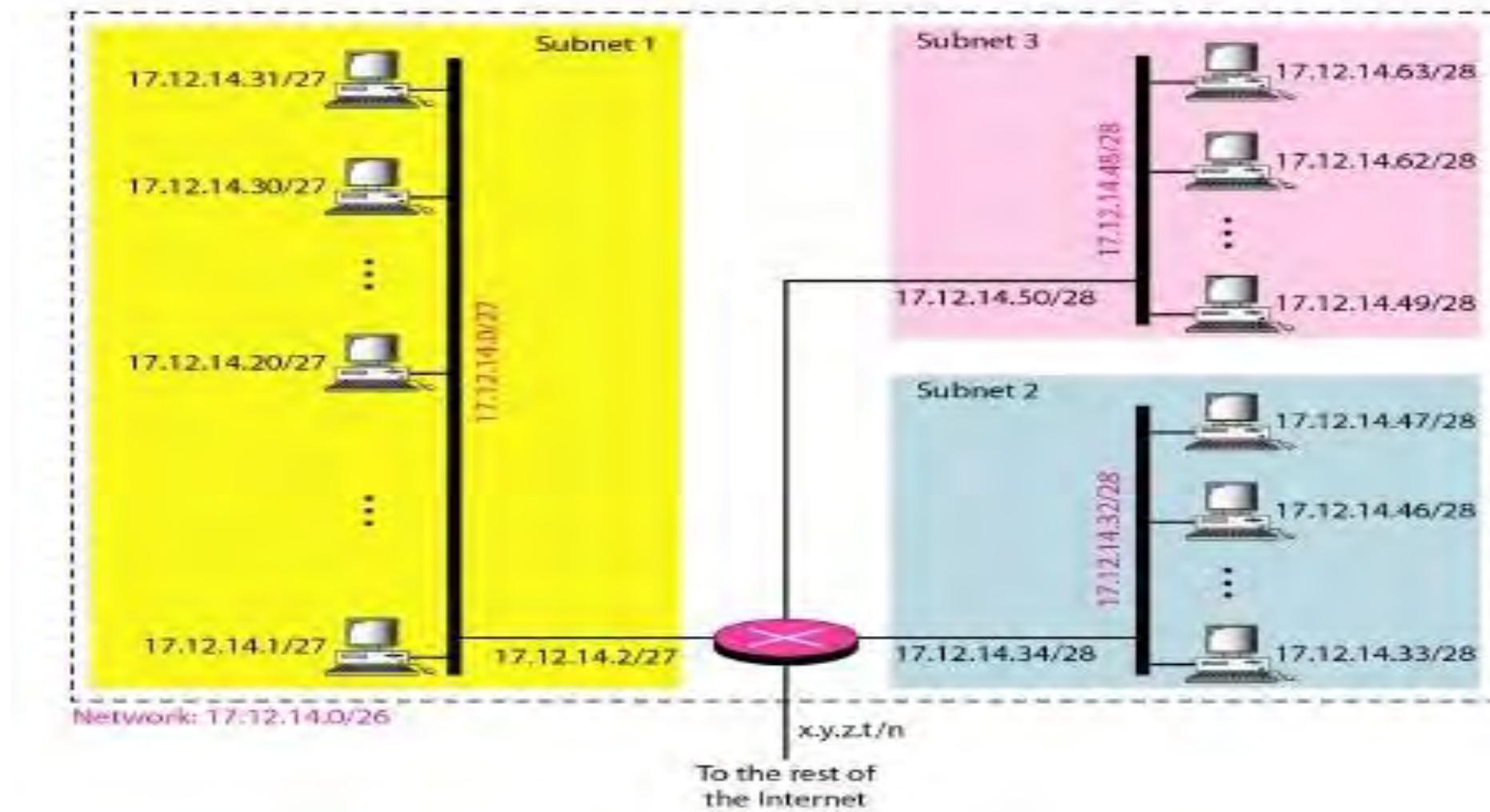
So, third office **subnet mask** becomes 255.255.255.240

**Any address in subnet 3**, say 17.12.14.50/28, can give us its address;

**1<sup>st</sup> address:** 17.12.14.48/28; **last address:** 17.12.14.63/28

Thus, **subnet 3 address:** 17.12.14.48/28.

Note → All the addresses are included.



**Fig:** Configuration and Addresses in a Subnetted Network

# Routing Table

Destination	Next Node	Interface
17.12.14.0/27	17.12.14.0/27	17.12.14.2/27
17.12.14.1/27		
.		
.		
.	-----same-----	-----same-----
17.12.14.31/27		
17.12.14.32/28	17.12.14.32/28	17.12.14.34/28
17.12.14.33/28		
.		
.		
.	-----same-----	-----same-----
17.12.14.47/28		
17.12.14.48/28	17.12.14.48/28	17.12.14.50/28
17.12.14.49/28		
.		
.		
.	-----same-----	-----same-----
17.12.14.63/28		

## HYU 4.9:-

**Problem:** An ISP is granted a block of addresses starting with **190.100.0.0/16** (65,536 addresses). The ISP needs to distribute these addresses to three groups of customers as follows:

- a. The first group has 64 customers; each needs 256 addresses.
- b. The second group has 128 customers; each needs 128 addresses.
- c. The third group has 128 customers; each needs 64 addresses.

Design the subblocks and find out how many addresses are still available after these allocations.

**Brain Exercise:** Observe the number of unused addresses .

## HYU 4.10:-

**Problem:** Assume you are going for a start-up. The ISP has allocated to you 162.12.0.0 as an IP address. But you want to create seven offices at Bhubaneswar, Mumbai, Patna, Kolkata, Guwahati, Pune, and Delhi consisting of 7800 hosts each.

As you have studied DCN course, you do not want to hire any one as a network engineer. Design the network by keeping the following points in mind.

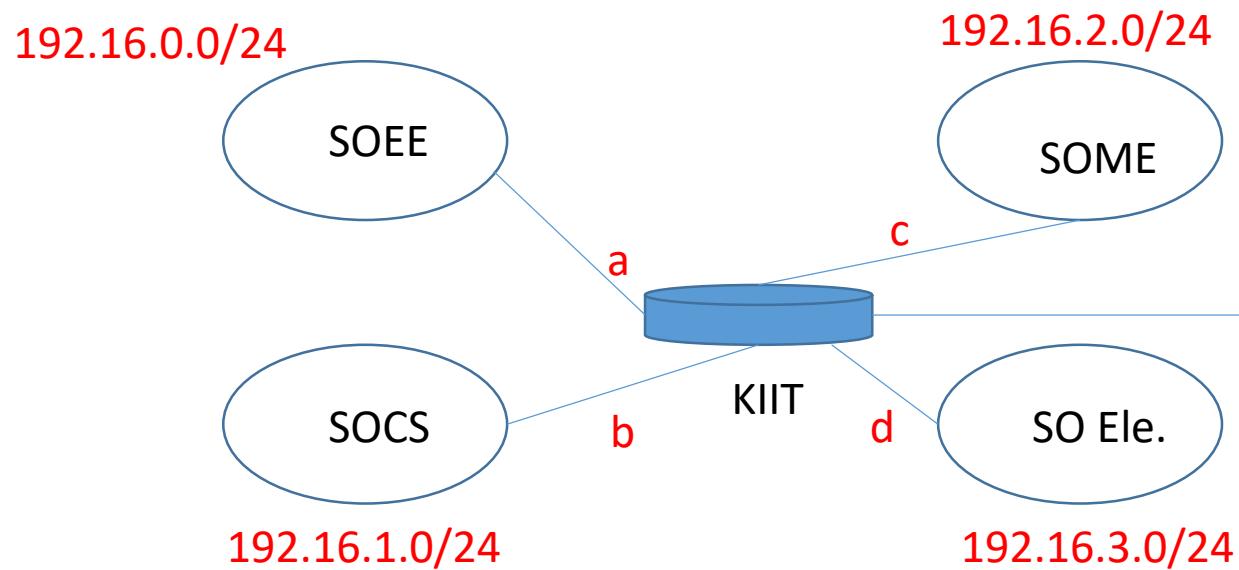
- a) Mask and Network address for each office.
- b) Pool of IP addresses
- c) First and last address
- d) Broadcast address

**Brain Exercise:** Repeat the this Homework if you don't wish to choose 162.12.0.0 as an IP address for any of the subnetworks and observe the number of unused addresses .

# Give me a Solution !!!!!!!

- Assume, your friend is sending a message from Alaska to you in Bhubaneswar using Internet. Can you guess how many entries will be there in routing table to which your friend is connected?

# Supernetting or aggregation



Routing Table at KIIT Router

Network address	Mask	Interface
192.16.0.0/24	255.255.255.0	a
192.16.1.0/24	255.255.255.0	b
192.16.2.0/24	255.255.255.0	c
192.16.3.0/24	255.255.255.0	d



Routing Table at Bhubaneswar Router

Network address	Mask	Interface
192.16.0.0/24	255.255.255.0	a
192.16.1.0/24	255.255.255.0	a
192.16.2.0/24	255.255.255.0	a
192.16.3.0/24	255.255.255.0	a
<b>For Other Part</b>		<b>The world</b>

**Note:** From the routing table at Bhubaneswar, it is clear that it will have all information of routing table at KIIT along with routing information of other part of the world.

Thus we need supernetting to reduce the burden of routers by combining several small networks into a super network.

### **Key point for Supernet**

- It has to follow the CIDR restrictions.
- Networks to be supernetted must be of same size; otherwise choose those which are of same size and repeat.
- **Blind Rule:** Borrow from the network id, thus in supernetting decreases the number of 1s in the mask.

### **Solution:**

**Supernet Mask:** As four networks are to be combined with  $n= 24$  each. Thus for supernet, the value of  $n$  becomes = 22

Thus supernet Mask = **11111111.11111111.11111100.00000000 = 255.255.252.0**

**First Network address:** take any one of the given network address and make  $32-22=10$  right most bits 0.

Thus we get **192.16.0.0/22** as first network address.

**Alternate Solution:** Take any one of the given network address and do 'AND' operation with mask.

$$\begin{array}{r} 192. 16. 2. \quad 0 \\ \text{AND} \quad \underline{255. 255. 252. 0} \\ 192. 16. 0. \quad 0 \end{array}$$

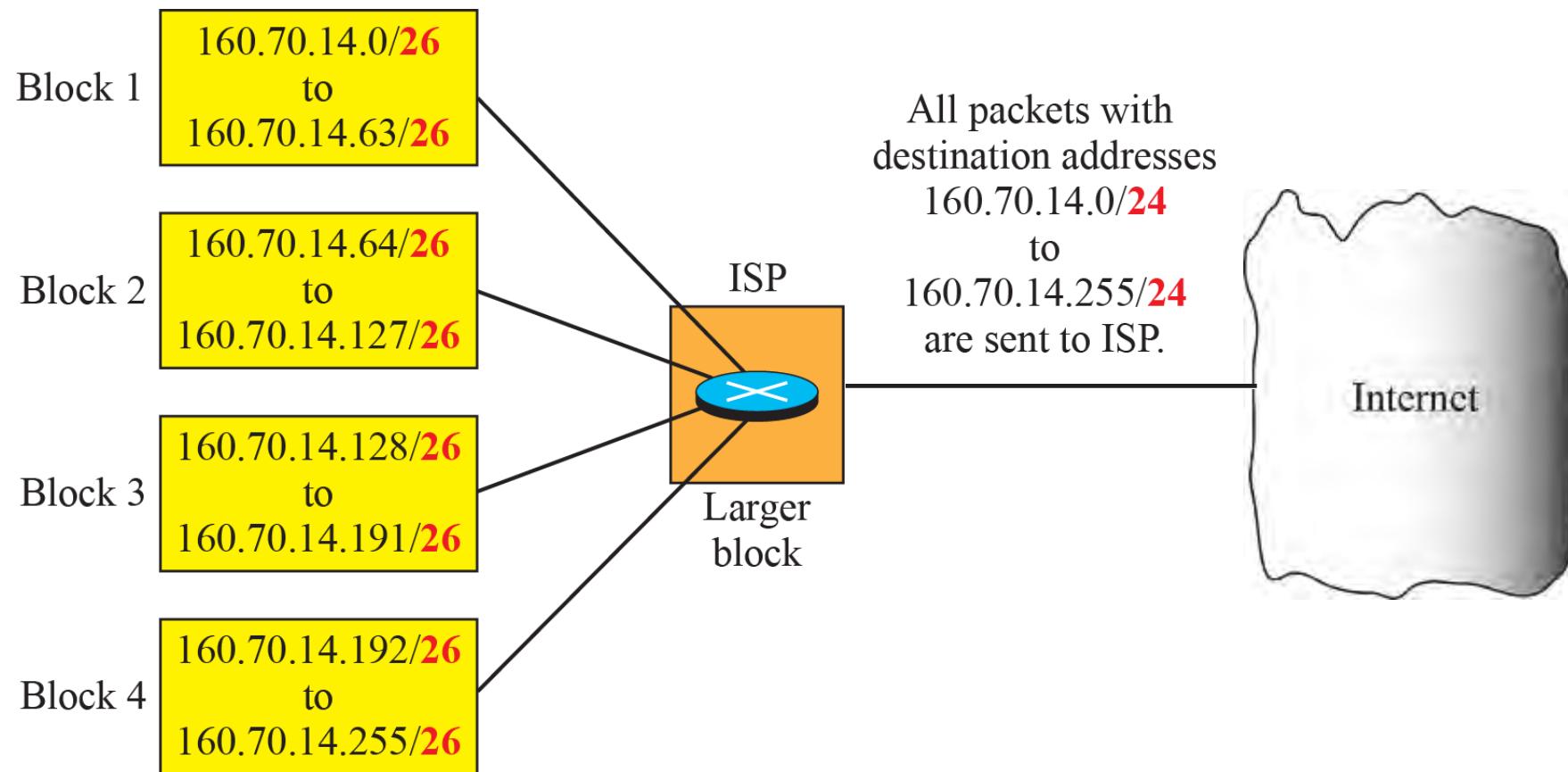
First Network address

## Routing Table at Bhubaneswar Router

Network address	Mask	Interface
192.16.0.0/22	255.255.252.0	a
<b>For Other Part</b>	<b>Of</b>	<b>The world</b>

**Example 4.15:-** Figure 4.37 shows how four small blocks of addresses are assigned to four organizations by an ISP. The ISP combines these four blocks into one single block and advertises the larger block to the rest of the world. Any packet destined for this larger block should be sent to this ISP. It is the responsibility of the ISP to forward the packet to the appropriate organization.

**Analogy:-** This is similar to routing we can find in a postal network. All packages coming from outside a country are sent first to the capital and then distributed to the corresponding destination.



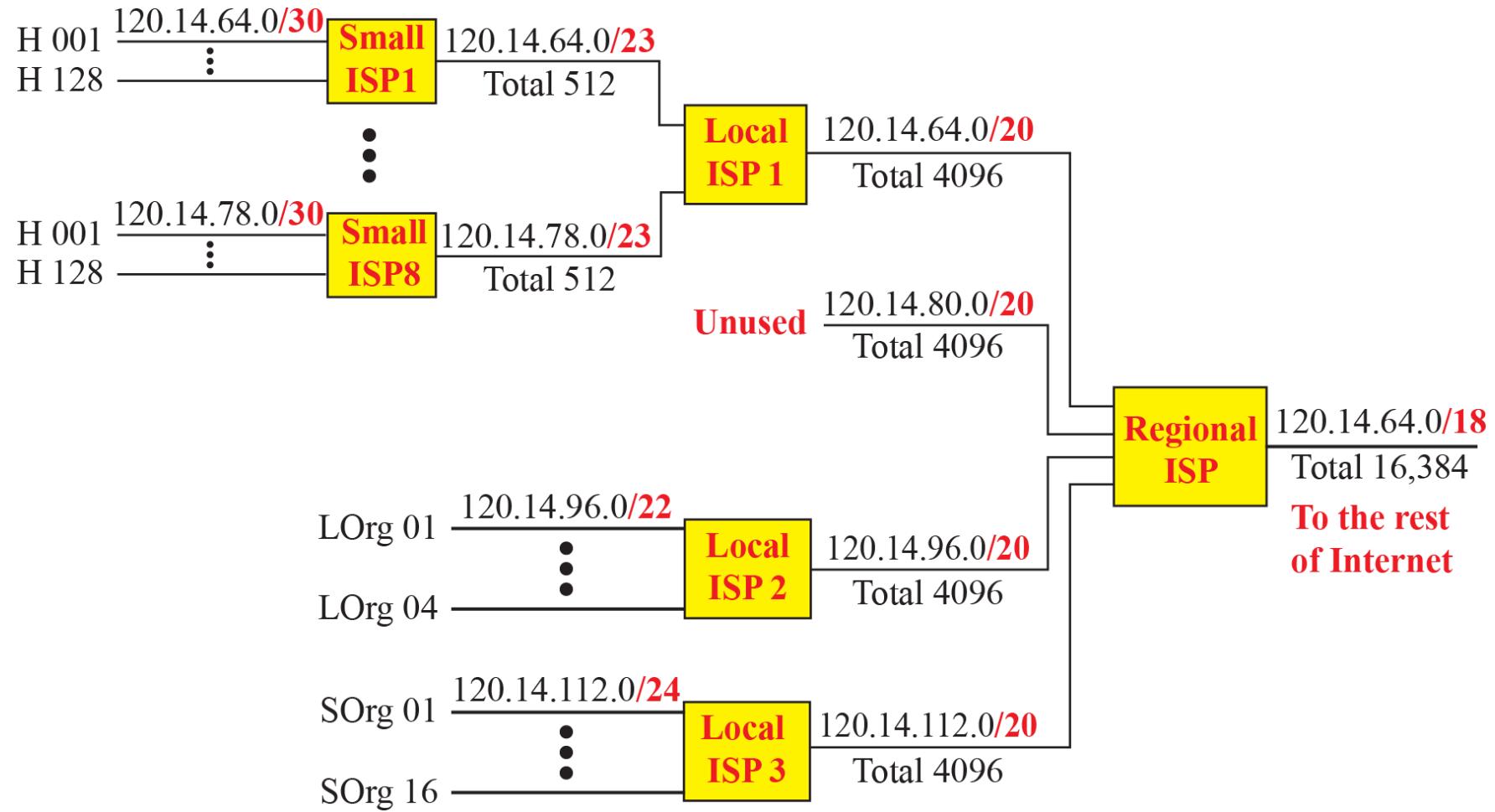
## HYU 4.11:-

- Assume 16 networks with network addresses from 192.16.32.0/24 to 192.16.47.0/24 are connected to router R1 which itself is connected to router R2. Give the routing table of R2.
- **Hint:** Routing table will have only one information in it which is first network address, its mask and interface. You can name interface as per your wish.



**Example 4.16:-** Consider a regional ISP is granted 16,384 addresses, starting from 120.14.64.0/18. The regional ISP has decided to divide these addresses into 4 subblocks, each with 4096 addresses. All these subblocks are assigned to Local ISP namely, *Local ISP1*, *Local ISP2*, *Local ISP3* and *Local ISP4*. The *Local ISP1* has divided its assigned subblock into 8 smaller sub blocks and assigned each to 8 smaller ISP, namely *Small ISP1* to *Small ISP8*. Each Small ISP provides services to 128 households, namely *H001* to *H128*, each using 4 addresses. The *Local ISP3* has divided its block into 4 blocks and has assigned addresses to 4 large organizations, namely *LOrg1* to *LOrg4*, with 1024 addresses in each. The *Local ISP4* has divided its block into 16 blocks and has assigned each block to 16 small organizations, namely *SOrg1* to *SOrg16*, with 256 addresses in each. Design the hierarchy of addressing using the concept of VLSM. Clearly state the assumptions, if any.

**Solution:-** refer next slide



# CN (IT-3001)

## **Network Layer: Protocols Associated to Network Layer**

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose. Images portrayed (if any) are not to hurt the sentiments of any person.

# Objectives

- The objective of this module is to discuss following concepts...
  1. NAT
  2. DHCP
  3. ARP
  4. ICMP
  5. IPv6

# To be discussed....

- Network Address Translation (NAT)
- Methods to allocate the IP address
  - Static
  - Dynamic → DHCP
- Address Mapping
  - From logical address to physical address
  - From Physical address to logical address
- Error reporting
  - ICMP ..... Unicast application
  - IGMP ..... Multicast application

# Special addresses from the set of IP Addresses

- The following groups has special IP addresses for special purpose:
- **This-host Address:** The only address in the block **0.0.0.0/32** is called *this-host* address. It is used whenever a host needs to send an IP datagram but it does not know its own address to use as the source address.
- **Limited-broadcast Address:** The only address in the block **255.255.255.255/32** is called *limited-broadcast* address. It is used whenever a router or a host needs to send an IP datagram to all devices on a network. The routers in a network, however, block the packet having this address as the destination; the packet can not travel outside the network.
- **Loopback Address:** The block **127.0.0.0/8** (**available ranges are: 127.0.0.0/8 to 127.255.255.255/8**) is called the *loopback* address. A packet with one of the addresses in this block as the destination address never leaves the host; it will remain in the host. Any address in this block is used to test a piece of software in the machine. For e.g., we can write a client and server program in which one of the addresses in the block is used as the server address. We can test the programs using the same host to see if they work before running them on different computers. Typically, we use 127.0.0.178 as loop back IP address. Thus, 16, 777, 215 IP wasted.
- **Private Address:** Four blocks are assigned as private addresses as shown in table below. The significance of this is discussed later in NAT.
- **Multicast Address:** The block **224.0.0.0/4** is reserved for multicast addresses to be discussed later on.

# Network Address Translation (NAT)

- Let us say, you have been given an IP address by ISP. But now you want to use 5 hosts to connect to the Internet using only one IP address. How is this possible?

# Network Address Translation (NAT)

- This is possible using the concept of NAT. So, to do this, it must be ensured that router must support NAT software.

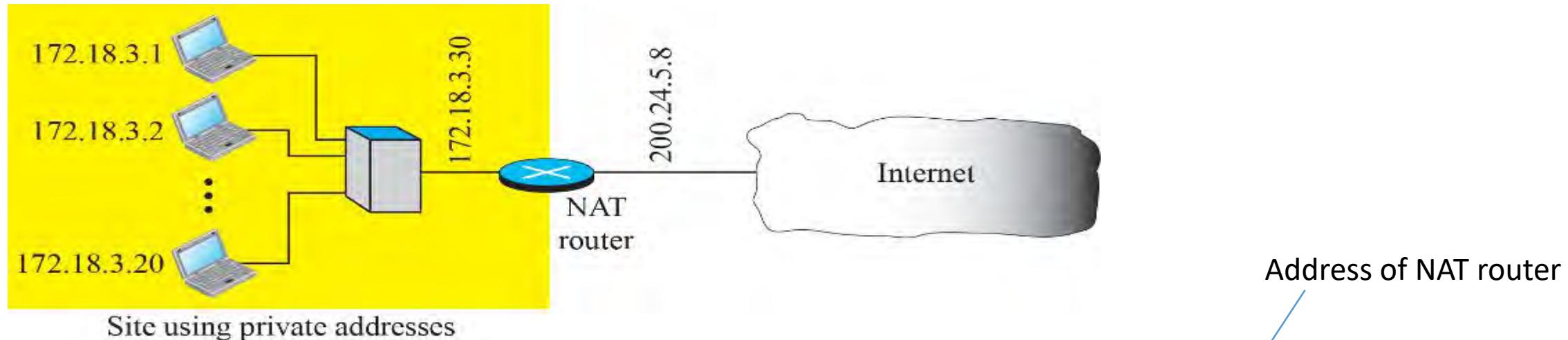
**NAT:** It enables a user to have a larger set of addresses internally and one address or a small set of addresses externally.

- To separate the addresses used inside the home or business and the ones used for the Internet, the Internet authorities have reserved three set of addresses as a private address which are unique inside the organization but not unique globally.
- *The list of private IP addresses are summarized below:*

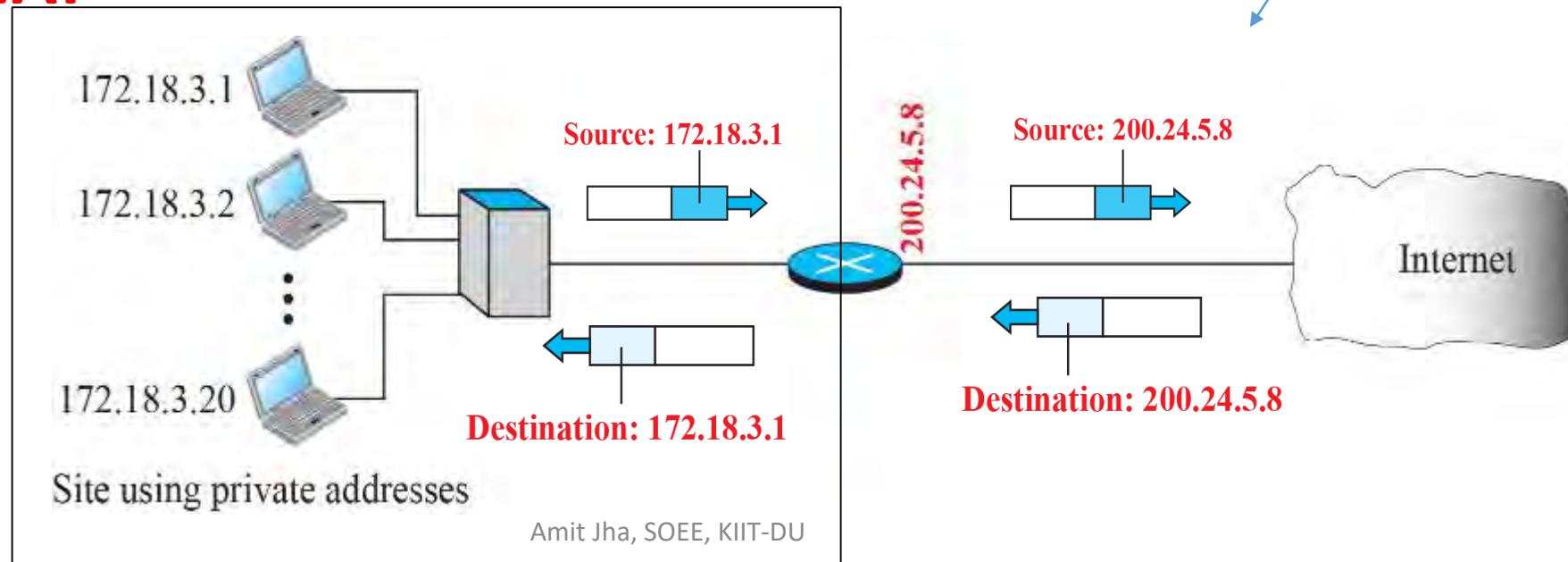
Short Representation	Range	Total
10.0.0.0/8	10.0.0.0 to 10.255.255.255	$2^{24}$
172.16.0.0/12	172.16.0.0 to 172.31.255.255	$2^{20}$
192.168.0.0/16	192.168.0.0 to 192.168.255.255	$2^{16}$

# A NAT implementation

*Note: Private IP addresses are non routable.*



## Address in a NAT



# Time to Think

- **Question:** Who does provide the IP addresses to larger organizations?
- **Answer:** ICANN → Internet Corporation for Assigned Names and Numbers.
- **Question:** Who does provide the IP addresses to smaller organizations?
- **Answer:** ISP → Internet Service Provider
- **Question:** After assigning IP addresses to the organizations, who will manage these IP addresses?
  - **Answer:** Network administrator of the corresponding organization.
- **Question:** How does the network administrator distribute these IP addresses?
  - **Answer:** Two Methods → 1) Static or 2) Dynamic



# Dynamic Host Configuration Protocol (DHCP)

- This is used to dynamically assign the IP addresses in an organization.
- The Dynamic Host Configuration Protocol is used for this purpose.
- DHCP is an application layer program based on client-server paradigm which actually helps the TCP/IP at the Network layer.
- A network manager can configure DHCP to assign IP addresses dynamically which can be either permanent or temporary.
- It also allows an ISP with 1000 granted addresses to provide services to 4000 households, assuming not more than  $1/4^{\text{th}}$  of customers use the Internet at the same time.
- The DHCP can be used to provide following piece of information to the host → the computer address, prefix, the address of a router and the IP address of a name server.

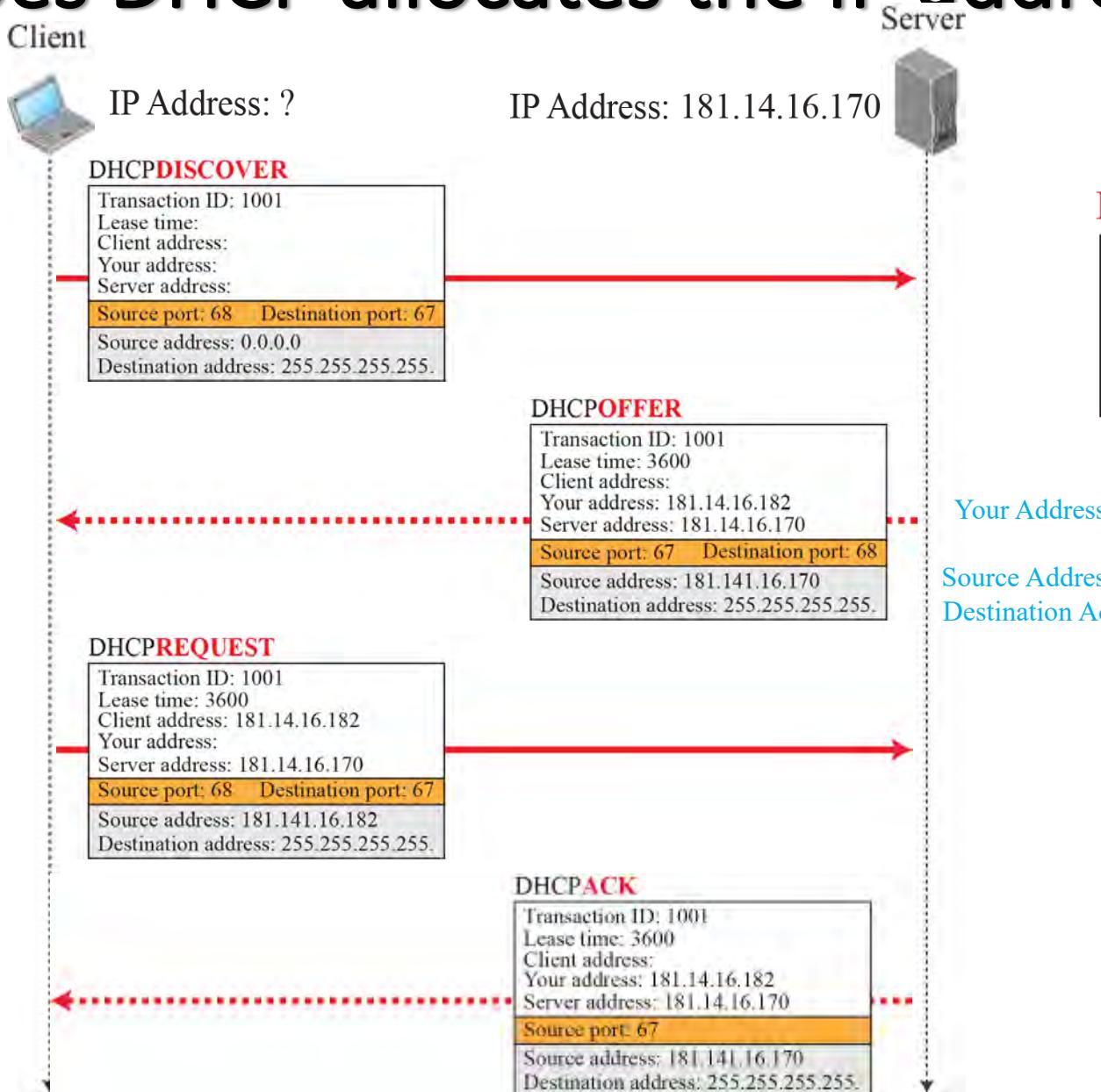
# How does DHCP allocates the IP address?

**Note:**  
Only partial information  
is given.

Transaction id → Random

Source Address → “this host”

Destination Address → “broadcast”



Your Address → “Offered IP Address”

Source Address → “DHCP Server IP Address”

Destination Address → “Broadcast Address”

# **HYU: Answer the following Questions shortly.**

12. What should be the transaction id in DHCPDISCOVER message?
13. Which of the available protocols of the transport layer have been utilized in the previous example?
14. Why does the DHCPOFFER message contains the Destination address as a broadcast address in spite of offering the IP address to the Host?
15. What is the need of DHCPREQUEST message as it knows the IP address offered to it on the reception of DHCPOFFER mesage?
16. Why does the DHCPREQUEST message contains the Destination address as a broadcast address in spite of offering the IP address to the Host?
17. What is the need of DHCPACK message?
18. Do we need DHCPNACK message, and if yes, then what case?
19. Why do we need two well known ports as we know for client, one can use ephemeral port?
20. As DHCP uses UDP, which is unreliable, thus how does DHCP provide error control?

# Address Mapping

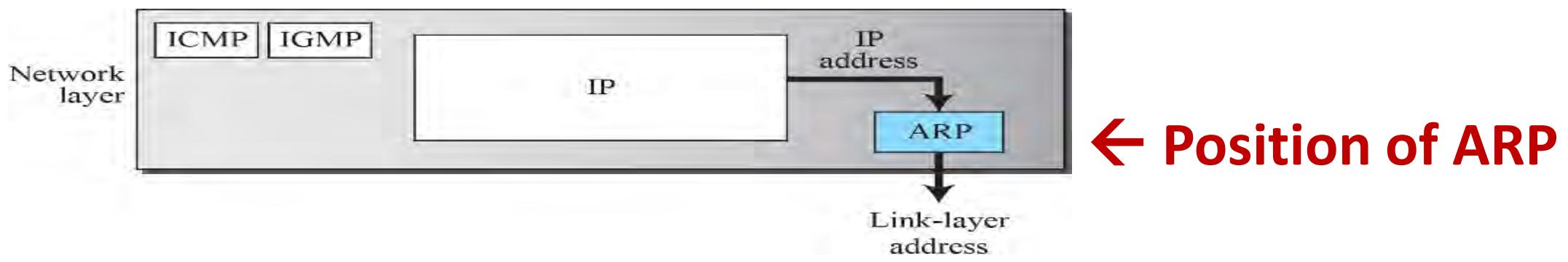
- **Motivation:** The hosts and routers are recognized **at the network level** by their logical (IP) addresses.
- However, packets pass through physical networks to reach these hosts and routers.
- At the physical level, the hosts and routers are recognized by their **physical addresses**.
- This means that delivery of a packet to a host or a router requires two levels of addressing: **logical and physical**.
- Thus, we need to be able to map a logical address to its corresponding physical address and vice versa. These can be done by using either static or dynamic mapping.

# Static Address Mapping

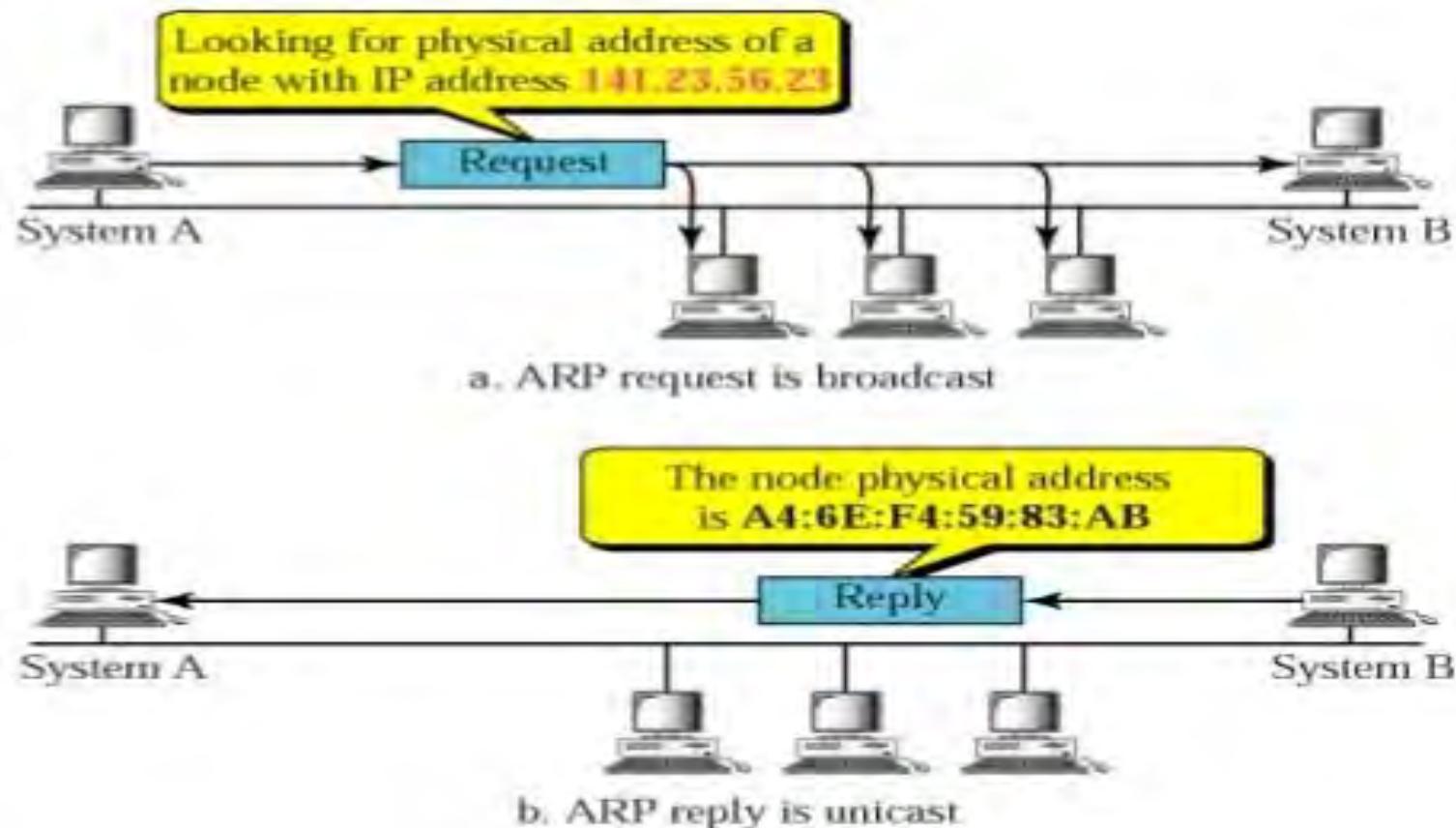
- Static mapping involves in the creation of a table that associates a logical address with a physical address.
- But this method is not feasible because physical addresses may change in the following ways:
  - A machine could change its NIC, resulting in a new physical address.
  - In some LANs, such as Local Talk, the physical address changes every time the computer is turned on.
- To implement these changes, a static mapping table must be updated periodically. This overhead could affect network performance.
- In dynamic address mapping, each time a machine knows one of the two addresses (logical or physical), it can use **a protocol** to find the other one.

# Mapping Logical to Physical Address: ARP

- ARP stands for Address Resolution Protocol.
- It is done by broadcasting **ARP query packet** containing physical and logical addresses of sender and **only logical address of receiver**.
- Every host or router on the network receives and processes the ARP query packet, but only the intended recipient **recognizes its IP address** and sends back an **ARP response packet**.
- The response packet contains the recipient's IP and physical addresses.



# ARP Operation



# Packet Format of ARP

- **Hardware Type:** It defines the type of link-layer protocol. For Ethernet, this value is 1.
- **Protocol Type:** It defines the network-layer protocol. For IPv4 protocol, this value is (0800)16.
- **Hardware Length:** It defines the length of the hardware address in bytes. For MAC address, the value is 6 bytes.
- **Protocol Length:** It defines the length of the protocol address in bytes. For IPv4 address, the value is 4 bytes.
- **Source Hardware Address:** It defines the link-layer address i.e., MAC address of the source.
- **Source Protocol Address:** It defines the Network-layer address i.e., IP address of the source.
- **Destination Hardware Address:** It defines the link-layer address i.e., MAC address of the destination.
- **Destination Protocol Address:** It defines the Network-layer address i.e., IP address of the destination.
- Operation: It defines the nature of ARP message. For request, value is 1; and for reply, value is 2.

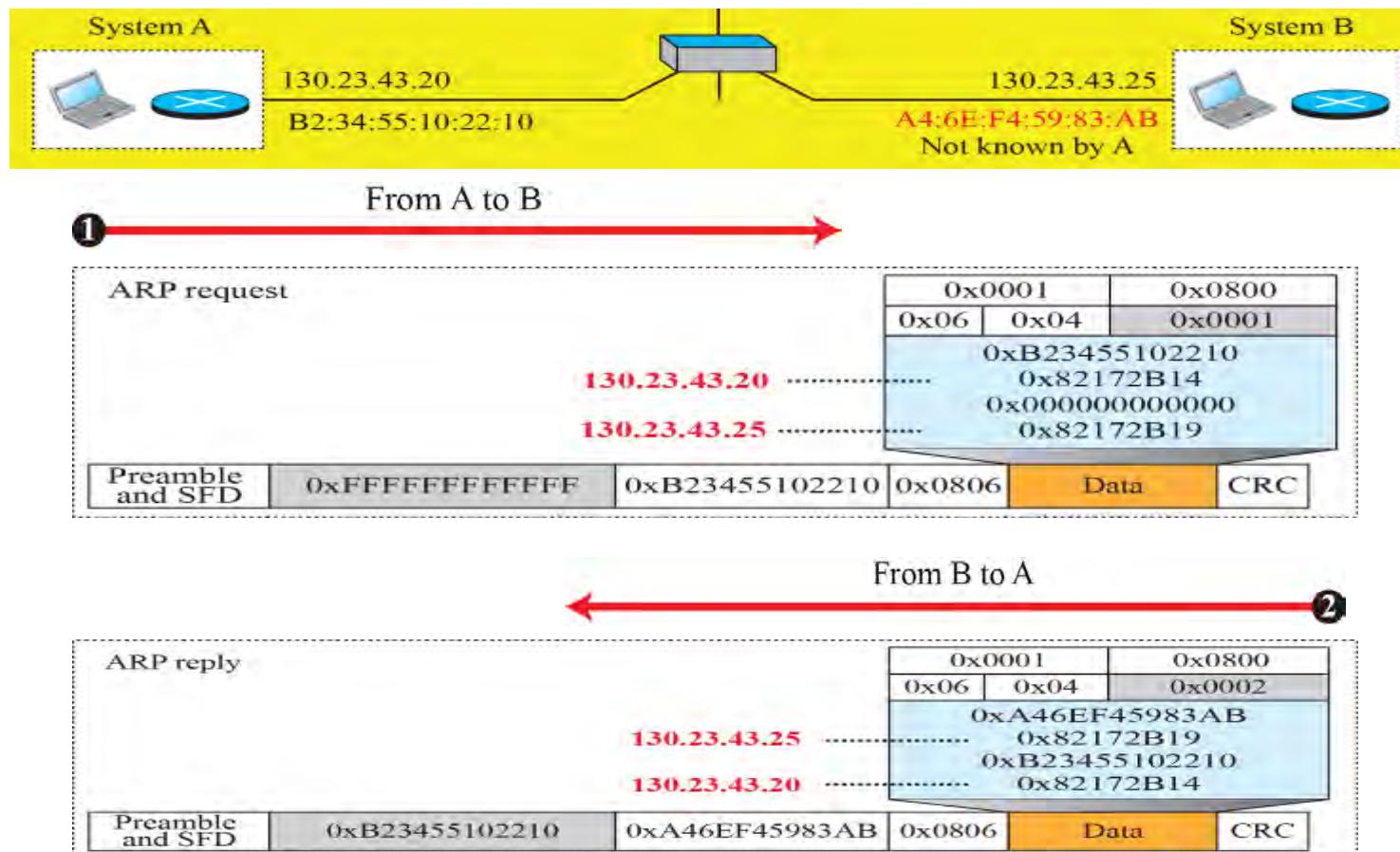
0	8	16	31		
Hardware Type		Protocol Type			
Hardware length	Protocol length	Operation <b>Request:1, Reply:2</b>			
Source hardware address					
Source protocol address					
Destination hardware address (Empty in request)					
Destination protocol address					

**Hardware:** LAN or WAN protocol

**Protocol:** Network-layer protocol

**Example 4.17:** A host with IP address N<sub>1</sub>= 130.23.43.20 and MAC address L<sub>1</sub>= B2: 34: 55: 10: 22:10 has a packet to send to another host with IP address N<sub>2</sub>= 130.23.43.25 and physical address L<sub>2</sub> = A4: 6E: F4: 59: 83: AB (which is unknown to the first host). The two hosts are on the same network. Show the ARP request and reply packets encapsulated in Ethernet frames.

**Solution:-** Figure below shows the ARP request and reply packets. Note that the ARP data field in this case is 28 bytes, and that the individual addresses do not fit in the 4-byte boundary. That is why we do not show the regular 4-byte boundaries for these addresses. Also note that the IP addresses are shown in hexadecimal.



- For more details of ARP, you can refer Section 5.4, page no-421-427 of the Text Book.

# Mapping Physical to Logical Address: RARP

- **Need:** There are occasions in which a host knows its physical address, but needs to know its logical address. This may happen in two cases:
  1. **A disk less station is just booted.** The station can find its physical address by checking its interface, but it does not know its IP address.
  2. An organization does not have enough IP addresses to assign to each station; it needs to assign IP addresses on demand. The station can send its physical address and ask for a short time lease.

Three protocols are used for finding out logical address from known physical address: **RARP, BOOTP, DHCP.**

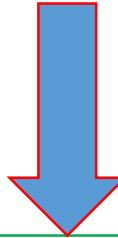
# Internet Control Message Protocol (ICMP)

- **Need:** The IP protocol is a best-effort delivery service i.e., it provides **unreliable** and **connectionless** datagram delivery.

It was designed to make an **efficient** use of network resources.

However, it has ***two deficiencies:***

1. Lack of error control & 2. Lack of assistance mechanisms.



## What happens if

- Something goes wrong?
- A router discards a datagram because TTL field has 0?
- Final destination host discards all fragments of a data gram because it has not received all fragments within a predetermined time?



- A host sometimes needs to determine if a router or another host is alive.
- And sometimes a network administrator needs information From another host or router

# ICMP or ICMPv4: Types of Messages

- ICMP messages are divided into two broad categories: **error-reporting messages** and **query messages**.
- **Error Reporting:** ICMP does **not correct errors**, it simply **reports error message** to the original source. This is because the only information available in the datagram about the route is source and destination IP addresses.

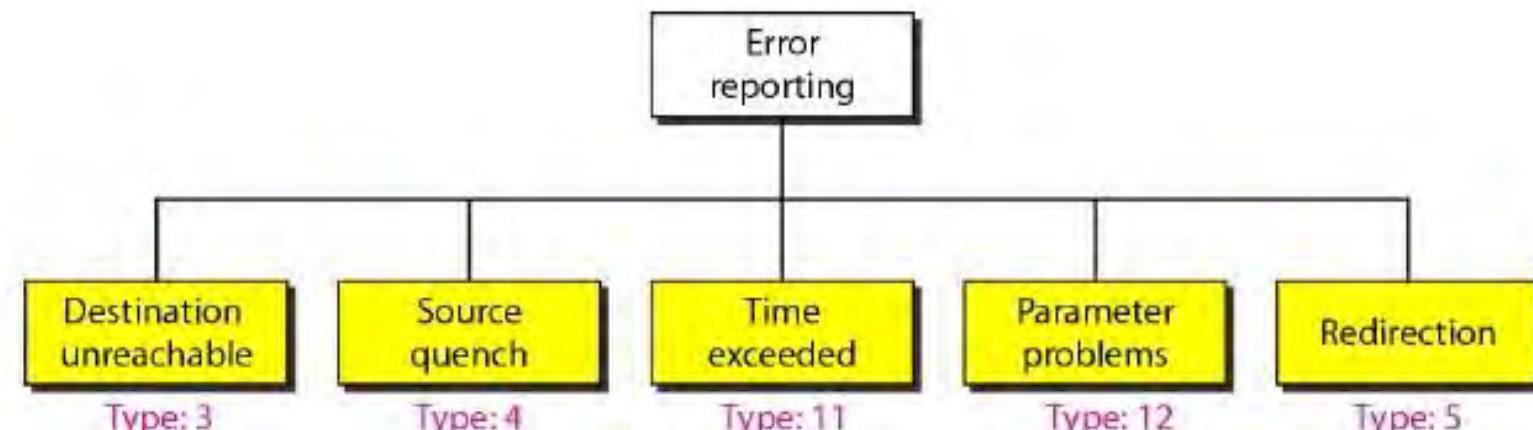


Fig: Error Reporting Message

Cont....

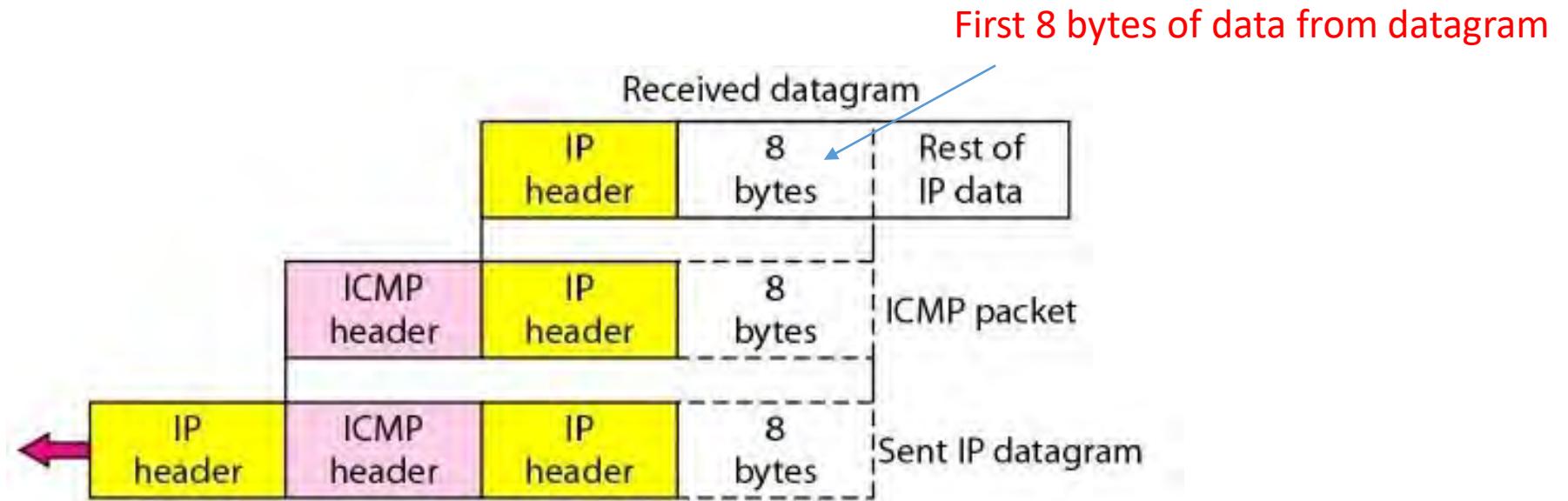


Fig: Contents of data field for the error messages

- **Note:** All error messages contain a data section that includes the IP header of the original datagram plus the **first 8 bytes of data in that datagram**.
- The original datagram header is added to give the original source, which receives the error message, information about the datagram itself. The 8 bytes of data are included because the first 8 bytes provide information about the **port numbers (UDP and TCP)** and **sequence number (TCP)**. This information is needed so the source can inform the protocols (TCP or UDP) about the error.

# Types of ICMP Error Reporting Message

1. ***Destination Unreachable***: This message is created when a router can not route a datagram or a host can not deliver a datagram.
2. ***Source Quench***: The source-quench message in ICMP was designed to add a kind of flow control to the IP. When a router or host discards a datagram due to congestion, it sends a source-quench message to the sender of the datagram.
3. ***Time Exceeded***: This message is created when datagram can not reach to the destination because of either congestion or travelling in a loop or cycle endlessly.

Cont...

4. **Parameter Problem:** If a router or the destination host discovers an ambiguous or missing value in any field of the datagram, it discards the datagram and sends a parameter-problem message back to the source.
5. **Redirection:** The hosts usually use static routing and its routing table has a limited number of entries. It usually knows the IP address of only one router, the default router. For this reason, the host may send a datagram, which is destined for another network, to the wrong router. In this case, the router that receives the datagram will forward the datagram to the correct router. However, to update the routing table of the host, it sends a redirection message to the host.

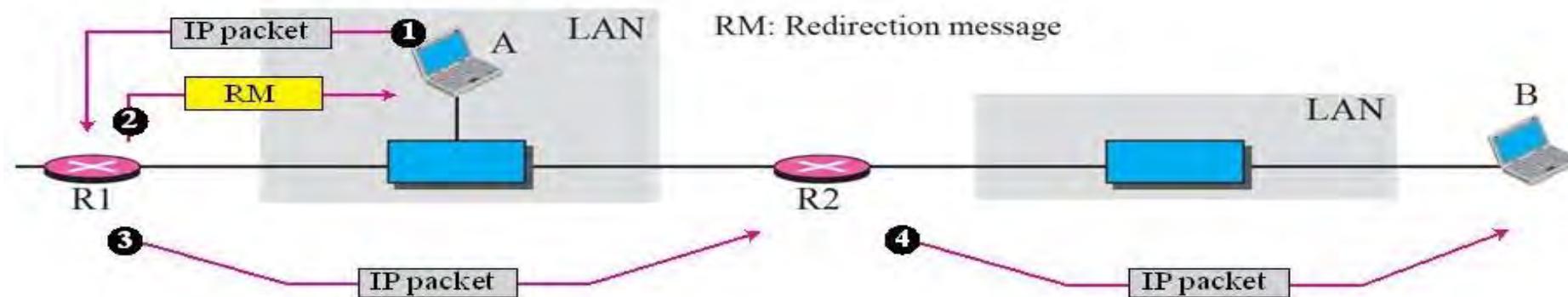
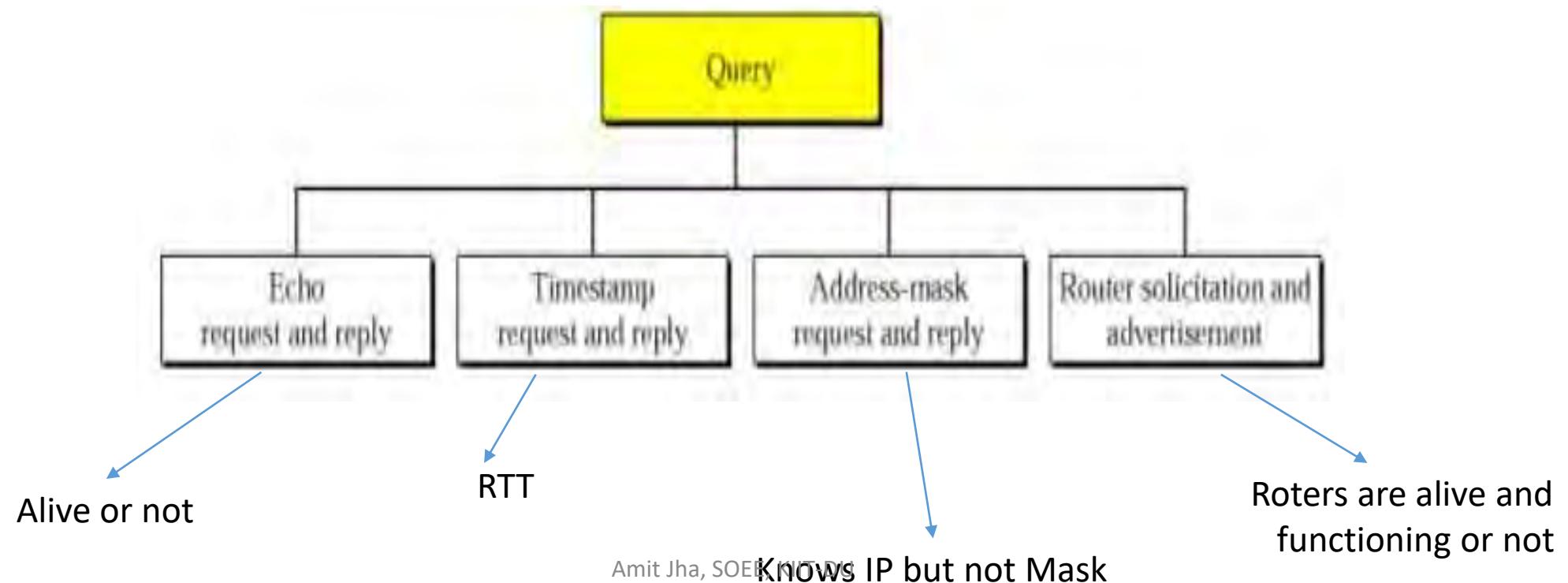


Fig: Redirection concept

# Types of ICMP Query Message

- In addition to error reporting, ICMP can diagnose some network problems using the *query messages*.



# Cont...

1. ***Echo Request and Reply:*** The combination of echo-request and echo reply messages determines whether two systems (hosts or routers) can communicate with each other at IP level or not. **Ping** command can create a series (instead of just one) of echo-request and echo reply messages, providing statistical information.
2. ***Time stamp Request and Reply:*** This is used to determine the round-trip time needed for an IP datagram to travel between two hosts or routers.
3. ***Address-Mask Request and Reply:*** This message is sent to the corresponding LAN when a host knows its IP address but does not know its mask.

# Cont...

**4. Router Solicitation and Advertisement:** When a host wants to communicate with a host on other network, it must know the address of routers connected to its own network. Also, the host must know if the routers are alive and functioning. The router-solicitation and router-advertisement messages can help in this situation. A host can broadcast(or multicast) a router-solicitation message. The router or routers that receive the solicitation message broadcast not only their own routing information but also information of other routers it is aware of, using the router-advertisement message.

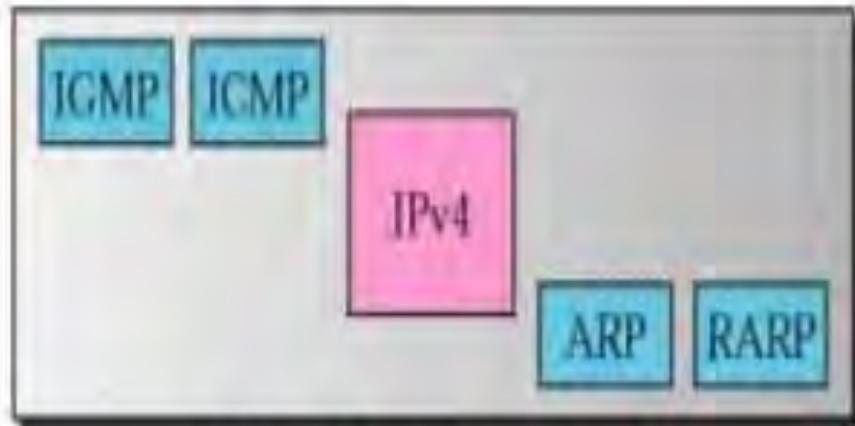
**HYU 21:** Explore the use of *ping* and *tracert* from the point of network administrator.

**Q. Will ICMP help if we want to send error report and query (network management) to more than one routers i.e. group management?**

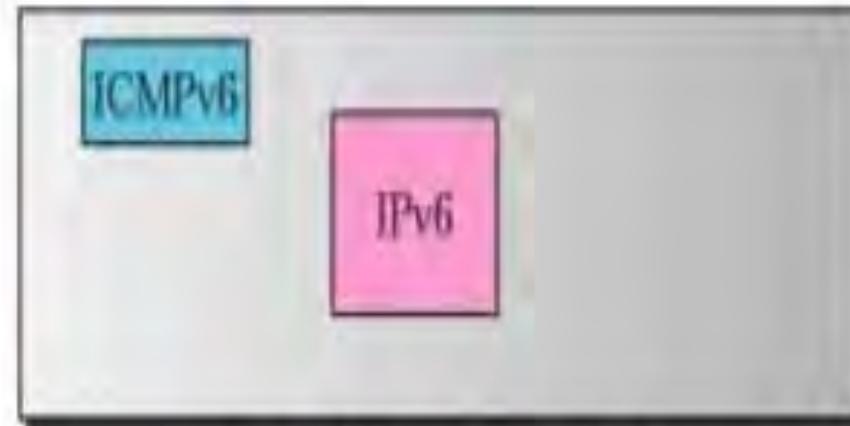
**Q. Will ICMP help if we want to send error report and query (network management) to more than one routers i.e. group management?**

**Ans:** No, it won't work. This is because ICMP was designed for unicast application. For multicast application we need to use Internet Group Management Protocol (IGMP)

**Note:** IGMP is not in syllabus. If you wish then you can refer to Forouzan, P.N 630, 4<sup>th</sup> edition.



Network layer in version 4



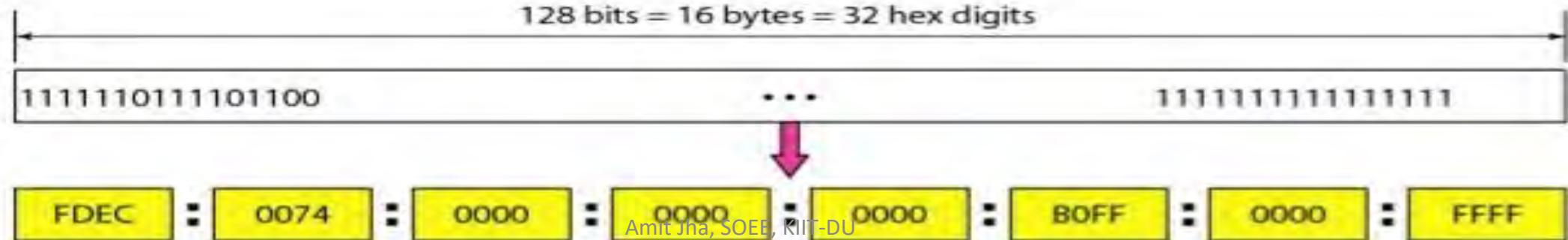
Network layer in version 6

**Fig:** Comparison of network layers in version 4 and version 6.

- In IPv6, ICMPv4 has been modified as ICMPv6 to make it more suitable for IPv6.
- ICMPv6 not in syllabus. If you wish then you can refer to Forouzan P.N 638, 4<sup>th</sup> edition.

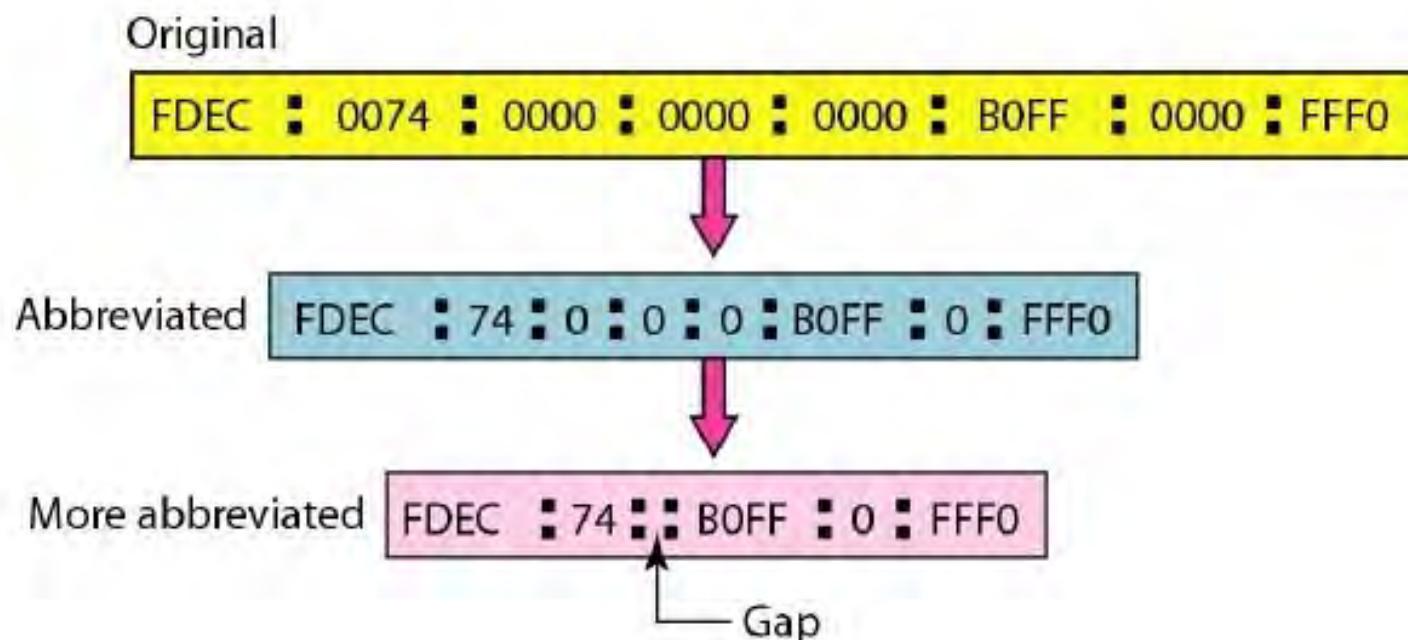
# Internet Protocol version 6 (IPv6)

- **Need/Motivation:**
  - Despite all **short-term solutions**, such as **classless addressing**, Dynamic Host Configuration Protocol (**DHCP**), and **NAT**, address depletion is still a long-term problem for the Internet.
  - For **real-time audio and video transmission**, **minimum delay** strategies and **reservation of resources** are required in transmission which were not provided in the IPv4 design.
  - The Internet must accommodate **encryption and authentication** of data for some applications. No encryption or authentication is provided by IPv4.
- **IPv6 Address Structure:** An IPv6 address is **16 bytes or 128 bits long**.  
This format is known as **colon hexadecimal notation**.



# Abbreviation of IPv6

- Although the IP address, even in hexadecimal format, is very long, many of the digits are zeroes. In this case, we can abbreviate the address.

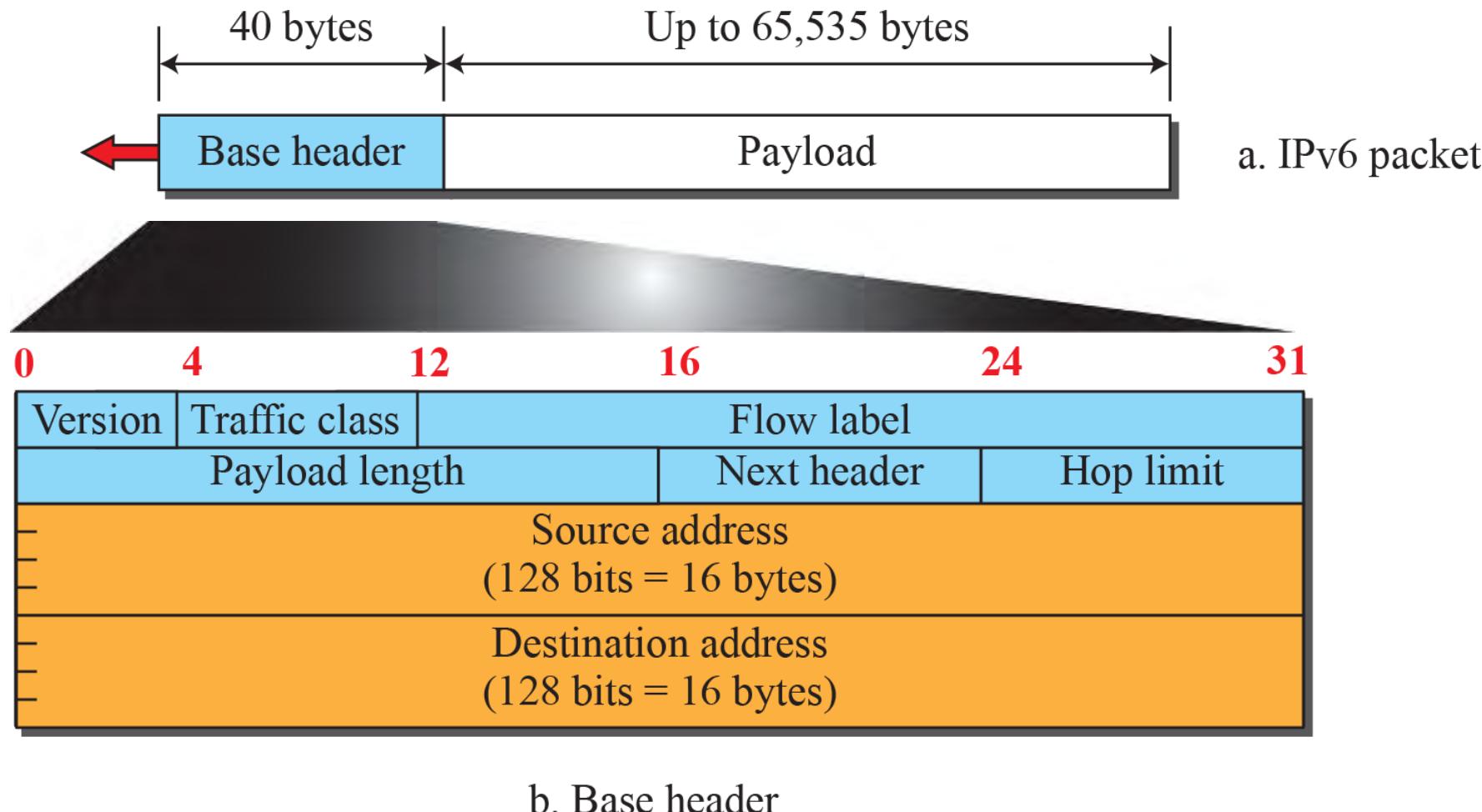


# Advantages of IPv6 over IPv4

1. **Larger Address Space:** 128 bits long, so  $2^{96}$  increase in the address space.
2. **Better Header Format:** Options are separated from the base header. So, speed up in the routing process as options do not need to be checked by the routers most of the time.
3. **New Options:** This field allows more functionalities in IPv6.
4. **Allowance for Extension:** Designed to allow extension of the protocol if desired.
5. **Support for Resource Allocation:** Flow control is added
6. **Support for More Security:** The encryption and authentication options in IPv6 provide confidentiality and integrity of the packet.

# Packet Format of IPv6

- Each packet is composed of a mandatory base **header of 40 bytes** (fixed size) followed by the payload.
- The **payload consists of two parts**: optional extension headers and **data** from an upper layer.

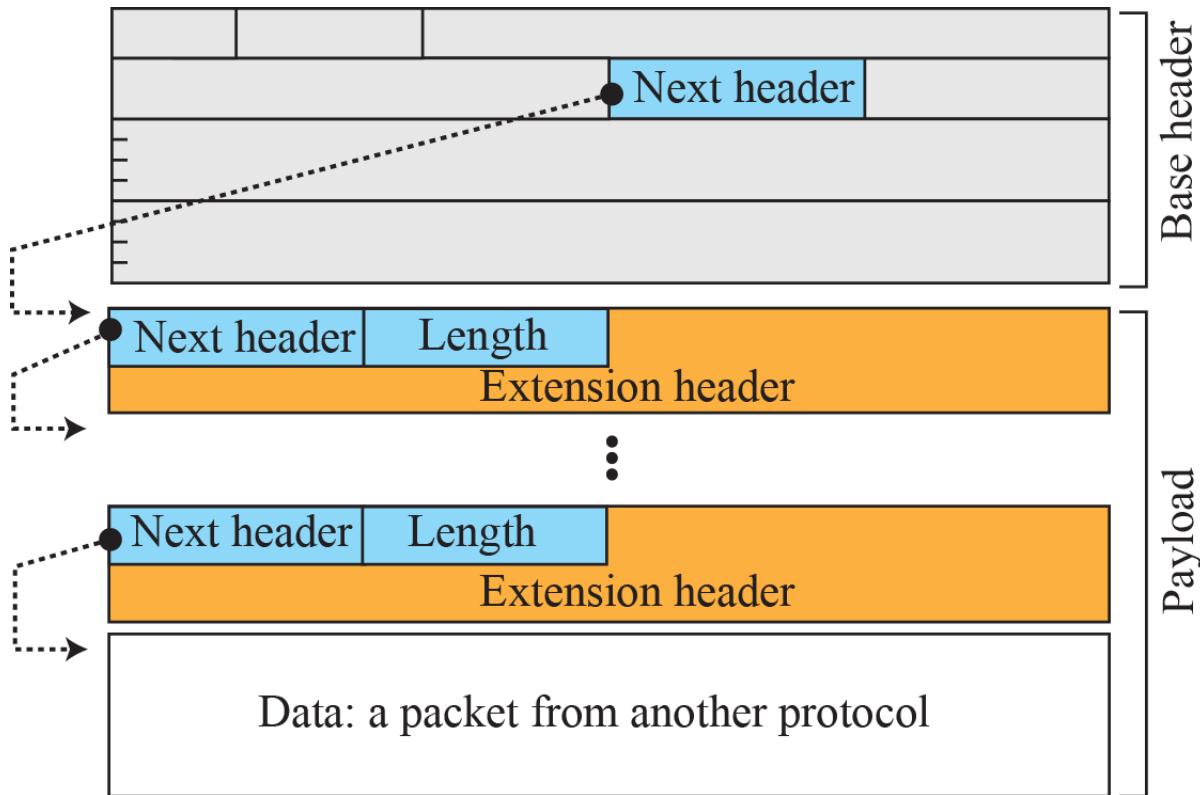


# IPv6 Base Header

- The IPv6 packet, or datagram is shown in diagram below.
- *Version*: To define IPv4 or IPv6
- *Traffic Class or Priority*: identify priority among datagrams
- *Flow Label*: Used for flow control
- *Payload Length*: It defines the size of the IP datagram excluding the header.
- *Next Header*: It defines the type of first extension header (if present) or the type of the data that follows the base header in the datagram. This field is similar to protocol field of IPv4 header.
- *Hop Limit*: It serves the same purpose as the TTL field in IPv4 header.

**Note:** The size of IPv6 base header is 40 bytes.

## Structure of Payload in IPv6 Datagram:



### Some next-header codes

- 00: Hop-by-hop option
- 02: ICMPv6
- 06: TCP
- 17: UDP
- 43: Source-routing option
- 44: Fragmentation option
- 50: Encrypted security payload
- 51: Authentication header
- 59: Null (no next header)
- 60: Destination option

Significance of Extension Header

# Comparison Between IPv4 and IPv6 Headers

Comparison
1. The header length field is eliminated in IPv6 because the length of the header is fixed in this version.
2. The service type field is eliminated in IPv6. The priority and flow label fields together take over the function of the service type field.
3. The total length field is eliminated in IPv6 and replaced by the payload length field.
4. The identification, flag, and offset fields are eliminated from the base header in IPv6. They are included in the fragmentation extension header.
5. The TTL field is called hop limit in IPv6.
6. The protocol field is replaced by the next header field.
7. The header checksum is eliminated because the checksum is provided by upper-layer protocols; it is therefore not needed at this level.
8. The option fields in IPv4 are implemented as extension headers in IPv6.

# CN (IT-3001)

## Network Layer: Routing Algorithm

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose. Images portrayed (if any) are not to hurt the sentiments of any person.

# Objectives

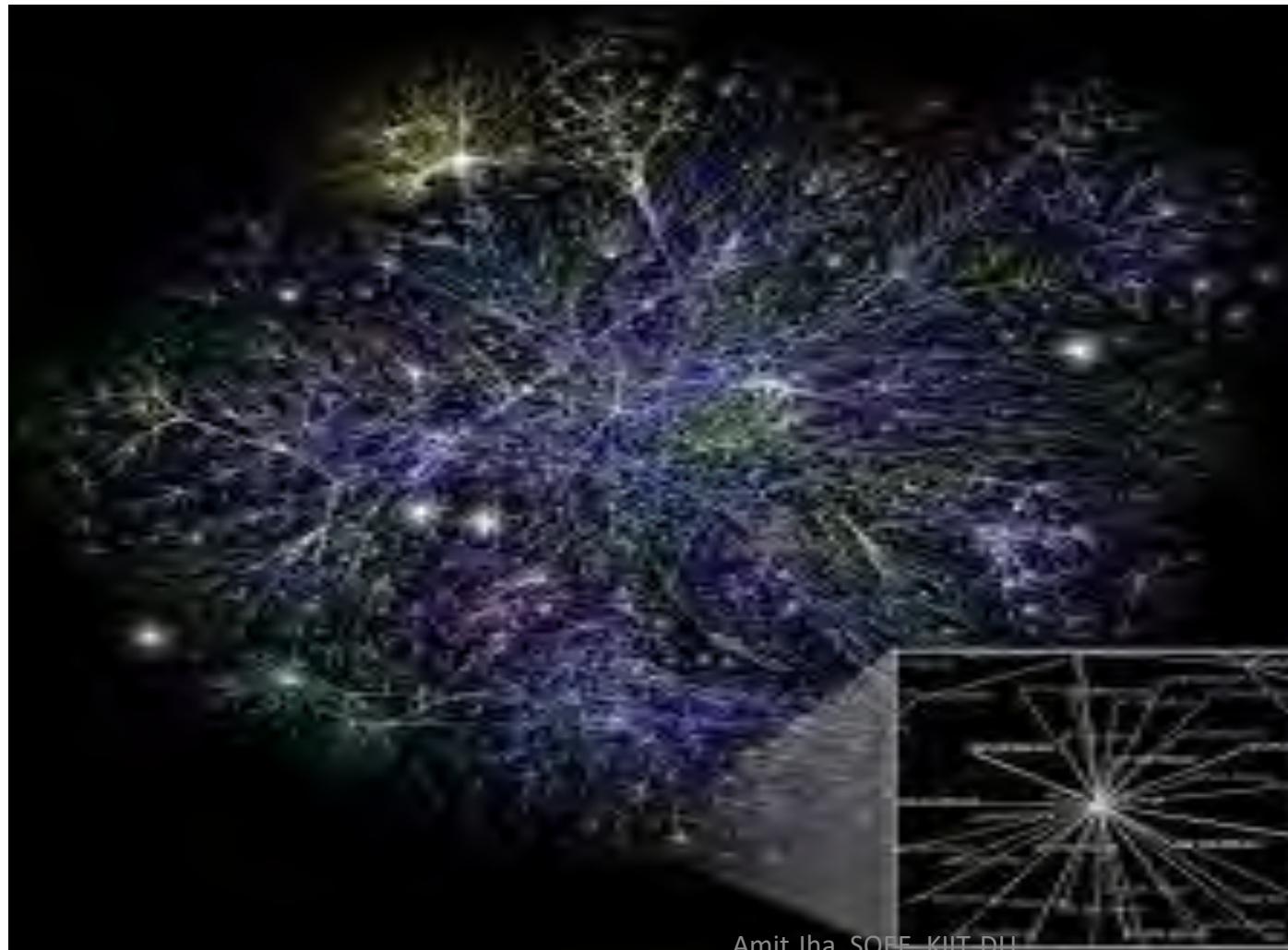
- The objective of this module is to discuss following concepts...
  1. Routing Algorithm

# Delivery, Forwarding, and Routing: A view

- **Delivery:** The handling of the packets by the network layer. If the final destination of the packet is a host connected to the same physical network as the deliverer, it is known as *direct delivery*; otherwise *indirect delivery*.
- **Forwarding:** It means to place the packet in its route to its destination, i.e., to show a path to packets.
- **Routing:** A mechanism to show an **optimum** path for forwarding.



# Why do we need Routing?



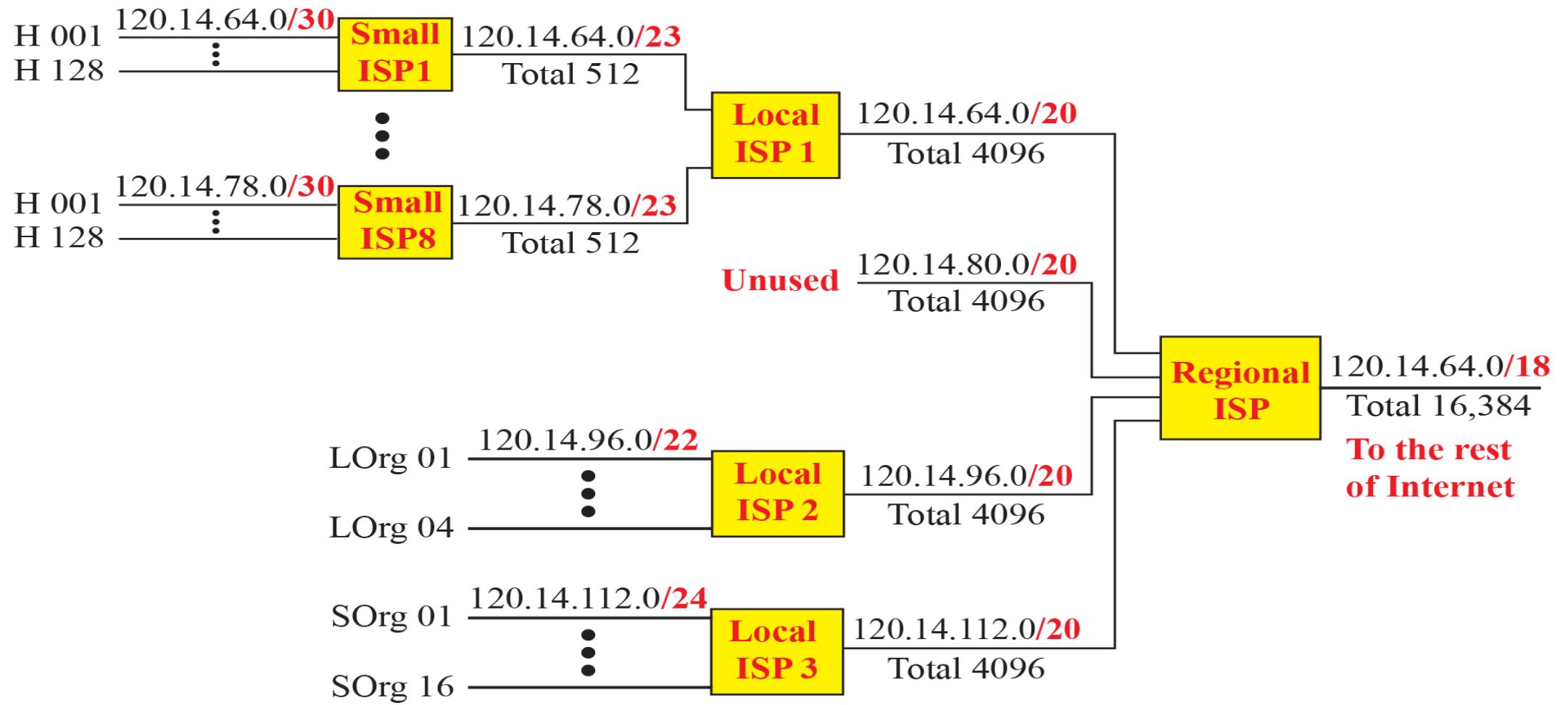
# Forwarding

Let's retake example 4.16 of this module to understand the significance of forwarding.

**Example 4.16:-** Consider a regional ISP is granted 16,384 addresses, starting from 120.14.64.0/18. The regional ISP has decided to divide these addresses into 4 subblocks, each with 4096 addresses. All these subblocks are assigned to Local ISP namely, *Local ISP1*, *Local ISP2*, *Local ISP3* and *Local ISP4*. The *Local ISP1* has divided its assigned subblock into 8 smaller sub blocks and assigned each to 8 smaller ISP, namely *Small ISP1* to *Small ISP8*. Each Small ISP provides services to 128 households, namely *H001* to *H128*, each using 4 addresses. The *Local ISP3* has divided its block into 4 blocks and has assigned addresses to 4 large organizations, namely *LOrg1* to *LOrg4*, with 1024 addresses in each. The *Local ISP4* has divided its block into 16 blocks and has assigned each block to 16 small organizations, namely *SOrg1* to *SOrg16*, with 256 addresses in each. Design the hierarchy of addressing using the concept of VLSM. Clearly state the assumptions, if any.

**Solution:-** refer next slide.

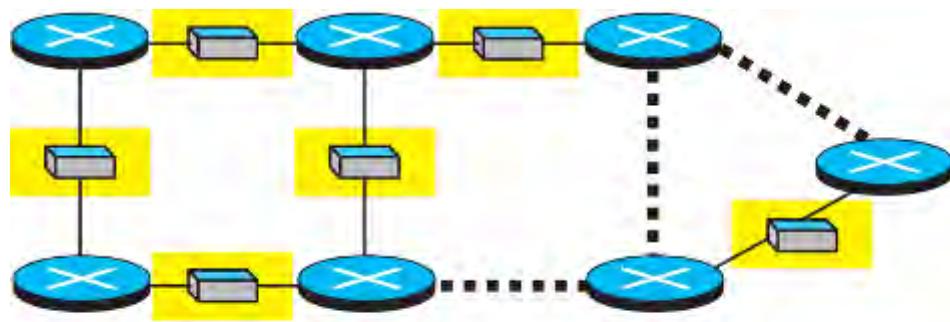
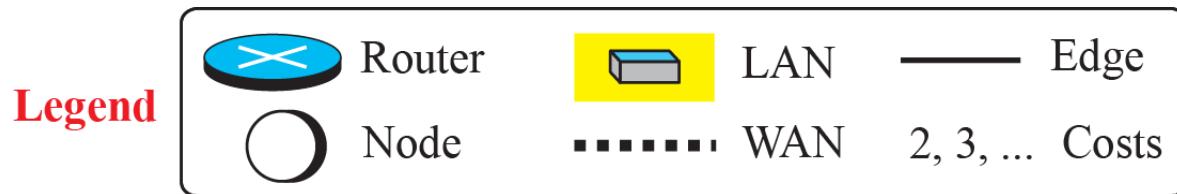
- To be noted,
  - This example of IP address distribution shows the perfect example, where the hierarchical routing has been used.
  - All routers in the Internet send a packet with destination address 120.14.64.0 to 120.14.127.255 to the regional ISP.
  - The regional ISP sends every packet with destination address 120.14.64.0 to 120.14.79.255 to the local ISP1.
  - The Local ISP1 sends every packet with destination address 120.14.64.0 to 120.14.64.3 to the household H001.



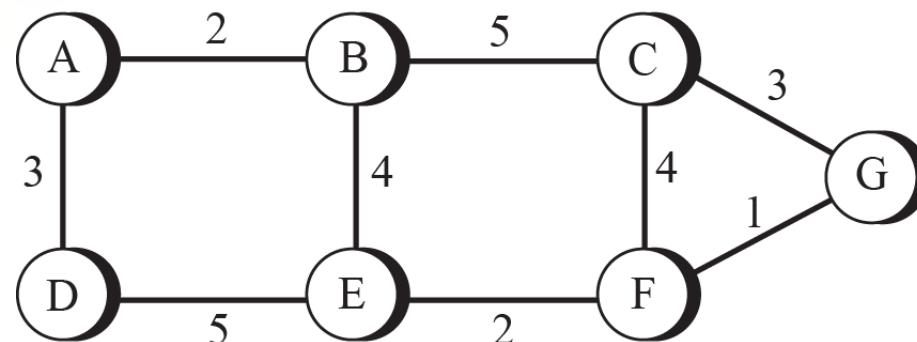
# Basics of Routing:

- The following are the broad classification of routing.
  1. *Unicast Routing*: In this, a datagram is destined for only one destination.
  2. *Multicast Routing*: In this, a datagram is destined for several destination.
  3. *Broadcast Routing*: In this, a datagram is supposed to be delivered to all hosts on the Internet.

# Basics of Routing: An internet and its graphical representation



a. An internet



b. The weighted graph

# Unicast Routing:

- The unicast routing in the Internet, with a large number of routers and a huge number of hosts, can be done only by using hierarchical routing: routing in several steps using different routing algorithm.
- First, we discuss the different routing algorithms and then we will see their implementation in the today's Interenet.

# Routing Table

- It consists of routing information.
- It is of two types:
  1. **Static Routing Table**
  2. **Dynamic Routing Table**
- ***Static Routing Table:*** Routing information is entered and maintained manually by administrator.
- ***Dynamic Routing Table:*** Routing table is updated periodically by using one of the dynamic routing protocols such as RIP, OSPF, or BGP.

**Note:** Despite of having drawback of static routing table, it can be used in small internet that does not change very often or in an experimental internet for troubleshooting.

# Routing Algorithm

- Two most popular routing algorithm:

1) **Bellman-Ford Algorithm:** It is used in **Distance Vector Routing Protocols.**

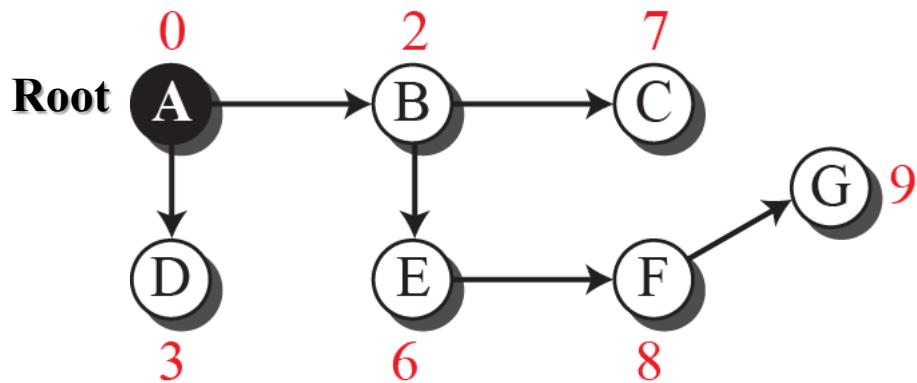
- Neighbours exchange list of distances to destinations.
- Best next hop is determined for each destination.
- Uses distributed approach

2) **Dijkstra's Algorithm:** It is used in **Link State Routing Protocols.**

- Link state information is flooded to all routers.
- Uses centralized approach

**Note:** Both computes the shortest path (i.e., least cost) from a single node vertex.

# What is Distance Vector?



a. Tree for node A

Root → A

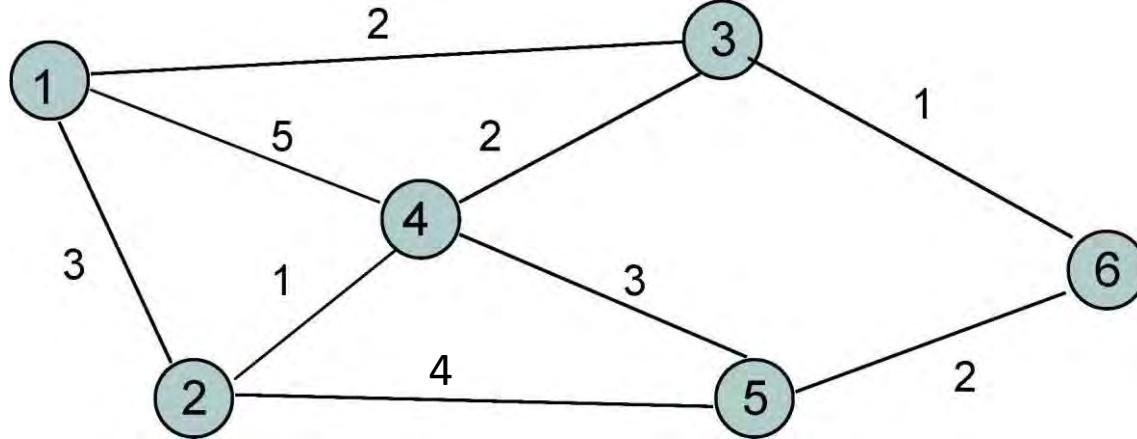
Destination {

A	0
B	2
C	7
D	3
E	6
F	8
G	9

b. Distance vector for node A

# Bellman-Ford (Distance Vector Protocols)

## The simplest approach.....



*Prob: Find the shortest path to 6.*

Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(-1, $\infty$ )				

Annotations:

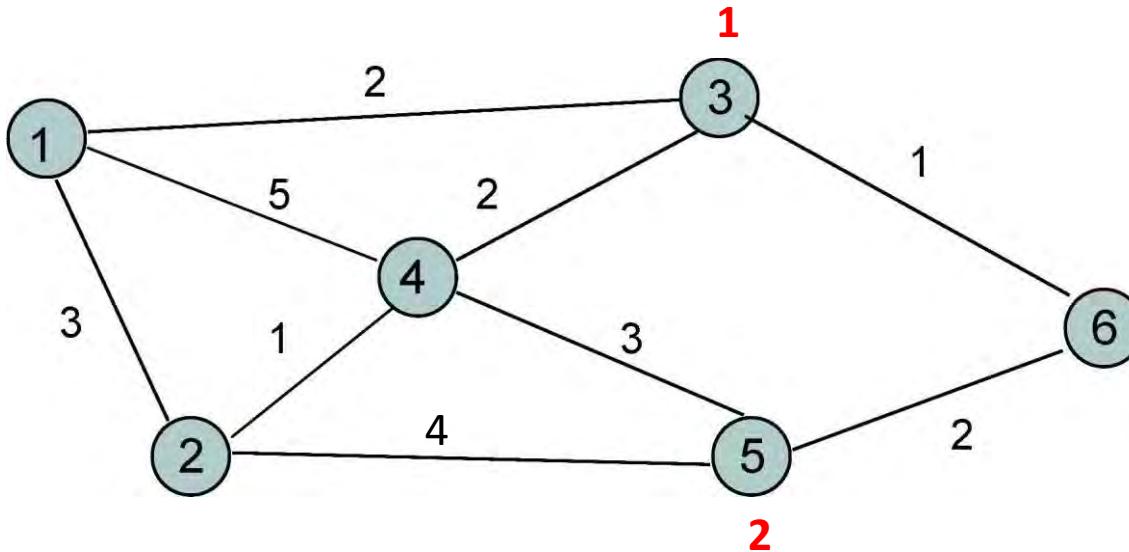
- Next node from Node 6: Points to the 'Node 5' column.
- Distance of node 6 from Node 5: Points to the value  $\infty$  in the 'Node 5' column.

**Blind Rule:** The only question to be asked to yourself at every node  $N_x$  for given destination N:

**Can we reach to node N from node  $N_x$  using the nodes which have been previously defined ?????**

# Bellman-Ford (Distance Vector Protocols)

The simplest approach.....



*Prob: Find the shortest path to 6.*

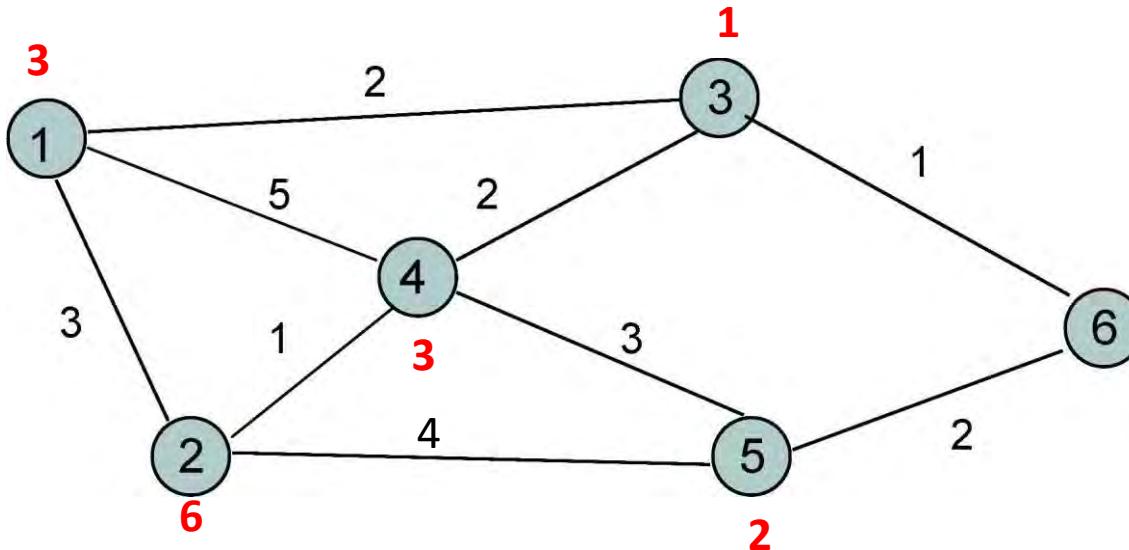
Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(-1, $\infty$ )				
1 <sup>st</sup>	(-1, $\infty$ )	(-1, $\infty$ )	(6, 1)	(-1, $\infty$ )	(6, 2)

Next node from Node 6

Distance of node 6  
from Node 5

# Bellman-Ford (Distance Vector Protocols)

The simplest approach.....



*Prob: Find the shortest path to 6.*

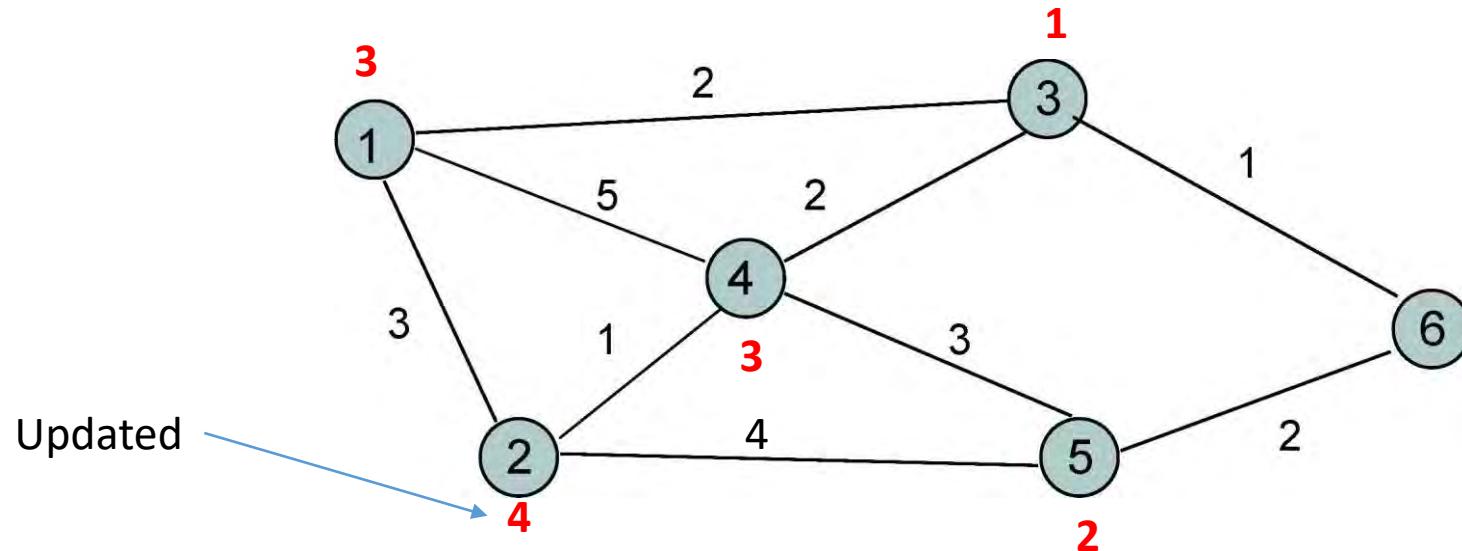
Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(-1, $\infty$ )				
1 <sup>st</sup>	(-1, $\infty$ )	(-1, $\infty$ )	(6, 1)	(-1, $\infty$ )	(6, 2)
2 <sup>nd</sup>	(3, 3)	(5, 6)	(6, 1)	(3, 3)	(6, 2)

Annotations for the 2<sup>nd</sup> iteration:

- Next node from Node 6: Points to the value 2 in the Node 5 column.
- Distance of node 6 from Node 5: Points to the value 2 in the Node 5 column.

# Bellman-Ford (Distance Vector Protocols)

The simplest approach.....



*Prob: Find the shortest path to 6.*

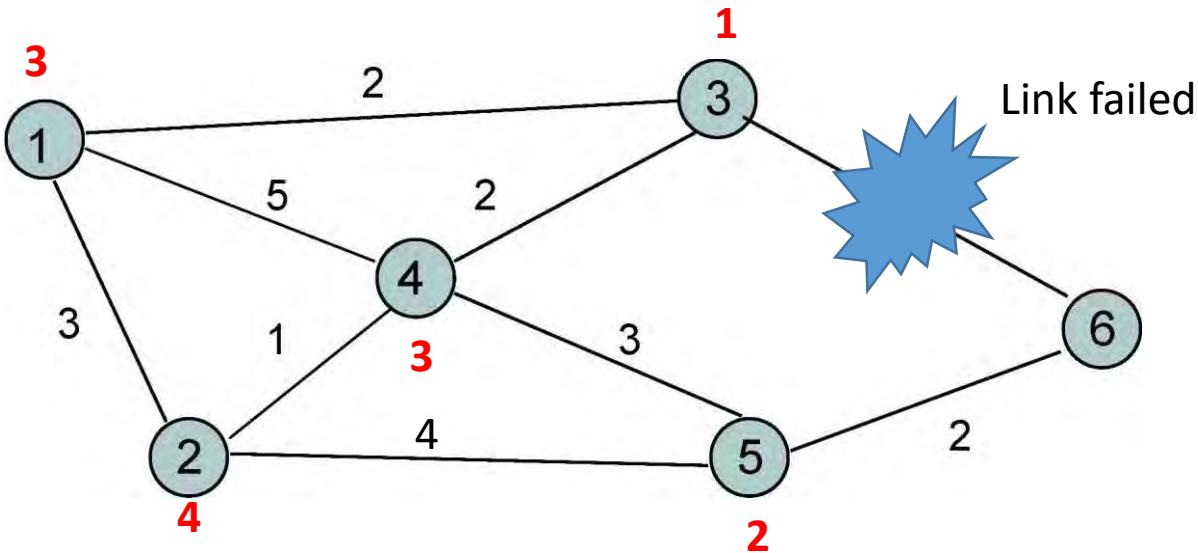
Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(-1, $\infty$ )				
1 <sup>st</sup>	(-1, $\infty$ )	(-1, $\infty$ )	(6, 1)	(-1, $\infty$ )	(6, 2)
2 <sup>nd</sup>	(3, 3)	(5, 6)	(6, 1)	(3, 3)	(6, 2)
3 <sup>rd</sup>	(3, 3)	(4, 4)	(6, 1)	(3, 3)	(6, 2)

Next node from Node 6  
Distance of node 6  
from Node 5

**Have you understood**



# Solve This.....



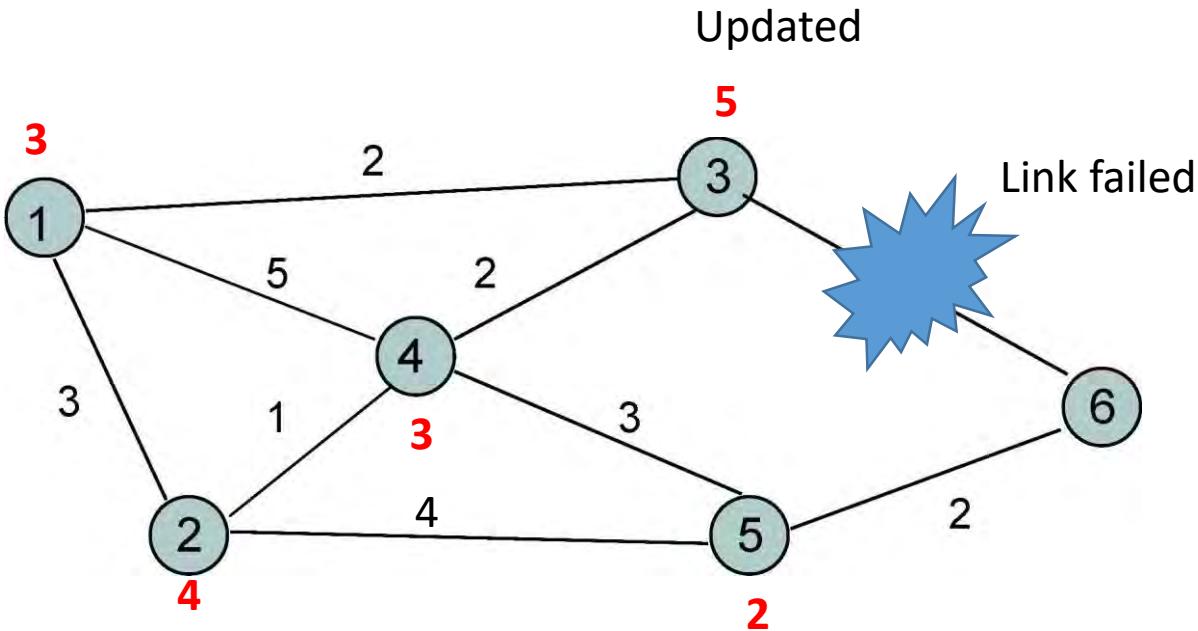
*Prob: Find the shortest path to 6.*

Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(3, 3)	(4, 4)	(6, 1)	(3, 3)	(6, 2)

Next node from Node 6

Distance of node 6  
from Node 5

# Solve This.....



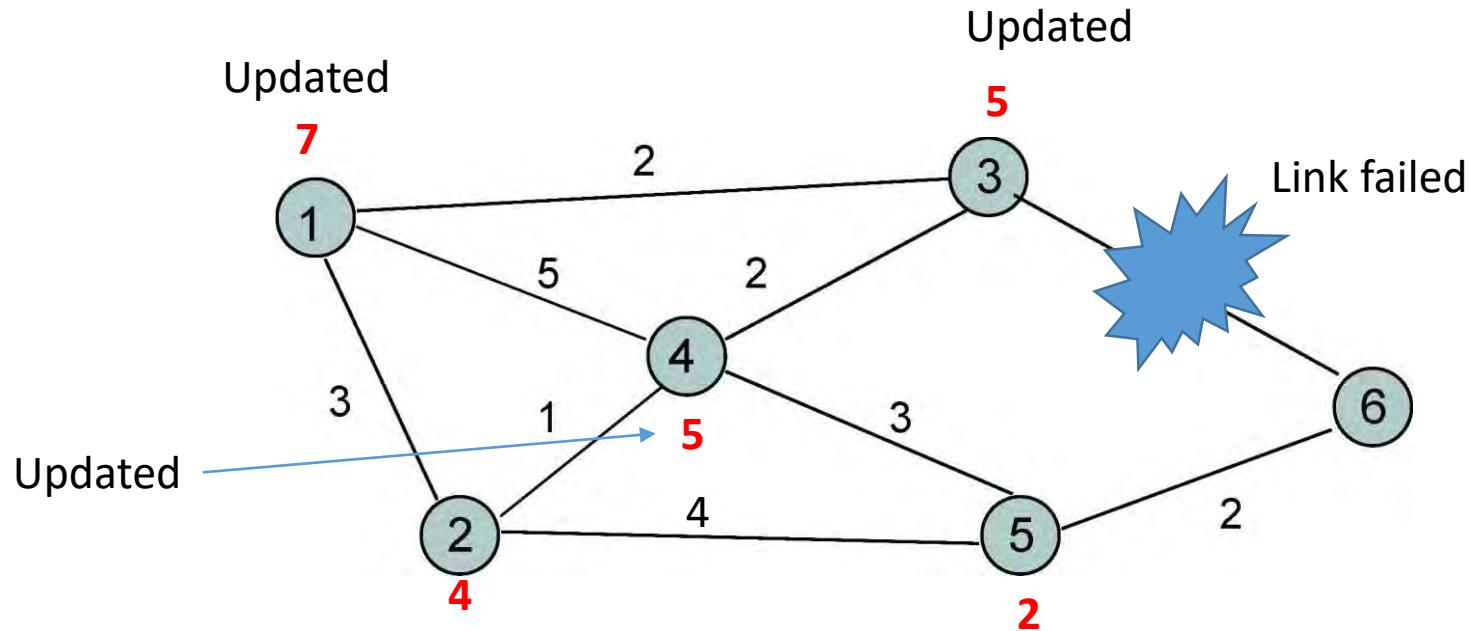
**Prob:** Find the shortest path to 6.

Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(3, 3)	(4, 4)	(6, 1)	(3, 3)	(6, 2)
1 <sup>st</sup>	(3, 3)	(4, 4)	(4, 5)	(3, 3)	(6, 2)

Next node from Node 6

Distance of node 6  
from Node 5

# Solve This.....

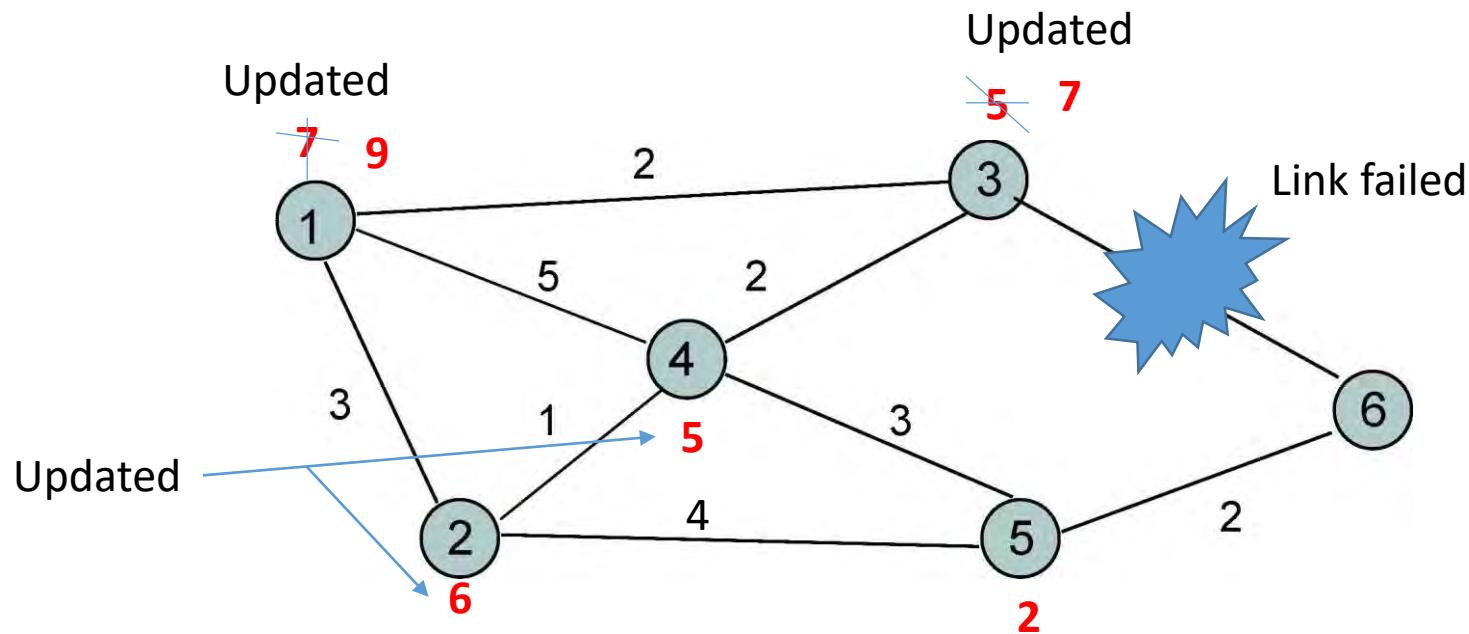


**Prob:** Find the shortest path to 6.

Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
Initial	(3, 3)	(4, 4)	(6, 1)	(3, 3)	(6, 2)
1 <sup>st</sup>	(3, 3)	(4, 4)	(4, 5)	(3, 3)	(6, 2)
2 <sup>nd</sup>	(3, 7)	(4, 4)	(4, 5)	(5, 5)	(6, 2)

Next node from Node 6  
Distance of node 6  
from Node 5

# Solve This.....



**Prob:** Find the shortest path to 6.

Iteration	Node 1	Node 2	Node 3	Node 4	Node 5
<b>Initial</b>	(3, 3)	(4, 4)	(6, 1)	(3, 3)	(6, 2)
<b>1<sup>st</sup></b>	(3, 3)	(4, 4)	(4, 5)	(3, 3)	(6, 2)
<b>2<sup>nd</sup></b>	(3, 7)	(4, 4)	(4, 5)	(5, 5)	(6, 2)
<b>3<sup>rd</sup></b>	(3, 7)	(5, 6)	(4, 7)	(5, 5)	(6, 2)
<b>4<sup>th</sup></b>	(2, 9)	(5, 6)	(4, 7)	(5, 5)	(6, 2)

Next node from Node 6

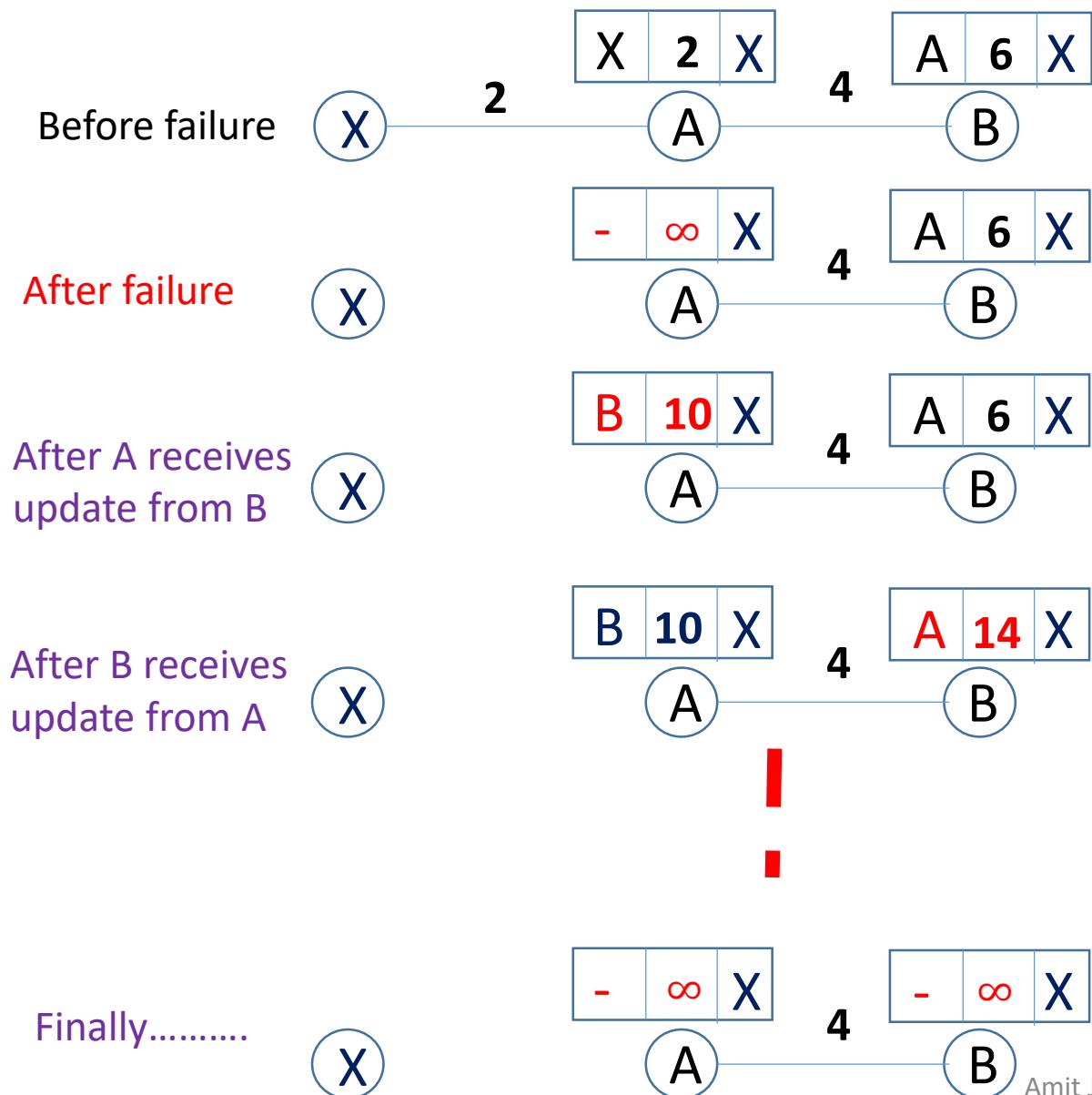
Distance of node 6  
from Node 5

# Some Key Points of Distance Vector(DV)

**Note:** DV uses hop counts to assign values to edges.

- **Advantage:** This algorithm **does not** require all of the nodes to operate in **synchronous manner**.
- **Disadvantage:** It does not **scale** well.
  - Changes in network topology are **not reflected quickly**, since updates are spread node-by-node.
  - May exists a loop and thus **count to infinity**, (if node failed is not traceable by other set of nodes).
  - **Thus, Can't be used in larger networks** which require more robust and precise algorithm.

## Counting to Infinity Problem in DV

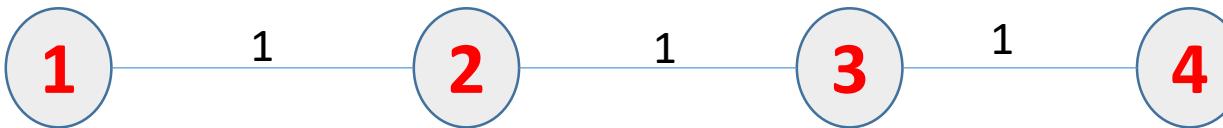


Iteration	Node A	Node B
Before Break	(X, 2)	(A, 6)
After Break		
1 <sup>st</sup>	(B, 10)	(A, 6)
2 <sup>nd</sup>	(B, 10)	(A, 14)
3 <sup>rd</sup>	(B, 18)	(A, 14)
4 <sup>th</sup>	(B, 18)	(A, 22)
...	...	...

# Remedies to count to Infinity Problem

- **Note:** This problem does not exist if link failed node shares its routing information before the other nodes. In our scenario, if A shares its information to node B before, node B shares its information to A, this problem will not exist.
- Remedies to the problem:
  1. **Defining infinity:** redefine infinity to a smaller number such as 100. But in this case, it can not be useful for a larger network.
  2. **Split Horizon:** Share only part of the routing table instead of all to each interface. If, according to its table, node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A; the information has come from A (A already knows). Taking information from node A, modifying it, and sending it back to node A creates the confusion. In our scenario, node B eliminates the last line of its routing table before it sends it to A. In this case, node A keeps the value of infinity as the distance to X. Later when node A sends its routing table to B, node B also corrects its routing table. The system becomes stable after the first update: both node A and B know that X is not reachable.
  3. **Split Horizon and Poison Reverse:** Using the split horizon strategy has one draw back. Normally, the distance vector protocol uses a timer, and if there is no news about a route, the node deletes the route from its table. When node B in the previous scenario eliminates the route to X from its advertisement to A, node A can not guess that this is due to the split horizon strategy (the source of information was A) or because B has not received any news about X recently. The split horizon strategy can be combined with the poison reverse strategy. Node B can still advertise the value for X, but if the source of information is A, it can replace the distance with infinity as a warning: "Do not use this value; what I know about this route comes from you."

# Practice: Observe Counting to Infinity Problem for the given Network



Before break

Destination is node 4



After break

Update	Node 1	Node 2	Node 3
Before Break	(2, 3)	(3, 2)	(4, 1)
After Break	(2, 3)	(3, 2)	(2, 3)
1 <sup>st</sup>	(2, 3)	(3, 4)	(2, 3)
2 <sup>nd</sup>	(2, 5)	(3, 4)	(2, 5)
3 <sup>rd</sup>	(2, 5)	(3, 6)	(2, 5)
4 <sup>th</sup>	(2, 7)	(3, 6)	(2, 7)
...	...	...	...

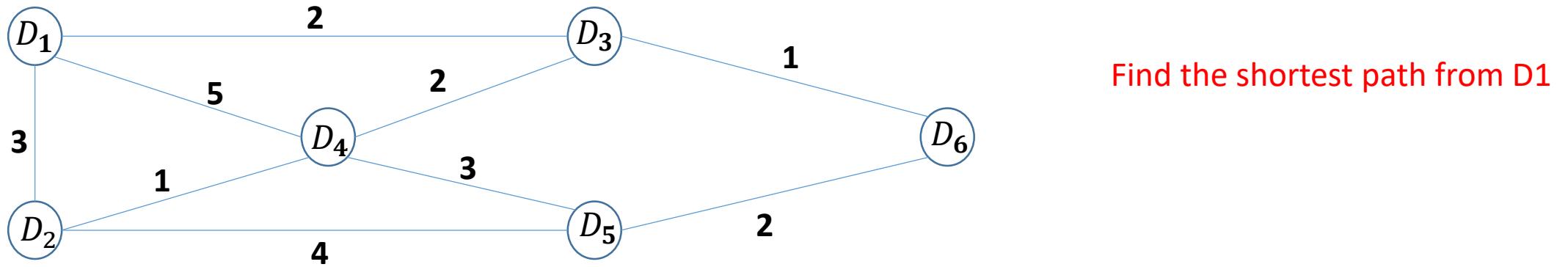
# Dijkstra Algorithm (Link State Protocol)

- Some key-points to remember:

- Initially, choose the source node and make distance from source node to itself as 0, and from source to all other nodes infinity.
- In the first iteration, select the source node in the visited queue and find the distance from source node to all other nodes which is directly connected to it.
- In the second iteration, choose the node with minimum cost from the source node as visited node; and find the minimum distance from the source node to all other nodes using the visited node.
- Repeat second step until all nodes become visited nodes.

**Note:** If a node can not be reached through from the source using all of the visited nodes then distance of the node from source will be infinite.

# Execution of Dijkstra's Algorithm



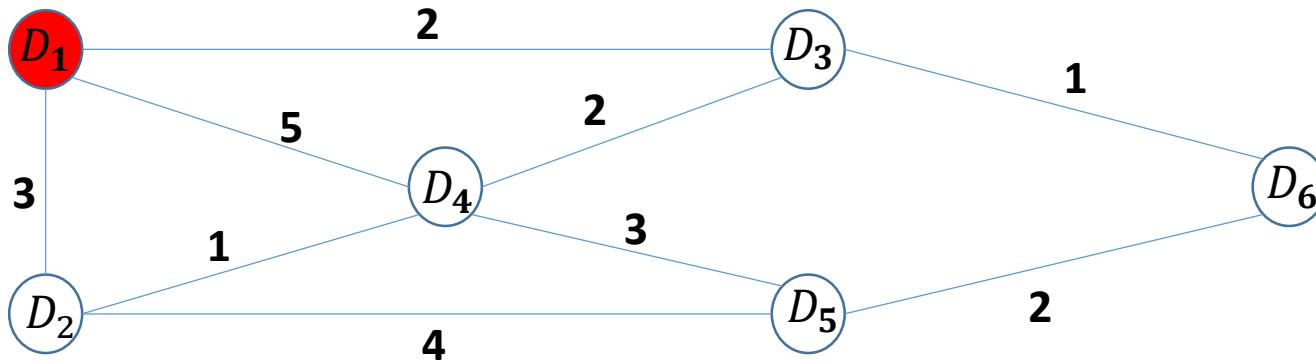
Iteration	Visited Nodes	$D_2$	$D_3$	$D_4$	$D_5$	$D_6$
Initial	{-}	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$

**Blind Rule:** The only question to be asked to yourself at every node  $N_x$ :

**Can we reach to node  $N_x$  from the source node N using the nodes in the visited queue ?????**

**Note:** The node having minimum cost will become a member of visited queue in every iteration.

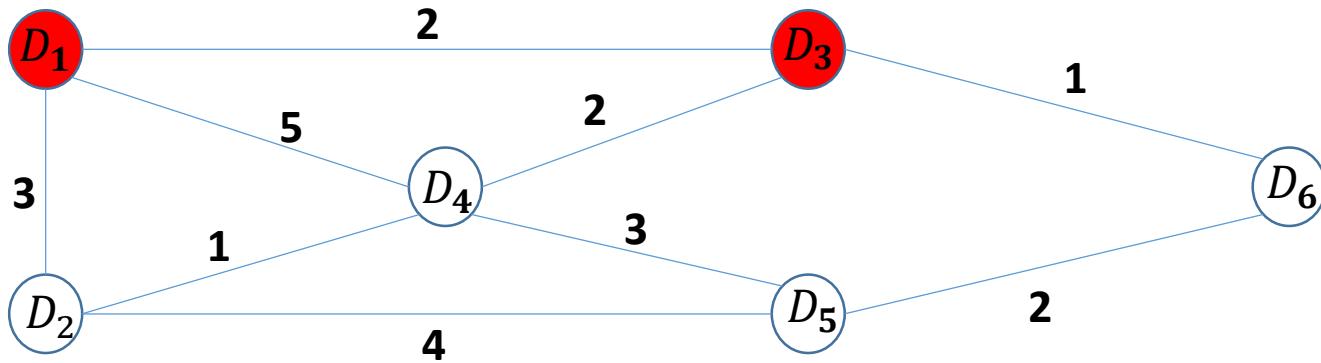
# Execution of Dijkstra's Algorithm



Find the shortest path from  $D_1$

Iteration	Visited Nodes	$D_2$	$D_3$	$D_4$	$D_5$	$D_6$
Initial	{-}	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
1 <sup>st</sup>	{ $D_1$ }	3 ( $D_1-D_2$ )	2 ( $D_1-D_3$ )	5 ( $D_1-D_4$ )	$\infty$	$\infty$

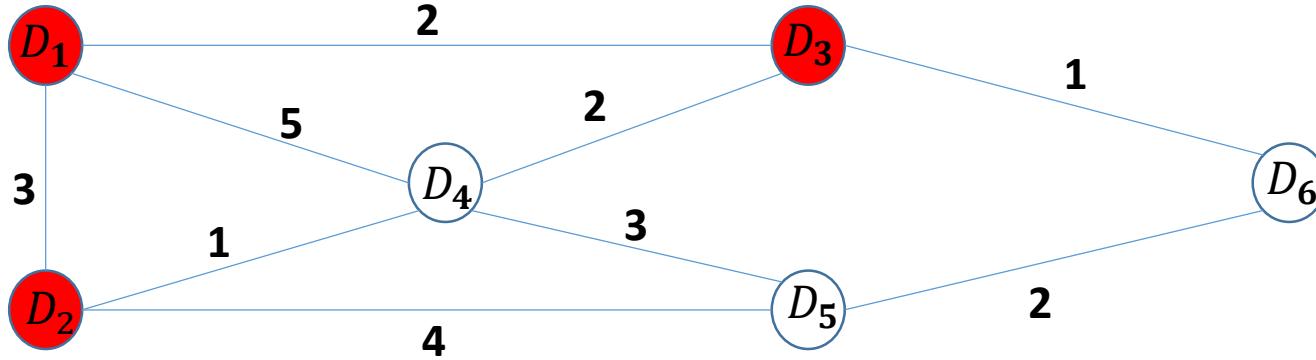
# Execution of Dijkstra's Algorithm



Find the shortest path from  $D_1$

Iteration	Visited Nodes	$D_2$	$D_3$	$D_4$	$D_5$	$D_6$
Initial	{-}	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
1 <sup>st</sup>	{ $D_1$ }	3 ( $D_1$ - $D_2$ )	2 ( $D_1$ - $D_3$ )	5 ( $D_1$ - $D_4$ )	$\infty$	$\infty$
2 <sup>nd</sup>	{ $D_1$ , $D_3$ }	3 ( $D_1$ - $D_2$ )	2 ( $D_1$ - $D_3$ )	4 ( $D_1$ - $D_3$ - $D_4$ )	$\infty$	3 ( $D_1$ - $D_3$ - $D_6$ )

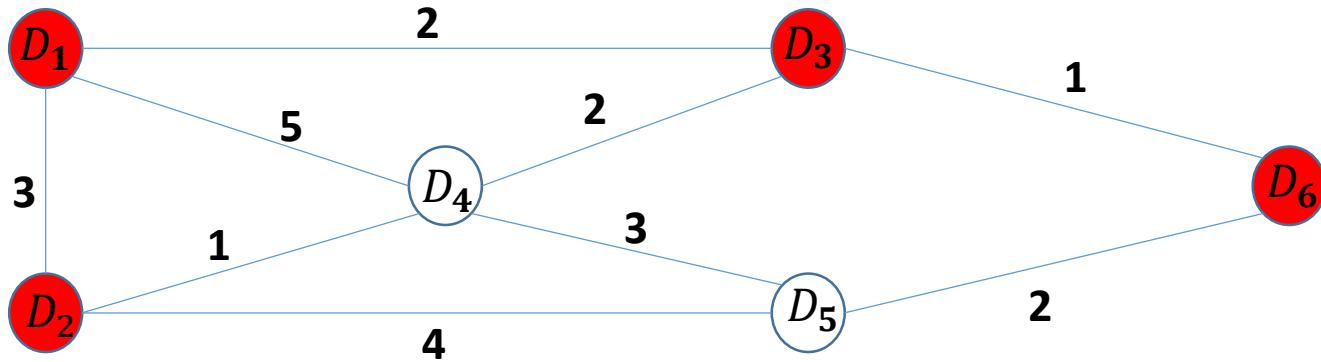
# Execution of Dijkstra's Algorithm



Find the shortest path from D<sub>1</sub>

Iteration	Visited Nodes	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>
Initial	{-}	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
1 <sup>st</sup>	{D <sub>1</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	5 (D <sub>1</sub> -D <sub>4</sub> )	$\infty$	$\infty$
2 <sup>nd</sup>	{D <sub>1</sub> , D <sub>3</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	$\infty$	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )
3 <sup>rd</sup>	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	7 (D <sub>1</sub> -D <sub>2</sub> -D <sub>5</sub> )	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )

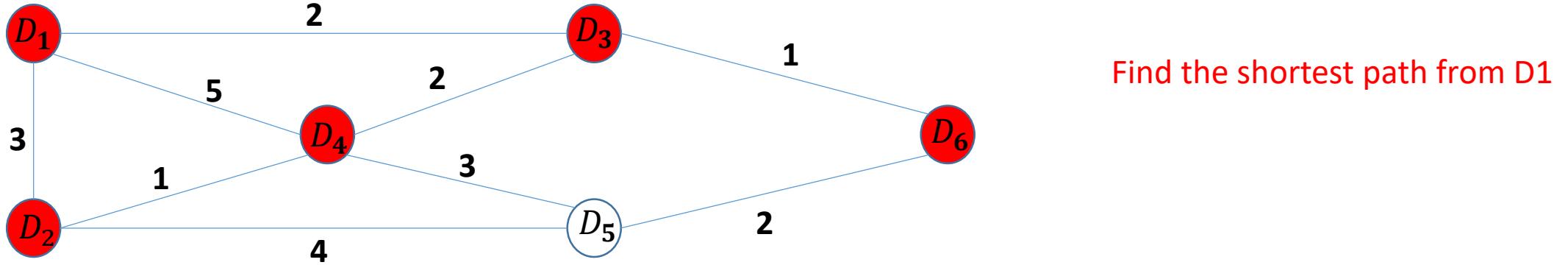
# Execution of Dijkstra's Algorithm



Find the shortest path from D<sub>1</sub>

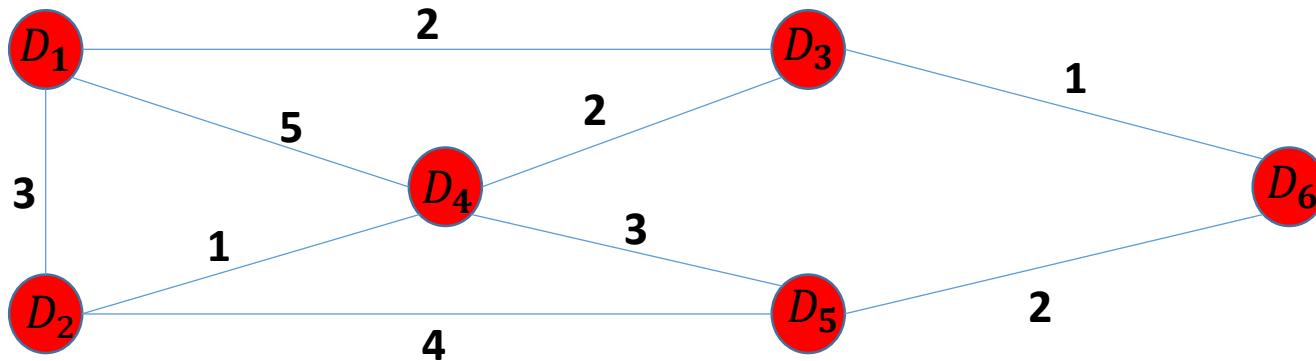
Iteration	Visited Nodes	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>
Initial	{-}	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
1 <sup>st</sup>	{D <sub>1</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	5 (D <sub>1</sub> -D <sub>4</sub> )	$\infty$	$\infty$
2 <sup>nd</sup>	{D <sub>1</sub> , D <sub>3</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	$\infty$	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )
3 <sup>rd</sup>	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	7 (D <sub>1</sub> -D <sub>2</sub> -D <sub>5</sub> )	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )
4 <sup>th</sup>	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> , D <sub>6</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	5 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> -D <sub>5</sub> )	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )

# Execution of Dijkstra's Algorithm



Iteration	Visited Nodes	$D_2$	$D_3$	$D_4$	$D_5$	$D_6$
Initial	{-}	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
1 <sup>st</sup>	{ $D_1$ }	3 ( $D_1-D_2$ )	2 ( $D_1-D_3$ )	5 ( $D_1-D_4$ )	$\infty$	$\infty$
2 <sup>nd</sup>	{ $D_1, D_3$ }	3 ( $D_1-D_2$ )	2 ( $D_1-D_3$ )	4 ( $D_1-D_3-D_4$ )	$\infty$	3 ( $D_1-D_3-D_6$ )
3 <sup>rd</sup>	{ $D_1, D_2, D_3$ }	3 ( $D_1-D_2$ )	2 ( $D_1-D_3$ )	4 ( $D_1-D_3-D_4$ )	7 ( $D_1-D_2-D_5$ )	3 ( $D_1-D_3-D_6$ )
4 <sup>th</sup>	{ $D_1, D_2, D_3, D_6$ }	3 ( $D_1-D_2$ )	2 ( $D_1-D_3$ )	4 ( $D_1-D_3-D_4$ )	5 ( $D_1-D_3-D_6-D_5$ )	3 ( $D_1-D_3-D_6$ )
5 <sup>th</sup>	{ $D_1, D_2, D_3, D_4, D_6$ }	3 ( $D_1-D_2$ )	2 ( $D_1-D_3$ )	4 ( $D_1-D_3-D_4$ )	5 ( $D_1-D_3-D_6-D_5$ )	3 ( $D_1-D_3-D_6$ )

# Execution of Dijkstra's Algorithm



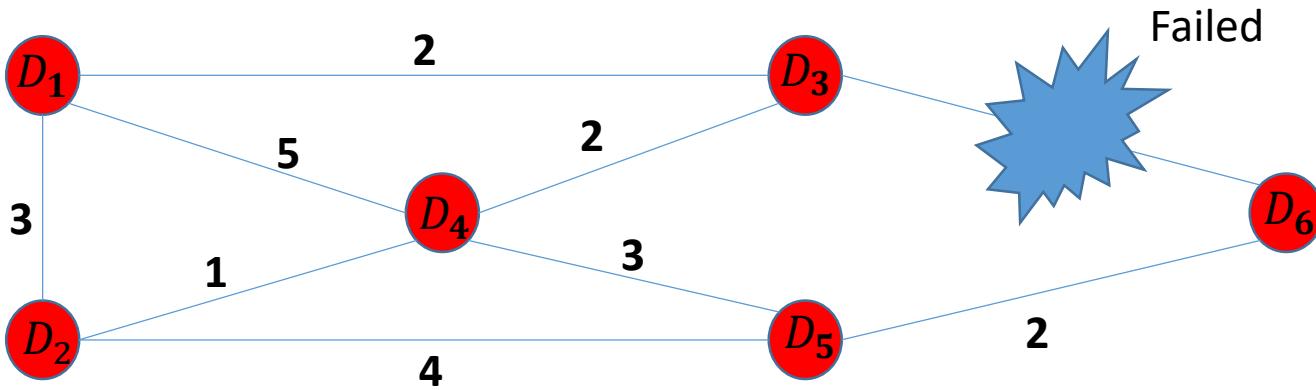
Find the shortest path from D<sub>1</sub>

Iteration	Visited Nodes	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>
Initial	{-}	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$
1 <sup>st</sup>	{D <sub>1</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	5 (D <sub>1</sub> -D <sub>4</sub> )	$\infty$	$\infty$
2 <sup>nd</sup>	{D <sub>1</sub> , D <sub>3</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	$\infty$	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )
3 <sup>rd</sup>	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	7 (D <sub>1</sub> -D <sub>2</sub> -D <sub>5</sub> )	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )
4 <sup>th</sup>	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> , D <sub>6</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	5 (D <sub>1</sub> -D <sub>2</sub> -D <sub>5</sub> )	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )
5 <sup>th</sup>	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> , D <sub>4</sub> , D <sub>6</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	5 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> -D <sub>5</sub> )	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )
6 <sup>th</sup>	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> , D <sub>4</sub> , D <sub>5</sub> , D <sub>6</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	5 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> -D <sub>5</sub> )	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )

# Routing Table at Node $D_1$

Destination	Next Node	Cost
$D_2$	$D_2$	3
$D_3$	$D_3$	2
$D_4$	$D_3$	4
$D_5$	$D_3$	5
$D_6$	$D_3$	3

# Reaction to Failure



Iteration	Visited Nodes	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>
Initial	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> , D <sub>4</sub> , D <sub>5</sub> , D <sub>6</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	5 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> -D <sub>5</sub> )	3 (D <sub>1</sub> -D <sub>3</sub> -D <sub>6</sub> )
1 <sup>st</sup>	{D <sub>1</sub> , D <sub>2</sub> , D <sub>3</sub> , D <sub>4</sub> , D <sub>5</sub> , D <sub>6</sub> }	3 (D <sub>1</sub> -D <sub>2</sub> )	2 (D <sub>1</sub> -D <sub>3</sub> )	4 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> )	7 (D <sub>1</sub> -D <sub>3</sub> -D <sub>4</sub> -D <sub>5</sub> )	9 (D <sub>1</sub> -D <sub>2</sub> -D <sub>5</sub> -D <sub>6</sub> )

Updated quickly

# Advantages and Disadvantages of Link State

- **Advantages:**

- Fast and loopless convergence (thus, it can be used in larger networks) as it uses centralized approach.
- Support for precise metrics , and multiple metrics if necessary (throughput, delay, cost, reliability)
- Supporting for multiple paths to a destination
  - Algorithm can be supported to find the two best paths to a destination
- Scales better than distance vector
- Each routers computes routes independently from original data (i.e. not relying on intermediate nodes)

- **Disadvantage:**

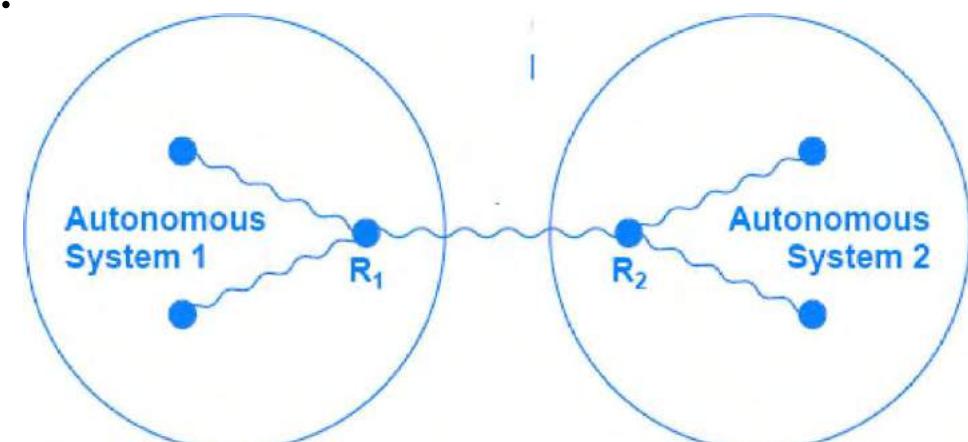
- Computational load on routers and
- Negative edge cost can not be handled

# Key points to Remember

- **For Bellman Ford:**
  - **Blind Rule:** The only question to be asked to yourself at every node  $N_x$  for given destination  $N$ :
    - Can we reach to node  $N$  from node  $N_x$  *using the nodes which have been previously defined ?????*
- **For Dijkstra's Algorithm:**
  - **Blind Rule:** The only question to be asked to yourself at every node  $N_x$  for given Source  $N$ :
    - Can we reach to node  $N_x$  from the source node  $N$  *using the nodes in the visited queue ?????*
- **For Both:**
  - If you get two or more different paths of same cost then you can choose any one as per your wish.
  - Thus, if different students solve question **independently**, then path in their answer may vary but the cost of different nodes has to match for all students.

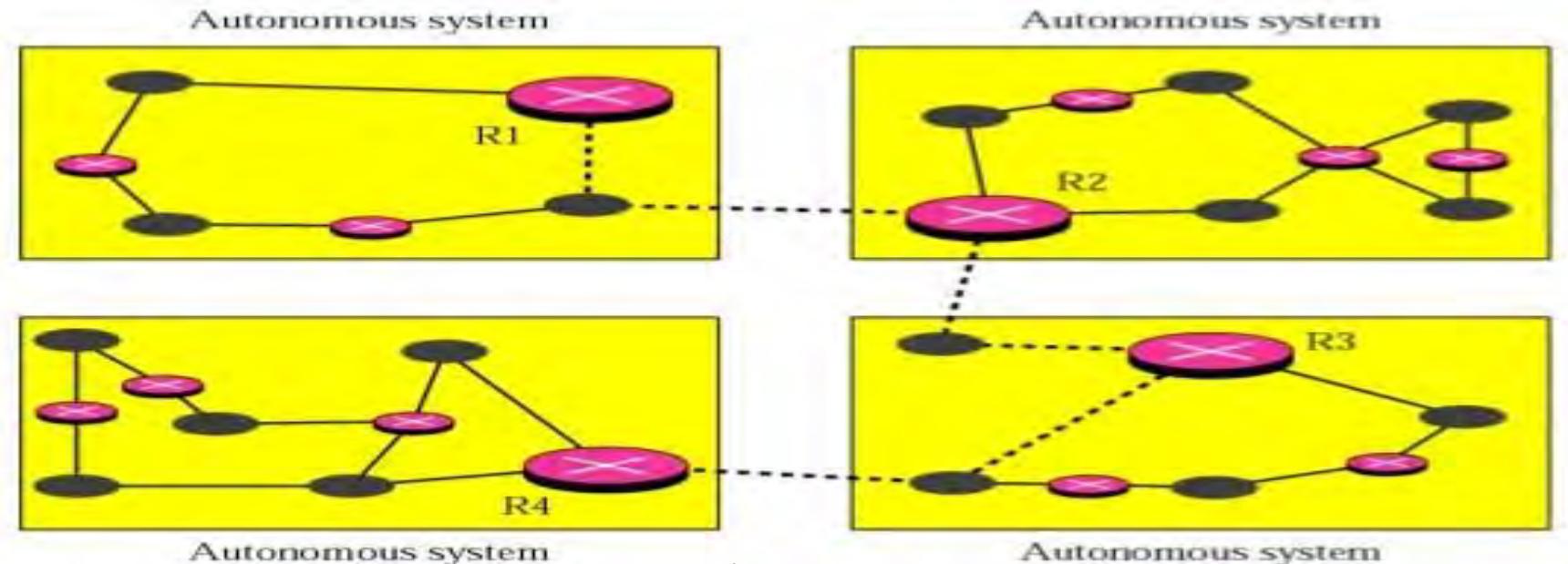
# The concept of Autonomous System

- We cannot run an automatic routing protocol for the **entire Global Internet**, because of its complexity.
- So, networks and routers are owned by organizations and individuals.
- Within each, an **administrative authority** can guarantee that internal routes remain consistent and viable.
- For purposes of routing, a group of networks controlled by a single administrative authority is called **an autonomous system (AS)** and identified by an *autonomous system number*.



# Implementation of Routing in Today's Internet

- The **Global Internet** consists of **AS interconnected** with each other.
- *Intradomain Routing*: Routing within an *Autonomous System (AS)*.
- *Interdomain Routing*: Routing between Autonomous Systems (ASs).



# Routing Protocol

- A routing protocol is a combination of **rules** and **procedures** that lets routers in the internet to **share whatever they know** about the internet or their neighbourhood. The sharing of information allows a router in **Bhubaneswar** to know about the failure of a network in **Greenwich**.

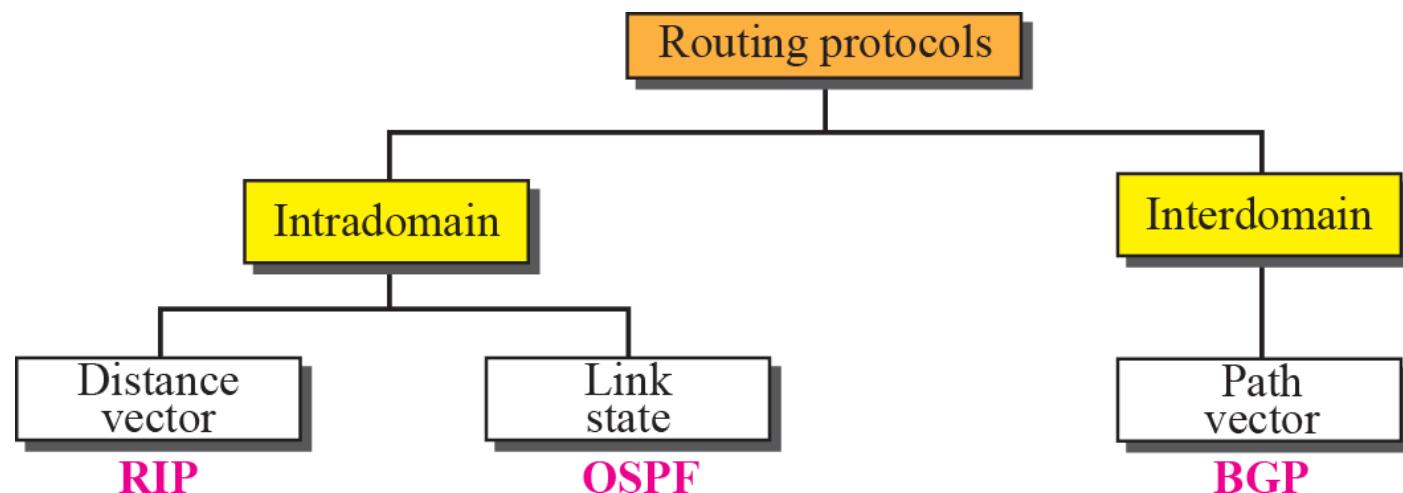


Fig: Popular Routing Protocols

# Routing Information Protocol (RIP)

- RFC 1058
- It is an intradomain routing protocol used inside an autonomous system
- It implements the distance vector algorithm (like Bellman ford) in the Internet.
- Runs on UDP, port number 520
- Uses number of hops as metric
- Maximum limit is 15
  - Suitable for smaller networks (local area environment)
  - Value of 16 is reserved to represent infinity

# RIP Operation: Implementation of DVP

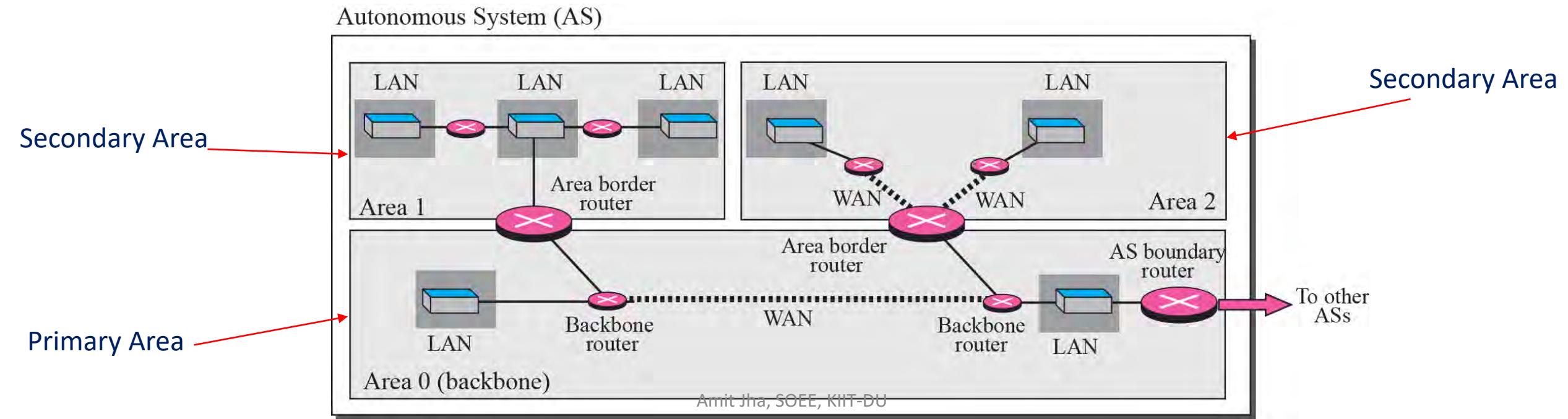
- Router sends update message to neighbours every 30 sec.
- A router expects to receive an update message from each of its neighbours within 180 sec in the worst case.
- If a router does not receive an update message from its neighbour X within 180 sec, it assumes that the link to X has failed and sets the minimum cost to 16 (i.e., infinity).
- Uses ***split horizon with poisoned reverse***.
- Two RIP packet types:
  - 1) Request to ask neighbours for distance vector table
  - 2) Response to advertise distance vector table

# Open Shortest Path First (OSPF)

- RFC 2328 (v2)
- It is also an intradomain routing protocol i.e., used within an AS.
- Uses the link state algorithm (like Dijkstra's algorithm)
- Fixes some of the deficiencies of RIP (Distance Vector algorithm)
- Its domain is also an autonomous system.
- **Area:** unlike RIP, it is used to handle **larger networks**, so its domain system is divided into smaller areas. An area is a collection of networks, hosts, and routers all contained within an autonomous system.

# OSPF: Operation

- Routers inside an area **flood the area** with routing information.
- At the border of an area, special routers called area ***border routers*** summarize the information about the area and send it to the other areas.
- Among the areas inside an AS, there exists a ***backbone area*** (primary area) to which all other areas are connected (called secondary area).



Can we use Distance Vector (Bellman Ford) or Link State (Dijkstra) For Inter domain  
(between Autonomous Systems) routing ??



Can we use Distance Vector (Bellman Ford) or Link State (Dijkstra) For Inter domain  
(between Autonomous Systems) routing ??

No.... Because of the following issues !!!

**Both are not useful for a larger network**

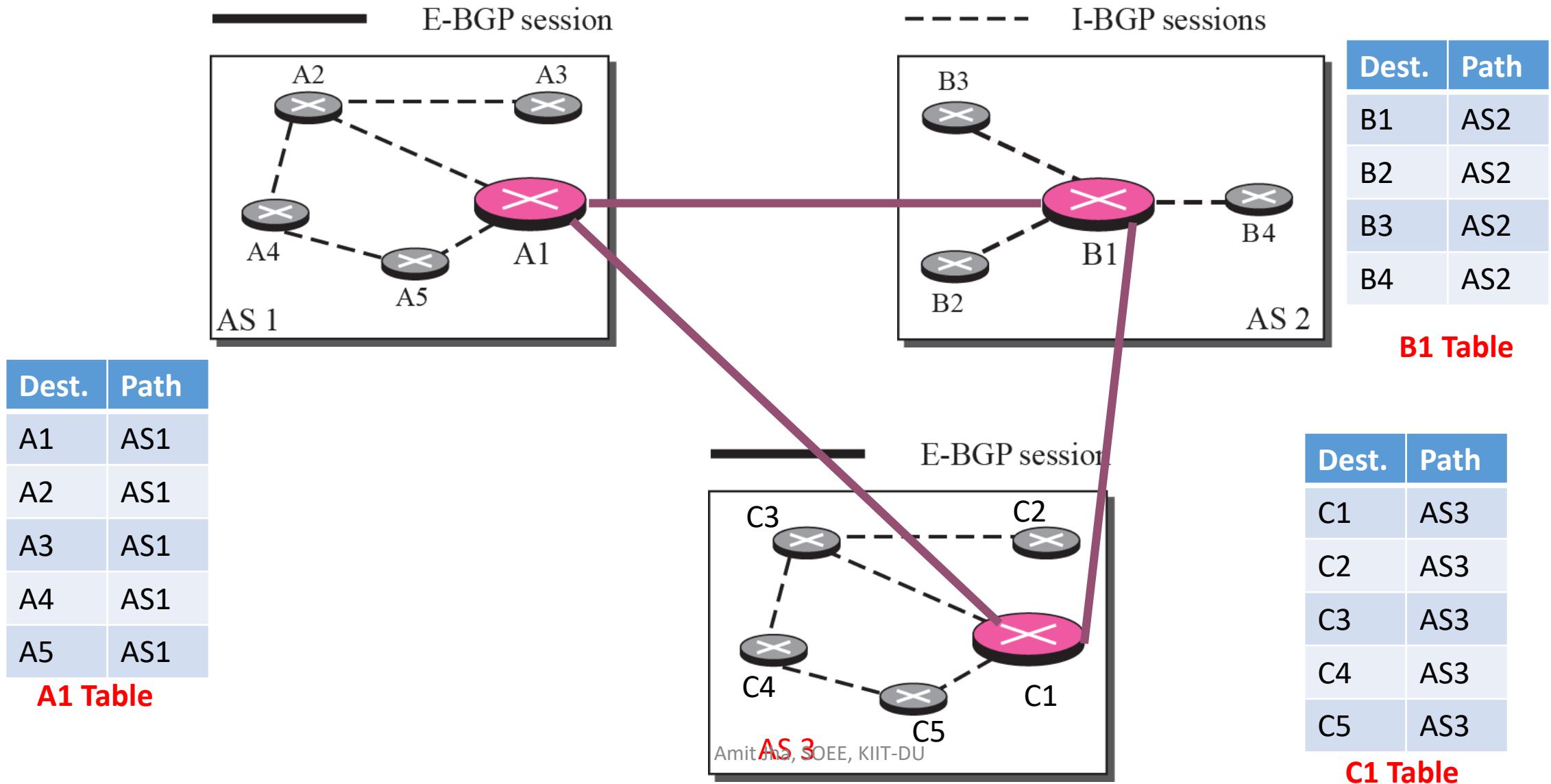
- **Bellman Ford** can not be used for a large network because; not well scalable, update is not quick, counting to infinity problem.
- **Dijkstra's Algorithm** is not useful because it floods complete routing table thus may create congestion in the network.
- Moreover, different AS may use different metric for their internal routing. For e.g., if routing is to be done between AS1 (using Link state) and AS2 (using Distance Vector), then AS1 can use delay as its metric whereas AS2 is **compelled** to use hop count as its metric.



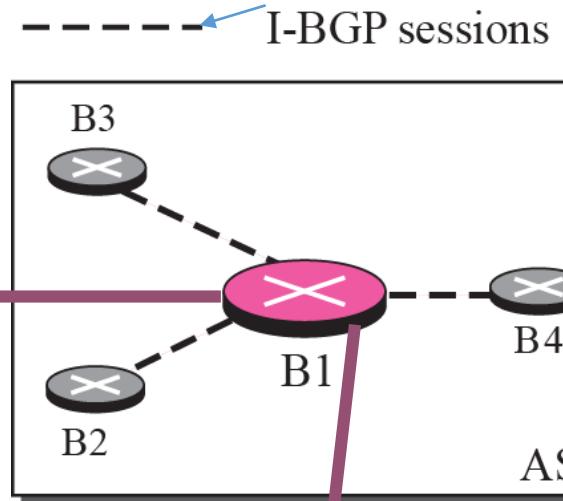
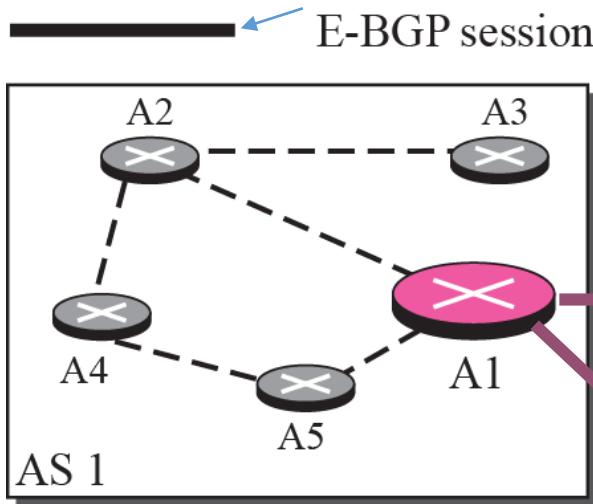
# Path Vector Routing: Introduction

- It works on the principle of Distance Vector routing.
- Each AS has one node which acts on the behalf of it called *speaker node*.
- The speaker node in an AS creates a routing table and advertises it to speaker nodes in the neighbouring ASs.
- **Only speaker** node in each AS **can communicate** to each other.
- A speaker node **advertises the path, not the metric** of the nodes, in its autonomous system or other autonomous systems.

# Path Vector Routing: Operation: Initially



# Path Vector Routing: Operation: After becoming Stable

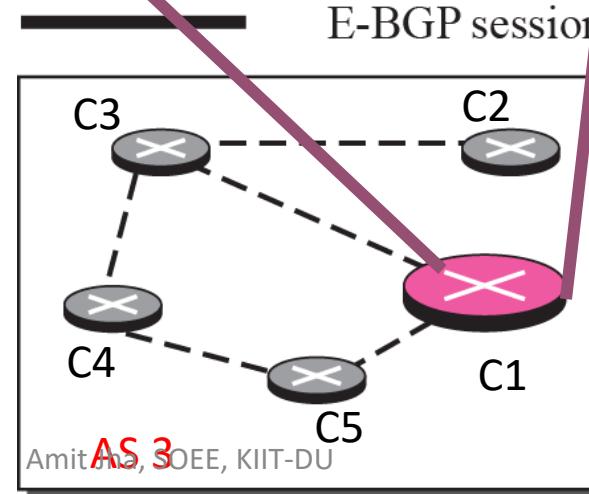


Dest.	Path
A1 to A5	AS2 – AS1
B1 to B4	AS2
C1 to C5	AS2 – AS3

**B1 Table**

Dest.	Path
A1 to A5	AS1
B1 to B4	AS1 - AS2
C1 to C5	AS1 - AS3

**A1 Table**



Dest.	Path
A1 to A5	AS3 - AS1
B1 to B4	AS3 – AS2
C1 to C4	AS3

**C1 Table**

**What is the optimum path in path vector routing ?**

**Ans:** While deciding the optimum path we can not include the metric. This is because two different AS can use different metric as seen previously (DV using hop count whereas, link state using delay as metric ). Thus, **we choose optimum path as a path having less number of autonomous system.**

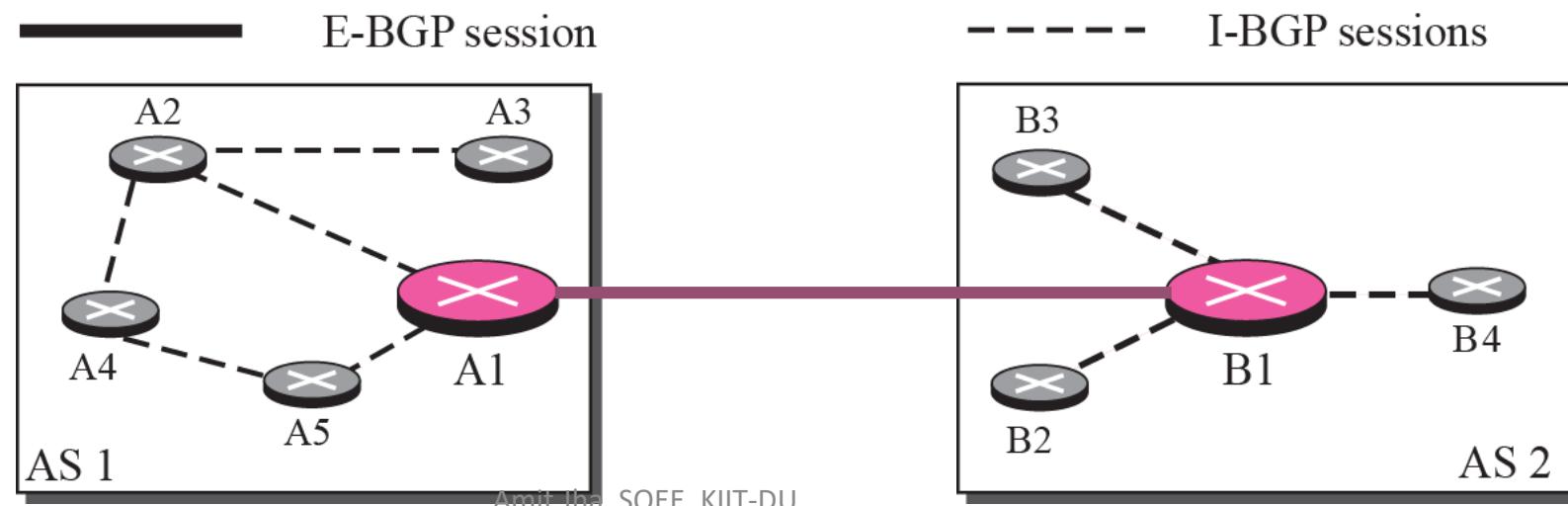
**For e.g.,** if a sender from AS1 wants to send a packet to a receiver in AS3 then the optimum path will be AS1 - AS3, but not AS1 - AS2 - AS3.

# Border Gateway Protocol (BGP)

- It is an inter domain routing protocol.
- It is based on path vector routing algorithm.
- BGP uses the services of TCP to create a reliable environment.
- BGP Session: A session is a connection that is established between two BGP routers only for the sake of exchanging routing information.
- There are two types of BGP Session:
  1. External BGP (E-BGP)
  2. Internal BGP (I-BGP)

# Border Gateway Protocol (BGP): BGP Sessions

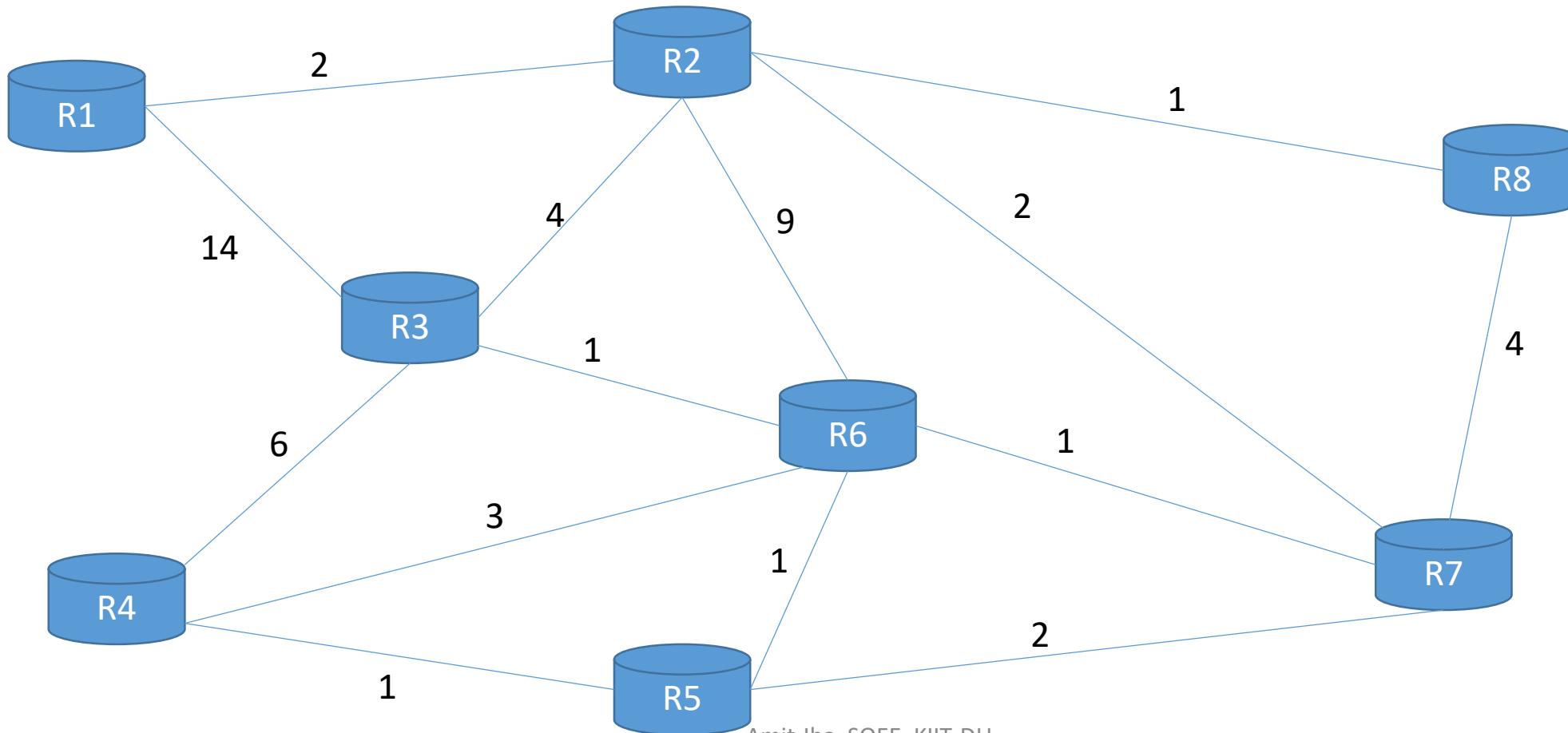
- External BGP (E-BGP): The E-BGP session is used to exchange information between two speaker nodes belonging to two different autonomous systems.
- Internal BGP (I-BGP): The I-BGP session, on the other hand, is used to exchange routing information between two routers inside an autonomous system.



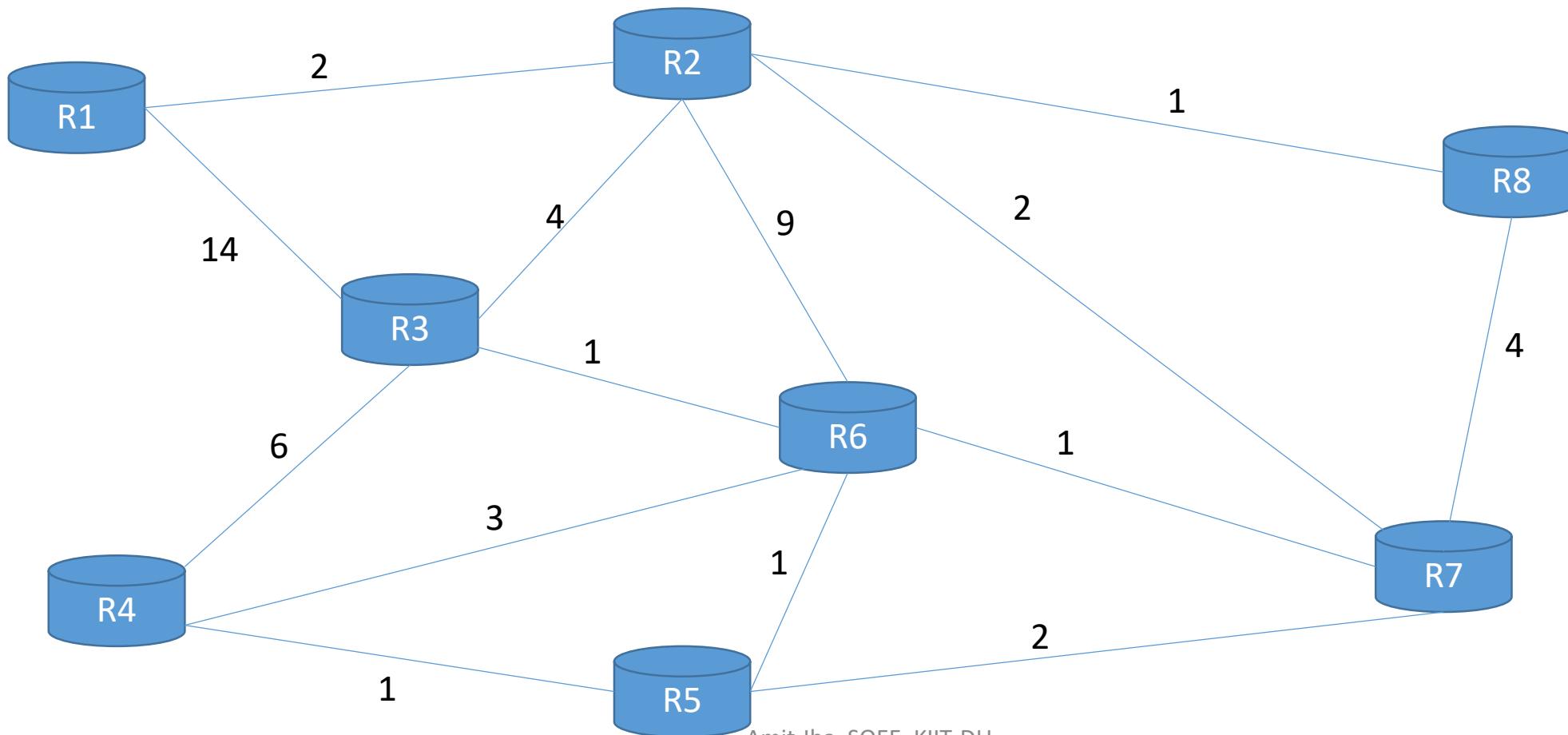
# Practice Questions

- Compare the following:
  1. Bellman Ford Vs. Dijkstra's Algorithm
  2. Distance Vector Vs. Link State
- Write short notes on the following:
  1. RIP
  2. OSPF
- Explain briefly what do you understand by BGP.

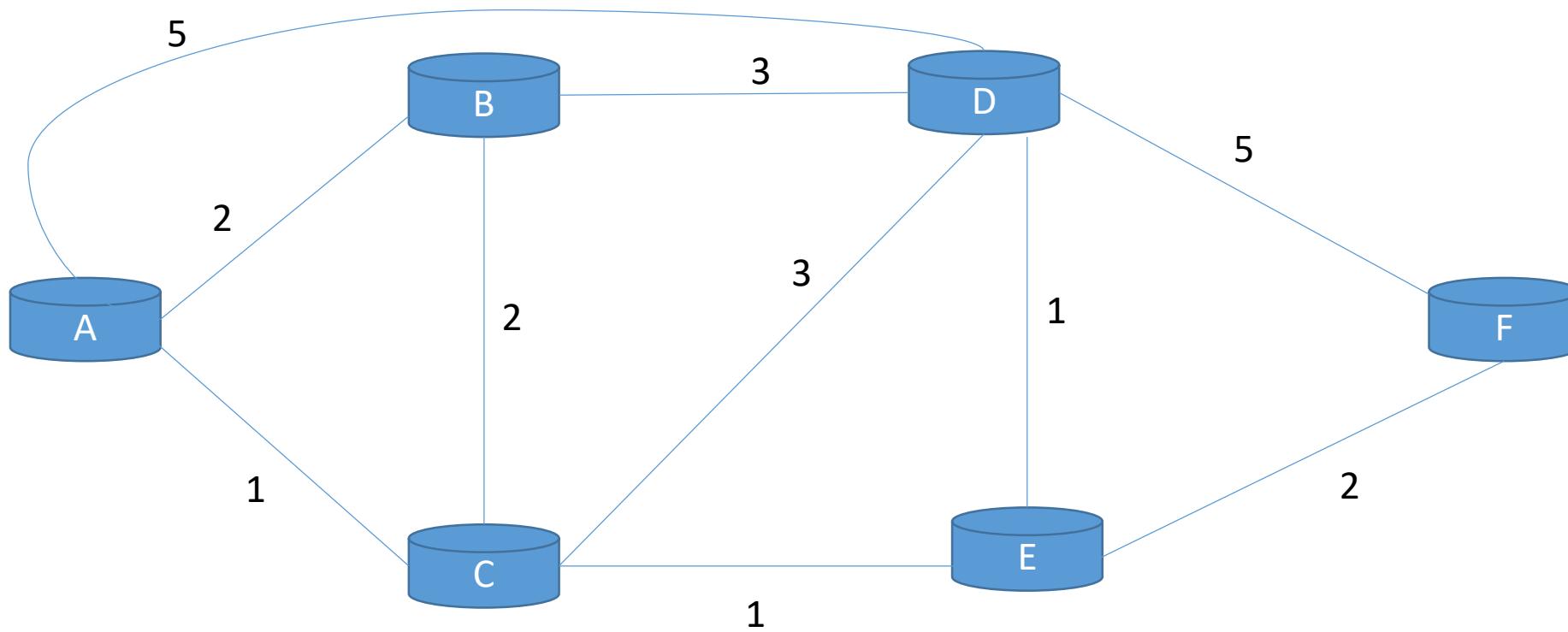
# HYU-1: Find the shortest path from router R1 to all routers using Dijkstra's Algorithm.



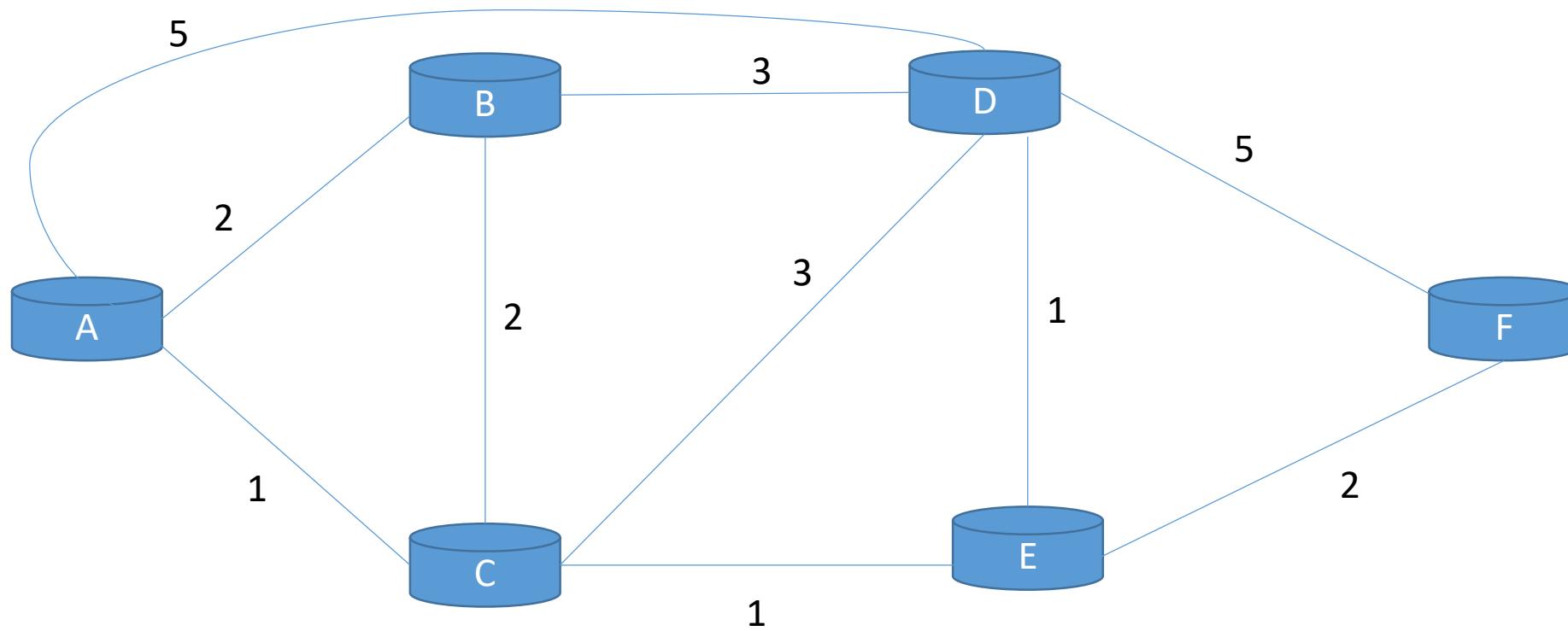
# HYU-2: Find the shortest path to router R8 from all routers using Bellman Ford Algorithm.



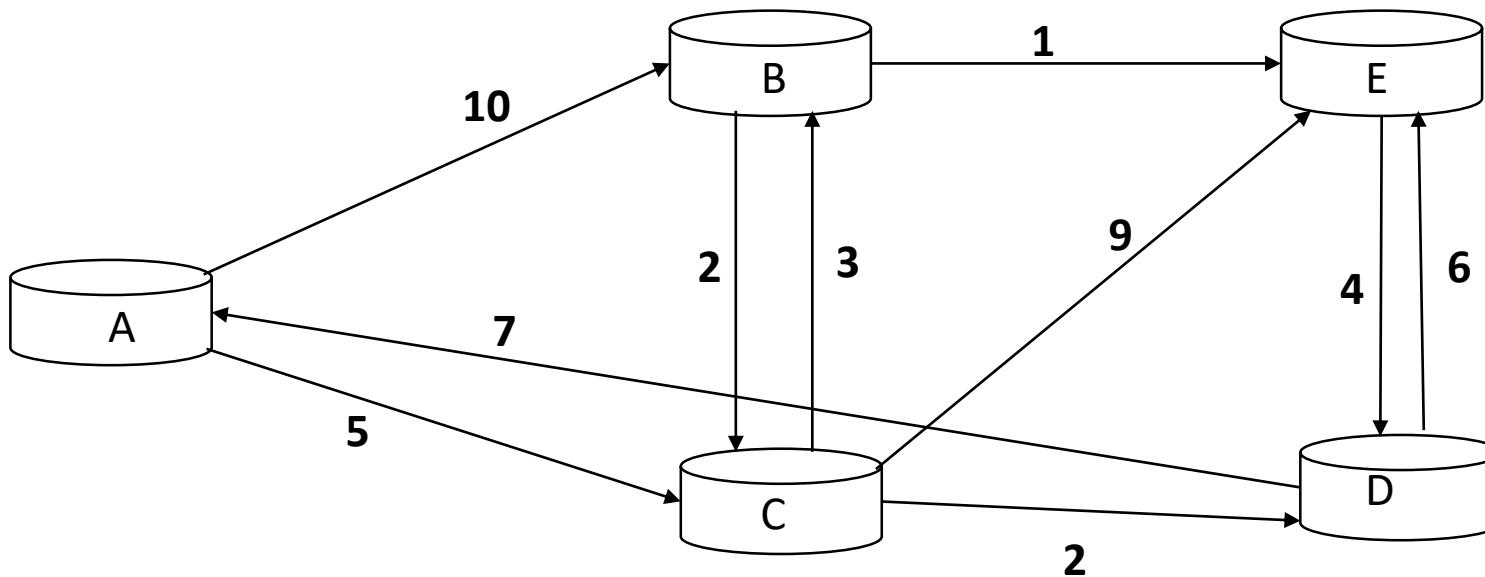
**HYU-3:** Find the shortest path from node A to node F using Dijkstra's algorithm. Also, mention the routing table at node E.



# HYU-4: Find the shortest path to node F using Bellman Ford algorithm.

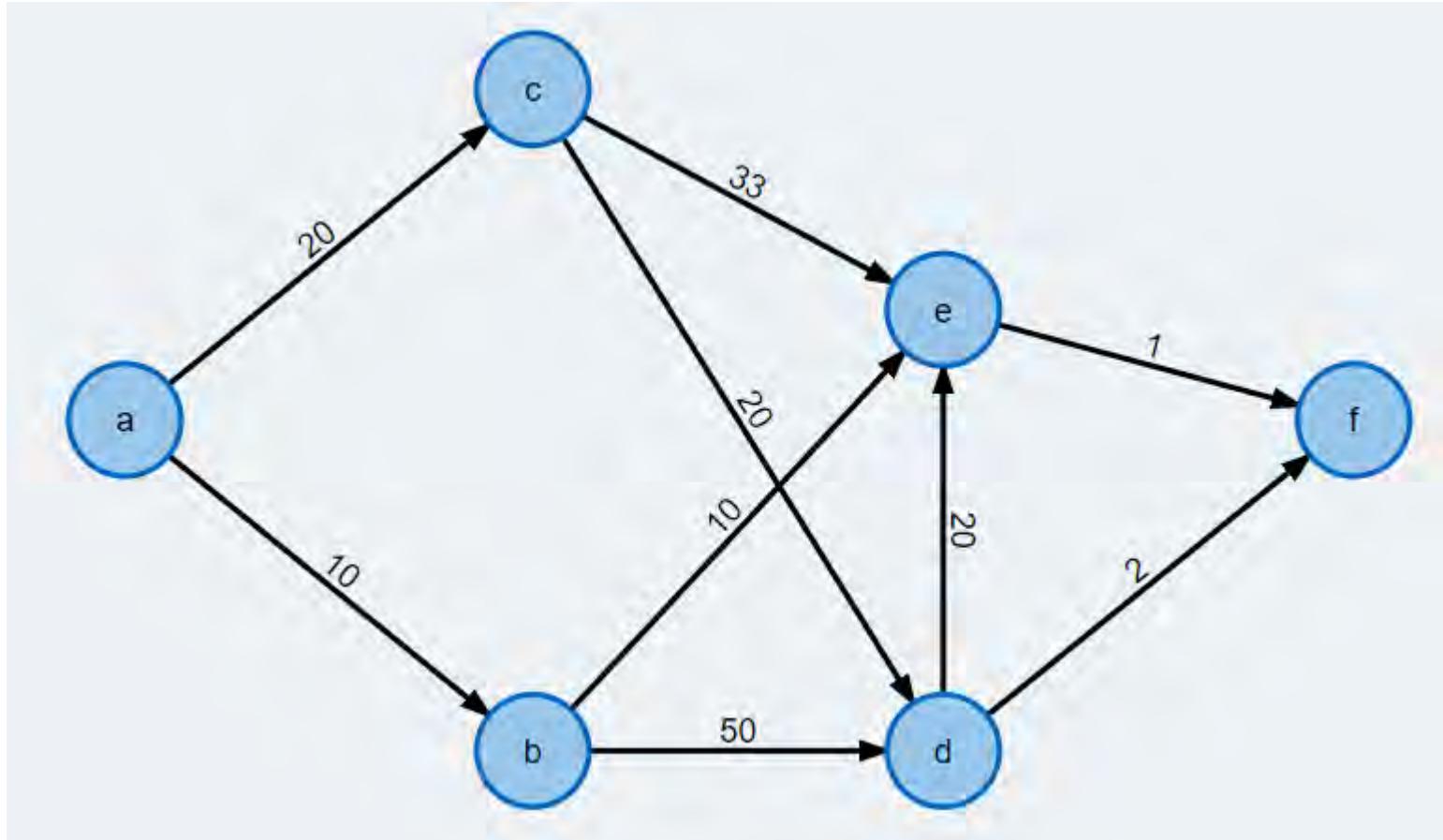


# HYU-6: Find Shortest path using Dijkstra's and Bellman Ford algorithm.

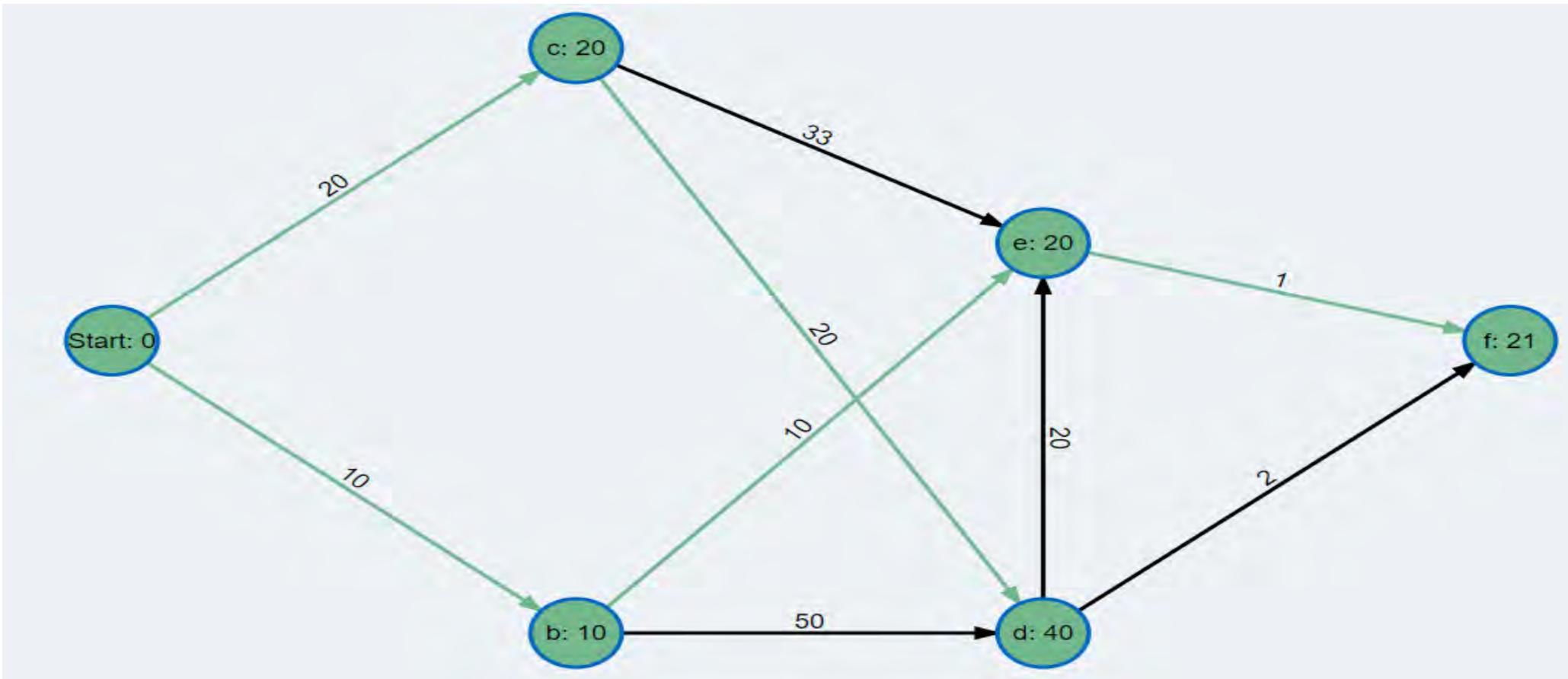


**Note:** When source or destination is not mentioned, you can assume any one as source and destination for Dijkstra and Bellman ford respectively.

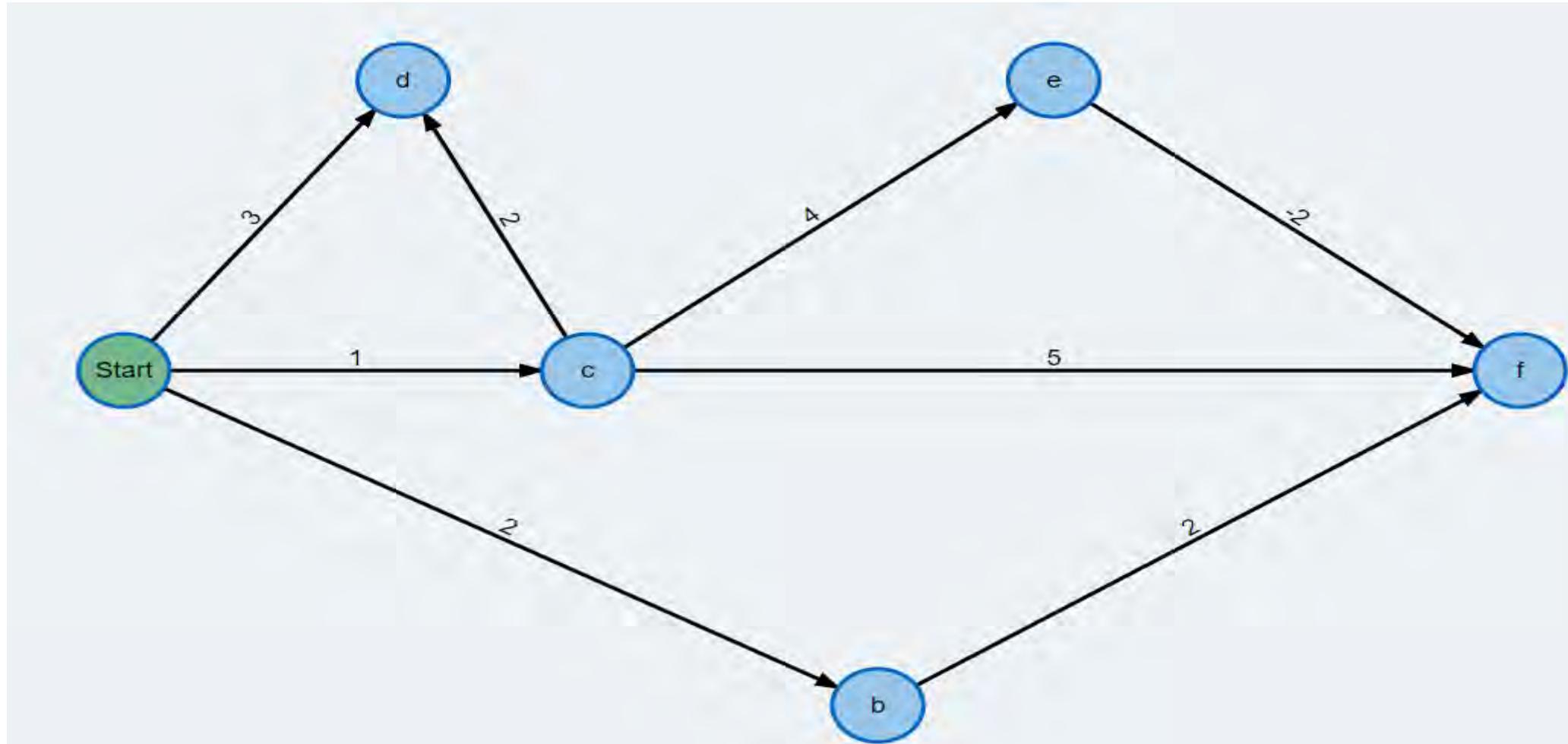
**HYU-7:** Use Dijkstra's Algorithm to find Shortest path from **a** to all other nodes.



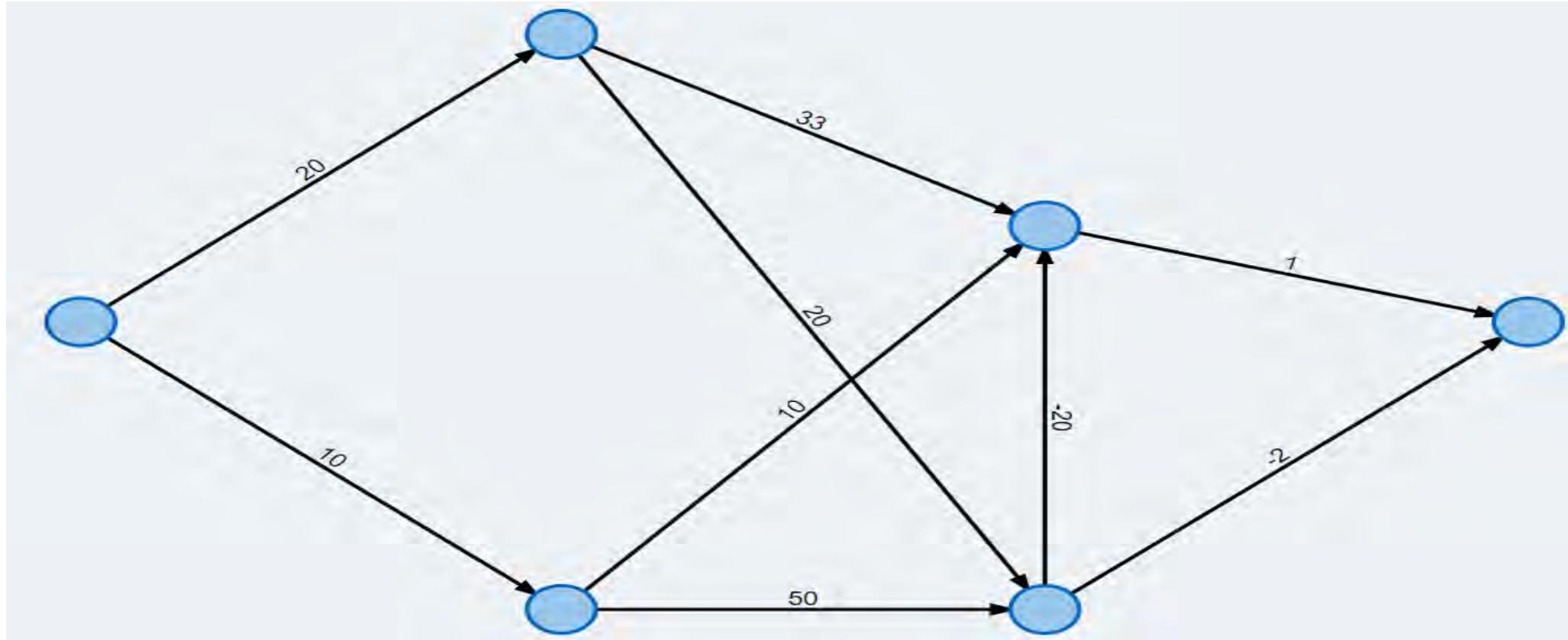
# Final Answer



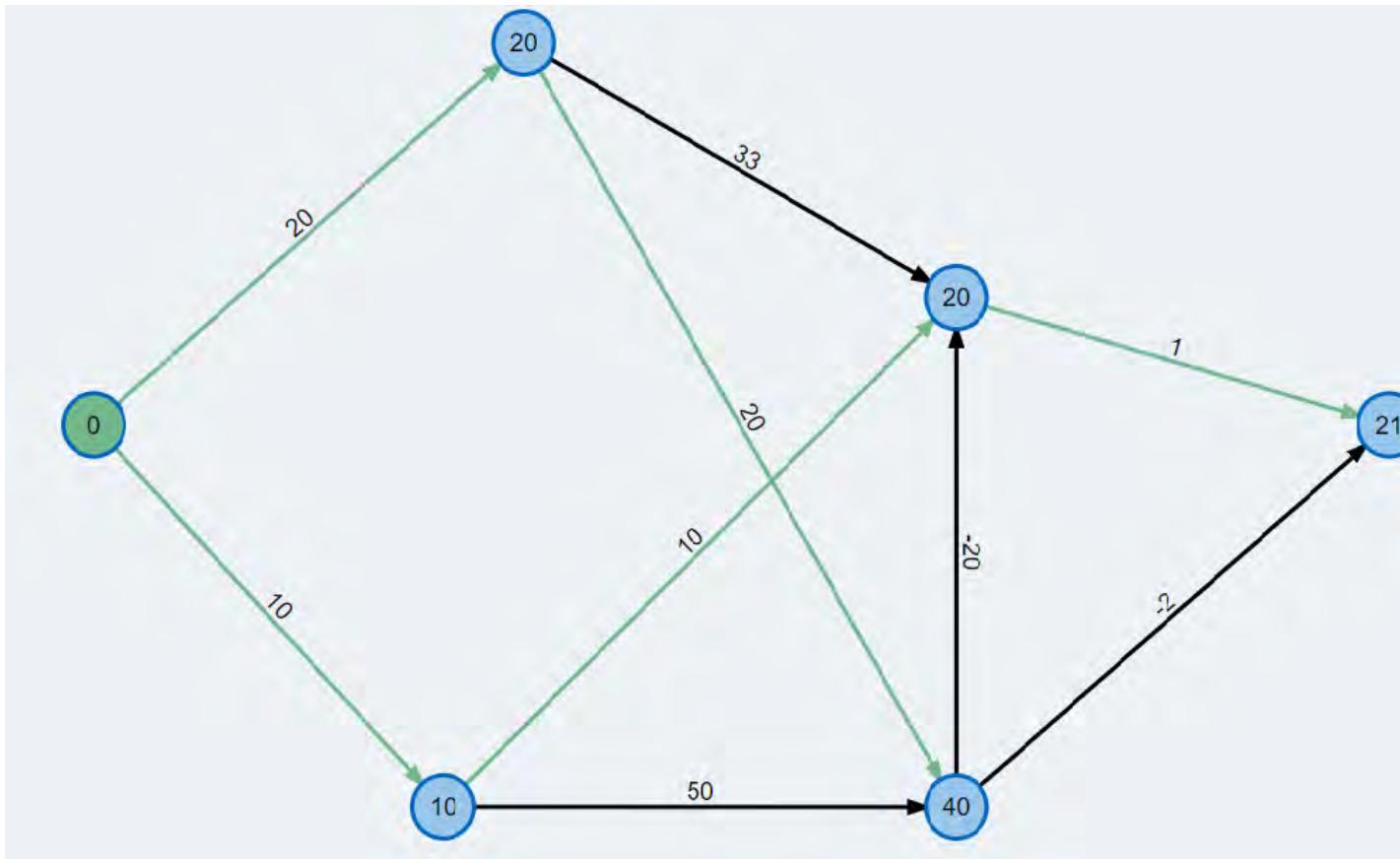
# HYU-8: Find Shortest path using Dijkstra's and Bellman Ford algorithm.



HYU-9: Use Bellman Ford to find shortest path from leftmost node to all others.



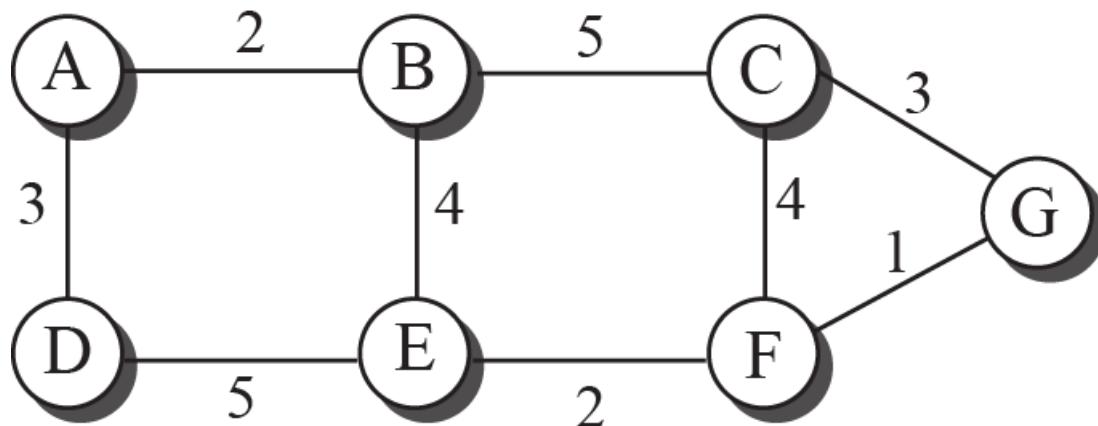
# Final Answer



A	0
B	2
C	$\infty$
D	3
E	$\infty$
F	$\infty$
G	$\infty$

A	2
B	0
C	5
D	$\infty$
E	4
F	$\infty$
G	$\infty$

A	$\infty$
B	5
C	0
D	$\infty$
E	$\infty$
F	4
G	3



A	$\infty$
B	$\infty$
C	3
D	$\infty$
E	$\infty$
F	1
G	0

A	3
B	$\infty$
C	$\infty$
D	0
E	5
F	$\infty$
G	$\infty$

A	$\infty$
B	4
C	$\infty$
D	5
E	0
F	2
G	$\infty$

A	$\infty$
B	$\infty$
C	4
D	$\infty$
E	2
F	0
G	1

New B	Old B	A
A 2	A 2	A 0
B 0	B 0	B 2
C 5	C 5	C $\infty$
D 5	D $\infty$	D 3
E 4	E 4	E $\infty$
F $\infty$	F $\infty$	F $\infty$
G $\infty$	G $\infty$	G $\infty$

$B[ ] = \min(B[ ], 2 + A[ ])$

a. First event: B receives a copy of A's vector.

**Note:**  
 $X[ ]$ : the whole vector

New B	Old B	E
A 2	A 2	A $\infty$
B 0	B 0	B 4
C 5	C 5	C $\infty$
D 5	D 5	D 5
E 4	E 4	E 0
F 6	F $\infty$	F 2
G $\infty$	G $\infty$	G $\infty$

$B[ ] = \min(B[ ], 4 + E[ ])$

b. Second event: B receives a copy of E's vector.

**Example 5:** Calculate the checksum for IPv4 header shown below:

4	5	0	28
	1	0	0
4	17	0	
10.12.14.5			
12.6.7.9			

Checksum ?

**Problem:** Calculate the checksum for IPv4 header shown below:

4	5	0	28
1		0	0
4	17	0	
10.12.14.5			
12.6.7.9			
4, 5, and 0	4	5	0
28	0	0	C
1	0	0	I
0 and 0	0	0	0
4 and 17	0	4	I
0	0	0	0
10.12	0	A	C
14.5	0	E	5
12.6	0	C	6
7.9	0	7	9
Sum	7	4	4 E
Checksum	B	B	B T

# **End of Module 4**

# CN (IT-3001)

## Link Layer: DLC Sublayer

Prof. Amit Jha  
School of Electronics Engineering (SOEE)  
KIIT Deemed to be University

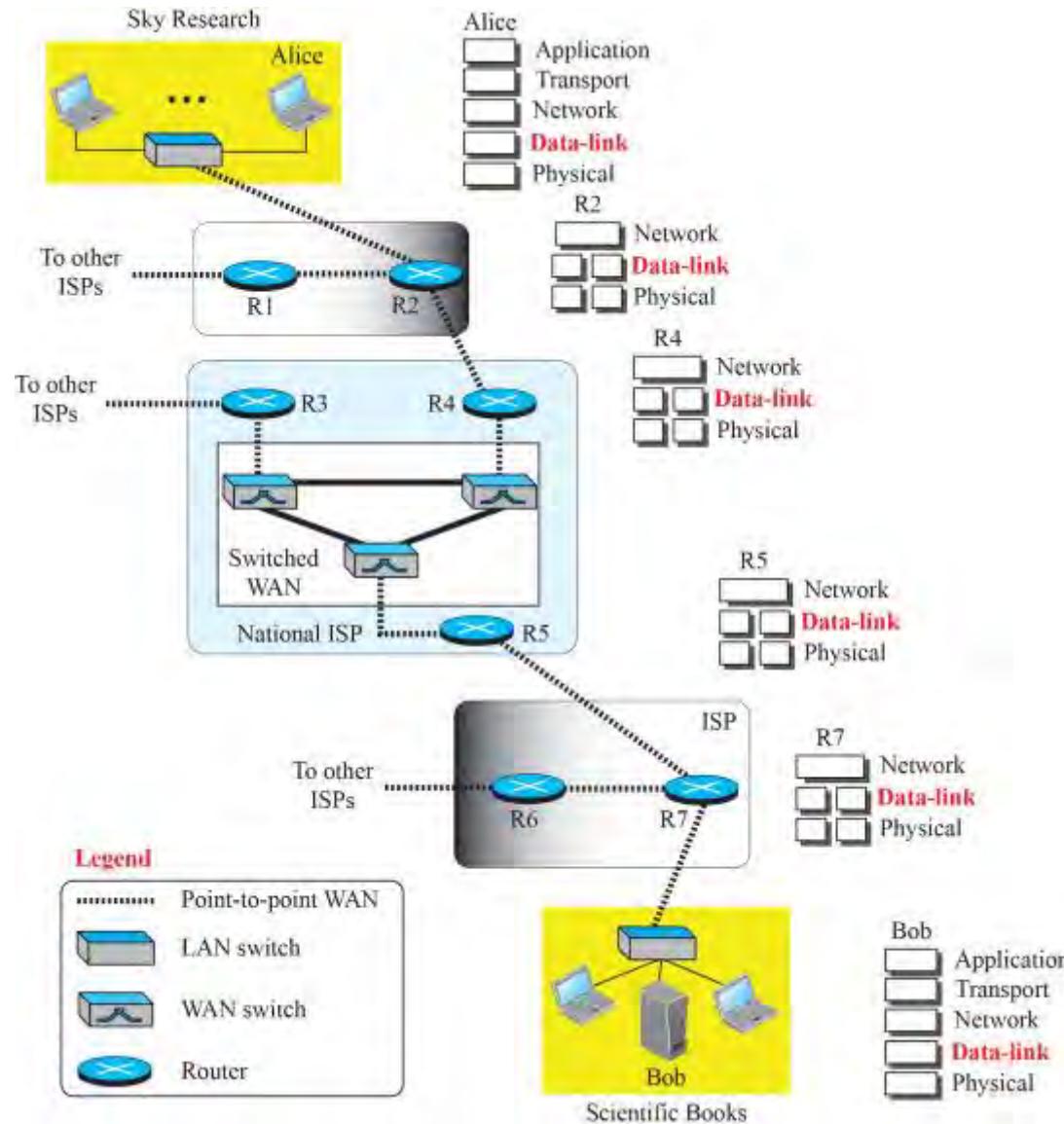


**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose. Images portrayed (if any) are not to hurt the sentiments of any person.

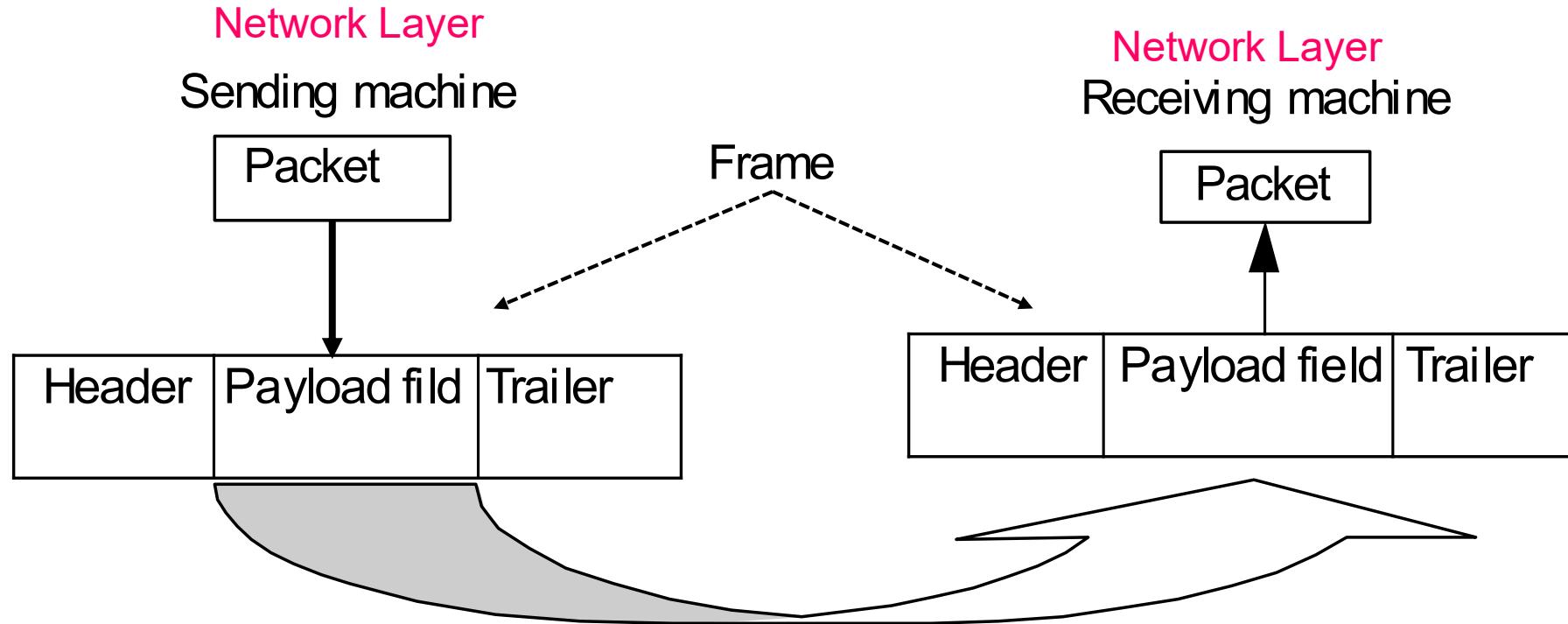
# Objective

- To understand the introduction and services of Data Link Layer
- Data Link Control
  - Error Detection and Correction
  - Framing
- Multiple Access
- Ethernet Frame Format and Addressing
- GIGABIT Ethernet
- Link Layer Switching and VLANs

# Data Link Layer: An Overview



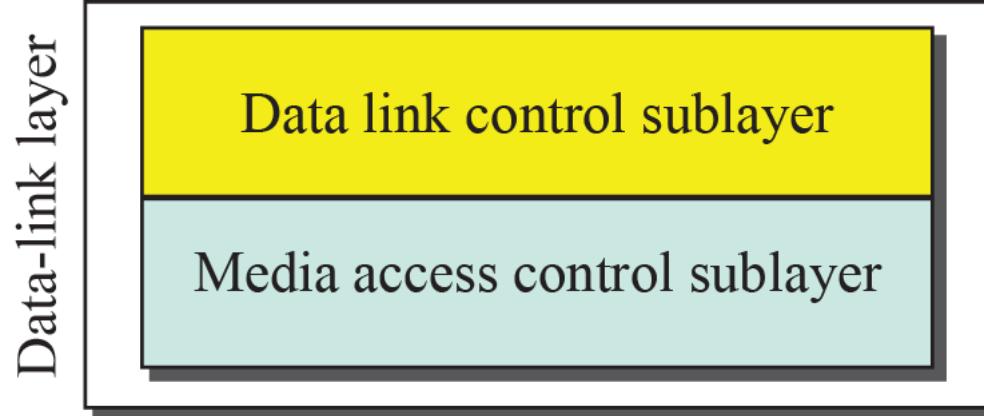
# “Packet” and “Frame” relationship



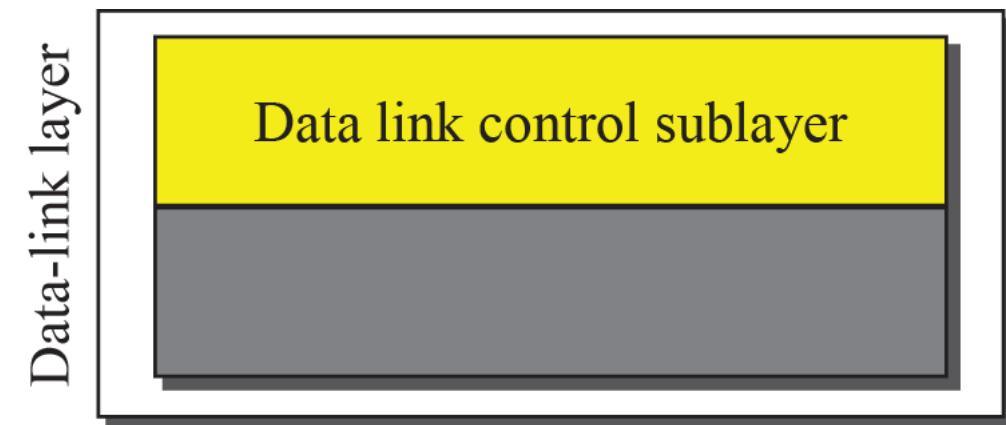
In some cases, functions of error control and flow control are allocated in transport or other upper layer protocols and not in the DLL, but principles are pretty much the same.

# Two Sublayers of the Data-Link-Layer

- The data link layer is divided into two sublayers as shown below.
  1. Data Link Control (DLC) sublayer
  2. Media Access Control (MAC) Sublayer

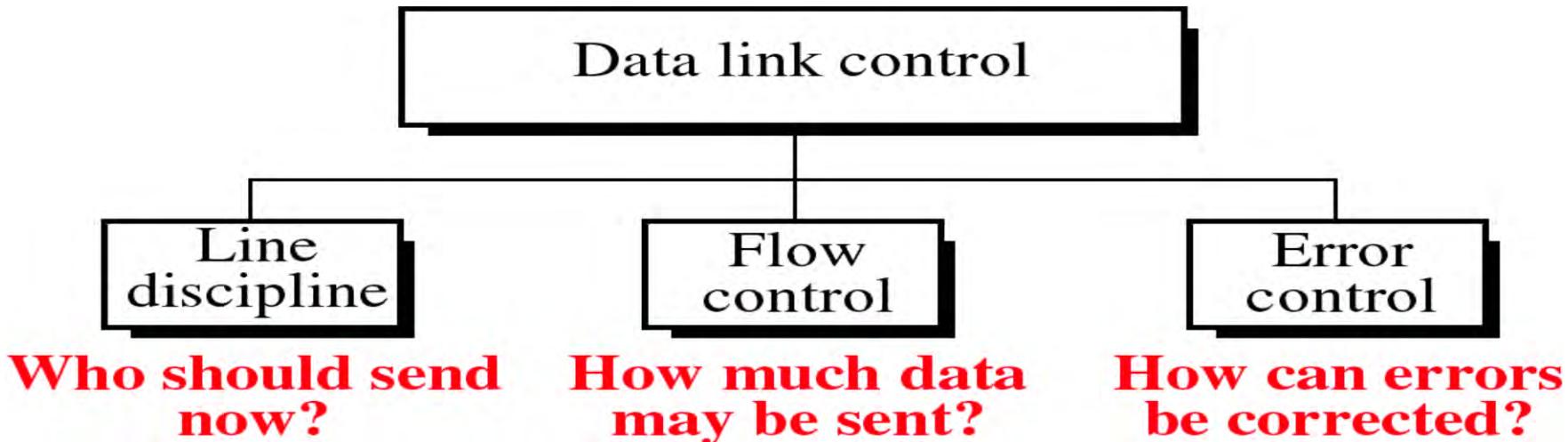


a. Data-link layer of a broadcast link



b. Data-link layer of a point-to-point link

# Responsibilities of Data Link Layer



- Specific responsibilities of the data link layer include ***framing, addressing, flow control, error control, and media access control.***
- It provides service interface to the network layer.
- The data link layer adds a **header** to the frame to define the addresses of the sender and receiver of the frame.

# Responsibilities of DLL

- *Framing:*
  - Data link layer deals data in chunks generally called **Frames**.
  - Size of frame is typically a few hundreds/thousands of bytes.
  - The data link layer is concerned with local delivery of frames between devices on the same LAN.
  - It creates/recognizes **frame boundaries** i.e., start and stop of the frame in order to distinguish between them.

# Responsibilities of DLC and MAC sublayer of DLL

*DLC sublayer is responsible for →*

1. *Framing,*
2. *flow and Error control, and*
3. *error detection and correction.*

*MAC sublayer is responsible for →*

1. *Channel access*
2. *Media control*

- *The several DLC protocol includes*
  - *HDLC*
  - *ATM*
  - *Frame Relay*

# Framing

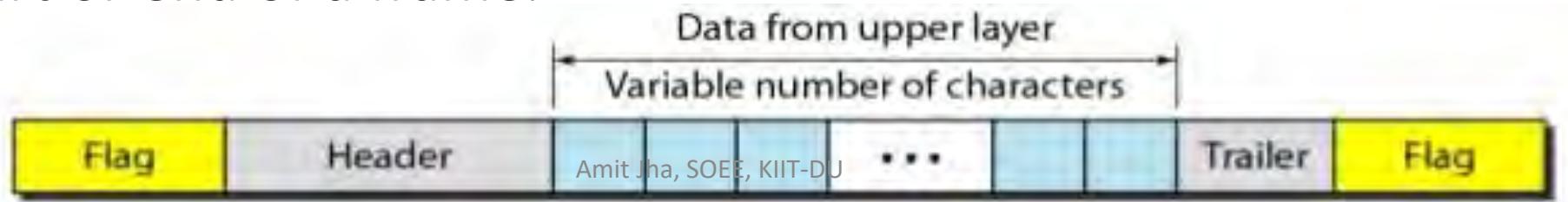
- Data link layer converts stream of bits (from physical layer) into frames.
- Each frame is made distinguishable from one another by appending source and destination address.
- Although, the **whole message can be packed in one frame**, that is not normally done. This is because, a very large frame will not make **flow and error control** very efficient.
- For e.g., even for a **single bit error**, we need to **retransmit** the complete frame.

# Frame Size

- Size of the frame can be either fixed or variable.
- **Fixed-Size Framing:** In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. An example of this type of framing is the **ATM wide-area network**, which uses frames of fixed size called cells.
- **Variable-Size framing:** It is the most important as it is widely used in **local-area networks**. In variable-size framing, we need a way to define boundaries, i.e., beginning and end of the frames.
  - Historically, two approaches were used for this purpose:
    - 1) Character-oriented approach
    - 2) Bit-oriented approach

# Variable-Size Framing: Character-Oriented Protocol

- In character oriented protocol, data to be carried are 8-bit characters from a coding system such as ASCII.
- The **header** which consists of source and destination addresses and other control information, and the **trailer**, which carries error detection and correction redundant bits, are also multiple of 8 bits.
- To separate one frame from the next, an 8-bit(1-byte) flag is added at the **beginning** and the **end** of the frame.
- The flag composed of protocol dependent special characters, signals that start or end of a frame.



- **A challenge:** Flag can be any thing which is not used in the data to be sent using DLL. This is well when we use only text as a data to be sent. But in general, we are sending audio, video, images, etc. as data which can have the same pattern as used for flag.

**Solution:** To fix this problem, we use byte stuffing (or character stuffing).

- **Byte stuffing:** The data section is stuffed with an extra byte, called escape character (ESC), whenever there is a character with the same pattern as flag in the data. ESC character has a predefined bit pattern.

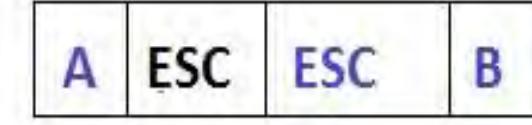
Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data, not a delimiting flag.

# Byte Stuffing

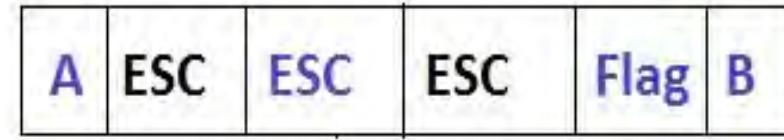
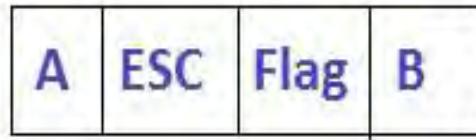
Case:1



Case:2



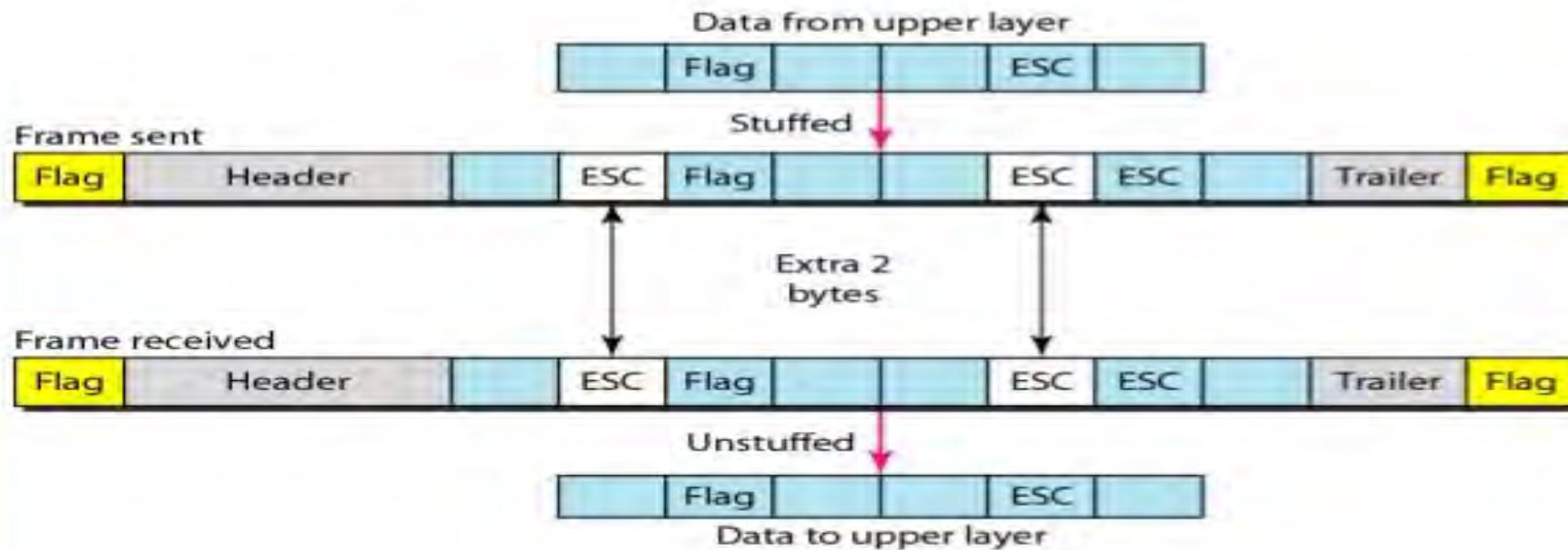
Case:3



Note: Byte stuffing is the process of adding 1 extra byte whenever there is a flag or escape character in the text.

# Review of Byte stuffing

## ***Byte stuffing and unstuffing***



E.g. of byte stuffing when data to be transmitted is: WE WILL NOT ESCAPE FROM THE QUIZ FOR THE FLAG HOSTING

*Data to be sent*

WE WILL NOT ESCAPE FROM THE QUIZ FOR THE FLAG HOISTING



*Frame sent*

FLAG	HEADER	WE WILL NOT <b>ESCAPE</b> ESCAPE FROM THE QUIZ FOR THE <b>ESCAPE</b> FLAG HOISTING	TRAILER	FLAG
------	--------	--	---------	------

*Frame received*

FLAG	HEADER	WE WILL NOT <b>ESCAPE</b> ESCAPE FROM THE QUIZ FOR THE <b>ESCAPE</b> FLAG HOISTING	TRAILER	FLAG
------	--------	--	---------	------



*Data after unstuffing*

WE WILL NOT ESCAPE FROM THE QUIZ FOR THE FLAG HOISTING

- Is ASCII code sufficient for all the characters used today in Internet?



- Is ASCII code sufficient for all the characters used today in Internet?

Ans: *No, because ASCII code is of 7-bits, and using these 7-bits, we can represent only 128 characters.*



*So, most of the time, we use UNICODE which is of 16-bits and 32-bits.*

*So, we need to move to other stuffing techniques.....*

# Variable-Size Framing: Bit-Oriented Protocol

- In a bit-oriented protocol, the data section of a frame is a sequence of bits to be interpreted by the upper layer as text, graphic, audio, video, etc.
- Apart from header and trailer, we still need a delimiter, called flag to separate one frame from other.
- Most of the protocols use a special 8-bit pattern flag **01111110** as the delimiter to define beginning and end of the frame.

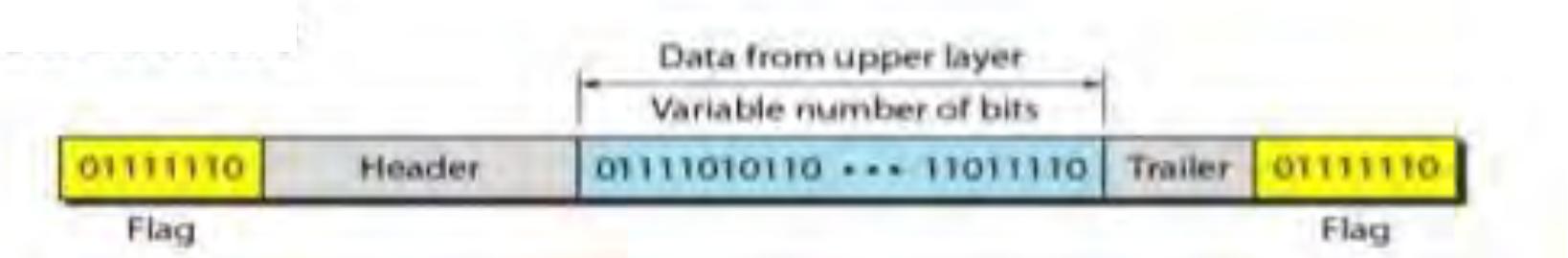
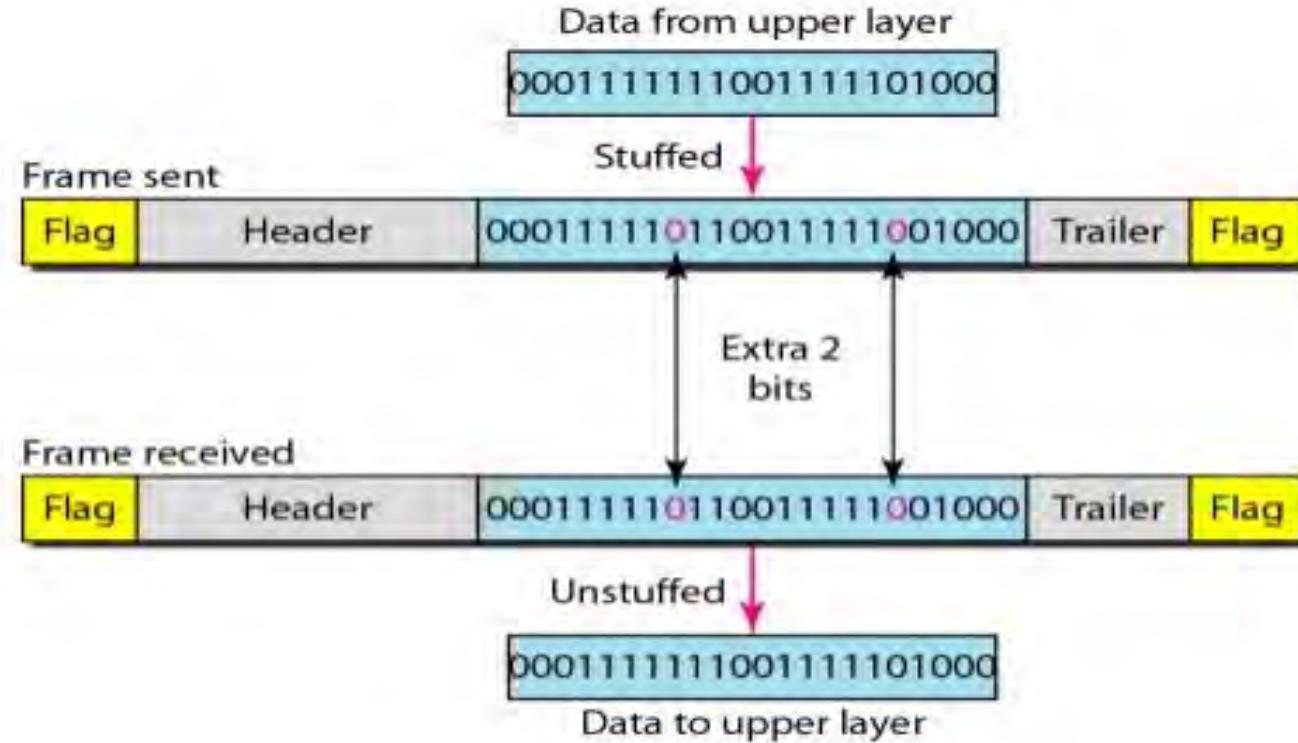


Fig. A frame in bit-oriented protocol

- **A Challenge:** The same type of problem (as happened in byte-oriented protocol) can occur here, if the flag pattern occurs in the data.
- **Sol:** *This problem is solved here by stuffing a single bit (instead of one byte in character oriented protocol) to prevent the pattern looking like a flag. This process is called bit-stuffing.*
- **Bit stuffing:** In this process, one extra 0 is added whenever five consecutive 1s follow a 0 in the data. This extra stuffed bit is eventually removed from the data by receiver.

# Bit stuffing and unstuffing



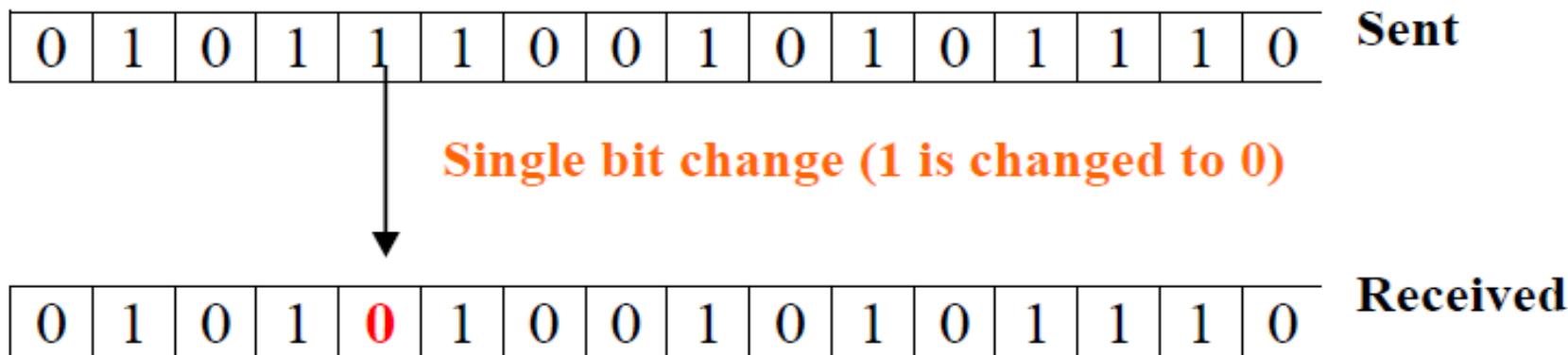
**Note:** A **bit-oriented protocol** is actually implemented by using the High-level Data Link Control (**HDLC**) Protocol. Whereas a popular byte-oriented protocol is implemented by using Point-to-Point Protocol (**PPP**).

**Error Control → Error Detection**

- Error control requires because
  - there are some applications which can tolerate certain level of error like video and voice applications.
  - But, there are some applications which can not tolerate error of any level like data applications.
- In data communication, **there are two types of errors:**
  - 1) *Single-Bit Error*
  - 2) *Burst Error*
- Error control has two phases:
  - 1) Error detection &
  - 2) Error correction

# Types of Error: Single-Bit Error

- In a single-bit error, only 1 bit in the data unit has changed.

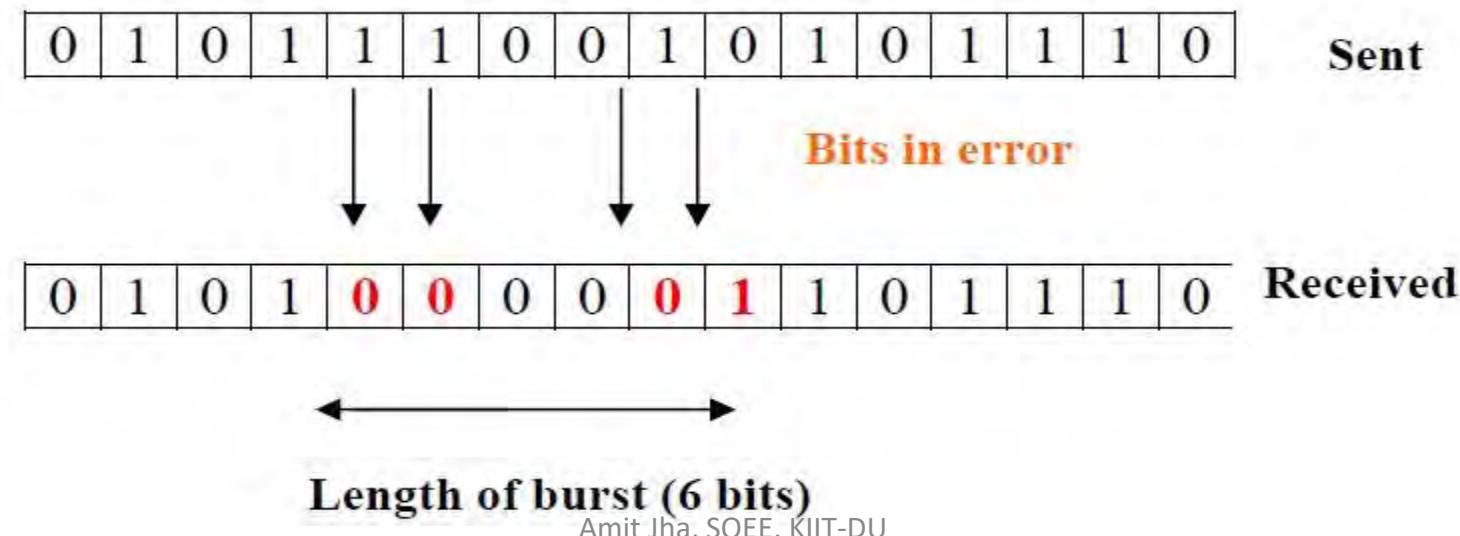


- Single-bit errors are the **least likely** type of error in **serial data transmission**.
- To understand why, imagine data sent at 1 Mbps. This means that each bit lasts only  $1/1,000,000$  s, or 1  $\mu$ sec.
- For a single-bit error to occur, the **noise must have a duration of only 1  $\mu$ sec**, which is very rare; noise normally lasts much longer than this.

- However, a single-bit error can happen if we are having a parallel data transmission.
- For example, if 16 wires are used to send all 16 bits of a word at the same time and one of the wires is noisy, one bit is corrupted in each word.

# Types of Error: Burst Error

- The term ***burst error*** means that 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1.
- Note that burst error doesn't necessary means that error occurs in consecutive bits.
- The length of the burst error is measured from the first corrupted bit to the last corrupted bit. Some bits in between may not be corrupted.



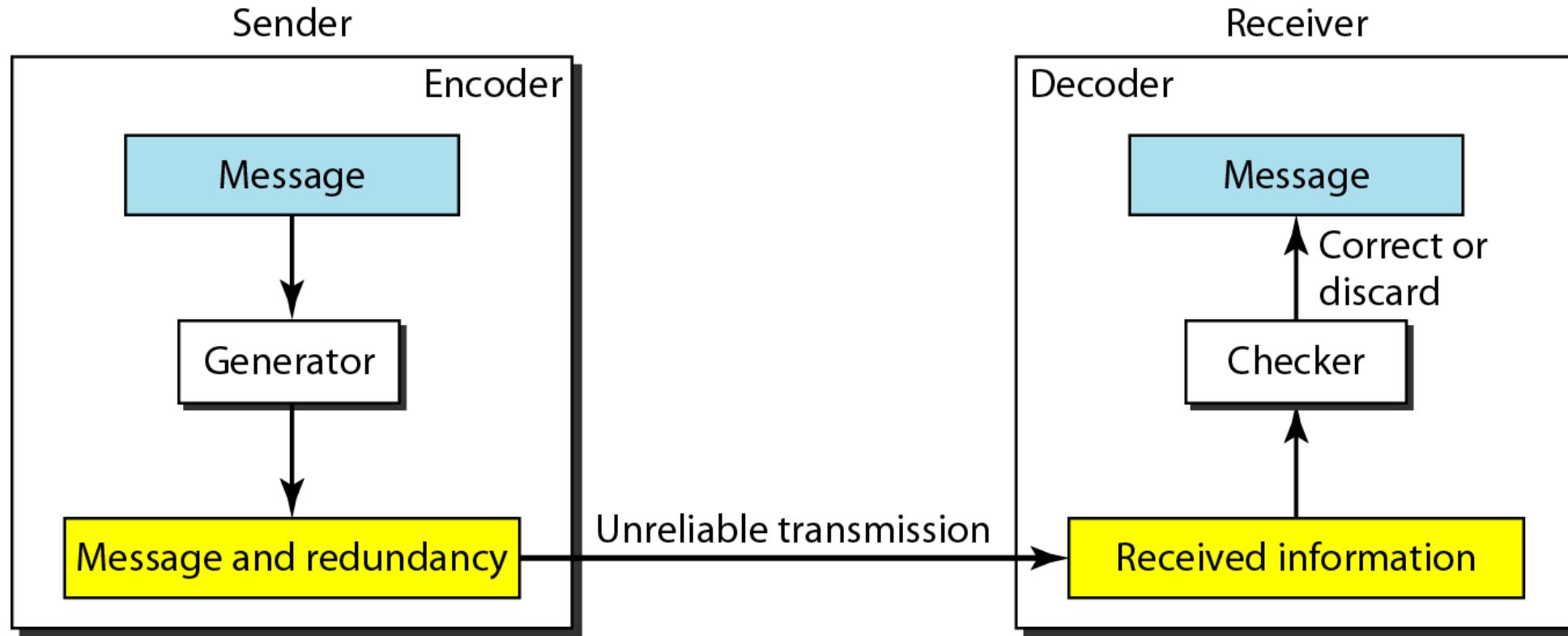
- A burst error is more likely to occur than a single-bit error.
- This is because, the duration of noise is generally more than the duration of 1 bit, which means that when noise affects data, it affects a set of bits.
- The number of bits affected depends on the data rate and duration of noise.
- For e.g., if we are sending data at 1kbps, a noise of 1/100 sec duration can affect 10 bits.

Let, A be **error detection** and B be **error correction**, then which of the following holds true?

- a) A is easier than B
- b) A is more difficult than B
- c) A is easier than B for less than 10 bits of data
- d) A is more difficult than B for less than 10 bits of data

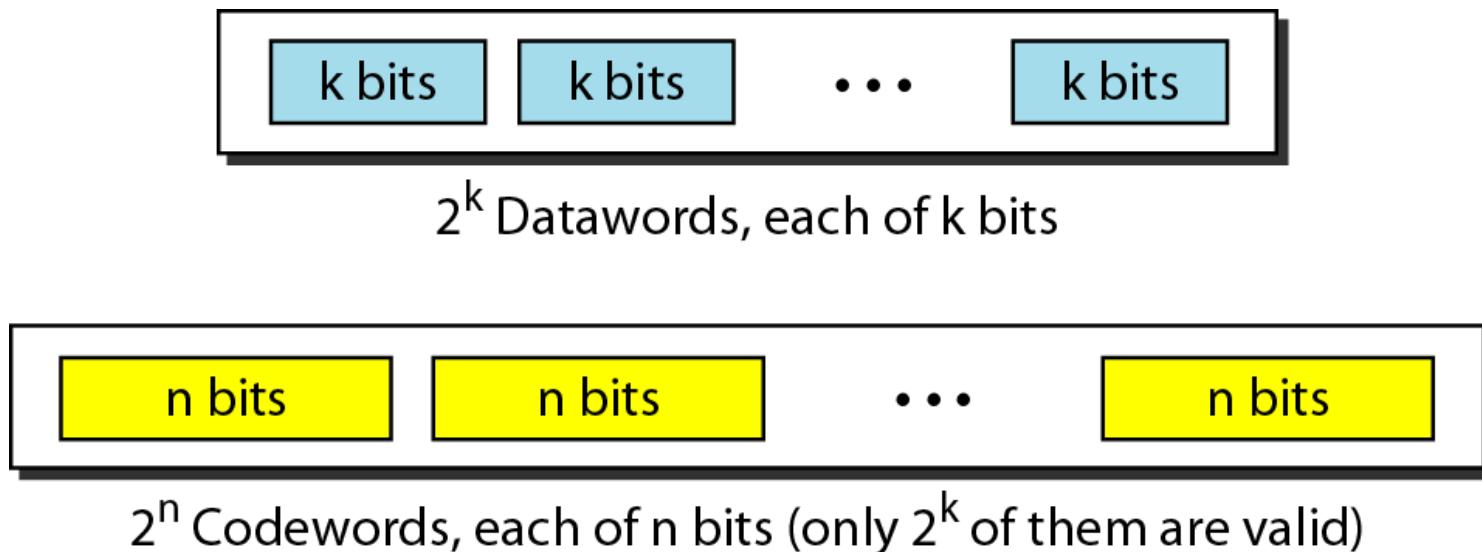


# *The structure of encoder and decoder*



# Introduction to Block Code

- In block coding, we divide our message into blocks, each of  $k$  bits, called **datawords**. We add  $r$  redundant bits to each block to make the length  $n = k + r$ . The resulting  $n$ -bit blocks are called **codewords**.*



**Fig: Datawords and codewords in block coding**

## • **Block Coding: Key points**

- *In 4B/5B coding scheme,  
 $k = 4$  and  $n = 5$ .*
- *As we saw, we have  $2^k = 16$  datawords and  $2^n = 32$  codewords. We saw that 16 out of 32 codewords are used for message transfer and the rest are either used for other purposes or unused.*

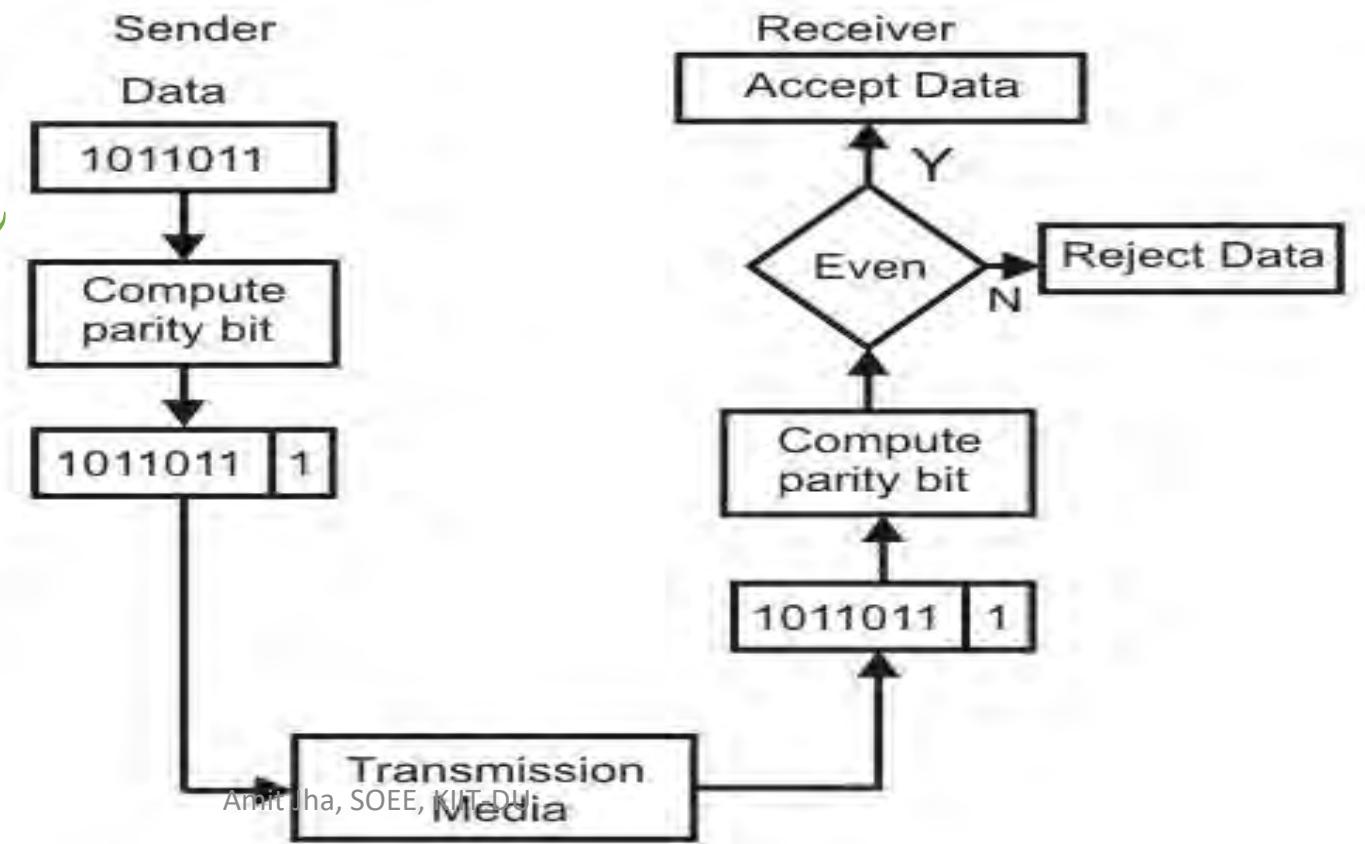
# Error Detecting Codes

- The central concept in detecting or correcting errors is ***redundancy***, which means adding some extra bits along with data.
- The sender adds some redundant bits, whereas receiver removes it.
- Redundant bits (extra bits) facilitate detection and corrections of errors.
- **Popular techniques** for error detection are:
  1. Simple parity check
  2. Two-dimensional parity check
  3. Checksum
  4. Cyclic redundancy check

# Error Detecting Codes: Simple parity check

- It is also known as **one-dimensional parity check**.
- In this technique, a redundant bit called **parity bit**, is appended to every data unit so that number of 1's in the unit (including the parity bit) becomes even.

Even parity checking scheme



- *At the Transmitter side*, parity bit generator adds
  - 1 if the data block contains odd number of 1's
  - 0 if the data block contains even number of 1's
- *At the receiver side*, parity bit is computed from the received data bits.
- If the receiver finds odd number of 1's in received data, then it ensures that an error has occurred.

- For an e.g., the complete list of data words (for 4-bit) and the corresponding code words are given below:

<b>Decimal value</b>	<b>Data Block</b>	<b>Parity bit</b>	<b>Code word</b>
0	0000	0	00000
1	0001	1	00011
2	0010	1	00101
3	0011	0	00110
4	0100	1	01001
5	0101	0	01010
6	0110	0	01100
7	0111	1	01111
8	1000	1	10001
9	1001	0	10010
10	1010	0	10100
11	1011	1	10111
12	1100	0	11000
13	1101	1	11011
14	1110	1	11101
15	1111	0	11110

# Key points

- It is also possible to use ***odd-parity*** checking, where the number of 1's should be odd.
- From the last table it can be observed that
  - If we move from one code word to another, at least 2 data bits should be changed.
  - Thus, these set of code words are said to have a minimum distance (***hamming distance***) of 2.
  - So, if there exists one bit error then receiver will be able to detect it, but it can't detect two bit error.
  - This is because, a code word with **two bits error** again becomes a valid member of the set.
- For a linear code,

$$\text{number of errors detected} = d_{\min} - 1$$

where,  $d_{\min}$  = minimum hamming distance

- The *Hamming distance* between two words is the number of differences between corresponding bits.

1) The Hamming distance  $d(000, 011)$  is 2 because

$$000 \oplus 011 \text{ is } 011 \text{ (two 1s)}$$

2) The Hamming distance  $d(10101, 11110)$  is 3 because

$$10101 \oplus 11110 \text{ is } 01011 \text{ (three 1s)}$$

- The *minimum Hamming distance* is the smallest Hamming distance between **all possible** pairs in a set of words.

**Q)** Find the minimum hamming distance for the following codewords.

<i>Datawords</i>	<i>Codewords</i>
00	000
01	011
10	101
11	110

- Sol:

*We first find all Hamming distances.*

$$\begin{array}{llll} d(000, 011) = 2 & d(000, 101) = 2 & d(000, 110) = 2 & d(011, 101) = 2 \\ d(011, 110) = 2 & d(101, 110) = 2 & & \end{array}$$

*The  $d_{min}$  in this case is 2.*

**Q) Find minimum hamming distance for the codewords given in the following table.**

<i>Dataword</i>	<i>Codeword</i>
00	00000
01	01011
10	10101
11	11110

**Q) Find minimum hamming distance for the codewords given in the following table.**

<i>Dataword</i>	<i>Codeword</i>
00	00000
01	01011
10	10101
11	11110

**Ans: 3**

- *Will this method suffice our requirements of detecting errors?*

- *Will this method suffice our requirements of detecting errors?*

*Ans:* No, because it is able to detect only a single-bit error.

*Explanation:* Let us assume that you are sending **01100**, but because of noise, it has been received as **10100**. In this scenario, receiver will not be able to identify the received data is in error because 10100 is a valid member of its code word set.

# Error Detecting Codes: Two-dimensional parity check

- In a two-dimensional parity check, the block of bits are organized in the form of a table.
- Parity check bits are calculated
  - For each row
  - as well as for each column
- Then, the complete table is sent in the form of block including data as well as both parity bits.
- At the receiver side, these are calculated with the parity bits calculated on the received data.
- The existence of odd number of 1's ensures that the data are corrupted.

# *Two-dimensional parity checking*

Original data

10110011 : 10101011 : 01011010 : 11010101

Column parities

1	0	1	1	0	1	1	1
1	0	1	0	1	1	1	1
0	1	0	1	1	0	1	0
1	1	0	1	0	1	1	1
<hr/>							
1	0	0	1	0	1	1	1

Row parities

101100111 : 101010111 : 010110100 : 110101011 : 100101111

Data to be sent

# Key points

- Two-dimension parity checking increases the likelihood of detecting burst error.
- A 2-D parity check of  $n$  bits can detect a burst error of  $n$  bits.

- *Will this method suffice our requirements of detecting errors?*

- *Will this method suffice our requirements of detecting errors?*

*Ans: No....*

*Because if two bits in one data unit is damaged and two bits in another data unit is damaged at exactly same position, the 2-D parity checker will not detect any error.*

***Explanation:** Let us assume that data to be sent after parity is  
101100111 : 101010111 : 0101101100 : 110101011 : 100101111*

*But due to noise we receive the following data:*

**101100111 : 001110111 : 0101101100 : 110101011 : 000001111**

# Error Detecting Codes: Checksum

- In checksum error detection scheme, the data is divided into  $k$  segments each of  $m$  bits.
- Traditionally, the Internet has been using a 16-bit checksum, called *Internet checksum*.

# Concept behind *checksum*

- Its very simple.....
- Suppose our data is a list of five 4-bit numbers that we want to send to a destination.
  - In addition to sending these numbers, we send the sum of the numbers. For example, if the set of numbers is (7, 11, 12, 0, 6), we send (7, 11, 12, 0, 6, 36), where 36 is the sum of the original numbers. The receiver adds the five numbers and compares the result with the sum.
  - If the two sums are same then receiver assumes no error and accepts the data; otherwise discard it.
- Now, we can further reduce the burden of receiver as follows:
  - Instead of sending sum, we send negative of sum, called ***checksum***. In this case, we send (7, 11, 12, 0, 6, -36). The receiver adds all the numbers received (including the checksum). If result is 0, it assumes no error, otherwise there is an error.

# How to calculate the Internet Checksum

- The sender calculates the checksum as follows:
  - The message is divided into **16-bit words**.
  - The value of checksum word is set to 0.
  - All words including the checksum are added using one's complement addition.
  - The sum is complemented and becomes the checksum.
- The checksum segment is sent along with the data segments.
- The receiver uses the following steps to detect error:
  - The message (including checksum) is divided into 16-bit words.
  - All words are added using **one's complement addition**.
  - The sum is complemented and becomes the new checksum.
  - If the value of checksum is 0, the message is accepted; otherwise it is rejected.

**Ex.1:** For the given data, **10110011101010110101101011010101**, calculate the checksum by dividing the data into words of 8 bits. Also, find whether the error has occurred or not, if the received data is

- a) **10110011101010110101101011010101**
- b) **10110011101010110101101011010011**

- Sol: for part a)

Since, 1's complement addition

$$\begin{array}{r}
 \text{Word size} = 8\text{bits} \\
 \text{k=4, m=8} \\
 \begin{array}{r}
 10110011 \\
 10101011 \\
 \hline
 01011110 \\
 1 \\
 \hline
 01011111 \\
 01011010 \\
 \hline
 10111001 \\
 11010101 \\
 \hline
 10001110 \\
 1 \\
 \hline
 \end{array} \\
 \text{Sum : } \frac{10001111}{\text{Checksum}} \\
 \text{Checksum } 01110000
 \end{array}$$

*Old checksum*

$$\begin{array}{r}
 \text{Received data} \\
 \begin{array}{r}
 10110011 \\
 10101011 \\
 \hline
 01011110 \\
 1 \\
 \hline
 01011111 \\
 01011010 \\
 \hline
 10111001 \\
 11010101 \\
 \hline
 10001110 \\
 1 \\
 \hline
 \end{array} \\
 \text{Sum: } \frac{11111111}{\text{Complement = 00000000}} \\
 \text{Conclusion = Accept data}
 \end{array}$$

- Solution of part b)

*Please try by your own*



- **Ex 2:** Calculate the checksum for a text of 8 characters (“Forouzan”). Assume that the checksum is of 16-bit, as used in Internet today. The ASCII code (in Hex) for the character used here is: F=0x46, o=0x6F, r=0x72, u= 0x75, z= 0x7A, a=0x61, n=0x6E.

- Sol:

1 0 1 3	Carries
4 6 6 F	(Fo)
7 2 6 . F	(ro)
7 5 7 A	(uz)
6 1 6 E	(an)
0 0 0 0	Checksum (initial)
8 F C 6	Sum (partial)
8 F C 7	Sum
7 0 3 8	Checksum (to send)

a. Checksum at the sender site

1 0 1 3	Carries
4 6 6 F	(Fo)
7 2 6 7	(ro)
7 5 7 A	(uz)
6 1 6 E	(an)
7 0 3 8	Checksum (received)
F F F E	Sum (partial)
F F F F	Sum
0 0 0 0	Checksum (new)

a. Checksum at the receiver site

**Example 5:** Calculate the checksum for IPv4 header shown below:

4	5	0	28
	1	0	0
4	17	0	
	10.12.14.5		
	12.6.7.9		

Checksum ?

**Problem:** Calculate the checksum for IPv4 header shown below:

4	5	0	28
1		0	0
4	17	0	
10.12.14.5			
12.6.7.9			
4, 5, and 0	4	5	0
28	0	0	C
1	0	0	I
0 and 0	0	0	0
4 and 17	0	4	I
0	0	0	0
10.12	0	A	C
14.5	0	E	5
12.6	0	C	6
7.9	0	7	9
Sum	7	4	4 E
Checksum	B	B	B T

**Ex 2:** Calculate the checksum for a text of 8 characters “I LIKE DCN”. Assume that the checksum is of 16-bit, as used in Internet today. The ASCII code (in Hex) for the character used here is: I=0x49, L=0x4C, K=0x4B, E=0x45, D=0x44, C=0x43, N=0x4E, space= 0x20, H=0x48, T=0x54, A=0x41.

Also, show that how will you verify that errors have been occurred if you receive the message as “I HATE DCN”.

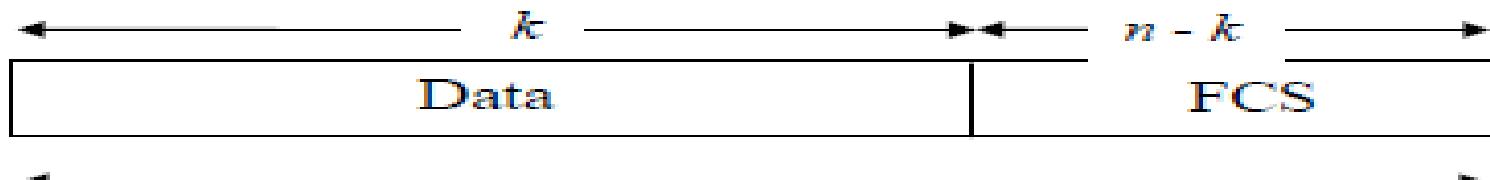
Please try by your own.....

# Internet checksum: Key points

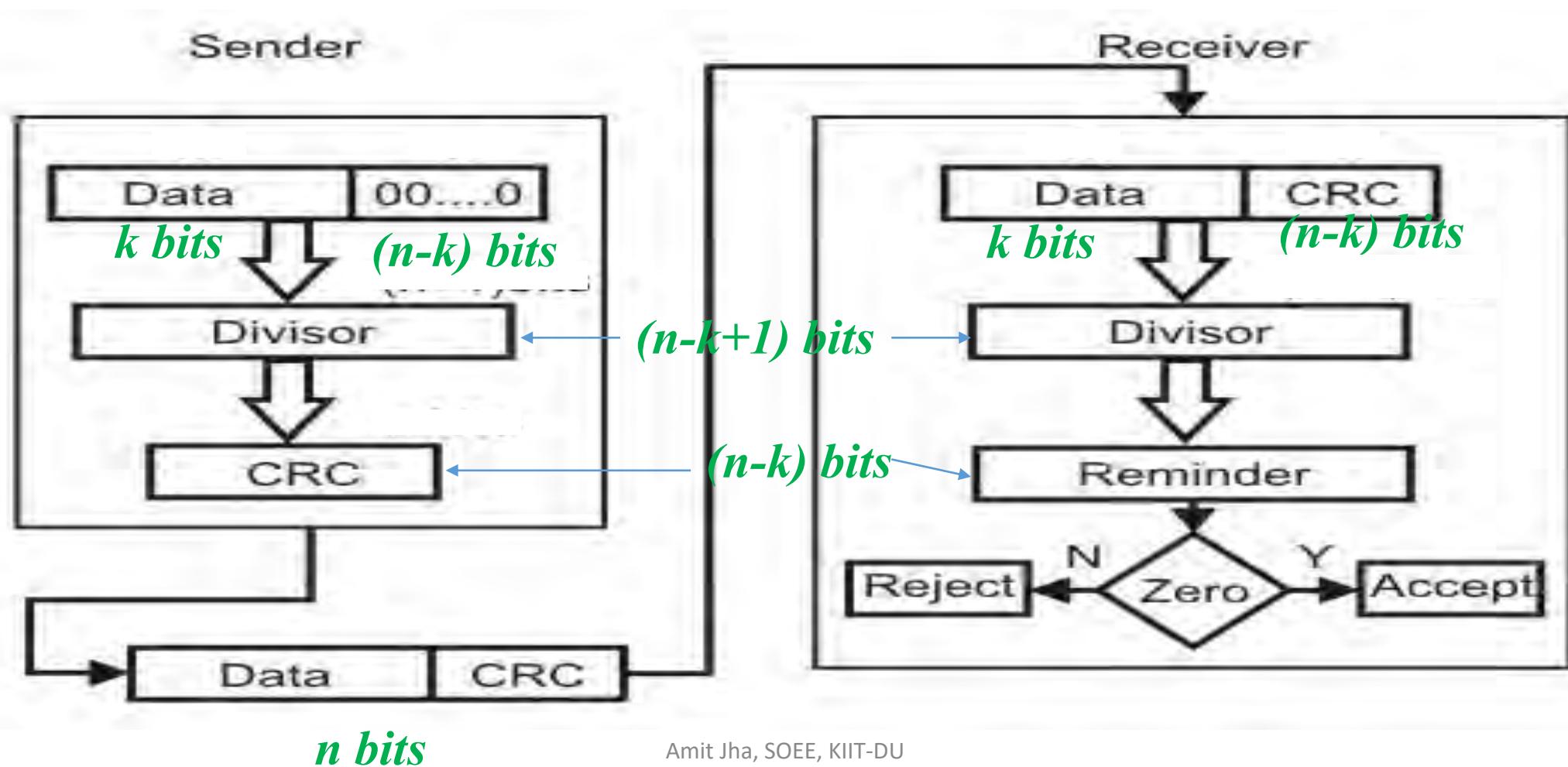
- It is used in several Internet protocols like **IP, TCP, UDP** to detect errors in IP header, or in the header and data for TCP/UDP.
- Generally, a checksum is calculated for header content and included in a special field.
- The checksum is calculated **at every router**.
- Overhead is less as compared to two-dimension parity check.
- **Disadvantage:** It will not be able to detect all errors if the **total sum remains constant**.

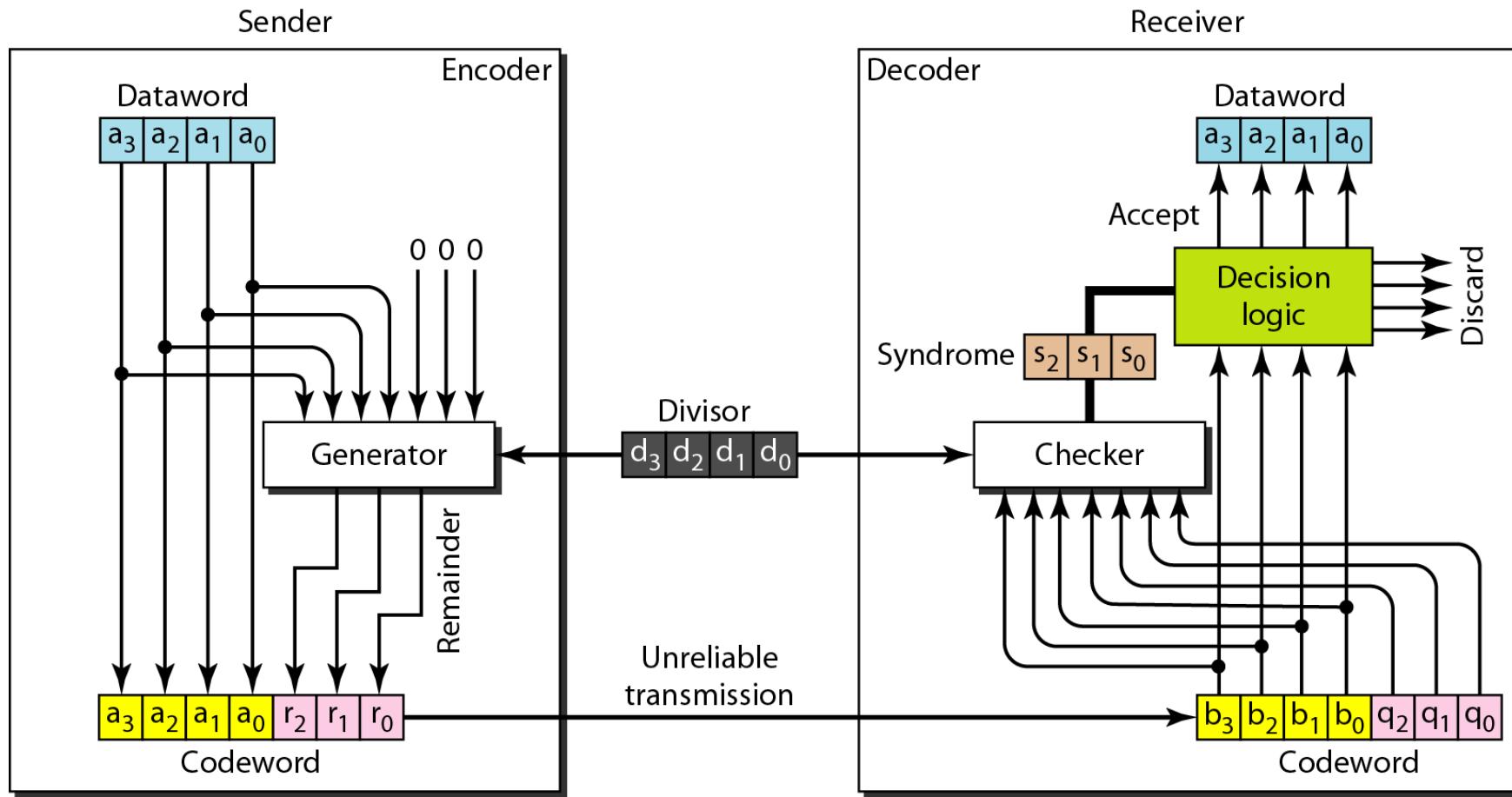
# Error Detecting Codes: Cyclic Redundancy Check (CRC)

- One of the **most powerful** and commonly used error detecting codes.
- Concept behind it.....
  - Given a  $k$ -bit block of bit sequence, the sender generates an  $(n-k)$  bit sequence, known as *frame check sequence(FCS)*, so that the resulting frame, consisting of  $n$  bits, is exactly divisible by same *predetermined number* (which is of  $(n-k+1)$  bits).
  - The receiver divides the incoming frame by that number and, if there is no remainder, it assumes that there was no error.
- Frame check sequence is also known as *cyclic redundancy check bits*.



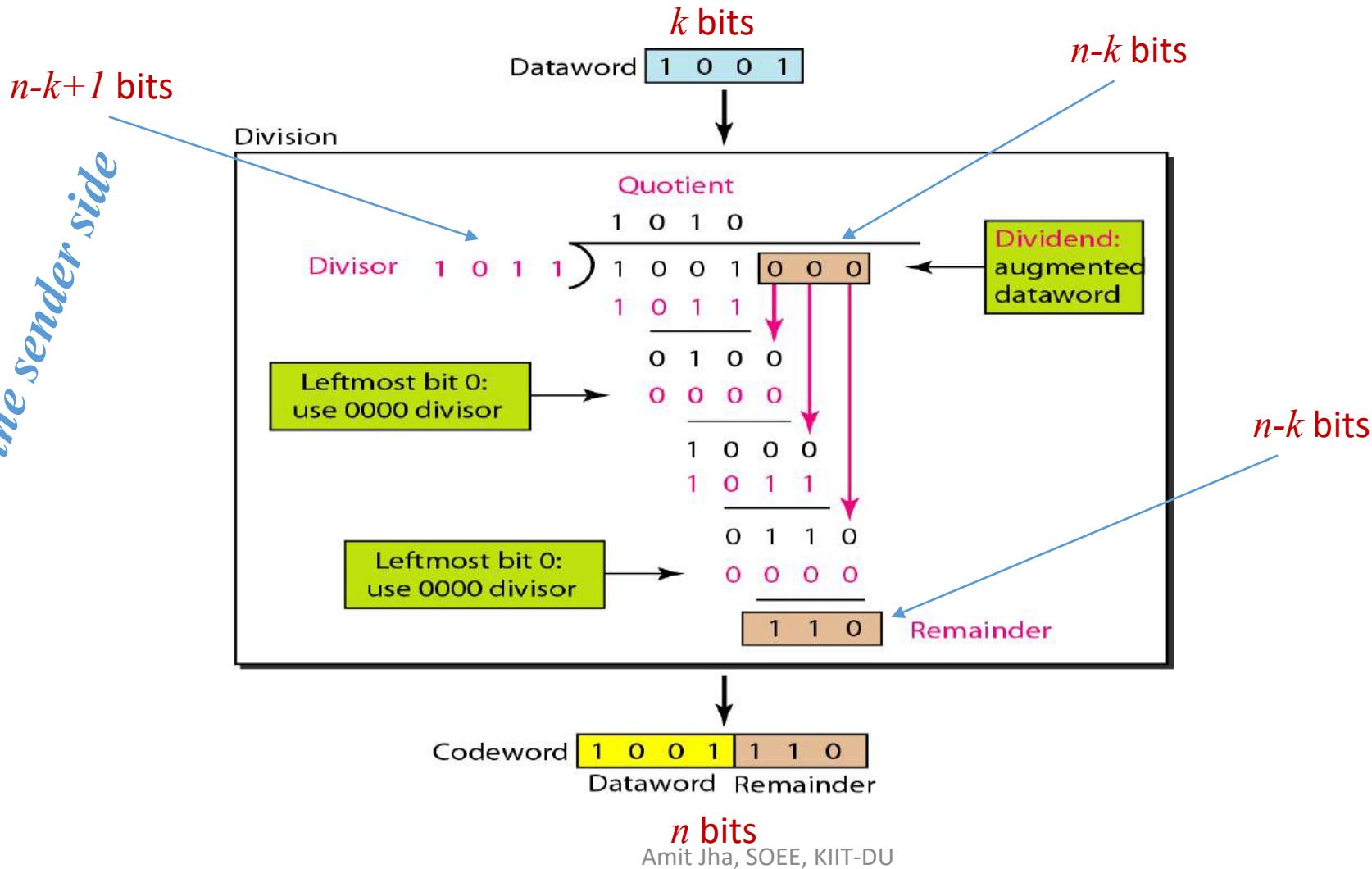
# Basic scheme for Cyclic Redundancy Checking



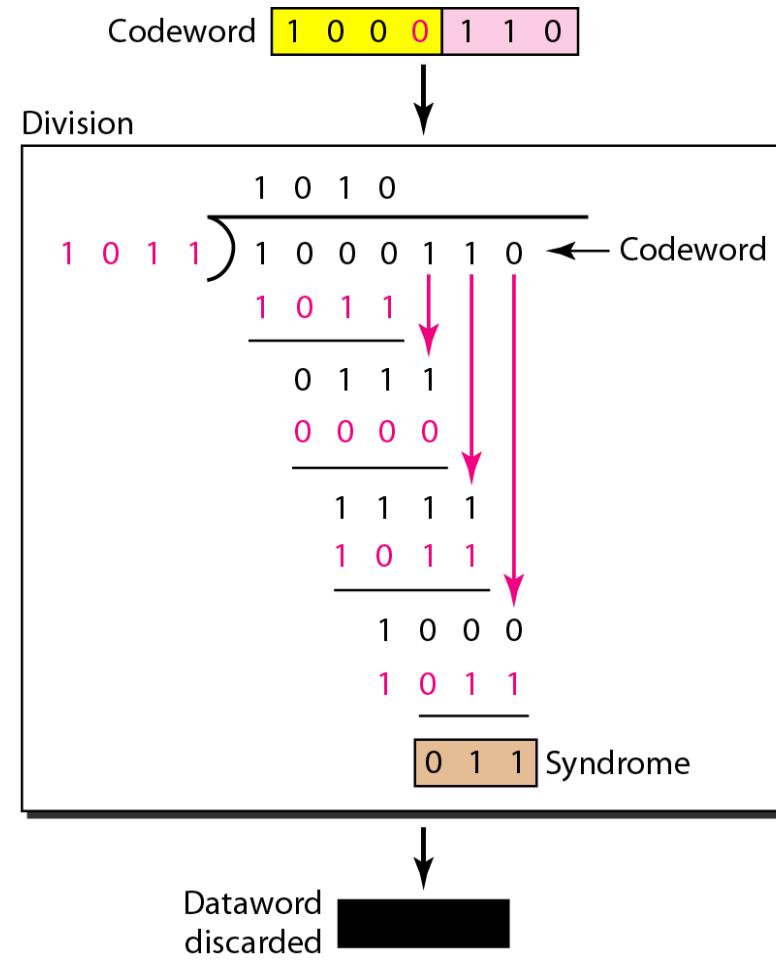
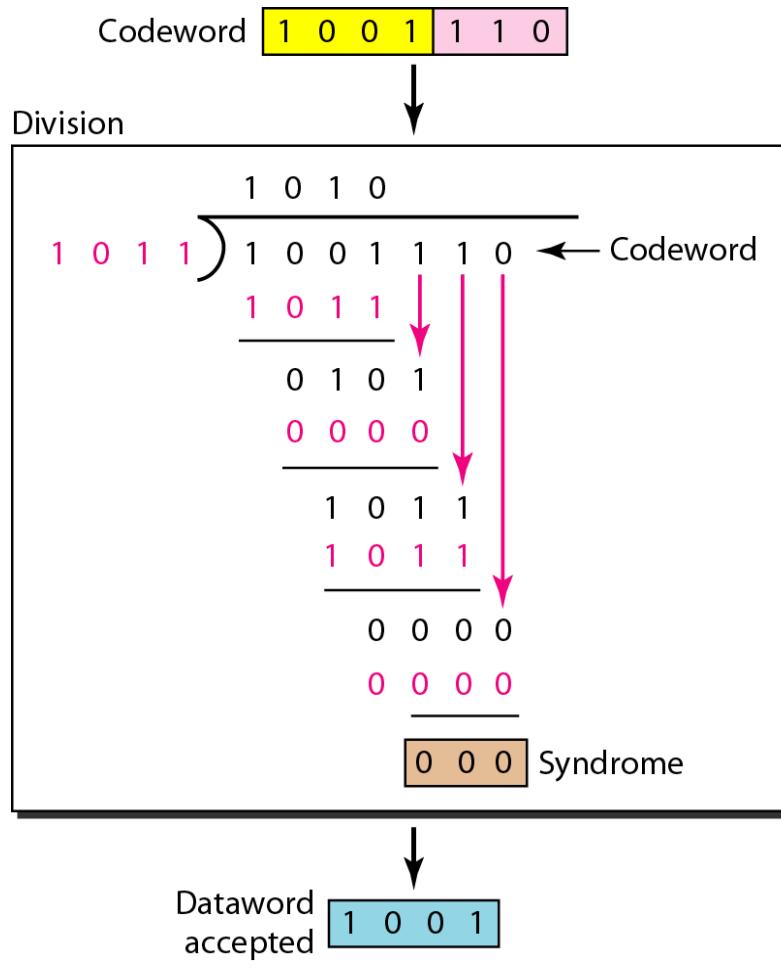


*Working of CRC at the sender side*

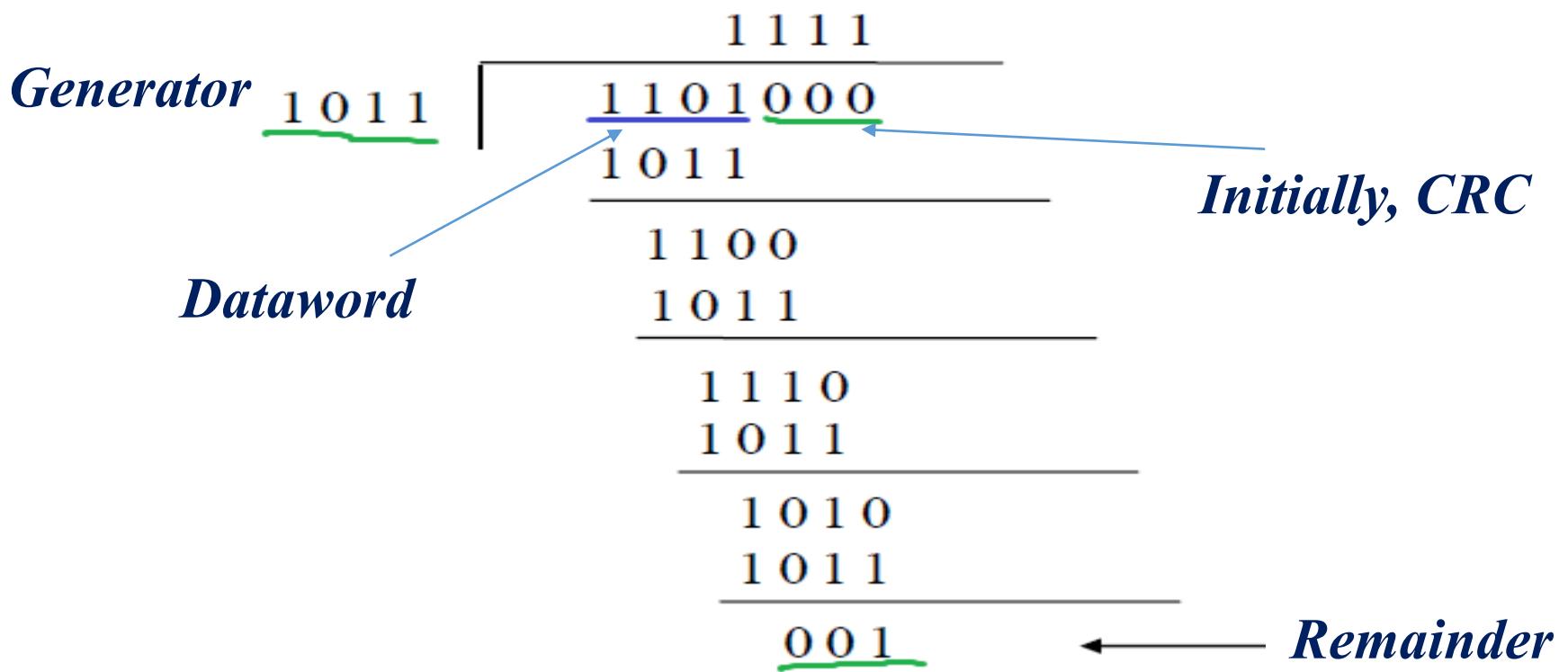
# How to do a binary division !!!



# Working of CRC at the receiver side



- **Ex:** Consider a case where data word is 1101 (i.e.,  $m=4$  bits) and predetermined number is 1011. Determine FCS.
- **Sol:**
  - Initially, we choose CRC as all zero bits. Since, predetermined number is of 4 bits, we choose 3 zero bits, i.e., 000 as CRC.
  - So, we append 3 zeros to dataword and then it will be divided by 1011.

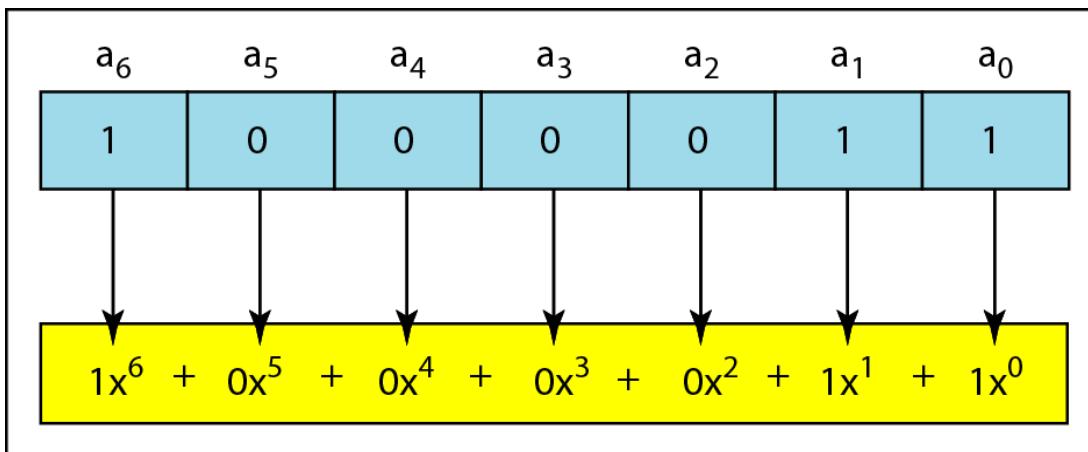


- Q) Find CRC/FCS for message **1010001101** for the given pattern, **110101**. Also verify whether the receiver will accept or reject, if the received data is **10100011000110**.
- **!!!!!! Just do it !!!!!**

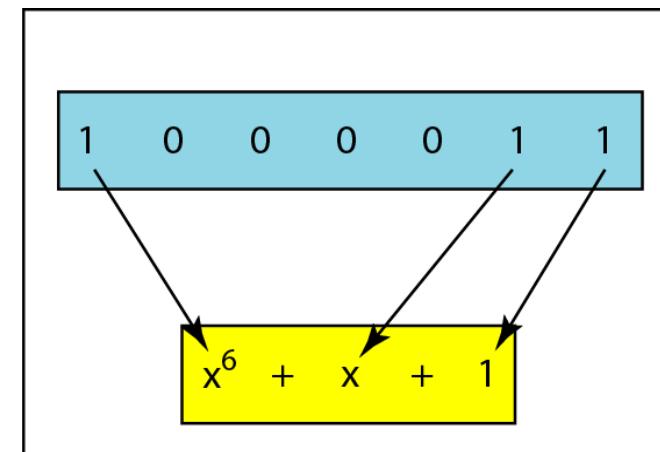
# CRC code using Polynomial

- It is a better way to understand cyclic code.
- Using polynomial, analysis of cyclic code is easier as compared to binary representation.
- We can use polynomial to represent a binary word.
- Each bit from right to left is mapped onto a power term.
- The rightmost bit represents the “0” power term. The bit next to it the “1” power term, so on.
- If the bit is of value zero, the power term is deleted from the expression.

## *A polynomial to represent a binary word*



a. Binary pattern and polynomial

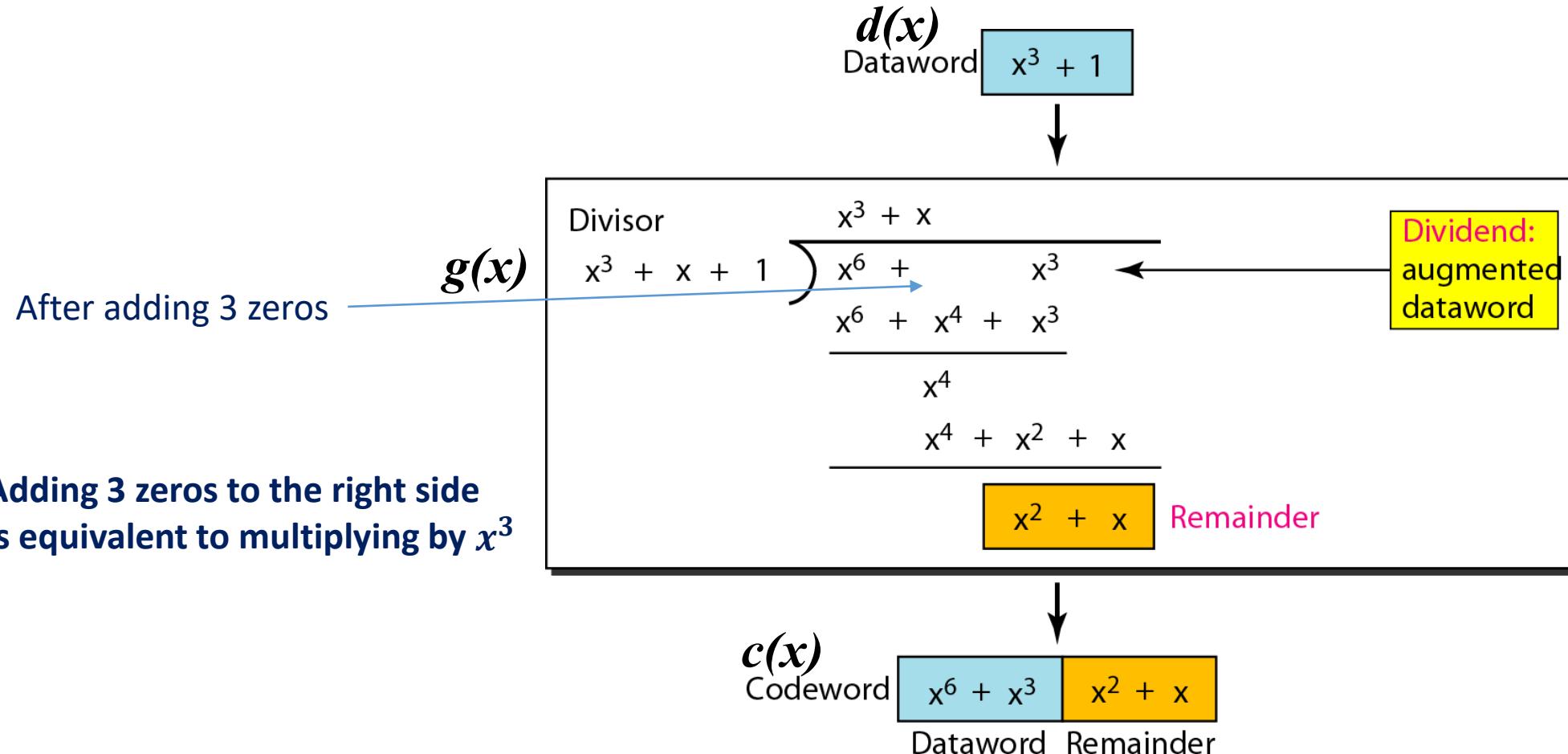


b. Short form

**Note: for n bits, polynomial can have a maximum degree of n-1.**

- We use the following notations in polynomial representation.
  - *Dataword*:  $d(x)$
  - *Codeword*:  $c(x)$
  - *Error*:  $e(x)$
- **Syndrome**: *It is the remainder produced by the checker (at Rx. side). It is also treated as set of patterns accepted. For e.g., generally in CRC, we say that if syndrome is all 0's then no error; otherwise there is some error.*

**Ex: Consider 1001 be the dataword and 1011 be the pattern (generator). Find CRC using polynomial approach.**



# CRC code: Key points

- In a cyclic code,
  - Divisor is called generator polynomial or simply generator, denoted by  $g(x)$ .
  - Remainder at the receiver side is called syndrome, denoted by  $s(x)$ .
  - Dataword and remainder together are called codeword, denoted by  $c(x)$ .
  - Error polynomial is represented as  $e(x)$ .
  - Received codeword is  $c'(x)=c(x)+e(x)$ .
- In cyclic code, those  $e(x)$  errors that are divisible by  $g(x)$  are not caught.  
 **$Received\ codeword\ (c(x) + e(x))/g(x) = c(x)/g(x) + e(x)/g(x)$**

- Properties of the CRC code are determined by the choice of the generator polynomial. Choose  $g(x)$  such that as many error patterns  $e(x)$  as possible may be detected.

# How to choose generator polynomial...

- Two main properties which must be satisfied by a generator polynomial.
  - 1) It should not be divisible by X.
  - 2) It should not be divisible by  $(X+1)$ .

## Performance:

- CRC can detect
  - All single-bit errors
  - All double-bit errors
  - All burst errors of less than the degree of the polynomial.
  - Most of the larger burst errors with a high probability.
  - For e.g., CRC-12 detects 99.97% of errors with a length 12 or more.

# Standard Generator Polynomials

CRC = cyclic redundancy check

CCITT = Consultative Committee for International  
Telephony and Telegraphy

## CRC-8:

$$= x^8 + x^2 + x + 1$$

ATM

## CRC-16:

$$= x^{16} + x^{15} + x^2 + 1$$

$$= (x + 1)(x^{15} + x + 1)$$

Bisync

## CCITT-16:

$$= x^{16} + x^{12} + x^5 + 1$$

HDLC, XMODEM

IEEE 802

## CCITT-32:

$$= x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

*Error Control → Error Correcting Codes*

# Two basic approach for error correction

- **Forward error correction:** In this method, receiver can use an error-correcting code, which automatically corrects certain errors.
- E.g., Hamming Code, BCH Code, etc.
  
- **Backward error correction:** In this method, when an error is discovered; the receiver can have the sender retransmit the entire data unit. This is also referred as *Retransmission* mechanism.
- E.g., Go-Back-N ARQ, Selective Repeat ARQ, etc.
- **HYU:** *Explain error correction Vs retransmission concept.*

**Q) How many errors can be corrected by using error correcting codes?**



## **Q) How many errors can be corrected by using error correcting codes?**

**Ans:** *In theory, it is possible to correct any number of errors atomically.*

*Error-correcting codes are more sophisticated than error detecting codes and require more redundant bits. The number of bits required to correct multiple-bit or burst error is so high that in most of the cases it is inefficient to do so.*

*For this reason, most error correction is limited to one, two or at the most three-bit errors.*



## In Hamming code, for 1-bit error

- **Requirements for error detection:** A code is an error detecting codes if and only if the minimum distance (Hamming distance) between any two code word is two.
- **Requirement for error correction:** For a code to be error-correcting, the minimum distance between any two code words must be more than two.

The # errors can be detected =  $d_{min}-1$   
and

The # errors can be corrected =  $(d_{min}-1)/2$

# Single-bit error correction: Hamming Code

**Hamming Code:** It is a binary linear block code denoted as  $(n,k,d)$ , where,  $n \leq 2^r - 1$ , and  $n = k+r$ .

$k = \# \text{ message bits}$ ,  $r = \# \text{ parity bits}$

*Q.1) How to decide number of parity bits for a given data/message?*

It is decided by,

$$k + r \leq 2^r - 1$$

1

**Some examples:**

**Ex.1:** If  $k=1$  then  $r=2$  satisfies result 1.

**Ex.2:** If  $k=2$  then  $r=3$  satisfies result 1.

**Ex.3:** If  $k=3$  then  $r=3$  satisfies result 1.

- **Ex.4:** If  $k = 4$  then  $r = 3$  satisfies result 1.
- **Ex.5:** If  $k = 5, 6, 7, 8$  then  $r = 4$  satisfies result 1.

**Observation:** *The number of parity bits required may be same for two different data words with different number of bits.*

## **Q.2) What are the positions of parity bits?**

**Ans:** Position of the parity bits is given by relation  $2^r$ , where  $r = 0, 1, 2, 3, \dots$

## **Q.3) What are the values of parity bits?**

**Ans:** For first parity bit, it is obtained by doing XOR operation of all those data bits whose position in binary has 1<sup>st</sup> bit 1 from the right side.

Simillarly, for  $i^{\text{th}}$  parity bit, it is obtained by doing XOR operation of all those data bits whose position in binary has  $i^{\text{th}}$  bit 1 from the right side.

- **Q.4) How an error is corrected?**

**Ans:** For received data, again calculate value of each parity bit, now there can be two cases:

**Case 1:** If there is error in any one of the parity bits.

The parity bit which will be in error will differ from the old parity bit.

**Case 2:** If there is error in any one of the data bits.

Any two or more parity bits will differ from the old parity bits. In this case, the position of data bit in error is calculated by doing the sum of positions of respective parity bits.

- **Example:** Let us assume that data to be transmitted be **1011** (here  $k=4$ ).
- Since  $k=4$ ,  $r=3$  satisfies equation 1. So, we have to add 3 parity bits.

$D_7$	$D_6$	$D_5$	$P_4$	$D_3$	$P_2$	$P_1$
1	0	1		1		
111	110	101	100	011	010	001

$$\therefore 2^0 = 1, 2^1 = 2, 2^2 = 4$$

- Parity bits will be at positions 1,2 and 4.

- **Example:** Let us assume that data to be transmitted be **1011** (here  $k=4$ ).
- **Values of parity bits:**

$D_7$	$D_6$	$D_5$	$P_4$	$D_3$	$P_2$	$P_1$
1	0	1		1		
111	110	101	100	011	010	001

$$P_1 = D_3 \oplus D_5 \oplus D_7 = 1 \oplus 1 \oplus 1 = 1$$

$$P_2 = D_3 \oplus D_6 \oplus D_7 = 1 \oplus 0 \oplus 1 = 0$$

$$P_4 = D_5 \oplus D_6 \oplus D_7 = 1 \oplus 0 \oplus 1 = 0$$

- **Example:** Let us assume that data to be transmitted be **1011** (here  $k=4$ ).
- **Values of parity bits:**

$D_7$	$D_6$	$D_5$	$P_4$	$D_3$	$P_2$	$P_1$
1	0	1	0	1	0	1
111	110	101	100	011	010	001

Data to be transmitted

$$P_1 = D_3 \oplus D_5 \oplus D_7 = 1 \oplus 1 \oplus 1 = 1$$

$$P_2 = D_3 \oplus D_6 \oplus D_7 = 1 \oplus 0 \oplus 1 = 0$$

$$P_4 = D_5 \oplus D_6 \oplus D_7 = 1 \oplus 0 \oplus 1 = 0$$

- **Example:** Let us assume that data to be transmitted be **1011** (here  $k=4$ ).
- **Case 1: At the Rx side, if received data is**

$D_7$	$D_6$	$D_5$	$P_4$	$D_3$	$P_2$	$P_1$
1	0	1	(1)	1	0	1
111	110	101	100	011	010	001

$$P_1 = D_3 \oplus D_5 \oplus D_7 = 1 \oplus 1 \oplus 1 = 1 \quad \dots\dots\dots\dots\dots \text{matched}$$

$$P_2 = D_3 \oplus D_6 \oplus D_7 = 1 \oplus 0 \oplus 1 = 0 \quad \dots\dots\dots\dots\dots \text{matched}$$

$$P_4 = D_5 \oplus D_6 \oplus D_7 = 1 \oplus 0 \oplus 1 = 0 \quad \dots\dots\dots\dots\dots \text{not matched}$$

So, error is in the 4<sup>th</sup> position, which is a parity bit, P4

- Case 2: At the Rx side, if received data is

$D_7$	$D_6$	$D_5$	$P_4$	$D_3$	$P_2$	$P_1$
1	0	1	0	0	0	1
111	110	101	100	011	010	001

$$P_1 = D_3 \oplus D_5 \oplus D_7 = 0 \oplus 1 \oplus 1 = 0 \quad \dots \dots \dots \text{not matched}$$

$$P_2 = D_3 \oplus D_6 \oplus D_7 = 0 \oplus 0 \oplus 1 = 1 \quad \dots \dots \dots \text{not matched}$$

$$P_4 = D_5 \oplus D_6 \oplus D_7 = 1 \oplus 0 \oplus 1 = 0 \quad \dots \dots \dots \text{matched}$$

So, position is  $1+2=3$ , where 3 is position of data bit  $D_3$

**Problem:** For 8 bits of data **11000010**, using hamming code, find

- a) The number of parity bits required.
- b) The values of parity bits.
- c) The data to be transmitted.
- d) Correct the error, if received codeword is **110010010010**.

!!!! Just do it !!!!

## Quick review:

1. Error detection is usually done in \_\_\_\_\_ layer of OSI.
2. \_\_\_\_\_ uses the one's complement arithmetic.
3. \_\_\_\_\_ is the error detection method which consists of a parity bit for each data unit as well as an entire data unit of parity bit.
4. The number of bits position in which code words differ is called the \_\_\_\_\_ distance.
5. To detect d errors, you need a distance \_\_\_\_\_ code.
6. To correct d errors, you need a distance \_\_\_\_\_ code.
7. \_\_\_\_\_ error means that only one bit of given data unit (such as a byte, character, or data unit) is changed from 1 to 0 or from 0 to 1.

## Quick review:

8. Which Error detection method can detect a burst error? \_\_\_\_\_.
9. \_\_\_\_\_ involves polynomials.
10. In cyclic redundancy check, CRC is \_\_\_\_\_.
11. In Cyclic Redundancy Check, the divisor is \_\_\_\_\_ the CRC.
12. When an error is discovered; the receiver can have the sender retransmit the entire data unit. This is known as \_\_\_\_\_ **correction**.
13. When receiver can use an error-correcting code, which automatically corrects certain errors. This is known as \_\_\_\_\_ **correction**.

# CN (IT-3001)

## Data Link Layer: MAC Protocol

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University



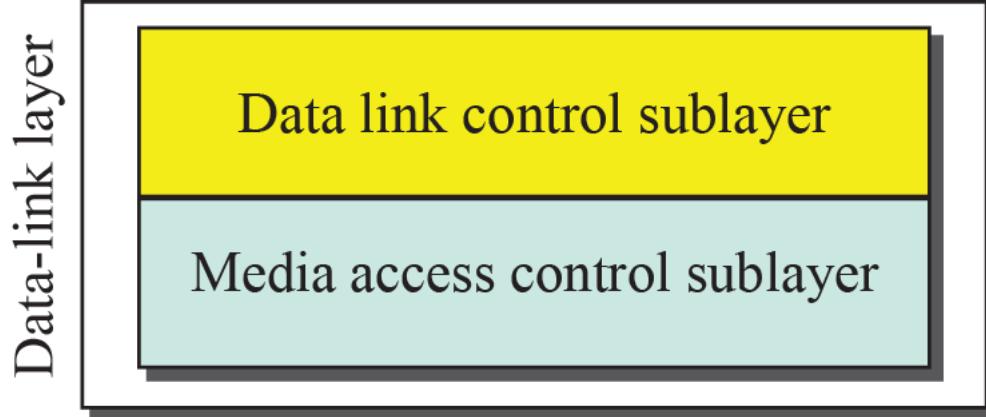
**Disclaimer:** The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose. Images portrayed (if any) are not to hurt the sentiments of any person.

# Objective

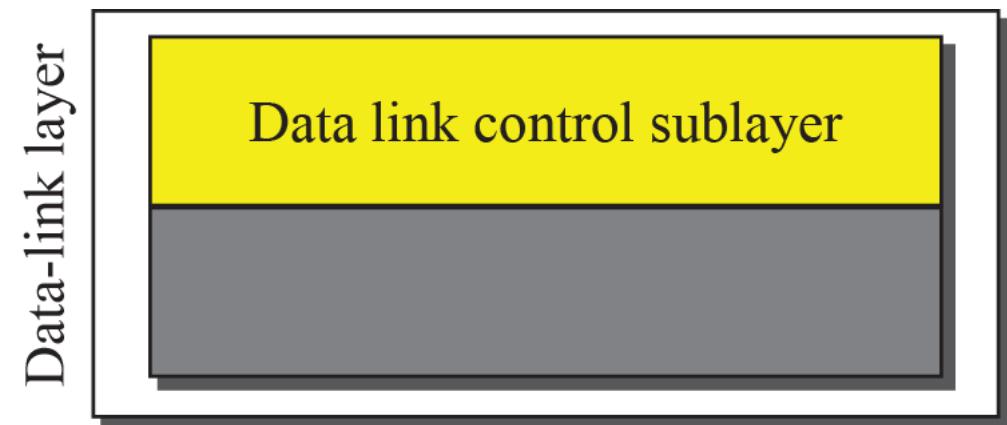
- Media Access/Multiple Access
  - Random Access
    - ALOHA
    - CSMA
    - CSMA/CD
    - CSMA/CA
  - Controlled Access
    - Reservation
    - Polling
    - Token passing
  - Channelization
    - FDMA
    - TDMA
    - CDMA

# Two Sublayers of the Data-Link-Layer

- The data link layer is divided into two sublayers as shown below.
  1. Data Link Control (DLC) sublayer
  2. Media Access Control (MAC) Sublayer



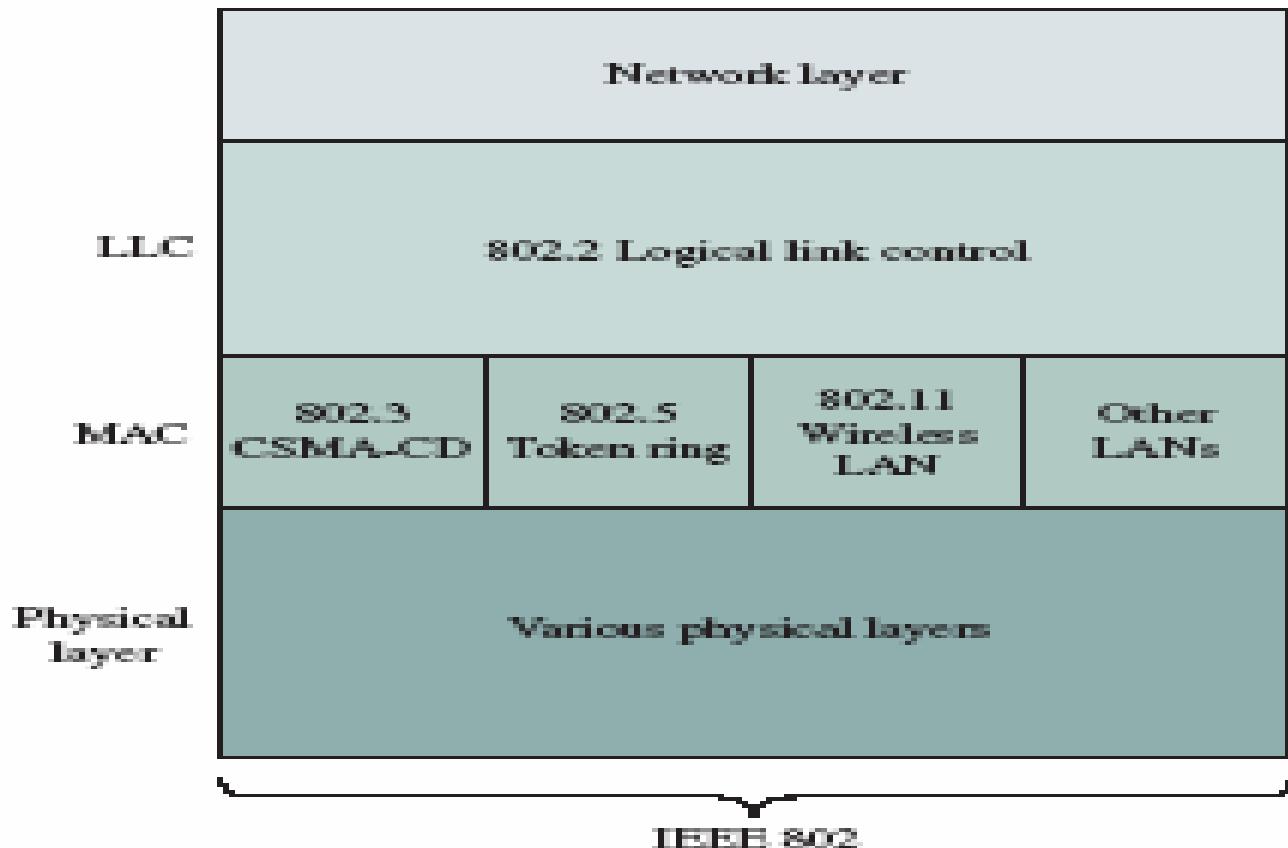
a. Data-link layer of a broadcast link



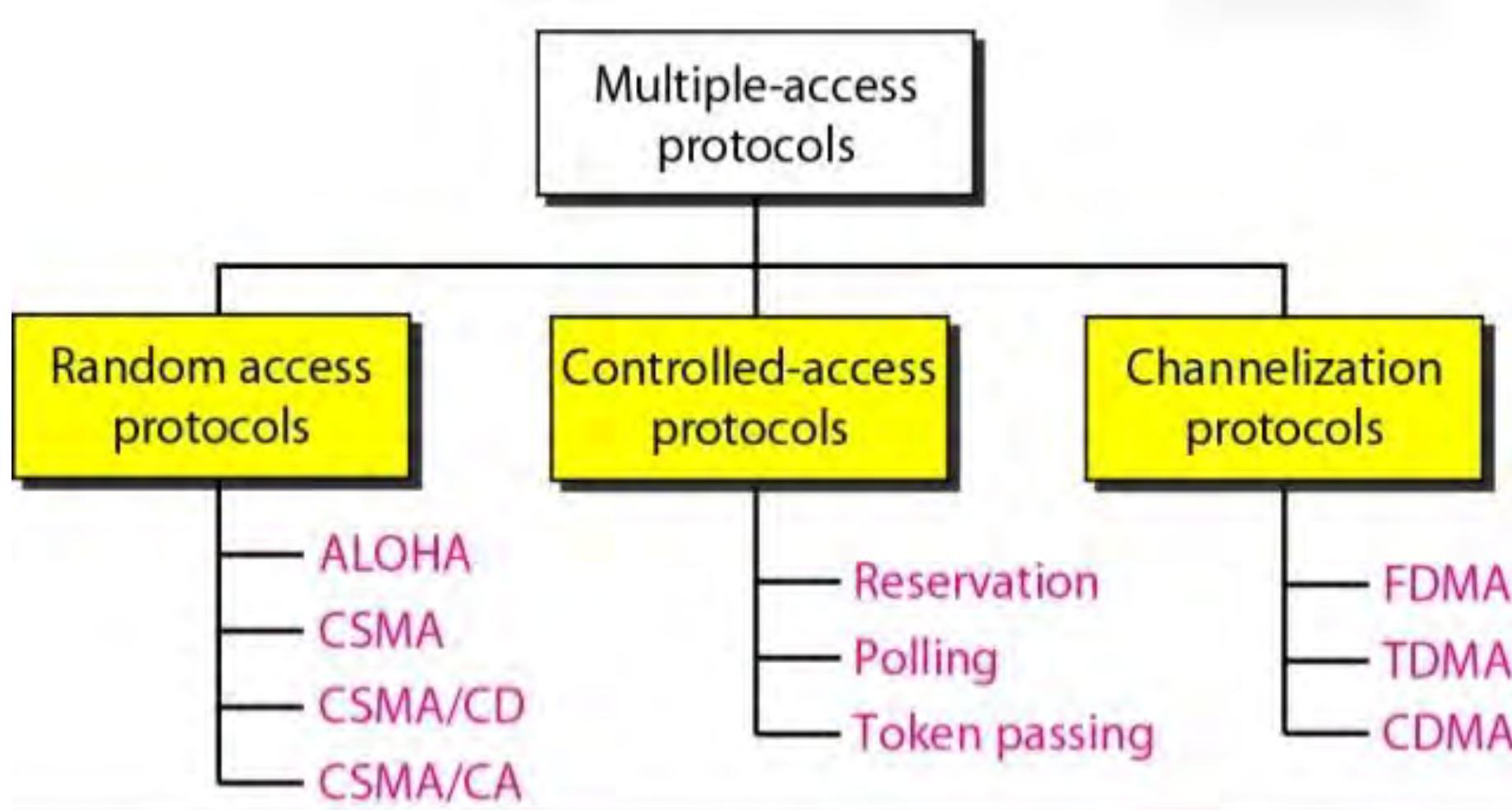
b. Data-link layer of a point-to-point link

- The upper sublayer that is responsible for flow and error control is called the *logical link control* (LLC) layer.
- The lower sublayer that is mostly responsible for multiple access resolution is called the *media access control* (MAC) layer.
- **Why do we need multiple-access protocol?**

**Ans:** In a broadcast or multipoint, nodes use a common link. To use this common link efficiently, we need a multiple-access protocol to coordinate access to the link.



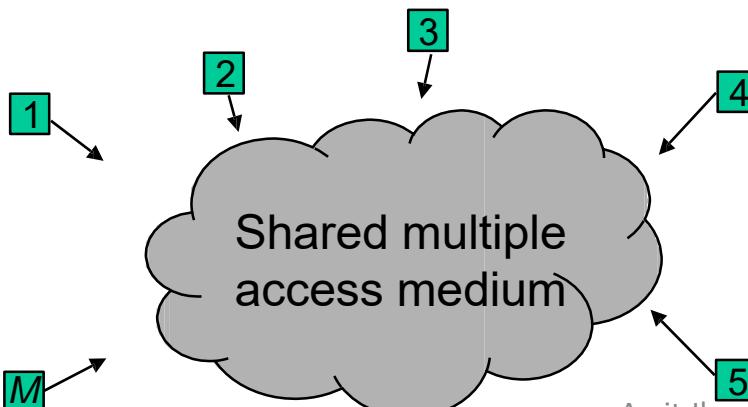
# Types of Multiple-Access Protocol/ MAC Protocol



# Random Access Protocol

- Why the name *random access*?
  - There is no scheduled time for a station to transmit.
  - Transmission is random among the stations.
  - No station is superior to another station and none is assigned the controlled over another.
- In a random access method, each station has the right to the medium without being controlled by any other station. However, if more than one station tries to send, there is an access conflict-collision-and the frames will be either destroyed or modified.

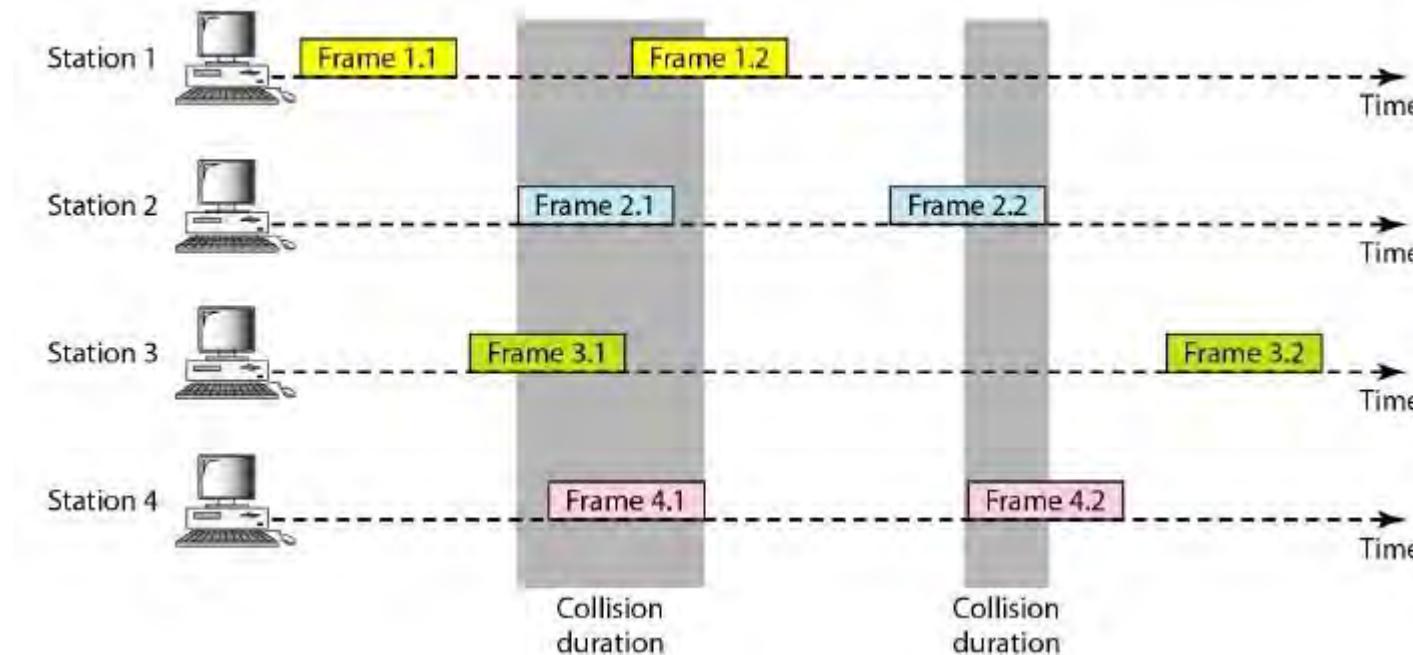
- To avoid access conflict or to resolve it when it happens, each station follows a procedure that answers the following questions:
  - When can the station access the medium?
  - What can the station do if the medium is busy?
  - How can the station determine the success or failure of the transmission?
  - What can the station do if there is an access conflict?



# ALOHA: Pure ALOHA

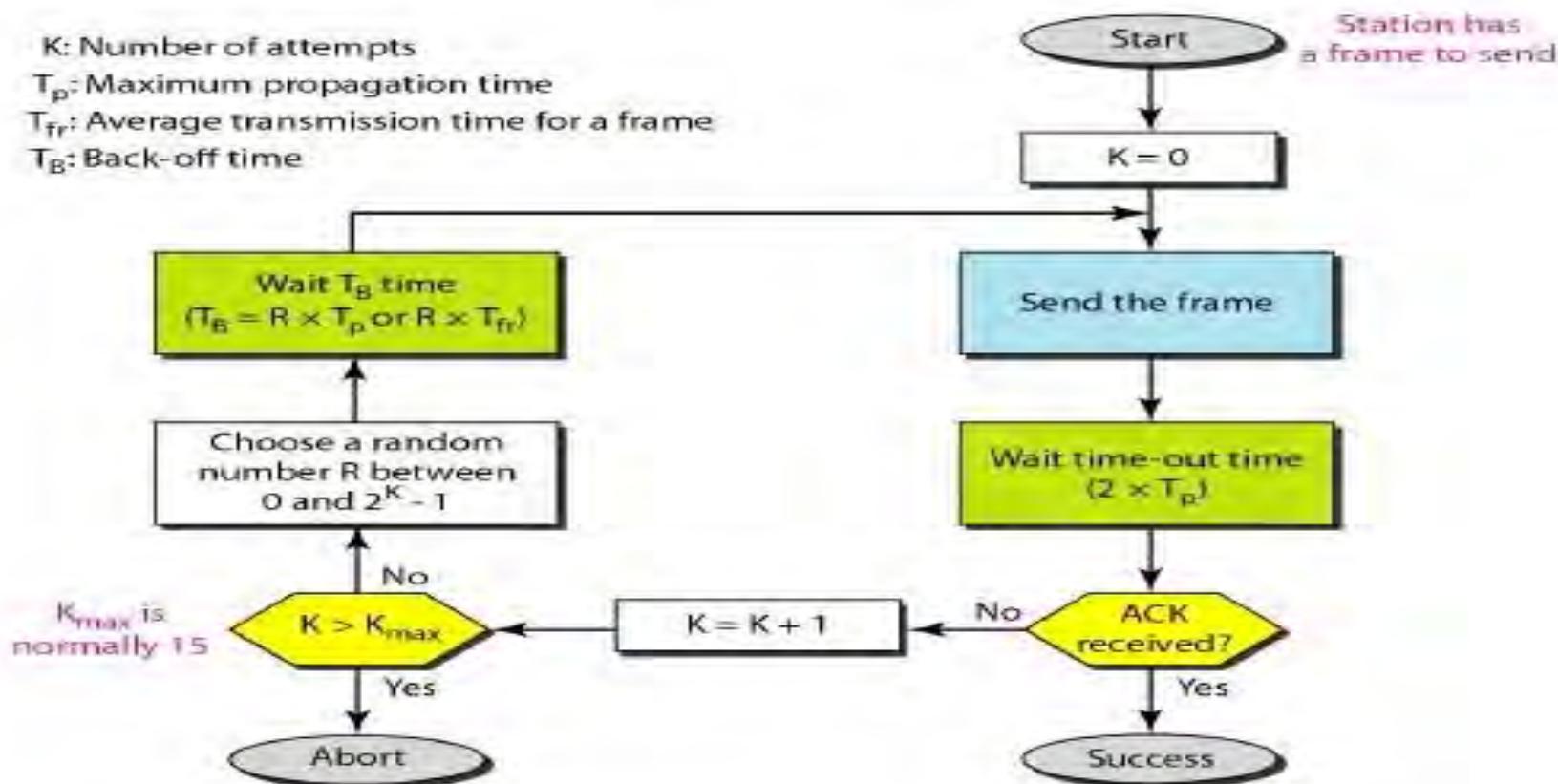
- Based upon the simplest solution: **just do it**
  - A station transmits whenever it has data to transmit.
  - If more than one frames are transmitted, they interfere with each other (collide) and are lost.
  - If ACK not received within timeout, then a station picks **random back-off time** (to avoid repeated collision).
  - Station retransmits frame after back-off time denoted as  $T_B$
- **Note:** A collision involves two or more stations. If all these stations try to resend their frames after the time-out, the frames will collide again.
- After a maximum number of retransmission attempts  $K_{max}$  a station must give up and try later.

*Four stations transmitting 2 frames each.  
Out of all the frames, only two frames survive: frame 1.1 and frame 3.2*



**Fig.** Example of frame collisions in pure ALOHA

# Procedure for pure ALOHA protocol



**Note:** R is a random number chosen from the range 0 to  $2^k - 1$ , and value of the random number increases after each collision.

**Example 1:** The stations on a wireless ALOHA network are a maximum of 600 km apart. If we assume that signals propagate at  $3 \times 10^8 \text{ m/s}$ , we find  $T_P = (600 \times 10^3) / (3 \times 10^8) = 2 \text{ ms}$ . Now we can find the value of  $T_B$  for different values of  $K$ .

- a) For  $K=1$ , the range of  $R$  is  $\{0, 1\}$ . The station needs to generate a random number with value 0 or 1. So,  $T_B$  is either 0 or 2ms, based on outcome of the random variable.
- b) For  $K=2$ , the range of  $R$  is  $\{0, 1, 2, 3\}$ . So,  $T_B$  can be 0, 2, 4 or 6ms, based on outcome of the random variable.
- c) For  $K=3$ , the range of  $R$  is  $\{0, 1, 2, 3, \dots, 7\}$ . So,  $T_B$  is can be 0, 2, 4, 6, 8, 10, 12, or 14ms, based on outcome of the random variable.
- d) So on.....
- e) We need to mention that if  $k > 10$ , it is normally set to 10.

**Vulnerable time:** It is the time duration , in which there is a possibility of collision. **Vulnerable time in pure ALOHA =  $2 \times T_{fr}$**

$T_{fr}$  = frame transmission time

B already sent a frame

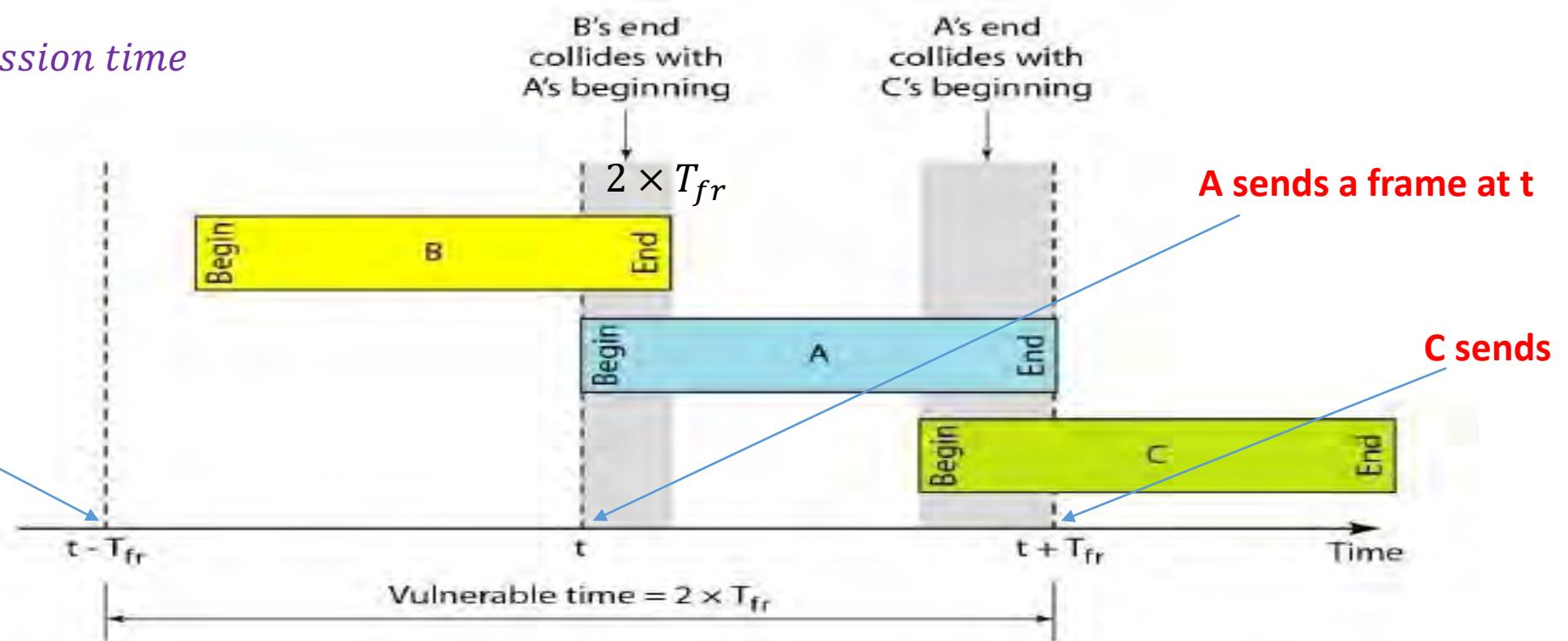


Fig: Vulnerable time for pure ALOHA

**Example 2:** A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the requirement to make this frame collision-free?

**Sol:**

*Average frame transmission time  $T_{fr}$  is 200 bits/200 kbps or 1 ms. The vulnerable time is  $2 \times 1 \text{ ms} = 2 \text{ ms}$ .*

*This means no station should send later than 1 ms before this station starts transmission and no station should start sending during the one 1-ms period that this station is sending.*

# Pure ALOHA Model

- Definitions and assumptions
  - $T_{fr}$  frame transmission time (assume constant)
  - $S$ : throughput (average # successful frame transmissions per  $T_{fr}$  seconds)
  - $G$ : load (average # transmission attempts per  $T_{fr}$  sec.)
  - $P_{success}$  : probability a frame transmission is successful

**Note:** Any transmission that begins during vulnerable period leads to collision. Success if and only if no arrivals during  $2 T_{fr}$  seconds.

Throughput is given by,

$$S = GP_{success}$$

## Abramson's assumption for calculation of $P_{Success}$

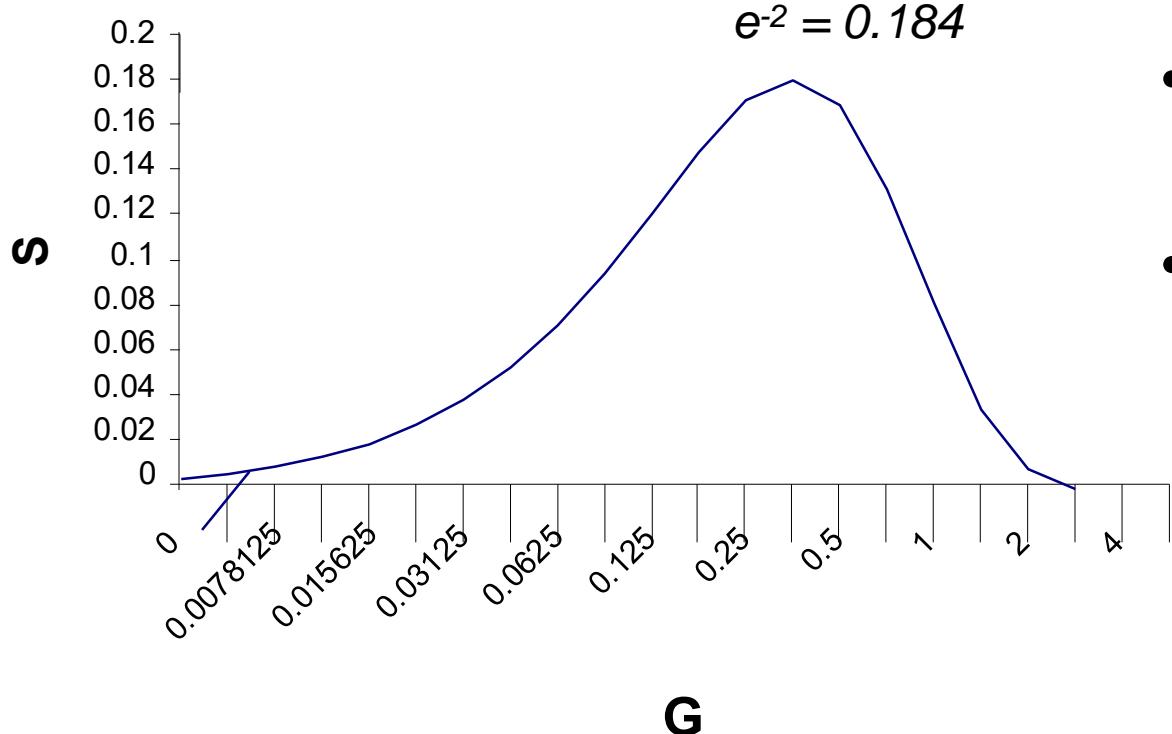
- *What is probability of no arrivals in vulnerable period?*
- **Abramson's assumption:** Effect of back-off algorithm is that frame arrivals are equally likely to occur at any time interval.
- $G$  is avg. # arrivals per  $T_{fr}$  seconds
- Divide  $T_{fr}$  into  $n$  intervals of duration  $\Delta = T_{fr}/n$
- $p$  = probability of arrival in  $\Delta$  interval, then

$$G = n p \quad \text{since there are } n \text{ intervals in } T_{fr} \text{ seconds}$$

$$\begin{aligned} P_{success} &= P[0 \text{ arrivals in } 2T_{fr} \text{ seconds}] = \\ &= P[0 \text{ arrivals in } 2n \text{ intervals}] \quad \dots \dots \dots \text{Abramson's assumption:} \\ &= (1 - p)^{2n} = \left(1 - \frac{G}{n}\right)^{2n} \rightarrow e^{-2G} \quad \text{as } n \rightarrow \infty \end{aligned}$$

# Throughput of ALOHA

$$S = GP_{success} = Ge^{-2G}$$



- Collisions are means for coordinating access
- Max throughput is  $\rho_{max} = 1/2e (18.4\%)$
- Bimodal behavior:
  - Small  $G$ ,  $S \approx G$
  - Large  $G$ ,  $S \downarrow 0$

Use basic maths

**Example 3:** A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces

- a. 1000 frames per second
- b. 500 frames per second
- c. 250 frames per second

**Sol:** Here,  $T_{fr}$  is 200 bits/200 kbps or 1 ms.

**a.** If the system creates 1000 frames per second, this is 1 frame per millisecond. The load is 1. In this case  $S = G \times e^{-2G}$  or  $S = 0.135$  (13.5 percent). This means that the throughput is  $1000 \times 0.135 = 135$  frames. Only 135 frames out of 1000 will probably survive.

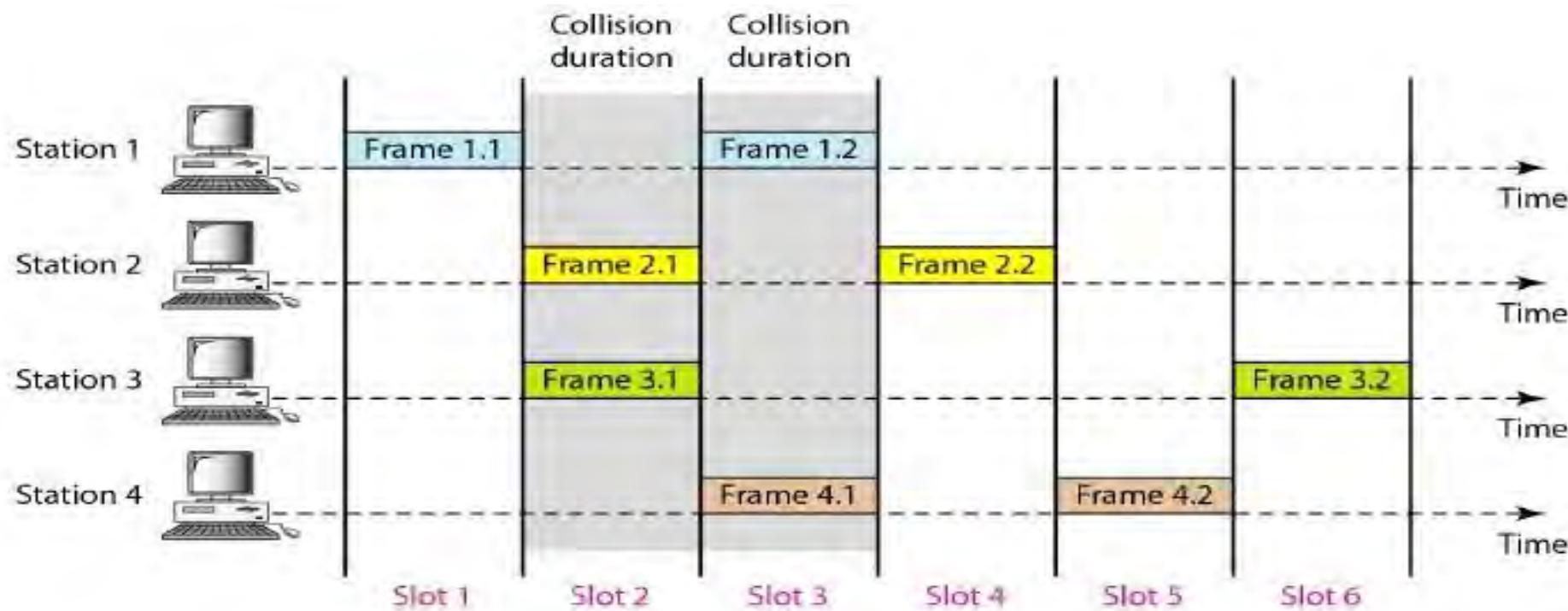
**b.** If the system creates 500 frames per second, this is  $(1/2)$  frame per millisecond. The load is  $(1/2)$ . In this case  $S = G \times e^{-2G}$  or  $S = 0.184$  (18.4 percent). This means that the throughput is  $500 \times 0.184 = 92$  and that only 92 frames out of 500 will probably survive.

Note that this is the maximum throughput case, percentagewise.

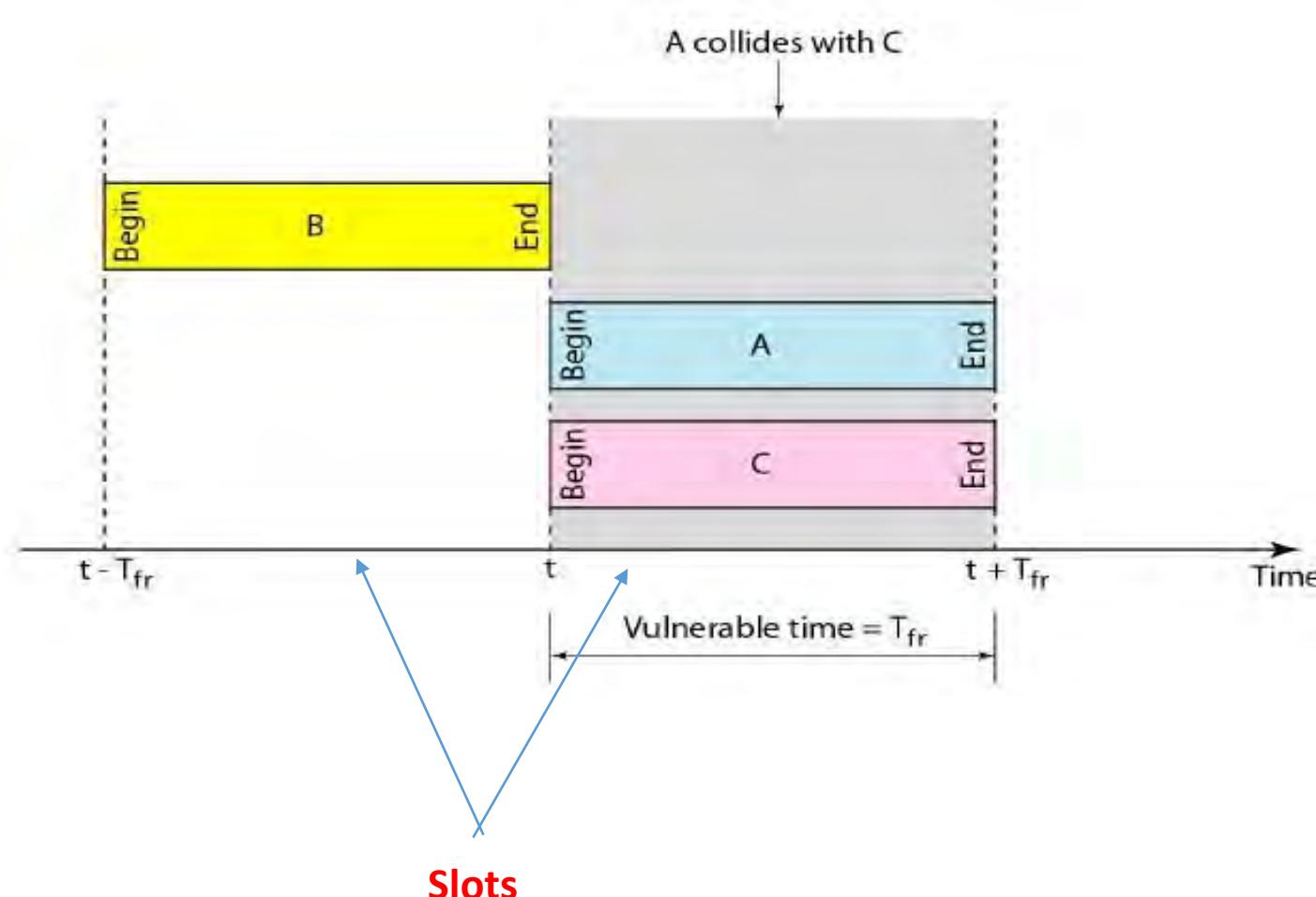
**c.** If the system creates 250 frames per second, this is  $(1/4)$  frame per millisecond. The load is  $(1/4)$ . In this case  $S = G \times e^{-2G}$  or  $S = 0.152$  (15.2 percent). This means that the throughput is  $250 \times 0.152 = 38$ . Only 38 frames out of 250 will probably survive.

# Slotted ALOHA

Time is slotted in  $T_{fr}$  seconds slots  
Stations synchronized to frame times  
Stations transmit frames in first slot after frame arrival  
Backoff intervals are in multiples of slots



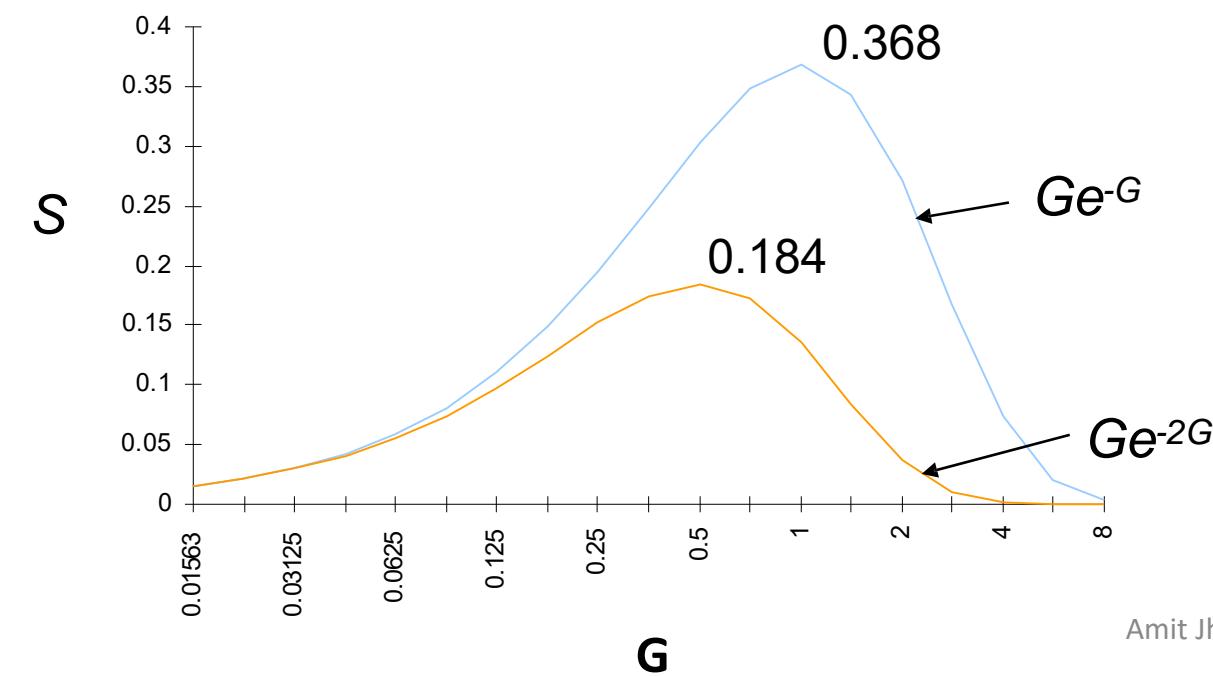
# Vulnerable time for slotted aloha



# Throughput of Slotted ALOHA

$$\begin{aligned} P_{success} &= P[0 \text{ arrivals in } T_{fr} \text{ seconds}] \\ &= P[0 \text{ arrivals in } n \text{ intervals}] \quad \dots \text{Abramson's assumption} \\ &= (1 - P)^n = \left(1 - \frac{G}{n}\right)^n \\ &\rightarrow e^{-G} \quad \dots \dots \dots \text{as } n \rightarrow \infty \end{aligned}$$

$$\therefore S = GP_{Success} = Ge^{-G}$$



## Limitations of ALOHA

The throughput for pure ALOHA is  $S = G \times e - 2G$ .

The maximum throughput  $S_{max} = 0.184$  when  $G = (1/2)$ .

The throughput for slotted ALOHA is  $S = G \times e - G$ .

The maximum throughput  $S_{max} = 0.368$  when  $G = 1$ .

**Homework:** 1) Repeat Example 3 for slotted ALOHA, and observe the conclusion.  
2) Derive the formulae for the maximum throughput for Pure and Slotted ALOHA.

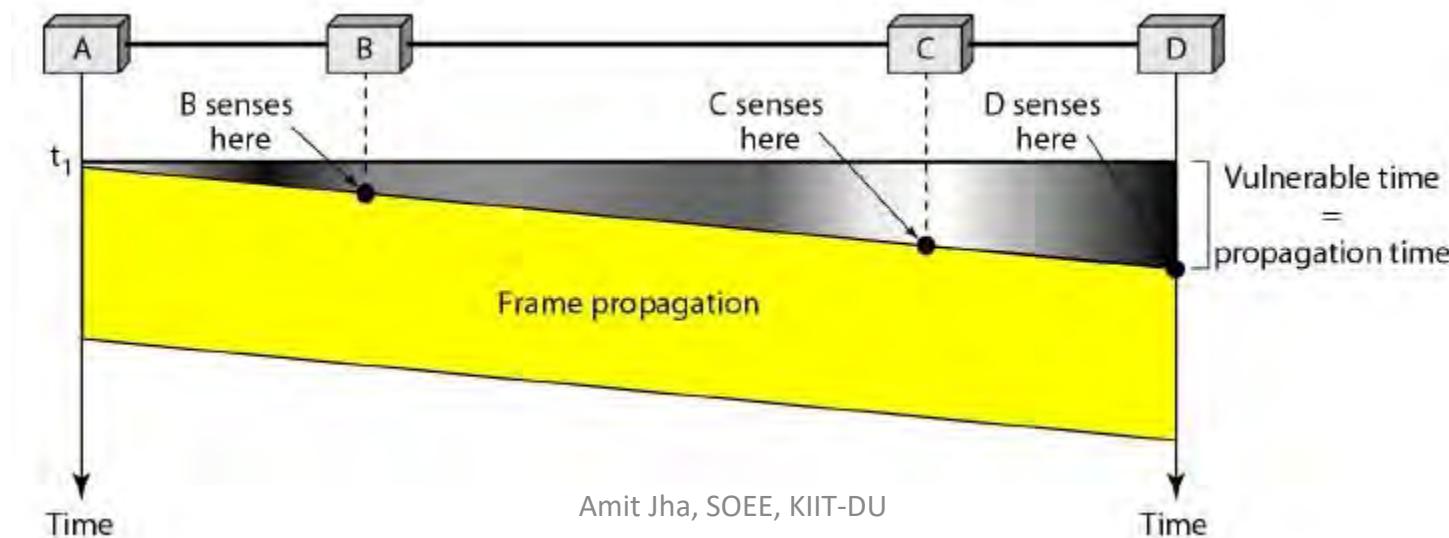
# Carrier Sense Multiple Access (CSMA)

- **Principle:** CSMA is based on the principle "sense **before** transmit" or "listen **before** talk."
- CSMA **can reduce** the possibility of collision, but it **cannot eliminate** it.
- **Why collision exists even after sensing?** The possibility of collision still exists because of **propagation delay**; when a station sends a frame, it still takes time (although very short) for the first bit to reach **every station** and for every station to sense it.
- In other words, a station may sense the medium and find it idle, only because the first bit sent by another station has not yet been received.

**Note:**  $T_{fr}$  *is generally greater than*  $T_p$ .

# *Vulnerable Time for CSMA*

- The vulnerable time for CSMA is the propagation time  $T_P$ . This is the time needed for a signal to propagate from one end of the medium to the other.
- In other words, it is the time taken by a bit to reach to the last station from first station.



# Persistence Methods

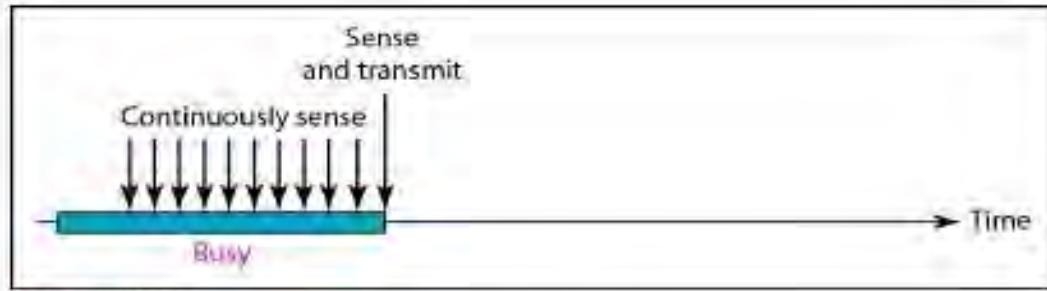
- What should a station do if the channel is busy?
- Three methods have been developed for this:
  1. The non-persistent method
  2. The 1-persistent method
  3. The p-persistent method
- **Non-persistent method (least greedy):** a station that has a frame to send **senses** the line. If the line is idle, it sends immediately. If the line is not idle, it **waits a random amount** of time and then **senses** the line again.
- The non-persistent approach **reduces the chance of collision** because it is unlikely that **two or more stations will wait the same amount of time** and retry to send simultaneously.
- However, this method **reduces the efficiency** of the network because the medium remains idle when there may be stations with frames to send.
- **1-persistent method:** Simplest method. Any station sends frame **with probability 1** as soon as it finds channel idle. Thus, it has highest probability of collisions.

Not continuously sensing

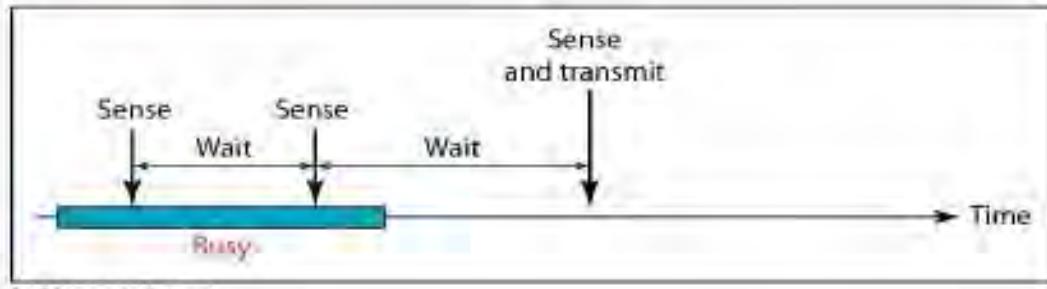
**Note:** Persistent means *continuous*

- **P-persistent method:** It combines the advantages of the other two strategies. It reduces the chance of **collision** and **improves efficiency**.
  - In this, a channel continuously senses the channel, this improves efficiency.
  - In this, after the station finds the **line idle** it follows these steps:
    1. With probability  $p$ , the station sends its frame.
    2. With probability  $q = 1 - p$ , the station waits for the beginning of the next time slot and **checks the line again**.
      - a. If the line is idle, it goes to step 1.
      - b. If the line is busy, it acts as though a collision has occurred and uses the back-off procedure.

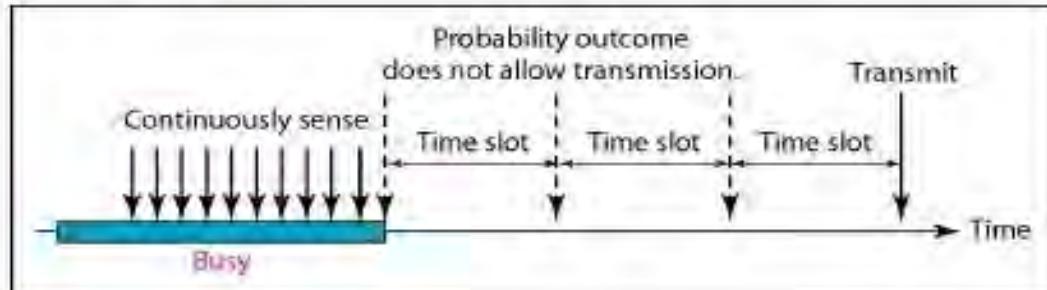
# Persistence methods



a. 1-persistent

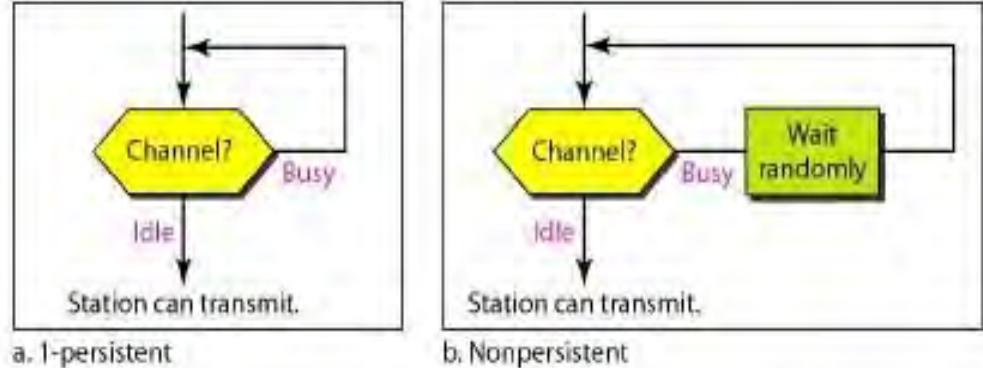


b. Nonpersistent

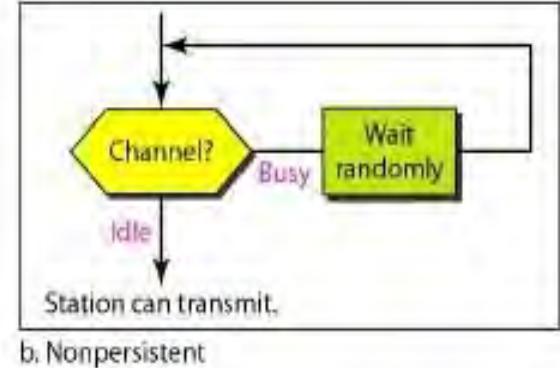


c. p-persistent

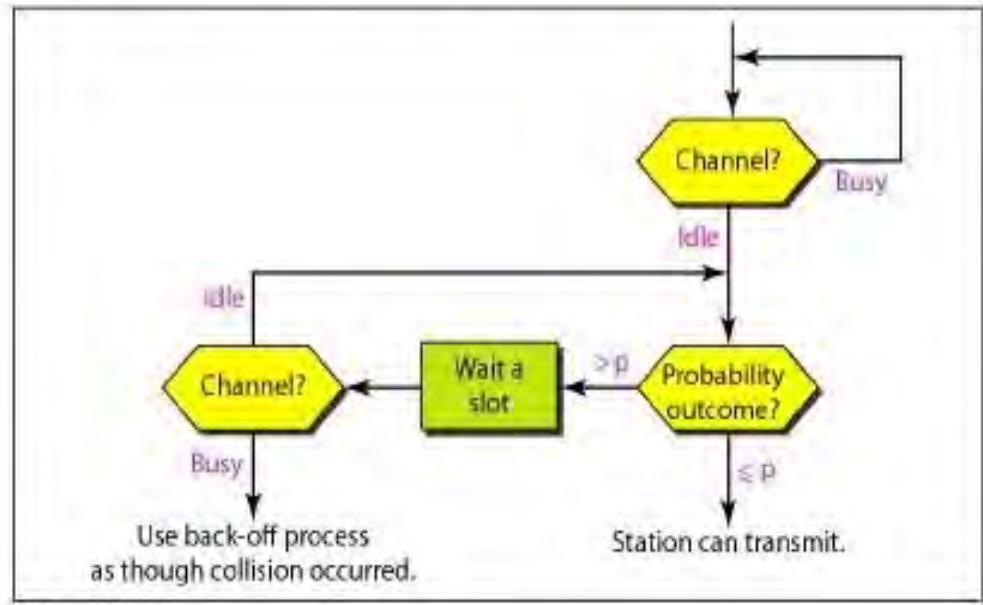
Fig: Behaviour of three persistence methods



a. 1-persistent



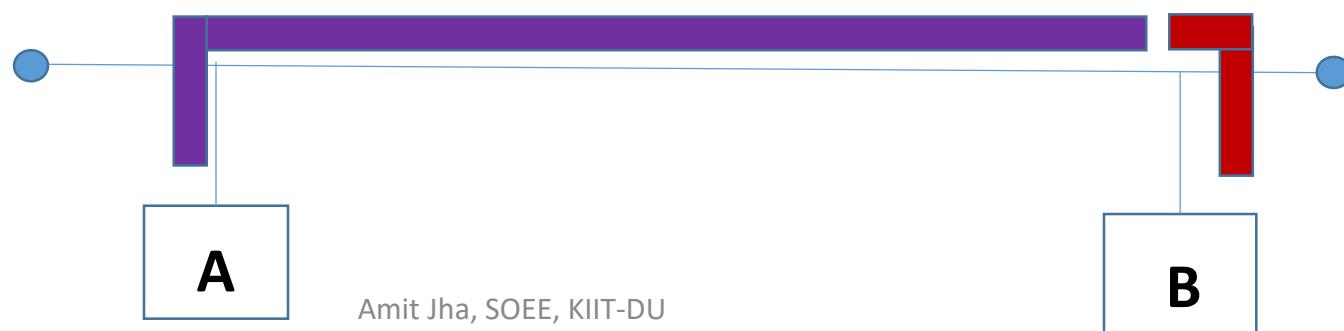
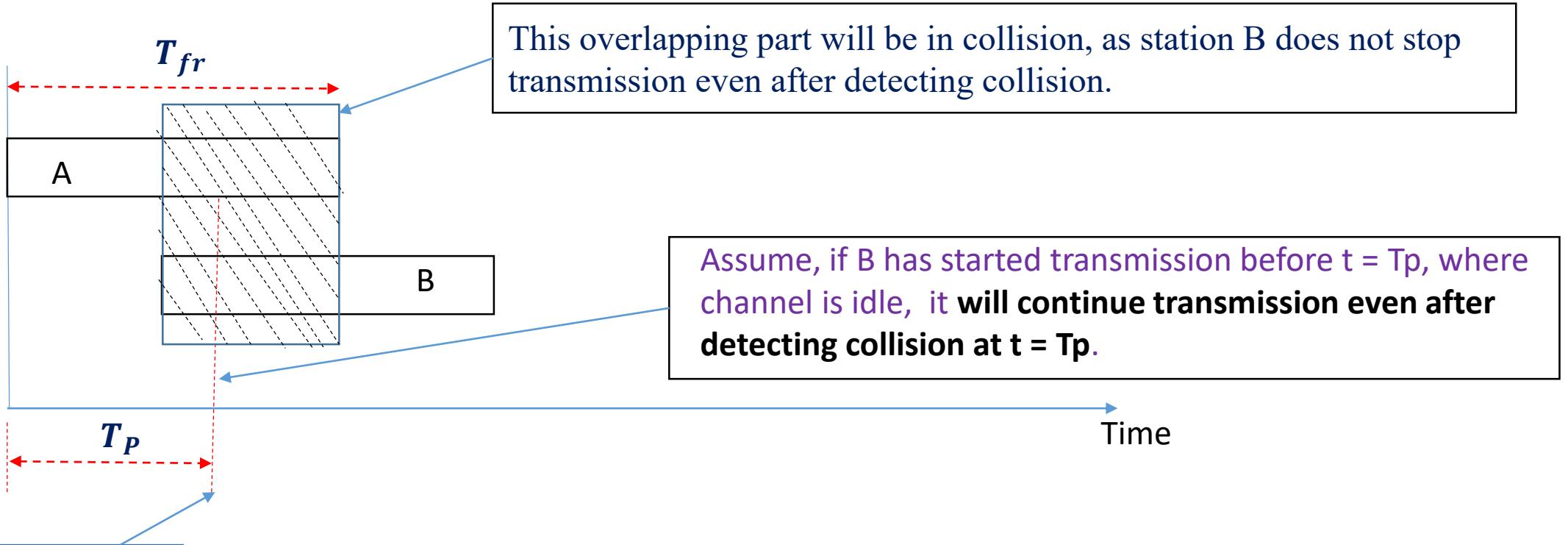
b. Nonpersistent



c. p-persistent

Fig: Flow diagram of three persistence methods

# Limitations of CSMA



# Limitations of CSMA

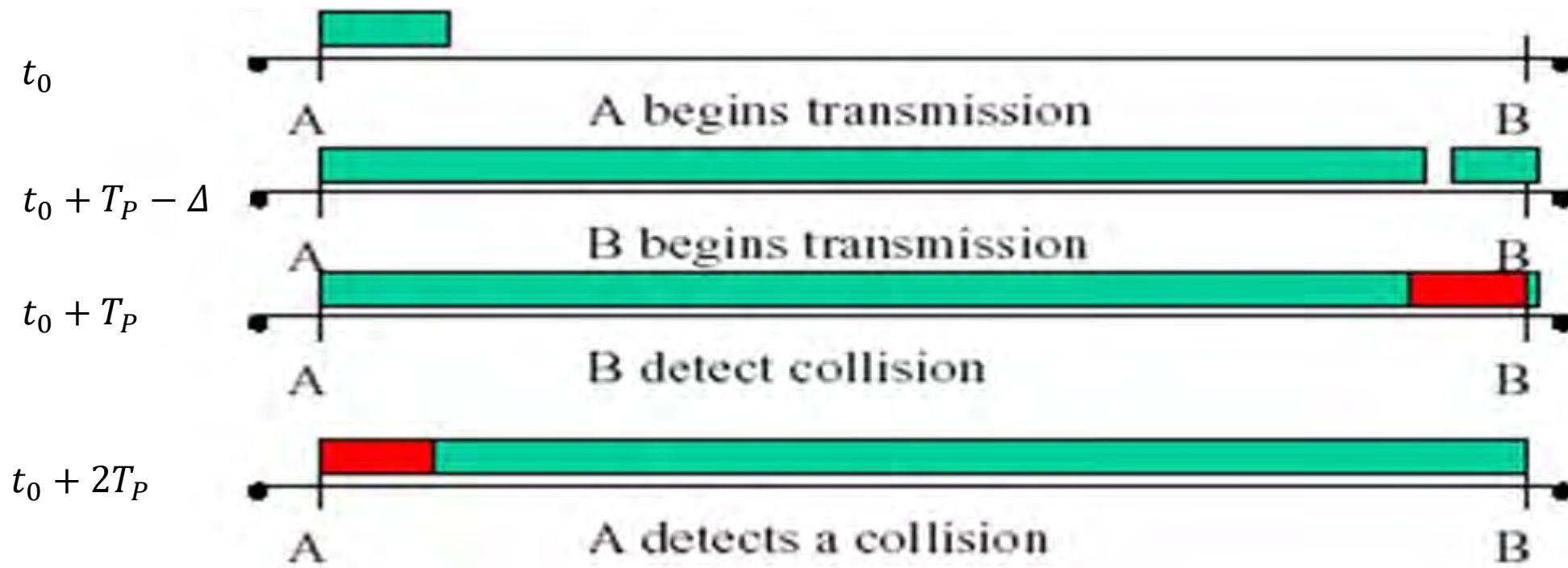
- A station can not sense a channel busy even when another station has already started sending .
- This is because of the propagation delay. Although, the propagation delay is very small, but still it exists there.
- Whenever a station detects that the collision has occurred, it does not stop transmission.

# Carrier Sense Multiple Access with Collision Detection (CSMA/CD)

- In this method, a station **listens while transmitting (LWT)** to see if the transmission was successful.
- Possible cases when channel is idle:
  1. Packet is transmitted in case of non-persistent or 1-persistent
  2. For p-persistent, the packet is sent with probability  $p$  or delayed by end-to-end propagation delay ( $T_P$ ) with probability  $(1-p)$ .
- If channel is busy, possible cases:
  1. For non-persistent, the packet is **backed off** and the algorithm is repeated.
  2. For 1-persistent, the station **differs** the transmission until the channel is sensed idle, and then immediately transmits when channel becomes idle.
  3. For p-persistent, the station **differs** until the channel is idle, then follow the channel idle procedure (as used in p-persistent CSMA case).

*Question:* How long does it take to detect a collision?

*Ans:* In the worst case, twice the maximum propagation delay of the medium, i.e.,  $2T_p$



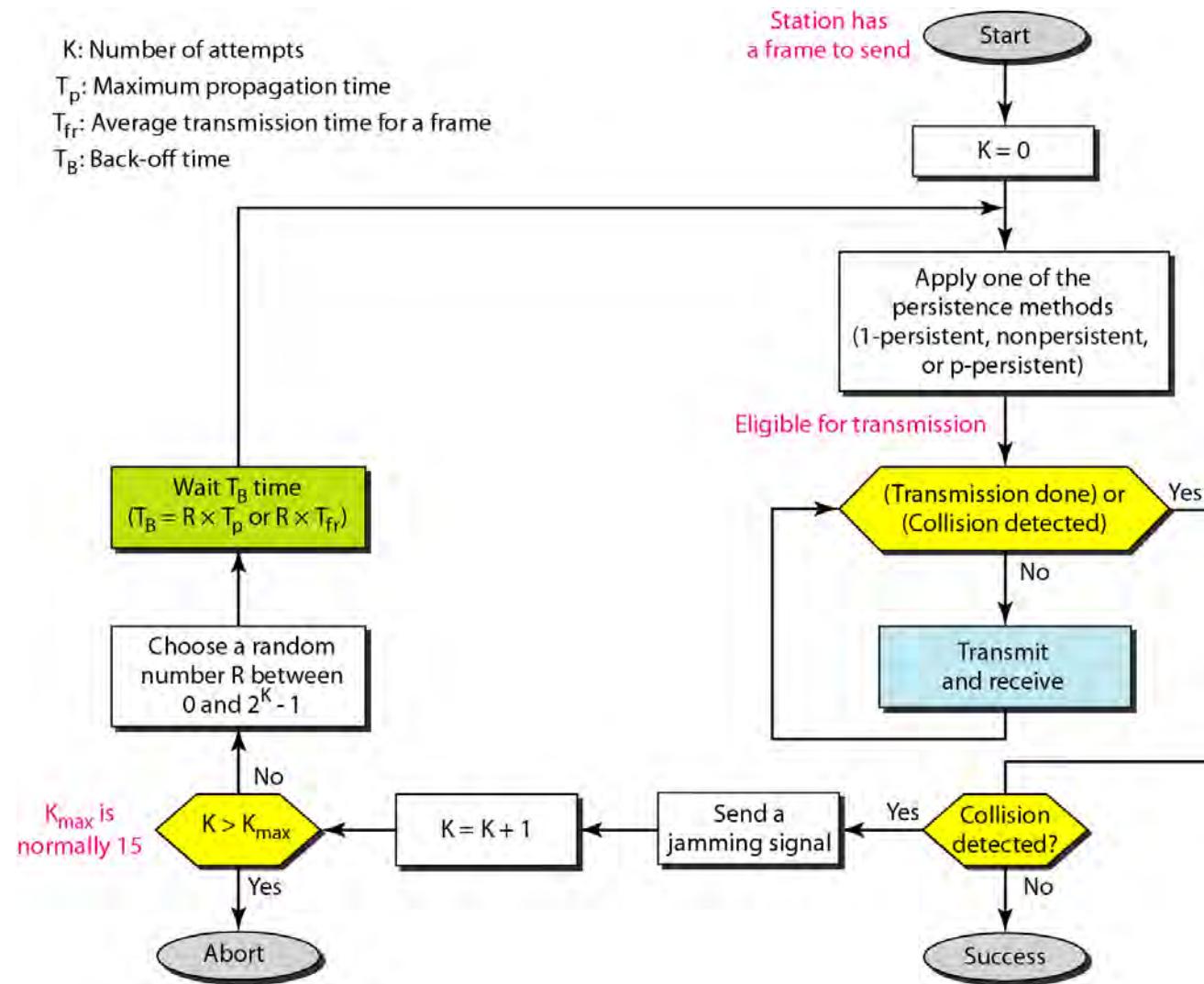
# Restrictions in CSMA/CD

- Frame/Packet **transmission time** should be **at least** as long as the time needed to detect a collision + time needed to send a jamming signal (i.e.,  $2 * \text{maximum propagation delay} + \text{jam sequence transmission time}$ ).
- Otherwise, CSMA/CD does not have an advantage over CSMA

$$\therefore T_{fr} \geq 2T_P + \text{jam sequence transmission time}$$

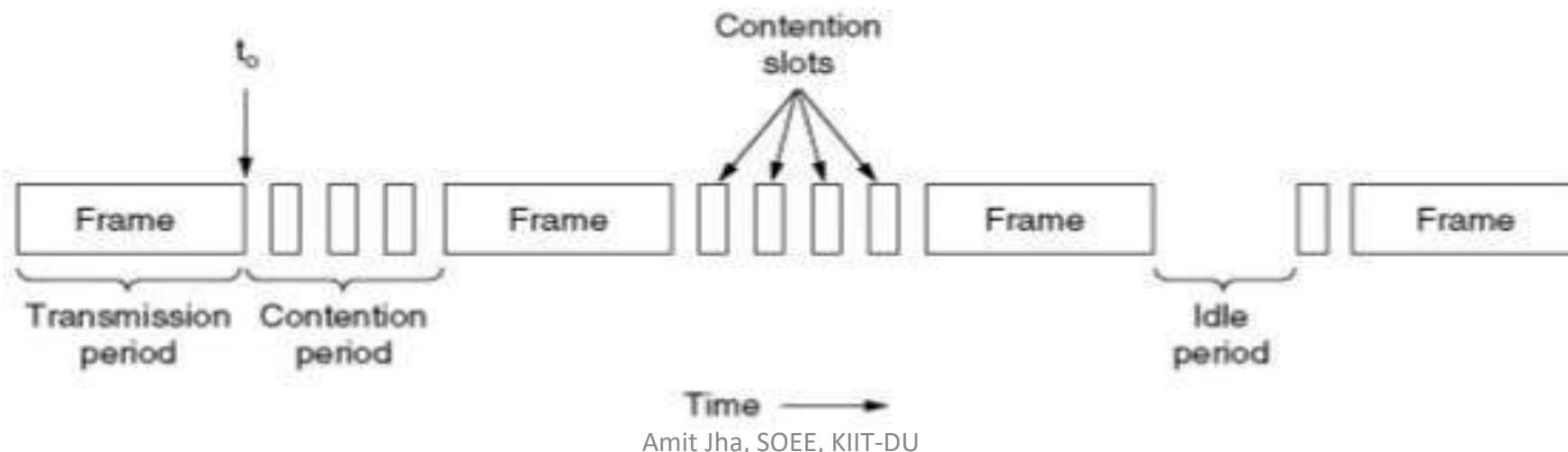
**Note:** The station which detects collision, sends a jamming signal to all other stations to inform that a collision has detected. So, please stop transmission.

# Flow diagram for the CSMA/CD



# The three periods in CSMA/CD

- The following three periods exists in CSMA/CD:
  1. **Contention period:** During this period, each station continuously senses (content) the channel to send a packet.
  2. **Transmission period:** During this, a station has found channel idle and sent a packet over it.
  3. **Idle:** No packet during this period; possible in case of light traffic channel.

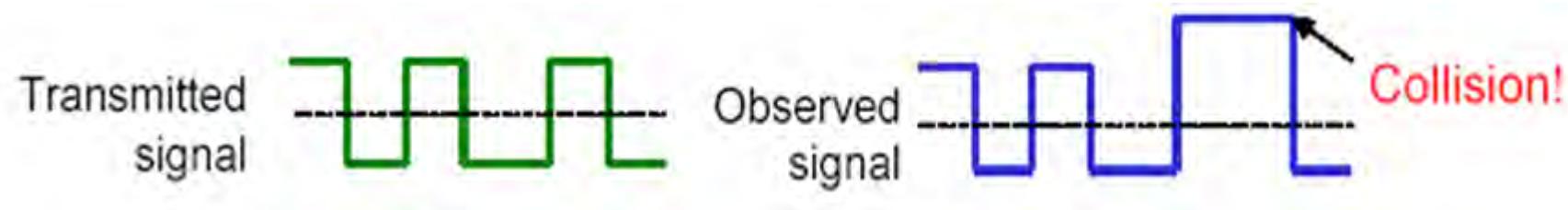


**Q.) Collision detection is not possible/difficult in a wireless environment, why?**



## Q.) Collision detection is not possible/difficult in a wireless environment, why?

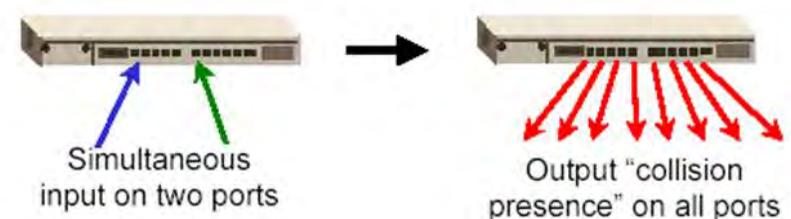
**Ans:** When there is a collision, the station receives two signals: its own signal and a signal transmitted by the second station. To distinguish between, these two signals, they must differ by a significant amount of energy. In wired network, the received signal has almost same energy as sent signal. This is because, either the length of cable is short or repeaters are used. It means, in collision, the detected energy is almost double as shown below.



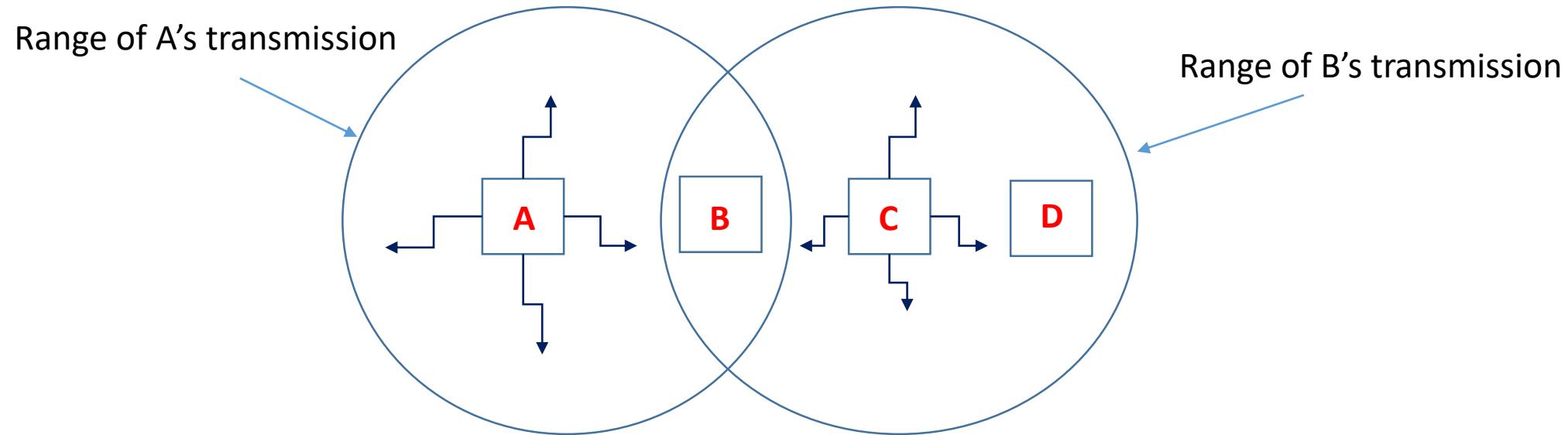
However, in a wireless network, much of sent energy is lost in transmission. The received signal has very little energy. Therefore, a collision may add only 5 to 10 % additional energy. This is not useful for collision detection.

Not only this but also there exists the **hidden station** and **exposed station** problem in wireless environment.

**Note:** In case of hub, if input occurs simultaneously on two ports, it indicates a collision. In this case Hub sends a collision presence signal on all ports.

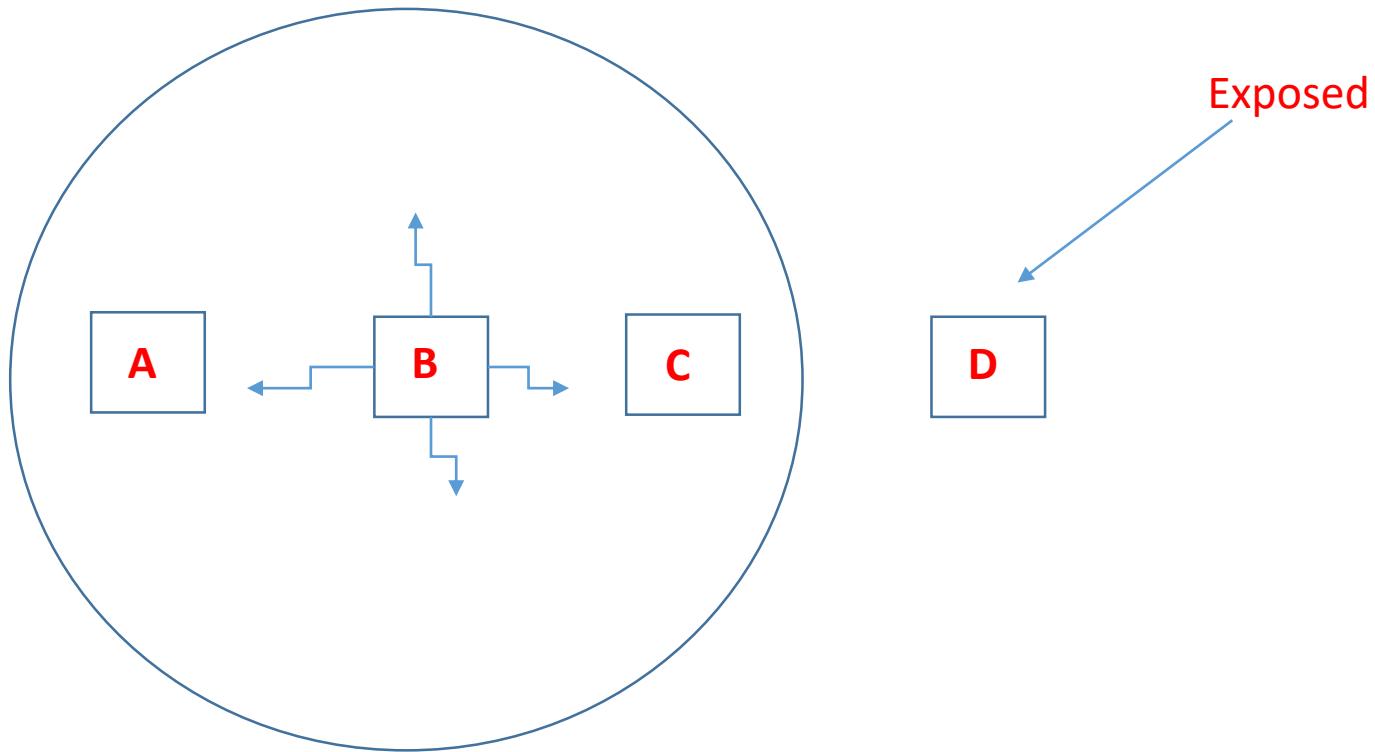


# The Hidden Station Problem



- Both the stations A and C, tries to communicate with B simultaneously.
- Both the signals will reach to B and **collision will occur at B**.
- **But A and C will not be able to detect** this collision because **A does not come in the range of C and vice-versa**.

# The Exposed Station Problem

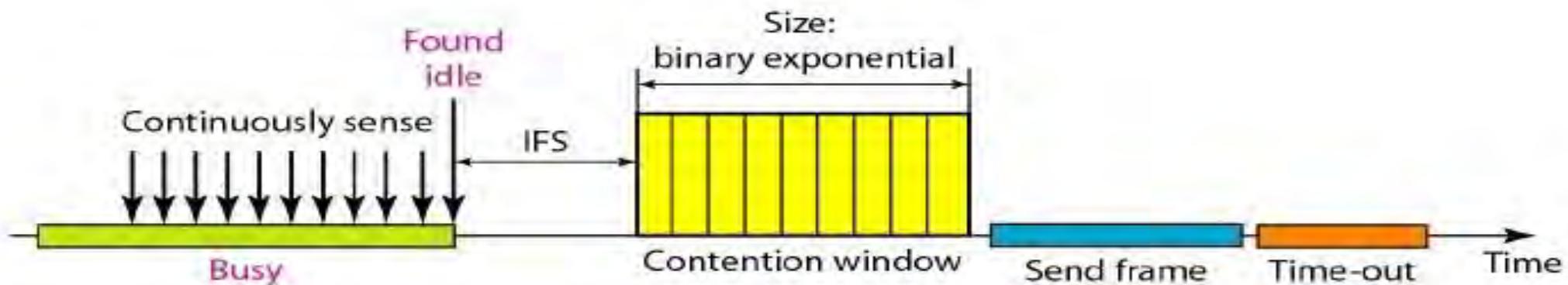


- Station B wants to communicate with A, which is also listened by C.
- Thus, **C assumes** that the medium is not free and thus **it can not communicate with D**.
- But in practice, C can communicate with D.

# CSMA/CA → Its not in your syllabus.

- Developed because **collision detection** is not possible in wireless system.
- In this case, collisions are avoided through the use of CSMA/CA's three strategies:
  1. The inter-frame space
  2. The contention window
  3. Acknowledgement
- **Interframe space:** When an idle channel is found, the station does not send immediately. It waits for a period of time called ***Interframe space*** or **IFS**.

- **Contention window:** It is the amount of time divided into slots. A station which is ready to send chooses a random number of slots as its wait time.
- **Acknowledgement:** Even with all these precautions, there still may be a collision resulting in destroyed data, or corrupted data.



# CSMA: Key-points

- In CSMA/CD Collisions are detected.
- In CSMA/CA collisions are avoided.
- CSMA/CD is used in wired LAN, Ethernet LAN (802.3).
- CSMA/CA is used in wireless LAN, 802.11 standard.
- Collisions are not completely eliminated in CSMA/CA. However, it is reduced as compared to CSMA/CD.
- Collision detection is not possible in wireless LAN because of:
  - Reduction in the strength of the received signal to a greater extent
  - Hidden station problem
  - Exposed station problem

# Refer→

- Chapter 5, Computer Networks- B.A. Forouzan, et al.