

JUNAID GIRKAR

SAP ID: 60004190057

SE COMPUTERS A - 3

COMPUTER NETWORKS

SEM - 4

EXPERIMENT 1.1

AIM: To study different networking devices

What are network devices?

Hardware devices that are used to connect computers, printers, fax machines and other electronic devices to a network are called network devices. These devices transfer data in a fast, secure and correct way over same or different networks. Network devices may be inter-network or intra-network. Some devices are installed on the device, like NIC card or RJ45 connector, whereas some are part of the network, like router, switch, etc. Let us explore some of these devices in greater detail.

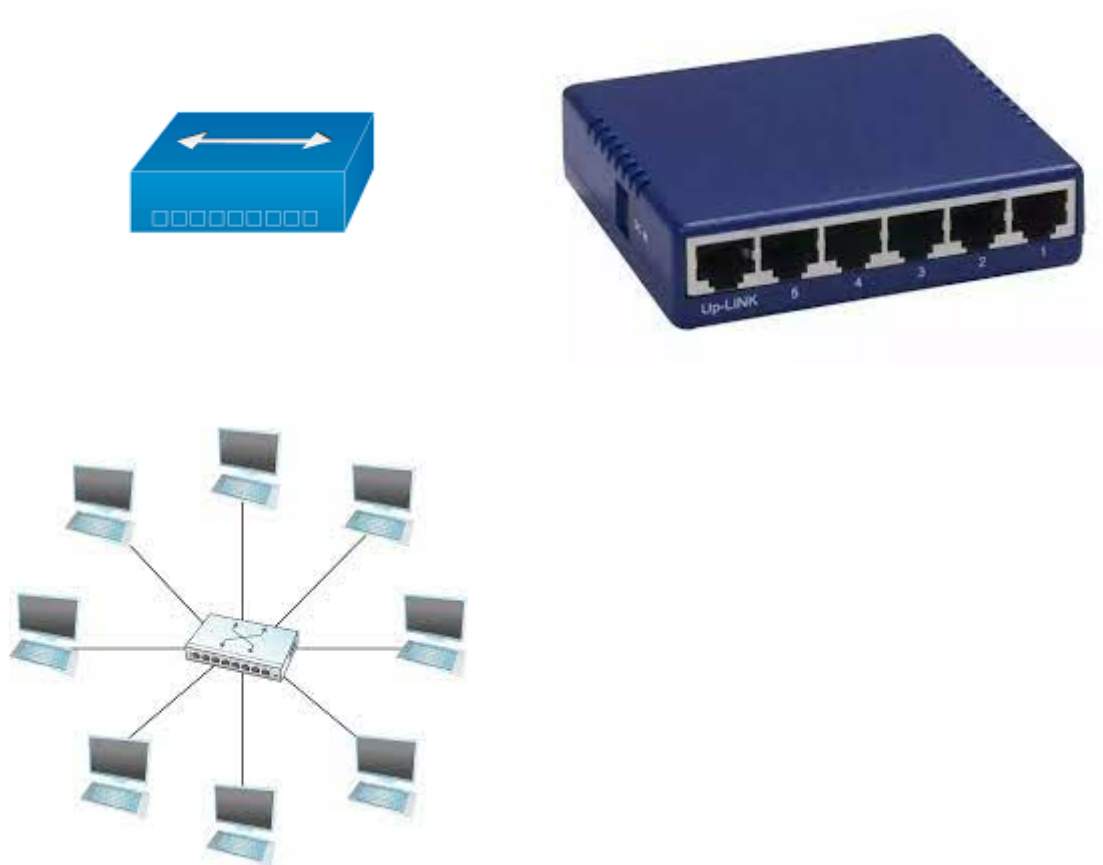
Types of network devices:

Here is a list of commonly used network devices.

- **Hub**
- **Switch**
- **Router**

- Bridge
- Gateway
- Modem
- Repeater
- Access Point

HUB:



Hubs connect multiple computer networking devices together. A hub also acts as a repeater in that it amplifies signals that deteriorate after traveling

long distances over connecting cables. A hub is the simplest in the family of network connecting devices because it connects LAN components with identical protocols.

A hub can be used with both digital and analog data, provided its settings have been configured to prepare for the formatting of the incoming data.

Hubs operate at the Physical layer of the Open Systems Interconnection (OSI) model.

Types of Hub

- **Active Hub:-** These are the hubs which have their own power supply and can clean, boost, and relay the signal along with the network. It serves both as a repeater as well as wiring centre. These are used to extend the maximum distance between nodes.
- **Passive Hub :-** These are the hubs which collect wiring from nodes and power supply from active hub. These hubs relay signals onto the network without cleaning and boosting them and can't be used to extend the distance between nodes.
- **Intelligent Hub :-** It work like active hubs and include remote management capabilities. They also provide flexible data rates to network devices. It also enables an administrator to monitor the traffic passing through the hub and to configure each port in the hub.

APPLICATIONS:

- Hubs are used to create small Home Networks.
- Hubs are used for monitoring the networks.
- Hubs are used in Organizations and Computer Labs for connectivity.
- It Makes one device or peripheral available throughout the whole network.

Working of hub

Network Hub allows the connection with multiple devices through ports for forwarding of the data.

The communication between the hosts can be classified into three forms,

1. a) unicast
2. b) broadcast
3. c) multicast

The unicast of communication can be possible from one device to another.

Multicast communication happens when a commune is formed between one host to a few.

A broadcast of communication can be formed when one host communicates with all the other hosts in the network.

So the hub is a broadcast way of communication.

It works on half-duplex mode, means enables communication to one device at a time or in general terms only one-way transmission is possible in a period and at the other side only one node receives the data.

When the data or packet approach the port then the hub copies the data and sends it to the other devices. It not only sends the data to the destination device but to all the other devices that are connected to the hub.

This generally doesn't read IP addresses and also doesn't filter the data in this process. So, the hub doesn't have any knowledge about where to send the data and cannot identify the better path to traverse too. Hub doesn't consider about which mac address or what IP address is, simply it transmits. Along with forwarding it broadcast. Due to this the congestion increases as the hub is sending and broadcasting data to multiple device.

Even if the data is not required by the device or the system it unnecessarily sends to them due to this bandwidth get wasted.

The external collision is happening but internally also the collision may occur. The internal collision happens when multiple devices start transmitting the data constantly likewise if a device A and B start transmitting at a time. N collisions are proportional to the N nodes. The only task that the hub aware of is when devices are connected to the ports the data packets are traveled. The traveling data packets are first received by the port then the data gets copied to all additional ports. So all the nodes can see that data packet. This was the reason the hub gets replaced by the switch.

The additional functionality about the hub is, It says whether the connection is on/off by blinking the light.

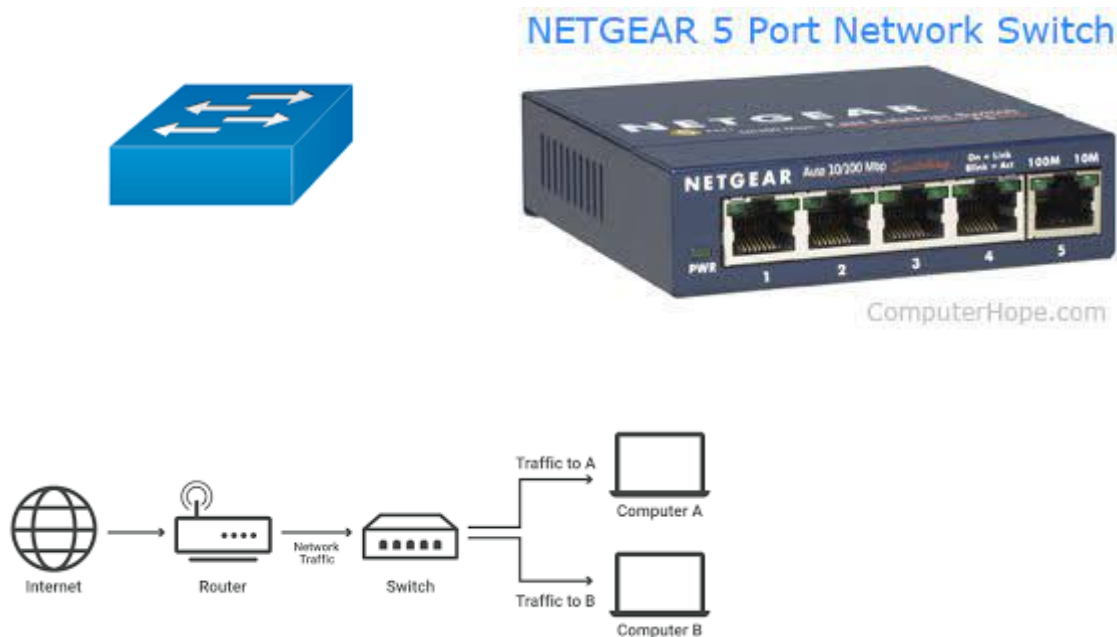
Advantages of Hub

- Usage of hubs is simple and understandable.
- Less expensive than other devices.
- The total distance of the network is expanded.
- It is affordable that means cost efficient.
- Able to connect different media types.
- The performance of the network is not so disturbed.
- The special reason to buy a hub is, using the hub so it can tap the traffic of the internet and analyse it.

Drawbacks of the hub in a Network

- Creates the traffic or congestion in the network.
- Security concerns will be raised because of sending data to every device.
- Disturbs the user with unnecessary data.
- No storage of memory in the hub.
- Collision may or may not occur if there is a possibility of sending the data at a time in a network by a two or more devices.
- This need to be only used in the private network for not raising the security issues.
- If a hub has 10 ports connected then the bandwidth that is shared is 100 Mbps such that each port will be receiving the share of 10 Mbps.
- This can accommodate large networks.

SWITCH:



switches generally have a more intelligent role than hubs. A switch is a multiport device that improves network efficiency. The switch maintains limited routing information about nodes in the internal network, and it allows connections to systems like hubs or routers. Strands of LANs are usually connected using switches. Generally, switches can read the hardware addresses of incoming packets to transmit them to the appropriate destination.

A switch generally can work on the Data Link layer but a multilayer switch is one that can operate at Data Link layer or the Network layer of the OSI model, which means that it can operate as both a switch and a router.

APPLICATIONS

- Switch increases the bandwidth of the network.
- To reduce the workload on individual PCs as it sends the information to only that device which has been addressed.
- To increase the overall performance of the network by reducing the traffic on the network.
- To reduce frame collision as switch creates the collision domain for each connection.

WORKING OF A SWITCH

Once a device is connected to a switch, the switch notes its media access control (MAC) address, a code that's baked into the device's network-interface card (NIC) that attaches to an ethernet cable that attaches to the switch. The switch uses the MAC address to identify which attached device outgoing packets are being sent from and where to deliver incoming packets.

So the MAC address identifies the physical device as opposed to the network layer (Layer 3) IP address, which can be assigned dynamically to a device and change over time.

When a device sends a packet to another device, it enters the switch and the switch reads its header to determine what to do with it. It matches the destination address or addresses and sends the packet out through the appropriate ports that leads to the destination devices.

To reduce the chance for collisions between network traffic going to and from a switch and a connected device at the same time, most switches offer full-duplex functionality in which packets coming from and going to a device have access to the full bandwidth of the switch connection. (Picture two people talking on a cell phone as opposed to a walkie-talkie).

While it's true that switches operate at Layer 2, they can also operate at Layer 3, which is necessary for them to support virtual LANs (VLAN), logical network segments that can span subnets. In order for traffic to get from one subnet to another it must pass between switches, and this is facilitated by routing capabilities built into the switches.

Advantages of Switches :

1. Increases Capacity –

They increment the accessible data transfer capacity of the organization.

2. Reduces Burden –

They help in lessening the outstanding burden on individual host PCs.

3. Increment Presentation –

They increment the presentation of the organization.

4. Less casing Impacts –

Networks that use switches will have fewer casing impacts. This is because of the way that switches make impact areas for every association.

5. Straightforward –

Switches can be associated straightforwardly with workstations.

6. Increases Bandwidth –

It increases the available bandwidth of the network.

7. Less frame collisions –

Networks that use switches will have fewer frame collisions

8. More secure –

Since the switch is isolated, data will go only to the destination.

Disadvantages of Switches :

1. Costly –

They are more costly in contrast with network spans.

2. Tough Availability issues –

Network availability issues are hard to be followed through the organization switch.

3. Issues in traffic broadcasting –

Broadcast traffic might be problematic.

4. Defenseless –

If switches are in the indiscriminate mode, they are defenseless against security assaults for example caricaturing IP address or catching Ethernet outlines.

5. Need for Proper Planning –

Proper planning and arrangement are required to deal with multicast parcels.

6. Mechanical Component can wear out –

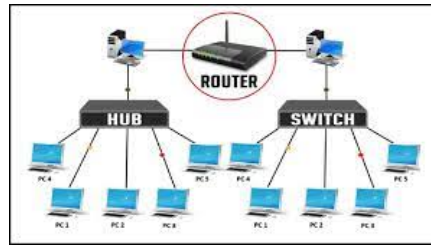
The switch's mechanical component can wear out with time.

7. Physical contact is mandatory –

Must have physical contact with the object to be actuated.

ROUTER:





A router is a network layer hardware device that transmits data from one LAN to another if both networks support the same set of protocols. So a router is typically connected to at least two LANs and the internet service provider (ISP). It receives its data in the form of packets, which are data frames with their destination address added. Router also strengthens the signals before transmitting them. That is why it is also called repeater.

Types of Routers

I . Edge Routers

This type of router is placed at the edge of the ISP network, that is normally configured to external protocol like BGP (Border gateway protocol) to another BGP of other ISP or large organization.

II . Subscriber Edge Routers

This type of router belongs to an end user (enterprise) organization. It's configured to broadcast external BGP to its provider's AS(s)

III. Inter-provider Border Routers

This type of router is for Interconnecting ISPs. This is a BGP speaking router that maintains BGP sessions with other BGP speaking routers in other providers' ASes.

IV. Core Routers

A router that resides within the middle or backbone of the LAN network rather than at its periphery. In some instances, a core router provides a stepdown backbone, interconnecting the distribution routers from multiple building of a campus (LAN), or Large Enterprise Location (WAN). They tend to be optimized for a high bandwidth.

V . Wired and Wireless Routers

Home and small office networking is becoming popular by day by the use of IP wired and wireless router. Wired and wireless router are able to maintain routing and configuration information in their routing table. They also provide the service of filtering traffic of incoming and outgoing packets based on IP addresses.

Some wireless routers can be combined the functions of router with those of a network switch and that of a firewall in one.

Applications of Routers

There are various areas where a router is used:

- Routers are used to connect hardware equipment with remote location networks like **BSC, MGW, IN, SGSN**, and other servers.
- It provides support for a fast rate of data transmission because it uses high STM links for connectivity; that's why it is used in both wired or wireless communication.
- Internet service providers widely use routers to send the data from source to destination in the form of e-mail, a web page, image, voice, or a video file. Furthermore, it can send data all over the world with the help of an IP address of the destination.
- Routers offer access restrictions. It can be configured in a way that allows for few users to access the overall data and allows others to access the few data only, which is defined for them.
- Routers are also used by software testers for WAN communications. For example, the software manager of an organization is located in Agra, and its executive is located at a different place like Pune or Bangalore. Then the router provides the executive the method to share his software tools and other applications with the manager with the help of routers by connecting their PCs to the router using WAN architecture.
- In wireless networks, by configuring VPN in routers, it can be used in the client-server model, which allows sharing the internet, video, data, voice, and hardware resources.

How a router works

A router examines a packet header's destination IP address and compares it against a routing table to determine the packet's best next hop. Routing tables list directions for forwarding data to particular network destinations, sometimes in the context of other variables, like cost. They amount to an algorithmic set of rules that calculate the best way to transmit traffic toward any given IP address.

A routing table often specifies a default route, which the router uses whenever it fails to find a better forwarding option for a given packet. For example, the typical home office router directs all outbound traffic along a single default route to its internet service provider (ISP).

Routing tables can be static -- i.e., manually configured -- or dynamic. Dynamic routers automatically update their routing tables based on network activity, exchanging information with other devices via routing protocols.

Many routers also perform network address translation (NAT), shielding the private IP addresses of a local area network (LAN) by readdressing all outgoing traffic with a single shared public IP address. NAT helps both conserve globally valid IP addresses and improve network security.

Advantages of Router :

1. Association –

The essential capacity of switch is to divide a solitary organization association between various machines. Utilizing this different clients can be associated with the web so in general profitability can be expanded. Other than that switches likewise capacities to associate diverse organization structures and medias.

2. Security –

Having a Routers is unquestionably an initial move towards making sure about an organization association. Since interfacing with the web straightforwardly with a modem opens your PC to assortment of security dangers. Subsequently, switches can be utilized as a transitionally between 2 organizations so the climate is secure somewhat. While this isn't a trade for an antivirus or a firewall, positively it is commendable.

3. Dynamic Routing –

For encouraging internetwork correspondence, the Routers utilizes dynamic directing strategies. Dynamic directing decides the best way accessible for the internetwork. And furthermore it makes broadcast and impact areas. This would overall be able to diminish the organization traffic.

4. Bundle Filtering –

Different administrations of Routers incorporates bundle exchanging and parcel separating Routers channel the organization utilizing a bunch of separating rules. As per this standards the bundles are either permitted or gone through.

5. Reinforcement Plan –

On the off chance that in the event that one of the outside organization segments ends up falling flat, switches utilize elective parts for staying away from issues in directing rush hour gridlock. Associations particularly of huge scope take utilization of this to oversee traffic effectively.

6. NAT Usage –

Routers take utilization of Network Address Translation (NAT). With NAT Routers can share the association by utilizing single public IP address and portion of UDP ports. Additionally it is practically unimaginable for huge organizations to associate with the web without NAT.

7. Reconciliations –

Routers can normally be coordinated with modems. This guarantees that Wireless Access Points are given to make little organizations.

Disadvantages of Router :

1. Speed –

In contrast to repeaters and scaffolds, Routers doesn't simply peruse 2 layers of information's. It totally investigates information from physical to arrange layer. Subsequently the association could turn out to be moderate. Also utilizing switches, numerous PCs can share the organization for which the switch goes through a circumstance known as "Connection Wait". This perhaps eases back the association more.

2. Cost Routers –

Are expensive more than some other systems administration gadgets. This incorporates center point, extension and security. Accordingly, Routers are not generally the best choice regarding cost.

3. Similarity –

There is likewise a similarity issues for the Routers particularly for the 5GHz recurrence. Except if your PC and its connectors do uphold designs of 5GHz recurrence you can't get its advantages. Subsequently, you ought to think about going for a more less expensive Routers.

4. Unwavering quality –

Not all the time switches are dependable. Still some cutting edge gadgets utilize 2.4GHz range which often gets disengaged. Most now and again these sorts of separations are capable by individuals living in pads and condos.

5. Usage –

An average Router requires heaps of beginning arrangements and NAT to arrangement. And furthermore in any event, for the easiest association there should be a private IP address doled out. On the off chance that there are more administrations empowered it needs more designs too. This fundamentally makes more complexities in the arrangement.

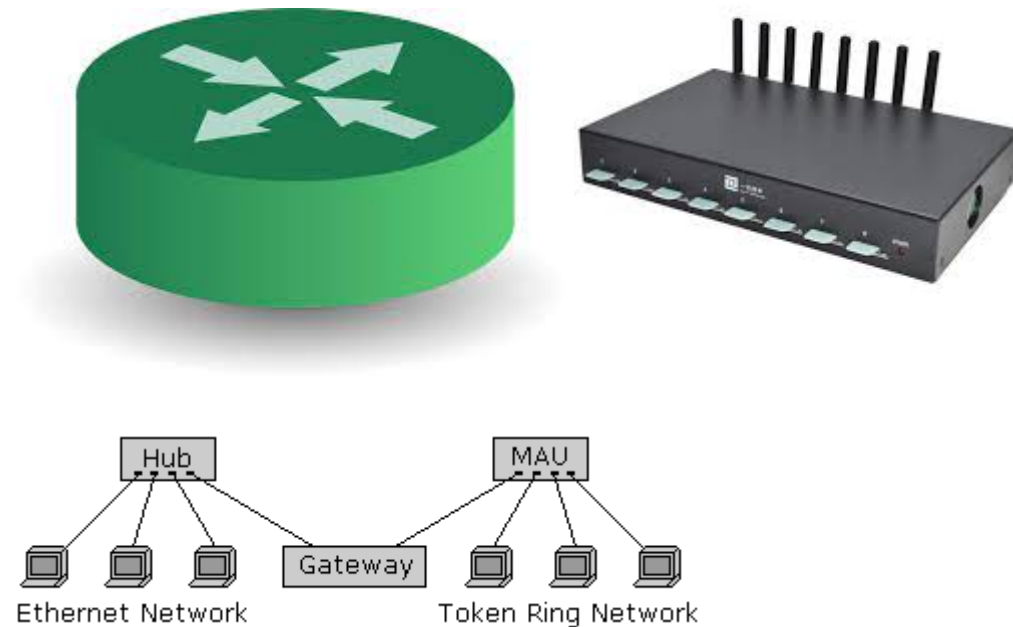
6. Data transmission Shortage –

Dynamic directing methods are utilized by the switches for the correspondence purposes. This conceivably causes additionally organizing overheads. Systems administration overheads devour huge measure of transmission capacities bringing about transfer speed deficiencies. Moreover for keeping up steering tables, switches routinely update on the organization. This also can cause data transfer capacity utilization.

7. Convention Support –

Routers just does its tasks utilizing routable conventions. A convention that is routable can give remarkable IP delivers to the gadgets so it very well may be recognized across networks.

GATEWAY:



Gateways normally work at the Transport and Session layers of the OSI model. At the Transport layer and above, there are numerous protocols and standards from different vendors; gateways are used to deal with them. Gateways provide translation between networking technologies such as Open System Interconnection (OSI) and Transmission Control Protocol/Internet Protocol (TCP/IP). Because of this, gateways connect two or more autonomous networks, each with its own routing algorithms, protocols, topology, domain name service, and network administration procedures and policies.

Gateways perform all of the functions of routers and more. In fact, a router with added translation functionality is a gateway.

Types of gateways

Gateways can take several forms and perform a variety of tasks. Examples of this include:

- [Web application firewalls](#)- This type filters traffic to and from a web server and looks at application-layer data.
- [Cloud storage gateways](#)- This type translates storage requests with various cloud storage service API calls. It allows organizations to integrate storage from a private cloud into applications without migrating into a public cloud.
- [API](#), [SOA](#) or [XML](#) gateways – This type manages traffic flowing into and out of a service, microservices-oriented architecture or XML-based web service.
- [IoT gateways](#)-This type aggregates sensor data from devices in an IoT environment, translates between sensor protocols and processes sensor data before sending it onward.
- [Media gateways](#)- This type converts data from the format required for one type of network to the format required for another.
- [Email security gateways](#)- This type prevents the transmission of emails that break company policy or will transfer information with malicious intent.
- [VoIP trunk gateways](#)- This type facilitates the use of plain old telephone service equipment, such as landline phones and fax machines, with a voice over IP (VoIP) network.

APPLICATIONS OF A GATEWAY:

1. The gateways work as the software sometimes performs as the hardware and sometimes it can be both. The voice and data communication are the techniques which are being used as the data types.
2. It provides complete security to a local network and also connects a local network to a public network gateway.
3. The gateways are also an important aspect or we can utter that is the important technique for any of the telephony signaling process.
4. It provides a way of communication between the telephone network and the internet.
5. It implements the significant interface between the wide-area and local protocols such as TCP/IP on the major intranet/internet.

WORKING OF A GATEWAY

It is a point of a network that can access other networks. Usually, in the intranet, a router or node can act as a gateway node or the router that links the networks are called gateways. In large scale enterprises, the computers manage the traffic between enterprise networks are termed as gateway nodes. Such as that the computers used by Internet service providers to link varied users to each other at an instant time to the internet are gateway nodes. In any development team of any commercial enterprise computer server functions as gateway nodes and it may also be a proxy server or a firewall at times.

Advantages of Gateway

1. Connectivity

As mentioned earlier, the main benefit of gateway is the connectivity it provides. A gateway can expand the network by connecting computers with different systems together. Through this, same kind of information will be able to be accessed by different computers.

2. Security

Gateways are known to possess improved security since they allow user authentication. Forms of security such as User ID and Password can be imposed on gateway so that all the unwanted access will be prevented. This not only protects sensitive information, it also ensures that only the authorized users have access to the informations.

3. Filtering Process

Filtering process is another important capability of a gateway. Without them, whatever the services that arrives at the gateway carries the risk of theft. Therefore, gateway performs the filtering process by inspecting each data packet that passed through the gateway.

4. Domain Control

When the number of Collision and Broadcast domain increases, the network provider can assure that they will be able to provide better bandwidth. A networking gateway has the capability to control collision as well as broadcast domain.

5. Protocol Conversion

Besides filtering, a gateway can also convert data packets according to the destination needs. If the destination network or architecture has different needs, it can also convert the data format accordingly. That is the reason why it is also called as the Protocol Converter.

Disadvantages of a Gateway

1. Implementation

Generally gateways on default are installed on the routers itself. This makes it more difficult for the network administrators to install or configure them. Apart from this the cost involved in the implementation process is too high.

2. Configuration

Configuration of devices through a gateway is made even more difficult or impossible. There must be special system administration for this purpose.

3. Time Delay

Gateway networks always causes time delay since information must be translated. There is no way can a instant transfer take place. In addition to that, a gateway must also return old cache information that is not cleared properly. This can take more time resulting in time constraints.

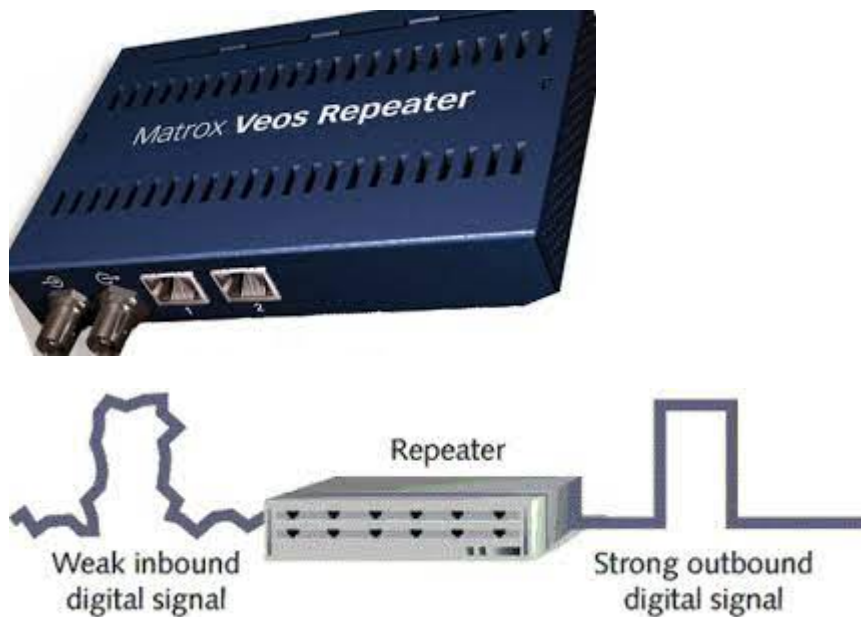
4. Connection Failure

If there are possibilities of failure occurring at the gateway, it can lead to communication loss. Devices on the opposite side will no longer be able to communicate until the problem is resolved.

5. Troubleshooting

Computers on a network are with different protocols. Therefore, if there is any problem, each of these computers needed to be troubleshooted individually. This makes the process more complicated since different tools must be present.

REPEATER:



A repeater operates at the physical layer. Its job is to regenerate the signal over the same network before the signal becomes too weak or corrupted so as to extend the length to which the signal can be transmitted over the same network. An important point to be noted about repeaters is that they do not amplify the signal. When the signal becomes weak, they copy the signal bit by bit and regenerate it at the original strength. It is a 2 port device.

Types of Repeaters

According to the types of signals that they regenerate, repeaters can be classified into two categories –

- **Analog Repeaters** – They can only amplify the analog signal.
- **Digital Repeaters** – They can reconstruct a distorted signal.

According to the types of networks that they connect, repeaters can be categorized into two types –

- **Wired Repeaters** – They are used in wired LANs.
- **Wireless Repeaters** – They are used in wireless LANs and cellular networks.

According to the domain of LANs they connect, repeaters can be divided into two categories -

- **Local Repeaters** – They connect LAN segments separated by small distance.
- **Remote Repeaters** – They connect LANs that are far from each other.

APPLICATION OF A REPEATER

- To retransmit the data and strengthen the weak signals as the Signals get weaker as it travels to the longer distances.

WORKING OF A REPEATER

As we have mentioned, the repeaters are the network devices that retransmit the data and directs the signals for weak network access locations. The Data transmission and receiving of the data have different frequencies for both sender and receiver sides. The Repeater works when the sender's frequency and the receiver's frequencies are matched.

Let us take an example to understand in detail about the Repeaters

Many of you may have heard about the term Walkie Talkie. Walkie Talkie has a simple connection. The data can be transmitted without any congestions or error to the receiver. The data can be transferred only if there are no obstacles in the middle of the transmission. Let go in deep to understand the concept in detail. Suppose a person interacts with the other person by using a one to one communication device like a walkie talkie. The data can be successfully transmitted if there is a clear path between the distance. The data cannot be delivered perfectly if there is a mountain or hill in between the distance.

So in order to avoid this situation, an Antenna is placed in between the two devices. This Device retransmits the data to the receiver side and leads the signals to the weak areas. This is called the main functioning of the repeater.

Advantages of Repeaters

- Repeaters are simple to install and can easily extend the length or the coverage area of networks.
- They are cost effective.
- Repeaters don't require any processing overhead. The only time they need to be investigated is in case of degradation of performance.
- They can connect signals using different types of cables.

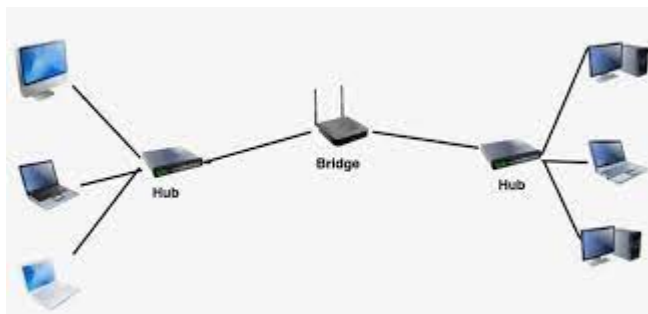
Disadvantages of Repeaters

- Repeaters cannot connect dissimilar networks.
- They cannot differentiate between actual signal and noise.
- They cannot reduce network traffic or congestion.
- Most networks have limitations upon the number of repeaters that can be deployed.

BRIDGE:



Bridge



A bridge operates at data link layer. A bridge is a repeater, with add on the functionality of filtering content by reading the MAC addresses of source and destination. It is also used for interconnecting two LANs working on the same protocol. It has a single input and single output port, thus making it a 2 port device.

Bridges have mostly fallen out of favor in recent years and have been replaced by switches, which offer more functionality. In fact, switches are sometimes referred to as “multiport bridges” because of how they operate.

Types of Bridges

- **Transparent Bridges:-** These are the bridge in which the stations are completely unaware of the bridge's existence i.e. whether or not a bridge is added or deleted from the network, reconfiguration of the stations is unnecessary. These bridges make use of two processes i.e. bridge forwarding and bridge learning.
- **Source Routing Bridges:-** In these bridges, routing operation is performed by source station and the frame specifies which route to follow. The host can discover frame by sending a special frame called discovery frame, which spreads through the entire network using all possible paths to destination.

APPLICATIONS OF BRIDGE:

- Bridges connects two or more different LANs that has a similar protocol and provides communication between the devices (nodes) in them.
- By joining multiple LANs, bridges help in multiplying the network capacity of a single LAN.
- Since they operate at data link layer, they transmit data as data frames. On receiving a data frame, the bridge consults a database to decide whether to pass, transmit or discard the frame.
 - If the frame has a destination MAC (media access control) address in the same network, the bridge passes the frame to that node and then discards it.
 - If the frame has a destination MAC address in a connected network, it will forward the frame toward it.
- By deciding whether to forward or discard a frame, it prevents a single faulty node from bringing down the entire network.
- In cases where the destination MAC address is not available, bridges can broadcast data frames to each node. To discover new segments, they maintain the MAC address table.
- In order to provide full functional support, bridges ideally need to be transparent. No major hardware, software or architectural changes should be required for their installation.

- Bridges can switch any kind of packets, be it IP packets or AppleTalk packets, from the network layer above. This is because bridges do not examine the payload field of the data frame that arrives, but simply looks at the MAC address for switching.
- Bridges also connect virtual LANs (VLANs) to make a larger VLAN.
- A wireless bridge is used to connect wireless networks or networks having a wireless segment.

Working Principle

The working principle of a bridge is, it blocks or forwards the data depending on the destination MAC address and this address is written into every data frame.

In a computer network, a bridge separates a LAN into different segments like segment1 & segment2, etc and the MAC address of all the PCs can be stored into the table. For instance, PC1 transmits the data to PC2, where the data will transmit to the bridge first. So the bridge reads the MAC address & decides whether to transmit the data to segment1 or segment2. Therefore, the PC2 is accessible in segment1, which means the bridge transmits the data in segment1 only & eliminates all the connected PCs in segment2. In this way, the bridge reduces traffic in a computer network.

Advantages of Network Bridges

Following are the benefits or **advantages of Network Bridges**:

- ➡ It helps in extension of physical network.
- ➡ It reduces network traffic with minor segmentation.
- ➡ It creates separate collision domains. Hence it increases available bandwidth to individual nodes as fewer nodes share a collision domain.
- ➡ It reduces collisions.
- ➡ Some bridges connect networks having different architectures and media types.

Disadvantages of Network Bridges

Following are the **disadvantages of Bridges**:

- ➡ It is slower compare to repeaters due to filtering.
- ➡ It does not filter broadcasts.
- ➡ It is more expensive compare to repeaters

CONCLUSION: We learnt about the different types of networking devices that are used, their applications, working, advantages and disadvantages. We also learnt their symbols which are used to represent those devices in a network diagram.

EXPERIMENT 1.2

AIM: To study different network topologies

What is Network Topology?

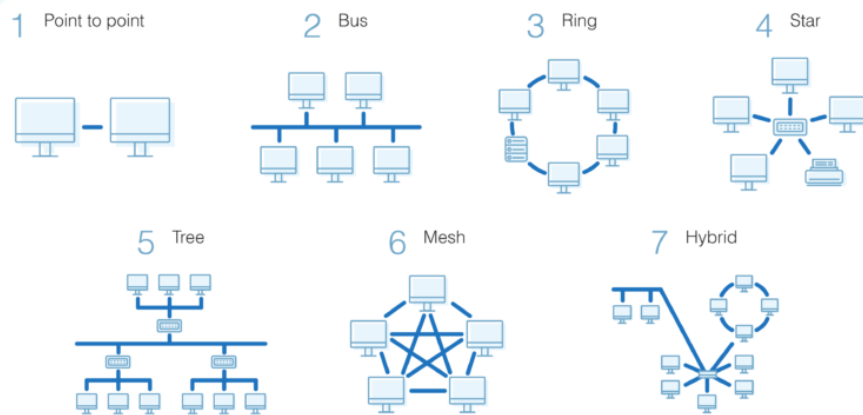
Network topology refers to the manner in which the links and nodes of a network are arranged to relate to each other. Topologies are categorized as either physical network topology, which is the physical signal transmission medium, or logical network topology, which refers to the manner in which data travels through the network between devices, independent of physical connection of the devices. Logical network topology examples include twisted pair Ethernet, which is categorized as a logical bus topology, and token ring, which is categorized as a logical ring topology.

Physical network topology examples include star, mesh, tree, ring, point-to-point, circular, hybrid, and bus topology networks, each consisting of different configurations of nodes and links. The ideal network topology depends on each business's size, scale, goals, and budget. A network topology diagram helps visualize the communicating devices, which are modeled as nodes, and the connections between the devices, which are modeled as links between the nodes.

TYPES OF NETWORK TOPOLOGIES

Building a local area network (LAN) topology can be make-or-break for your business, as you want to set up a resilient, secure, and easy-to-maintain topology. There are several different types of network topology and all are suitable for different purposes, depending on the overall network size and your objectives.

Network Topology Types

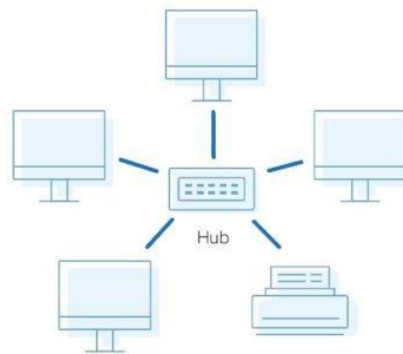


As with most things, there's no "right" or one-size-fits-all option. With this in mind, I'll walk you through the most common network topology definitions to give you a feel for the advantages and disadvantages of each.

What Is Star Topology?

A star topology, the most common network topology, is laid out so every node in the network is directly connected to one central hub via coaxial, twisted-pair, or fiber-optic cable. Acting as a server, this central node manages data transmission—as information sent from any node on the network has to pass through the central one to reach its destination—and functions as a repeater, which helps prevent data loss.

Star Topology



Features of star topology are :

1. Star topology is easy to install.
2. Star topology requires more cable than bus topology.
3. In star topology all devices revolve around the central hub.
4. No disruption to the network when removing or connecting devices.
5. In star topology switch or hub controls the network communications and can communicate with other hubs.
6. In star network if hub fails then entire network fails.

ARCHITECTURE:

In a star architecture, communication flows from network nodes to a central hub over one set of channels, and from the hub to the nodes over a separate set of channels.

Since star architectures have only one hop between a network node and the central hub, they tend to be more predictable and reliable; however, if that connection is weak, then there is no alternative connection. A big advantage of star architectures is that they can use centralized control functions at the hub for channel estimation, access, routing, and resource allocation. This centralized control usually results in a more efficient and reliable network, and for this reason many commercial wireless networks use the star architecture.

Advantages of Star Topology

Star topologies are common since they allow you to conveniently manage your entire network from a single location. Because each of the nodes is independently connected to the central hub, should one go down, the rest of the network will continue functioning unaffected, making the star topology a stable and secure network layout.

Additionally, devices can be added, removed, and modified without taking the entire network offline.

On the physical side of things, the structure of the star topology uses relatively little cabling to fully connect the network, which allows for both straightforward setup and management over time as the network expands or contracts. The simplicity of the network design makes life easier for administrators, too, because it's easy to identify where errors or performance issues are occurring.

Disadvantages of Star Topology

On the flipside, if the central hub goes down, the rest of the network can't function. But if the central hub is properly managed and kept in good health, administrators shouldn't have too many issues.

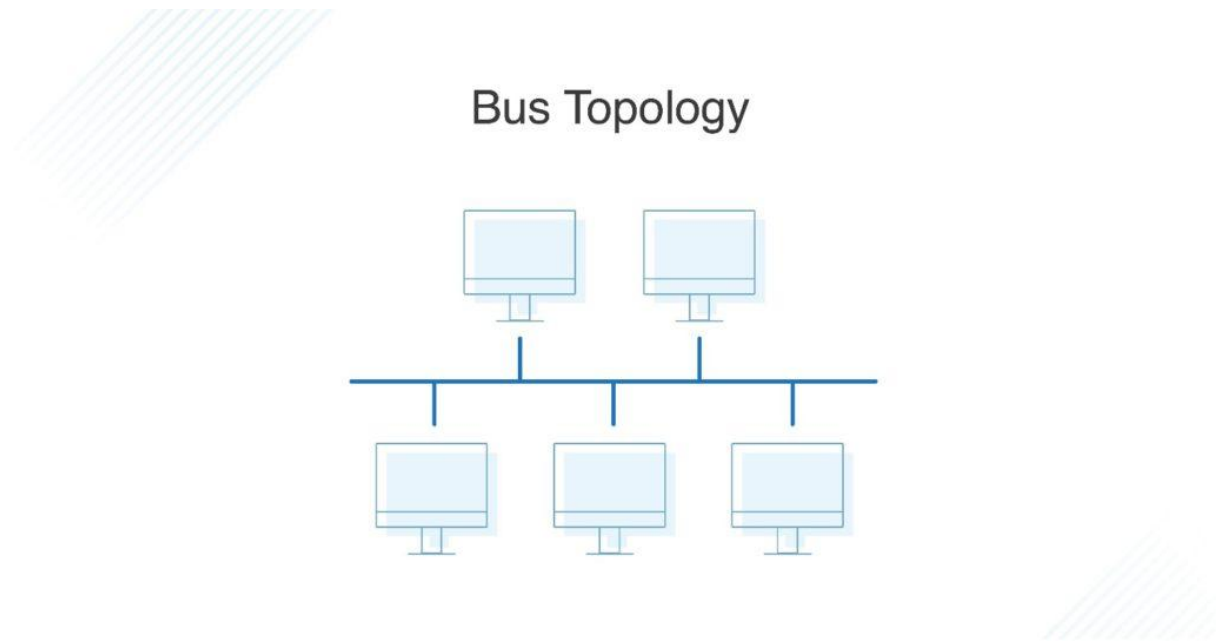
The overall bandwidth and performance of the network are also limited by the central node's configurations and technical specifications, making star topologies expensive to set up and operate.

Applications of Star topology:

1. Star topology is used in Local Area Network (LAN)
2. High speed LAN often uses the star topology.
3. Star topology is often used in homes and offices.
4. Star topology is also used to transmit data along the central hub between the network nodes.
5. By connecting all the systems to the central hubs, star topology can ease the probability of network failure.

What Is Bus Topology?

A bus topology orients all the devices on a network along a single cable running in a single direction from one end of the network to the other—which is why it's sometimes called a “line topology” or “backbone topology.” Data flow on the network also follows the route of the cable, moving in one direction.



Features of Bus Topology

1. It transmits data only in one direction.
2. Every device is connected to a single cable

ARCHITECTURE:

In Bus Topology used in computer networks, all the computers are connected through a single cable. Usually Ethernet cable is used for Bus Topology. In this topology, the information intended for the last node has to pass through all the computers present in the network. If this cable is damaged then the connection of all the computers will be lost.

Instead of cable either network card, co-axial cable or RJ-47 can be used depending on the type of computers used in the network. When Bus Topology has only two endpoints, it is known as Linear Topology. In Bus topology data is transmitted only in one direction

Advantages of Bus Topology

Bus topologies are a good, cost-effective choice for smaller networks because the layout is simple, allowing all devices to be connected via a single coaxial or RJ45 cable. If needed, more nodes can be easily added to the network by joining additional cables.

Disadvantages of Bus Topology

However, because bus topologies use a single cable to transmit data, they're somewhat vulnerable. If the cable experiences a failure, the whole network goes down, which can be time-consuming and expensive to restore, which can be less of an issue with smaller networks.

Bus topologies are best suited for small networks because there's only so much bandwidth, and every additional node will slow transmission speeds.

Furthermore, data is "half-duplex," which means it can't be sent in two opposite directions at the same time, so this layout is not the ideal choice for networks with huge amounts of traffic.

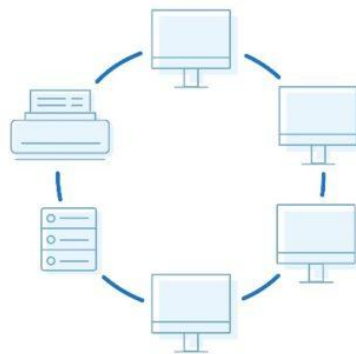
APPLICATIONS:

- Small workgroup local area networks (LANs) whose computers are connected using a thinnet cable
- Trunk cables connecting hubs or switches of departmental LANs to form a larger LAN
- Backboning, by joining switches and routers to form campus-wide networks

What Is Ring Topology? Single vs. Dual

Ring topology is where nodes are arranged in a circle (or ring). The data can travel through the ring network in either one direction or both directions, with each device having exactly two neighbors.

Ring Topology



Features of Ring Topology

1. A number of repeaters are used for Ring topology with large number of nodes, because if someone wants to send some data to the last node in the ring topology with 100 nodes, then the data will have to pass through 99 nodes to reach the 100th node. Hence to prevent data loss repeaters are used in the network.
2. The transmission is unidirectional, but it can be made bidirectional by having 2 connections between each Network Node, it is called Dual Ring Topology.
3. In Dual Ring Topology, two ring networks are formed, and data flow is in opposite direction in them. Also, if one ring fails, the second ring can act as a backup, to keep the network up.
4. Data is transferred in a sequential manner that is bit by bit. Data transmitted, has to pass through each node of the network, till the destination node.

ARCHITECTURE:

The ring architecture is a distributed architecture, with minimal connectivity and a topology of two links connected to every node. Transmitted messages travel from node to node around the ring. Each node must be able to recognize its own address in order to accept messages. Information travels around the ring in only one direction, with each attached station or node serving as, a repeater. Rings generally are coaxial cable or fiber in nature, operating at raw transmission rates of 4, 16, 20 or 100Mbps or more. Rings are deterministic in nature, employing token passing as the method of media access control to ensure the ability of all nodes to access the network within a predetermined time interval.

Advantages of Ring Topology

Since each device is only connected to the ones on either side, when data is transmitted, the packets also travel along the circle, moving through each of the intermediate nodes until they arrive at their destination. If a large network is arranged in a ring topology, repeaters can be used to ensure packets arrive correctly and without data loss.

Only one station on the network is permitted to send data at a time, which greatly reduces the risk of packet collisions, making ring topologies efficient at transmitting data without errors.

By and large, ring topologies are cost-effective and inexpensive to install, and the intricate point-to-point connectivity of the nodes makes it relatively easy to identify issues or misconfigurations on the network.

Disadvantages of Ring Topology

Even though it's popular, a ring topology is still vulnerable to failure without proper network management. Since the flow of data transmission moves unidirectionally between nodes along each ring, if one node goes down, it can take the entire network with it. That's why it's imperative for each of the nodes to be monitored and kept in good health. Nevertheless, even if you're vigilant and attentive to node performance, your network can still be taken down by a transmission line failure.

The question of scalability should also be taken into consideration. In a ring topology, all the devices on the network share bandwidth, so the addition of more devices can contribute to overall communication delays. Network administrators need to be mindful of the devices added to the topology to avoid overburdening the network's resources and capacity.

Additionally, the entire network must be taken offline to reconfigure, add, or remove nodes. And while that's not the end of the world, scheduling downtime for the network can be inconvenient and costly.

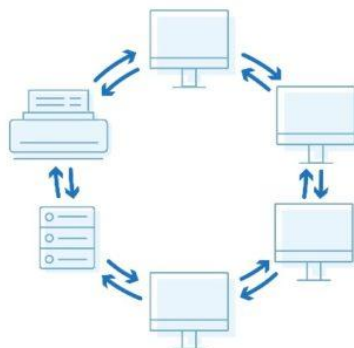
APPLICATIONS:

- Ring Topology is deployed in a Local area network (LAN) and a Wide area network (MAN) as well.
- SONET (Synchronous optical network) fiber networks in the Telecommunication domain uses Ring topology quite extensively.
- It provides a standard for global telecommunication networks which will replace many older systems and it offers many advantages like:

What Is Dual-Ring Topology?

A network with ring topology is half-duplex, meaning data can only move in one direction at a time. Ring topologies can be made full-duplex by adding a second connection between network nodes, creating a dual ring topology.

Dual Ring Topology



Advantages of Dual-Ring Topology

The primary advantage of dual ring topology is its efficiency: because each node has two connections on either side, information can be sent both clockwise and counterclockwise along the network. The secondary ring included in a dual-ring topology setup can act as a redundant layer and backup, which helps solve for many of the disadvantages of traditional ring topology. Dual ring topologies offer a little extra security, too: if one ring fails within a node, the other ring is still able to send data.

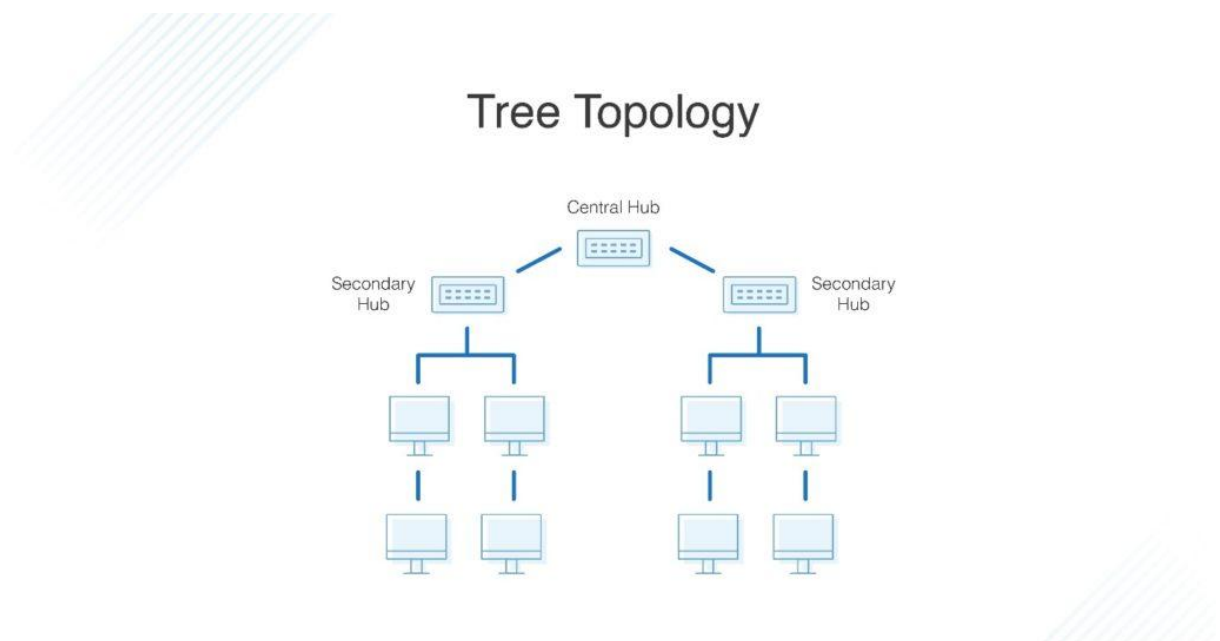
APPLICATIONS:

Dual-ring topology is a network redundant topology where nodes are connected using two concentric rings with four branches. Dual-ring topology is ideal for applications with cabling issues or small networks that are not frequently reconfigured.

Though more expensive than star or extended star topologies, dual-ring is the most cost-efficient redundant topology.

What Is Tree Topology?

The tree topology structure gets its name from how the central node functions as a sort of trunk for the network, with nodes extending outward in a branch-like fashion. However, where each node in a star topology is directly connected to the central hub, a tree topology has a parent-child hierarchy to how the nodes are connected. Those connected to the central hub are connected linearly to other nodes, so two connected nodes only share one mutual connection. Because the tree topology structure is both extremely flexible and scalable, it's often used for wide area networks to support many spread-out devices.



Features of Tree Topology

1. Ideal if workstations are located in groups.
2. Used in Wide Area Network.

ARCHITECTURE:

The overlay topology consists of several layers of peers. Each layer maintains information about its peers. The main objective of this overlay topology is to leverage the capabilities of a heterogeneous network of computers by using a hierarchical structure. The top layers are composed of more stable and efficient peers and the bottom layers consist of unstable and unreliable peers. The concept of hierarchy based on tessellations of space provides a powerful organisational framework. In other words, the tessellation of the network provides a generic basis for a broad class of spatially constrained hierarchical and fractal networks. The algorithms used in tessellations can also be adapted to take into account the locality of peers with low overhead.

Pros of Tree Topology

Combining elements of the star and bus topologies allows for the easy addition of nodes and network expansion. Troubleshooting errors on the network is also a straightforward process, as each of the branches can be individually assessed for performance issues.

Cons of Tree Topology

As with the star topology, the entire network depends on the health of the root node in a tree topology structure. Should the central hub fail, the various node branches will become disconnected, though connectivity within—but not between—branch systems will remain.

Because of the hierarchical complexity and linear structure of the network layout, adding more nodes to a tree topology can quickly make proper management an unwieldy, not to mention costly, experience. Tree topologies are expensive because of the sheer amount of cabling required to connect each device to the next within the hierarchical layout.

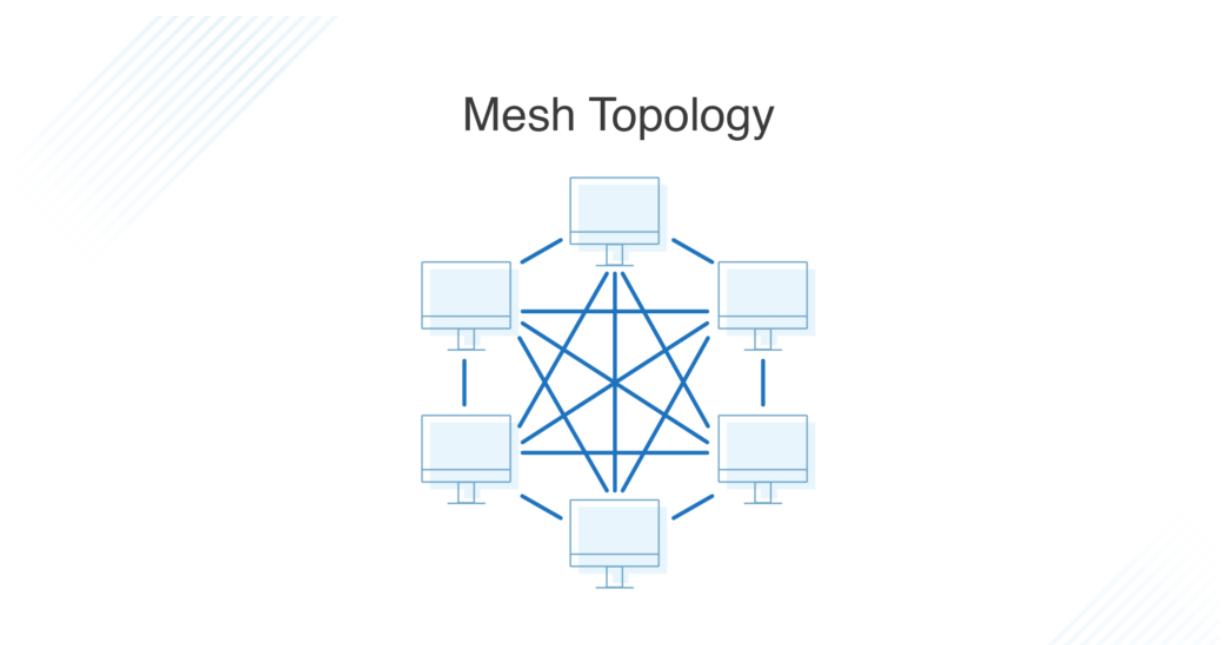
APPLICATIONS:

A tree topology is a special type of structure in which many connected elements are arranged like the branches of a tree. For example, tree topologies are frequently used to organize the computers in a corporate network, or the information in a database.

They are used extensively in database systems like MySQL, PostgreSQL, and Redis, and filesystems such as NTFS, HFS+, and Ext4.

What Is Mesh Topology?

A mesh topology is an intricate and elaborate structure of point-to-point connections where the nodes are interconnected. Mesh networks can be full or partial mesh. Partial mesh topologies are mostly interconnected, with a few nodes with only two or three connections, while full-mesh topologies are—surprise!—fully interconnected.



The web-like structure of mesh topologies offers two different methods of data transmission: routing and flooding. When data is routed, the nodes use logic to determine the shortest distance from the source to destination, and when data is flooded, the information is sent to all nodes within the network without the need for routing logic.

Features of Mesh Topology

1. Fully connected.
2. Robust.
3. Not flexible.

ARCHITECTURE:

Wireless mesh architecture is a first step towards providing cost effective and low mobility over a specific coverage area. Wireless mesh infrastructure is, in effect, a network of routers minus the cabling between nodes. It is built of peer radio devices that do not have to be cabled to a wired port like traditional WLAN access points (AP) do. Mesh infrastructure carries data over large distances by splitting the distance into a series of short hops. Intermediate nodes not only boost the signal, but cooperatively pass data from point A to point B by making forwarding decisions based on their knowledge of the network, i.e. perform routing by first deriving the topology of the network.

Wireless mesh networks is a relatively "stable-topology" network except for the occasional failure of nodes or addition of new nodes. The path of traffic, being aggregated from a large number of end users, changes infrequently. Practically all the traffic in an infrastructure mesh network is either forwarded to or from a gateway, while in wireless ad hoc networks or client mesh networks the traffic flows between arbitrary pairs of nodes.

If rate of mobility among nodes are high, i.e., link breaks happen frequently, wireless mesh networks start to break down and have low communication performance.

Advantages of Mesh Topology

Mesh topologies are reliable and stable, and the complex degree of interconnectivity between nodes makes the network resistant to failure. For instance, no single device going down can bring the network offline.

Disadvantages of Mesh Topology

Mesh topologies are incredibly labor-intensive. Each interconnection between nodes requires a cable and configuration once deployed, so it can also be time-consuming to set up. As with other topology structures, the cost of cabling adds up fast, and to say mesh networks require a lot of cabling is an understatement.

APPLICATIONS:

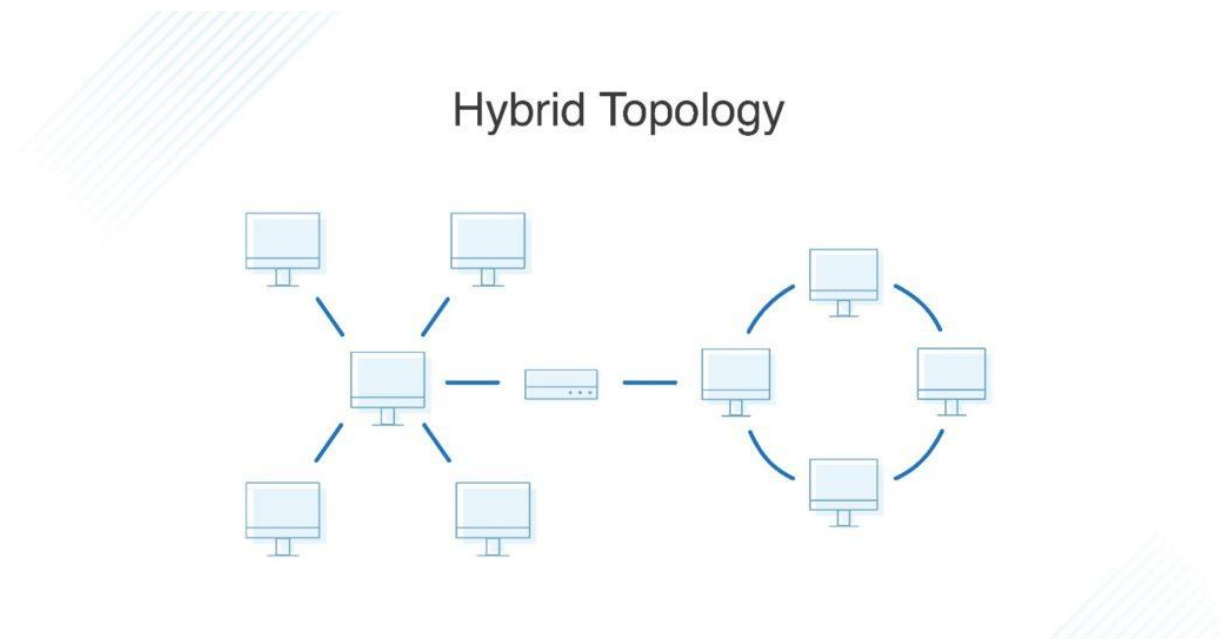
Due to the *Zigbee* platform, you can now find various Mesh Network Topology Applications in real life. You can use it in various scenarios wherever sensor are involved. All you need to do is to configure these useful sensors in the form of *Wireless PAN*. This sort of application will help you in different situations. However, following are some popular applications:

- Patient Monitoring System
- Security Systems
- Sports
- Defense Systems
- Strategic Communications
- Smart Agriculture Control and Monitoring
- Smart Home Control and Monitoring

What Is Hybrid Topology?

Hybrid topologies combine two or more different topology structures—the tree topology is a good example, integrating the bus and star layouts.

Hybrid structures are most commonly found in larger companies where individual departments have personalized network topologies adapted to suit their needs and network usage.



Features of Hybrid Topology

1. It is a combination of two or topologies
2. Inherits the advantages and disadvantages of the topologies included

ARCHITECTURE:

A hybrid architecture is one that combines or adapts one of the previously discussed systems. For example, system manufacturers will connect multiple SMP machines using a high-speed interconnect to create a hybrid system with a communications model involving two different levels of service. *On-node* communication (where a node is a single SMP machine) is significantly faster than *cross-node* communication. Another configuration might connect small MPP (i.e., 16-node) machines, each of which shares some memory with other small MPP machines within a single box. The use of a hybrid architecture may be very dependent on the specific application, because some systems may be better suited to the concurrency specifics associated with each application

Advantages of Hybrid Topology

The main advantage of hybrid structures is the degree of flexibility they provide, as there are few limitations on the network structure itself that a hybrid setup can't accommodate.

Disadvantages of Hybrid Topology

However, each type of network topology comes with its own disadvantages, and as a network grows in complexity, so too does the experience and know-how required on the part of the admins to keep everything functioning optimally. There's also the monetary cost to consider when creating a hybrid network topology.

Application and Uses of Hybrid Topology

- Automation organizations
- Banks sector
- Multi-National companies
- Educational campus
- Research Organizations

CONCLUSION: We learnt about the different type of network topologies that are being used, the features, architecture, advantages, disadvantages and applications of each of them.

EXPERIMENT 2

AIM: To study different networking command

ipconfig Command

Another indispensable and frequently used utility that is used for finding network information about your local machine like IP addresses, DNS addresses etc

Basic Use: Finding Your IP Address and Default Gateway

Ip config has a number of switches the most common are:

`ipconfig /all` - displays more information about the network setup on your systems including the MAC address.

`ipconfig /release` - release the current IP address

`ipconfig /renew` - renew IP address

`ipconfig /?` -shows help

`ipconfig/flushdns` - flush the dns cache

OUTPUT:

```
C:\Users\junai>ipconfig
```

Windows IP Configuration

Wireless LAN adapter Local Area Connection* 1:

Media State : Media disconnected

Connection-specific DNS Suffix . :

Wireless LAN adapter Local Area Connection* 10:

Media State : Media disconnected

Connection-specific DNS Suffix . :

Wireless LAN adapter Wi-Fi:

Connection-specific DNS Suffix . :

IPv6 Address. : 2409:4042:2211:aef1:d0c9:2cb:24a9:1206

Temporary IPv6 Address. : 2409:4042:2211:aef1:f17b:61f8:61f:e6b9

Link-local IPv6 Address : fe80::d0c9:2cb:24a9:1206%15

IPv4 Address. : 192.168.43.197

Subnet Mask : 255.255.255.0

Default Gateway : fe80::aa96:75ff:fe6a:ff70%15

192.168.43.1

Ethernet adapter Bluetooth Network Connection:

Media State : Media disconnected

Connection-specific DNS Suffix . :

```
C:\Users\junai>ipconfig /all
```

Windows IP Configuration

Host Name : DESKTOP-TO06OTG

Primary Dns Suffix :

Node Type : Hybrid

IP Routing Enabled. : No

WINS Proxy Enabled. : No

Wireless LAN adapter Local Area Connection* 1:

Media State : Media disconnected

Connection-specific DNS Suffix . :

Description : Microsoft Wi-Fi Direct Virtual Adapter

Physical Address. : 40-74-E0-D6-31-6C

DHCP Enabled. : Yes

Autoconfiguration Enabled : Yes

Wireless LAN adapter Local Area Connection* 10:

Media State : Media disconnected

Connection-specific DNS Suffix . :

Description : Microsoft Wi-Fi Direct Virtual Adapter #2

Physical Address. : 42-74-E0-D6-31-6B

DHCP Enabled. : Yes

Autoconfiguration Enabled : Yes

Wireless LAN adapter Wi-Fi:

Connection-specific DNS Suffix . :

Description : Intel(R) Wireless-AC 9560 160MHz

Physical Address. : 40-74-E0-D6-31-6B

DHCP Enabled. : Yes

Autoconfiguration Enabled : Yes

IPv6 Address. :

2409:4042:2211:aef1:d0c9:2cb:24a9:1206(Preferred)

Temporary IPv6 Address. :

2409:4042:2211:aef1:f17b:61f8:61f:e6b9(Preferred)

Link-local IPv6 Address : fe80::d0c9:2cb:24a9:1206%15(Preferred)

IPv4 Address. : 192.168.43.197(Preferred)

Subnet Mask : 255.255.255.0

Lease Obtained. : 25 February 2021 12.47.24 PM

Lease Expires : 25 February 2021 3.12.28 PM

Default Gateway : fe80::aa96:75ff:fe6a:ff70%15

192.168.43.1

DHCP Server : 192.168.43.1

DHCPv6 IAID : 121664736

DHCPv6 Client DUID. :

00-01-00-01-27-B9-31-76-40-74-E0-D6-31-6B

DNS Servers : 192.168.43.1

NetBIOS over Tcpip. : Enabled

Ethernet adapter Bluetooth Network Connection:

```
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix  . :
Description . . . . . : Bluetooth Device (Personal Area Network)
Physical Address. . . . . : 40-74-E0-D6-31-6F
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
```

Netstat command

The netstat command, meaning *network statistics*, is a Command Prompt command used to display *very* detailed information about how your computer is communicating with other computers or network devices.

If you are experiencing problems with network communications, then network statistics can sometimes help point you toward the root cause of the problem. That's where the aptly named NetStat command comes into play. This command has a number of different functions, but the most useful of these is to display network summary information for the device. To see this type of summary information, just type NetStat -e.

```
C:\Users\junai>netstat -e
Interface Statistics


```

	Received	Sent
Bytes	2085195300	322460532

Unicast packets	2723430	1026888
Non-unicast packets	138	7794
Discards	0	0
Errors	0	0
Unknown protocols	0	

C:\Users\junai>netstat

Active Connections

Proto	Local Address	Foreign Address	State
TCP	127.0.0.1:9012	DESKTOP-TO06OTG:63852	ESTABLISHED
TCP	127.0.0.1:62389	DESKTOP-TO06OTG:62390	ESTABLISHED
TCP	127.0.0.1:62390	DESKTOP-TO06OTG:62389	ESTABLISHED
TCP	127.0.0.1:62391	DESKTOP-TO06OTG:62392	ESTABLISHED
TCP	127.0.0.1:62392	DESKTOP-TO06OTG:62391	ESTABLISHED
TCP	127.0.0.1:62393	DESKTOP-TO06OTG:62394	ESTABLISHED
TCP	127.0.0.1:62394	DESKTOP-TO06OTG:62393	ESTABLISHED
TCP	127.0.0.1:62395	DESKTOP-TO06OTG:62396	ESTABLISHED
TCP	127.0.0.1:62396	DESKTOP-TO06OTG:62395	ESTABLISHED
TCP	127.0.0.1:63852	DESKTOP-TO06OTG:9012	ESTABLISHED
TCP	192.168.43.197:49191	bom07s28-in-f2:https	ESTABLISHED
TCP	192.168.43.197:49516	52.139.250.253:https	ESTABLISHED
TCP	192.168.43.197:54657	bom07s28-in-f2:https	ESTABLISHED

Netsat -an command

This command shows TCP as well as UDP connections. The ‘*|*’ notation is for those devices who want to mask their ip address.

```
C:\Users\junai>netstat -an
```

Active Connections

Proto	Local Address	Foreign Address	State
TCP	0.0.0.0:22	0.0.0.0:0	LISTENING
TCP	0.0.0.0:135	0.0.0.0:0	LISTENING
TCP	0.0.0.0:445	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5040	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5357	0.0.0.0:0	LISTENING
TCP	0.0.0.0:5700	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49664	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49665	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49666	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49667	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49668	0.0.0.0:0	LISTENING
TCP	0.0.0.0:49670	0.0.0.0:0	LISTENING
TCP	0.0.0.0:50080	0.0.0.0:0	LISTENING
TCP	0.0.0.0:50443	0.0.0.0:0	LISTENING
TCP	127.0.0.1:5354	0.0.0.0:0	LISTENING
TCP	127.0.0.1:8884	0.0.0.0:0	LISTENING
TCP	127.0.0.1:9012	0.0.0.0:0	LISTENING

TCP	127.0.0.1:9012	127.0.0.1:63852	ESTABLISHED
TCP	127.0.0.1:49938	0.0.0.0:0	LISTENING
TCP	127.0.0.1:62389	127.0.0.1:62390	ESTABLISHED
TCP	127.0.0.1:62390	127.0.0.1:62389	ESTABLISHED
TCP	127.0.0.1:62391	127.0.0.1:62392	ESTABLISHED
TCP	127.0.0.1:62392	127.0.0.1:62391	ESTABLISHED
TCP	127.0.0.1:62393	127.0.0.1:62394	ESTABLISHED
TCP	127.0.0.1:62394	127.0.0.1:62393	ESTABLISHED
TCP	127.0.0.1:62395	127.0.0.1:62396	ESTABLISHED
TCP	127.0.0.1:62396	127.0.0.1:62395	ESTABLISHED
TCP	127.0.0.1:63852	127.0.0.1:9012	ESTABLISHED
TCP	192.168.43.197:139	0.0.0.0:0	LISTENING
TCP	192.168.43.197:49516	52.139.250.253:443	ESTABLISHED
TCP	192.168.43.197:54885	151.101.2.217:443	TIME_WAIT
TCP	192.168.43.197:55158	151.101.154.137:443	ESTABLISHED
TCP	192.168.43.197:55269	52.109.124.116:443	TIME_WAIT
TCP	192.168.43.197:55273	104.47.29.22:443	TIME_WAIT
TCP	192.168.43.197:55276	52.109.56.48:443	TIME_WAIT
TCP	192.168.43.197:55277	52.109.56.48:443	TIME_WAIT
TCP	192.168.43.197:55278	52.109.124.127:443	TIME_WAIT
TCP	192.168.43.197:55280	52.109.20.0:443	TIME_WAIT
TCP	192.168.43.197:55282	52.114.158.91:443	TIME_WAIT
TCP	192.168.43.197:55288	23.21.140.195:443	CLOSE_WAIT
TCP	192.168.43.197:55289	40.126.47.17:443	ESTABLISHED
TCP	192.168.43.197:55290	52.114.32.112:443	ESTABLISHED
TCP	192.168.43.197:62319	23.54.24.38:443	ESTABLISHED
TCP	192.168.43.197:62438	52.114.7.161:443	ESTABLISHED
TCP	192.168.43.197:62558	52.114.15.55:443	ESTABLISHED


```
TCP 192.168.43.197:62562 52.109.124.92:443 ESTABLISHED
TCP 192.168.43.197:62565 52.111.244.0:443 ESTABLISHED
TCP 192.168.43.197:62580 52.114.40.59:443 ESTABLISHED
TCP 192.168.43.197:62599 52.114.88.22:443 ESTABLISHED
TCP 192.168.43.197:62665 52.114.132.20:443 ESTABLISHED
TCP 192.168.43.197:62666 52.114.132.20:443 ESTABLISHED
TCP 192.168.43.197:63528 34.193.242.66:443 ESTABLISHED
TCP [::]:22 [::]:0 LISTENING
TCP [::]:135 [::]:0 LISTENING
TCP [::]:445 [::]:0 LISTENING
TCP [::]:5357 [::]:0 LISTENING
TCP [::]:5700 [::]:0 LISTENING
TCP [::]:49664 [::]:0 LISTENING
TCP [::]:49665 [::]:0 LISTENING
TCP [::]:49666 [::]:0 LISTENING
TCP [::]:49667 [::]:0 LISTENING
TCP [::]:49668 [::]:0 LISTENING
TCP [::]:49670 [::]:0 LISTENING
TCP [::]:50080 [::]:0 LISTENING
TCP [::]:50443 [::]:0 LISTENING
TCP [::1]:49669 [::]:0 LISTENING
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:54660
[2404:6800:4009:821::2002]:443 TIME_WAIT
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:54708
[2a03:2880:f2ff:c0:face:b00c:0:167]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:54728
[2404:6800:4009:81d::200a]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:54730
```

[2404:6800:4009:80b::200e]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:54827
[2404:6800:4009:810::2002]:443 TIME_WAIT
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:54851
[2606:4700:8d7c:3157:eaca:30:3f76:eefd]:80 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:54852
[2606:4700:8d7c:3157:eaca:30:3f76:eefd]:80 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55130
[2404:6800:4009:807::200e]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55140
[2404:6800:4009:828::200e]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55172
[2404:6800:4009:82b::200e]:443 TIME_WAIT
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55174
[2404:6800:4003:c05::9c]:443 TIME_WAIT
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55275
[2a01:111:f400:febe::16]:443 TIME_WAIT
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55279
[2603:1046:900:2c::2]:443 TIME_WAIT
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55286
[2404:6800:4009:820::2003]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55287
[2404:6800:4009:80b::200e]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55294
[2a03:2880:f2ff:c0:face:b00c:0:167]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55295
[2a03:2880:f2ff:c0:face:b00c:0:167]:443 ESTABLISHED
TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55296

[2603:1046:900:40::2]:443 ESTABLISHED

TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:55297

[2603:1046:900:40::2]:443 ESTABLISHED

TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:62664

[2404:6800:4003:c04::bc]:5228 ESTABLISHED

TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:65340

[2603:1046:500:d::2]:443 ESTABLISHED

TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:65424

[2603:1046:900:c::2]:443 ESTABLISHED

TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:65425

[2603:1046:900:c::2]:443 ESTABLISHED

TCP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:65426

[2603:1046:900:c::2]:443 ESTABLISHED

UDP 0.0.0.0:3702 *:*

UDP 0.0.0.0:3702 *:*

UDP 0.0.0.0:3702 *:*

UDP 0.0.0.0:3702 *:*

UDP 0.0.0.0:5050 *:*

UDP 0.0.0.0:5353 *:*

UDP 0.0.0.0:5353 *:*

UDP 0.0.0.0:5353 *:*

UDP 0.0.0.0:5353 *:*

UDP 0.0.0.0:5353 *:*

UDP 0.0.0.0:5355 *:*

UDP 0.0.0.0:49664 *:*

UDP 0.0.0.0:51460 *:*

UDP 0.0.0.0:59764 *:*

UDP 0.0.0.0:63183 *:*

UDP	0.0.0.0:63404	*.*
UDP	127.0.0.1:1900	*.*
UDP	127.0.0.1:49666	*.*
UDP	127.0.0.1:59803	*.*
UDP	192.168.43.197:137	*.*
UDP	192.168.43.197:138	*.*
UDP	192.168.43.197:1900	*.*
UDP	192.168.43.197:5353	*.*
UDP	192.168.43.197:50001	*.*
UDP	192.168.43.197:50025	*.*
UDP	192.168.43.197:50050	*.*
UDP	192.168.43.197:50054	*.*
UDP	192.168.43.197:59802	*.*
UDP	:::3702	*.*
UDP	:::3702	*.*
UDP	:::3702	*.*
UDP	:::3702	*.*
UDP	:::5353	*.*
UDP	:::5353	*.*
UDP	:::5353	*.*
UDP	:::5355	*.*
UDP	:::49665	*.*
UDP	:::51460	*.*
UDP	:::59765	*.*
UDP	:::63183	*.*
UDP	:::63405	*.*
UDP	::1:1900	*.*
UDP	::1:5353	*.*

UDP [::1]:59801 *.*
UDP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:50017 *.*
UDP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:50024 *.*
UDP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:50052 *.*
UDP [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]:50058 *.*
UDP [fe80::d0c9:2cb:24a9:1206%15]:1900 *.*
UDP [fe80::d0c9:2cb:24a9:1206%15]:59800 *.*

Ping command

The ping command is one of the most often used networking utilities for detecting devices on a network and for troubleshooting network problems.

When you ping a device you send that device a short message, which it then sends back (**the echo**).

The general format is **ping hostname** or **ping IPaddress**.

Example

ping www.google.com or **ping 216.58.208.68**

```
C:\Users\junai>ping -t
```

IP address must be specified.

```
C:\Users\junai>ping www.google.com
```

Pinging www.google.com [2404:6800:4009:80c::2004] with 32 bytes of data:

Reply from 2404:6800:4009:80c::2004: time=139ms

Reply from 2404:6800:4009:80c::2004: time=148ms

Reply from 2404:6800:4009:80c::2004: time=90ms

Reply from 2404:6800:4009:80c::2004: time=126ms

Ping statistics for 2404:6800:4009:80c::2004:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 90ms, Maximum = 148ms, Average = 125ms

```
PS C:\> ping 216.58.208.68
```

Pinging 216.58.208.68 with 32 bytes of data:

Reply from 216.58.208.68: bytes=32 time=454ms TTL=109

Reply from 216.58.208.68: bytes=32 time=454ms TTL=109

Reply from 216.58.208.68: bytes=32 time=457ms TTL=109

Reply from 216.58.208.68: bytes=32 time=457ms TTL=109

Ping statistics for 216.58.208.68:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 454ms, Maximum = 457ms, Average = 455ms

Pathping command

Entering the PathPing command followed by a host name initiates what looks like a somewhat standard Tracert process. Once this process completes however, the tool takes 300 seconds (five minutes) to gather statistics, and then reports latency and packet loss statistics that are more detailed than those provided by Ping or Tracert.

```
C:\Users\junai>pathping www.google.com
```

Tracing route to www.google.com [2404:6800:4009:81f::2004]
over a maximum of 30 hops:


```
0 DESKTOP-TO06OTG [2409:4042:2211:aef1:f17b:61f8:61f:e6b9]
1 2409:4042:2211:aef1::2
2 * * *
```

Computing statistics for 25 seconds...

	Source	to Here	This Node/Link
Hop	RTT	Lost/Sent = Pct	Lost/Sent = Pct Address
0			DESKTOP-TO06OTG [2409:4042:2211:aef1:f17b:61f8:61f:e6b9] 100/ 100 =100%
1	---	100/ 100 =100%	0/ 100 = 0% 2409:4042:2211:aef1::2

Trace complete.

ARP command

Displays and modifies entries in the Address Resolution Protocol (ARP) cache, which contains one or more tables that are used to store IP addresses and their resolved Ethernet or Token Ring physical addresses. There is a separate table for each Ethernet or Token Ring network adapter installed on your computer.

Syntax

```
arp [-a [InetAddr] [-N IfaceAddr]] [-g [InetAddr] [-N IfaceAddr]] [-d InetAddr  
[IfaceAddr]] [-s InetAddr EtherAddr [IfaceAddr]]
```

```
C:\Users\junai>arp -a
```

```
Interface: 192.168.43.197 --- 0xf
```

Internet Address	Physical Address	Type
192.168.43.1	a8-96-75-6a-ff-70	dynamic
192.168.43.255	ff-ff-ff-ff-ff-ff	static
224.0.0.22	01-00-5e-00-00-16	static
224.0.0.251	01-00-5e-00-00-fb	static
224.0.0.252	01-00-5e-00-00-fc	static
239.255.255.250	01-00-5e-7f-ff-fa	static
255.255.255.255	ff-ff-ff-ff-ff-ff	static

NSLookup command

NSLookup is a great utility for diagnosing DNS name resolution problems. Just type the NSLookup command, and Windows will display the name and IP address of the device's default DNS server. From there, you can type host names in an effort to see if the DNS server is able to resolve the specified host name.

```
C:\Users\junai>nslookup
```

Default Server: UnKnown

Address: 192.168.43.1

CONCLUSION: We learnt a lot of networking commands , their functionality and their variations. We also implemented all of the major networking commands. While researching I came across a great functionality of telnet where you can watch Star Wars Episode 4 rendered completely in ASCII inside your command prompt.

EXPERIMENT 3

Aim: - To implement CRC and hamming code as error detection and correction codes.

CRC:

CRC or Cyclic Redundancy Check is a method of detecting accidental changes/errors in the communication channel.

CRC uses **Generator Polynomial** which is available on both sender and receiver side. An example generator polynomial is of the form like $x^3 + x + 1$. This generator polynomial represents key 1011.

Another example is $x^2 + 1$ that represents key 101.

n : Number of bits in data to be sent
from sender side.

k : Number of bits in the key obtained
from generator polynomial.

Sender Side (Generation of Encoded Data from Data and Generator Polynomial (or Key)):

1. The binary data is first augmented by adding $k-1$ zeros in the end of the data
2. Use *modulo-2 binary division* to divide binary data by the key and store remainder of division.
3. Append the remainder at the end of the data to form the encoded data and send the same

Receiver Side (Check if there are errors introduced in transmission)

Perform modulo-2 division again and if the remainder is 0, then there are no errors.

In this article we will focus only on finding the remainder i.e. check word and the code word

Advantages:

- This is just a simple code and can be produced and implemented in any form of applications.
- Easy to view single bit errors and multiple burst errors.
- Based on the severity and complexity of the error the error correction mechanism can be generated. This type of mechanism is commonly found in file repairing software and utilities.

- Because of its simplicity it can fit any type of operating system and all versions, including the latest.

Disadvantages:

- Though the Cyclic Redundancy Check look like an authentication mechanism, it is non trivial and easy to crack mechanism. It is not suitable for security purpose.
- Without the error correcting mechanism using CRC alone will be a useless thing.

CODE:

```
import java.util.Scanner;
class CRC2{
public static void main(String args[]){
Scanner sc = new Scanner(System.in);
//Input Data Stream
//System.out.println("\n");
System.out.println("*****  TRANSMITTER SIDE  *****");
System.out.print("Enter data stream: ");
String datastream = sc.nextLine();
System.out.print("Enter generator: ");
String generator = sc.nextLine();
int data[] = new int[datastream.length() - 1 + generator.length()];
int divisor[] = new int[generator.length()];
for(int i=0;i<datastream.length();i++)
data[i] = Integer.parseInt(datastream.charAt(i)+"");
```

```

for(int i=0;i<generator.length();i++)
divisor[i] = Integer.parseInt(generator.charAt(i)+"");

//Calculation of CRC
for(int i=0;i<datastream.length();i++){
if(data[i]==1)
for(int j=0;j<divisor.length;j++)
data[i+j] ^= divisor[j];
}

//Display CRC
System.out.print("The CRC code is: ");
for(int i=0;i<datastream.length();i++)
data[i] = Integer.parseInt(datastream.charAt(i)+"");
for(int i=0;i<data.length;i++) System.out.print(data[i]);
System.out.println();
System.out.println("***** RECEIVER SIDE *****");
//Check for input CRC code
System.out.print("Enter CRC code: ");
datastream = sc.nextLine();
System.out.print("Enter generator: ");
generator = sc.nextLine();
data = new int[datastream.length() + generator.length() - 1];
divisor = new int[generator.length()];
for(int i=0;i<datastream.length();i++)
data[i] = Integer.parseInt(datastream.charAt(i)+"");
for(int i=0;i<generator.length();i++)
divisor[i] = Integer.parseInt(generator.charAt(i)+"");

```

```

//Calculation of remainder
for(int i=0;i<datastream.length();i++){
    if(data[i]==1)
        for(int j=0;j<divisor.length;j++)
            data[i+j] ^= divisor[j];
}

//Display validity of data
boolean valid = true;
for(int i=0;i<data.length;i++)
    if(data[i]==1){
        valid = false;
        break;
    }

if(valid==true) System.out.println("\nData stream is valid");
else System.out.println("\nINVALID. CRC error occurred.");
sc.close();
}
}

```

OUTPUT:

```

*****  TRANSMITTER SIDE  *****
Enter data stream: 1010100010
Enter generator: 11111
The CRC code is: 10101000100100
*****  RECEIVER SIDE  *****
Enter CRC code: 10101000100100
Enter generator: 11111
Data stream is valid

```


Hamming Code

Hamming code is a block code that is capable of detecting up to two simultaneous bit errors and correcting single-bit errors. It was developed by R.W. Hamming for error correction.

In this coding method, the source encodes the message by inserting redundant bits within the message. These redundant bits are extra bits that are generated and inserted at specific positions in the message itself to enable error detection and correction. When the destination receives this message, it performs recalculations to detect errors and find the bit position that has error.

Encoding a message by Hamming Code

The procedure used by the sender to encode the message encompasses the following steps –

- **Step 1** – Calculation of the number of redundant bits.
- **Step 2** – Positioning the redundant bits.
- **Step 3** – Calculating the values of each redundant bit.

Once the redundant bits are embedded within the message, this is sent to the user.

Step 1 – Calculation of the number of redundant bits.

If the message contains m number of data bits, r number of redundant bits are added to it so that $m+r$ is able to indicate at least $(m+r+1)$ different states. Here, $(m+r)$ indicates location of an error in each of $(m+r)$ bit positions and one additional state indicates no error. Since, r bits can indicate 2^r states, 2^r must be at least equal to $(m+r+1)$. Thus the following equation should hold $2^r \geq m+r+1$

Step 2 – Positioning the redundant bits.

The r redundant bits placed at bit positions of powers of 2, i.e. 1, 2, 4, 8, 16 etc. They are referred in the rest of this text as r_1 (at position 1), r_2 (at position 2), r_3 (at position 4), r_4 (at position 8) and so on.

Step 3 – Calculating the values of each redundant bit.

The redundant bits are parity bits. A parity bit is an extra bit that makes the number of 1s either even or odd. The two types of parity are –

- **Even Parity** – Here the total number of bits in the message is made even.
- **Odd Parity** – Here the total number of bits in the message is made odd.

Each redundant bit, r_i , is calculated as the parity, generally even parity, based upon its bit position. It covers all bit positions whose binary representation includes a 1 in the i th position except the position of r_i . Thus

—

- r_1 is the parity bit for all data bits in positions whose binary representation includes a 1 in the least significant position excluding 1 (3, 5, 7, 9, 11 and so on)
- r_2 is the parity bit for all data bits in positions whose binary representation includes a 1 in the position 2 from right except 2 (3, 6, 7, 10, 11 and so on)
- r_3 is the parity bit for all data bits in positions whose binary representation includes a 1 in the position 3 from right except 4 (5-7, 12-15, 20-23 and so on)

Decoding a message in Hamming Code

Once the receiver gets an incoming message, it performs recalculations to detect errors and correct them. The steps for recalculation are –

- **Step 1** – Calculation of the number of redundant bits.
- **Step 2** – Positioning the redundant bits.
- **Step 3** – Parity checking.
- **Step 4** – Error detection and correction

Step 1 – Calculation of the number of redundant bits

Using the same formula as in encoding, the number of redundant bits are ascertained.

$2^r \geq m + r + 1$ where m is the number of data bits and r is the number of redundant bits.

Step 2 – Positioning the redundant bits

The r redundant bits placed at bit positions of powers of 2, i.e. 1, 2, 4, 8, 16 etc.

Step 3 – Parity checking

Parity bits are calculated based upon the data bits and the redundant bits using the same rule as during generation of c_1, c_2, c_3, c_4 etc. Thus

$c_1 = \text{parity}(1, 3, 5, 7, 9, 11 \text{ and so on})$

$c_2 = \text{parity}(2, 3, 6, 7, 10, 11 \text{ and so on})$

$c_3 = \text{parity}(4-7, 12-15, 20-23 \text{ and so on})$

Step 4 – Error detection and correction

The decimal equivalent of the parity bits binary values is calculated. If it is 0, there is no error. Otherwise, the decimal value gives the bit position which has error. For example, if $c_1c_2c_3c_4 = 1001$, it implies that the data bit at position 9, decimal equivalent of 1001, has error. The bit is flipped to get the correct message.

Advantages of Hamming code

- Hamming code method is effective on networks where the data streams are given for the single-bit errors.
- Hamming code not only provides the detection of a bit error but also helps you to indent bit containing error so that it can be corrected.
- The ease of use of hamming codes makes it best them suitable for use in computer memory and single-error correction.

Disadvantages of Hamming code

- Single-bit error detection and correction code. However, if multiple bits are founded error, then the outcome may result in another bit which should be correct to be changed. This can cause the data to be further errored.
- Hamming code algorithm can solve only single bits issues.

CODE:

```
import java.util.*;
class Hamming
{
    public static void main(String args[])
    {

        Scanner scan = new Scanner(System.in);
        System.out.println("Enter the number of bits for the Hamming data:");
```

```

int n = scan.nextInt();
int a[] = new int[n];
System.out.println("Enter the digits with a gap after every digit");
for(int i=0 ; i < n ; i++)
{
    //System.out.println("Enter bit no. " + (n-i) + ":");
    a[n-i-1] = scan.nextInt();
}
System.out.println("You entered:");
for(int i=0 ; i < n ; i++)
{
    System.out.print(a[n-i-1]);
}
System.out.println();
int b[] = generateCode(a);
System.out.println("Generated code is:");
for(int i=0 ; i < b.length ; i++) {
    System.out.print(b[b.length-i-1]);
}
System.out.println();
System.out.println("Enter position of a bit to alter to check for error detection
at the receiver end (0 for no error):");
int error = scan.nextInt();
if(error != 0)
{
    b[error-1] = (b[error-1]+1)%2;
}
System.out.println("Sent code is:");
for(int i=0 ; i < b.length ; i++) {

```

```
    System.out.print(b[b.length-i-1]);
}
System.out.println();
receive(b, b.length - a.length);
}
static int[] generateCode(int a[])
{
    int b[];
    int i=0, parity_count=0 ,j=0, k=0;
    while(i < a.length)
    {
        if(Math.pow(2,parity_count) == i+parity_count + 1)
        {
            parity_count++;
        }
        else
        {
            i++;
        }
    }
    b = new int[a.length + parity_count];
    for(i=1 ; i <= b.length ; i++)
    {
        if(Math.pow(2, j) == i)
        {
            b[i-1] = 2;
            j++;
        }
        else
```

```

    {
        b[k+j] = a[k++];
    }
}
for(i=0 ; i < parity_count ; i++)
{
    b[((int) Math.pow(2, i))-1] = getParity(b, i);
}
return b;
}
static int getParity(int b[], int power) {
int parity = 0;
for(int i=0 ; i < b.length ; i++)
{
    if(b[i] != 2)
    {
        int k = i+1;
        String s = Integer.toBinaryString(k);
        int x = ((Integer.parseInt(s))/((int) Math.pow(10, power)))%10;
        if(x == 1)
        {
            if(b[i] == 1)
            {
                parity = (parity+1)%2;
            }
        }
    }
}
}
return parity;

```



```

}
static void receive(int a[], int parity_count) {
int power;
int parity[] = new int[parity_count];
String syndrome = new String();
for(power=0 ; power < parity_count ; power++)
{
for(int i=0 ; i < a.length ; i++)
{
int k = i+1;
String s = Integer.toBinaryString(k);
int bit = ((Integer.parseInt(s))/((int) Math.pow(10, power)))%10;
if(bit == 1)
{
if(a[i] == 1)
{
parity[power] = (parity[power]+1)%2;
}
}
}
syndrome = parity[power] + syndrome;
}
int error_location = Integer.parseInt(syndrome, 2);
if(error_location != 0)
{
System.out.println("Error is at location " + error_location + ".");
a[error_location-1] = (a[error_location-1]+1)%2;
System.out.println("Corrected code is:");
for(int i=0 ; i < a.length ; i++)

```

```
{
    System.out.print(a[a.length-i-1]);
}
System.out.println();
}
else
{
    System.out.println("There is no error in the received data.");
}
System.out.println("Original data sent was:");
power = parity_count-1;
for(int i=a.length ; i > 0 ; i--)
{
    if(Math.pow(2, power) != i)
    {
        System.out.print(a[i-1]);
    }
    else
    {
        power--;
    }
}
System.out.println();
}
}
```

OUTPUT:

```
Enter the number of bits for the Hamming data:
10
Enter the digits with a gap after every digit
1 0 1 0 1 0 0 0 1 0
You entered:
1010100010
Generated code is:
10101010011001
Enter position of a bit to alter to check for error detection at the receiver end (0 for no error):
3
Sent code is:
10101010011101
Error is at location 3.
Corrected code is:
10101010011001
Original data sent was:
1010100010
```

CONCLUSION: We learnt about error detection in binary data using two methods: CRC and Hamming Code. We also learnt why it is required and the advantages and disadvantages of each method.

Experiment 4.1

Aim: - To Study & implement Dijkstra's Algorithm.

THEORY:

Dijkstra's algorithm is very similar to Prim's algorithm for minimum spanning tree. Like Prim's MST, we generate a *SPT (shortest path tree)* with given source as root. We maintain two sets, one set contains vertices included in shortest path tree, other set includes vertices not yet included in shortest path tree. At every step of the algorithm, we find a vertex which is in the other set (set of not yet included) and has a minimum distance from the source.

Below are the detailed steps used in Dijkstra's algorithm to find the shortest path from a single source vertex to all other vertices in the given graph.

Algorithm

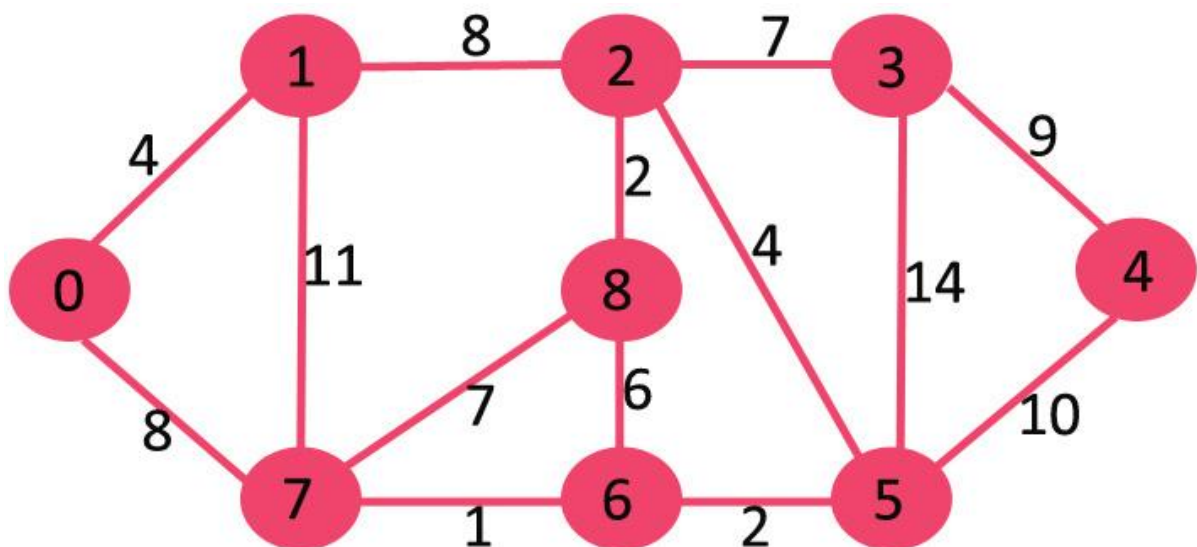
- 1) Create a set *sptSet* (shortest path tree set) that keeps track of vertices included in shortest path tree, i.e., whose minimum distance from source is calculated and finalized. Initially, this set is empty.
- 2) Assign a distance value to all vertices in the input graph. Initialize all distance values as INFINITE. Assign distance value as 0 for the source vertex so that it is picked first.
- 3) While *sptSet* doesn't include all vertices

....**a)** Pick a vertex u which is not there in $sptSet$ and has minimum distance value.

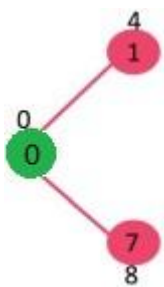
....**b)** Include u to $sptSet$.

....**c)** Update distance value of all adjacent vertices of u . To update the distance values, iterate through all adjacent vertices. For every adjacent vertex v , if sum of distance value of u (from source) and weight of edge $u-v$, is less than the distance value of v , then update the distance value of v .

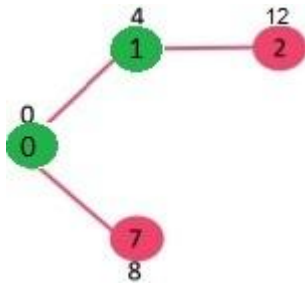
Let us understand with the following example:



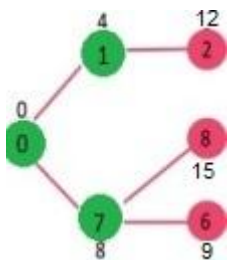
The set *sptSet* is initially empty and distances assigned to vertices are {0, INF, INF, INF, INF, INF, INF, INF} where INF indicates infinite. Now pick the vertex with minimum distance value. The vertex 0 is picked, include it in *sptSet*. So *sptSet* becomes {0}. After including 0 to *sptSet*, update distance values of its adjacent vertices. Adjacent vertices of 0 are 1 and 7. The distance values of 1 and 7 are updated as 4 and 8. Following subgraph shows vertices and their distance values, only the vertices with finite distance values are shown. The vertices included in SPT are shown in green colour.



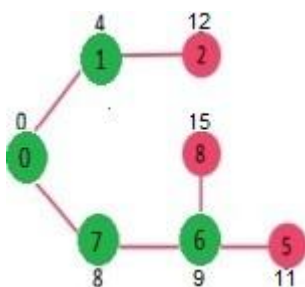
Pick the vertex with minimum distance value and not already included in SPT (not in *sptSet*). The vertex 1 is picked and added to *sptSet*. So *sptSet* now becomes {0, 1}. Update the distance values of adjacent vertices of 1. The distance value of vertex 2 becomes 12.



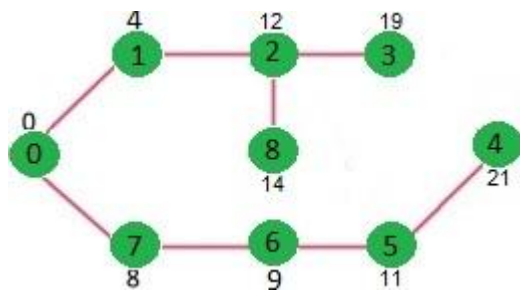
Pick the vertex with minimum distance value and not already included in SPT (not in sptSET). Vertex 7 is picked. So sptSet now becomes {0, 1, 7}. Update the distance values of adjacent vertices of 7. The distance value of vertex 6 and 8 becomes finite (15 and 9 respectively).



Pick the vertex with minimum distance value and not already included in SPT (not in sptSET). Vertex 6 is picked. So sptSet now becomes {0, 1, 7, 6}. Update the distance values of adjacent vertices of 6. The distance value of vertex 5 and 8 are updated.



We repeat the above steps until *sptSet* does include all vertices of given graph. Finally, we get the following Shortest Path Tree (SPT).



We use a boolean array *sptSet*[] to represent the set of vertices included in SPT. If a value *sptSet*[*v*] is true, then vertex *v* is included in SPT, otherwise not. Array *dist*[] is used to store shortest distance values of all vertices.

CODE:

```
import java.util.*;

public class Dijkstra2 {
    static HashSet<Integer> path = new HashSet<Integer>();

    public static int min(int dist[], boolean visit[], int V) {
        int mini = Integer.MAX_VALUE, index = -1;
        for (int v = 0; v < V; v++) {
            if (visit[v] == false && dist[v] < mini) {
                mini = dist[v];
                index = v;
            }
        }
    }
}
```



```

    }
}
return index;
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println();
    System.out.println("Dijkstra Routing Algorithm:");
    System.out.println();
    System.out.print("Enter the number of nodes:");
    int V = sc.nextInt();
    int cost[][] = new int[V][V];
    System.out.println();
    System.out.println("Enter the adjacency matrix: ");
    for (int i = 0; i < V; i++) {
        for (int j = 0; j < V; j++) {
            cost[i][j] = sc.nextInt();
        }
    }
    System.out.println();
    System.out.print("Enter the starting node:");
    int s = sc.nextInt();
    System.out.print("Enter the destination node:");
    int d = sc.nextInt();
    int dist[] = new int[V];
    boolean visit[] = new boolean[V];
    for (int i = 0; i < V; i++) {
        dist[i] = Integer.MAX_VALUE;
    }
}

```

```

        visit[i] = false;
    }
    dist[s] = 0;
    for (int i = 1; i < V - 1; i++) {
        int u = min(dist, visit, V);
        visit[u] = true;
        for (int v = 0; v < V; v++) {
            if (!visit[v] && cost[u][v] != 0 && dist[u] + cost[u][v] < dist[v])
                dist[v] = dist[u] + cost[u][v];
            path.add(u);
        }
    }
    System.out.println();
    System.out.println("Starting node:" + s);
    System.out.println("Destination node:" + d);
    System.out.println("Minimum Cost of destination from starting node:" +
dist[d]);
    System.out.print("Path: ");
    for (Integer p : path)
        System.out.print(p + "->");
    System.out.println(d);
    sc.close();
}
}

```

Dijkstra Routing Algorithm:

Enter the number of nodes:5

Enter the adjacency matrix:

```
1 2 3 4 5
2 4 6 8 10
3 6 9 12 15
4 8 12 16 20
5 10 15 20 25
```

Enter the starting node:2

Enter the destination node:4

Starting node:2

Destination node:4

Minimum Cost of destination from starting node:8

Path: 0->1->2->4

APPLICATIONS:

Dijkstra's Algorithm has several real-world use cases, some of which are as follows:

1. **Digital Mapping Services in Google Maps:** Many times we have tried to find the distance in G-Maps, from one city to another, or from your location to the nearest desired location. There encounters the Shortest Path Algorithm, as there are various routes/paths connecting them but it has to show the minimum distance, so Dijkstra's Algorithm is used to find the minimum distance between two locations along the path. Consider India as a graph and represent a city/place with a vertex and the route between two cities/places as an edge, then by using this algorithm, the shortest routes between any two cities/places or from one city/place to another city/place can be calculated.

2. **Social Networking Applications:** In many applications you might have seen the app suggests the list of friends that a particular user may know. How do you think many social media companies implement this feature efficiently, especially when the system has over a billion users. The standard Dijkstra algorithm can be applied using the shortest path between users measured through handshakes or connections among them. When the social networking graph is very small, it uses standard Dijkstra's algorithm along with some other features to find the shortest paths, and however, when the graph is becoming bigger and bigger, the standard algorithm takes a few several seconds to count and alternate advanced algorithms are used.

3. **Telephone Network:** As we know, in a telephone network, each line has a bandwidth, 'b'. The bandwidth of the transmission line is the highest frequency that that line can support. Generally, if the frequency of the signal is higher in a certain line, the signal is reduced by that line. Bandwidth represents the amount of information that can be transmitted by the line. If we imagine a city to be a graph, the vertices represent the switching stations, and the edges represent the transmission lines and the weight of the edges represents 'b'. So as you can see it can fall into the category of shortest distance problem, for which the Dijkstra is can be used.

4. **IP routing to find Open shortest Path First:** Open Shortest Path First (OSPF) is a link-state routing protocol that is used to find the best path between the source and the destination router using its own Shortest Path First. Dijkstra's algorithm is widely used in the routing protocols required by the routers to update their forwarding table. The algorithm provides the shortest cost path from the source router to other routers in the network.

5. **Flighting Agenda:** For example, If a person needs software for making an agenda of flights for customers. The agent has access to a database with all airports and flights. Besides the flight number, origin airport, and destination, the flights have departure and arrival time. Specifically, the agent wants to determine the earliest arrival time for the destination given an origin airport and start time. There this algorithm comes into use.

6. **Designate file server:** To designate a file server in a LAN(local area network), Dijkstra's algorithm can be used. Consider that an infinite amount of time is required for transmitting files from one computer to another computer. Therefore to minimize the number of "hops" from the file server to every other computer on the network the idea is to use Dijkstra's algorithm to minimize the shortest path between the networks resulting in the minimum number of hops.

7. **Robotic Path:** Nowadays, drones and robots have come into existence, some of which are manual, some automated. The drones/robots which are automated and are used to deliver the packages to a specific location or used for a task are loaded with this algorithm module so that when the source and destination is known, the robot/drone moves in the ordered direction by following the shortest path to keep delivering the package in a minimum amount of time.

CONCLUSION: We learnt about Dijkstra's algorithm which is one of the most popular algorithms used to find the shortest path between 2 nodes in a weighted graph. We implemented the algorithm and learnt about the various applications of the algorithm in real-time.

EXPERIMENT 4.2

AIM: To Study & implement Dijkstra's Algorithm.

THEORY:

Distance Vector Routing Algorithm

- **The Distance vector algorithm is iterative, asynchronous and distributed.**
 - **Distributed:** It is distributed in that each node receives information from one or more of its directly attached neighbors, performs calculation and then distributes the result back to its neighbors.
 - **Iterative:** It is iterative in that its process continues until no more information is available to be exchanged between neighbors.
 - **Asynchronous:** It does not require that all of its nodes operate in the lock step with each other.
- The Distance vector algorithm is a dynamic algorithm.
- It is mainly used in ARPANET, and RIP.
- Each router maintains a distance table known as **Vector**.

Three Keys to understand the working of Distance Vector Routing Algorithm:

- **Knowledge about the whole network:** Each router shares its knowledge through the entire network. The Router sends its collected knowledge about the network to its neighbors.
- **Routing only to neighbors:** The router sends its knowledge about the network to only those routers which have direct links. The router sends whatever it has about the network through the ports. The information is received by the router and uses the information to update its own routing table.

- **Information sharing at regular intervals:** Within 30 seconds, the router sends the information to the neighboring routers.

Distance Vector Routing Algorithm

Let $d_x(y)$ be the cost of the least-cost path from node x to node y . The least costs are related by Bellman-Ford equation,

$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

Where the \min_v is the equation taken for all x neighbors. After traveling from x to v , if we consider the least-cost path from v to y , the path cost will be $c(x,v)+d_v(y)$. The least cost from x to y is the minimum of $c(x,v)+d_v(y)$ taken over all neighbors.

With the Distance Vector Routing algorithm, the node x contains the following routing information:

- For each neighbor v , the cost $c(x,v)$ is the path cost from x to directly attached neighbor, v .
- The distance vector x , i.e., $D_x = [D_x(y) : y \text{ in } N]$, containing its cost to all destinations, y , in N .
- The distance vector of each of its neighbors, i.e., $D_v = [D_v(y) : y \text{ in } N]$ for each neighbor v of x .

Distance vector routing is an asynchronous algorithm in which node x sends the copy of its distance vector to all its neighbors. When node x receives the new distance vector from one of its neighboring vector, v , it saves the distance vector of v and uses the Bellman-Ford equation to update its own distance vector. The equation is given below:

$$d_x(y) = \min_v \{ c(x,v) + d_v(y) \} \quad \text{for each node } y \text{ in } N$$

The node x has updated its own distance vector table by using the above equation and sends its updated table to all its neighbors so that they can update their own distance vectors.

Algorithm

```
At each node x,  
  
Initialization  
  
for all destinations y in N:  
Dx(y) = c(x,y)    // If y is not a neighbor then c(x,y) = ∞  
for each neighbor w  
Dw(y) = ?    for all destination y in N.  
for each neighbor w  
send distance vector Dx = [ Dx(y) : y in N ] to w  
loop  
    wait(until I receive any distance vector from some neighbor w)  
    for each y in N:  
        Dx(y) = minv{c(x,v)+Dv(y)}  
If Dx(y) is changed for any destination y  
Send distance vector Dx = [ Dx(y) : y in N ] to all neighbors  
forever
```

CODE:

```
import java.util.*;  
  
public class DVR  
{  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        System.out.print("Enter no. of nodes:");
```

```

int n = sc.nextInt();
System.out.print("Nodes are : ");
for(int i=0;i<n;i++){
    System.out.print((char)(((int)'A')+i)+" ");
}
System.out.println();
System.out.print("Enter the node for which routing table is to be
generated: ");
int node = ((int)sc.next().charAt(0))-65;
int main_table[] = new int[n];
int main_table_parent[] = new int[n];
for(int i=0;i<n;i++){
    main_table[i]=Integer.MAX_VALUE;
}
main_table[node] = 0;
main_table_parent[node] = -1;
System.out.print("Enter the number of neighbours: ");
int m = sc.nextInt();
Neighbour neighbours[] = new Neighbour[m];
for(int i=0;i<m;i++){
    System.out.print("Enter the neighbour node: ");
    int neighbour_node=((int)sc.next().charAt(0))-65;
    System.out.print("Enter the delay between "+((char)(node+65))+
and "+((char)(neighbour_node+65))+ " : ");
    int delay=sc.nextInt();
    main_table[neighbour_node]=delay;
    main_table_parent[neighbour_node]=neighbour_node;
    neighbours[i]= new Neighbour(neighbour_node,delay,n);
    neighbours[i].distance[neighbours[i].node]=0;
    neighbours[i].distance[node]=neighbours[i].delay;
    System.out.println("Routing Table of
"+((char)(neighbour_node+65))+":");
    for(int j=0;j<neighbours[i].distance.length;j++){
System.out.println(((char)(j+65))+"\t"+neighbours[i].distance[j]);
    }
    System.out.println();

```

```

    }
    System.out.println();
    for(int i=0;i<n;i++){
        boolean flag=true;
        if(i==node){
            flag=false;
        }
        for(int j=0;j<m;j++){
            if(i==neighbours[j].node){
                flag=false;
            }
        }
        if(flag){
            int delay[] = new int[m];
            int delay_parent[] = new int[m];
            for(int j=0;j<m;j++){
                delay[j]= neighbours[j].delay+neighbours[j].distance[i];
                delay_parent[j]=neighbours[j].node;
            }
            int min_value=delay[0];
            int min_value_index=0;
            for(int j=0;j<m;j++){
                if(min_value>delay[j]){
                    min_value = delay[j];
                    min_value_index = j;
                }
            }
            main_table[i]=min_value;
            main_table_parent[i]=delay_parent[min_value_index];
        }
    }
    System.out.println("Node\tDelay\tParent");
    for(int i=0;i<n;i++){

System.out.println(((char)(i+65))+ "\t"+main_table[i]+ "\t"+((char)(main_table
_parent[i]+65))));
    }

```

```
    }  
}  
  
class Neighbour{  
    int node;  
    int delay;  
    int distance[];  
  
    Neighbour(int node,int delay,int d_length){  
        this.node = node;  
        this.delay = delay;  
        this.distance = new int[d_length];  
        for(int i=0;i<d_length;i++){  
            this.distance[i]=(int)(Math.random()*(9-1+1)+1);  
        }  
    }  
}
```

OUTPUT:

```
Enter no. of nodes:5
Nodes are : A B C D E
Enter the node for which routing table is to be generated:C
Enter the number of neighbours: 2
Enter the neighbour node: A
Enter the delay between C and A : 4
Routing Table of A:
A      0
B      7
C      4
D      2
E      1

Enter the neighbour node: B
Enter the delay between C and B : 5
Routing Table of B:
A      4
B      0
C      5
D      2
E      6

Node    Delay    Parent
A       4        A
B       5        B
C       0        @
D       6        A
E       5        A
```

CONCLUSION: We learnt about distance vector routing protocol and implemented it a java program.

Experiment 5

Aim: - To Study & implement different framing techniques.

THEORY:

Framing is function of Data Link Layer that is used to separate message from source or sender to destination or receiver or simply from all other messages to all other destinations just by adding sender address and destination address. The destination or receiver address is simply used to represent where message or packet is to go and sender or source address is simply used to help recipient to acknowledge receipt.

Frames are generally data unit of data link layer that is transmitted or transferred among various network points. It includes complete and full addressing, protocols that are essential, and information under control. Physical layers only just accept and transfer stream of bits without any regard to meaning or structure. Therefore it is up to data link layer to simply develop and recognize frame boundaries.

This can be achieved by attaching special types of bit patterns to start and end of the frame. If all of these bit patterns might accidentally occur in data, special care is needed to be taken to simply make sure that these bit patterns are not interpreted incorrectly or wrong as frame delimiters.

Framing is simply point-to-point connection among two computers or devices that consists or includes wire in which data is transferred as stream of bits. However, all of these bits should be framed into discernible blocks of information.

Methods of Framing :

There are basically three methods of framing as given below –

- 1. Character Count**
- 2. Flag Byte with Character Stuffing**
- 3. Starting and Ending Flags, with Bit Stuffing**

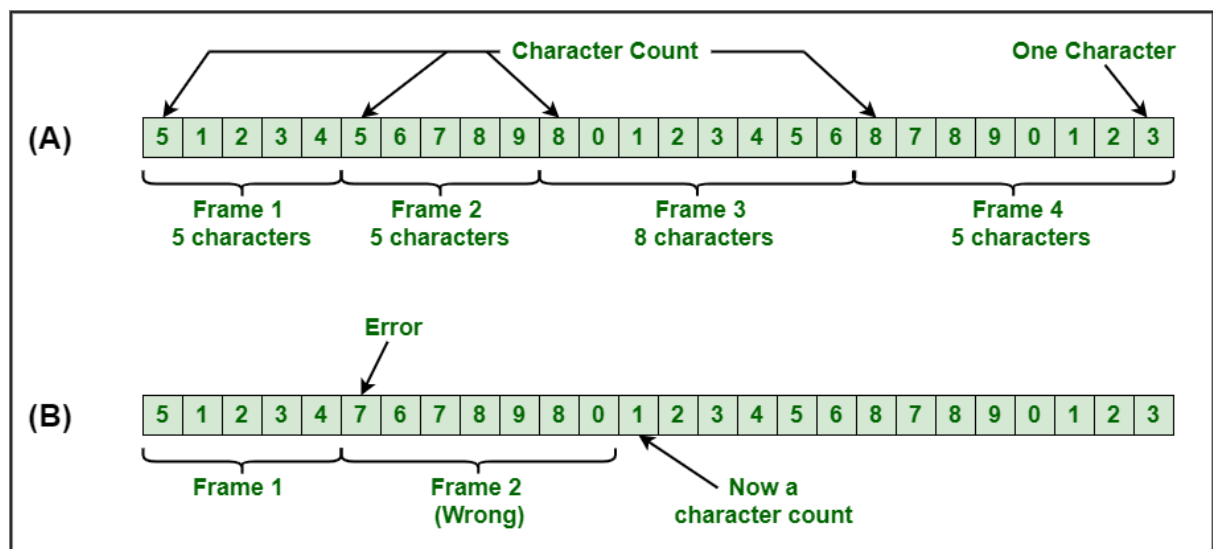
These are explained as following below.

1. Character Count :

This method is rarely used and is generally required to count total number of characters that are present in frame. This is be done by using field in header. Character count method ensures data link layer at the receiver or destination about total number of characters that follow, and about where the

frame ends.

There is disadvantage also of using this method i.e., if anyhow character count is disturbed or distorted by an error occurring during transmission, then destination or receiver might lose synchronization. The destination or receiver might also be not able to locate or identify beginning of next frame.



A Character Stream

(A) Without Errors

(B) With one Error

CODE:

```
import java.util.*;
class CharacterCount
{
    public static void main(String[] args)
```

```

{
    Scanner sc = new Scanner(System.in);
    System.out.println("\nSENDER SIDE");
    System.out.println("\nEnter the number of frames to be sent : ");
    int n = sc.nextInt();
    String frames[] = new String[n];
    for(int i=0;i<n;i++)
    {
        System.out.print("\nEnter data for frame " + i + " : ");
        frames[i] = sc.next();
    }
    int k=0; //Code Length
    int code[] = new int[100];
    for(int i=0;i<n;i++)
    {
        code[k++] = frames[i].length() + 1;
        for(int j=0;j<frames[i].length();j++)
        {
            code[k++] = Character.getNumericValue(frames[i].charAt(j));
        }
    }
    System.out.println("\nAfter Character Count Framing : ");
    for(int i=0;i<k;i++)
    {
        System.out.print(code[i]);
    }
    System.out.println("\n\nRECEIVER SIDE");
    System.out.println("\nEnter received Frame : ");
    String data = sc.next();
}

```

```

int i=0,flag=1,countOfFrames=0;
String f[] = new String[10];
while(flag==1 && i<data.length())
{
    int len = Character.getNumericValue(data.charAt(i));
    if(i+len > data.length())
    {
        System.out.println("\nReceived frame is incorrect");
        flag=0;
    }
    if(flag==1)
    {
        String s = "";
        int j;
        for(j=i+1;j<i+len;j++)
        {
            s+=data.charAt(j);
        }
        f[countOfFrames++]=s;
        i=j;
    }
}
if(flag==1)
{
    for(int a=0;a<countOfFrames;a++)
    {
        System.out.println("\nFrame " + a + " : " + f[a]);
    }
}

```

```
}  
}
```

OUTPUT:

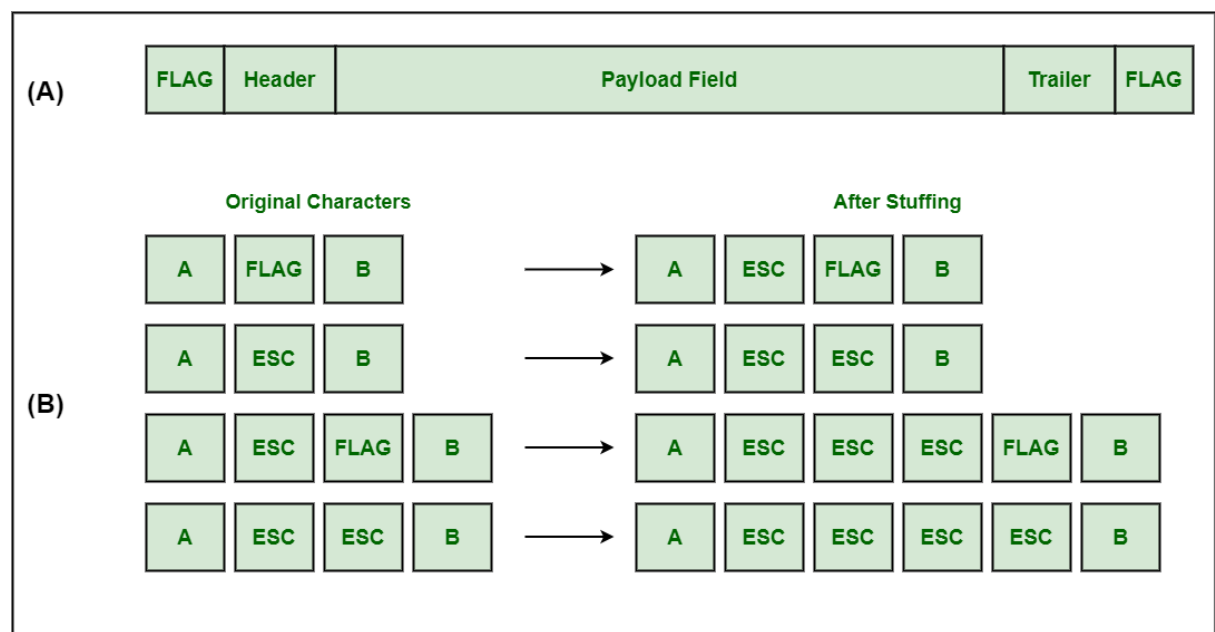
```
SENDER SIDE  
Enter the number of frames to be sent :  
3  
Enter data for frame 0 : 123  
Enter data for frame 1 : 45678  
Enter data for frame 2 : 34  
After Character Count Framing :  
4123645678334  
RECEIVER SIDE  
Enter received Frame :  
4123645678334  
Frame 0 : 123  
Frame 1 : 45678  
Frame 2 : 34  
D:\Users\june\OneDrive\ch01\file.p
```

2. Character Stuffing :

Character stuffing is also known as byte stuffing or character-oriented framing and is same as that of bit stuffing but byte stuffing actually operates on bytes whereas bit stuffing operates on bits. In byte stuffing, special byte that is basically known as ESC (Escape Character) that has predefined pattern is generally added to data section of the data stream or frame when there is message or character that has same pattern as that of flag

byte.

But receiver removes this ESC and keeps data part that causes some problems or issues. In simple words, we can say that character stuffing is addition of 1 additional byte if there is presence of ESC or flag in text.



A Character Stuffing

(A) A frame delimited by flag bytes

(B) Four examples of byte sequences before and after byte stuffing

CODE:

```
import java.util.*;

public class Framing {
    public static void main(String[] args) {
        Scanner ob = new Scanner(System.in);
        System.out.println("Enter number of frames:");
```

```
int num = ob.nextInt();
System.out.println("Choose Framing Mechanism");
System.out.println("1: Character Count\n2. Byte
Stuffing\n3. Bit Stuffing");
int choice = ob.nextInt();
String[] inputFrames = new String[num];
String output = "";
for (int i = 0; i < num; i++) {
    System.out.println("Enter Frame " + (i + 1));
    inputFrames[i] = ob.next();
}
ob.close();
switch (choice) {

case 1:
    output = charCount(inputFrames);
    break;

case 2:
    output = byteStuff(inputFrames);
    break;

case 3:
    output = bitStuff(inputFrames);
    break;

default:
    System.out.println("Please Enter A Valid
```

```
Choice");
        output = "Error";
    }

    System.out.println("Result: " + output);

    System.out.println("-----
    -----");
    System.out.println("Received String: " + output);

    ArrayList<String> receiver = new ArrayList<>();

    switch (choice) {

        case 1:
            receiver = charCountReceive(output);
            break;

        case 2:
            receiver = byteStuffReceive(output);
            break;

        case 3:
            receiver = bitStuffReceive(output);
            break;

        default:
```

```
        System.out.println("Please Enter A Valid  
Choice");  
        output = "Error";  
    }  
  
    System.out.println("Received Frames are: ");  
    for (String frame : receiver) {  
        System.out.println(frame);  
    }  
  
}  
  
public static String charCount(String input[]) {  
    String result = "";  
    for (int i = 0; i < input.length; i++) {  
        result += String.valueOf(input[i].length() + 1)  
+ input[i];  
    }  
    return result;  
}  
  
public static String byteStuff(String input[]) {  
    String result = "";  
    String flag = "$";  
    String escape = "#";  
  
    for (int i = 0; i < input.length; i++) {  
        result += flag;
```



```

        for (int j = 0; j < input[i].length(); j++) {
            if (input[i].charAt(j) == '$' ||
input[i].charAt(j) == '#')
                result += escape;
            result += input[i].charAt(j);
        }
        result += flag;
    }

    return result;
}

```

```

public static String bitStuff(String input[]) {
    String result = "";
    String flag = "01111110";
    int count = 0;
    for (int i = 0; i < input.length; i++) {
        result += flag;
        for (int j = 0; j < input[i].length(); j++) {
            if (input[i].charAt(j) == '1')
                count++;
            else
                count = 0;
            result += input[i].charAt(j);
            if (count == 5)
                result += "0";
        }
        result += flag;
    }
}

```

```
    }  
    return result;  
}  
  
public static ArrayList<String> charCountReceive(String  
input) {  
    ArrayList<String> received = new ArrayList<>();  
    int start = 0;  
    int slice =  
Character.getNumericValue(input.charAt(0));  
  
    while (start + slice <= input.length()) {  
  
        if (start + slice == input.length()) {  
            received.add(input.substring(start + 1));  
            break;  
        }  
  
        else  
            received.add(input.substring(start + 1,  
start + slice));  
        start = slice + start;  
        slice =  
Character.getNumericValue(input.charAt(start));  
        // System.out.println("Start: " + start);  
        // System.out.println("Slice: " + slice);  
  
    }  
}
```

```

        return received;
    }

    public static ArrayList<String> byteStuffReceive(String
input) {
        ArrayList<String> result = new ArrayList<>();
        int j = 1;

        String frame = "";
        while (j < input.length()) {

            // System.out.println(input.charAt(j));

            if (input.charAt(j) == '#') {
                // System.out.println("Escape");
                frame += input.charAt(j + 1);
                // System.out.println(input.charAt(j + 1) +
" Added");
                j += 2;
                if (j >= input.length())
                    break;
                continue;
            }

            if (input.charAt(j) == '$') {
                // System.out.println("End of Frame");
                result.add(frame);
            }
        }
    }

```

```

        j += 2;
        frame = "";
        continue;
    }

    frame += input.charAt(j);
    j++;

}

return result;
}

```

```

public static ArrayList<String> bitStuffReceive(String
input) {
    ArrayList<String> receiver = new ArrayList<>();
    String result = "";
    String flag = "01111110";
    int start = 8;
    int count = 0;

    // System.out.println("Length: " + input.length());

    while (start + 8 <= input.length() + 1) {
        // System.out.println(start);
        // System.out.println("Result: " + result);

        if (start + 8 >= input.length()) {
            receiver.add(result);

```

```

        break;
    } else if (input.substring(start, start +
8).equals(flag)) {
        // System.out.println("Frame Added");
        start += 16;
        // System.out.println(start);
        receiver.add(result);
        result = "";
        count = 0;
        continue;
    }

    if (input.charAt(start) == '1')
        count++;
    else
        count = 0;

    if (count == 5) {
        // System.out.println("Stuffed Bit
removed");

        result += input.charAt(start);
        start += 2;
        // System.out.println(start);
        continue;
    }

    result += input.charAt(start);
    start++;

```

```

        // System.out.println(start);

    }

    return receiver;
}
}

```

OUTPUT:

```

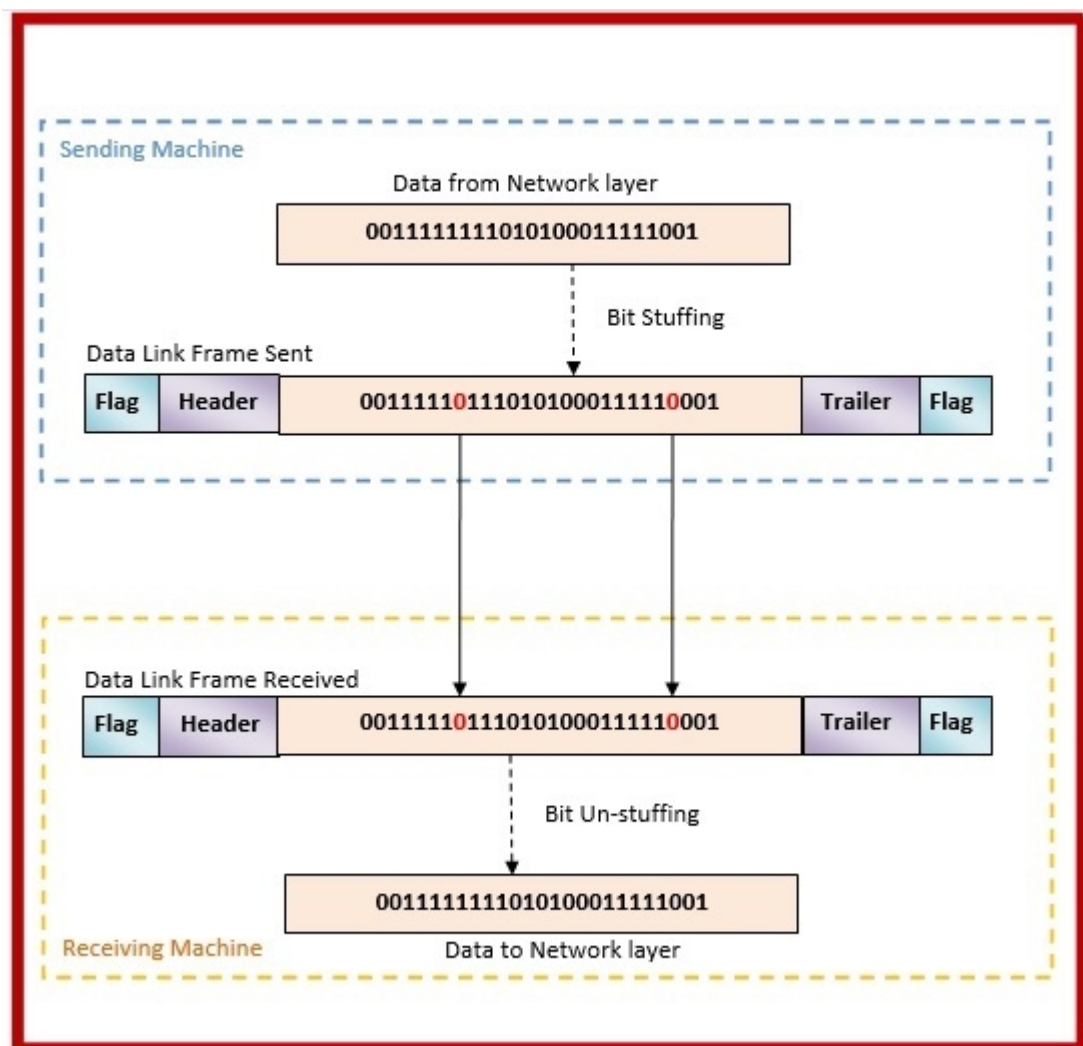
Enter number of frames:
3
Choose Framing Mechanism
1: Character Count
2: Byte Stuffing
3: Bit Stuffing
2
Enter Frame 1
123
Enter Frame 2
1#3
Enter Frame 3
1$4
Result: $123$$1##3$$1#$4$
-----
Received String: $123$$1##3$$1#$4$
Received Frames are:
123
1#3
1$4

```

3. Bit Stuffing :

Bit stuffing is also known as bit-oriented framing or bit-oriented approach. In bit stuffing, extra bits are being added by network protocol designers to data streams. It is generally insertion or addition of extra bits into transmission unit or message to be transmitted as simple way to provide and give signaling information and data to receiver and to avoid or ignore

appearance of unintended or unnecessary control sequences. It is type of protocol management simply performed to break up bit pattern that results in transmission to go out of synchronization. Bit stuffing is very essential part of transmission process in network and communication protocol. It is also required in USB.



CODE:

```
import java.util.*;
```

```

public class BitStuffing{
    public static void main(String[] args){
        Scanner sc= new Scanner(System.in);
        System.out.println("Enter flag pattern :");
        String flag = sc.nextLine();
        System.out.println("Enter data : ");
        String data = sc.nextLine();
        checkBit(data, flag);
        System.out.println("enter stuffed data from client");
        String stuffedData = sc.nextLine();
        SeperateBits(stuffedData);
    }
    static void SeperateBits(String stuffedData){
        System.out.println("Stuffed data from client: " + stuffedData);
        int count=0;
        for(int i=8;i<stuffedData.length()-8;i++){
            char ch = stuffedData.charAt(i);
            if(ch=='1'){
                count++;
                System.out.print(ch);
                if(count==5){
                    i++;
                    count=0;
                }
            }
        }
    }
}

```



```

    }
    }else{
        System.out.print(ch);
        count=0;
    }
}
System.out.println();
}

static void checkBit(String data, String flag){
    int countOne=0;
    String s ="";
    for(int i=0;i<data.length();i++){
        char ch = data.charAt(i);
        if(ch=='1'){
            countOne++;
            if(countOne<5)
                s =s+ch;
            else{
                s = s+ ch+'0';
                countOne=0;
            }
        }else{
            s = s+ch;
            countOne=0;
        }
    }
}

```

```

        }
    }
    s = flag + s + flag;

    System.out.println("Data stuffed in client: " + s);
}
}

```

OUTPUT:

```

op\DJSC\SEM 4\PRACTICALS\CN\Practical 7> java BitStuffing
Enter flag pattern :
01111110
Enter data :
10101011111
Data stuffed in client: 0111111010101011111010111110
enter stuffed data from client
0111111010101011111010111110
Stuffed data from client: 0111111010101011111010111110
10101011111

```

CONCLUSION: We learnt about the different framing techniques that are available, studied about their need and implemented them in a java program.

Experiment 6

Aim: - To implement socket communication in Java.

THEORY:

Java Socket programming is used for communication between the applications running on different JRE.

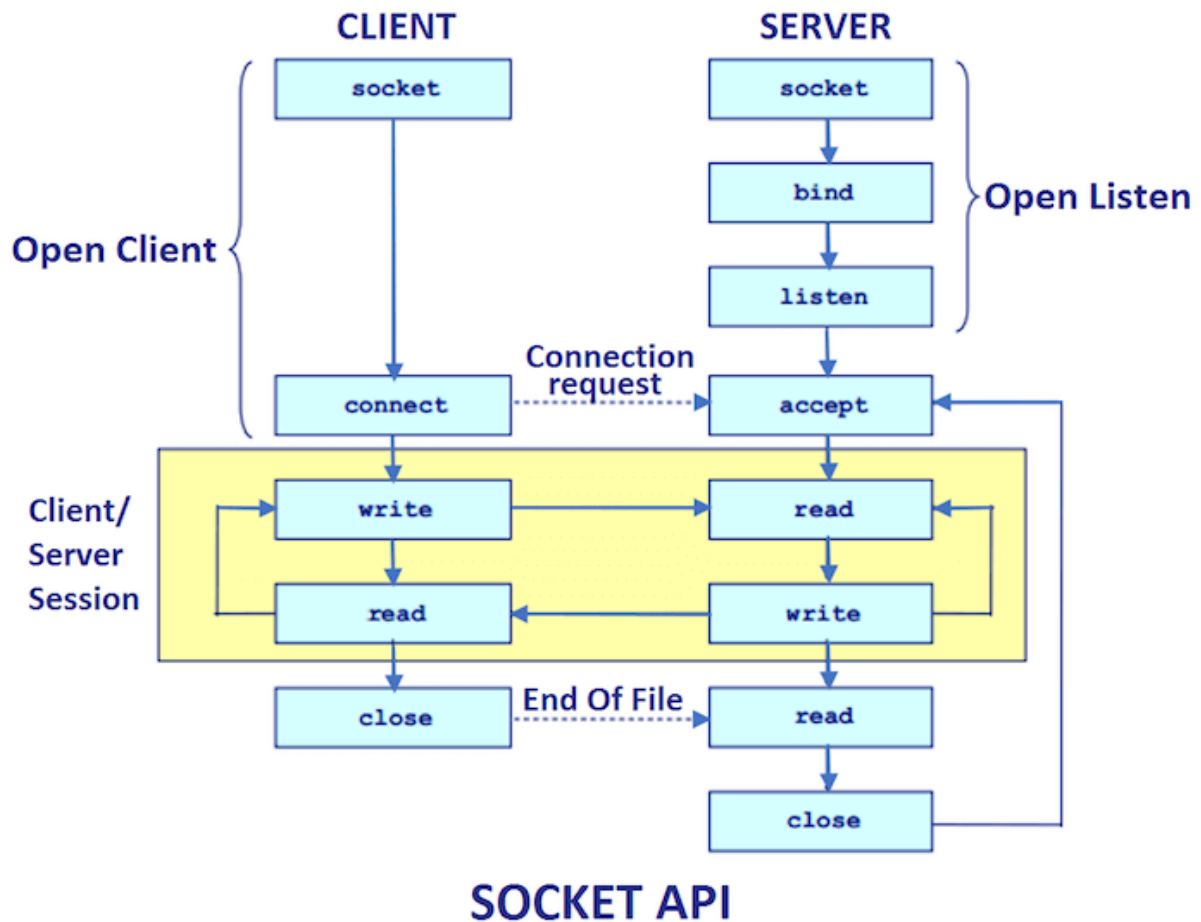
Java Socket programming can be connection-oriented or connection-less.

Socket and ServerSocket classes are used for connection-oriented socket programming and DatagramSocket and DatagramPacket classes are used for connection-less socket programming.

The client in socket programming must know two information:

1. IP Address of Server, and
2. Port number.

Here, we are going to make one-way client and server communication. In this application, client sends a message to the server, server reads the message and prints it. Here, two classes are being used: Socket and ServerSocket. The Socket class is used to communicate client and server. Through this class, we can read and write message. The ServerSocket class is used at server-side. The accept() method of ServerSocket class blocks the console until the client is connected. After the successful connection of client, it returns the instance of Socket at server-side.



Socket class

A socket is simply an endpoint for communications between the machines. The Socket class can be used to create a socket.

Important methods

Method	Description
1) public InputStream getInputStream()	returns the InputStream attached with this socket.

2) public OutputStream getOutputStream()	returns the OutputStream attached with this socket.
3) public synchronized void close()	closes this socket

ServerSocket class

The ServerSocket class can be used to create a server socket. This object is used to establish communication with the clients.

Important methods

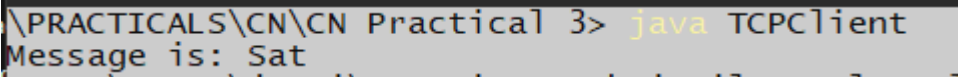
Method	Description
1) public Socket accept()	returns the socket and establish a connection between server and client.
2) public synchronized void close()	closes the server socket.

CODE:

CLIENT.java

```
// Server side code
import java.io.*;
import java.net.*;
class TCPServer{
    public static void main(String args[]) throws Exception {
        String outmsg;
        ServerSocket ss1 = new ServerSocket(80);
        while(true){
            Socket s1 = ss1.accept();
            String m[] = {"Sun", "Mon", "Tues", "Wed", "Thurs", "Fri", "Sat"};
            int i = (int)(Math.random() * m.length);
            outmsg = m[i];
            PrintStream d1 = new PrintStream(s1.getOutputStream());
            d1.println(outmsg);
        }
    }
}
```

OUTPUT:



```
\PRACTICALS\CN\CN Practical 3> java TCPClient
Message is: Sat
```

SERVER.java

```
// Server side code
import java.io.*;
import java.net.*;
class TCPServer{
    public static void main(String args[]) throws Exception {
        String outmsg;
        ServerSocket ss1 = new ServerSocket(80);
        while(true){
            Socket s1 = ss1.accept();
            String m[] = {"Sun", "Mon", "Tues", "Wed", "Thurs", "Fri", "Sat"};
            int i = (int)(Math.random() * m.length);
            outmsg = m[i];
            PrintStream d1 = new PrintStream(s1.getOutputStream());
            d1.println(outmsg);
        }
    }
}
```

OUTPUT:

```
op\DJSC\SEM 4\PRACTICALS\CN\CN Practical 3> java TCPServer
```


CONCLUSION: We learnt about socket programming and its implementation in java using java.net.* package. We then implemented a Client Server program using the same package.

Experiment 7

Aim: - Creation of Duplex links in ns2 between two nodes.

1. What is NS2

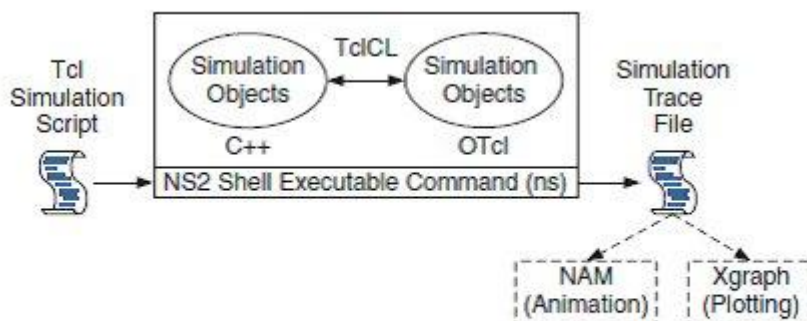
NS2 stands for Network Simulator Version 2. It is an open-source event-driven simulator designed specifically for research in computer communication networks.

2. Features of NS2

1. It is a discrete event simulator for networking research.
2. It provides substantial support to simulate bunch of protocols like TCP, FTP, UDP, https and DSR.
3. It simulates wired and wireless network.
4. It is primarily Unix based.
5. Uses TCL as its scripting language.
6. Otcl: Object oriented support
7. Tclcl: C++ and otcl linkage
8. Discrete event scheduler

3. Basic Architecture

NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a backend) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events. The C++ and the OTcl are linked together using TclCL



Basic architecture of NS.

A **duplex** communication system is a point-to-point system composed of two or more connected parties or devices that can communicate with one another in both directions. Duplex systems are employed in many communications networks, either to allow for simultaneous communication in both directions between two connected parties or to provide a reverse path for the monitoring and remote adjustment of equipment in the field. There are two types of duplex communication systems: full-duplex (FDX) and half-duplex (HDX).

In a **full-duplex** system, both parties can communicate with each other simultaneously. An example of a full-duplex device is plain old telephone service; the parties at both ends of a call can speak and be heard by the other party simultaneously. The earphone reproduces the speech of the remote party as the microphone transmits the speech of the local party. There is a two-way communication channel between them, or more strictly speaking, there are two communication channels between them.

In a **half-duplex** or **semiduplex** system, both parties can communicate with each other, but not simultaneously; the communication is one direction at a time. An example of a half-duplex device is a walkie-talkie, a two-way radio that has a push-to-talk button. When the local user wants to speak to the remote person, they push this button, which turns on the transmitter and turns off the receiver, preventing them from hearing the remote person while talking. To listen to the remote person, they release the button, which turns on the receiver and turns off the transmitter.

CODE:

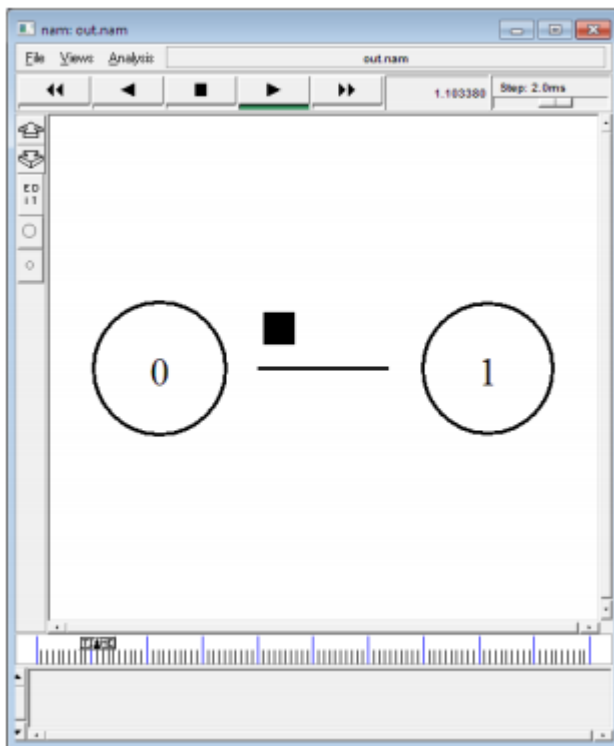
```
#===== # Simulation parameters setup
#===== set val(stop) 10.0 ;# time of simulation
end #===== # Initialization
#===== #Create a ns simulator set ns [new
Simulator] #Open the NS trace file set tracefile [open out.tr w] $ns trace-all $tracefile
#Open the NAM trace file set namfile [open out.nam w] $ns namtrace-all $namfile
#===== # Nodes Definition
#===== #Create 2 nodes set n0 [$ns node] set n1
[$ns node] #===== # Links Definition
#===== #Createlinks between nodes $ns
```

```

duplex-link $n0 $n1 100.0Mb 10ms DropTail $ns queue-limit $n0 $n1 50 #Give node
position (for NAM) $ns duplex-link-op $n0 $n1 orient right
#===== # Agents Definition
#===== #Setup a TCP connection set tcp0 [new
Agent/TCP] $ns attach-agent $n0 $tcp0 set sink1 [new Agent/TCPSink] $ns attach-agent
$n1 $sink1 $ns connect $tcp0 $sink1 $tcp0 set packetSize_ 1500
#===== # Applications Definition
#===== #Setup a FTP Application over TCP
connection set ftp0 [new Application/FTP] $ftp0 attach-agent $tcp0 $ns at 1.0 "$ftp0 start"
$ns at 2.0 "$ftp0 stop" #===== # Termination
#===== #Define a 'finish' procedure proc finish
{} { global ns tracefile namfile $ns flush-trace close $tracefile close $namfile exec nam
out.nam & exit 0 } $ns at $val(stop) "$ns nam-end-wireless $val(stop)" $ns at $val(stop)
"finish" $ns at $val(stop) "puts \"done\" ; $ns halt" $ns run

```

OUTPUT:



CONCLUSION: We learnt about duplex communication systems and implemented them in a Network Simulation (NS2).

Experiment 8

Aim: - Creation of TCP and UDP in ns2

THEORY:

What is the TCP?

The TCP stands for **Transmission Control Protocol**. If we want the communication between two computers and communication should be good and reliable. For example, we want to view a web page, then we expect that nothing should be missing on the page, or we want to download a file, then we require a complete file, i.e., nothing should be missing either it could be a text or an image. This can only be possible due to the TCP. It is one of the most widely used protocols over the TCP/IP network.

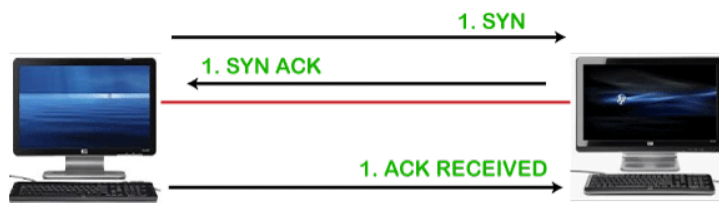
FEATURES OF TCP:

- **Data delivery**

TCP protocol ensures that the data is received correctly, no data is missing and in order. If TCP protocol is not used, then the incorrect data can be received or out of order. For example, if we try to view the web page or download a file without using TCP, then some data or images could be missing.

- **Protocol**

TCP is a connection-oriented protocol. Through the word **connection-oriented**, we understand that the computers first establish a connection and then do the communication. This is done by using a three-way handshake. In a **three-way handshake**, the first sender sends the SYN message to the receiver then the receiver sends back the SYN ACK message to confirm that the message has been received. After receiving the **SYN ACK** message, the sender sends the acknowledgment message to the receiver. In this way, the connection is established between the computers. Once the connection is established, the data will be delivered. This protocol guarantees the data delivery means that if the data is not received then the TCP will resend the data.



Connection oriented protocol

What is UDP?

The UDP stands for **User Datagram Protocol**. Its working is similar to the TCP as it is also used for sending and receiving the message. The main difference is that UDP is a connectionless protocol. Here, connectionless means that no connection establishes prior to communication. It also does not guarantee the delivery of data packets. It does not even care whether the data has been received on the receiver's end or not, so it is also known as the "fire-and-forget" protocol. It is also known as the "**fire-and-forget**" protocol as it sends the data and does not care whether the data is received or not. UDP is faster than TCP as it does not provide the assurance for the delivery of the packets.

What is the TCP?

The TCP stands for Transmission Control Protocol. If we want the communication between two computers and communication should be good and reliable. For example, we want to view a web page, then we expect that nothing should be missing on the page, or we want to download a file, then we require a complete file, i.e., nothing should be missing either it could be a text or an image. This can only be possible due to the TCP. It is one of the most widely used protocols over the TCP/IP network.

Features of TCP

- **Data delivery**

TCP protocol ensures that the data is received correctly, no data is missing and in order. If TCP protocol is not used, then the incorrect data can be received or out of order. For example, if we try to view the web page or download a file without using TCP, then some data or images could be missing.

- **Protocol**

TCP is a connection-oriented protocol. Through the word **connection-oriented**, we understand that the computers first establish a connection and then do the communication. This is done by using a three-way handshake. In a **three-way handshake**, the first sender sends the SYN message to the receiver then the receiver sends back the SYN ACK message to confirm that the message has been received. After receiving the **SYN ACK** message, the sender sends the acknowledgment message to the receiver. In this way, the connection is established between the computers. Once the connection is established, the data will be delivered. This protocol guarantees the data delivery means that if the data is not received then the TCP will resend the data.

Differences between the TCP and UDP

- **Type of protocol**

Both the protocols, i.e., TCP and UDP, are the transport layer protocol. TCP is a connection-oriented protocol, whereas UDP is a connectionless protocol. It means that TCP requires connection prior to the communication, but the UDP does not require any connection.

- **Reliability**

TCP is a reliable protocol as it provides assurance for the delivery of the data. It follows the acknowledgment mechanism. In this mechanism, the sender receives the acknowledgment from the receiver and checks whether the acknowledgment is positive or negative. If the ACK is positive means, the data has been received successfully. If ACK is negative, then TCP will resend the data. It also follows the flow and error control mechanism.

UDP is an unreliable protocol as it does not ensure the delivery of the data.

- **Flow Control**

TCP follows the flow control mechanism that ensures a large number of packets are not sent to the receiver at the same time, while UDP does not follow the flow control mechanism.

- **Ordering**

TCP uses ordering and sequencing techniques to ensure that the data packets are received in the same order in which they are sent. On the other hand, UDP does not follow any ordering and sequencing technique; i.e., data can be sent in any sequence.

- **Speed**

Since TCP establishes a connection between a sender and receiver, performs error checking, and also guarantees the delivery of data packets while UDP neither creates a connection nor it guarantees the delivery of data packets, so UDP is faster than TCP.

- **Flow of data**

In TCP, data can flow in both directions means that it provides the full-duplex service. On the other hand, UDP is mainly suitable for the unidirectional flow of data.

CODE:

```
#Create a simulator object

set ns [new Simulator]


#Define different colors for data flows (for NAM)

$ns color 1 Blue

$ns color 2 Red


#Open the NAM trace file

set nf [open out.nam w]

$ns namtrace-all $nf


#Define a 'finish' procedure

proc finish {} {

global ns nf
```

```
$ns flush-trace
```

```
#Close the NAM trace file
```

```
close $nf
```

```
#Execute NAM on the trace file
```

```
exec nam out.nam &
```

```
exit 0
```

```
}
```

```
#Create four nodes
```

```
set n0 [$ns node]
```

```
set n1 [$ns node]
```

```
set n2 [$ns node]
```

```
set n3 [$ns node]
```

```
#Create links between the nodes
```

```
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
```

```
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
```

```
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail
```

#Set Queue Size of link (n2-n3) to 10

\$ns queue-limit \$n2 \$n3 10

#Give node position (for NAM)

\$ns duplex-link-op \$n0 \$n2 orient right-down

\$ns duplex-link-op \$n1 \$n2 orient right-up

\$ns duplex-link-op \$n2 \$n3 orient right

#Monitor the queue for link (n2-n3). (for NAM)

\$ns duplex-link-op \$n2 \$n3 queuePos 0.5

#Setup a TCP connection

set tcp [new Agent/TCP]

\$tcp set class_ 2

\$ns attach-agent \$n0 \$tcp

set sink [new Agent/TCPSink]

\$ns attach-agent \$n3 \$sink

\$ns connect \$tcp \$sink

\$tcp set fid_ 1

#Setup a FTP over TCP connection

set ftp [new Application/FTP]

\$ftp attach-agent

\$tcp \$ftp set type_ FTP

#Setup a UDP connection

set udp [new Agent/UDP]

\$ns attach-agent \$n1 \$udp

set null [new Agent/Null]

\$ns attach-agent \$n3 \$null

\$ns connect \$udp \$null

\$udp set fid_ 2

#Setup a CBR over UDP connection

set cbr [new Application/Traffic/CBR]

\$cbr attach-agent \$udp

\$cbr set type_ CBR

\$cbr set packet_size_ 1000


```
$cbr set rate_ 1mb
```

```
$cbr set random_ false
```

```
#Schedule events for the CBR and FTP agents
```

```
$ns at 0.1 "$cbr start"
```

```
$ns at 1.0 "$ftp start"
```

```
$ns at 4.0 "$ftp stop"
```

```
$ns at 4.5 "$cbr stop"
```

```
#Detach tcp and sink agents (not really necessary)
```

```
$ns at 4.5 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"
```

```
#Call the finish procedure after 5 seconds of simulation time $ns at 5.0  
"finish"
```

```
#Print CBR packet size and interval
```

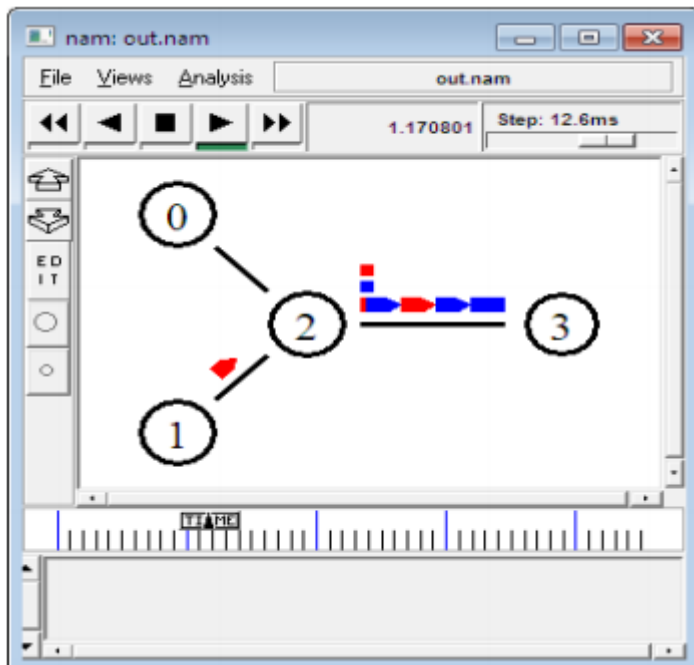
```
puts "CBR packet size = [$cbr set packet_size_]"
```

```
puts "CBR interval = [$cbr set interval_]"
```

#Run the simulation

\$ns run

OUTPUT:



CONCLUSION: We learnt about TCP and UDP networking protocols and implemented them in a Network Simulator (NS2)

Experiment 09

Aim: - Creation of Stop and Wait in ns2

THEORY:

What is Stop and Wait protocol?

Here stop and wait means, whatever the data that sender wants to send, he sends the data to the receiver. After sending the data, he stops and waits until he receives the acknowledgment from the receiver. The stop and wait protocol is a flow control protocol where flow control is one of the services of the data link layer.

It is a data-link layer protocol which is used for transmitting the data over the noiseless channels. It provides unidirectional data transmission which means that either sending or receiving of data will take place at a time. It provides flow-control mechanism but does not provide any error control mechanism.

The idea behind the usage of this frame is that when the sender sends the frame then he waits for the acknowledgment before sending the next frame.

Primitives of Stop and Wait Protocol

Sender side

Rule 1: Sender sends one data packet at a time.

Rule 2: Sender sends the next packet only when it receives the acknowledgment of the previous packet.

Therefore, the idea of stop and wait protocol in the sender's side is very simple, i.e., send one packet at a time, and do not send another packet before receiving the acknowledgment.

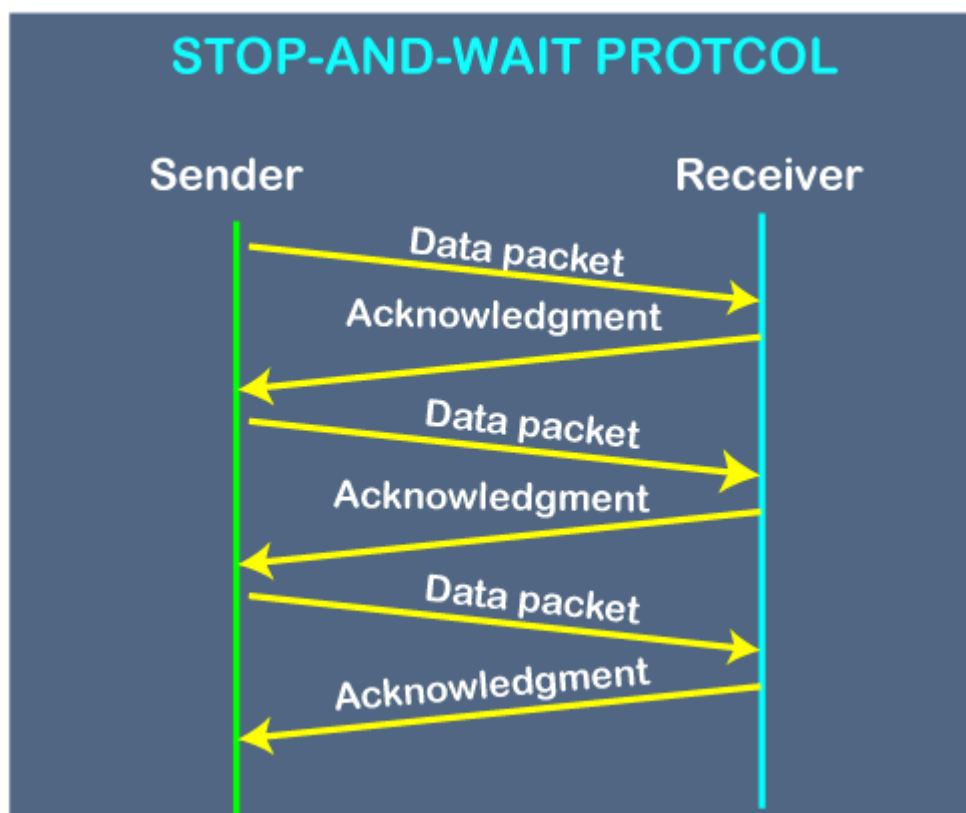
Receiver side

Rule 1: Receive and then consume the data packet.

Rule 2: When the data packet is consumed, receiver sends the acknowledgment to the sender.

Therefore, the idea of stop and wait protocol in the receiver's side is also very simple, i.e., consume the packet, and once the packet is consumed, the acknowledgment is sent. This is known as a flow control mechanism.

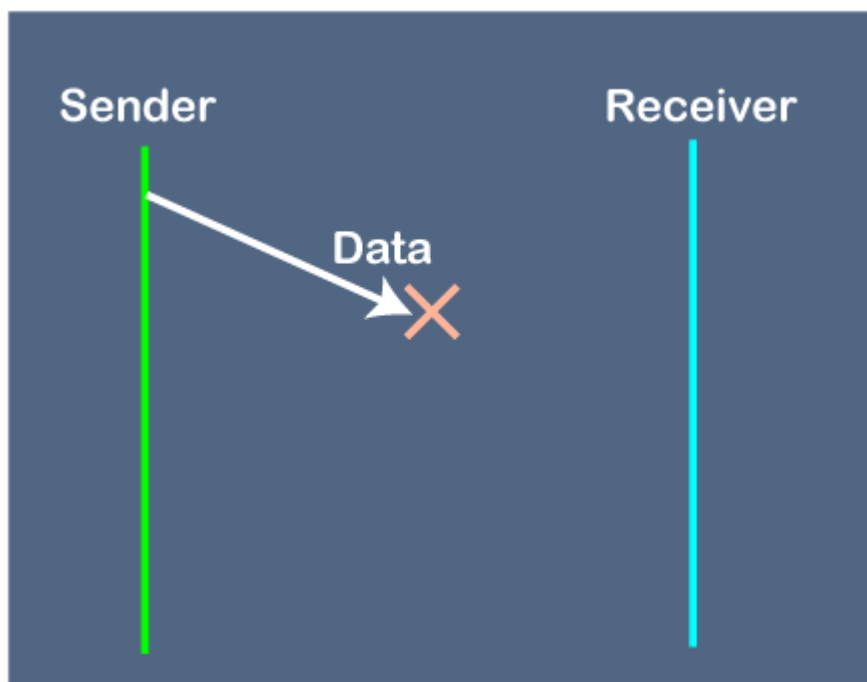
Working of Stop and Wait protocol



The above figure shows the working of the stop and wait protocol. If there is a sender and receiver, then sender sends the packet and that packet is known as a data packet. The sender will not send the second packet without receiving the acknowledgment of the first packet. The receiver sends the acknowledgment for the data packet that it has received. Once the acknowledgment is received, the sender sends the next packet. This process continues until all the packet are not sent. The main advantage of this protocol is its simplicity but it has some disadvantages also. For example, if there are 1000 data packets to be sent, then all the 1000 packets cannot be sent at a time as in Stop and Wait protocol, one packet is sent at a time.

DISADVANTAGES OF STOP AND WAIT

1. Problems occur due to lost data

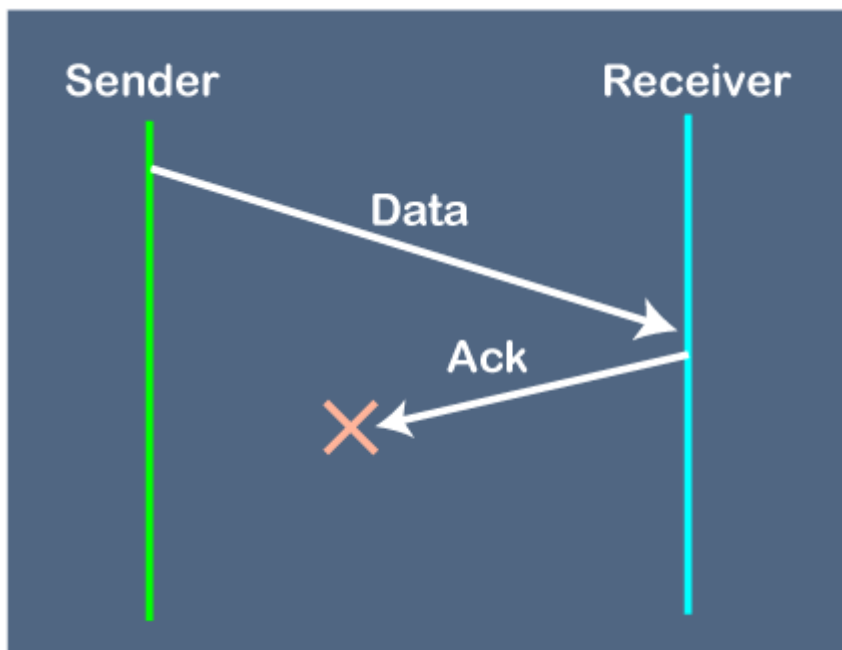


Suppose the sender sends the data and the data is lost. The receiver is waiting for the data for a long time. Since the data is not received by the receiver, so it does not send any acknowledgment. Since the sender does not receive any acknowledgment so it will not send the next packet. This problem occurs due to the lost data.

In this case, two problems occur:

- Sender waits for an infinite amount of time for an acknowledgment.
- Receiver waits for an infinite amount of time for a data.

2. Problems occur due to lost acknowledgment

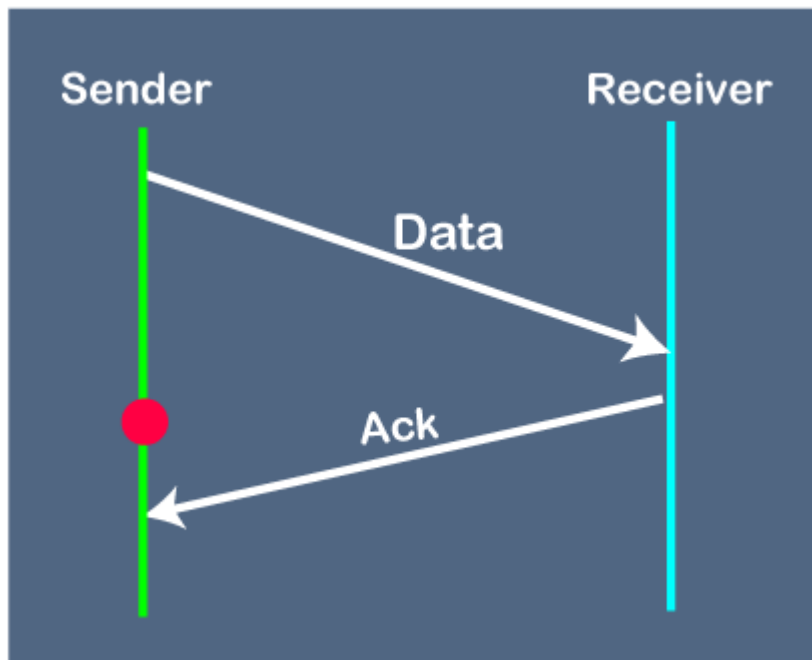


Suppose the sender sends the data and it has also been received by the receiver. On receiving the packet, the receiver sends the acknowledgment. In this case, the acknowledgment is lost in a network, so there is no chance for the sender to receive the acknowledgment. There is also no chance for the sender to send the next packet as in stop and wait protocol, the next packet cannot be sent until the acknowledgment of the previous packet is received.

In this case, one problem occurs:

- Sender waits for an infinite amount of time for an acknowledgment.

3. Problem due to the delayed data or acknowledgment



Suppose the sender sends the data and it has also been received by the receiver. The receiver then sends the acknowledgment but the acknowledgment is received after the timeout period on the sender's side. As the acknowledgment is received late, so acknowledgment can be wrongly considered as the acknowledgment of some other data packet.

CODE:

```
# stop and wait protocol in normal situation
# features : labeling, annotation, nam-graph, and window size
monitoring
set ns [new Simulator]

set n0 [$ns node]
set n1 [$ns node]
```

```
$ns at 0.0 "$n0 label Sender"
```

```
$ns at 0.0 "$n1 label Receiver"
```

```
set nf [open A1-stop-n-wait.nam w]
```

```
$ns namtrace-all $nf
```

```
set f [open A1-stop-n-wait.tr w]
```

```
$ns trace-all $f
```

```
$ns duplex-link $n0 $n1 0.2Mb 200ms DropTail
```

```
$ns duplex-link-op $n0 $n1 orient right
```

```
$ns queue-limit $n0 $n1 10
```

```
Agent/TCP set nam_tracevar_ true
```

```
set tcp [new Agent/TCP]
```

```
$tcp set window_ 1
```

```
$tcp set maxcwnd_ 1
```

```
$ns attach-agent $n0 $tcp
```

```
set sink [new Agent/TCPSink]
```

```
$ns attach-agent $n1 $sink
```

```
$ns connect $tcp $sink
```

```
set ftp [new Application/FTP]
```

```
$ftp attach-agent $tcp
```


\$ns add-agent-trace \$tcp tcp

\$ns monitor-agent-trace \$tcp

\$tcp tracevar cwnd_

\$ns at 0.1 "\$ftp start"

\$ns at 3.0 "\$ns detach-agent \$n0 \$tcp ; \$ns detach-agent \$n1 \$sink"

\$ns at 3.5 "finish"

\$ns at 0.0 "\$ns trace-annotate \"Stop and Wait with normal operation\""

\$ns at 0.05 "\$ns trace-annotate \"FTP starts at 0.1\""

\$ns at 0.11 "\$ns trace-annotate \"Send Packet_0\""

\$ns at 0.35 "\$ns trace-annotate \"Receive Ack_0\""

\$ns at 0.56 "\$ns trace-annotate \"Send Packet_1\""

\$ns at 0.79 "\$ns trace-annotate \"Receive Ack_1\""

\$ns at 0.99 "\$ns trace-annotate \"Send Packet_2\""

\$ns at 1.23 "\$ns trace-annotate \"Receive Ack_2 \""

\$ns at 1.43 "\$ns trace-annotate \"Send Packet_3\""

\$ns at 1.67 "\$ns trace-annotate \"Receive Ack_3\""

\$ns at 1.88 "\$ns trace-annotate \"Send Packet_4\""

\$ns at 2.11 "\$ns trace-annotate \"Receive Ack_4\""

\$ns at 2.32 "\$ns trace-annotate \"Send Packet_5\""

\$ns at 2.55 "\$ns trace-annotate \"Receive Ack_5 \""

\$ns at 2.75 "\$ns trace-annotate \"Send Packet_6\""

```
$ns at 2.99 "$ns trace-annotate \"Receive Ack_6\""
```

```
$ns at 3.1 "$ns trace-annotate \"FTP stops\""
```

```
proc finish {} {
```

```
  global ns nf
```

```
  $ns flush-trace
```

```
  close $nf
```

```
  puts "filtering..."
```

```
  exec tclsh ../ns-allinone-2.1b5/nam-1.0a7/bin/namfilter.tcl
```

```
  A1-stop-n-wait.nam
```

```
  puts "running nam..."
```

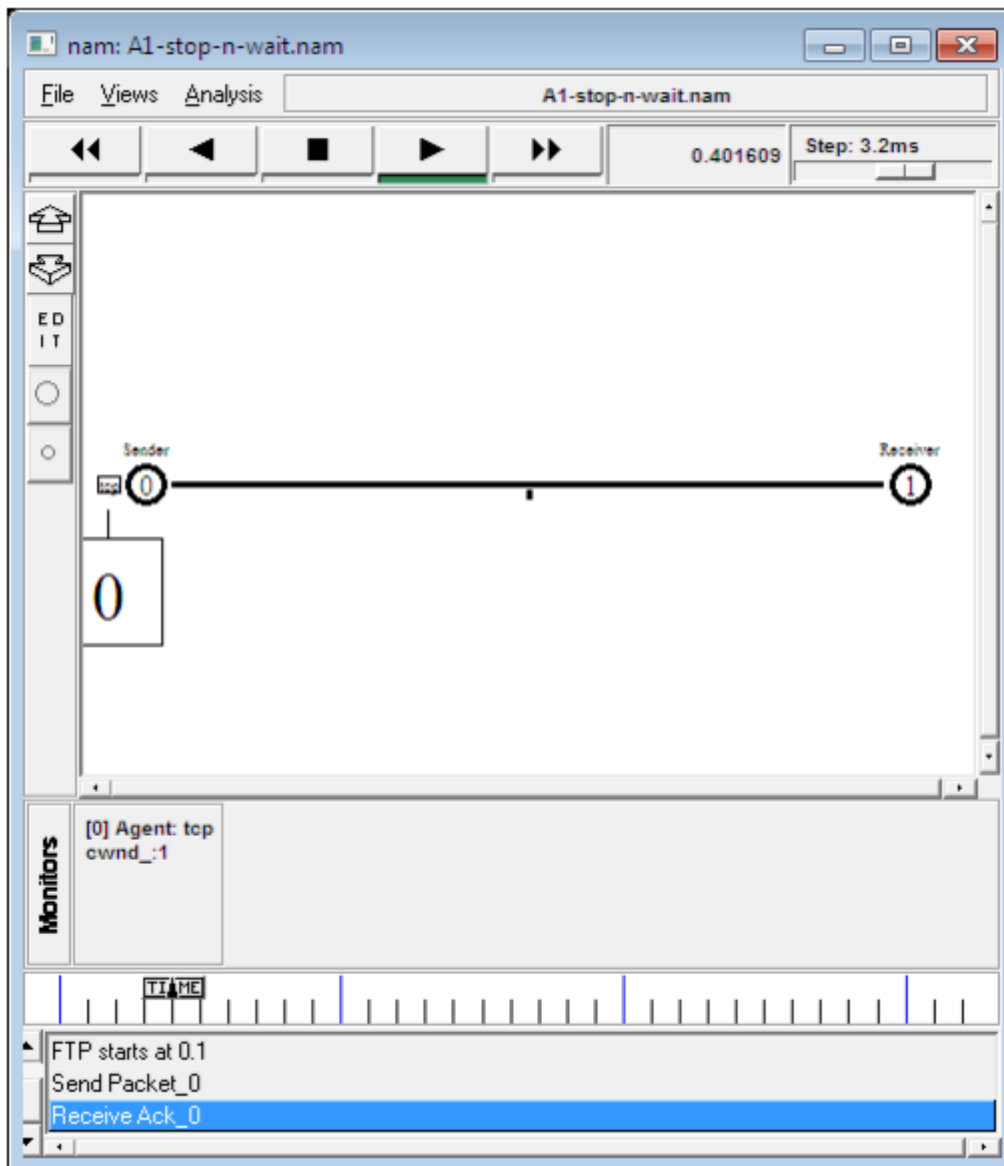
```
  exec nam A1-stop-n-wait.nam &
```

```
  exit 0
```

```
}
```

```
$ns run
```

OUTPUT:



CONCLUSION: We learnt about stop and wait and implemented that using NS2.

Experiment 10

Aim: - Create different networking topologies in packet tracer.

THEORY:

What is Network Topology?

Network topology refers to how various nodes, devices, and connections on your network are physically or logically arranged in relation to each other. Think of your network as a city, and the topology as the road map. Just as there are many ways to arrange and maintain a city—such as making sure the avenues and boulevards can facilitate passage between the parts of town getting the most traffic—there are several ways to arrange a network. Each has advantages and disadvantages and depending on the needs of your company, certain arrangements can give you a greater degree of connectivity and security.

There are two approaches to network topology: physical and logical. Physical network topology, as the name suggests, refers to the physical connections and interconnections between nodes and the network—the wires, cables, and so forth. Logical network topology is a little more abstract and strategic, referring to the conceptual understanding of how and why the network is arranged the way it is, and how data moves through it.

Why Is Network Topology Important?

The layout of your network is important for several reasons. Above all, it plays an essential role in how and how well your network functions. Choosing the right topology for your company's operational model can increase performance while making it easier to locate faults, troubleshoot errors, and more effectively allocate resources across the network to ensure optimal network health. A streamlined and properly managed network topology can increase energy and data efficiency, which can in turn help to reduce operational and maintenance costs.

The design and structure of a network are usually shown and manipulated in a software-created network topology diagram. These diagrams are essential for a few reasons, but especially for how they can provide visual representations of both physical and logical layouts, allowing administrators to see the connections between devices when troubleshooting.

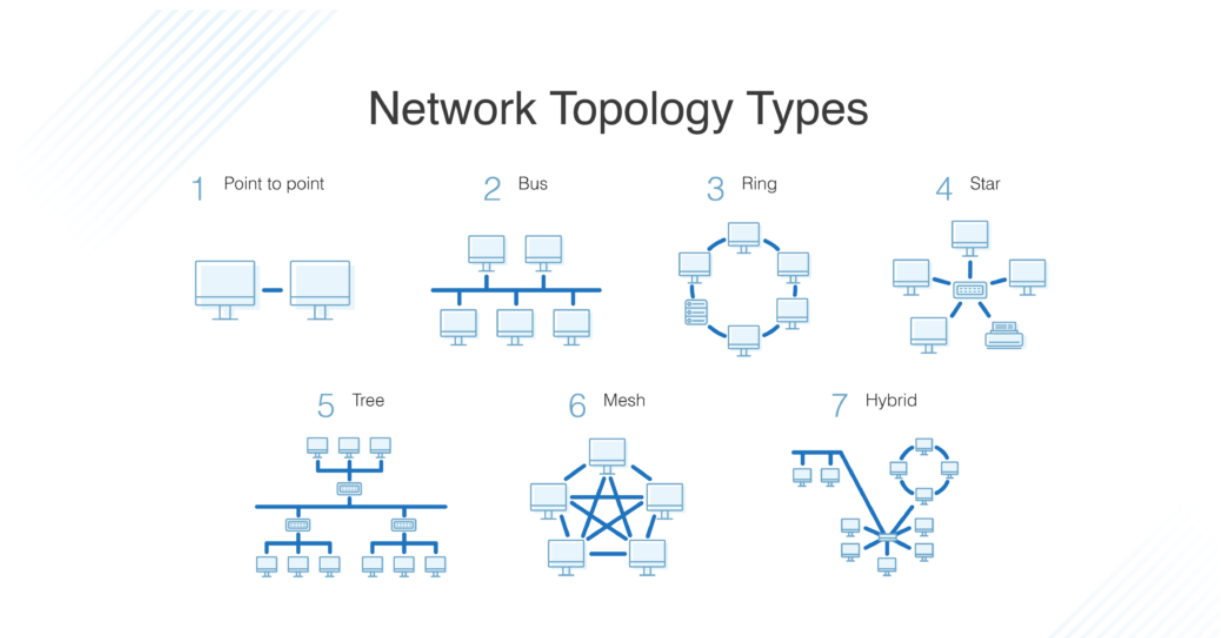
The way a network is arranged can make or break network functionality, connectivity, and protection from downtime. The question of, “What is network topology?” can be answered with an explanation of the two categories in the network topology.

1. Physical – The physical network topology refers to the actual connections (wires, cables, etc.) of how the network is arranged. Setup, maintenance, and provisioning tasks require insight into the physical network.
2. Logical – The logical network topology is a higher-level idea of how the network is set up, including which nodes connect to each other and in which ways, as well as how data is transmitted through the network. Logical network topology includes any virtual and cloud resources.

Effective network management and monitoring require a strong grasp of both the physical and logical topology of a network to ensure your network is efficient and healthy.

What’s the Most Common Type of Network Topology?

Building a local area network (LAN) topology can be make-or-break for your business, as you want to set up a resilient, secure, and easy-to-maintain topology. There are several different types of network topology and all are suitable for different purposes, depending on the overall network size and your objectives.

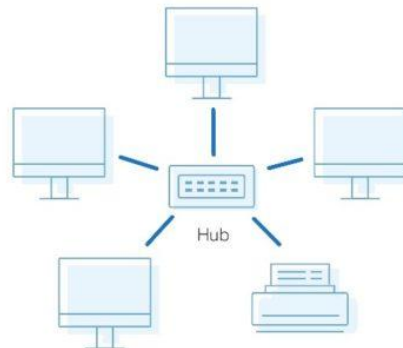


As with most things, there's no "right" or one-size-fits-all option. With this in mind, I'll walk you through the most common network topology definitions to give you a feel for the advantages and disadvantages of each.

STAR TOPOLOGY

A star topology, the most common network topology, is laid out so every node in the network is directly connected to one central hub via coaxial, twisted-pair, or fiber-optic cable. Acting as a server, this central node manages data transmission—as information sent from any node on the network has to pass through the central one to reach its destination—and functions as a repeater, which helps prevent data loss.

Star Topology



Advantages of Star Topology

Star topologies are common since they allow you to conveniently manage your entire network from a single location. Because each of the nodes is independently connected to the central hub, should one go down, the rest of the network will continue functioning unaffected, making the star topology a stable and secure network layout.

Additionally, devices can be added, removed, and modified without taking the entire network offline.

On the physical side of things, the structure of the star topology uses relatively little cabling to fully connect the network, which allows for both straightforward setup and management over time as the network expands or contracts. The simplicity of the network design makes life easier for administrators, too, because it's easy to identify where errors or performance issues are occurring.

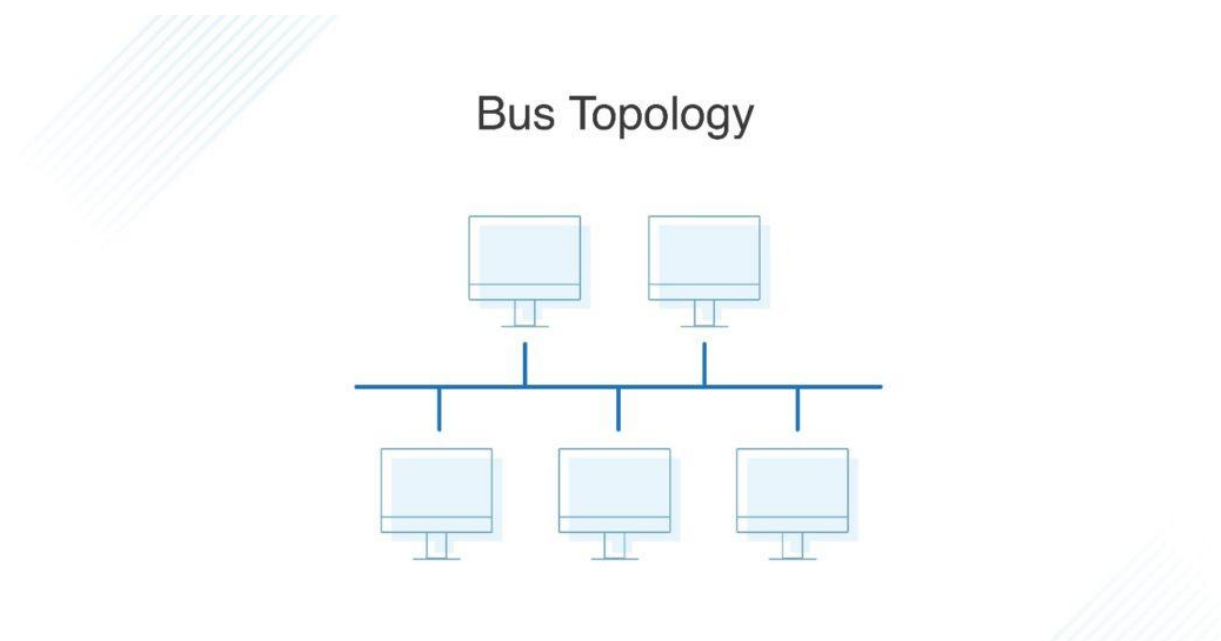
Disadvantages of Star Topology

On the flipside, if the central hub goes down, the rest of the network can't function. But if the central hub is properly managed and kept in good health, administrators shouldn't have too many issues.

The overall bandwidth and performance of the network are also limited by the central node's configurations and technical specifications, making star topologies expensive to set up and operate.

What Is Bus Topology?

A bus topology orients all the devices on a network along a single cable running in a single direction from one end of the network to the other—which is why it's sometimes called a “line topology” or “backbone topology.” Data flow on the network also follows the route of the cable, moving in one direction.



Advantages of Bus Topology

Bus topologies are a good, cost-effective choice for smaller networks because the layout is simple, allowing all devices to be connected via a single coaxial or RJ45 cable. If needed, more nodes can be easily added to the network by joining additional cables.

Disadvantages of Bus Topology

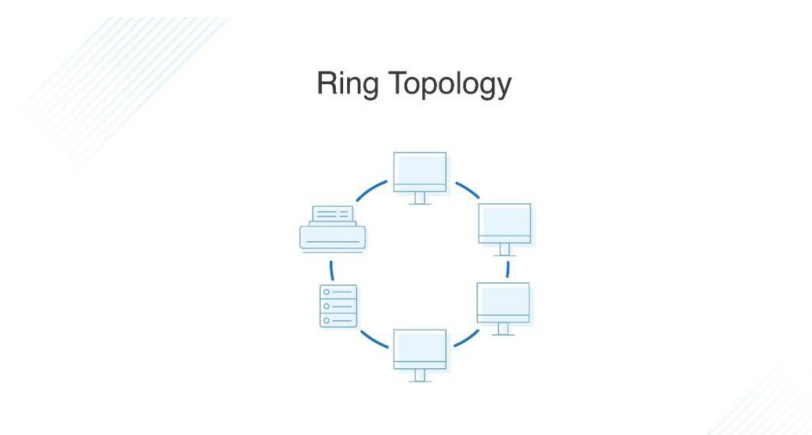
However, because bus topologies use a single cable to transmit data, they're somewhat vulnerable. If the cable experiences a failure, the whole network goes down, which can be time-consuming and expensive to restore, which can be less of an issue with smaller networks.

Bus topologies are best suited for small networks because there's only so much bandwidth, and every additional node will slow transmission speeds.

Furthermore, data is "half-duplex," which means it can't be sent in two opposite directions at the same time, so this layout is not the ideal choice for networks with huge amounts of traffic.

What Is Ring Topology? Single vs. Dual

Ring topology is where nodes are arranged in a circle (or ring). The data can travel through the ring network in either one direction or both directions, with each device having exactly two neighbors.



Pros of Ring Topology

Since each device is only connected to the ones on either side, when data is transmitted, the packets also travel along the circle, moving through each of the intermediate nodes until they arrive at their destination. If a large network is arranged in a ring topology, repeaters can be used to ensure packets arrive correctly and without data loss.

Only one station on the network is permitted to send data at a time, which greatly reduces the risk of packet collisions, making ring topologies efficient at transmitting data without errors.

By and large, ring topologies are cost-effective and inexpensive to install, and the intricate point-to-point connectivity of the nodes makes it relatively easy to identify issues or misconfigurations on the network.

Cons of Ring Topology

Even though it's popular, a ring topology is still vulnerable to failure without proper network management. Since the flow of data transmission moves unidirectionally between nodes along each ring, if one node goes down, it can take the entire network with it. That's why it's imperative for each of the nodes to be monitored and kept in good health. Nevertheless, even if you're vigilant and attentive to node performance, your network can still be taken down by a transmission line failure.

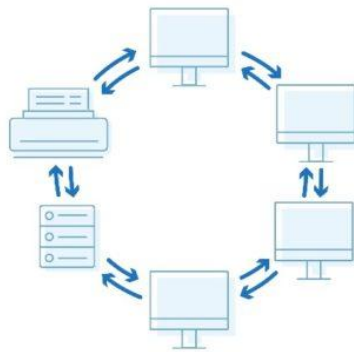
The question of scalability should also be taken into consideration. In a ring topology, all the devices on the network share bandwidth, so the addition of more devices can contribute to overall communication delays. Network administrators need to be mindful of the devices added to the topology to avoid overburdening the network's resources and capacity.

Additionally, the entire network must be taken offline to reconfigure, add, or remove nodes. And while that's not the end of the world, scheduling downtime for the network can be inconvenient and costly.

What Is Dual-Ring Topology?

A network with ring topology is half-duplex, meaning data can only move in one direction at a time. Ring topologies can be made full-duplex by adding a second connection between network nodes, creating a dual ring topology.

Dual Ring Topology

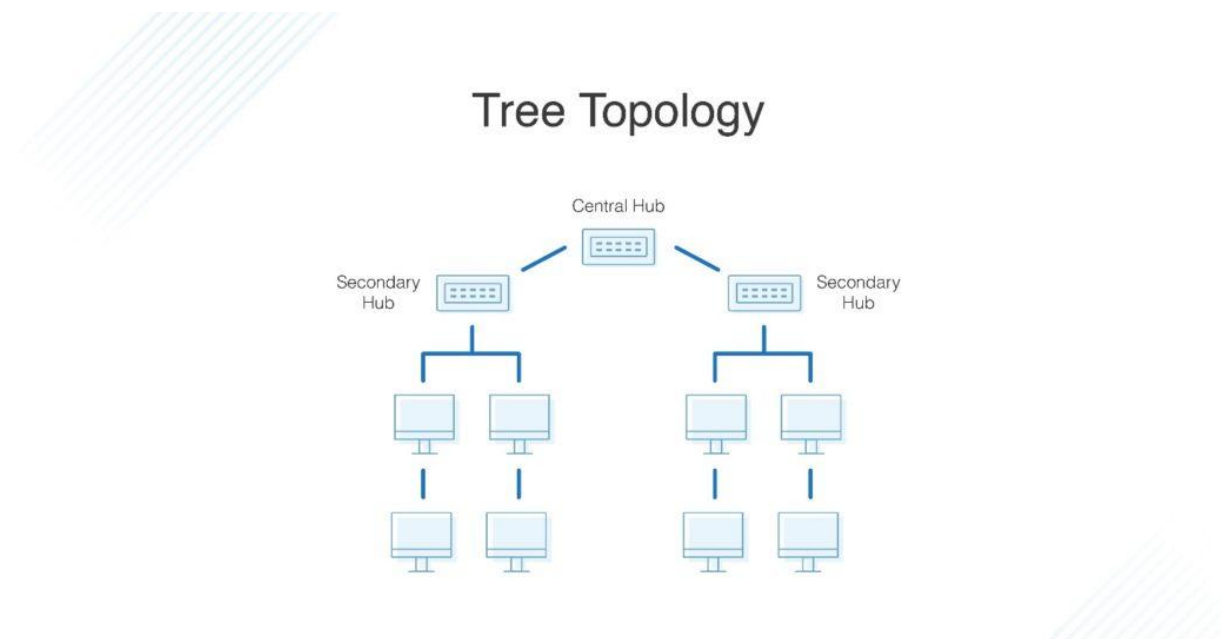


Advantages of Dual-Ring Topology

The primary advantage of dual ring topology is its efficiency: because each node has two connections on either side, information can be sent both clockwise and counterclockwise along the network. The secondary ring included in a dual-ring topology setup can act as a redundant layer and backup, which helps solve for many of the disadvantages of traditional ring topology. Dual ring topologies offer a little extra security, too: if one ring fails within a node, the other ring is still able to send data.

What Is Tree Topology?

The tree topology structure gets its name from how the central node functions as a sort of trunk for the network, with nodes extending outward in a branch-like fashion. However, where each node in a star topology is directly connected to the central hub, a tree topology has a parent-child hierarchy to how the nodes are connected. Those connected to the central hub are connected linearly to other nodes, so two connected nodes only share one mutual connection. Because the tree topology structure is both extremely flexible and scalable, it's often used for wide area networks to support many spread-out devices.



Pros of Tree Topology

Combining elements of the star and bus topologies allows for the easy addition of nodes and network expansion. Troubleshooting errors on the network is also a straightforward process, as each of the branches can be individually assessed for performance issues.

Cons of Tree Topology

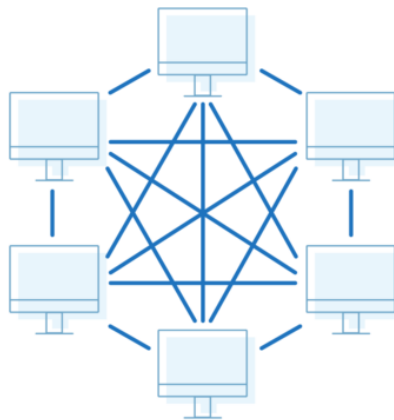
As with the star topology, the entire network depends on the health of the root node in a tree topology structure. Should the central hub fail, the various node branches will become disconnected, though connectivity within—but not between—branch systems will remain.

Because of the hierarchical complexity and linear structure of the network layout, adding more nodes to a tree topology can quickly make proper management an unwieldy, not to mention costly, experience. Tree topologies are expensive because of the sheer amount of cabling required to connect each device to the next within the hierarchical layout.

What Is Mesh Topology?

A mesh topology is an intricate and elaborate structure of point-to-point connections where the nodes are interconnected. Mesh networks can be full or partial mesh. Partial mesh topologies are mostly interconnected, with a few nodes with only two or three connections, while full-mesh topologies are—surprise!—fully interconnected.

Mesh Topology



The web-like structure of mesh topologies offers two different methods of data transmission: routing and flooding. When data is routed, the nodes use logic to determine the shortest distance from the source to destination, and when data is flooded, the information is sent to all nodes within the network without the need for routing logic.

Advantages of Mesh Topology

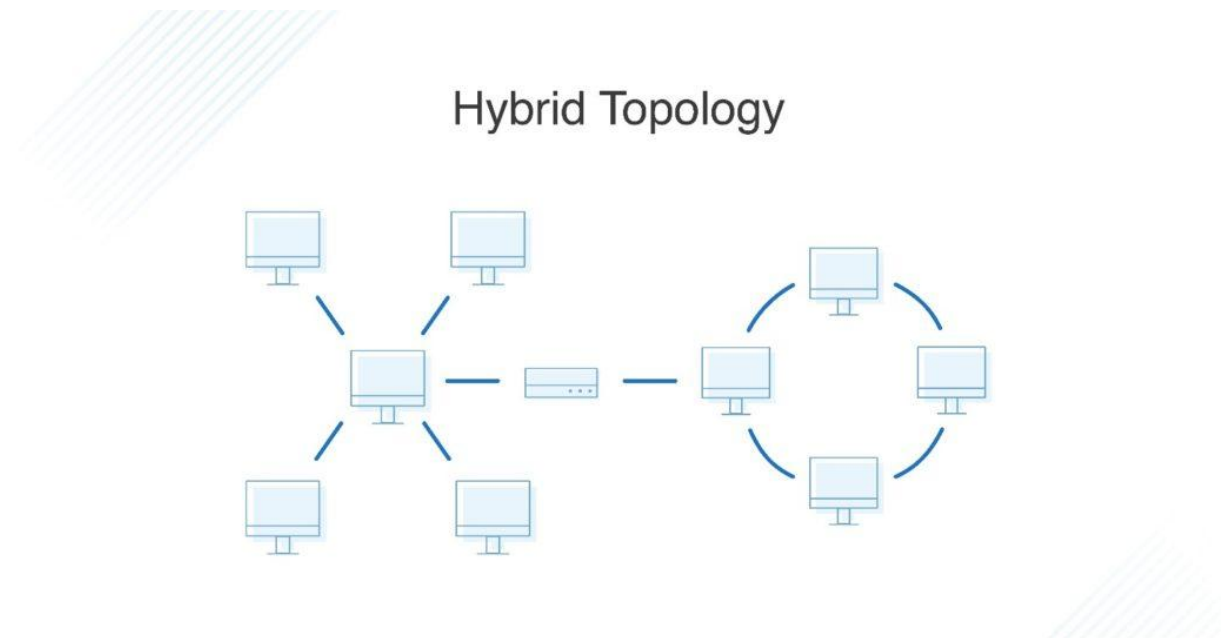
Mesh topologies are reliable and stable, and the complex degree of interconnectivity between nodes makes the network resistant to failure. For instance, no single device going down can bring the network offline.

Disadvantages of Mesh Topology

Mesh topologies are incredibly labor-intensive. Each interconnection between nodes requires a cable and configuration once deployed, so it can also be time-consuming to set up. As with other topology structures, the cost of cabling adds up fast, and to say mesh networks require a lot of cabling is an understatement.

What Is Hybrid Topology?

Hybrid topologies combine two or more different topology structures—the tree topology is a good example, integrating the bus and star layouts. Hybrid structures are most commonly found in larger companies where individual departments have personalized network topologies adapted to suit their needs and network usage.



Advantages of Hybrid Topology

The main advantage of hybrid structures is the degree of flexibility they provide, as there are few limitations on the network structure itself that a hybrid setup can't accommodate.

Disadvantages of Hybrid Topology

However, each type of network topology comes with its own disadvantages, and as a network grows in complexity, so too does the experience and know-how required on the part of the admins to keep everything functioning optimally. There's also the monetary cost to consider when creating a hybrid network topology.

Which Topology Is Best for Your Network?

No network topology is perfect, or even inherently better than the others, so determining the right structure for your business will depend on the needs and size of your network. Here are the key elements to consider:

- Length of cable needed
- Cable type
- Cost
- Scalability

Cable Length

Generally, the more cable involved in network topology, the more work it'll require to set up. The bus and star topologies are on the simpler side of things, both being fairly lightweight, while mesh networks are much more cable- and labor-intensive.

Cable Type

The second point to consider is the type of cable you'll install. Coaxial and twisted-pair cables both use insulated copper or copper-based wiring, while fiber-optic cables are made from thin and pliable plastic or glass tubes. Twisted-pair cables are cost-effective but have less bandwidth than coaxial cables. Fiber-optic cables are high performing and can transmit data far faster than twisted-pair or coaxial cables, but they also tend to be far more expensive to install, because they require additional components like optical receivers. So, as with your choice of network topology, the wiring you select depends on the needs of your network, including which applications you'll be running, the transmission distance, and desired performance.

Cost

As I've mentioned, the installation cost is important to account for, as the more complex network topologies will require more time and funding to set up. This can be compounded if you're combining different elements, such as connecting a more complex network structure via more expensive cables (though using fiber-optic cables in a mesh network is overdoing it, if you ask me, because of how interconnected the topology is). Determining the right topology for your needs, then, is a matter of striking the right balance between installation and operating costs and the level of performance you require from the network.

Scalability

The last element to consider is scalability. If you anticipate your company and network expanding—or if you'd like it to be able to—it'll save you time and hassle down the line to use an easily modifiable network topology. Star topologies are so common because they allow you to add, remove, and alter nodes with minimal disruption to the rest of the network. Ring networks, on the other hand, have to be taken entirely offline for any changes to be made to any of the nodes.

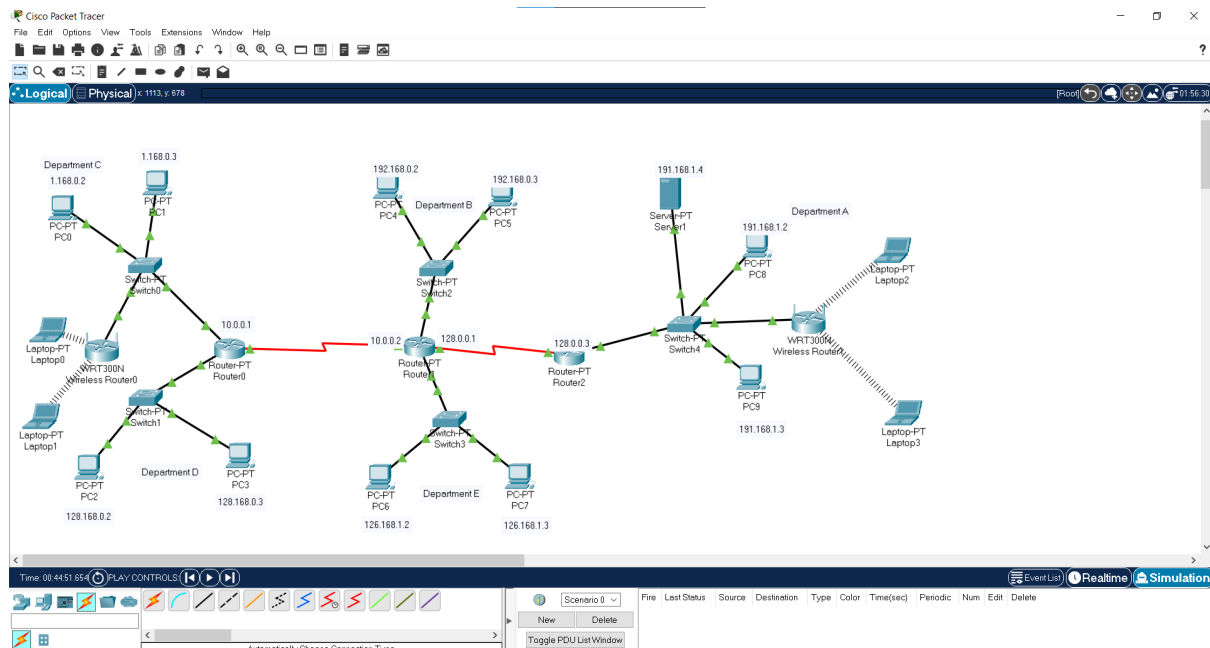
How to Map Network Topology

When you're starting to design a network, topology diagrams come in handy. They allow you to see how the information will move across the network, which, in turn, allows you to predict potential choke points. Visual representation makes it easier to create a streamlined and efficient network design, while also acting as a good reference point if you find yourself needing to troubleshoot errors.

A topology diagram is also essential for having a comprehensive understanding of your network's functionality. In addition to assisting with the troubleshooting process, the bird's-eye view provided by a topology diagram can help you visually identify the pieces of the infrastructure your network is lacking, or which nodes need monitoring, upgrading, or replacing.

The good news is you don't have to do it manually: you can easily create a map of your network topology with tools.

OUTPUT:



Packet Tracer PC Command Line 1.0

```
C:\>ping 192.168.0.101
```

Pinging 192.168.0.101 with 32 bytes of data:

Reply from 192.168.0.101: bytes=32 time=18ms TTL=128

Reply from 192.168.0.101: bytes=32 time=28ms TTL=128

Reply from 192.168.0.101: bytes=32 time=28ms TTL=128

Reply from 192.168.0.101: bytes=32 time=35ms TTL=128

Ping statistics for 192.168.0.101:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 18ms, Maximum = 35ms, Average = 27ms

C:\>

CONCLUSION: We learnt about different network topologies and implemented them virtually using Packet Tracer. To check if our simulation is proper, we then did a ping test that ensured that all the devices were properly connected.

Experiment 11

Aim: - To Implement RIP in packet tracer.

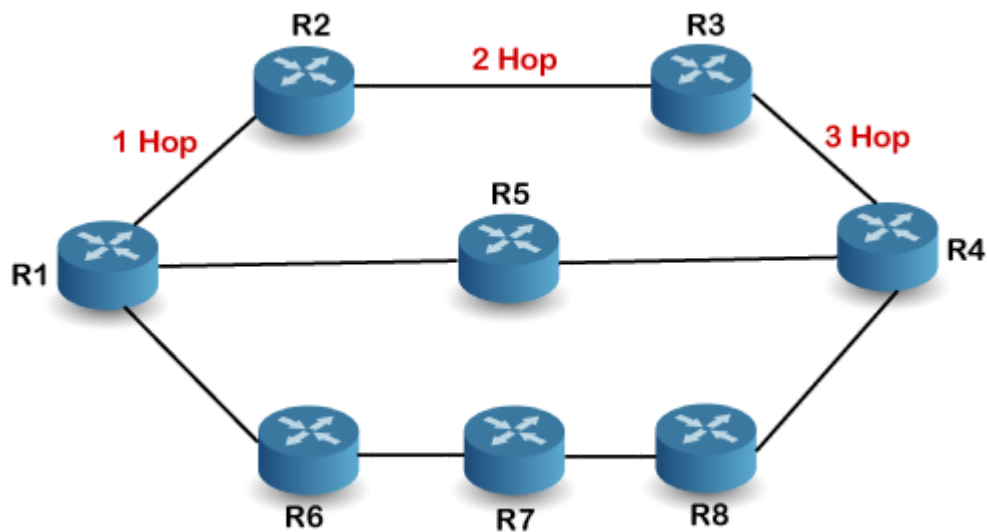
THEORY:

RIP stands for Routing Information Protocol. RIP is an intra-domain routing protocol used within an autonomous system. Here, intra-domain means routing the packets in a defined domain, for example, web browsing within an institutional area. To understand the RIP protocol, our main focus is to know the structure of the packet, how many fields it contains, and how these fields determine the routing table.

- RIP is based on the distance vector-based strategy, so we consider the entire structure as a graph where nodes are the routers, and the links are the networks.
- In a routing table, the first column is the destination, or we can say that it is a network address.
- The cost metric is the number of hops to reach the destination. The number of hops available in a network would be the cost. The hop count is the number of networks required to reach the destination.
- In RIP, infinity is defined as 16, which means that the RIP is useful for smaller networks or small autonomous systems. The maximum number of hops that RIP can contain is 15 hops, i.e., it should not have more than 15 hops as 16 is infinity.
- The next column contains the address of the router to which the packet is to be sent to reach the destination.

How is hop count determined?

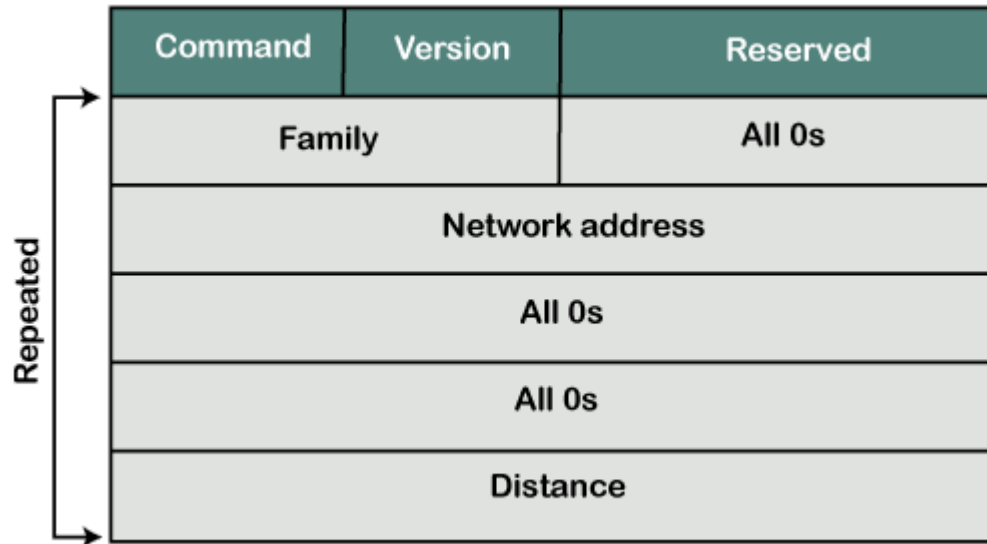
When the router sends the packet to the network segment, then it is counted as a single hop.



In the above figure, when the router 1 forwards the packet to the router 2 then it will count as 1 hop count. Similarly, when the router 2 forwards the packet to the router 3 then it will count as 2 hop count, and when the router 3 forwards the packet to router 4, it will count as 3 hop count. In the same way, RIP can support maximum upto 15 hops, which means that the 16 routers can be configured in a RIP.

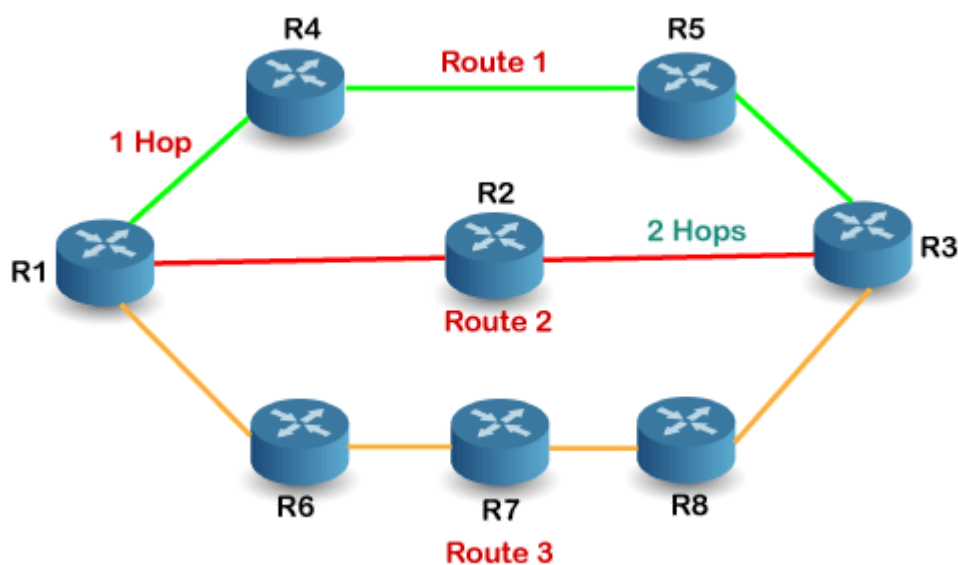
RIP Message Format

Now, we look at the structure of the RIP message format. The message format is used to share information among different routers. The RIP contains the following fields in a message:



- **Command:** It is an 8-bit field that is used for request or reply. The value of the request is 1, and the value of the reply is 2.
- **Version:** Here, version means that which version of the protocol we are using. Suppose we are using the protocol of version 1, then we put the 1 in this field.
- **Reserved:** This is a reserved field, so it is filled with zeroes.
- **Family:** It is a 16-bit field. As we are using the TCP/IP family, so we put 2 value in this field.
- **Network Address:** It is defined as 14 bytes field. If we use the IPv4 version, then we use 4 bytes, and the other 10 bytes are all zeroes.
- **Distance:** The distance field specifies the hop count, i.e., the number of hops used to reach the destination.

How does the RIP work?



If there are 8 routers in a network where Router 1 wants to send the data to Router 3. If the network is configured with RIP, it will choose the route which has the least number of hops. There are three routes in the above network, i.e., Route 1, Route 2, and Route 3. The Route 2 contains the least number of hops, i.e., 2 where Route 1 contains 3 hops, and Route 3 contains 4 hops, so RIP will choose Route 2.

Features of RIP :

1. Updates of the network are exchanged periodically.
2. Updates (routing information) are always broadcast.
3. Full routing tables are sent in updates.
4. Routers always trust on routing information received from neighbor routers. This is also known as *Routing on rumours*.

RIP versions :

There are three versions of routing information protocol – **RIP Version1**, **RIP Version2** and **RIPng**.

RIP v1	RIP v2	RIPng
Sends update as broadcast	Sends update as multicast	Sends update as multicast
Broadcast at 255.255.255.255	Multicast at 224.0.0.9	Multicast at FF02::9 (RIPng can only run on IPv6 networks)
Doesn't support authentication of update messages	Supports authentication of RIPv2 update messages	—
Classful routing protocol	Classless protocol, supports classful	Classless updates are sent

RIP v1 is known as *Classful* Routing Protocol because it doesn't send information of subnet mask in its routing update.

RIP v2 is known as *Classless* Routing Protocol because it sends information of subnet mask in its routing update.

>> Use debug command to get the details :

```
# debug ip rip
```

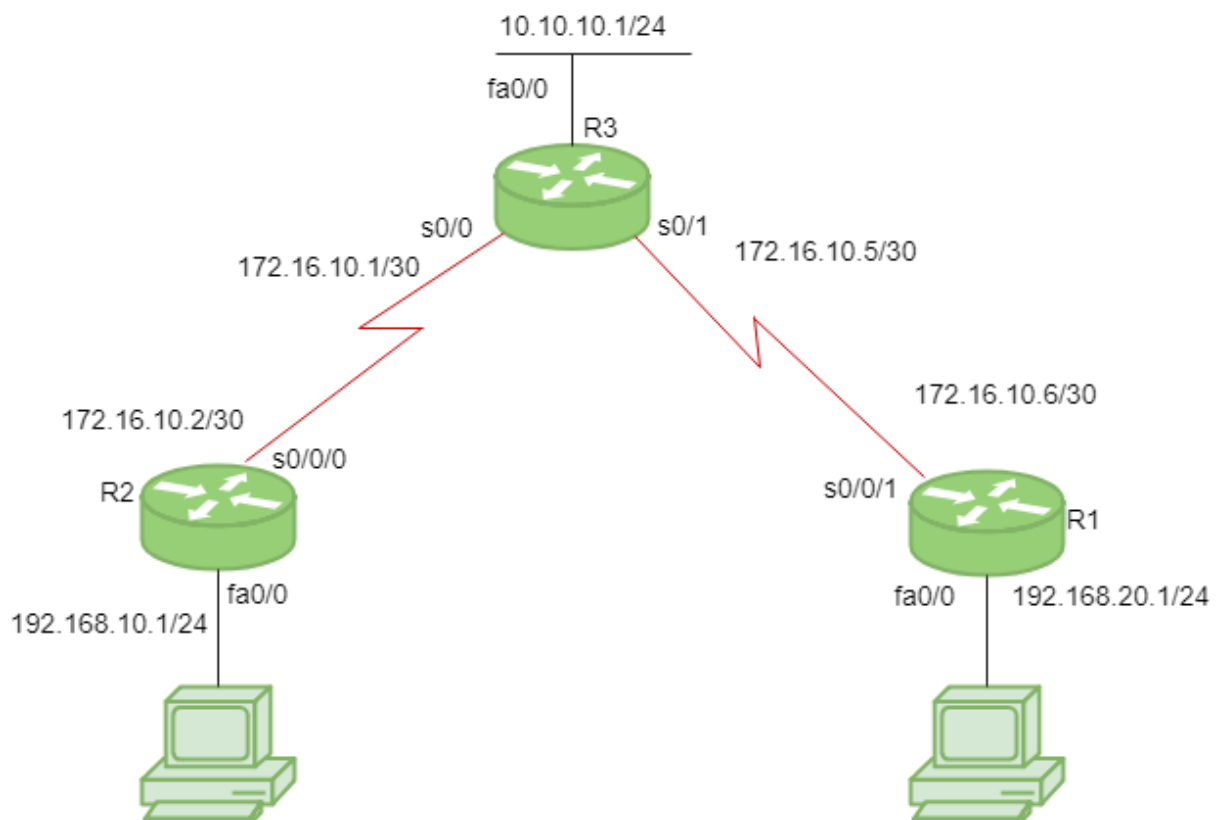
>> Use this command to show all routes configured in router, say for router R1 :

```
R1# show ip route
```

>> Use this command to show all protocols configured in router, say for router R1 :

```
R1# show ip protocols
```

Configuration :



Consider the above given topology which has 3-routers R1, R2, R3.

R1 has IP address 172.16.10.6/30 on s0/0/1, 192.168.20.1/24 on fa0/0.

R2 has IP address 172.16.10.2/30 on s0/0/0, 192.168.10.1/24 on fa0/0.

R3 has IP address 172.16.10.5/30 on s0/1, 172.16.10.1/30 on s0/0,
10.10.10.1/24 on fa0/0.

Configure RIP for R1 :

```
R1(config)# router rip
R1(config-router)# network 192.168.20.0
R1(config-router)# network 172.16.10.4
R1(config-router)# version 2
R1(config-router)# no auto-summary
```

Note : no auto-summary command disables the auto-summarisation. If we don't select no auto-summary, then subnet mask will be considered as classful in Version 1.

Configure RIP for R2 :

```
R2(config)# router rip
R2(config-router)# network 192.168.10.0
R2(config-router)# network 172.16.10.0
R2(config-router)# version 2
R2(config-router)# no auto-summary
```

Similarly, Configure RIP for R3 :

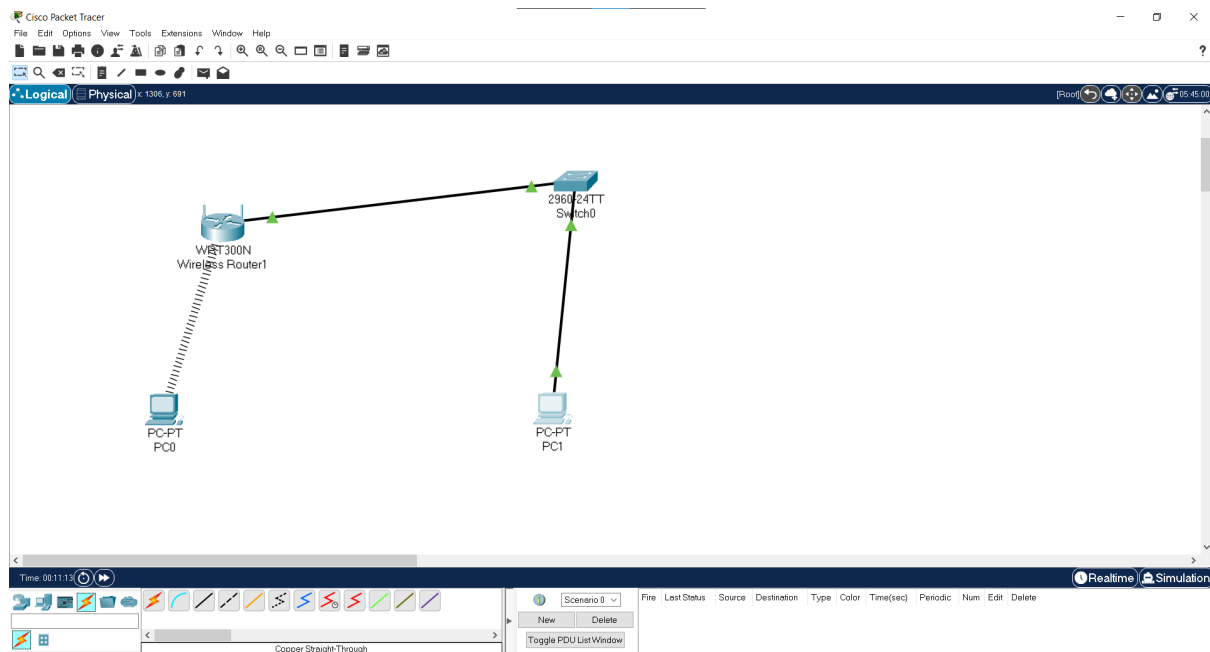
```
R3(config)# router rip
R3(config-router)# network 10.10.10.0
R3(config-router)# network 172.16.10.4
R3(config-router)# network 172.16.10.0
R3(config-router)# version 2
R3(config-router)# no auto-summary
```

RIP timers :

- **Update timer :** The default timing for routing information being exchanged by the routers operating RIP is 30 seconds. Using Update timer, the routers exchange their routing table periodically.
- **Invalid timer:** If no update comes until 180 seconds, then the destination router consider it as invalid. In this scenario, the destination router mark hop count as 16 for that router.
- **Hold down timer :** This is the time for which the router waits for neighbour router to respond. If the router isn't able to respond within a given time then it is declared dead. It is 180 seconds by default.
- **Flush time :** It is the time after which the entry of the route will be flushed if it doesn't respond within the flush time. It is 60 seconds by default. This timer starts after the route has been declared invalid and after 60 seconds i.e time will be $180 + 60 = 240$ seconds.

Note that all these times are adjustable. Use this command to change the timers :

```
R1(config-router)# timers basic  
R1(config-router)# timers basic 20 80 80 90
```



Packet Tracer PC Command Line 1.0

```
C:\>ping 192.168.0.101
```

Pinging 192.168.0.101 with 32 bytes of data:

Reply from 192.168.0.101: bytes=32 time=18ms TTL=128

Reply from 192.168.0.101: bytes=32 time=28ms TTL=128

Reply from 192.168.0.101: bytes=32 time=28ms TTL=128

Reply from 192.168.0.101: bytes=32 time=35ms TTL=128

Ping statistics for 192.168.0.101:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 18ms, Maximum = 35ms, Average = 27ms



```
C:\>
```

CONCLUSION: We learnt about Routing Information Protocols and implemented them in a simulator using Cisco Packet Tracer.

Experiment 12

**Aim: - To study and analyze the Packet details
using WireShark Tool**

THEORY:

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named **Ethereal**, the project was renamed Wireshark in May 2006 due to trademark issues.^[4]

Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets; it runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows. There is also a terminal-based (non-GUI) version called TShark. Wireshark, and the other programs distributed with it such as TShark, are free software, released under the terms of version 2 of the GNU General Public License.

Types of Protocols as seen in WireShark:

- TCP
- UDP
- ARP
- MDNS
- IGMPv2
- SSDP
- TLS
- QUIC
- ICMP

TCP:

The **Transmission Control Protocol (TCP)** is one of the main protocols of the Internet protocol suite. It originated in the initial network implementation in which it complemented the Internet Protocol (IP).

Therefore, the entire suite is commonly referred to as *TCP/IP*. TCP provides reliable, ordered, and error-checked delivery of a stream of octets (bytes) between applications running on hosts communicating via an IP network. Major internet applications such as the World Wide Web, email, remote administration, and file transfer rely on TCP, which is part of the Transport Layer of the TCP/IP suite. SSL/TLS often runs on top of TCP.

TCP is connection-oriented, and a connection between client and server is established before data can be sent. The server must be listening (passive open) for connection requests from clients before a connection is established. Three-way handshake (active open), retransmission, and error-detection adds to reliability but lengthens latency. Applications that do not require reliable data stream service may use the User Datagram Protocol (UDP), which provides a connectionless datagram service that prioritizes time over reliability. TCP employs network congestion avoidance. However, there are vulnerabilities to TCP including denial of service, connection hijacking, TCP veto, and reset attack.

UDP:

In computer networking, the **User Datagram Protocol (UDP)** is one of the core members of the Internet protocol suite. With UDP, computer applications can send messages, in this case referred to as *datagrams*, to other hosts on an Internet Protocol (IP) network. Prior communications are not required in order to set up communication channels or data paths.

UDP uses a simple connectionless communication model with a minimum of protocol mechanisms. UDP provides checksums for data integrity, and port numbers for addressing different functions at the source and destination of the datagram. It has no handshaking dialogues, and thus exposes the user's program to any unreliability of the underlying network; there is no guarantee of delivery, ordering, or duplicate protection. If error-correction facilities are needed at the network interface level, an application may use Transmission Control Protocol (TCP) or Stream Control Transmission Protocol (SCTP) which are designed for this purpose.

UDP is suitable for purposes where error checking and correction are either not necessary or are performed in the application; UDP avoids the overhead of such processing in the protocol stack. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for packets delayed due to retransmission, which may not be an option in a real-time system

ARP

The **Address Resolution Protocol (ARP)** is a communication protocol used for discovering the link layer address, such as a MAC address, associated with a given internet layer address, typically an IPv4 address. This mapping is a critical function in the Internet protocol suite. ARP was defined in 1982 by RFC 826, which is Internet Standard STD 37.

ARP has been implemented with many combinations of network and data link layer technologies, such as IPv4, Chaosnet, DECnet and Xerox PARC Universal Packet (PUP) using IEEE 802 standards, FDDI, X.25, Frame Relay and Asynchronous Transfer Mode (ATM).

In Internet Protocol Version 6 (IPv6) networks, the functionality of ARP is provided by the Neighbor Discovery Protocol (NDP).

MDNS

In computer networking, the **multicast DNS (mDNS)** protocol resolves hostnames to IP addresses within small networks that do not include a local name server. It is a zero-configuration service, using essentially the same programming interfaces, packet formats and operating semantics as the unicast Domain Name System (DNS). Although Stuart Cheshire designed mDNS as a stand-alone protocol, it can work in concert with standard DNS servers.

The mDNS protocol is published as RFC 6762, uses IP multicast User Datagram Protocol (UDP) packets, and is implemented by the Apple Bonjour and open source Avahi software packages, included in most Linux distributions. mDNS has also been implemented in Windows 10, initially limited to discovering networked printers, later becoming capable of resolving hostnames as well.

mDNS can work in conjunction with DNS Service Discovery (DNS-SD), a companion zero-configuration networking technique specified separately in RFC 6763.

IGMP

The **Internet Group Management Protocol (IGMP)** is a communications protocol used by hosts and adjacent routers on IPv4 networks to establish multicast group memberships. IGMP is an integral part of IP multicast and allows the network to direct multicast transmissions only to hosts that have requested them.

IGMP can be used for one-to-many networking applications such as online streaming video and gaming, and allows more efficient use of resources when supporting these types of applications.

IGMP is used on IPv4 networks. Multicast management on IPv6 networks is handled by Multicast Listener Discovery (MLD) which is a part of ICMPv6 in contrast to IGMP's bare IP encapsulation.

SSDP

The **Simple Service Discovery Protocol (SSDP)** is a network protocol based on the Internet protocol suite for advertisement and discovery of network services and presence information. It accomplishes this without assistance of server-based configuration mechanisms, such as Dynamic Host Configuration Protocol (DHCP) or Domain Name System (DNS), and without special static configuration of a network host. SSDP is the basis of the discovery protocol of Universal Plug and Play (UPnP) and is intended for use in residential or small office environments. It was formally described in an Internet Engineering Task Force (IETF) Internet Draft by Microsoft and Hewlett-Packard in 1999. Although the IETF proposal has since expired (April, 2000), SSDP was incorporated into the UPnP protocol stack, and a description of the final implementation is included in UPnP standards documents.

TLS

Transport Layer Security (TLS), and its now-deprecated predecessor, **Secure Sockets Layer (SSL)**, are cryptographic protocols designed to provide communications security over a computer network. Several versions of the protocols are widely used in applications such as email, instant messaging, and voice over IP, but its use as the *Security* layer in HTTPS remains the most publicly visible.

The TLS protocol aims primarily to provide privacy and data integrity between two or more communicating computer applications.

The TLS protocol comprises two layers: the TLS record and the TLS handshake protocols.

TLS is a proposed Internet Engineering Task Force (IETF) standard, first defined in 1999, and the current version is TLS 1.3 defined in August 2018. TLS builds on the earlier SSL specifications (1994, 1995, 1996) developed by Netscape Communications for adding the HTTPS protocol to their Navigator web browser.

QUIC

QUIC (pronounced "quick") is a general-purpose transport layer network protocol initially designed by Jim Roskind at Google, implemented, and deployed in 2012, announced publicly in 2013 as experimentation broadened, and described to the IETF. QUIC is used by more than half of all connections from the Chrome web browser to Google's servers. Microsoft Edge, Firefox, and Safari support it, even if not enabled by default.

Although its name was initially proposed as the acronym for "Quick UDP Internet Connections", IETF's use of the word QUIC is not an acronym; it is simply the name of the protocol. QUIC improves performance of connection-oriented web applications that are currently using TCP. It does this by establishing a number of multiplexed connections between two endpoints using User Datagram Protocol (UDP), and is designed to obsolesce TCP at the network layer for many applications, thus earning the protocol the occasional nickname "TCP/2".

QUIC works hand-in-hand with HTTP/2's multiplexed connections, allowing multiple streams of data to reach all the endpoints independently, and hence independent of packet losses involving other streams. In contrast, HTTP/2 hosted on Transmission Control Protocol (TCP) can suffer head-of-line-blocking delays of all multiplexed streams if any of the TCP packets are delayed or lost.

QUIC's secondary goals include reduced connection and transport latency, and bandwidth estimation in each direction to avoid congestion. It also moves congestion control algorithms into the user space at both endpoints, rather than the kernel space, which it is claimed will allow these algorithms to improve more rapidly. Additionally, the protocol can be extended with forward error correction (FEC) to further improve performance when errors are expected, and this is seen as the next step in the protocol's evolution.

ICMP

The **Internet Control Message Protocol (ICMP)** is a supporting protocol in the Internet protocol suite. It is used by network devices, including routers, to send error messages and operational information indicating success or failure when communicating with another IP address, for example, an error is indicated when a requested service is not available or that a host or router could not be reached. ICMP differs from transport protocols such as TCP and UDP in that it is not typically used to exchange data between systems, nor is it regularly employed by end-user network applications (with the exception of some diagnostic tools like ping and traceroute).

ICMP for IPv4 is defined in RFC 792. A separate ICMPv6, defined by RFC 4443, is used with IPv6.

Capturing from Wi-Fi						
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help						
Apply a display filter ... <Ctrl-/>						
No.	Time	Source	Destination	Protocol	Length	Info
5	1.051318	192.168.0.175	52.98.58.50	TCP	54	[TCP ACKed unseen segment] 55683 → 443 [ACK]
6	1.745033	192.168.0.175	192.168.0.255	UDP	170	57122 → 51007 Len=128
7	1.762517	HonHaiPr_1d:30:81	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.133
8	1.855369	192.168.0.175	172.217.160.206	UDP	76	53851 → 443 Len=34
9	1.903292	192.168.0.175	172.217.160.206	UDP	76	53851 → 443 Len=34
10	1.907214	172.217.160.206	192.168.0.175	UDP	69	443 → 53851 Len=27
11	2.172304	192.168.0.1	224.0.0.251	MDNS	86	Standard query 0x3213 PTR 192.168.0.120.in-a
12	2.172304	192.168.0.1	224.0.0.251	MDNS	86	Standard query 0x3214 PTR 192.168.0.120.in-a
13	2.749503	HonHaiPr_1d:30:81	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.133
14	2.991595	192.168.0.163	224.0.0.251	MDNS	130	Standard query 0x0000 PTR _homekit._tcp.loca
15	2.991595	fe80::14f7:705a:4c6...	ff02::fb	MDNS	150	Standard query 0x0000 PTR _homekit._tcp.loca
16	3.298551	192.168.0.163	224.0.0.251	IGMPv2	46	Membership Report group 224.0.0.251
17	4.015835	192.168.0.163	224.0.0.251	MDNS	130	Standard query 0x0000 PTR _homekit._tcp.loca
18	4.015835	fe80::14f7:705a:4c6...	ff02::fb	MDNS	150	Standard query 0x0000 PTR _homekit._tcp.loca
19	4.527088	HonHaiPr_1d:30:81	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.133
20	4.915608	HonHaiPr_1d:30:81	IntelCor_d6:31:6b	ARP	42	192.168.0.133 is at d8:0f:99:1d:30:81
21	4.925022	HonHaiPr_1d:30:81	IntelCor_d6:31:6b	ARP	42	192.168.0.133 is at d8:0f:99:1d:30:81
22	4.934788	HonHaiPr_1d:30:81	IntelCor_d6:31:6b	ARP	42	192.168.0.133 is at d8:0f:99:1d:30:81
23	4.934788	HonHaiPr_1d:30:81	IntelCor_d6:31:6b	ARP	42	192.168.0.133 is at d8:0f:99:1d:30:81
24	4.945497	HonHaiPr_1d:30:81	IntelCor_d6:31:6b	ARP	42	192.168.0.133 is at d8:0f:99:1d:30:81
> Frame 1: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{AB602647-81B6-4FBD-B765-9102}						
> Ethernet II, Src: Tp-LinkT_89:e7:a8 (d8:07:b6:89:e7:a8), Dst: IntelCor_d6:31:6b (40:74:e0:d6:31:6b)						
> Internet Protocol Version 4, Src: 52.98.58.50, Dst: 192.168.0.175						
> Transmission Control Protocol, Src Port: 443, Dst Port: 55682, Seq: 1, Ack: 1, Len: 0						
0000	40 74 e0 d6 31 6b d8 07	b6 89 e7 a8 08 00 45 00	@t..1k..			
0010	00 28 00 00 40 00 3d 06	0d e5 34 62 3a 32 c0 a8	..(..@.=. ..4b:2..			
0020	00 af 01 bb d9 82 62 24	40 39 7d af fd 68 50 10b\$ @9}..hP-			
0030	08 05 7f 30 00 00		...0..			
Wi-Fi: <live capture in progress> Packets: 364 · Displayed: 364 (100.0%) Profile: Default						

CONCLUSION: We learnt about different networking protocols that are being used today and used WireShark packet analyser to see live packet details and network protocols.