Kathmandu University
Department of Computer Science and Engineering
Dhulikhel, Kavre

Networking Lab Report

Subject: COMP - 204

Submitted by:
Samyak Maharjan
Roll no. : 23
Group: CE (II / II) 2023

Submitted to:
Mrs. Praynita Karki
Department of Computer Science and Engineering

Submission Date: 2025/08/09

Question No – 1

Objective:

To Write a program using to find the Even Parity bits/Odd parity bits in a message and append the redundant bit to the original message

Software Requirements:

- Python compiler or interpreter
- Text editor or IDE (e.g., VS Code, Notepad++, PyCharm)

Theory:

Parity bits serve as a fundamental error detection mechanism to verify the integrity of transmitted data in digital communication. A parity bit is an additional bit added to a binary message to control the total number of 1s in the resulting sequence.

Even Parity: The parity bit is set to 1 if the count of 1s in the message is odd, ensuring the total count becomes even. If the count is already even, the parity bit is 0.

Odd Parity: The parity bit is set to 1 if the count of 1s in the message is even, making the total count odd. If the count is already odd, the parity bit is 0.

Parity checking is effective for detecting single-bit errors but is limited in its ability to correct errors or detect multiple-bit errors. It is commonly employed in communication protocols, memory systems, and data storage applications.

Procedure:

The program is in Python and steps are:

1. Accept a binary message (string of 0s and 1s) from the user.
2. Validate the input to ensure it contains only valid binary digits.
3. Calculate the even parity bit by counting the number of 1s and determining if a 0 or 1 is needed to make the count even.
4. Calculate the odd parity bit by determining if a 0 or 1 is needed to make the count odd.
5. Append the respective parity bit to the original message.
6. Display the results for both even and odd parity.

Program (Python Example):

```
# Validate input
    if not message or not is_valid_binary(message):
        print("Error: Please enter a valid binary string (0s and 1s only).")
        return
# Calculate even and odd parity bits
    even_parity_bit = calculate_parity(message, "even")
    odd_parity_bit = calculate_parity(message, "odd")
# Append parity bits to the message
    even_parity_message = append_parity_bit(message, even_parity_bit)
    odd_parity_message = append_parity_bit(message, odd_parity_bit)
# Display results
    print(f"Original Message: {message}")
    print(f"Even Parity Bit: {even_parity_bit}")
    print(f"Message with Even Parity: {even_parity_message}")
    print(f"Odd Parity Bit: {odd_parity_bit}")
    print(f"Message with Odd Parity: {odd_parity_message}")
```

Expected Output/Result:

```
Original Message: 1010
Even Parity Bit: 0
Message with Even Parity: 10100
Odd Parity Bit: 1
Message with Odd Parity: 10101
```

Conclusion:

This experiment successfully demonstrated the calculation of even and odd parity bits for binary messages. By appending a redundant parity bit, basic single-bit error detection can be achieved during data transmission, improving communication reliability.

Question No – 2

Objective:

To establish a simple point-to-point network connection between two PCs using Cisco Packet Tracer.

Software Requirements:

- Cisco Packet Tracer software
- Two PCs
- One Copper Cross-Over Cable

Theory:

A point-to-point network connection is one of the simplest forms of network communication, where two devices (in this case, PCs) are directly connected to exchange data without intermediaries like switches or routers. This setup uses a wired medium, typically an Ethernet cable, to establish a direct link. In this configuration, each PC must be assigned a unique IP address within the same network subnet, along with a subnet mask, to facilitate communication.

The communication is tested using the **ping** utility, which sends Internet Control Message Protocol (ICMP) Echo Request packets from one PC to another and waits for an Echo Reply.

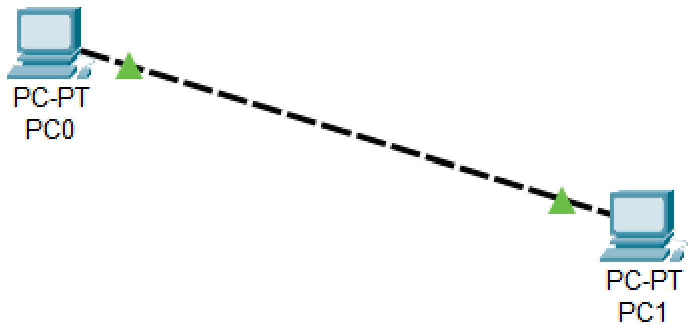
Procedure:

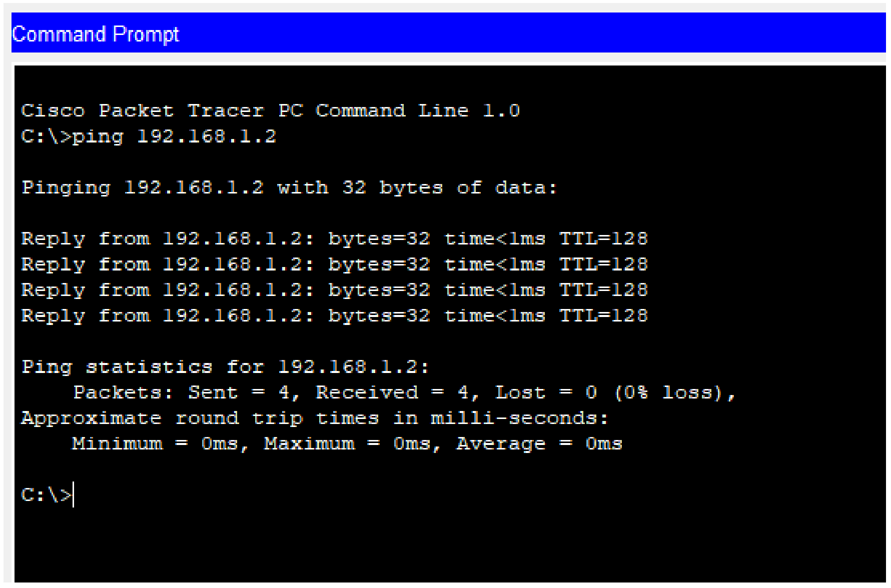
1. Open Cisco Packet Tracer and create a new project.
2. Add two PCs: Drag two PCs from the device menu to the workspace.
3. Connect PCs: Select a Copper Cross-Over Cable from the connections menu and connect PC0's FastEthernet port to PC1's FastEthernet port.
4. Configure IP addresses:
5. Click PC0, go to Desktop > IP Configuration. Set IP: 192.168.1.1, Subnet Mask: 255.255.255.0.
6. Click PC1, go to Desktop > IP Configuration. Set IP: 192.168.1.2, Subnet Mask: 255.255.255.0.
7. Test connectivity:
8. On PC0, go to Desktop > Command Prompt. Type ping 192.168.1.2.
9. Verify successful replies from PC1.
10. Repeat test from PC1: Ping 192.168.1.1 to confirm bidirectional communication.

11. Save project: Save the configuration for future reference.

Expected Output/Result:

If the configuration is successful, the ping command output should display:





```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

This confirms that the two PCs are connected and communicating properly.

Conclusion:

The experiment demonstrates how two PCs can communicate directly over a simple wired Ethernet connection when properly configured with IP addresses in the same subnet. This basic setup lays the foundation for understanding more complex network topologies.

Question No - 3

Objective:

To eastablish and verify a basic network between two PCs using Cisco Packet Tracer connected via a Hub.

Software Requirements:

- Cisco Packet Tracer software
- One Hub device
- Two or more PCs
- Copper Straight-Through Cables

Theory:

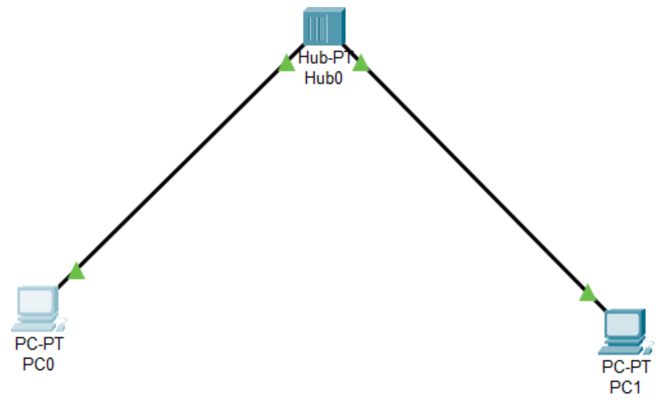
A basic network enables devices to communicate by transferring data packets over a shared medium. In this setup, two PCs are connected through a hub, a simple networking device that broadcasts data to all connected devices. Each PC requires a unique IP address within the same subnet and a subnet mask to ensure they are on the same logical network. The subnet mask defines the network portion of the IP address, facilitating communication. The **ping** utility tests connectivity by sending ICMP Echo Request packets and awaiting Echo Replies. A successful ping confirms proper IP configuration and physical connectivity. In Cisco Packet Tracer, this setup demonstrates fundamental networking concepts such as IP addressing, subnetting, and connectivity testing.

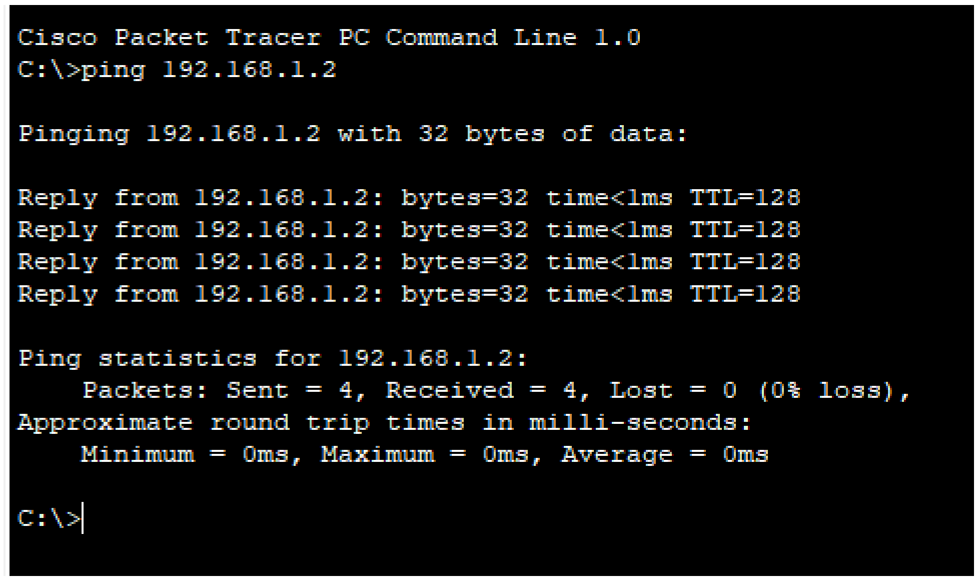
Procedure:

1. Open Cisco Packet Tracer and create a new project.
2. Add devices: Drag two PCs and one hub from the device menu to the workspace.
3. Connect devices:
    - Select a Copper Straight-Through Cable from the connections menu.
    - Connect PC0's FastEthernet port to a port on the hub.
    - Connect PC1's FastEthernet port to another port on the hub.
4. Configure IP addresses:
    - Click PC0, go to Desktop > IP Configuration. Set IP: 192.168.1.1, Subnet Mask: 255.255.255.0.
    - Click PC1, go to Desktop > IP Configuration. Set IP: 192.168.1.2, Subnet Mask: 255.255.255.0.
5. Test connectivity:
    - On PC0, go to Desktop > Command Prompt. Type ping 192.168.1.2.
    - Verify successful replies from PC1.
6. Repeat test from PC1: Ping 192.168.1.1 to confirm bidirectional communication.
7. Save project

Expected Output/Result:

If the network is configured properly, the ping command output will show successful replies between PCs:





```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

This verifies that all PCs are connected and able to communicate through the hub.

Conclusion:

This experiment demonstrates how a hub can be used to create a basic local area network connecting multiple devices. However, hubs lack intelligence and forward data to all connected devices, making them less secure and efficient compared to switches.

Question No – 4

Objective:

To create and verify a basic network between two PCs using Cisco Packet Tracer connected via a switch.

Software Requirements:

1. Cisco Packet Tracer (Software)
2. 2 PCs (PC-PT)
3. 1 Switch (2950-24)
4. Copper Straight-Through Cables

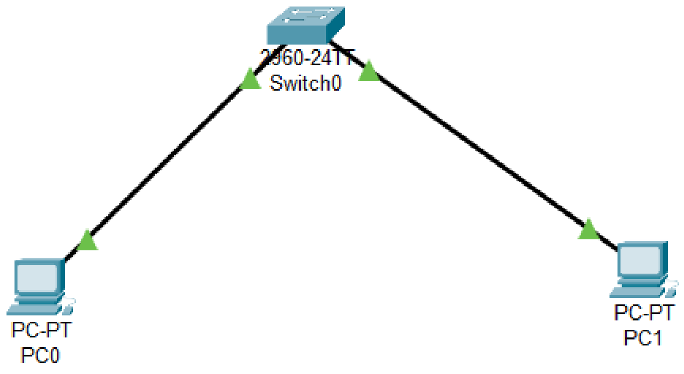
Theory:

This lab creates a Local Area Network (LAN) with two PCs connected via a Cisco 2950-24 switch, a Layer 2 device that forwards frames using MAC addresses. PCs are assigned IP addresses in the same subnet (192.168.1.0/24) for direct communication. The ping command, using ICMP, tests connectivity by sending echo requests and receiving replies, verifying proper IP configuration and switch operation.

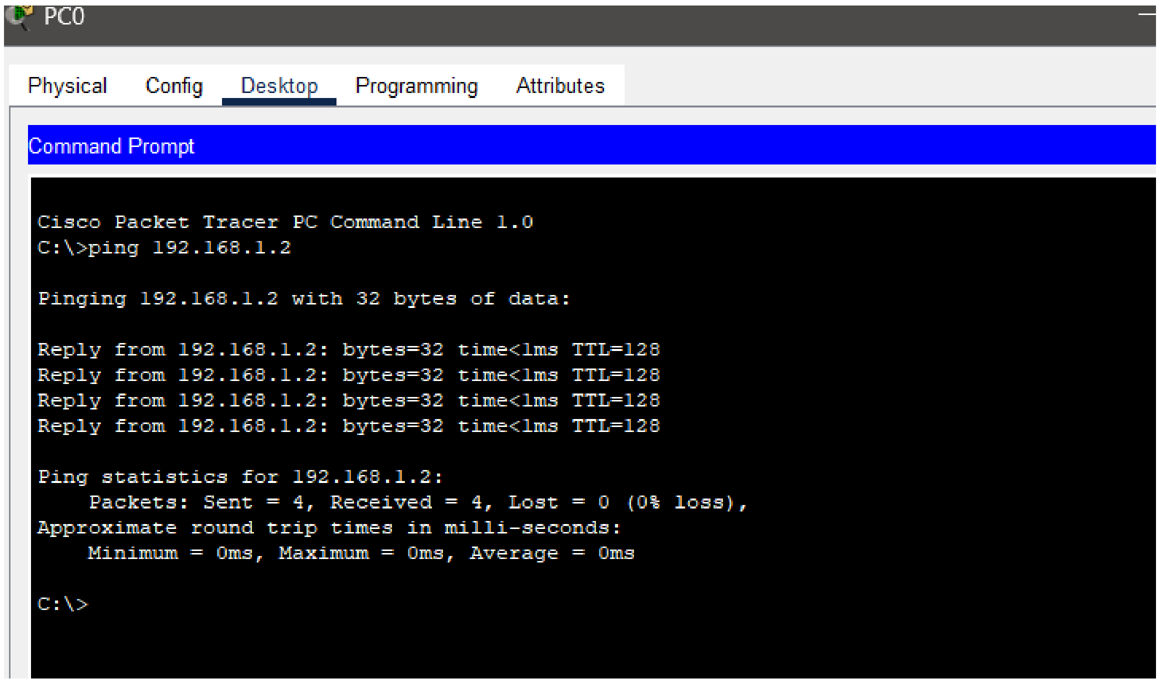
Procedure:

1. Launched Cisco Packet Tracer.
2. Added two PCs (PC0 and PC1) and one 2950-24 switch (Switch0) to the workspace from the device toolbar.
3. Connected PC0 to Switch0 using a Copper Straight-Through cable:
   - PC0: FastEthernet0 to Switch0: FastEthernet0/1
4. Connected PC1 to Switch0 using a Copper Straight-Through cable:
   - PC1: FastEthernet0 to Switch0: FastEthernet0/2
5. Configured IP addresses for the PCs:
   - PC0: IP Address = 192.168.1.1, Subnet Mask = 255.255.255.0
   - PC1: IP Address = 192.168.1.2, Subnet Mask = 255.255.255.0
6. Tested connectivity:
   - From PC0, opened Command Prompt and executed: ping 192.168.1.2
   - From PC1, opened Command Prompt and executed: ping 192.168.1.1
7. Saved the Packet Tracer

Expected Output/Result:



If the configuration is successful, the ping command output should display:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

This confirms that the two PCs are connected and communicating properly.

Conclusion:

The experiment demonstrates how two PCs can communicate directly over a simple wired Ethernet connection when properly configured with IP addresses in the same subnet. This basic setup lays the foundation for understanding more complex network topologies.

Question No – 5

Objective:

To create and verify a basic network between two PCs using Cisco Packet Tracer connected via a Router.

Software Requirements:

- Cisco Packet Tracer software
- One Router
- Two PCs (one per network)
- Copper Straight-Through Cables

Theory:

A network between two PCs via a router involves configuring the PCs and the router to communicate using IP addresses and routing protocols. The router acts as an intermediary, forwarding packets between the PCs based on their IP configurations. In Cisco Packet Tracer, this requires setting up:

- **PCs**: Assign IP addresses and default gateways.
- **Router**: Configure interfaces with IP addresses to connect the PCs' subnets.
- **Verification**: Use commands like ping to confirm connectivity.

This setup simulates a basic LAN-to-LAN communication scenario, where the router directs traffic between two different subnets.

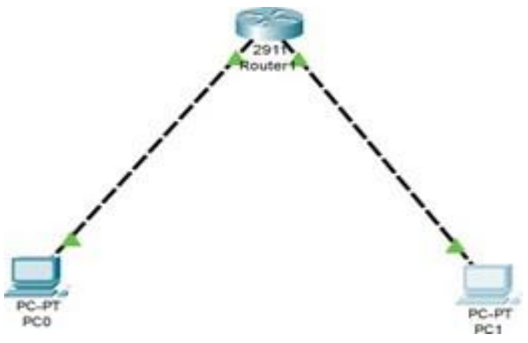
Procedure:

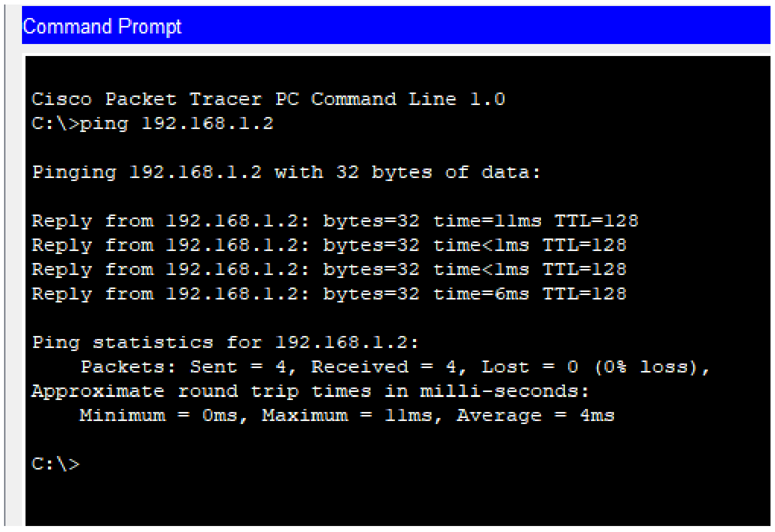
1. **Open Packet Tracer**: Start a new project.
2. **Add Devices**: Place two PCs and one router (e.g., 2811).
3. **Connect Devices**: Use straight-through cables: PC0 to Router's Fa0/0, PC1 to Router's Fa0/1.
4. **Configure PC0**: Desktop > IP Configuration: IP 192.168.1.2, Mask 255.255.255.0, Gateway 192.168.1.1.
5. **Configure PC1**: Desktop > IP Configuration: IP 192.168.2.2, Mask 255.255.255.0, Gateway 192.168.2.1.
6. **Configure Router**:
    - CLI: enable, configure terminal.
    - Fa0/0: interface FastEthernet0/0, ip address 192.168.1.1 255.255.255.0, no shutdown.
    - Fa0/1: interface FastEthernet0/1, ip address 192.168.2.1 255.255.255.0, no shutdown.

7. **Verify Connectivity**:
    - PC0 Command Prompt: ping 192.168.2.2.
    - PC1 Command Prompt: ping 192.168.1.2.
8. **Check Router** (Optional): CLI: show ip interface brief.
9. **Save**: Router CLI: write memory. Save project: File > Save.

Expected Output/Result:

If configured properly, the ping command should display:





```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time=11ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time<1ms TTL=128
Reply from 192.168.1.2: bytes=32 time=6ms TTL=128

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 11ms, Average = 4ms

C:\>
```

This confirms that the two networks are successfully connected through the router and can communicate.

Conclusion:

This experiment demonstrates how a router can be used to connect two different networks, enabling inter-network communication by routing data packets based on their destination IP addresses.

Question No – 6

Objective:

To understand and implement a simple VLAN (Virtual Local Area Network) configuration on Cisco Packet Tracer.

Software Requirements:

- Cisco Packet Tracer software
- One Switch
- Four PCs
- Copper Straight-Through Cables

Theory:

A Virtual Local Area Network (VLAN) is a logical segmentation of a physical network that allows multiple broadcast domains on the same switch. A **VLAN** allows network administrators to create separate broadcast domains within a single switch by logically grouping devices. This improves:

- Network segmentation
- Security
- Performance

Procedure:

1. Open Cisco Packet Tracer and place one Switch and four PCs on the workspace.
2. Connect all PCs to the Switch using Copper Straight-Through cables.
3. Assign IP addresses to PCs as follows:
   - VLAN 10:
     * PC0: 192.168.10.2 / 255.255.255.0
     * PC1: 192.168.10.3 / 255.255.255.0
   - VLAN 20:
     * PC2: 192.168.20.2 / 255.255.255.0
     * PC3: 192.168.20.3 / 255.255.255.0
4. Configure VLANs on the switch using CLI commands:

```
Switch> enable
Switch# configure terminal
Switch(config)# vlan 10
Switch(config-vlan)# name
Sales Switch(config)# vlan 20
Switch(config-vlan)# name HR
```

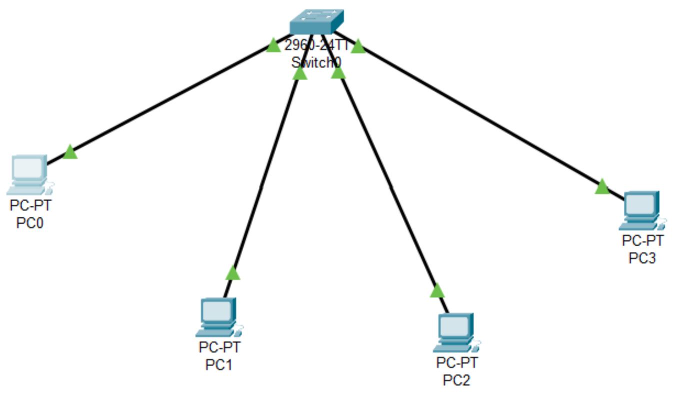
5. Assign switch ports to VLANs:

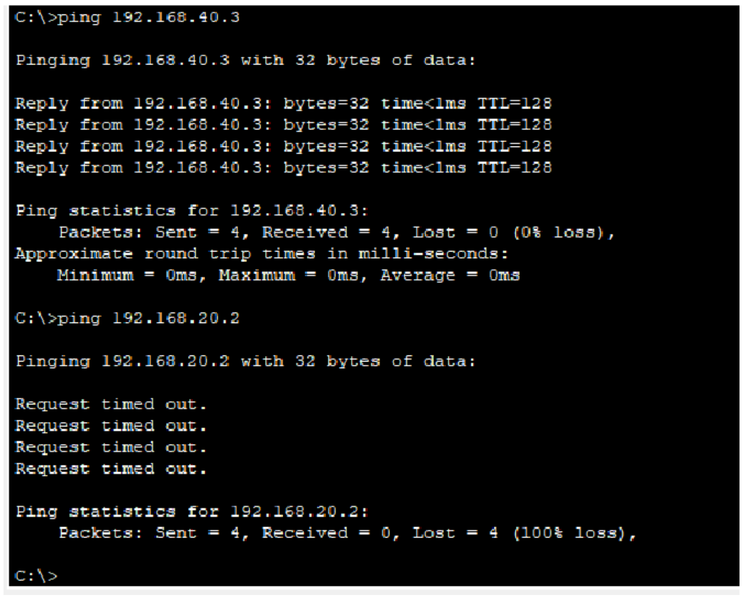
```
   Switch(config)# interface fastEthernet 0/1
   Switch(config-if)# switchport mode access
   Switch(config-if)# switchport access vlan 10
   Switch(config)# interface fastEthernet 0/2
   Switch(config-if)# switchport mode access
   Switch(config-if)# switchport access vlan 10
   Switch(config)# interface fastEthernet 0/3
   Switch(config-if)# switchport mode access
   Switch(config-if)# switchport access vlan 20
   Switch(config)# interface fastEthernet 0/4
   Switch(config-if)# switchport mode access
   Switch(config-if)# switchport access vlan 20
```

6. Save the configuration.
7. Test connectivity using ping:
   - PCs within the same VLAN should be able to ping each other.
   - PCs in different VLANs should not be able to communicate.

Expected Output/Result:

Ping results should be successful between PCs in the same VLAN (e.g., PC0 ↔ PC1) and fail between PCs in different VLANs (e.g., PC0 ↔ PC2). This demonstrates that VLANs logically separate networks on the same physical switch.





```
C:\>ping 192.168.40.3

Pinging 192.168.40.3 with 32 bytes of data:

Reply from 192.168.40.3: bytes=32 time<1ms TTL=128
Reply from 192.168.40.3: bytes=32 time<1ms TTL=128
Reply from 192.168.40.3: bytes=32 time<1ms TTL=128
Reply from 192.168.40.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.40.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.20.2

Pinging 192.168.20.2 with 32 bytes of data:

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

Ping statistics for 192.168.20.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\>
```

Conclusion:

This experiment demonstrates how VLANs are configured on a switch to logically segment a network. Devices in different VLANs cannot communicate directly, improving security and reducing unnecessary broadcast traffic.

Question No – 7

Objective:

To enable communication between two VLANs using a router in Cisco Packet Tracer.

Software Requirements:

- Cisco Packet Tracer software
- One Router
- One Switch
- Four PCs
- Copper Straight-Through Cables

Theory:

By default, devices on different VLANs cannot communicate because they are in separate broadcast domains. Inter-VLAN communication requires a Layer 3 device, such as a router or a Layer 3 switch, to route traffic between VLANs. This is achieved using a technique called Router-on-a-Stick, where a single router interface is configured with multiple sub-interfaces, each assigned to a different VLAN.

Procedure:

1. Open Cisco Packet Tracer and place one Router, one Switch, and four PCs on the workspace.
2. Connect all PCs to the Switch using Copper Straight-Through cables.
3. Connect the Switch to the Router using one cable (FastEthernet 0/1 on both devices).
4. Assign IP addresses to PCs:
   - VLAN 10:
     * PC0: 192.168.10.2 / 255.255.255.0, Gateway: 192.168.10.1
     * PC1: 192.168.10.3 / 255.255.255.0, Gateway: 192.168.10.1
   - VLAN 20:
     * PC2: 192.168.20.2 / 255.255.255.0, Gateway: 192.168.20.1
     * PC3: 192.168.20.3 / 255.255.255.0, Gateway: 192.168.20.1
5. Configure VLANs on the switch as done in Experiment 6, assigning ports accordingly.
6. Configure trunk port on switch:

```
Switch(config)# interface fa0/1
Switch(config-if)# switchport mode trunk
```

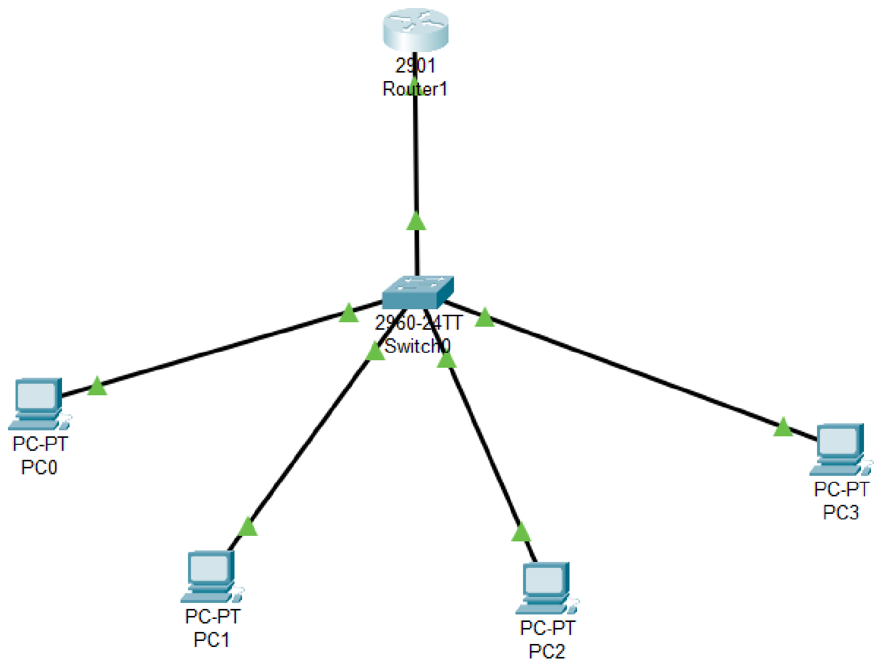
7. Configure router sub-interfaces:

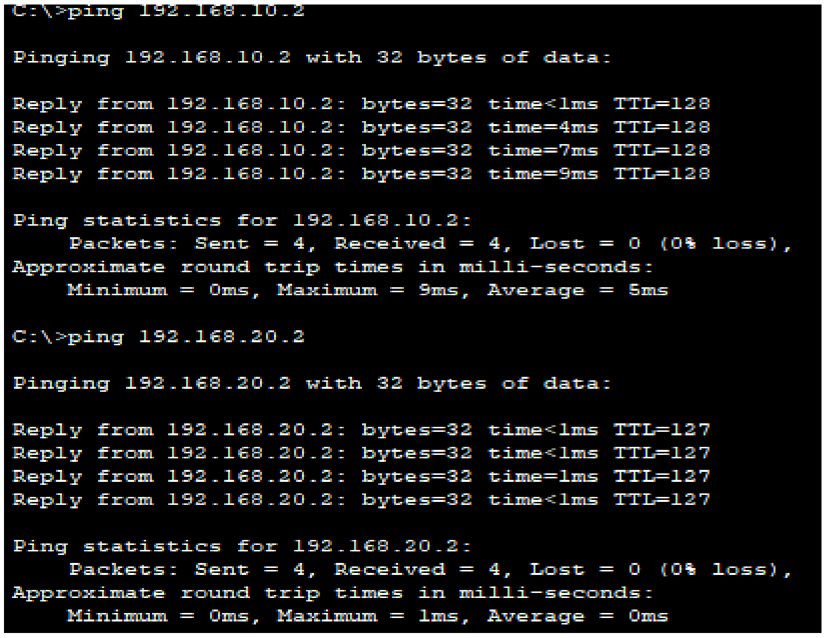
```
Router> enable
Router# configure terminal
Router(config)# interface fa0/0.10
Router(config-subif)# encapsulation dot1Q 10
Router(config-subif)# ip address 192.168.10.1 255.255.255.0
Router(config)# interface fa0/0.20
Router(config-subif)# encapsulation dot1Q 20
Router(config-subif)# ip address 192.168.20.1 255.255.255.0
Router(config)# interface fa0/0
Router(config-if)# no shutdown
```

8. Save the configuration.
9. Ping from PC1 to PC3 to test communication between VLANs.

Expected Output/Result:

If configured properly, the ping should succeed between PCs in different VLANs, proving that the router is correctly routing traffic between VLAN 10 and VLAN 20.





```
C:\>ping 192.168.10.2

Pinging 192.168.10.2 with 32 bytes of data:

Reply from 192.168.10.2: bytes=32 time<1ms TTL=128
Reply from 192.168.10.2: bytes=32 time=4ms TTL=128
Reply from 192.168.10.2: bytes=32 time=7ms TTL=128
Reply from 192.168.10.2: bytes=32 time=9ms TTL=128

Ping statistics for 192.168.10.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 9ms, Average = 5ms

C:\>ping 192.168.20.2

Pinging 192.168.20.2 with 32 bytes of data:

Reply from 192.168.20.2: bytes=32 time<1ms TTL=127
Reply from 192.168.20.2: bytes=32 time<1ms TTL=127
Reply from 192.168.20.2: bytes=32 time=1ms TTL=127
Reply from 192.168.20.2: bytes=32 time<1ms TTL=127

Ping statistics for 192.168.20.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

Conclusion:

This experiment demonstrates how to configure inter-VLAN routing using a router-on-a-stick setup. It allows devices in separate VLANs to communicate by routing traffic through a router.

Question No – 8

Objective:

To implement a basic firewall using Access Control Lists (ACLs) in Cisco Packet Tracer and control traffic between networks.

Software Requirements:

- Cisco Packet Tracer software
- One Router and two Switches
- Four PCs (two in each network)
- Copper Straight-Through Cables

Theory:

A firewall is a network security device that monitors and controls incoming and outgoing network traffic based on predefined security rules. In Cisco routers, firewalls can be implemented using Access Control Lists (ACLs). An ACL defines which traffic is allowed or denied based on source, destination IP addresses, and protocols.

Standard ACLs filter traffic based on source IP address, while extended ACLs can filter based on source, destination, and port numbers. In this experiment, we configure an ACL to block communication from one PC to another while allowing other traffic.

Procedure:

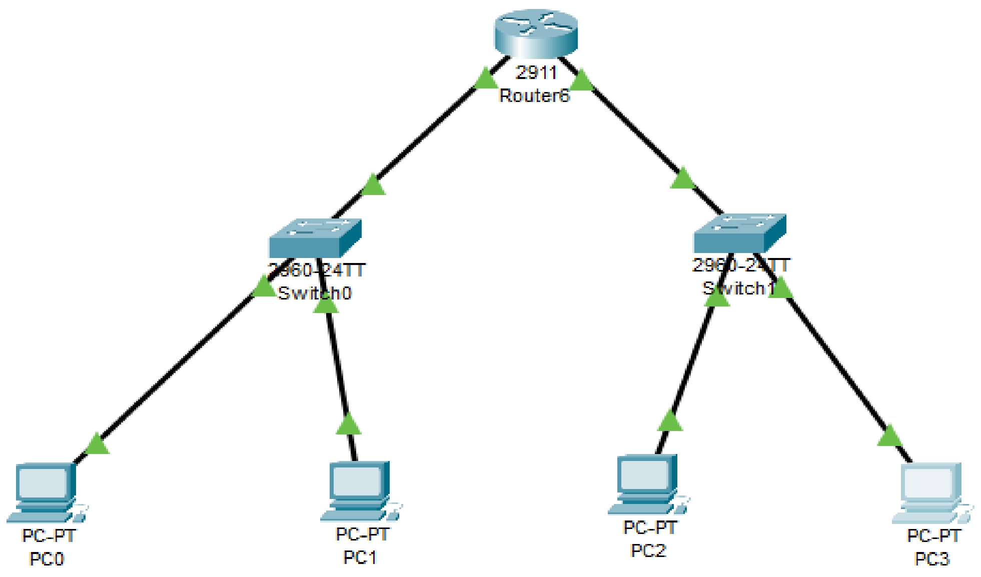
1. Open Cisco Packet Tracer and place one Router, two switches and four PCs on the workspace.
2. Connect each PCs to the switches using Copper Straight-Through cables.
3. Assign IP addresses:
   - Network 1:
     * PC0: 192.168.1.2 / 255.255.255.0, Gateway: 192.168.1.1
     * PC1: 192.168.1.3 / 255.255.255.0, Gateway: 192.168.1.1
     * Router gig0/0: 192.168.1.1 / 255.255.255.0
   - Network 2:
     * PC2: 192.168.2.2 / 255.255.255.0, Gateway: 192.168.2.1
     * PC3: 192.168.2.3 / 255.255.255.0, Gateway: 192.168.2.1
     * Router gig0/1: 192.168.2.1 / 255.255.255.0
4. Enable interfaces on the router using 'no shutdown'.
5. Verify connectivity between all PCs using the ping command.

6. Configure a standard ACL on the router to block PC0 (192.168.1.2) from accessing PC2's network:

```
  Router> enable
  Router# configure terminal
  Router(config)# access-list 1 deny 192.168.1.2 0.0.0.0
  Router(config)# access-list 1 permit any
  Router(config)# interface fa0/0
  Router(config-if)# ip access-group 1 out
```

7. Save the configuration.
8. Test connectivity:
   - PC1 should be blocked when pinging PC2.
   - PC0 should be able to ping PC2.

Expected Output/Result:



After applying the ACL, PC1's ping requests to PC2 will time out, while PC0 will be able to communicate with PC2. This demonstrates how a basic firewall rule can control traffic between networks.

Conclusion:

This experiment demonstrates how to implement a simple firewall using ACLs on a router in Cisco Packet Tracer. By applying ACL rules, network administrators can control which devices or networks are allowed or denied access, improving overall security.