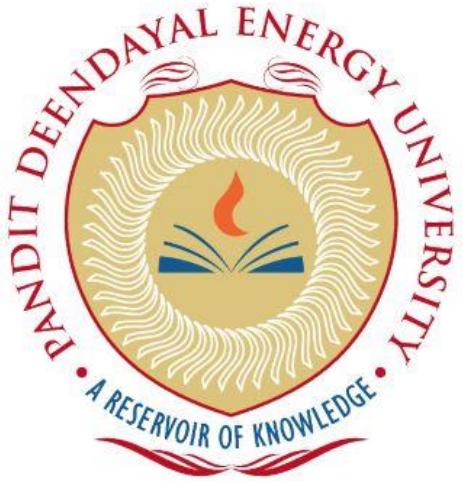
PANDIT DEENDAYAL ENERGY UNIVERSITY

SCHOOL OF TECHNOLOGY



Computer Networks Lab

20CP301P

LAB REPORT

B.Tech. (Computer Science and Engineering)

Semester 5

Submitted By:

HARSH SHAH

21BCP359

G11 Batch

INDEX

S No.	List of Practical	Date	Sign
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

EXPERIMENT 1

Aim

Simulation of Various Networking Topologies

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

Networking topologies refer to the arrangement of devices and their interconnections in a computer network. Different topologies offer unique advantages and drawbacks, influencing network performance, scalability, and fault tolerance. The choice of topology depends on specific requirements and use cases. Here is a brief overview of some common networking topologies:

1. **Bus Topology:** In a bus topology, all devices are connected to a central communication medium, typically a single cable or "bus." Devices share the same communication channel, and data is broadcast to all nodes on the bus. It is easy to implement and cost-effective for small networks, but it is susceptible to collisions, and a main bus failure can bring down the entire network.
2. **Star Topology:** In a star topology, all devices are connected to a central hub or switch. The hub manages communication between devices, making installation and troubleshooting simple. A failure in one connection usually does not affect the entire network, but dependence on the central hub can be a single point of failure.
3. **Ring Topology:** In a ring topology, devices are connected in a closed loop, where each device is connected to exactly two other devices. Data travels in one direction along the ring until it reaches the intended recipient. While simple and suitable for small networks, a failure in one node or connection can disrupt the entire network.
4. **Mesh Topology:** In a mesh topology, each device is directly connected to every other device, ensuring redundancy and fault tolerance. This topology offers high reliability due to multiple paths for data transmission. However, it is complex and expensive, as the number of connections increases significantly with the number of devices.
5. **Hybrid Topology:** A hybrid topology combines two or more different topologies, offering flexibility and optimization for network performance. It is suitable for tailoring networks to specific needs but can be complex to manage.

Procedure

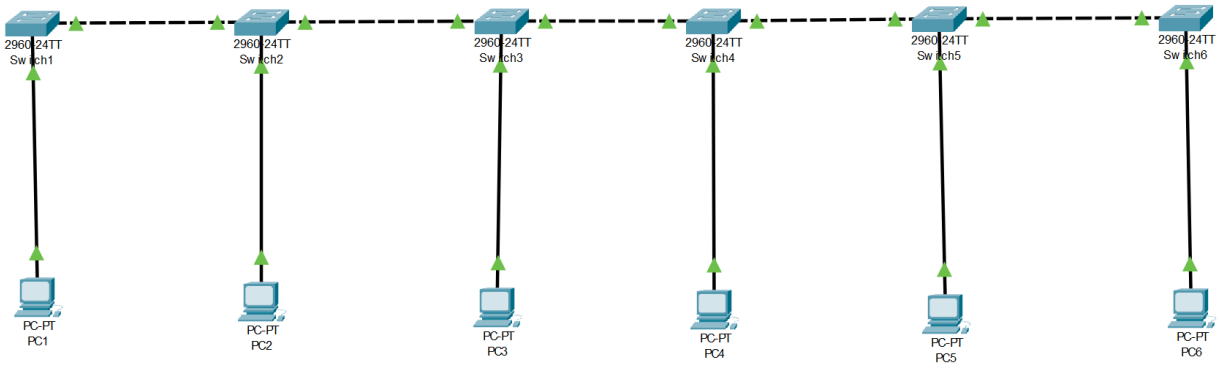
1. Open Cisco Packet Tracer and simulate the topologies of required size.
2. Assign the IP Addresses to the system.
3. Check the connectivity between the devices.

Steps

1. Open Cisco Packet Tracer software.
2. Build the topologies within Cisco Packet Tracer's workspace.
3. Add devices from the device list (computers, switches) and place them on the workspace.
4. Connect devices using appropriate cables (Ethernet cables or fiber optic cables).
5. Assign IP addresses and subnet masks to each device to enable communication within the same subnet.
6. Use the Ping tool located in the "Desktop" section of each device's configuration window.
7. Note the ease of setup, fault tolerance, scalability, and any limitations specific to each topology.
8. Utilize the simulation feature in Cisco Packet Tracer to simulate various scenarios.

Output

1. Bus Topology



2. Star

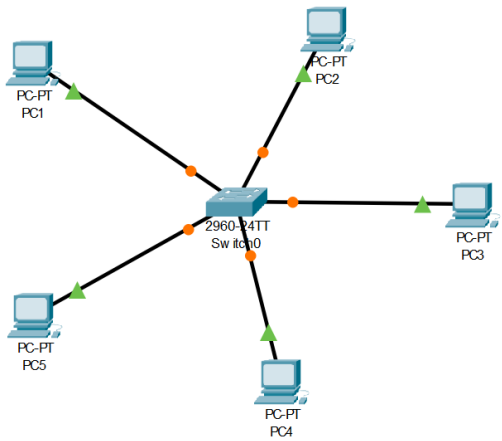


Fig 1: Using Switch

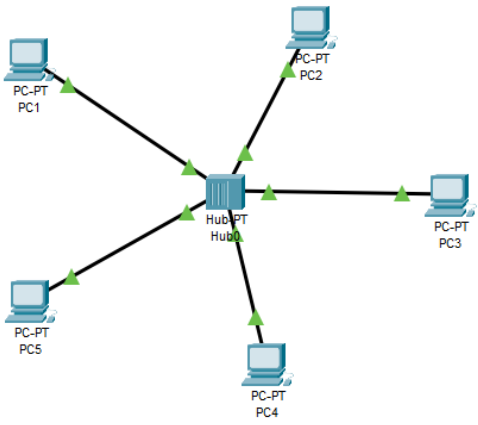
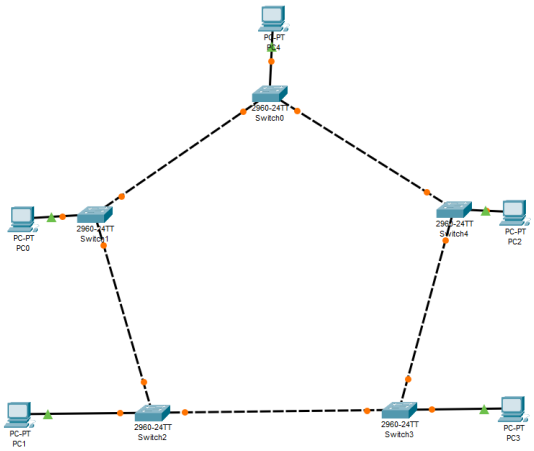
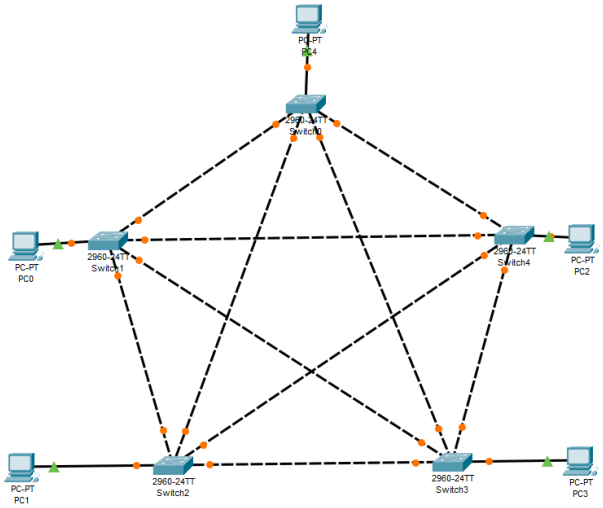


Fig 2: Using Hub

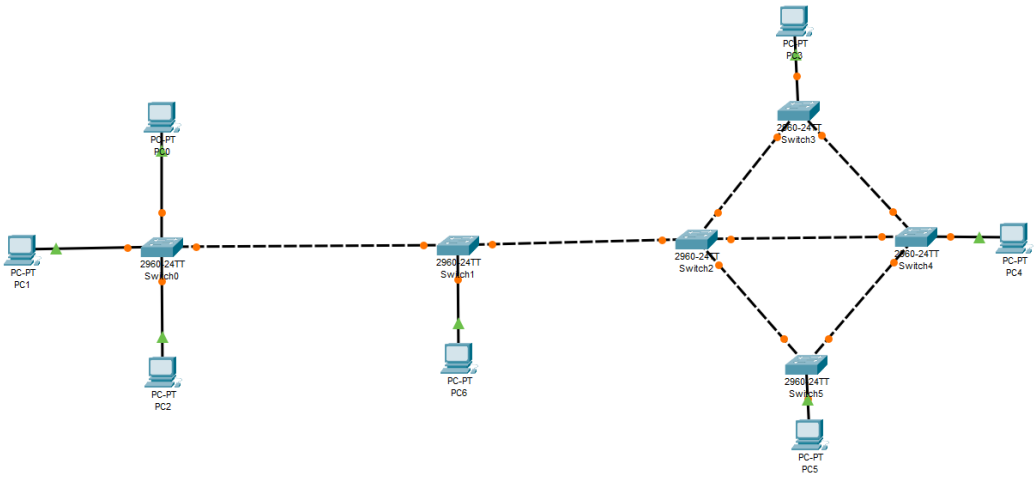
3. Ring



4. Mesh



5. Hybrid



Observation & Learning

The experiment revealed that the star topology is the most efficient in a LAN environment due to its centralized hub, ease of troubleshooting, and reduced risk of complete network failure. The hybrid topology provides flexibility, but proper planning and management are vital.

Understanding network requirements before selecting a topology is crucial, and network simulations play a key role in evaluating and optimizing different configurations. Continual evaluation ensures an optimal, reliable network.

Conclusion

In this experiment, we explored different networking topologies, considering their advantages and limitations. Bus topology is cost-effective but prone to collisions, while star topology is easy to troubleshoot but reliant on a central hub. Mesh topology offers fault tolerance, but complexity and cost are concerns. Hybrid topology allows flexibility by combining strengths.

Questions

1. **Which is the most efficient topology in LAN environment and Why?**
   In a LAN (Local Area Network) environment, the most efficient topology is the **Star Topology**. It offers several advantages that make it well-suited for LANs. Each device connects directly to a central hub or switch, allowing easy management and troubleshooting. Data transmission is directed through the hub, reducing the likelihood of collisions. Additionally, if a single connection fails, only that particular device is affected, leaving the rest of the network operational.

2. **How we can test the connectivity between the terminals?**
   Connectivity between terminals can be tested using various methods:
   - **Ping Test:** This test sends an ICMP (Internet Control Message Protocol) echo request from one terminal to another and checks for a response, verifying the connectivity and latency between the devices.
   - **Traceroute:** Traceroute tracks the path data packets take from the source terminal to the destination, helping identify any network bottlenecks or connectivity issues.
   - **Scanning:** Port scanning checks which network ports are open on a terminal, ensuring specific services or applications are reachable.

3. **What are the two categories of cable? In what type of connection, they are user?**
   The two categories of cables commonly used in networking are:
   - **Twisted Pair Cable:** Twisted pair cables are of two types: *Unshielded Twisted Pair (UTP)* and *Shielded Twisted Pair (STP)*. UTP cables are commonly used in LAN environments due to their cost-effectiveness and ease of installation. They are widely used for Ethernet connections in offices, homes, and data centres. STP cables offer better protection against electromagnetic interference and are used in environments with high levels of electrical noise.
   - **Fiber Optic Cable:** Fiber optic cables use light pulses to transmit data, offering high bandwidth and immunity to electromagnetic interference. They are used for long-distance connections, high-speed data transfer, and in environments where electrical interference is a concern, such as in data centres and telecommunications networks.

EXPERIMENT 2

Aim

Simulation of Virtual Local Area Network

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

A Virtual Local Area Network (VLAN) is a logical segmentation of a physical network into isolated groups. VLANs enable devices to communicate as if they are on separate networks, even if physically connected. This enhances efficiency, security, and management in various ways:

- **Efficiency:** VLANs segment network traffic, reducing congestion and enhancing performance. Broadcasts are limited to devices within the same VLAN, preventing unnecessary network overload.
- **Security:** Devices in different VLANs are isolated, limiting unauthorized communication. Inter-VLAN communication requires routing, enhancing control over data flow and security.
- **Flexibility:** VLANs can be organized by department, function, or need, regardless of physical location. This simplifies network management and optimizes resource allocation.
- **Management:** VLANs enable logical network segmentation, easing administration and troubleshooting. VLAN identification tags manage traffic flow.
- **Inter-VLAN Communication:** Routers or Layer 3 switches facilitate communication between VLANs by routing data.
- **Configuration:** VLAN setup involves configuring devices to recognize VLAN tags. Port-Based, Tagged, and Dynamic VLANs cater to various needs.

In summary, VLANs enhance network efficiency, security, and management by logically segmenting a physical network into isolated groups, enabling better resource utilization, security, and adaptability.

Procedure

1. Open Cisco Packet Tracer and simulate the sample topologies with required size of VLAN.
2. Perform Necessary Operation on Switch to create and configure VLAN.
3. Check the connectivity between the devices.

Steps

1. Launch Cisco Packet Tracer.
2. Create a basic network topology.
3. Connect devices using Ethernet cables.
4. Configure VLAN support on switches.
5. Assign ports to specific VLANs on switches.
6. Configure computers in the same VLAN.
7. Configure different computers in separate VLANs.
8. Test ping commands between devices in same/different VLANs.
9. Enable routing between sub-interfaces.
10. Test ping commands between devices in different VLANs.
11. Observe and analyse device behaviour within VLANs.

Output

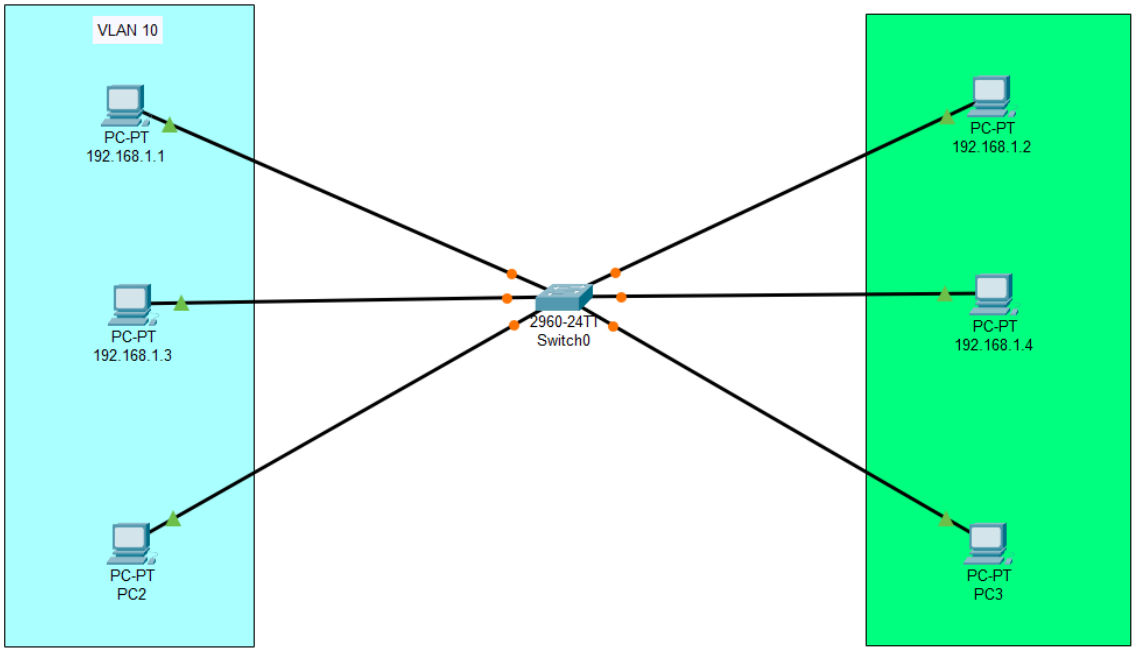


Figure 1. Topology

Configuration

```
Switch(config)#vlan 10
Switch(config-vlan)#name IT
Switch(config-vlan)#vlan 20
Switch(config-vlan)#name Sales

Switch#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#int range f0/1-3
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 10
Switch(config-if-range)#int range fa0/4-6
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 20
```

```
VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------------
1    default                           active    Fa0/7, Fa0/8, Fa0/9, Fa0/10
                                                 Fa0/11, Fa0/12, Fa0/13, Fa0/14
                                                 Fa0/15, Fa0/16, Fa0/17, Fa0/18
                                                 Fa0/19, Fa0/20, Fa0/21, Fa0/22
                                                 Fa0/23, Fa0/24, Gig0/1, Gig0/2
10   IT                                active    Fa0/1, Fa0/2, Fa0/3
20   Sales                             active    Fa0/4, Fa0/5, Fa0/6
1002 fddi-default                      active
1003 token-ring-default                active
1004 fddinet-default                   active
1005 trnet-default                     active
```

```
FastEthernet0 Connection:(default port)

   Connection-specific DNS Suffix..:
   Link-local IPv6 Address.....: FE80::2D0:BCFF:FE06:BC89
   IPv6 Address.....: ::
   IPv4 Address.....: 192.168.1.1
   Subnet Mask.....: 255.255.255.0
   Default Gateway.....: ::
                                 0.0.0.0

Bluetooth Connection:

   Connection-specific DNS Suffix..:
   Link-local IPv6 Address.....: ::
   IPv6 Address.....: ::
   IPv4 Address.....: 0.0.0.0
   Subnet Mask.....: 0.0.0.0
   Default Gateway.....: ::
                                 0.0.0.0
```

```
C:\>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\>ping 192.168.1.3

Pinging 192.168.1.3 with 32 bytes of data:

Reply from 192.168.1.3: bytes=32 time<1ms TTL=128
Reply from 192.168.1.3: bytes=32 time=1ms TTL=128
Reply from 192.168.1.3: bytes=32 time<1ms TTL=128
Reply from 192.168.1.3: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.1.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

Observation & Learning

- Devices within the same VLAN can communicate freely, while communication between devices in different VLANs requires routing.
- VLANs segregate broadcast domains, reducing unnecessary traffic.
- Trunk ports allow traffic exchange between switches for different VLANs.
- VLAN tagging aids in directing traffic accurately to specific VLANs.
- Inter-VLAN communication necessitates routing devices.
- VLANs enhance network efficiency by controlling traffic and broadcasts.
- Security is improved as devices in different VLANs are isolated.
- Inter-VLAN communication is enabled through routers or Layer 3 switches.
- Proper VLAN tagging and trunk configuration are vital for correct data routing.
- Simulations assist in understanding VLAN behaviour and configurations.

Conclusion

Simulating Virtual Local Area Networks (VLANs) using Cisco Packet Tracer offers valuable insights into network segmentation and behaviour. The experiment revealed that VLANs effectively isolate and manage network traffic, enhance security, and streamline network management. Inter-VLAN communication can be achieved through routing devices. VLAN tagging and trunk ports are pivotal for ensuring accurate data transmission. By understanding the concepts and simulating scenarios, network professionals can confidently implement VLANs to optimize network efficiency, security, and management in real-world scenarios.

Questions

1. **What is the maximum number of VLAN can be created in a network?**
   The maximum number of VLANs that can be created in a network depends on the networking equipment being used. In most cases, modern networking switches support up to 4096 VLANs, as defined by the IEEE 802.1Q standard. However, practical limits may vary based on the switch's hardware capabilities and the network's overall design.
2. **What is mean by MTU? What is the value of MTU in Ethernet?**
   MTU (Maximum Transmission Unit) refers to the maximum size of a data packet that can be transmitted over a network without being fragmented. In Ethernet, the standard MTU size is 1500 bytes for most networks. This value accounts for the Ethernet frame header and payload, leaving 1500 bytes for the actual data. Jumbo frames, which have larger MTU sizes, are sometimes used to improve data transfer efficiency, but their use requires compatible hardware and configurations across the network.
3. **What happen when the broadcast operation is performed from a system in certain VLAN?**
   When a broadcast operation, such as an ARP (Address Resolution Protocol) request, is performed from a system in a certain VLAN, the broadcast will only propagate within that specific VLAN. Broadcast traffic does not cross VLAN boundaries. This isolation ensures that broadcast storms and unnecessary traffic are contained within the intended VLAN, preventing them from affecting devices in other VLANs. This segmentation enhances network performance and security by preventing unnecessary broadcast traffic from affecting the entire network.

EXPERIMENT 3

Aim

Simulation of Spanning Tree Protocol (STP)

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

The Spanning Tree Protocol (STP) is a network protocol used in Ethernet networks to **prevent loops** in bridged or switched networks. Loops can cause **broadcast storms** and lead to network congestion and instability. STP accomplishes this by creating a **loop-free logical topology** within a network, ensuring that there is only **one active path** between any two devices in the network.

Here are some key concepts and theory about the Spanning Tree Protocol:

- **Loop Prevention**: STP ensures that Ethernet networks do not have redundant loops, preventing broadcast storms and network instability.
- **Root Bridge**: The Root Bridge is the central point in the network topology, serving as a reference for all other bridges to determine their paths.
- **Root Port**: Each non-Root Bridge selects a Root Port, which is the shortest path to the Root Bridge, ensuring efficient data forwarding.
- **Designated Ports**: Designated Ports are responsible for forwarding traffic within network segments, optimizing communication.
- **Blocked Ports**: STP places certain ports in a blocked state to break loops, temporarily preventing data flow through them.
- **BPDU**: Bridge Protocol Data Units (BPDUs) are exchanged between bridges to convey topology information, facilitating network stability.
- **STP States**: Ports transition through Blocking, Listening, Learning, and Forwarding states to establish a loop-free topology while reacting to topology changes.

STP is a crucial protocol for ensuring the stability and reliability of Ethernet networks, especially in environments where redundancy is required. By preventing loops and ensuring a single active path between devices, STP helps maintain network integrity and performance.

Procedure

1. **Set Up Physical Connections:**

   Connect the switches using Ethernet cables to create a network topology with redundant links, which can potentially form loops.

2. **Access Switches:**

   Use a computer with terminal software (e.g., PuTTY or Terminal) to access the command-line interface (CLI) of each switch. You may need the switch's console cable or access through SSH.

3. **Configure STP:**

   Enter the CLI of each switch.

   Enter global configuration mode by typing: **enable** or **configure terminal**.

4. **View STP Information:**

   Use the following command to view STP information: `show spanning-tree`

   Note the Root Bridge, Root Port, and Designated Ports for each switch.

5. **Introduce Changes:**

   Disconnect and reconnect Ethernet cables to simulate network changes. Observe how STP reacts to topology changes.

6. **Analyse Results:**

   Analyse the data obtained during the simulation, including the Root Bridge selection, port states, and convergence times.

Output

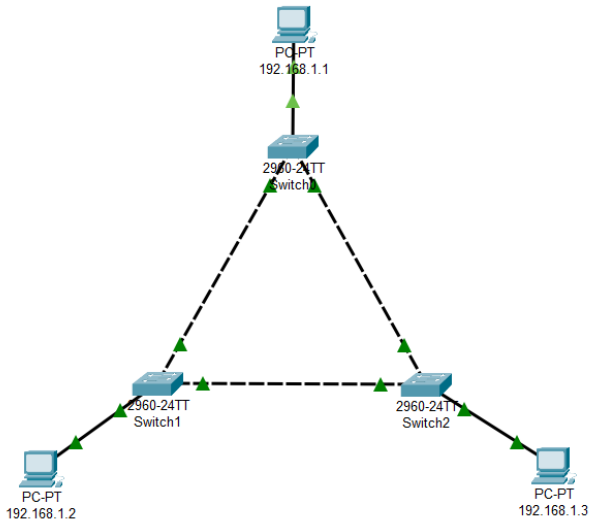


Figure 1: Without Spanning Tree

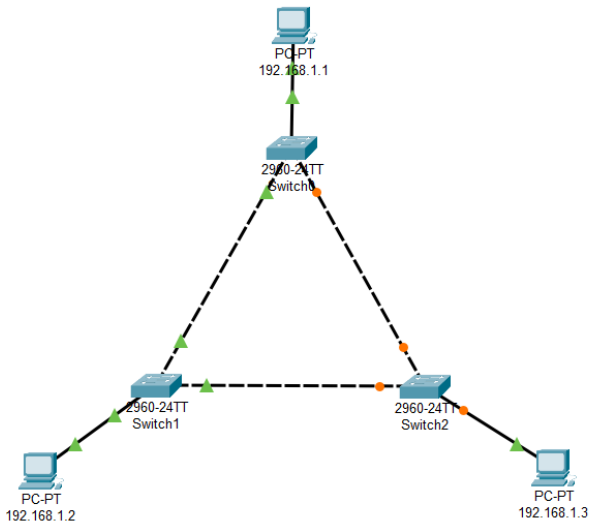


Figure 2: With Spanning Tree

```
Switch>
Switch>enable
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#spanning-tree vlan 1
Switch(config)#^Z
Switch#
%SYS-5-CONFIG_I: Configured from console by console

Switch#wr
Building configuration...
[OK]
```

Figure 1: Enabling Spanning Tree

```
Switch#enable
Switch#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#no sp
Switch(config)#no spanning-tree
Switch(config)#no spanning-tree vlan 1
Switch(config)#^Z
Switch#
%SYS-5-CONFIG_I: Configured from console by console

Switch#wr
Building configuration...
[OK]
```

Figure 2: Disabling Spanning Tree

```
Switch#sh spanning-tree
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    32769
              Address     0001.C9E2.C5E7
              Cost        19
              Port        2(FastEthernet0/2)
              Hello Time  2 sec  Max Age 20 sec  Forward Delay 15
sec

  Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
             Address     000A.F32E.19B0
             Hello Time  2 sec  Max Age 20 sec  Forward Delay 15
sec
             Aging Time  20

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- --------
--------------------------------
Fa0/1            Desg FWD 19        128.1    P2p
Fa0/2            Root FWD 19        128.2    P2p
Fa0/3            Desg FWD 19        128.3    P2p
```

Figure 5: Spanning Tree Details

Observation & Learning

Throughout the experiment, we observed that Spanning Tree Protocol (STP) played a pivotal role in preventing network loops by strategically blocking redundant connections while maintaining network stability. Notably, STP exhibited rapid adaptation to network changes, minimizing disruptions caused by topology modifications. The autonomous selection of a Root Bridge by STP influenced the network structure and dictated the behaviour of switch ports, ensuring a loop-free topology.

Understanding the various port states, including Blocking and Forwarding, was key to grasping how STP maintains network integrity. Overall, STP's significance in networks with multiple connections became evident as it effectively safeguarded against loops, promoted reliability, and showcased its ability to react swiftly to changes, reducing network downtime and enhancing overall network responsiveness.

Conclusion

In conclusion, the experiment demonstrated the effectiveness of the Spanning Tree Protocol in preventing network loops and maintaining a stable Ethernet network. STP successfully selected a Root Bridge and configured port states to ensure a loop-free topology. It reacted promptly to topology changes, which is essential for network reliability.

Questions

1. **What happens if STP is disabled on all the switches in the network?**

   Disabling STP on all switches in the network can lead to the **formation of network loops**. This can result in **broadcast storms**, **network congestion**, and **instability** due to redundant paths. It's essential to have STP enabled to prevent such issues.

2. **What is meant by Designated, Root, and Blocked Interfaces?**

   Designated Interface: These are ports on a switch that have been selected to forward traffic for a particular network segment. They are responsible for forwarding data within their respective segments.

   Root Interface: The Root Interface is the switch port that offers the shortest path to the Root Bridge. It is in the Forwarding state and serves as the best path to reach the Root Bridge.

   Blocked Interface: Blocked Interfaces are ports that are in a Blocking state. They do not forward data but exist to prevent loops by temporarily deactivating redundant paths.

3. **What are the different ways to identify the Root in the STP?**

   The Root Bridge in STP is identified based on the **Bridge ID**. The Bridge ID is composed of a **priority value** and a **MAC address**. The Root Bridge is the bridge with the **lowest Bridge ID**. If two bridges have the same priority, the one with the **lowest MAC address** becomes the Root Bridge. Additionally, you can use the `show spanning-tree` command on a switch to directly identify the Root Bridge in the output.

EXPERIMENT 4

Aim

Simulation of Routing Information Protocol (RIP)

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

RIP, or Routing Information Protocol, is one of the oldest and simplest routing protocols used in computer networks. It falls under the category of distance-vector routing protocols and is primarily used in small to medium-sized networks. Here's an explanation of RIP:

1. **Distance-Vector Protocol:**
   - RIP operates as a distance-vector protocol, which means it determines the best path to a destination based on the number of hops (routers) it takes to reach that destination. Each router maintains a routing table that contains information about the number of hops to various network destinations.
2. **RIP Versions:**
   - There are two main versions of RIP: RIP-1 and RIP-2.
   - **RIP-1:** The original RIP version, defined in RFC 1058, supports classful routing and uses hop count as its metric.
   - **RIP-2:** An improved version of RIP, defined in RFC 1723, supports classless routing (CIDR), VLSM (Variable Length Subnet Masking), and includes support for route authentication and multicast routing.
3. **Metric:**
   - RIP uses hop count as its metric to determine the best path to a destination. It assumes that the shortest path is the one with the fewest hops, which may not always be the most efficient route in terms of bandwidth or latency.
4. **Periodic Updates:**
   - RIP routers periodically broadcast their entire routing table to neighbouring routers. By default, RIP sends updates every 30 seconds.
   - These updates contain information about reachable networks and their associated hop counts.

5. **Split Horizon and Route Poisoning:**
   - To prevent routing loops, RIP employs mechanisms like "split horizon" and "route poisoning."
   - **Split Horizon:** Routers don't advertise routes back to the neighbour from which they learned them, reducing the risk of loops.
   - **Route Poisoning:** When a router detects a route has failed, it advertises the failed route to its neighbours with an infinite metric (usually 16 hops). This informs other routers that the route is no longer valid.
6. **Convergence Time:**
   - One limitation of RIP is its relatively slow convergence time. When a network change occurs, it takes time for RIP routers to recognize the change, update their routing tables, and propagate the changes throughout the network.
7. **Loop Prevention:**
   - RIP uses a maximum hop count of 15 to prevent routing loops. Routes with a hop count of 16 or higher are considered unreachable.
8. **Authentication (in RIP-2):**
   - RIP-2 introduces the option for route authentication, which helps secure routing information exchanges between routers.
9. **RIP Limitations:**
   - RIP has several limitations, including its limited support for large networks, slow convergence, and its reliance on hop count as the sole metric, which may not always reflect the best path in terms of performance.
10. **Common Use Cases:**
    - RIP is typically used in small to medium-sized networks, such as home networks or small office setups, where simplicity is more important than advanced routing features.

In summary, RIP is a basic routing protocol that uses hop count as a metric to determine the best paths in a network. While it's simple to configure and suitable for smaller networks, it has limitations in terms of scalability and convergence speed, making it less suitable for large, complex networks. RIP-2, with its additional features, is an improvement over the original RIP protocol.

Procedure

1. **Physical Setup:**

   Set up the routers in a network topology. Connect them using Ethernet cables.

2. **Access Routers:**

   Use a computer with terminal software to access the command-line interface (CLI) of each router. You may need a console cable or access through SSH.

3. **Configure RIP:**

   Enter the CLI of each router.

   Enter global configuration mode by typing: **enable** or **configure terminal**.

   Configure RIP on each router using the following commands:

   ```
   router rip
   version 2
   network <network-address>
   ```

4. **View Routing Table:**
   Use the following command to view the routing table:

   ```
   show ip route
   ```

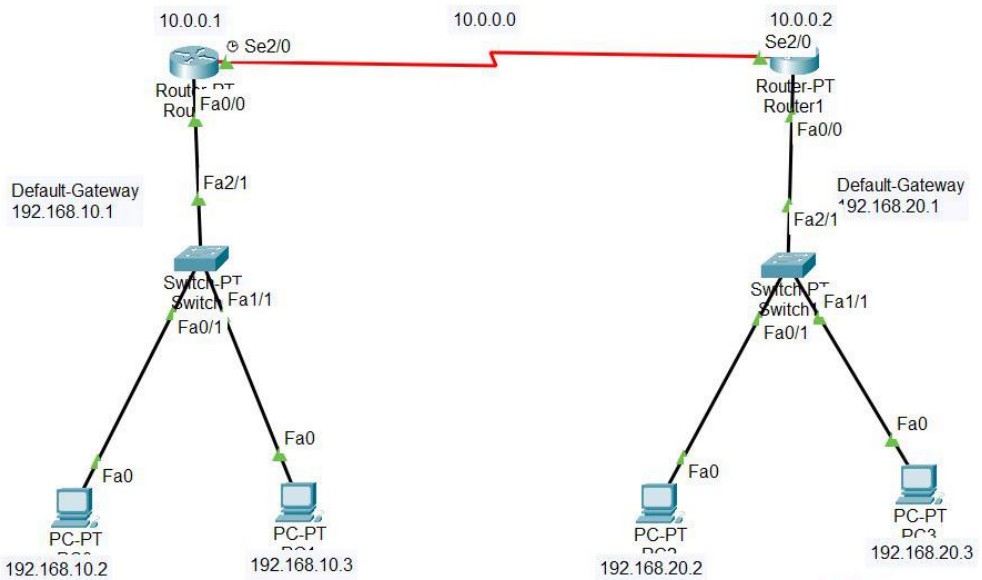
5. **Introduce Network Changes:**

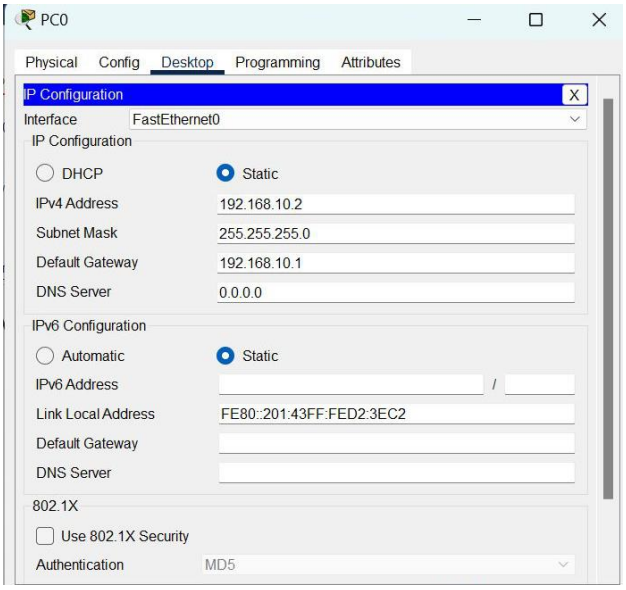
   Disconnect and reconnect Ethernet cables to simulate network changes.

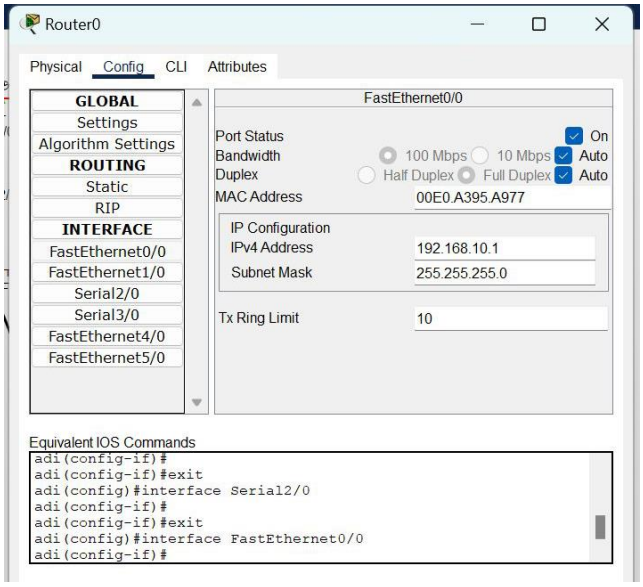
6. **Monitor RIP Updates:**

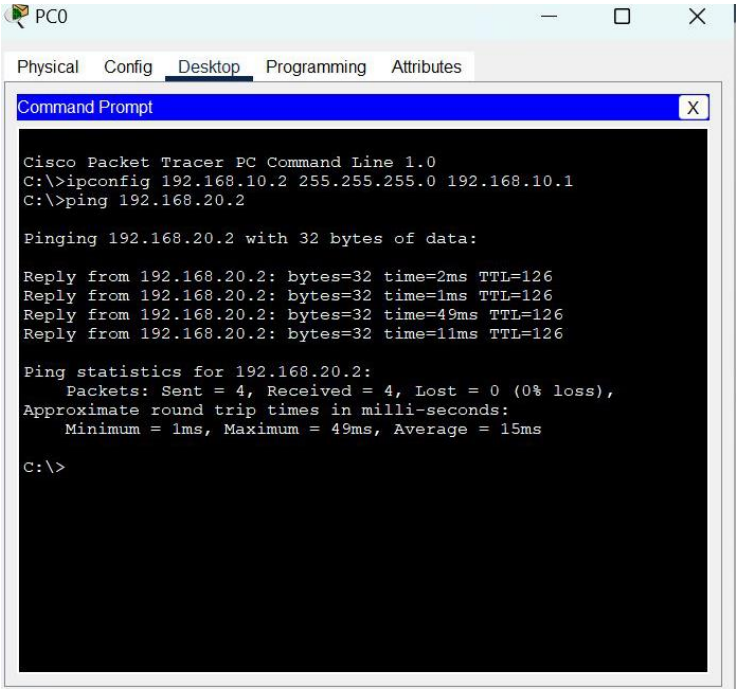
   Continuously monitor the routing tables on both routers using the **show ip route** command as RIP updates the routing information.

Output









Observation & Learning

- **RIP Updates**: During the experiment, we observed that RIP routers exchanged routing information with each other. This exchange of information allowed the routers to learn about available paths to various network destinations.
- **Routing Table**: The routers maintained routing tables containing information about the network topology, including network addresses and associated metrics (hop counts).
- **Path Selection**: RIP selected paths based on hop counts, preferring paths with fewer hops to reach a destination network.
- **Network Changes**: When we introduced network changes by disconnecting and reconnecting cables, RIP routers updated their routing tables to adapt to the new topology.

Conclusion

In conclusion, the experiment demonstrated the basic operation of the Routing Information Protocol (RIP) in exchanging routing information among routers to establish paths from source to destination. RIP routers efficiently updated their routing tables in response to network changes, showcasing their adaptability in maintaining network connectivity.

Questions

1. **What type of ports are used for the connection between the routers?**

   Ethernet ports (typically Fast Ethernet or Gigabit Ethernet) are commonly used for connecting routers in a network.

2. **How does RIP ensure the path from source to destination?**

   RIP ensures the path from source to destination by exchanging routing information between routers and selecting the path with the fewest hops (shortest path) based on hop counts.

3. **What is the importance of the gateway address?**

   The gateway address (also known as the default gateway) is crucial in routing as it serves as the exit point for traffic leaving a local network. It enables devices within a network to access resources outside of that network, such as the internet, by forwarding traffic to the appropriate router for further routing.

EXPERIMENT 5

Aim

Simulation of Static Routing in Cisco Packet Tracer

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

Static routing is a method of configuring network routers to use a fixed, predetermined path or route for forwarding data packets from the source to the destination. In contrast to dynamic routing, where routes are determined dynamically based on network conditions and routing protocols, static routing requires network administrators to manually configure and maintain the routing tables. Here's an explanation of static routing:

a. Manual Configuration:

In a static routing setup, network administrators manually define the routing tables on each router in the network. These routing tables contain information about which network addresses or subnets are reachable and through which next-hop router or interface data packets should be forwarded.

b. Predictable and Simple:

Static routing is straightforward and easy to configure. It is often used in small to medium-sized networks where the network topology is simple and stable. Because routes are manually defined, the network behavior is predictable and does not change dynamically.

c. Use Cases:

- Static routing is commonly used for simple network setups, such as small office networks or home networks.
- It can be used for routing between isolated networks or network segments where the topology rarely changes.
- In scenarios where security is a concern, static routing can be employed to ensure that traffic follows a specific path, reducing the risk of unauthorized routing changes.

d. Limitations:

- Static routing does not adapt to network changes. If a router or link fails or if the network topology changes, manual updates to the routing tables are required to maintain connectivity.
- In larger and more complex networks, static routing can become difficult to manage and may lead to routing inefficiencies.
- It's not suitable for load balancing or optimizing traffic in dynamic network environments.

e. Examples:

- A small office with a single router connecting to the Internet might use static routing to direct all outgoing traffic to the ISP's gateway.
- In a lab or test network, static routes can be configured to simulate specific network conditions for testing and troubleshooting.

In summary, static routing is a basic method of configuring routers to define fixed routes for data packets. It is ideal for small, stable networks where simplicity and predictability are more important than adaptability to changing network conditions. In larger, dynamic networks, dynamic routing protocols like OSPF and BGP are typically preferred for their ability to adapt to changes in the network.

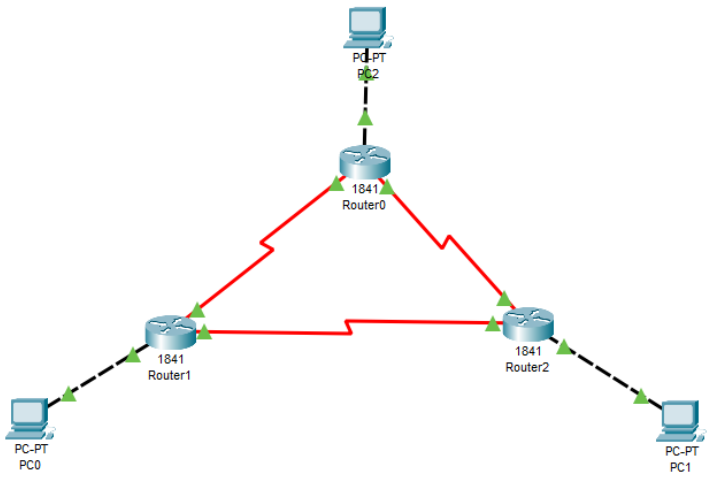
Procedure:

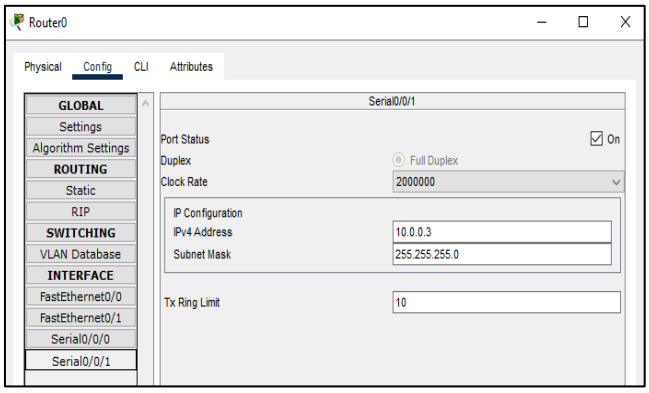
1. Open Cisco Packet Tracer and simulate the sample topologies for Static Routing.
2. Perform Necessary Operation on Router to create and configure Static Routing.
3. Check the connectivity between the devices.

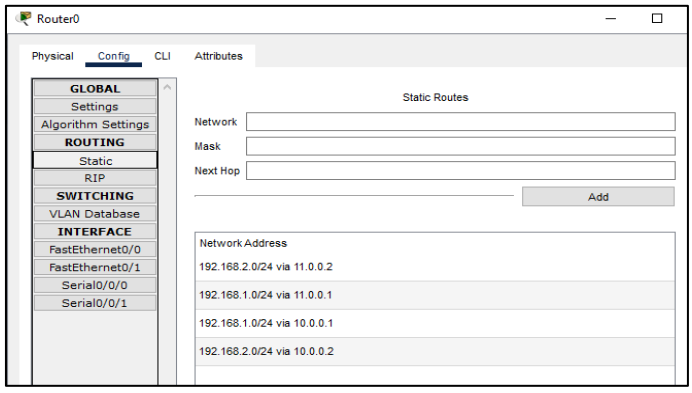
Steps:

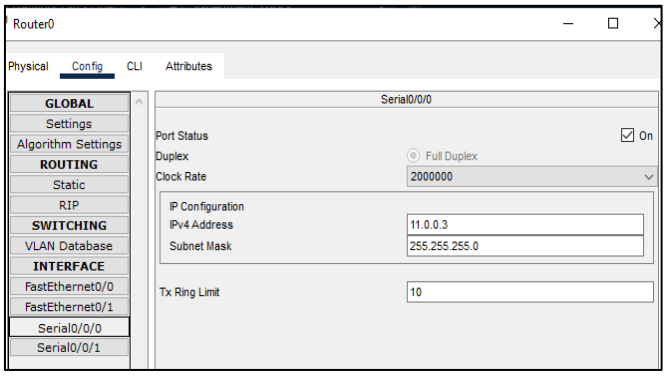
1. **Topology Setup:** Create your network topology in Cisco Packet Tracer with routers and connections.
2. **IP Address Configuration:** Assign IP addresses to router interfaces and connected devices.
3. **Basic Connectivity:** Ensure basic connectivity between devices.
4. **Static Route Configuration:** Set up static routes on routers to specify how to reach different subnets.
5. **Testing Static Routes:** Verify that the static routes work as expected by testing connectivity.
6. **Save Configurations:** Save router configurations to persist changes.

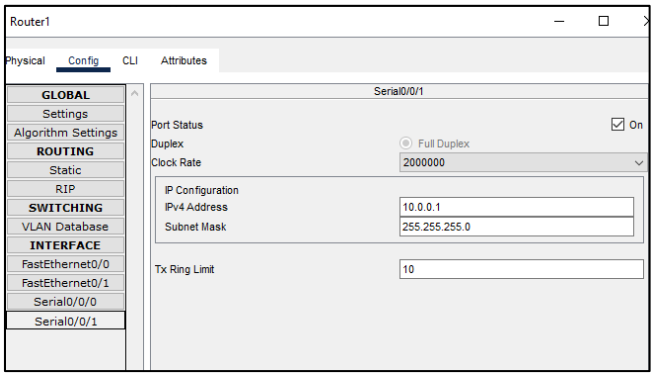
Output:

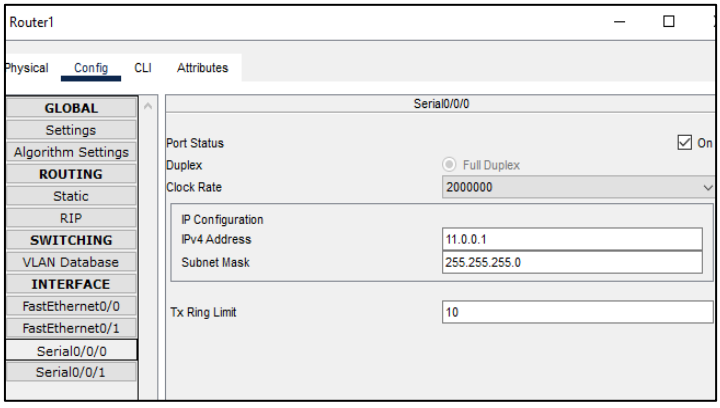

PC-PT
PC2
1841
Router0
1841
Router1
1841
Router2
PC-PT
PC0
PC-PT
PC1

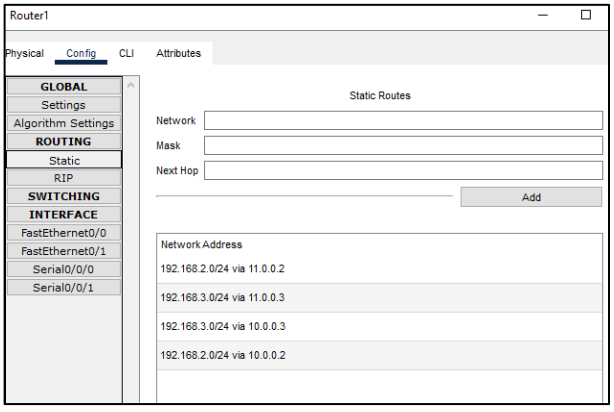

Router0
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
ROUTING
Static
RIP
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/0
FastEthernet0/1
Serial0/0/0
Serial0/0/1
Serial0/0/1
Port Status
On
Duplex
Full Duplex
Clock Rate
2000000
IP Configuration
IPv4 Address
10.0.0.3
Subnet Mask
255.255.255.0
Tx Ring Limit
10

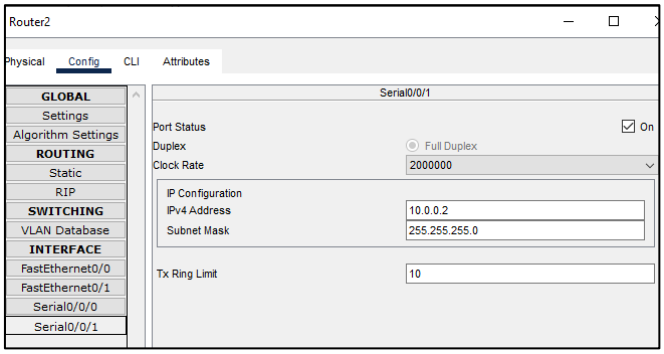

Router0
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
ROUTING
Static
RIP
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/0
FastEthernet0/1
Serial0/0/0
Serial0/0/1
Static Routes
Network
Mask
Next Hop
Add
Network Address
192.168.2.0/24 via 11.0.0.2
192.168.1.0/24 via 11.0.0.1
192.168.1.0/24 via 10.0.0.1
192.168.2.0/24 via 10.0.0.2

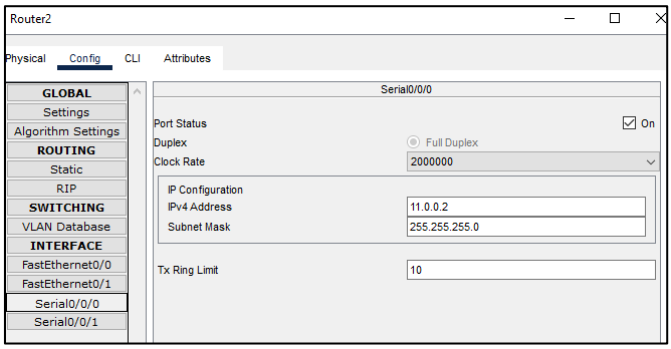

Router0
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
ROUTING
Static
RIP
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/0
FastEthernet0/1
Serial0/0/0
Serial0/0/1
Serial0/0/0
Port Status
On
Duplex
Full Duplex
Clock Rate
2000000
IP Configuration
IPv4 Address
11.0.0.3
Subnet Mask
255.255.255.0
Tx Ring Limit
10

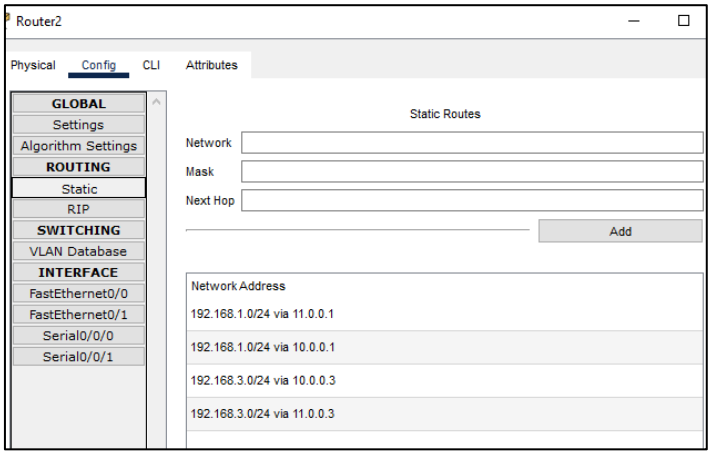

Router1
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
ROUTING
Static
RIP
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/0
FastEthernet0/1
Serial0/0/0
Serial0/0/1
Serial0/0/1
Port Status
On
Duplex
Full Duplex
Clock Rate
2000000
IP Configuration
IPv4 Address
10.0.0.1
Subnet Mask
255.255.255.0
Tx Ring Limit
10

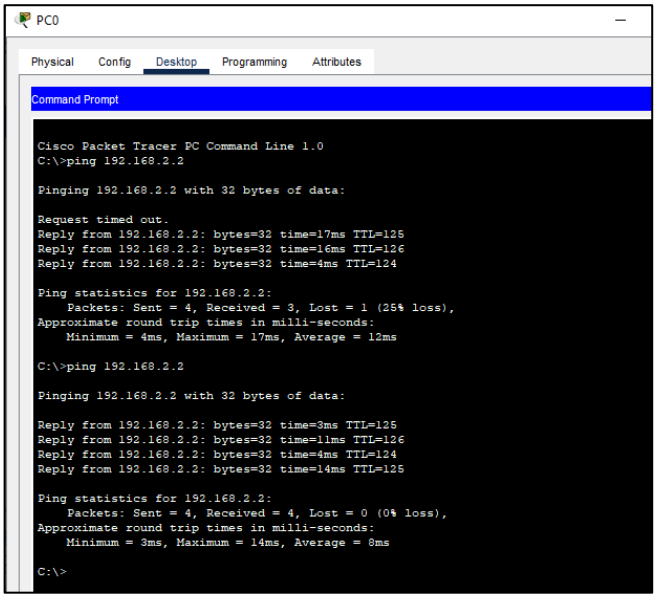

Router1
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
ROUTING
Static
RIP
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/0
FastEthernet0/1
Serial0/0/0
Serial0/0/1
Serial0/0/0
Port Status
On
Duplex
Full Duplex
Clock Rate
2000000
IP Configuration
IPv4 Address
11.0.0.1
Subnet Mask
255.255.255.0
Tx Ring Limit
10


Router1
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
ROUTING
Static
RIP
SWITCHING
INTERFACE
FastEthernet0/0
FastEthernet0/1
Serial0/0/0
Serial0/0/1
Static Routes
Network
Mask
Next Hop
Add
Network Address
192.168.2.0/24 via 11.0.0.2
192.168.3.0/24 via 11.0.0.3
192.168.3.0/24 via 10.0.0.3
192.168.2.0/24 via 10.0.0.2









Observation & Learning

In the experiment with Cisco Packet Tracer, I observed the successful setup of a network topology, IP assignment, and basic connectivity. Static routing was configured to enable inter-subnet communication, and testing validated its functionality. I learned the importance of network setup, the suitability of static routing for simple networks, the necessity of connectivity verification, and the limitations of static routing in dynamic or large networks.

Conclusion

In summary, the experiment showed the successful use of static routing for inter-subnet communication. It emphasized the importance of precise network setup and connectivity verification. However, for dynamic or larger networks, dynamic routing protocols are a more suitable choice due to their adaptability.

EXPERIMENT 6

Aim

Simulation of Open Shortest Path First (OSPF)

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

Open Shortest Path First (OSPF) is an Interior Gateway Protocol (IGP) that is used to route data within a single autonomous system (AS). OSPF is a link-state protocol, which means that it maintains a database of the topology of the network. This database is used to calculate the shortest path between any two nodes in the network. OSPF is a classless protocol, which means that it does not distinguish between different classes of IP addresses. This makes OSPF a good choice for networks that use a variety of IP address classes. OSPF is a reliable and scalable protocol that is widely used in enterprise networks.

OSPF uses a flooding algorithm to distribute routing information throughout the network. When a router joins an OSPF network, it sends a Hello packet to all of its neighbors. The Hello packet contains information about the router's interface, such as its IP address and the OSPF version that it supports. If a neighbor responds to the Hello packet, then the two routers form a neighbor relationship. Once a neighbor relationship is established, the routers can exchange routing information.

OSPF uses a link-state database to store information about the topology of the network. The link-state database is a collection of link-state advertisements (LSAs). Each LSA contains information about a single link, such as the link's IP address, the link's cost, and the link's state. The link-state database is used to calculate the shortest path between any two nodes in the network.

OSPF uses a Dijkstra algorithm to calculate the shortest path between any two nodes in the network. The Dijkstra algorithm is a shortest path algorithm that is used to find the shortest path between a source node and a destination node in a graph. The Dijkstra algorithm works by building a tree of nodes, where each node in the tree represents a possible path from the source node to the destination node. The algorithm starts by adding the source node to the tree. Then, it adds the node with the lowest cost to the tree. The algorithm continues to add nodes to the tree until the destination node is reached.

OSPF is a reliable and scalable protocol that is widely used in enterprise networks. OSPF is a link-state protocol that maintains a database of the topology of the network. This database is used to calculate the shortest path between any two nodes in the network. OSPF is a classless protocol that does not distinguish between different classes of IP addresses. This makes OSPF a good choice for networks that use a variety of IP address classes. OSPF is a reliable and scalable protocol that is widely used in enterprise networks.

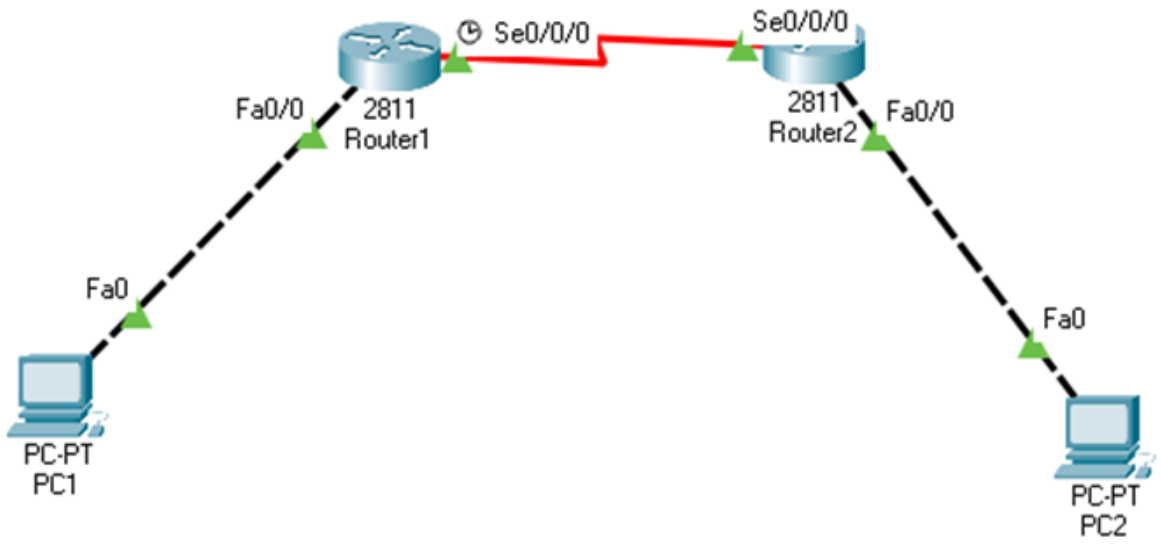
Procedure:

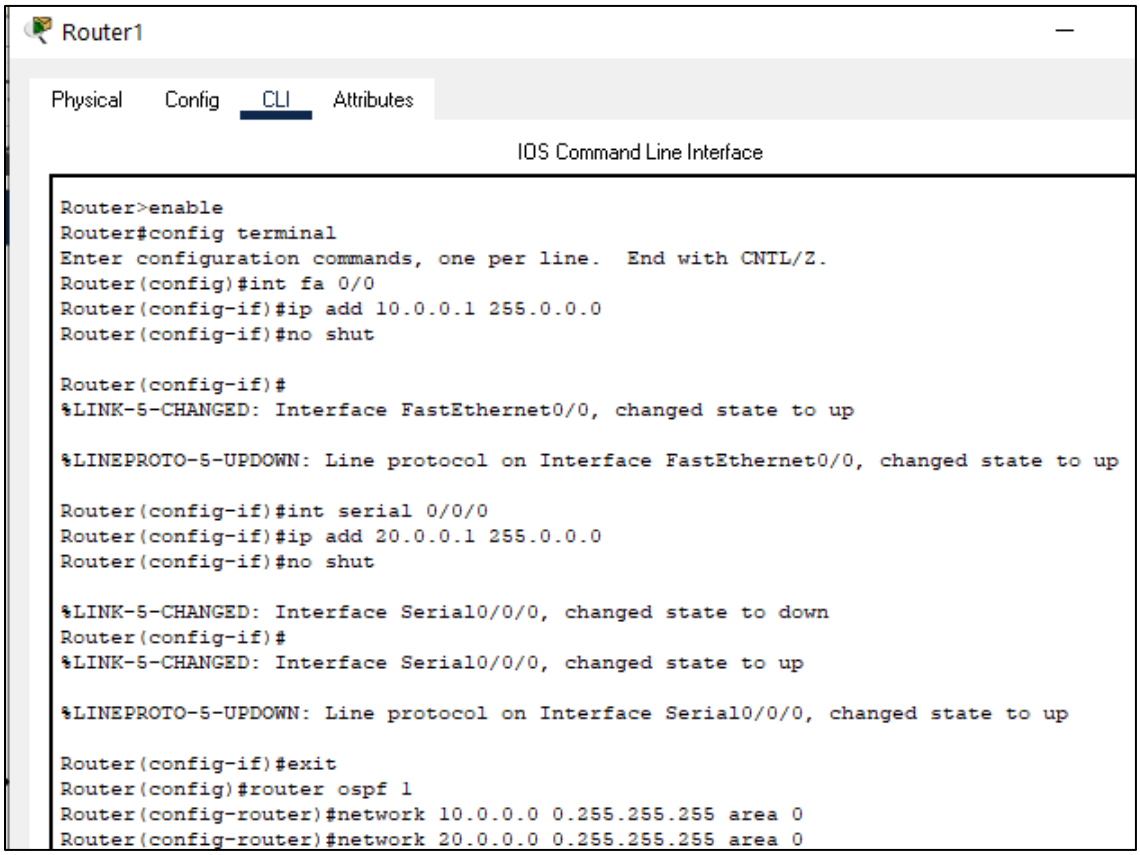
1. Open Cisco Packet Tracer and simulate the sample topologies for OSPF.
2. Perform Necessary Operation on Switch to create and configure OSPF.
3. Check the connectivity between the devices.

Steps:

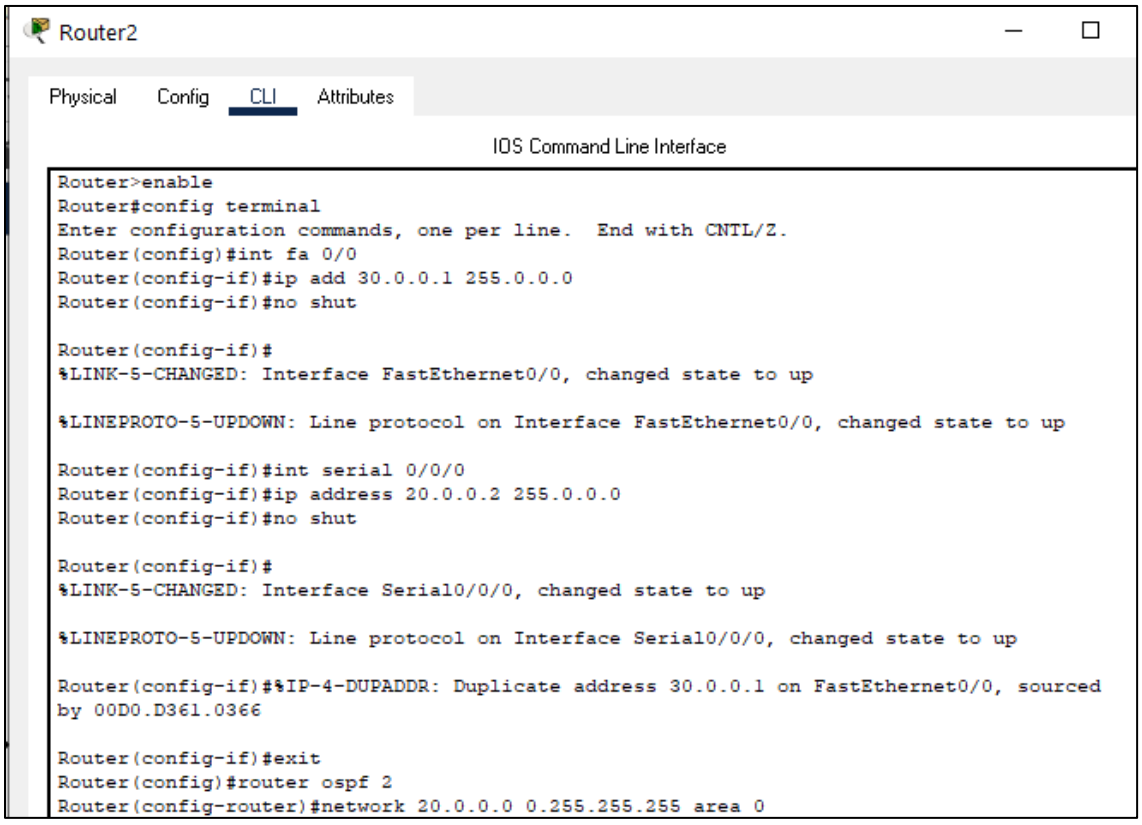
1. **Topology Setup:** Create your network topology in Packet Tracer with routers and connections.
2. **Router Configuration:** Assign IP addresses and default gateways to the interfaces of your routers.
3. **OSPF Configuration:** Activate OSPF routing on your routers and set unique process ID for each OSPF instance. Also, define the network and area for OSPF.
4. **Implement OSPF Single Area Network:** Configure the implementation of OSPF in the Network.
5. **Verify OSPF Implementation:** Use show commands to verify whether the implementation of OSPF is done correctly or not.

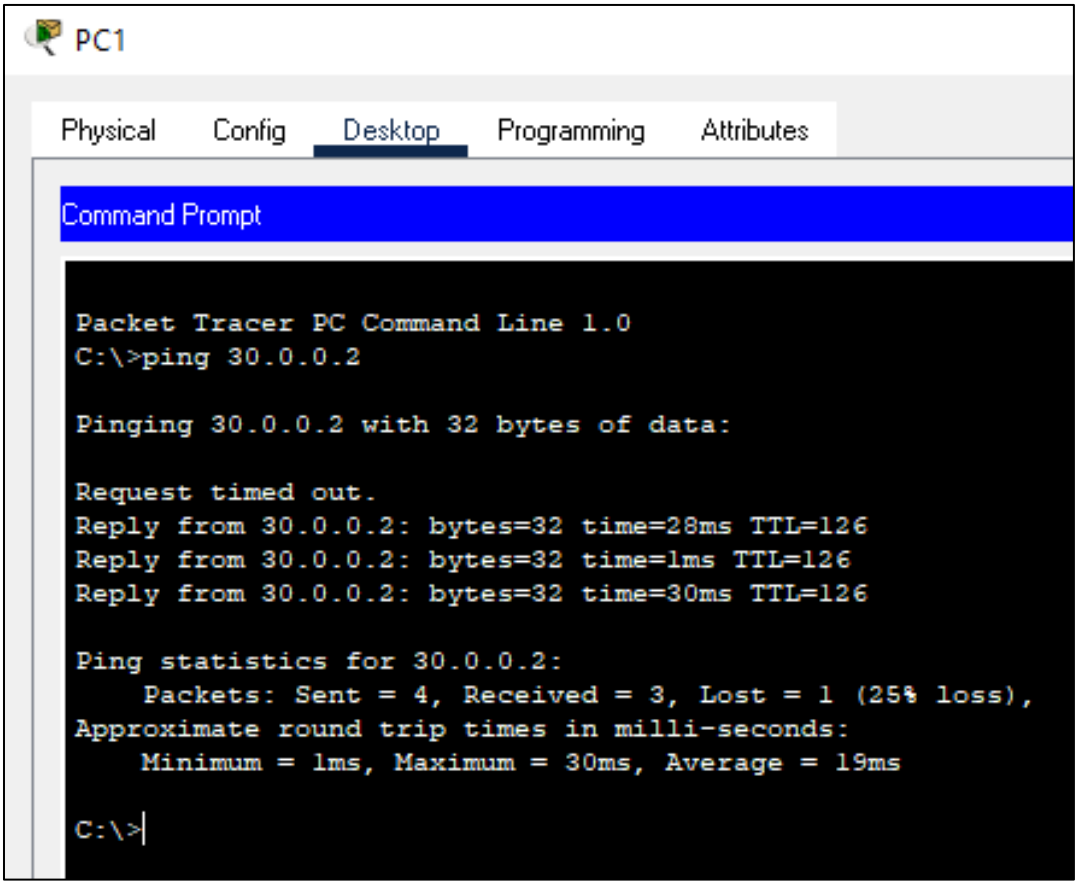
Output:






```
Router(config-if)#exit
Router(config)#router ospf 1
Router(config-router)#network 10.0.0.0 0.255.255.255 area 0
Router(config-router)#network 20.0.0.0 0.255.255.255 area 0
Router(config-router)#
00:07:18: %OSPF-5-ADJCHG: Process 1, Nbr 30.0.0.1 on Serial0/0/0 from LOADING to
Loading Done

Router(config-router)#exit
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console

Router#show ip ospf neighbor


Neighbor ID     Pri   State           Dead Time   Address         Interface
30.0.0.1          0   FULL/  -        00:00:37    20.0.0.2        Serial0/0/0
Router#show ip route ospf
O    30.0.0.0 [110/65] via 20.0.0.2, 00:01:26, Serial0/0/0

Router#
```


Router2
Physical
Config
CLI
Attributes
IOS Command Line Interface
Router>enable
Router#config terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int fa 0/0
Router(config-if)#ip add 30.0.0.1 255.0.0.0
Router(config-if)#no shut
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
Router(config-if)#int serial 0/0/0
Router(config-if)#ip address 20.0.0.2 255.0.0.0
Router(config-if)#no shut
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up
Router(config-if)#%IP-4-DUPADDR: Duplicate address 30.0.0.1 on FastEthernet0/0, sourced by 00D0.D361.0366
Router(config-if)#exit
Router(config)#router ospf 2
Router(config-router)#network 20.0.0.0 0.255.255.255 area 0


PC1
Physical
Config
Desktop
Programming
Attributes
Command Prompt
Packet Tracer PC Command Line 1.0
C:\>ping 30.0.0.2
Pinging 30.0.0.2 with 32 bytes of data:
Request timed out.
Reply from 30.0.0.2: bytes=32 time=28ms TTL=126
Reply from 30.0.0.2: bytes=32 time=1ms TTL=126
Reply from 30.0.0.2: bytes=32 time=30ms TTL=126
Ping statistics for 30.0.0.2:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 1ms, Maximum = 30ms, Average = 19ms
C:\>

Observation & Learning

In the OSPF implementation experiment in Cisco Packet Tracer, we observed that OSPF efficiently established dynamic routing, exchanged routing information, and maintained network connectivity. We learned the importance of consistent process IDs and accurate network address specification for OSPF. This experiment highlighted OSPF's role in automating routing processes, ensuring efficient data transmission, and adapting to network changes, underscoring its significance in creating robust networks.

Conclusion

In summary, the OSPF implementation experiment in Cisco Packet Tracer showcased OSPF's efficiency in dynamic routing, reinforcing the significance of process ID consistency and accurate network address configuration. It underscored OSPF's crucial role in building resilient and adaptable network infrastructures.

EXPERIMENT 7

Aim

Introduction to Socket Programming- Design and Implement client-server elements of a few network applications e.g., Echo client and server, Time client and server, Online Quiz and Buzzer Application, etc.

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

Socket programming is a fundamental technique for establishing communication between two or more processes, either on the same machine or over a network. Sockets serve as the endpoints for this communication, allowing data to be transmitted and received. These sockets can be created and utilized in various programming languages, with Java and Python being commonly used choices for implementing network applications.

Key concepts in socket programming include:

1. **Socket Types:** There are two primary types of sockets: stream sockets (e.g., TCP) and datagram sockets (e.g., UDP). Stream sockets provide reliable, connection-oriented communication, while datagram sockets offer connectionless, unreliable communication. The choice of socket type depends on the requirements of the application.
2. **IP Addresses and Port Numbers:** In socket programming, each socket is associated with an IP address and a port number. The IP address identifies the host (machine) in the network, and the port number specifies the endpoint within that host. This combination is used to establish connections and facilitate data exchange.
3. **Server-Client Model:** Socket programming typically follows a client-server architecture, where one entity (the server) listens for incoming connections, while other entities (clients) connect to the server to exchange data. This model is used in various network applications, including web servers, chat applications, and online games.
4. **Socket Operations:**
    - **Socket Creation:** Sockets are created using programming language-specific functions or classes. The creation process involves specifying the socket type (stream or datagram) and the communication protocol (e.g., TCP, UDP).
    - **Binding:** Servers must bind their sockets to a specific IP address and port number, enabling clients to connect to the server at that address and port.
    - **Listening:** Server sockets listen for incoming connection requests from clients. This listening state allows multiple clients to connect to a single server.

- **Connection Establishment:** Clients initiate connections to the server by specifying the server's IP address and port number. Once a connection is established, data can be exchanged bidirectionally.
- **Sending and Receiving Data:** Both clients and servers can send and receive data through their sockets. Stream sockets offer reliable, ordered data transmission, while datagram sockets provide a connectionless mode for sending data.
- **Closing Sockets:** Properly closing sockets is essential to release network resources and terminate communication. Failing to close sockets can lead to resource leaks and other issues.

Procedure

1. Write Simple Client Server Program using Java/Python Programming Languag
2. Execute the program using appropriate compiler.
3. Verify the working of the program.

Steps

1. Echo Server-Client Implementation

EchoServer.java

```
import java.io.*;
import java.net.*;

public class EchoServer {
    public static void main(String[] args) throws Exception {
        ServerSocket serverSocket = new ServerSocket(12345);
        System.out.println("Echo Server is running and listening on port 12345...");

        while (true) {
            Socket clientSocket = serverSocket.accept();
            System.out.println("Client connected: " + clientSocket.getInetAddress());

            BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
            PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);

            String message;
            while ((message = in.readLine()) != null) {
                System.out.println("Received from client: " + message);
                out.println("Server Echo: " + message);
            }

            clientSocket.close();
            System.out.println("Client disconnected.");
        }
    }
}
```

EchoClient.java

```
import java.io.*;
import java.net.*;

public class EchoClient {
   public static void main(String[] args) throws Exception {
      Socket socket = new Socket("127.0.0.1", 12345);
      BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
      PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

      String message = "Hello, server!";
      out.println(message)
      String response = in.readLine();
      System.out.println("Received from server: " + response);

      socket.close();

   }
}
```

2. Time Server-Client Implementation

TimeServer.java

```
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class TimeServer {
   public static void main(String[] args) throws Exception {

      ServerSocket serverSocket = new ServerSocket(12345);
      System.out.println("Time Server is running and listening on port 12345...");

      while (true) {
         Socket clientSocket = serverSocket.accept();
         System.out.println("Client connected: " + clientSocket.getInetAddress());

         PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
         out.println(new Date().toString());

         clientSocket.close();
         System.out.println("Client disconnected.");
      }
   }
}
```

TimeClient.java

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.Socket;

public class TimeClient {
    public static void main(String[] args) throws Exception {

        Socket socket = new Socket("127.0.0.1", 12345);
        BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

        String response = in.readLine();
        System.out.println("Server time: " + response);

        socket.close();

    }
}
```

3. Online Quiz Buffer Implementation

QuizServer.java

```
import java.io.*;
import java.net.*;
import java.util.*;

public class QuizServer {

    private Map<Integer, String> questions;
    private Map<Integer, String> answers;

    public QuizServer() {
        questions = new HashMap<>();
        answers = new HashMap<>();
        questions.put(1, "What is the capital of France?");
        answers.put(1, "Paris");

        questions.put(2, "Which planet is known as the Red Planet?");
        answers.put(2, "Mars");

        questions.put(3, "How many continents are there on Earth?");
        answers.put(3, "7");
    }
```

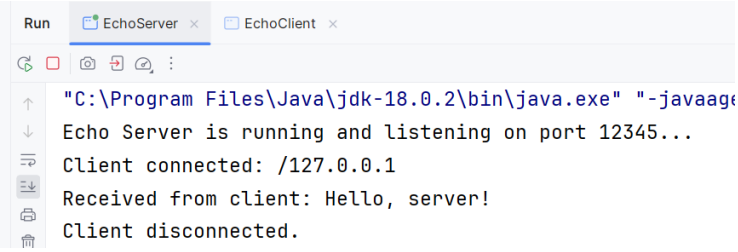
```
    public void start() throws IOException {
        ServerSocket serverSocket = new ServerSocket(12345);

        while (true) {
            Socket clientSocket = serverSocket.accept();

            BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
            PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);

            // Start the quiz
            for (int questionNumber = 1; questionNumber <= 3; questionNumber++) {
                // Send the question to the client
                out.println("Question " + questionNumber + ": " + questions.get(questionNumber));

                // Receive the client's answer
                String answer = in.readLine();

                // Check if the answer is correct
                if (answer.equals(answers.get(questionNumber))) {
                    // The answer is correct
                    out.println("Correct!");
                } else {
                    // The answer is incorrect
                    out.println("Incorrect. The correct answer is: " + answers.get(questionNumber));
                }
            }

            // The quiz is finished
            out.println("Quiz finished!");

            // Close the socket
            clientSocket.close();
        }
    }

    public static void main(String[] args) throws IOException {
        QuizServer quizServer = new QuizServer();
        quizServer.start();
    }
}
```

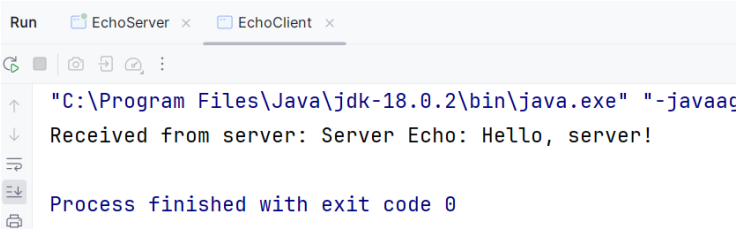
QuizClient.java

```
import java.io.*;
import java.net.*;

public class QuizClient {

   public static void main(String[] args) throws Exception {
      Socket socket = new Socket("localhost", 12345);

      BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
      PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

      // Start the quiz
      while (true) {
         // Receive the question from the server
         String question = in.readLine();

         // Print the question to the console
         System.out.println(question);

         // Get the user's answer
         BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
         String answer = reader.readLine();

         // Send the answer to the server
         out.println(answer);

         // Receive the result from the server
         String result = in.readLine();

         // Display the result to the user
         System.out.println(result);

         // Check if the quiz is finished
         if (result.equals("Quiz finished!")) {
            break;
         }
      }

      // Close the socket
      socket.close();
   }
}
```

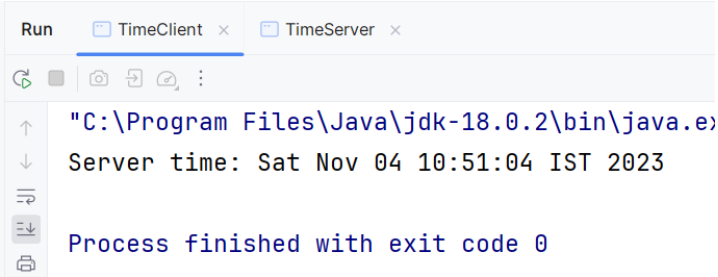
Output

Echo Server-Client

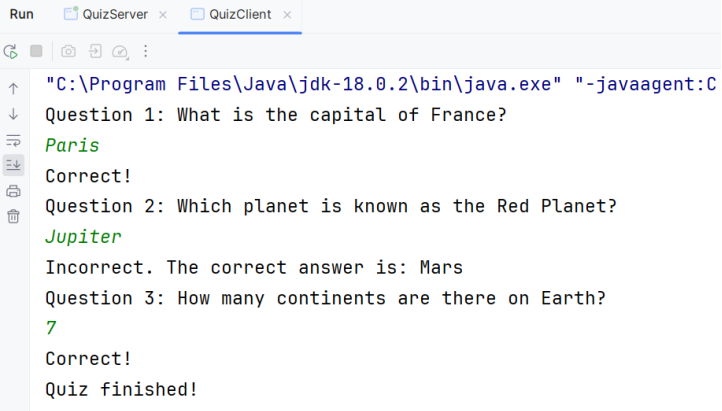




Time Server-Client



Quiz Buffer Server-Client



Observation & Learning

During the experiment, participants observed and learned the following key points:

- How to create and use socket objects for communication.
- The basics of establishing a connection between a client and a server.
- How data is exchanged between the client and server.
- The importance of port numbers and IP addresses in socket programming.
- How to implement specific functionalities for different network applications, such as Echo, Time, Quiz, or Buzzer.

Conclusion

In conclusion, this practical has demonstrated the implementation of various client-server applications using Java socket programming. These applications showcase the basic principles of client-server communication, including network protocols, socket programming, and error handling. The practical has also provided insights into designing scalable and efficient server applications.

Questions

1. **What is a Socket?**

   A socket is a software endpoint used for communication between processes over a network. It allows data to be sent and received over a network connection.

2. **Which socket is used for communication between the client and server?**

   Both the client and server use sockets to establish communication. The client and server each create a socket to send and receive data.

3. **What are the different operations supported on sockets?**

   Sockets support various operations, including:

   - Socket creation
   - Binding a socket to an IP address and port number
   - Listening for incoming connections (for servers)
   - Establishing connections (for clients)
   - Sending and receiving data
   - Closing the socket connection

EXPERIMENT 8

Aim

Write a program for interactive application using UDP socket.

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

1. Introduction to UDP

UDP (User Datagram Protocol) is a connectionless, lightweight transport layer protocol in computer networking. It is part of the Internet Protocol (IP) suite and is designed for fast, low-overhead data transmission. UDP is known for its simplicity and speed, making it ideal for real-time and time-sensitive applications such as streaming media, online gaming, and VoIP (Voice over Internet Protocol). Unlike TCP, UDP does not guarantee the delivery or order of data packets, making it suitable for scenarios where minor data loss is acceptable in exchange for reduced latency.

2. Socket Programming

Socket programming involves the use of APIs and libraries to create, send, receive, and manage data over a network using sockets. Sockets can be created and configured to use either TCP or UDP for communication.

Two primary socket types exist: TCP sockets, ensuring reliable, connection-oriented communication with data sent in a stream, and UDP sockets, enabling fast, connectionless data transfer through discrete packets. TCP is used in applications that require data integrity and order, like web browsing and file transfer, while UDP is ideal for real-time, low-latency tasks such as multimedia streaming and online gaming.

3. UDP Server-Client Interaction:

- **Server:** The server program initializes a UDP socket, binds it to a specific port and IP address, and waits for incoming data from clients. Upon receiving data, the server processes the information and may send a response back to the client.
- **Client:** The client program creates a UDP socket, specifies the server's address and port, and sends data to the server. It can also receive responses from the server.

Procedure

1. **Server Implementation:**

- Create a UDP socket using the appropriate programming language Java.
- Bind the socket to a specific IP address and port to listen for incoming data.
- Receive data from clients using receive().
- Process the received data as needed.
- Send a response back to the client using send().

2. **Client Implementation:**

- Create a UDP socket.
- Specify the server's IP address and port.
- Send data to the server using send().
- Receive responses from the server using receive().
- Process and display the received data.

Steps

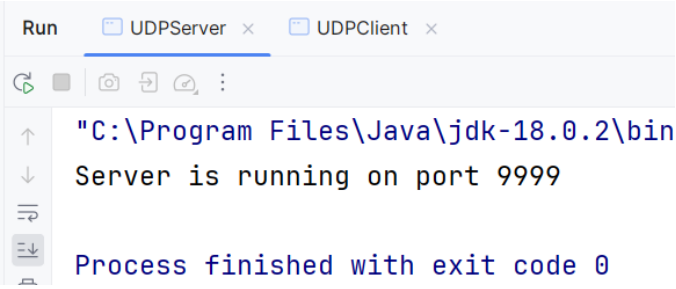
1. Server Implementation

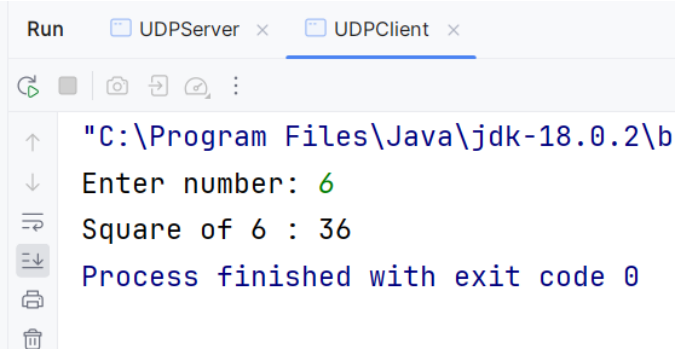
```
import java.net.*;
public class UDPServer {
   public static void main(String[] args) throws Exception{
      int port = 9999;
      DatagramSocket ds = new DatagramSocket(port);

      // Getting Data from client
      byte[] b1 = new byte[1024];
      DatagramPacket dp1 = new DatagramPacket(b1, b1.length);
      ds.receive(dp1);

      // Performing square operation on data received
      String str = new String(dp1.getData(), 0, dp1.getLength());
      int num = Integer.parseInt(str.trim());
      int result = num*num;

      // Sending data back to client
      byte[] b2 = String.valueOf(result).getBytes();
      InetAddress ip = InetAddress.getLocalHost();
      DatagramPacket dp2 = new DatagramPacket(b2, b2.length, ip, dp1.getPort());
      ds.send(dp2);
   }
}
```

2. Client Implementation

```
import java.net.*;
import java.utli.Scanner;
public class UDPClient {
    public static void main(String[] args) throws Exception {
        int port = 9999;
        DatagramSocket ds = new DatagramSocket();
        Scanner sc = new Scanner(System.in);  // Create a Scanner object
        System.out.print("Enter number: ");

        int num = sc.nextInt();
        byte[] b1 = String.valueOf(num).getBytes();
        // Sending Data to Server
        InetAddress ip = InetAddress.getLocalHost();
        DatagramPacket dp1 = new DatagramPacket(b1, b1.length, ip, port);
        ds.send(dp1);

        // Receiving Data from Server
        byte[] b2 = new byte[1024];
        DatagramPacket dp2 = new DatagramPacket(b2, b2.length);
        ds.receive(dp2);

        String str = new String(dp2.getData(), 0, dp2.getLength());
        System.out.println("Result: " + str);

    }
}
```

Output





Observation & Learning

- UDP sockets provide a lightweight and low-latency communication method suitable for real-time applications.
- The server program creates a UDP socket, binds it to a specific IP address and port, and listens for incoming data from clients.
- The client program establishes a UDP socket, specifies the server's address and port, and sends data to the server.
- UDP does not guarantee the order of delivery or data reliability, but it excels in scenarios where speed and minimal overhead are critical.
- The interactive application can include features like sending and receiving messages, commands, or data between the client and server.

Conclusion

In this experiment, we successfully implemented an interactive application using UDP sockets. UDP's characteristics, such as low latency and connectionless communication, make it suitable for real-time applications like chat programs, online games, or remote control. The client and server components worked together to send and receive data over the network. UDP socket programming offers flexibility and efficiency for various interactive applications, allowing for faster data exchange and responsiveness.

EXPERIMENT 9

Aim

Configure DHCP and SMTP in a small LAN

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

1. **Dynamic Host Configuration Protocol (DHCP)**
   DHCP is a network protocol used for the automatic configuration of IP addresses and other network parameters in a local network. In a Cisco Packet Tracer environment, DHCP plays a crucial role in simplifying IP address management. It allows a DHCP server to dynamically allocate IP addresses to client devices as they join the network. Key points to consider include:

   - **IP Address Assignment**: DHCP assigns IP addresses to devices within a defined range. This range is often referred to as a DHCP pool.
   - **Lease Duration**: Each IP address assignment comes with a lease duration, determining how long a device can use the assigned address. When the lease expires, the device may request a renewal.
   - **Subnet Masks and Gateways**: In addition to IP addresses, DHCP can provide subnet masks and default gateways, streamlining network configuration.
   - **Conflict Resolution**: DHCP servers detect and resolve IP address conflicts to prevent network disruptions.

2. **Simple Mail Transfer Protocol (SMTP):** SMTP is a protocol used for sending and receiving email messages. In the context of Cisco Packet Tracer, SMTP is employed to simulate email communication. Key aspects to consider include:

   - **SMTP Server Configuration**: To enable SMTP communication, an SMTP server is configured on a specific PC. This server handles the sending and receiving of email messages.
   - **Email Clients**: Other PCs in the network are configured as email clients, and they use the SMTP server for sending outgoing emails. The server routes these messages to their destinations.
   - **Email Accounts**: Proper configuration of email accounts on the email clients is essential for sending and receiving messages. This typically includes settings for incoming and outgoing email servers, usernames, and passwords.
   - **Email Routing**: SMTP is responsible for routing email messages from the sender to the recipient's mailbox. It communicates with other SMTP servers on the internet to relay messages.

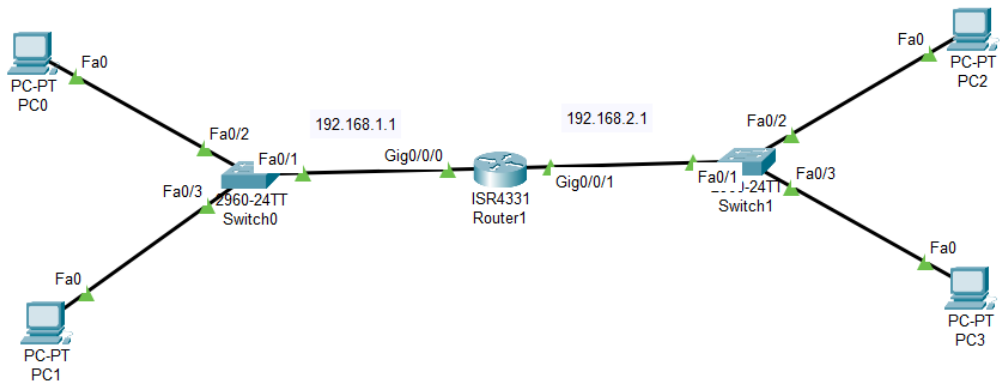
Understanding DHCP and SMTP is fundamental in network management and communication within Cisco Packet Tracer simulations, as these protocols automate IP address assignments and email transmission, ensuring efficient and reliable networking and messaging services.

Procedure

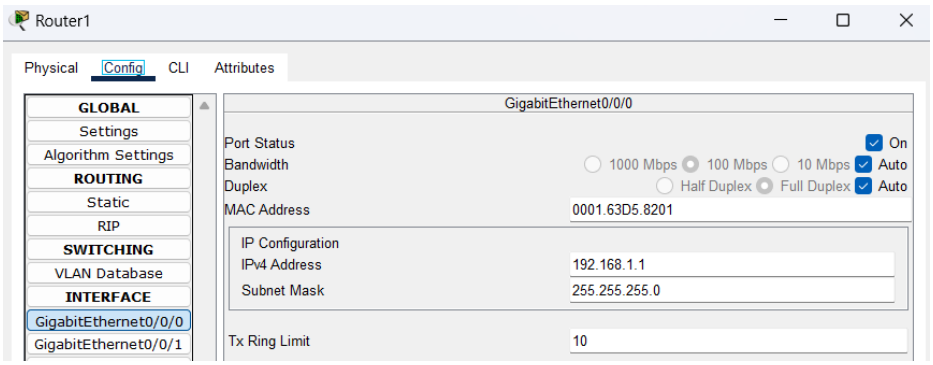
1. **Setting Up DHCP in Cisco Packet Tracer:**
   - Launch Cisco Packet Tracer and create a new project.
   - Add a router, a switch, and multiple PCs to the workspace.
   - Configure the router and switch as needed to create a LAN.
   - Set up a DHCP server on one of the PCs within the LAN.
   - Configure the DHCP server to define the IP address range and lease duration.
   - Connect client PCs to the switch and set them to obtain IP addresses automatically.
   - Observe the IP address assignment process and network connectivity within the simulation.
2. **Configuring SMTP in Cisco Packet Tracer:**
   - Add a server (another PC) to the workspace for hosting the SMTP server.
   - Configure the server as an SMTP server and set up email accounts.
   - Configure email clients on other PCs to use this SMTP server for sending emails.
   - Send test emails between clients to observe the SMTP communication within the Cisco Packet Tracer simulation.

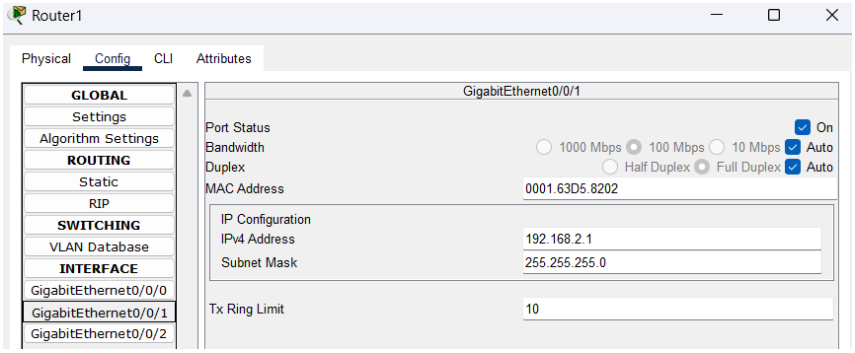
Output

1. **DHCP**



Router Configuration

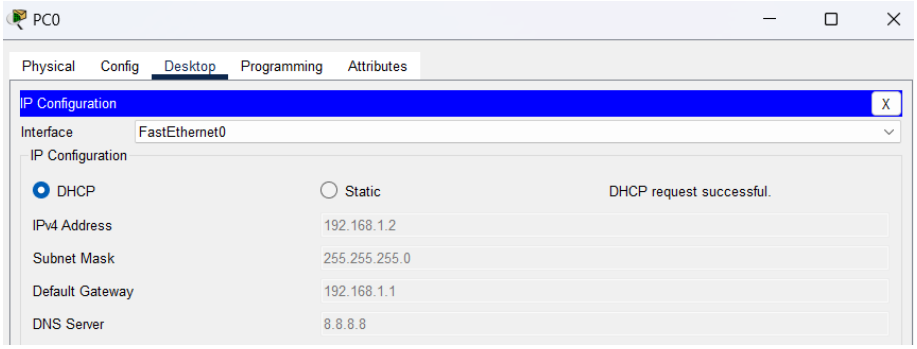


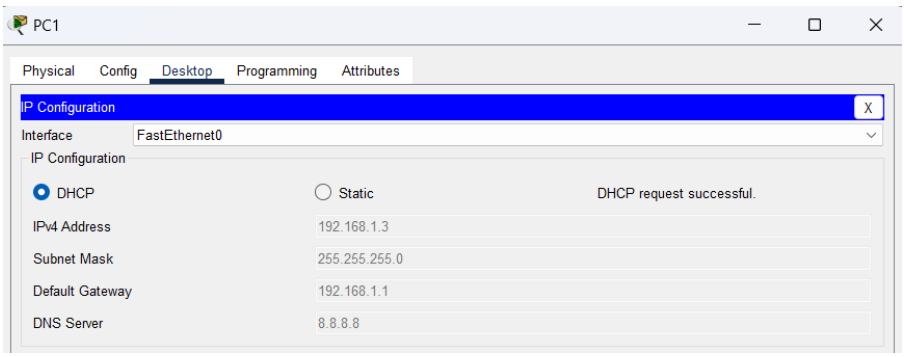


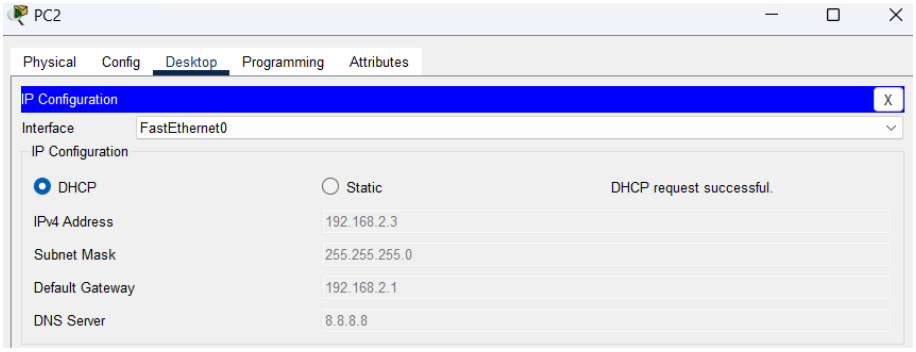
DHCP Configuration

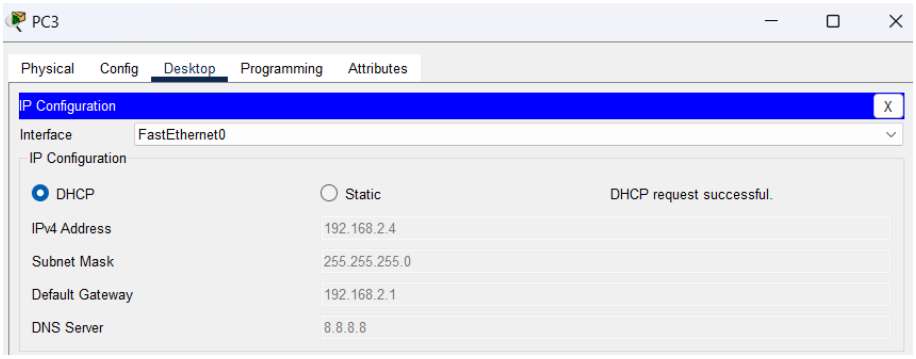
```
dhcp-server#config t
Enter configuration commands, one per line.  End with CNTL/Z.
dhcp-server(config)#ip dhcp excluded-address 192.168.1.1
dhcp-server(config)#ip dhcp excluded-address 192.168.2.1
dhcp-server(config)#ip dhcp pool 192.168.1.1
dhcp-server(dhcp-config)#network 192.168.1.0 255.255.255.0
dhcp-server(dhcp-config)#default-router %DHCPD-4-PING_CONFLICT:
DHCP address conflict:  server pinge
dhcp-server(dhcp-config)#
dhcp-server(dhcp-config)#network 192.168.1.0 255.255.255.0
dhcp-server(dhcp-config)#de
dhcp-server(dhcp-config)#default-router 192.168.1.1
dhcp-server(dhcp-config)#dns
dhcp-server(dhcp-config)#dns-server 8.8.8.8
dhcp-server(dhcp-config)#exit
dhcp-server(config)#ip
dhcp-server(config)#ip
dhcp-server(config)#ip dh
dhcp-server(config)#ip dhcp po
dhcp-server(config)#ip dhcp pool 192.268.2.1
dhcp-server(dhcp-config)#network 192.168.2.0 255.255.255.0
dhcp-server(dhcp-config)#default
dhcp-server(dhcp-config)#default-router 192.168.2.1
dhcp-server(dhcp-config)#dns
dhcp-server(dhcp-config)#dns-server 8.8.8.8
dhcp-server(dhcp-config)#
```

IP Address assigned to PCs



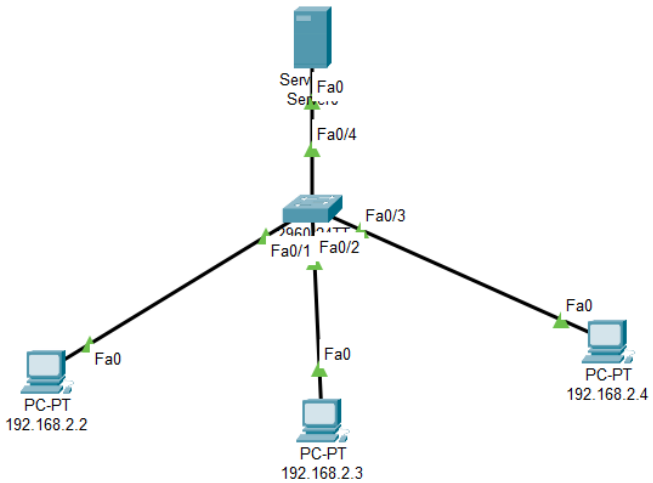


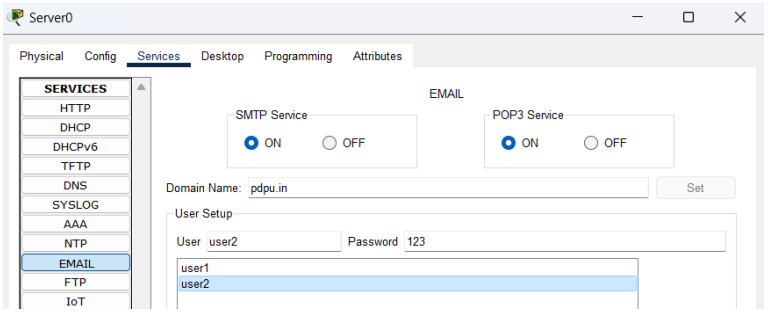


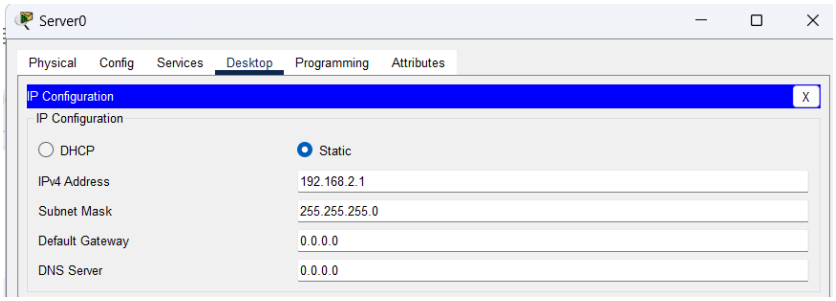


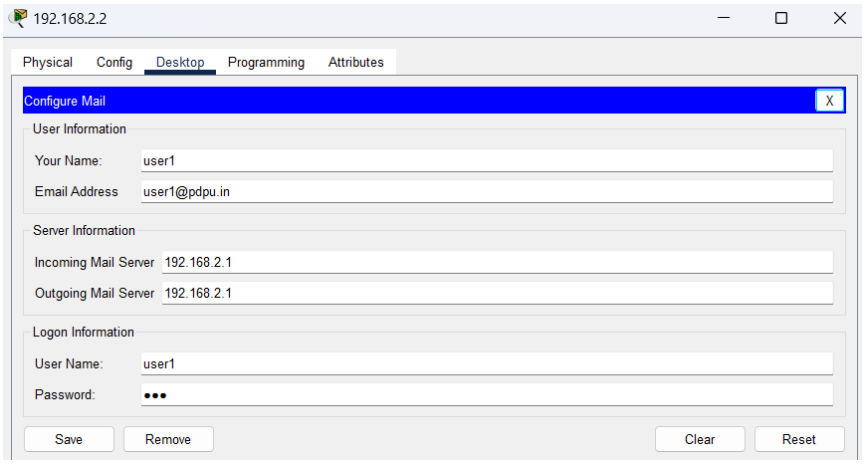

```
14065 51.358914    192.168.0.102   23.200.79.138   TLSv1.3  1020 Application Data
14066 51.372515    23.200.79.138   192.168.0.102   TCP        54 443 → 50121 [ACK] Seq=77338 Ack=6151 Win=64128 Len=0
14067 51.372515    23.200.79.138   192.168.0.102   TCP        54 443 → 50121 [ACK] Seq=77338 Ack=7591 Win=64128 Len=0
14068 51.372604    23.200.79.138   192.168.0.102   TCP        54 443 → 50121 [ACK] Seq=77338 Ack=8557 Win=64128 Len=0
14069 51.375044    192.168.0.102   23.200.79.138   TLSv1.3   175 Application Data
14070 51.391967    23.200.79.138   192.168.0.102   TCP        54 443 → 50120 [ACK] Seq=5205 Ack=1530 Win=64128 Len=0
14071 51.403821    23.200.79.138   192.168.0.102   TLSv1.3   440 Application Data
14072 51.403821    23.200.79.138   192.168.0.102   TLSv1.3    85 Application Data
14073 51.403899    192.168.0.102   23.200.79.138   TCP        54 50120 → 443 [ACK] Seq=1530 Ack=5622 Win=132352 Len=0
14074 51.485004    192.168.0.102   192.168.0.1     DHCP      346 DHCP Request  - Transaction ID 0xdac728ae
14075 51.496644    192.168.0.1     192.168.0.102   DHCP      590 DHCP ACK      - Transaction ID 0xdac728ae
14076 51.507513    fe80::4039:285d:a0f… ff02::16   ICMPv6     90 Multicast Listener Report Message v2
14077 51.507629    192.168.0.102   224.0.0.22      IGMPv3     54 Membership Report / Leave group 224.0.0.252
14078 51.514759    192.168.0.102   172.253.118.188 TCP        55 [TCP Keep-Alive] 49823 → 5228 [ACK] Seq=1 Ack=1 Win=510 Len=1
14079 51.514770    192.168.0.102   142.250.4.188   TCP        55 [TCP Keep-Alive] 49825 → 5228 [ACK] Seq=1 Ack=1 Win=512 Len=1
14080 51.529895    192.168.0.102   224.0.0.251     MDNS       75 Standard query 0x0000 ANY Smitpatel.local, "QM" question
14081 51.530075    192.168.0.102   224.0.0.22      IGMPv3     54 Membership Report / Join group 224.0.0.252 for any sources
14082 51.530171    fe80::4039:285d:a0f… ff02::16   ICMPv6     90 Multicast Listener Report Message v2
14083 51.530729    192.168.0.102   224.0.0.251     MDNS      113 Standard query response 0x0000 AAAA fe80::4039:285d:a0f3:37ee A 192.168.0.102
```

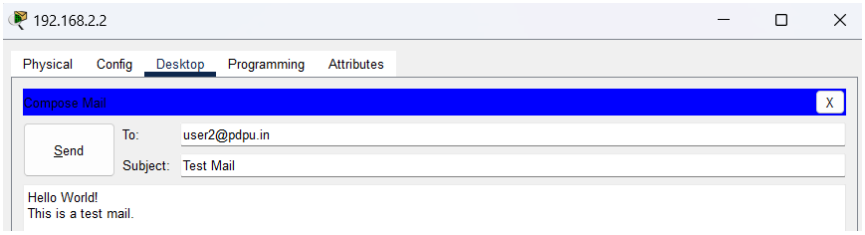
2. SMTP

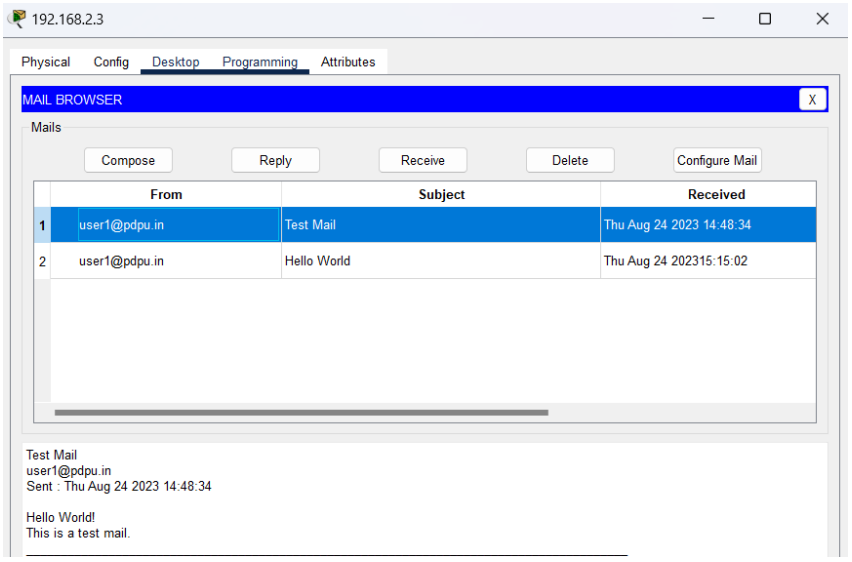


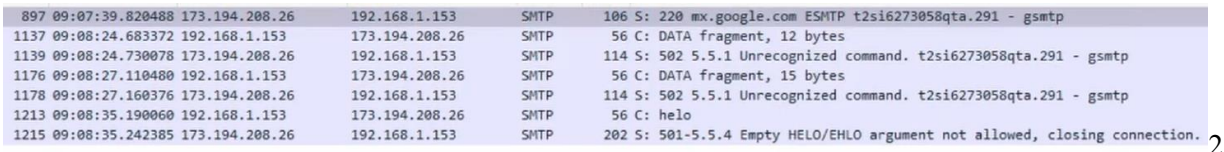

Server0
Physical Config Services Desktop Programming Attributes
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
IoT
EMAIL
SMTP Service
ON
OFF
POP3 Service
ON
OFF
Domain Name: pdpu.in
Set
User Setup
User user2
Password 123
user1
user2


Server0
Physical Config Services Desktop Programming Attributes
IP Configuration
X
IP Configuration
DHCP
Static
IPv4 Address 192.168.2.1
Subnet Mask 255.255.255.0
Default Gateway 0.0.0.0
DNS Server 0.0.0.0


192.168.2.2
Physical Config Desktop Programming Attributes
Configure Mail
X
User Information
Your Name: user1
Email Address user1@pdpu.in
Server Information
Incoming Mail Server 192.168.2.1
Outgoing Mail Server 192.168.2.1
Logon Information
User Name: user1
Password: •••
Save
Remove
Clear
Reset


192.168.2.2
Physical Config Desktop Programming Attributes
Compose Mail
X
Send
To: user2@pdpu.in
Subject: Test Mail
Hello World!
This is a test mail.


192.168.2.3
Physical Config Desktop Programming Attributes
MAIL BROWSER
X
Mails
Compose
Reply
Receive
Delete
Configure Mail
From
Subject
Received
1 user1@pdpu.in Test Mail Thu Aug 24 2023 14:48:34
2 user1@pdpu.in Hello World Thu Aug 24 202315:15:02
Test Mail
user1@pdpu.in
Sent : Thu Aug 24 2023 14:48:34
Hello World!
This is a test mail.



897	09:07:39.820488	173.194.208.26	192.168.1.153	SMTP	106	S: 220 mx.google.com ESMTP t2si6273058qta.291 - gsmtp
1137	09:08:24.683372	192.168.1.153	173.194.208.26	SMTP	56	C: DATA fragment, 12 bytes
1139	09:08:24.730078	173.194.208.26	192.168.1.153	SMTP	114	S: 502 5.5.1 Unrecognized command. t2si6273058qta.291 - gsmtp
1176	09:08:27.110480	192.168.1.153	173.194.208.26	SMTP	56	C: DATA fragment, 15 bytes
1178	09:08:27.160376	173.194.208.26	192.168.1.153	SMTP	114	S: 502 5.5.1 Unrecognized command. t2si6273058qta.291 - gsmtp
1213	09:08:35.190060	192.168.1.153	173.194.208.26	SMTP	56	C: helo
1215	09:08:35.242385	173.194.208.26	192.168.1.153	SMTP	202	S: 501-5.5.4 Empty HELO/EHLO argument not allowed, closing connection.

2

Observation & Learning

- DHCP successfully assigned IP addresses, subnet masks, and gateways to client PCs, demonstrating efficient network parameter management within Cisco Packet Tracer.
- SMTP facilitated seamless email communication as messages were routed through the SMTP server, emphasizing its role in relaying emails.
- Network connectivity was established swiftly, allowing clients with dynamically assigned IP addresses to access the internet and interact within the LAN.
- The experiment exhibited rapid response times for DHCP lease assignments and email message delivery, ensuring prompt network and communication services.
- Error handling mechanisms for DHCP and SMTP were observed, showcasing automatic resolution of issues like IP conflicts and email address validation, enhancing network reliability.

Conclusion

In this experiment, we successfully configured DHCP and SMTP services within a small LAN simulation. DHCP simplified IP address management, and SMTP facilitated the transmission of email messages. These services are essential for efficient network management and communication in real-world networks as well as simulations.

Questions

1. **What are the different ports used by the DHCP and SMTP within the Cisco Packet Tracer environment?2**

   In Cisco Packet Tracer, the port numbers used by DHCP and SMTP are the same as in real-world scenarios. DHCP uses **UDP port 67** for the server and **UDP port 68** for the client. SMTP uses **TCP port 25**.

2. **What are the benefits of using DHCP services in Cisco Packet Tracer simulations and real networks?**

   - DHCP simplifies IP address management, **reducing the risk of IP address conflicts** in Cisco Packet Tracer simulations and real networks.
   - It **automates** the configuration of network parameters, such as subnet masks and gateways.
   - It makes it **easier to add and remove** devices from the network without manual IP configuration.

3. **What is the role of Active Directory in the DHCP context within Cisco Packet Tracer?**

   In Cisco Packet Tracer simulations, Active Directory may not be available. However, in real-world networks, Active Directory can be used to **store and manage DHCP server configuration** information, providing centralized DHCP management, security, and access control features.

EXPERIMENT 10

Aim

Study of Wireshark and understand its functionality

Prerequisite

Nil

Outcome

To impart knowledge of Computer Networking Technology

Theory

Wireshark is a popular and powerful open-source network protocol analyser that allows users to capture, inspect, and analyse data traffic on a computer network. It is widely used by network administrators, security professionals, and developers to troubleshoot network issues, monitor network performance, and identify potential security vulnerabilities.

Wireshark boasts support for a wide array of network protocols, including common ones like TCP, IP, UDP, HTTP, and HTTPS, as well as less common or proprietary protocols. The tool can decode and display information about each packet, allowing users to understand the contents of network traffic and identify potential issues.

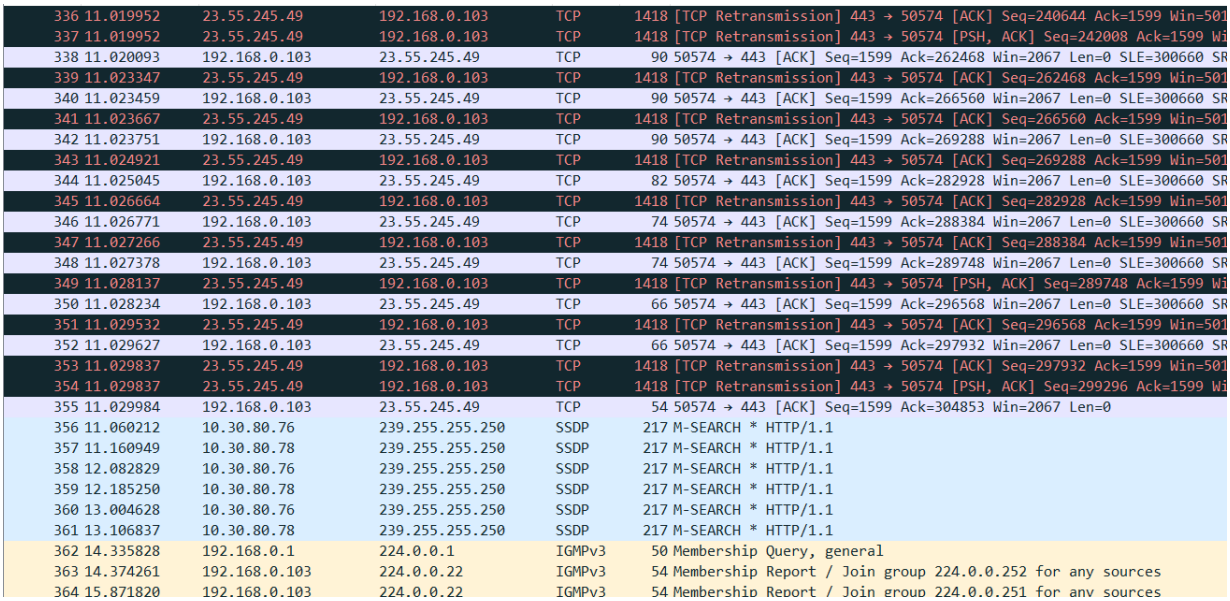
Here are some key aspects of Wireshark:

- Packet Capture
- Protocol Support
- Live Capture and Offline Analysis
- Display Filters
- Packet Inspection
- Statistics and Graphs
- Colour Coding
- Extensibility
- Cross-Platform
- Community and Support
- Security Analysis
- Educational Tool

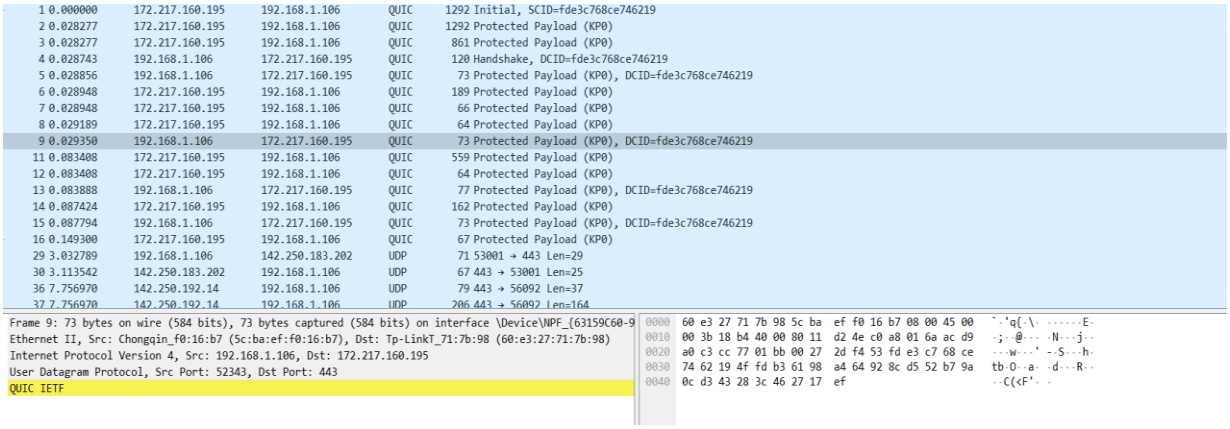
Procedure

1. Install the Wireshark and integrate it with network simulator
2. Analyse the traffic using Wireshark

Output



336	11.019952	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [ACK] Seq=240644 Ack=1599 Win=501
337	11.019952	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [PSH, ACK] Seq=242008 Ack=1599 Wi
338	11.020093	192.168.0.103	23.55.245.49	TCP	90 50574 → 443 [ACK] Seq=1599 Ack=262468 Win=2067 Len=0 SLE=300660 SR
339	11.023347	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [ACK] Seq=262468 Ack=1599 Win=501
340	11.023459	192.168.0.103	23.55.245.49	TCP	90 50574 → 443 [ACK] Seq=1599 Ack=266560 Win=2067 Len=0 SLE=300660 SR
341	11.023667	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [ACK] Seq=266560 Ack=1599 Win=501
342	11.023751	192.168.0.103	23.55.245.49	TCP	90 50574 → 443 [ACK] Seq=1599 Ack=269288 Win=2067 Len=0 SLE=300660 SR
343	11.024921	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [ACK] Seq=269288 Ack=1599 Win=501
344	11.025045	192.168.0.103	23.55.245.49	TCP	82 50574 → 443 [ACK] Seq=1599 Ack=282928 Win=2067 Len=0 SLE=300660 SR
345	11.026664	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [ACK] Seq=282928 Ack=1599 Win=501
346	11.026771	192.168.0.103	23.55.245.49	TCP	74 50574 → 443 [ACK] Seq=1599 Ack=288384 Win=2067 Len=0 SLE=300660 SR
347	11.027266	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [ACK] Seq=288384 Ack=1599 Win=501
348	11.027378	192.168.0.103	23.55.245.49	TCP	74 50574 → 443 [ACK] Seq=1599 Ack=289748 Win=2067 Len=0 SLE=300660 SR
349	11.028137	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [PSH, ACK] Seq=289748 Ack=1599 Wi
350	11.028234	192.168.0.103	23.55.245.49	TCP	66 50574 → 443 [ACK] Seq=1599 Ack=296568 Win=2067 Len=0 SLE=300660 SR
351	11.029532	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [ACK] Seq=296568 Ack=1599 Win=501
352	11.029627	192.168.0.103	23.55.245.49	TCP	66 50574 → 443 [ACK] Seq=1599 Ack=297932 Win=2067 Len=0 SLE=300660 SR
353	11.029837	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [ACK] Seq=297932 Ack=1599 Win=501
354	11.029837	23.55.245.49	192.168.0.103	TCP	1418 [TCP Retransmission] 443 → 50574 [PSH, ACK] Seq=299296 Ack=1599 Wi
355	11.029984	192.168.0.103	23.55.245.49	TCP	54 50574 → 443 [ACK] Seq=1599 Ack=304853 Win=2067 Len=0
356	11.060212	10.30.80.76	239.255.255.250	SSDP	217 M-SEARCH * HTTP/1.1
357	11.160949	10.30.80.78	239.255.255.250	SSDP	217 M-SEARCH * HTTP/1.1
358	12.082829	10.30.80.76	239.255.255.250	SSDP	217 M-SEARCH * HTTP/1.1
359	12.185250	10.30.80.78	239.255.255.250	SSDP	217 M-SEARCH * HTTP/1.1
360	13.004628	10.30.80.76	239.255.255.250	SSDP	217 M-SEARCH * HTTP/1.1
361	13.106837	10.30.80.78	239.255.255.250	SSDP	217 M-SEARCH * HTTP/1.1
362	14.335828	192.168.0.1	224.0.0.1	IGMPv3	50 Membership Query, general
363	14.374261	192.168.0.103	224.0.0.22	IGMPv3	54 Membership Report / Join group 224.0.0.252 for any sources
364	15.871820	192.168.0.103	224.0.0.22	IGMPv3	54 Membership Report / Join group 224.0.0.251 for any sources



1	0.000000	172.217.160.195	192.168.1.106	QUIC	1292 Initial, SCID=fde3c768ce746219
2	0.028277	172.217.160.195	192.168.1.106	QUIC	1292 Protected Payload (KP0)
3	0.028277	172.217.160.195	192.168.1.106	QUIC	861 Protected Payload (KP0)
4	0.028743	192.168.1.106	172.217.160.195	QUIC	120 Handshake, DCID=fde3c768ce746219
5	0.028856	192.168.1.106	172.217.160.195	QUIC	73 Protected Payload (KP0), DCID=fde3c768ce746219
6	0.028948	172.217.160.195	192.168.1.106	QUIC	189 Protected Payload (KP0)
7	0.028948	172.217.160.195	192.168.1.106	QUIC	66 Protected Payload (KP0)
8	0.029189	172.217.160.195	192.168.1.106	QUIC	64 Protected Payload (KP0)
9	0.029350	192.168.1.106	172.217.160.195	QUIC	73 Protected Payload (KP0), DCID=fde3c768ce746219
11	0.083408	172.217.160.195	192.168.1.106	QUIC	559 Protected Payload (KP0)
12	0.083408	172.217.160.195	192.168.1.106	QUIC	64 Protected Payload (KP0)
13	0.083888	192.168.1.106	172.217.160.195	QUIC	77 Protected Payload (KP0), DCID=fde3c768ce746219
14	0.087424	172.217.160.195	192.168.1.106	QUIC	162 Protected Payload (KP0)
15	0.087794	192.168.1.106	172.217.160.195	QUIC	73 Protected Payload (KP0), DCID=fde3c768ce746219
16	0.149300	172.217.160.195	192.168.1.106	QUIC	67 Protected Payload (KP0)
29	3.032789	192.168.1.106	142.250.183.202	UDP	71 53001 → 443 Len=29
30	3.113542	142.250.183.202	192.168.1.106	UDP	67 443 → 53001 Len=25
36	7.756970	142.250.192.14	192.168.1.106	UDP	79 443 → 56092 Len=37
37	7.756970	142.250.192.14	192.168.1.106	UDP	206 443 → 56092 Len=164

```
Frame 9: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface \Device\NPF_{63159C60-9
Ethernet II, Src: Chongqin_f0:16:b7 (5c:ba:ef:f0:16:b7), Dst: Tp-LinkT_71:7b:98 (60:e3:27:71:7b:98)
Internet Protocol Version 4, Src: 192.168.1.106, Dst: 172.217.160.195
User Datagram Protocol, Src Port: 52343, Dst Port: 443
QUIC IETF
```

```
0000  60 e3 27 71 7b 98 5c ba  ef f0 16 b7 08 00 45 00   `·'q{·\· ·····E·
0010  00 3b 18 b4 40 00 80 11  d2 4e c0 a8 01 6a ac d9   ·;··@··· ·N···j··
0020  a0 c3 cc 77 01 bb 00 27  2d f4 53 fd e3 c7 68 ce   ···w···' -·S···h·
0030  74 62 19 4f fd b3 61 98  a4 64 92 8c d5 52 b7 9a   tb·O··a· ·d···R··
0040  0c d3 43 28 3c 46 27 17  ef                       ··C(<F'· ·
```

```
Transmission Control Protocol, Src Port: 443, Dst Port: 50599, Seq: 0, Ack: 1, Len: 0
    Source Port: 443
    Destination Port: 50599
    [Stream index: 0]
    [Conversation completeness: Incomplete (2)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 3782687390
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 1544047550
    1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x012 (SYN, ACK)
    Window: 64240
    [Calculated window size: 64240]
    Checksum: 0x6348 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted, N
  > [Timestamps]
```

```
Internet Protocol Version 4, Src: 23.206.173.11, Dst: 192.168.0.103
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 52
    Identification: 0x0000 (0)
  > 010. .... = Flags: 0x2, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 49
    Protocol: TCP (6)
    Header Checksum: 0xc3db [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 23.206.173.11
    Destination Address: 192.168.0.103
```

```
0000  70 9c d1 de e7 02 ac 15  a2 da 8d 59 08 00 45 00   p······ ···Y··E·
0010  00 34 00 00 40 00 31 06  c3 db 17 ce ad 0b c0 a8   ·4··@·1· ········
0020  00 67 01 bb c5 a7 e1 77  3a 9e 5c 08 4b be 80 12   ·g·····w :·\·K···
0030  fa f0 63 48 00 00 02 04  05 54 01 01 04 02 01 03   ··cH···· ·T······
0040  03 07                                             ··
```

Observation & Learning

- **Packet Capture**: Wireshark effectively captured network traffic on the specified interface, allowing us to monitor the data packets as they passed through the network.
- **Protocol Diversity**: Wireshark identified and displayed a wide variety of network protocols, including common ones like TCP, IP, UDP, HTTP, and HTTPS. This diversity is essential for understanding the different types of traffic on the network.
- **Packet Analysis**: The tool enabled us to inspect individual packets, revealing detailed information about each one. This included source and destination IP addresses, port numbers, protocol-specific details, and the content of data payloads.
- **Real-time Analysis**: Wireshark's live capture feature provided real-time data, allowing us to monitor network activity as it happened. This proved valuable for troubleshooting and identifying issues promptly.
- **Display Filters**: The use of display filters made it easy to isolate and focus on specific aspects of network traffic, helping us pinpoint relevant information amid a large volume of data.

Conclusion

The use of Wireshark for analysing network traffic has proven to be an indispensable practice for maintaining a well-functioning and secure network. This tool not only provides a comprehensive view of network activity but also empowers us to diagnose issues, optimize network performance, and enhance our network's overall reliability and security. The insights gained from this analysis will be invaluable in our ongoing efforts to manage and improve our network infrastructure.