# Experiment No: 01

## Experiment Title:

**Design and Simulate in Cisco Packet Tracer: Create LAN and WAN Topologies & Configure Basic Device Settings (IP Addressing, Device Naming)**

## Objective:

To design and simulate Local Area Network (LAN) and Wide Area Network (WAN) topologies using Cisco Packet Tracer. Also, to configure IP addressing and basic device settings like device naming.

## Tools Required:

- Cisco Packet Tracer (version 7.2 or later)
- A system with Windows/Linux/Mac

## Experiment Steps

### PART A: Create LAN TopologyGoal: Connect multiple PCs via switches and configure IPs and names

### Step 1: Open Cisco Packet Tracer

- Launch Cisco Packet Tracer.
- Open a new blank workspace.

### Step 2: Add Devices

Drag and drop the following from the bottom Device-Type box:

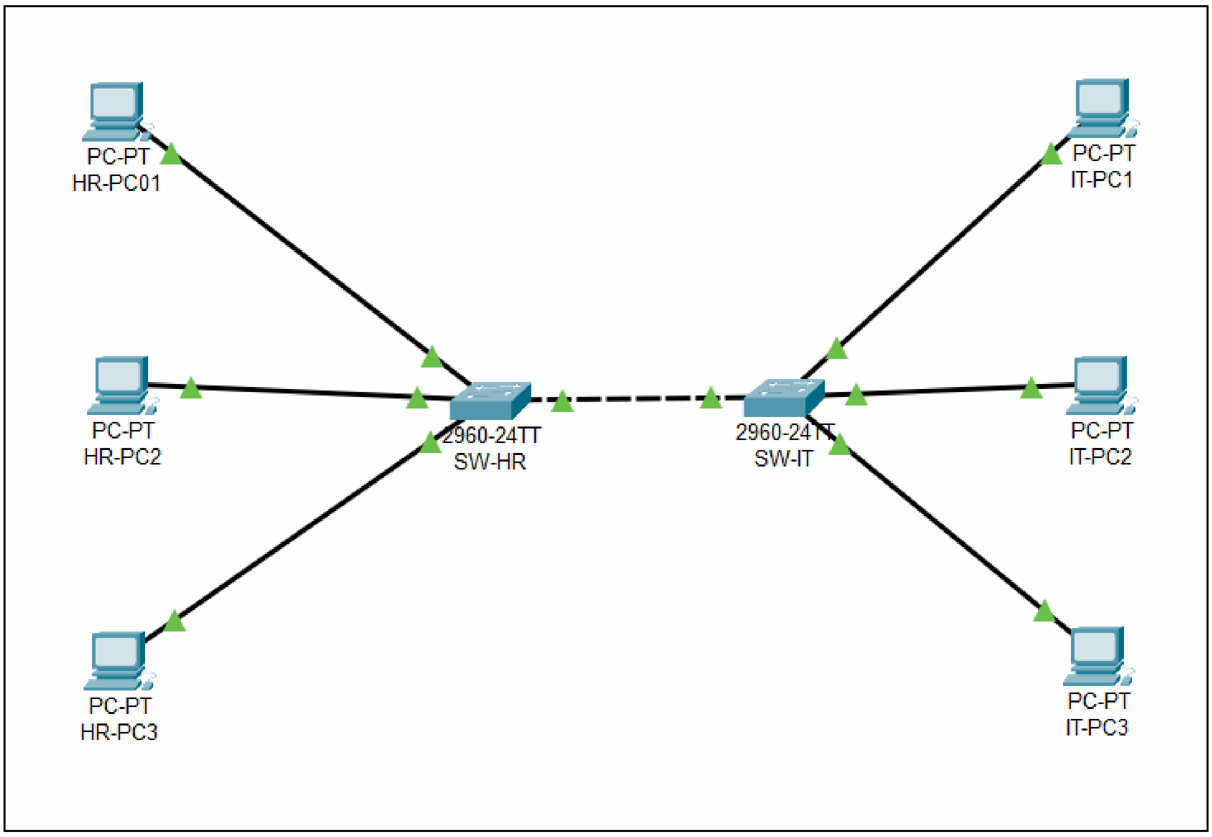
- **2 Switches** → Switches → `2960`
- **6 PCs** → End Devices → `PC`
- **1 Router** → Routers → `1841` (optional, if inter-LAN or gateway simulation is needed)

### Step 3: Connect Devices using Copper Cables

Use **"Copper Straight-Through Cable"** to connect:

- PC0 → Switch0 (FastEthernet 0)
- PC1 → Switch0 (FastEthernet 1)
- PC2 → Switch0 (FastEthernet 2)
- PC3 → Switch1 (FastEthernet 0)
- PC4 → Switch1 (FastEthernet 1)
- PC5 → Switch1 (FastEthernet 2)

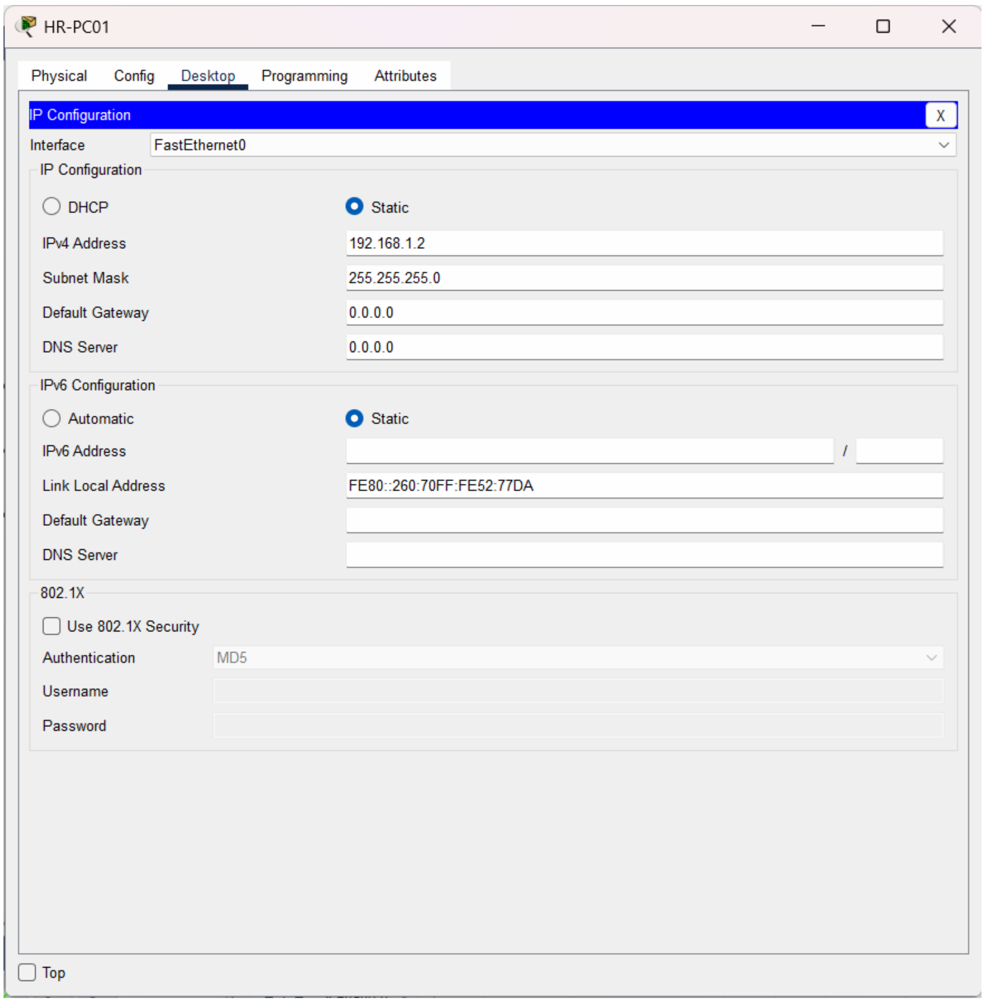
- Connect **Switch0 to Switch1** using FastEthernet port (any unused)



## Step 4: Assign IP Addresses to PCs

Click on each PC → Desktop → IP Configuration:

| **Device** | **IP Address** | **Subnet Mask** |
|---|---|---|
| PC0 | 192.168.1.2 | 255.255.255.0 |
| PC1 | 192.168.1.3 | 255.255.255.0 |
| PC2 | 192.168.1.4 | 255.255.255.0 |
| PC3 | 192.168.1.5 | 255.255.255.0 |
| PC4 | 192.168.1.6 | 255.255.255.0 |
| PC5 | 192.168.1.7 | 255.255.255.0 |


HR-PC01

Physical | Config | Desktop | Programming | Attributes

IP Configuration

| | |
|---|---|
| Interface | FastEthernet0 |

IP Configuration

| | |
|---|---|
| DHCP | [✓ Static] |
| IPv4 Address | 192.168.1.2 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 0.0.0.0 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

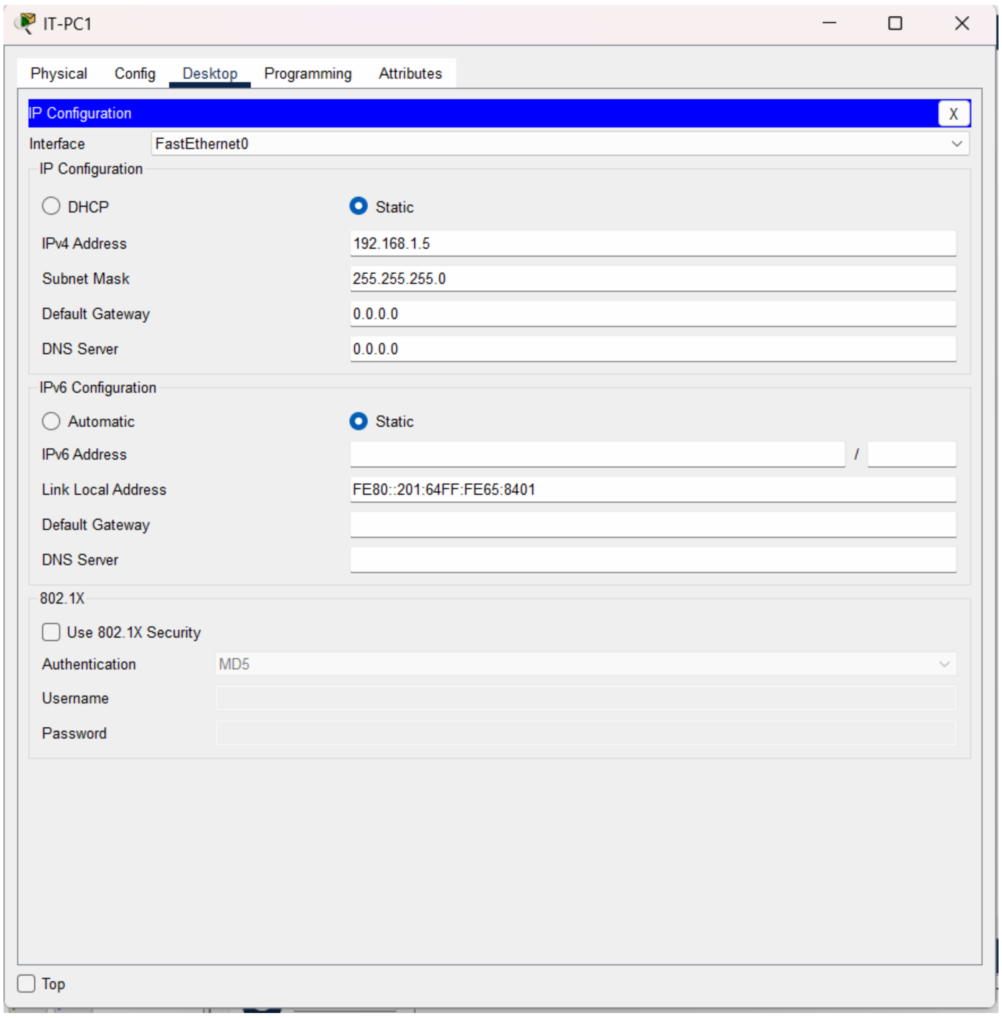
| | |
|---|---|
| Automatic | [✓ Static] |
| IPv6 Address | / |
| Link Local Address | FE80::260:70FF:FE52:77DA |
| Default Gateway | |
| DNS Server | |

802.1X

Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

Top


IT-PC1

Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

IPv4 Address: 192.168.1.5

Subnet Mask: 255.255.255.0

Default Gateway: 0.0.0.0

DNS Server: 0.0.0.0

IPv6 Configuration

Automatic [✓ Static]

IPv6 Address: /

Link Local Address: FE80::201:64FF:FE65:8401

Default Gateway:

DNS Server:

802.1X

☐ Use 802.1X Security

Authentication: MD5

Username:

Password:

☐ Top

## PART B: Create WAN Topology

### Goal: Connect 2 LANs using Routers

### Step 1: Add Devices

- **2 Routers** → Router0, Router1 (e.g., 1841)
- **2 Switches**
- **4 PCs** (2 for each LAN)

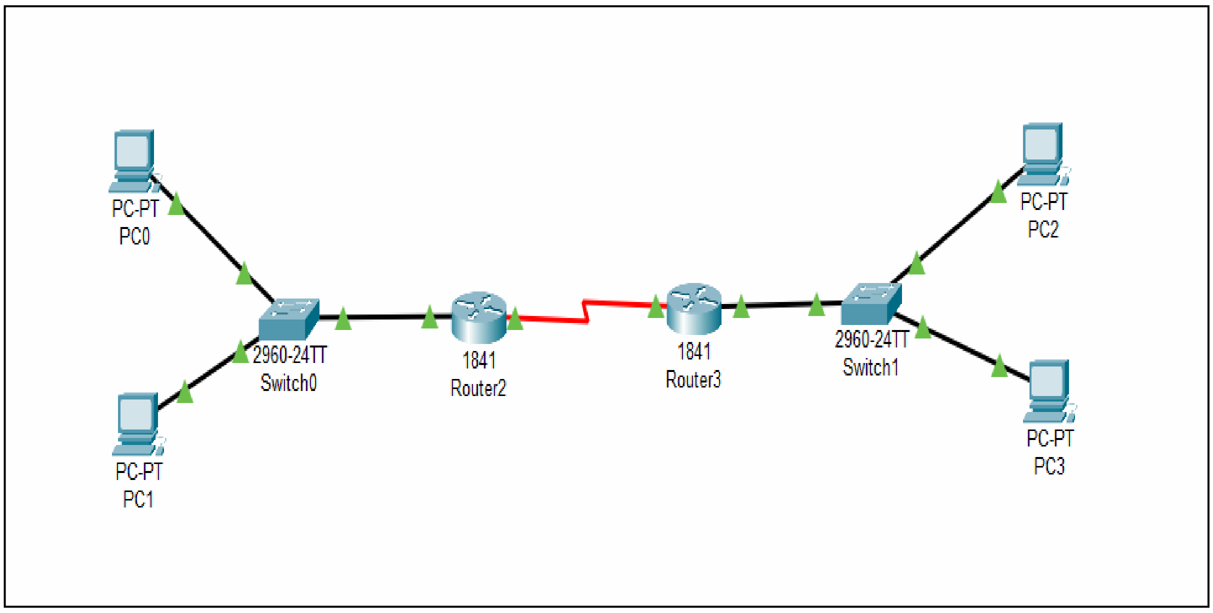
### Step 2: Connect Each LAN

LAN A:
PC0 & PC1 → Switch0 → Router0 (via FastEthernet 0/0)

LAN B:
PC2 & PC3 → Switch1 → Router1 (via FastEthernet 0/0)

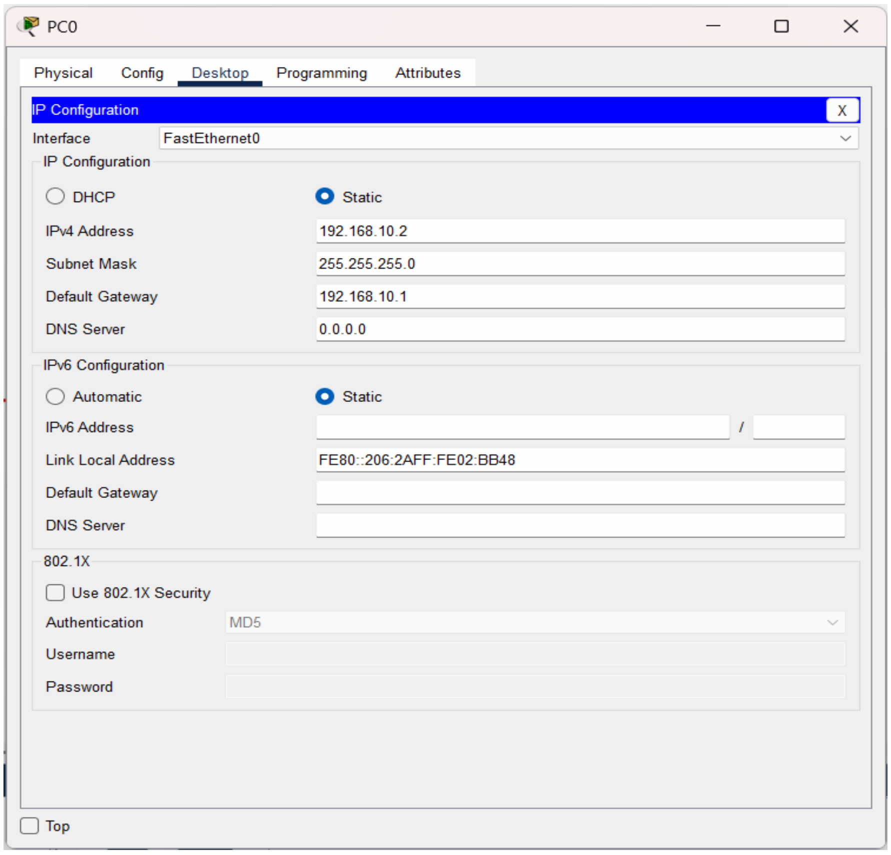
Use **Serial connection** between Router0 and Router1:

- Serial 0/0/0 ↔ Serial 0/0/0



## Step 3: Assign IP Addresses

| Device | Interface | IP Address | Subnet Mask |
|---|---|---|---|
| PC0 | NIC | 192.168.10.2 | 255.255.255.0 |
| PC1 | NIC | 192.168.10.3 | 255.255.255.0 |
| Router0 | FastEthernet 0/0 | 192.168.10.1 | 255.255.255.0 |
| Router0 | Serial 0/0/0 | 10.0.0.1 | 255.0.0.0 |
| Router1 | Serial 0/0/0 | 10.0.0.2 | 255.0.0.0 |
| Router1 | FastEthernet 0/0 | 192.168.20.1 | 255.255.255.0 |
| PC2 | NIC | 192.168.20.2 | 255.255.255.0 |
| PC3 | NIC | 192.168.20.3 | 255.255.255.0 |



PC0

Physical | Config | Desktop | Programming | Attributes

IP Configuration

| | |
|---|---|
| Interface | FastEthernet0 |

IP Configuration

DHCP [✓ Static]

| | |
|---|---|
| IPv4 Address | 192.168.10.2 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 192.168.10.1 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

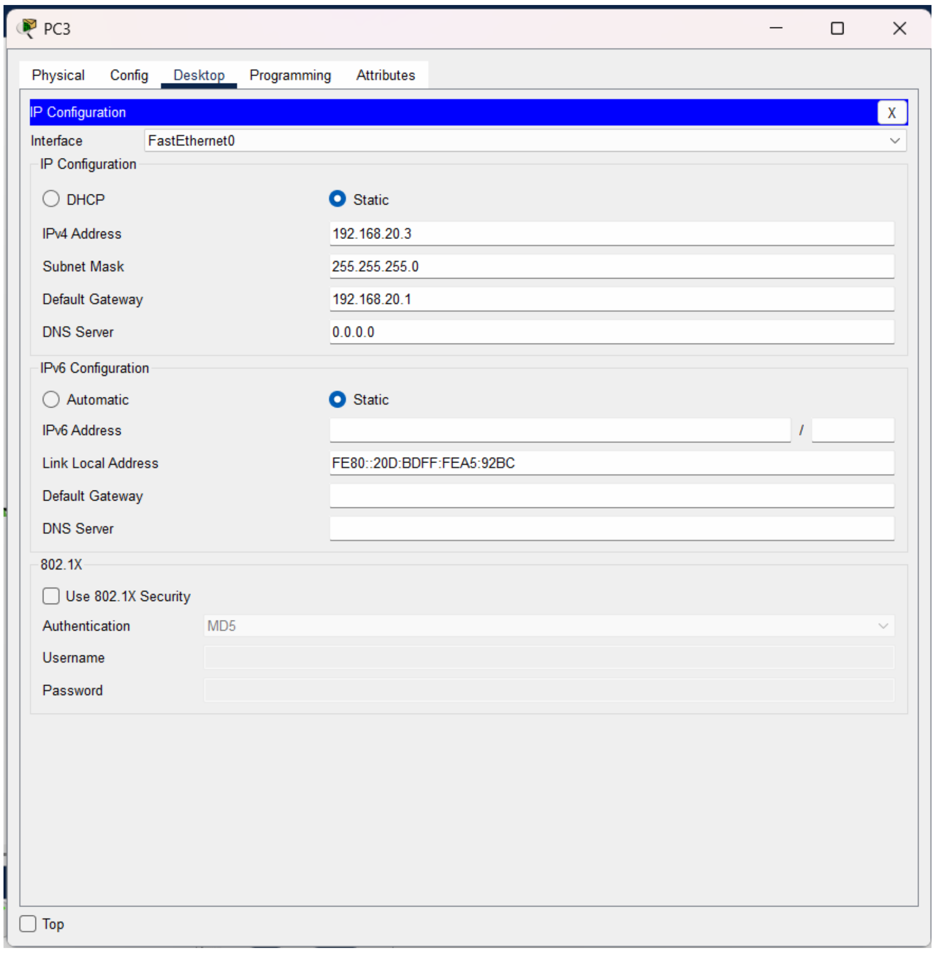
| | |
|---|---|
| IPv6 Address | / |
| Link Local Address | FE80::206:2AFF:FE02:BB48 |
| Default Gateway | |
| DNS Server | |

802.1X

☐ Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

☐ Top



PC3

Physical | Config | Desktop | Programming | Attributes

IP Configuration

| | |
|---|---|
| Interface | FastEthernet0 |

IP Configuration

DHCP [✓ Static]

| | |
|---|---|
| IPv4 Address | 192.168.20.3 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 192.168.20.1 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

| | |
|---|---|
| IPv6 Address | / |
| Link Local Address | FE80::20D:BDFF:FEA5:92BC |
| Default Gateway | |
| DNS Server | |

802.1X

☐ Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

☐ Top

## Step 4: Configure Router Interfaces

Click on Router → CLI or Config tab:

**Router0:**

```
arduino
Router> enable
Router# configure terminal
Router(config)# interface fa0/0
Router(config-if)# ip address 192.168.10.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# interface s0/0/0
Router(config-if)# ip address 10.0.0.1 255.0.0.0
Router(config-if)# clock rate 64000
Router(config-if)# no shutdown
```

**Router1:**

```
arduino
Router> enable
Router# configure terminal
Router(config)# interface fa0/0
Router(config-if)# ip address 192.168.20.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# interface s0/0/0
Router(config-if)# ip address 10.0.0.2 255.0.0.0
Router(config-if)# no shutdown
```

## Step 5: Add Static Routes (Optional)

**On Router0:**

```
arduino
CopyEdit
Router(config)# ip route 192.168.20.0 255.255.255.0 10.0.0.2
```

**On Router1:**

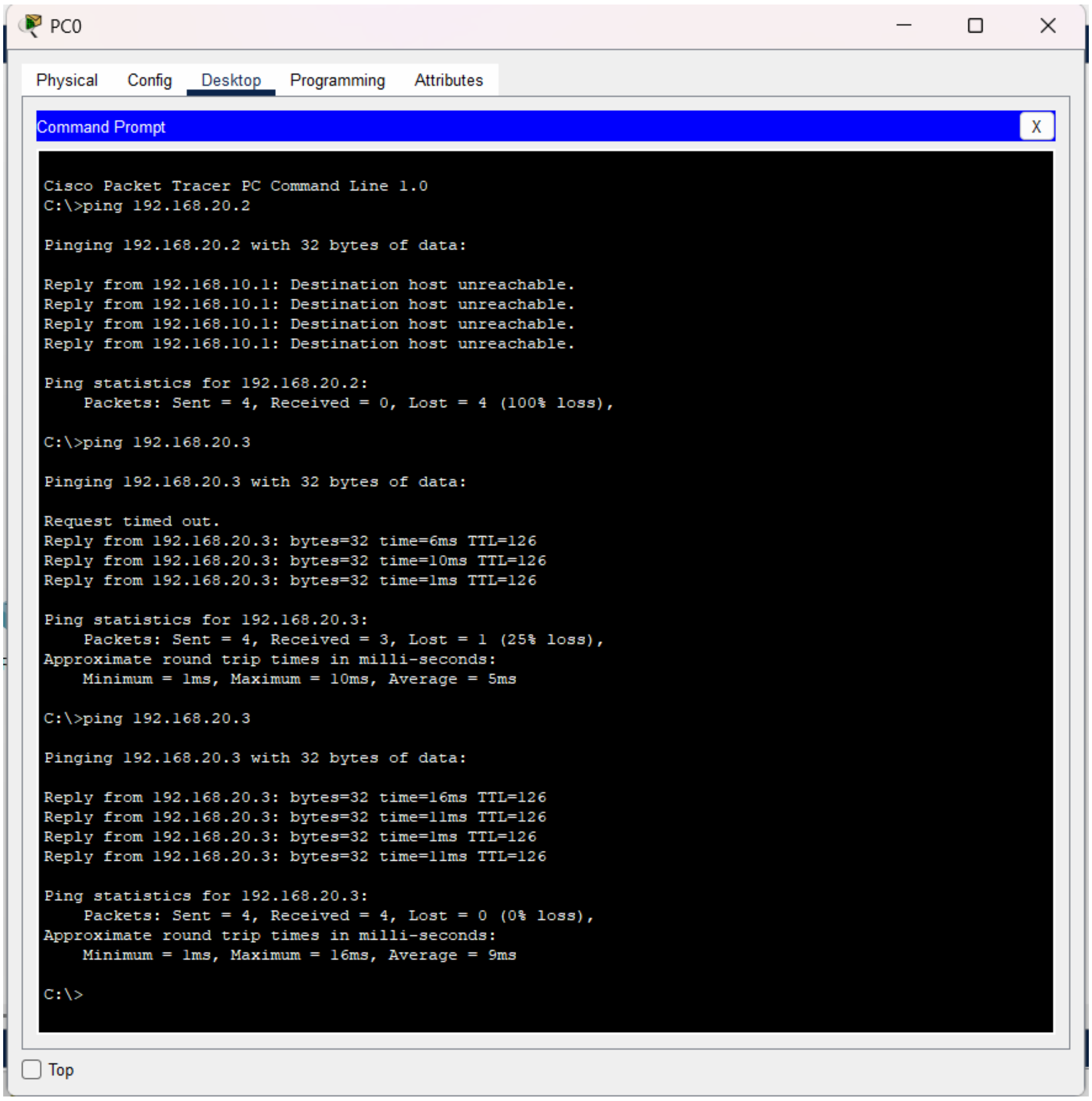
```
arduino
Router(config)# ip route 192.168.10.0 255.255.255.0 10.0.0.1
```

## Step 6: Test End-to-End Connectivity

From PC0, ping PC3:

- `ping 192.168.20.3`

You should receive replies.



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.20.2

Pinging 192.168.20.2 with 32 bytes of data:

Reply from 192.168.10.1: Destination host unreachable.
Reply from 192.168.10.1: Destination host unreachable.
Reply from 192.168.10.1: Destination host unreachable.
Reply from 192.168.10.1: Destination host unreachable.

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

C:\>ping 192.168.20.3

Pinging 192.168.20.3 with 32 bytes of data:

Request timed out.
Reply from 192.168.20.3: bytes=32 time=6ms TTL=126
Reply from 192.168.20.3: bytes=32 time=10ms TTL=126
Reply from 192.168.20.3: bytes=32 time=1ms TTL=126

Ping statistics for 192.168.20.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 10ms, Average = 5ms

C:\>ping 192.168.20.3

Pinging 192.168.20.3 with 32 bytes of data:

Reply from 192.168.20.3: bytes=32 time=16ms TTL=126
Reply from 192.168.20.3: bytes=32 time=11ms TTL=126
Reply from 192.168.20.3: bytes=32 time=1ms TTL=126
Reply from 192.168.20.3: bytes=32 time=11ms TTL=126

Ping statistics for 192.168.20.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 16ms, Average = 9ms

C:\>
```

## Suggested Screenshots for Assignment Submission

1. LAN topology layout with device labels
2. IP configuration window for 1–2 PCs
3. CLI of Router showing interface config
4. Ping result window showing connectivity
5. WAN topology layout

## Conclusion:

In this experiment, we successfully designed and simulated both LAN (Local Area Network) and WAN (Wide Area Network) topologies using Cisco Packet Tracer. We interconnected multiple end devices, switches, and routers to form efficient network structures and configured basic device settings such as IP addresses and device naming.

**The exercise reinforced the importance of:**

Proper IP address assignment using subnetting principles
Accurate device labelling for effective network management
Understanding LAN vs. WAN topology differences
Testing connectivity using basic tools like ping

Through this simulation, we gained hands-on experience with core network setup tasks that form the foundation for all advanced networking configurations. The successful communication between devices validated the correctness of the topology and configuration.

**Experiment No: 02**

**Experiment Title:**

**Create and compare different topologies (Bus, Ring, Star, Mesh) using Packet Tracer.**

## Tools Required:

- **Cisco Packet Tracer** (version 8.x recommended)

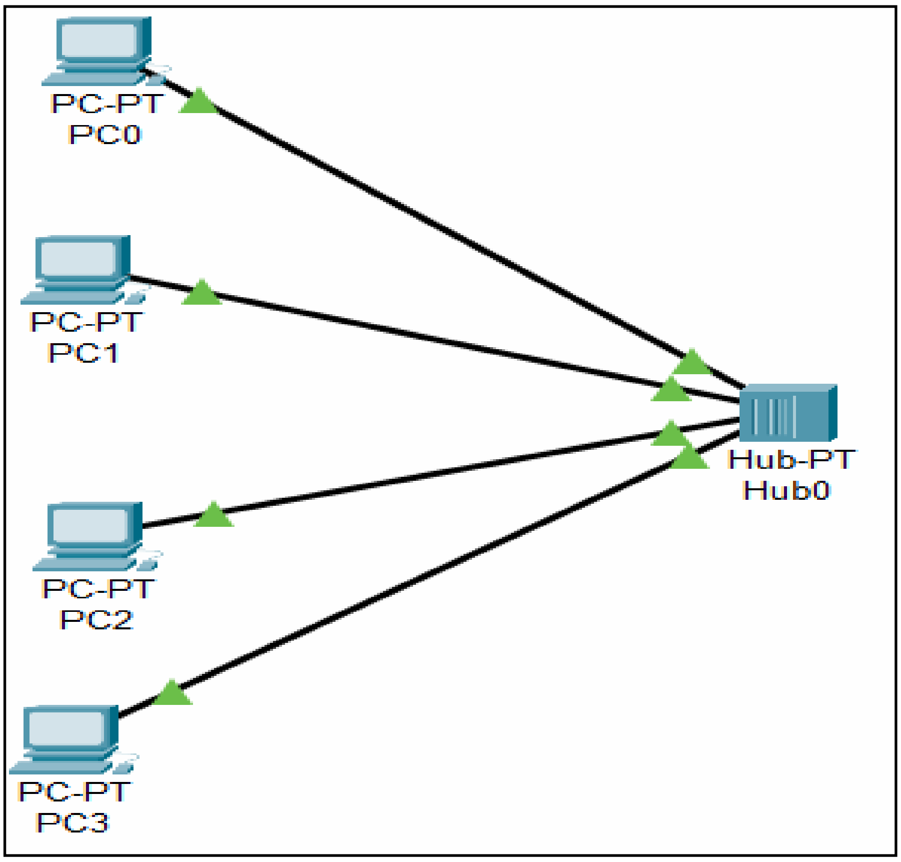
## Objective:

Design and simulate four network topologies — **Bus, Ring, Star, and Mesh** — and compare their configuration, working, and performance basics.

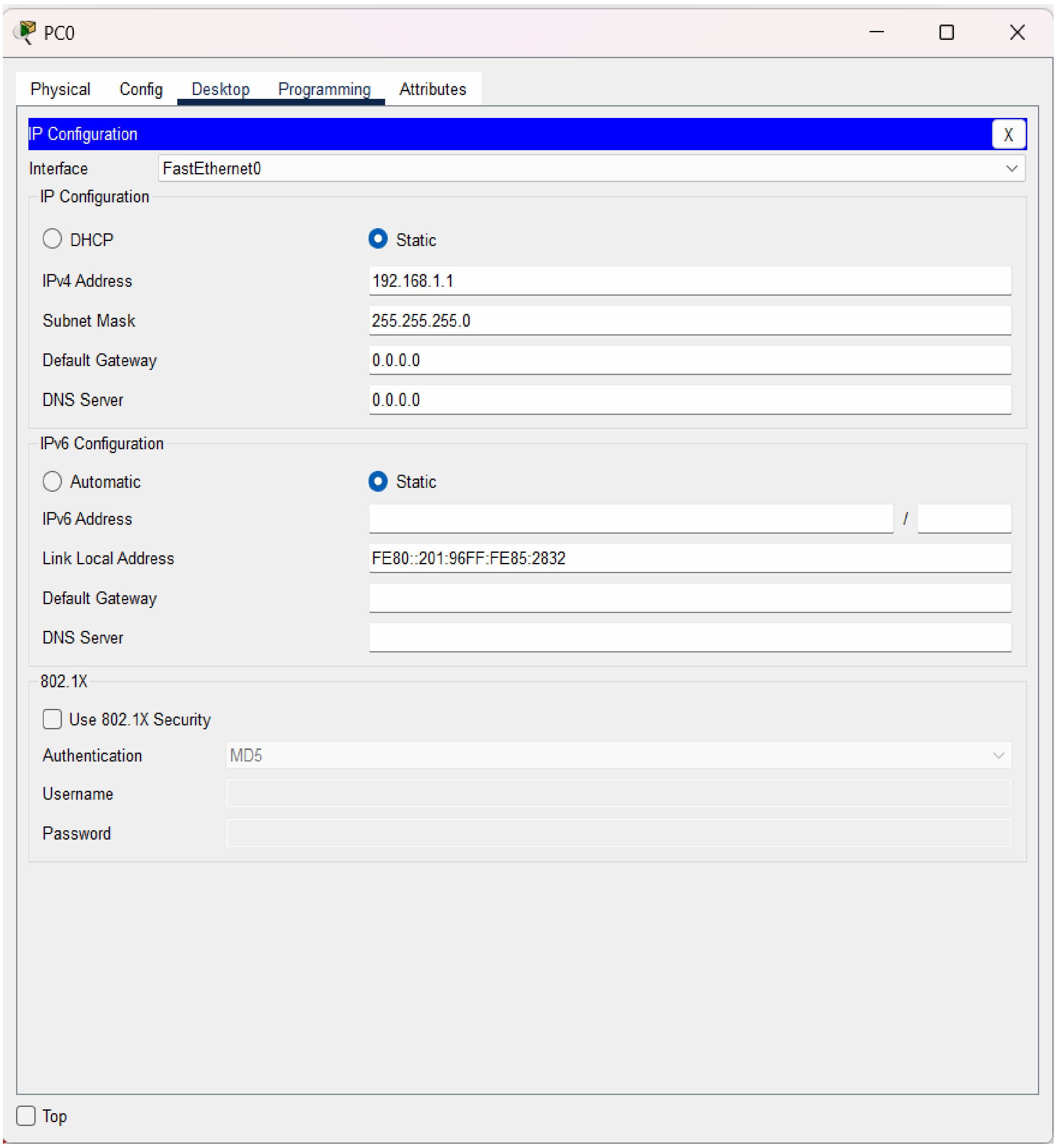
## 1. Bus Topology

### Steps:

1. **Open Cisco Packet Tracer**
2. From **End Devices**, drag and drop **4 PCs** (PC0 to PC3).
3. From **Network Devices → Hubs**, drag and drop **1 Hub**.
4. Arrange devices in a straight line (to represent a bus).
5. **Connections:**
   - Use **Copper Straight-Through** cable.
   - Connect each PC to the **Hub** (PC0 → Hub, PC1 → Hub, etc.)



6. **IP Configuration:**
    - Click each PC → Desktop → IP Configuration.
    - Assign the following IPs:
        - PC0: 192.168.1.1 / 255.255.255.0
        - PC1: 192.168.1.2 / 255.255.255.0
        - PC2: 192.168.1.3 / 255.255.255.0
        - PC3: 192.168.1.4 / 255.255.255.0



PC0

Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

| | |
|---|---|
| IPv4 Address | 192.168.1.1 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 0.0.0.0 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

| | |
|---|---|
| IPv6 Address | / |
| Link Local Address | FE80::201:96FF:FE85:2832 |
| Default Gateway | |
| DNS Server | |

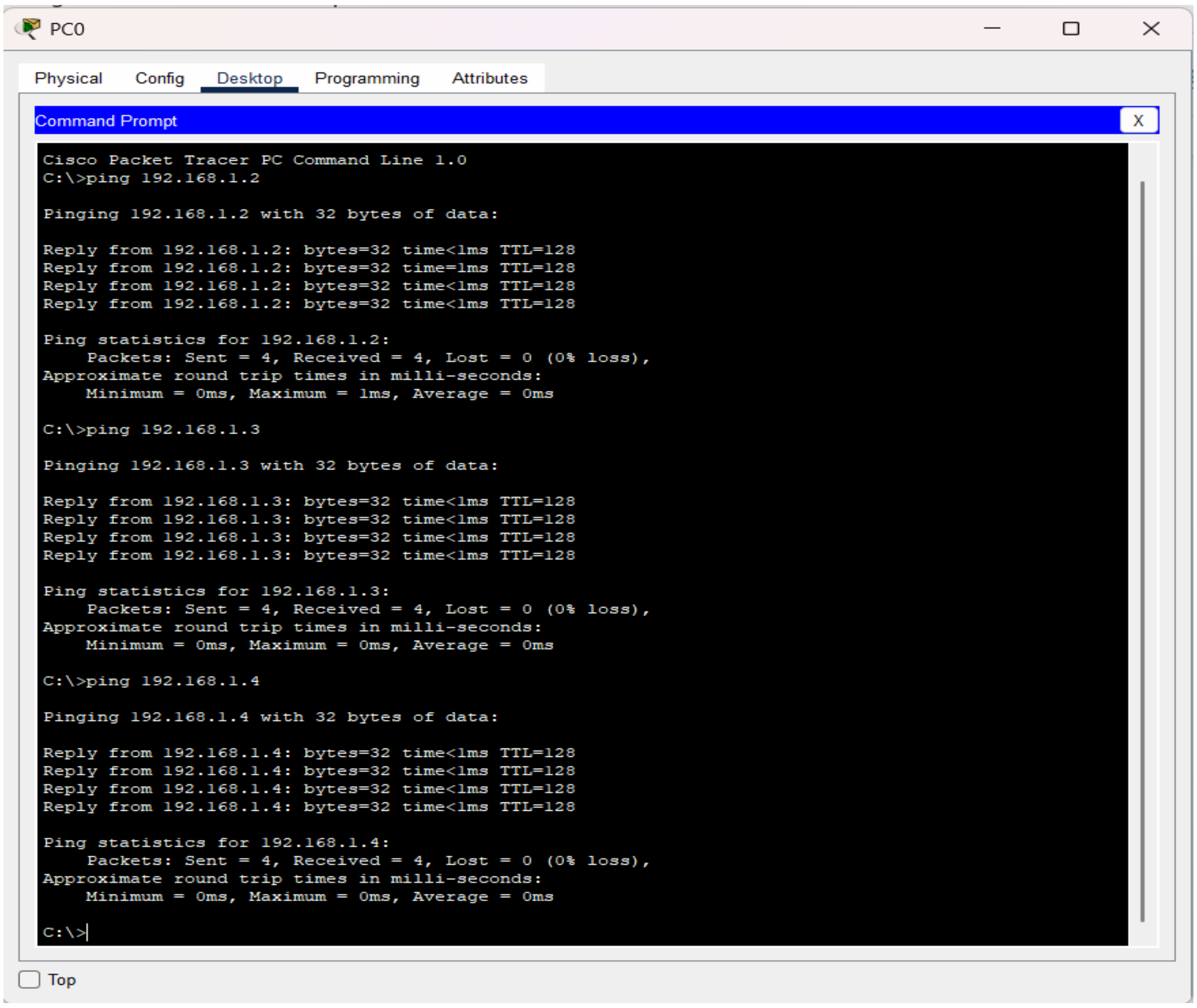
802.1X

Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

Top

7. **Testing:**
   - Open **Command Prompt** on PC0 → Type `ping 192.168.1.2` and check connectivity.
   - Repeat for all PCs.

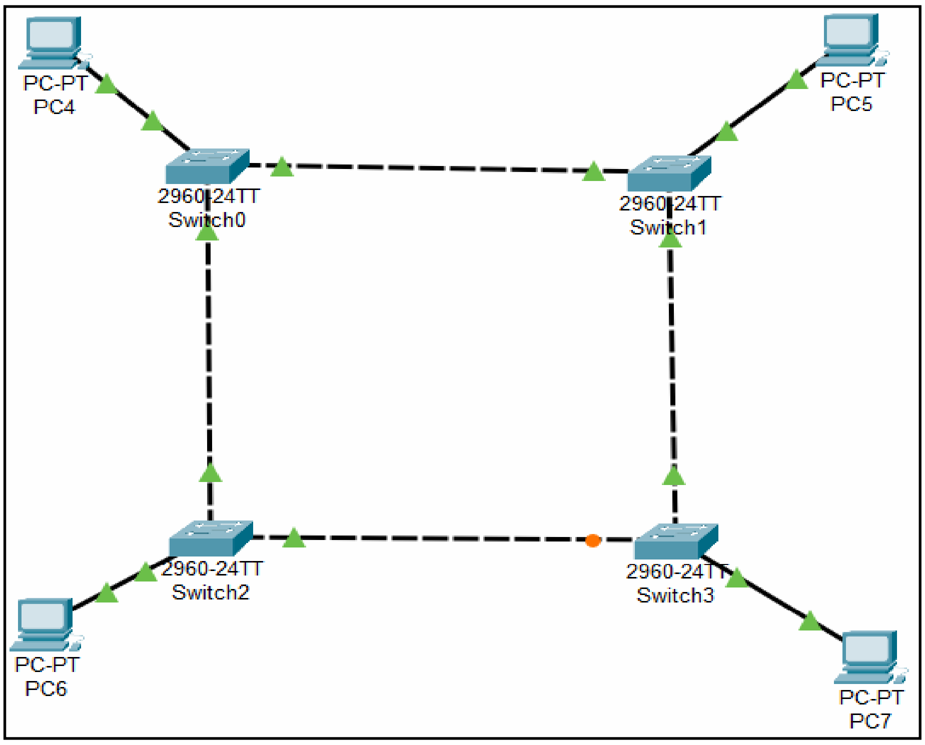


```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.2

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 = 1ms, Average = 0ms

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 = 0ms, Average = 0ms

C:\>ping 192.168.1.4

Pinging 192.168.1.4 with 32 bytes of data:

Reply from 192.168.1.4: bytes=32 time<1ms TTL=128
Reply from 192.168.1.4: bytes=32 time<1ms TTL=128
Reply from 192.168.1.4: bytes=32 time<1ms TTL=128
Reply from 192.168.1.4: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

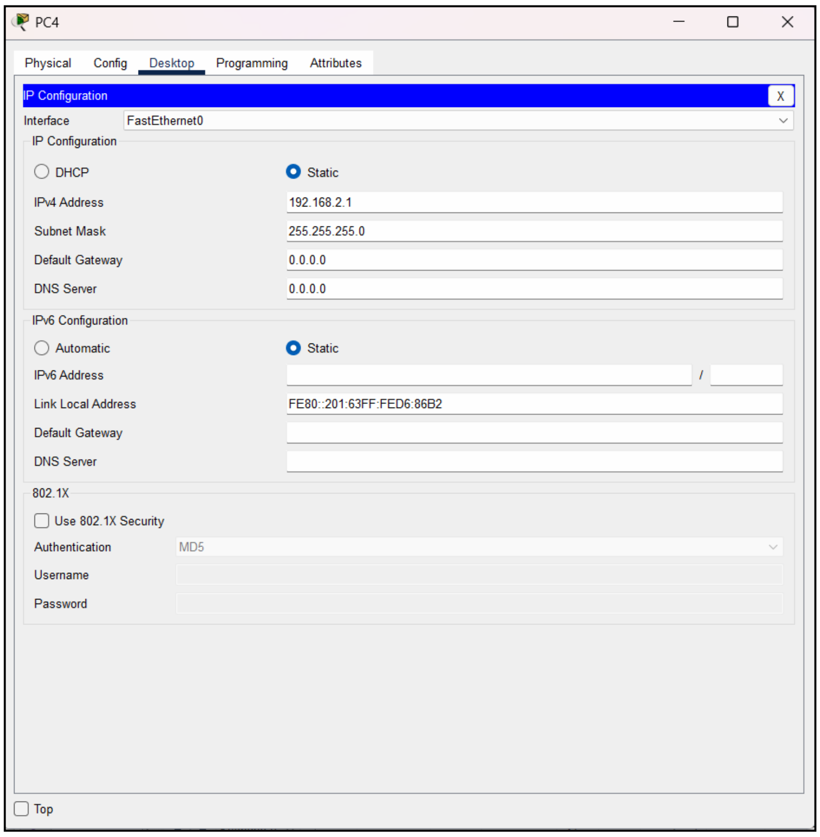
## 2. Ring Topology

### Steps:

1. **Add 4 PCs** (PC0 to PC3).
2. **No Hub/Switch** is required.
3. Arrange them in a circle.
4. **Connections:**
    - Use **Cross-Over Cable**.
    - Connect as follows:
        - PC0 ↔ PC1
        - PC1 ↔ PC2
        - PC2 ↔ PC3
        - PC3 ↔ PC0 (to complete the ring)



5. **IP Configuration:**
   - Assign similar IPs:
     - PC0: 192.168.2.1 / 255.255.255.0
     - PC1: 192.168.2.2 / 255.255.255.0
     - PC2: 192.168.2.3 / 255.255.255.0
     - PC3: 192.168.2.4 / 255.255.255.0



Interface: FastEthernet0

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 |

IPv6 Configuration

Automatic [✓ Static]

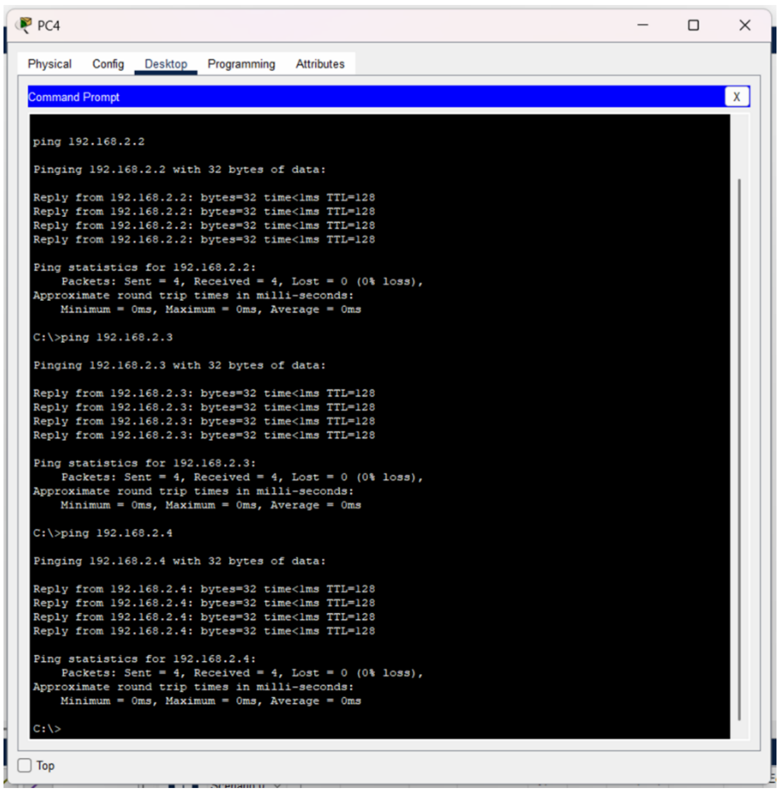
| | |
|---|---|
| IPv6 Address | / |
| Link Local Address | FE80::201:63FF:FED6:86B2 |
| Default Gateway | |
| DNS Server | |

802.1X

Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

6. **Testing:**
   - Ping all PCs from PC0 and verify connectivity.

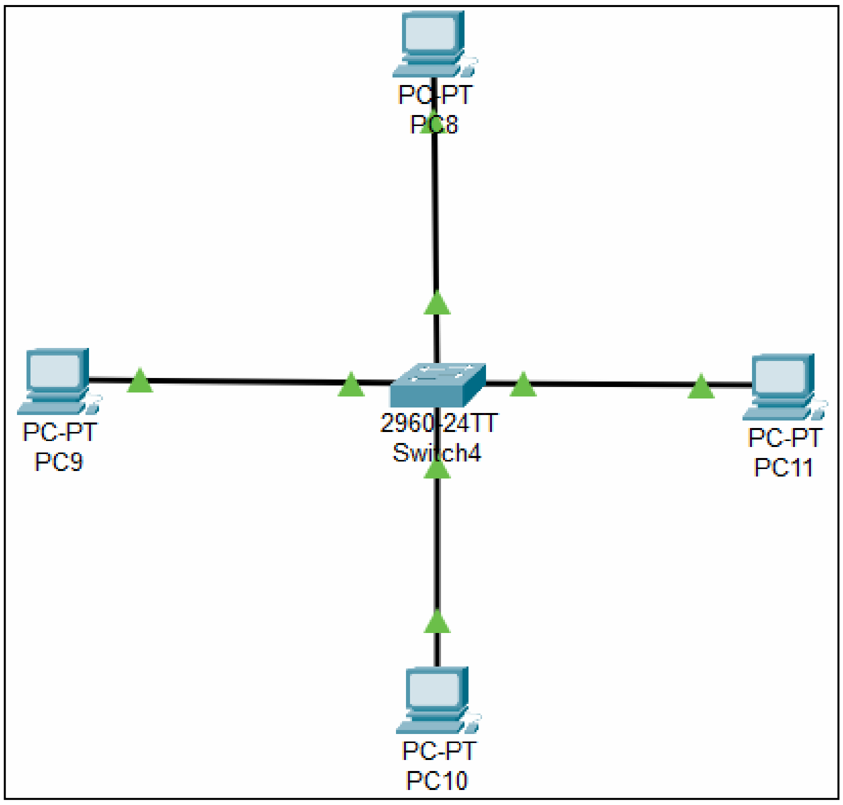


```
ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time<1ms TTL=128
Reply from 192.168.2.2: bytes=32 time<1ms TTL=128
Reply from 192.168.2.2: bytes=32 time<1ms TTL=128
Reply from 192.168.2.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.2.3

Pinging 192.168.2.3 with 32 bytes of data:

Reply from 192.168.2.3: bytes=32 time<1ms TTL=128
Reply from 192.168.2.3: bytes=32 time<1ms TTL=128
Reply from 192.168.2.3: bytes=32 time<1ms TTL=128
Reply from 192.168.2.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.2.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.2.4

Pinging 192.168.2.4 with 32 bytes of data:

Reply from 192.168.2.4: bytes=32 time<1ms TTL=128
Reply from 192.168.2.4: bytes=32 time<1ms TTL=128
Reply from 192.168.2.4: bytes=32 time<1ms TTL=128
Reply from 192.168.2.4: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.2.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

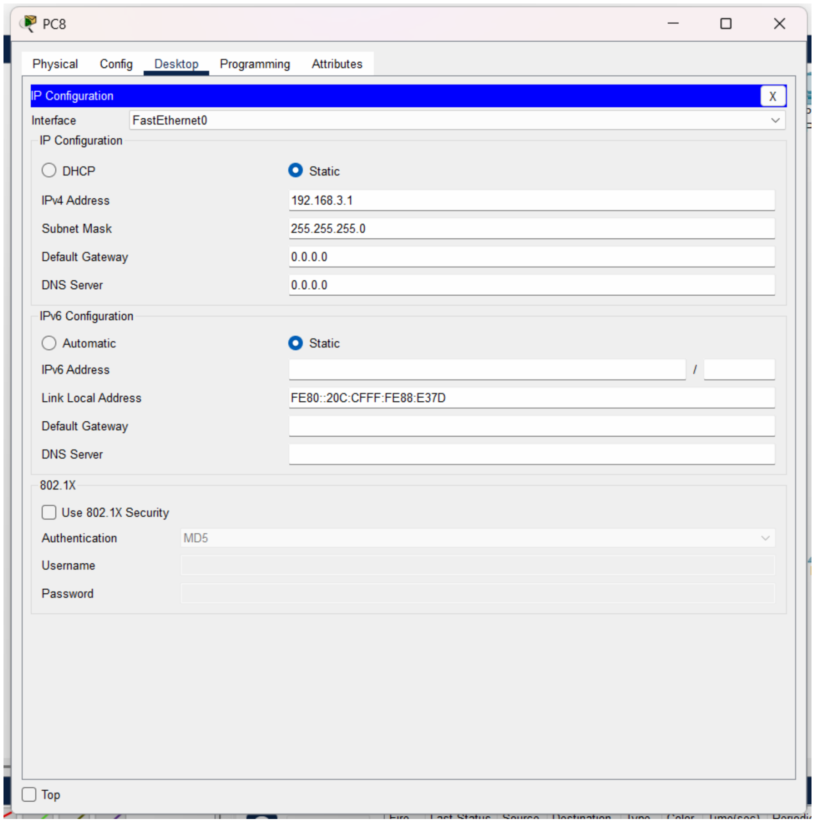
# 3. Star Topology

## Steps:

1. **Add 4 PCs** and **1 Switch**.
2. Arrange PCs around the switch in star shape.
3. **Connections:**
   - Use **Straight-Through Cable**.
   - Connect each PC to the **Switch**.



4. **IP Configuration:**
    - PC0: 192.168.3.1 / 255.255.255.0
    - PC1: 192.168.3.2 / 255.255.255.0
    - PC2: 192.168.3.3 / 255.255.255.0
    - PC3: 192.168.3.4 / 255.255.255.0



Interface FastEthernet0

IP Configuration

DHCP [✓ Static]

IPv4 Address 192.168.3.1
Subnet Mask 255.255.255.0
Default Gateway 0.0.0.0
DNS Server 0.0.0.0

IPv6 Configuration

Automatic [✓ Static]

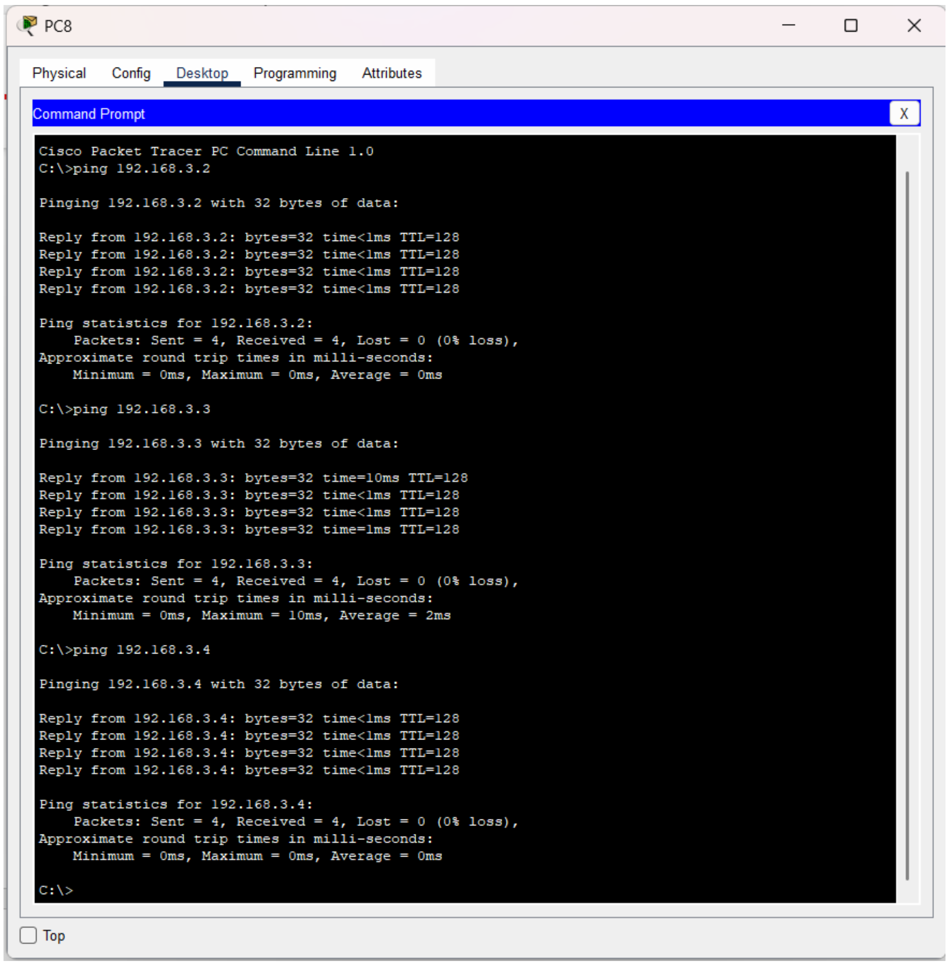
IPv6 Address /
Link Local Address FE80::20C:CFFF:FE88:E37D
Default Gateway
DNS Server

802.1X

Use 802.1X Security
Authentication MD5
Username
Password

Top

5. **Testing:**
   - Ping from PC0 to others to verify connectivity.

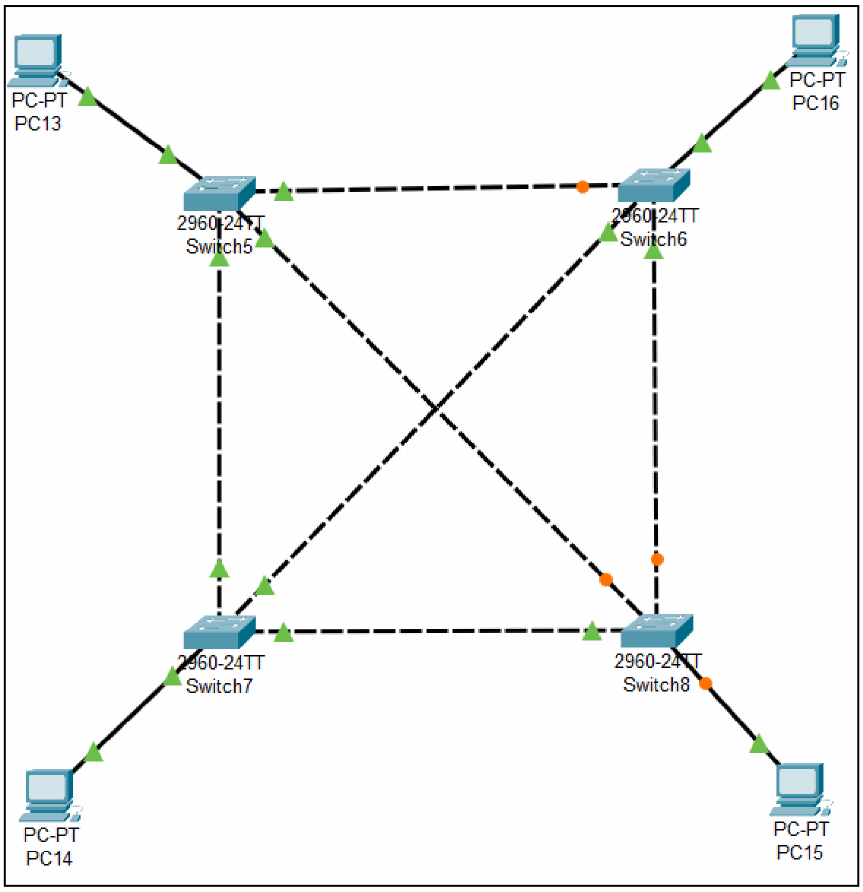


```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.3.2

Pinging 192.168.3.2 with 32 bytes of data:

Reply from 192.168.3.2: bytes=32 time<1ms TTL=128
Reply from 192.168.3.2: bytes=32 time<1ms TTL=128
Reply from 192.168.3.2: bytes=32 time<1ms TTL=128
Reply from 192.168.3.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.3.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.3.3

Pinging 192.168.3.3 with 32 bytes of data:

Reply from 192.168.3.3: bytes=32 time=10ms TTL=128
Reply from 192.168.3.3: bytes=32 time<1ms TTL=128
Reply from 192.168.3.3: bytes=32 time<1ms TTL=128
Reply from 192.168.3.3: bytes=32 time=1ms TTL=128

Ping statistics for 192.168.3.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 10ms, Average = 2ms

C:\>ping 192.168.3.4

Pinging 192.168.3.4 with 32 bytes of data:

Reply from 192.168.3.4: bytes=32 time<1ms TTL=128
Reply from 192.168.3.4: bytes=32 time<1ms TTL=128
Reply from 192.168.3.4: bytes=32 time<1ms TTL=128
Reply from 192.168.3.4: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.3.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

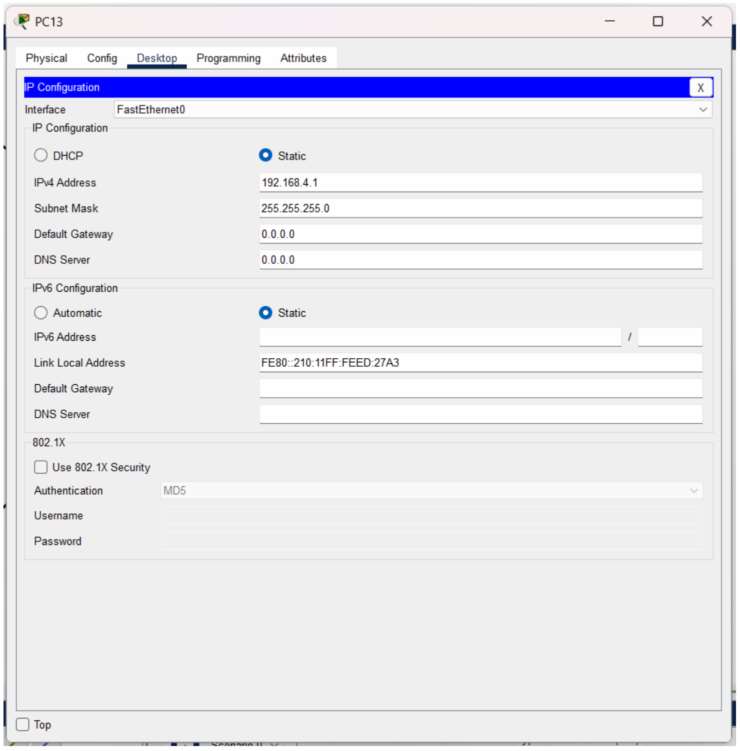
# 4. Mesh Topology

## Steps:

1. **Add 4 PCs**.
2. In mesh topology, **each device is connected to every other device**.
3. **Connections:**
   - Use **Cross-Over Cable**.
   - Connect each pair:
     - PC0 ↔ PC1
     - PC0 ↔ PC2
     - PC0 ↔ PC3
     - PC1 ↔ PC2
     - PC1 ↔ PC3
     - PC2 ↔ PC3



4. **IP Configuration (Manual/Custom):**
   - Each PC can have **multiple IPs on multiple interfaces** (Advanced).
   - For simple simulation, use routers or multi-interface PCs (optional).
   - Alternatively, simulate using **Switch and routers for scalability**.



Interface FastEthernet0

IP Configuration

DHCP [✓ Static]

IPv4 Address 192.168.4.1

Subnet Mask 255.255.255.0

Default Gateway 0.0.0.0

DNS Server 0.0.0.0

IPv6 Configuration

Automatic [✓ Static]

IPv6 Address /

Link Local Address FE80::210:11FF:FEED:27A3

Default Gateway

DNS Server

802.1X

☐ Use 802.1X Security

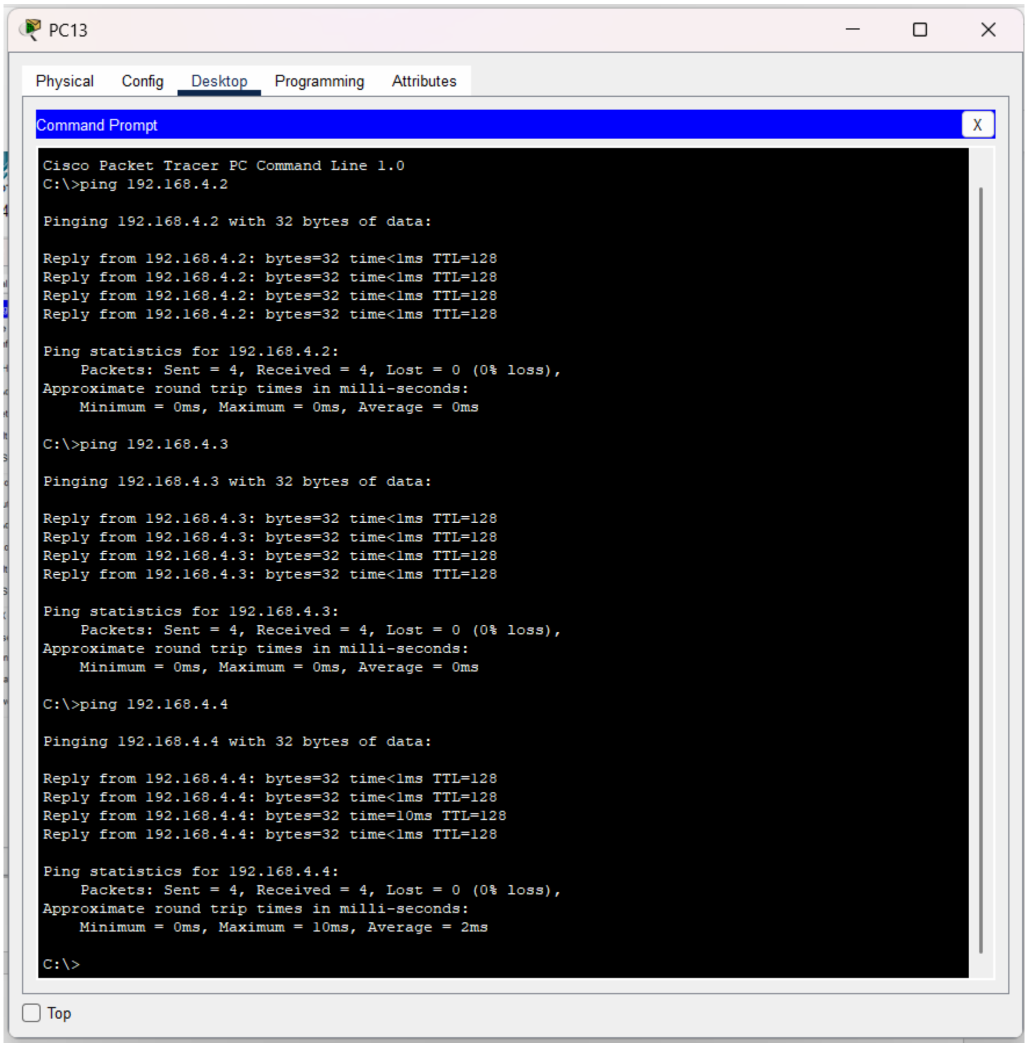
Authentication MD5

Username

Password

☐ Top

5. **Testing:**
    - Ping between all possible pairs.



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.4.2

Pinging 192.168.4.2 with 32 bytes of data:

Reply from 192.168.4.2: bytes=32 time<1ms TTL=128
Reply from 192.168.4.2: bytes=32 time<1ms TTL=128
Reply from 192.168.4.2: bytes=32 time<1ms TTL=128
Reply from 192.168.4.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.4.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.4.3

Pinging 192.168.4.3 with 32 bytes of data:

Reply from 192.168.4.3: bytes=32 time<1ms TTL=128
Reply from 192.168.4.3: bytes=32 time<1ms TTL=128
Reply from 192.168.4.3: bytes=32 time<1ms TTL=128
Reply from 192.168.4.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.4.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.4.4

Pinging 192.168.4.4 with 32 bytes of data:

Reply from 192.168.4.4: bytes=32 time<1ms TTL=128
Reply from 192.168.4.4: bytes=32 time<1ms TTL=128
Reply from 192.168.4.4: bytes=32 time=10ms TTL=128
Reply from 192.168.4.4: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.4.4:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 10ms, Average = 2ms

C:\>
```

## Comparison Table:

| **Topology** | **Devices Needed** | **Cabling Complexity** | **Central Device** | **Fault Tolerance** | **Scalability** | **Cost** |
|---|---|---|---|---|---|---|
| Bus | Few | Low | Hub | Low | Low | Low |
| Ring | Moderate | Moderate | No | Low | Moderate | Low |
| Star | Moderate | Low | Switch | High (central fails = network down) | High | Medium |
| Mesh | High | Very High | No | Very High | Low (for full mesh) | High |

## Conclusion:

In this experiment, we successfully designed and simulated four different network topologies **Bus, Ring, Star, and Mesh** — using Cisco Packet Tracer. Each topology demonstrated unique characteristics in terms of **structure, communication, fault tolerance, cost, and scalability**.

- **Bus topology** was simple and economical but suffered from high chances of **data collisions** and lacked fault tolerance.
- **Ring topology** ensured orderly data transmission but was vulnerable — a **single link failure** could disrupt the entire network.
- **Star topology** offered **efficient communication and scalability** with centralized control, but it heavily depended on the central switch.
- **Mesh topology** provided **maximum redundancy and reliability**, ideal for fault-tolerant systems, but required **complex wiring** and was **cost-intensive**.

Through this comparison, we observed that:

- **Star topology** is best suited for most practical LAN implementations due to its **balance of cost, performance, and reliability**.
- **Mesh topology** is preferable in **critical environments** where high **fault tolerance** is essential.
- **Bus and Ring** topologies are less commonly used today due to their **limited scalability** and **fragility**.

This experiment highlighted the importance of selecting an appropriate topology based on network requirements, budget constraints, and performance expectations.

**Experiment No: 03**

**Experiment Title: Setup and Use an FTP Server (FileZilla or Similar**

## Objective:

To install, configure, and use an FTP (File Transfer Protocol) server to transfer files between client and server over a network.

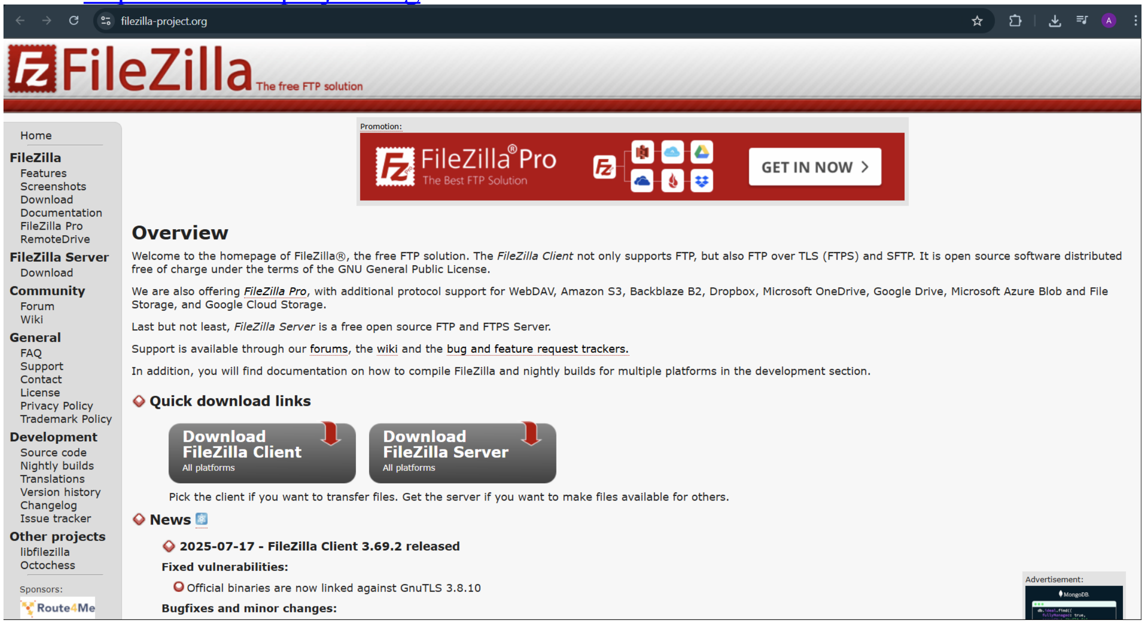
## Tools Required:

- FileZilla FTP Server (Windows) OR vsftpd (Linux)
- FileZilla FTP Client or web browser
- Two networked devices (can be PCs or VMs)
- Local network setup (LAN) or simulation using Cisco Packet Tracer with FTP-compatible OS (e.g., Server OS or simulation of FTP server on Packet Tracer)

## PART A: Setting Up FTP Server using FileZilla (Windows)

### Step 1: Download & Install FileZilla Server

1. Go to [https://filezilla-project.org](https://filezilla-project.org)



Overview

Welcome to the homepage of FileZilla®, the free FTP solution. The *FileZilla Client* not only supports FTP, but also FTP over TLS (FTPS) and SFTP. It is open source software distributed free of charge under the terms of the GNU General Public License.

We are also offering *FileZilla Pro*, with additional protocol support for WebDAV, Amazon S3, Backblaze B2, Dropbox, Microsoft OneDrive, Google Drive, Microsoft Azure Blob and File Storage, and Google Cloud Storage.

Last but not least, *FileZilla Server* is a free open source FTP and FTPS Server.

Support is available through our forums, the wiki and the bug and feature request trackers.

In addition, you will find documentation on how to compile FileZilla and nightly builds for multiple platforms in the development section.

Quick download links

Pick the client if you want to transfer files. Get the server if you want to make files available for others.

News

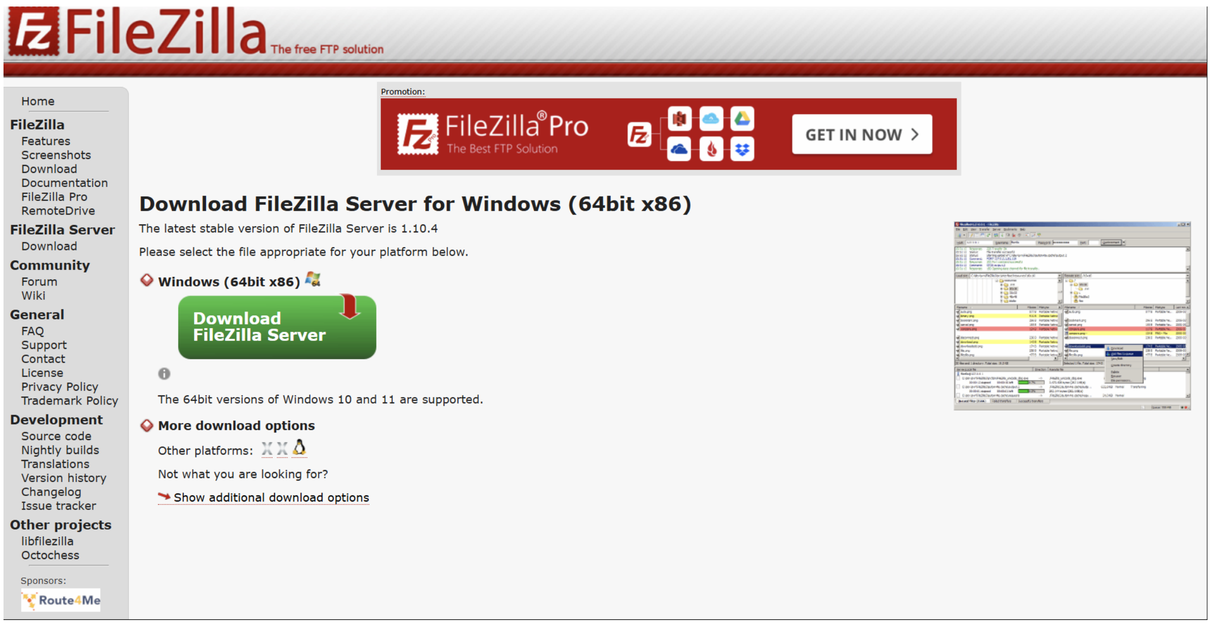
2025-07-17 - FileZilla Client 3.69.2 released

Fixed vulnerabilities:

Official binaries are now linked against GnuTLS 3.8.10

Bugfixes and minor changes:

2. Download **FileZilla Server** (not the client).



Download FileZilla Server for Windows (64bit x86)

The latest stable version of FileZilla Server is 1.10.4

Please select the file appropriate for your platform below.

Windows (64bit x86)

The 64bit versions of Windows 10 and 11 are supported.

More download options

Other platforms:

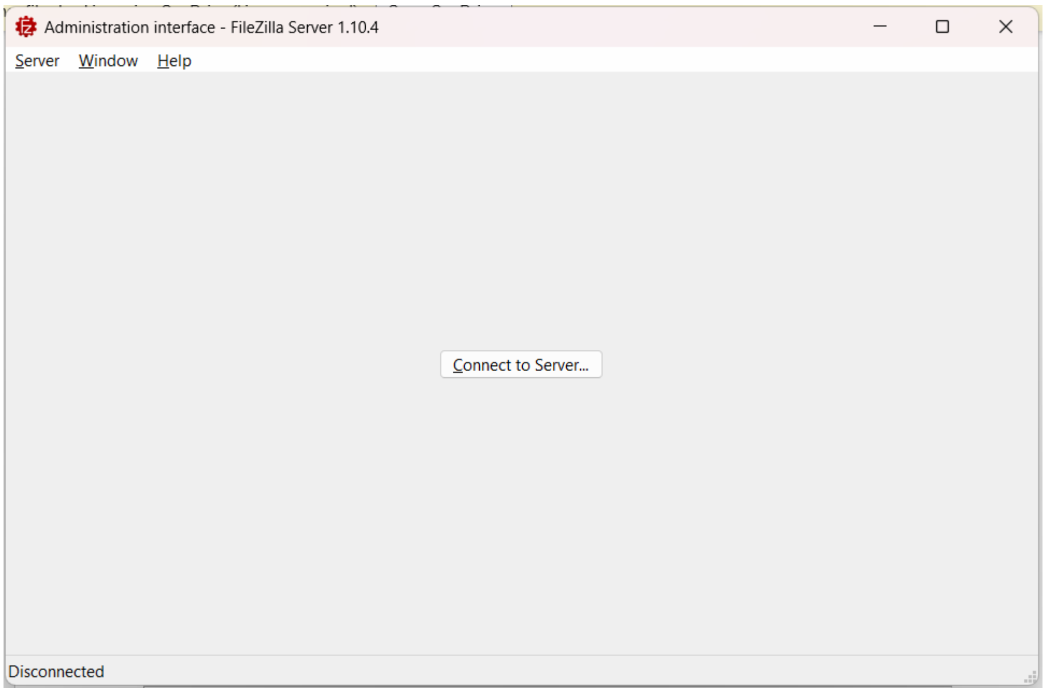
Not what you are looking for?

Show additional download options

3. Run the installer and follow the steps.
4. Choose default settings and start the service after installation.

## Step 2: Configure FTP Server

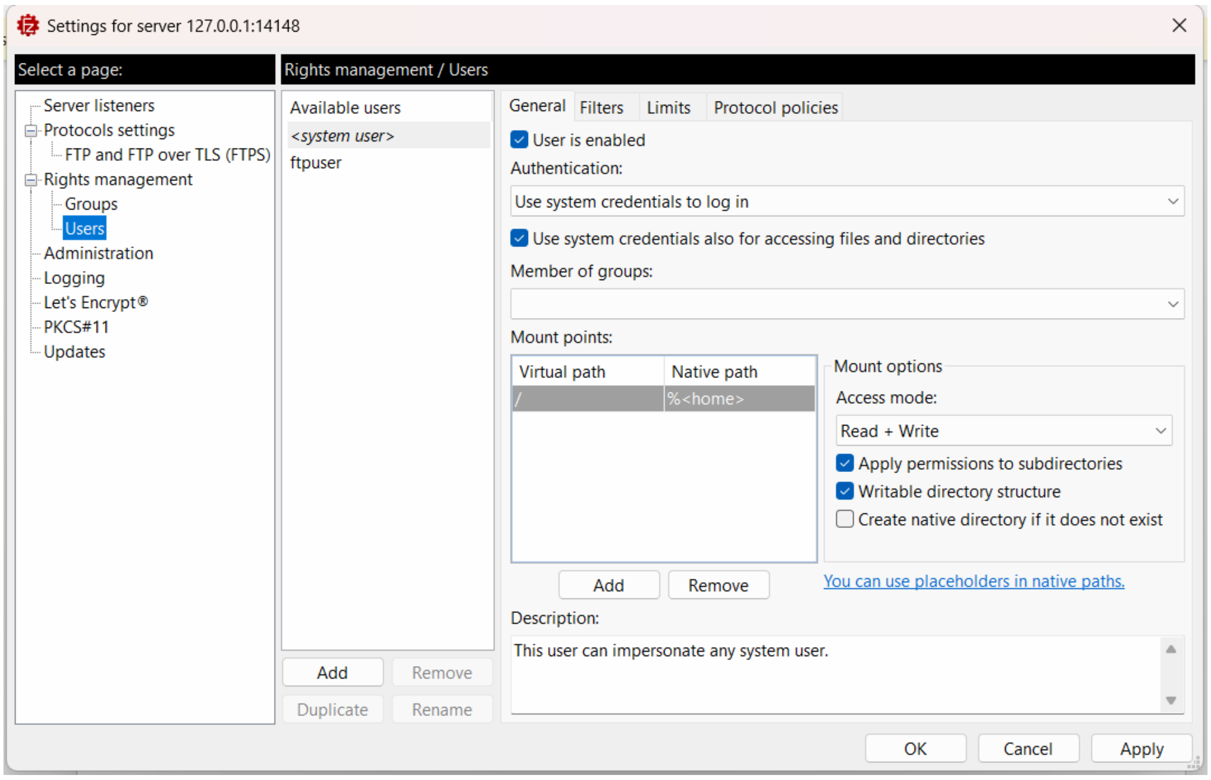
1. Launch **FileZilla Server Interface**



2. If prompted, connect to localhost (127.0.0.1), port 14147.

## Step 3: Create a User Account

1. Click on **"Edit"** → **"Users"**



Settings for server 127.0.0.1:14148

Select a page:
- Server listeners
- Protocols settings
  - FTP and FTP over TLS (FTPS)
- Rights management
  - Groups
  - Users
- Administration
- Logging
- Let's Encrypt®
- PKCS#11
- Updates

Rights management / Users

Available users: *<system user>*, ftpuser

General | Filters | Limits | Protocol policies

☑ User is enabled

Authentication: Use system credentials to log in

☑ Use system credentials also for accessing files and directories

Member of groups:

Mount points:

| Virtual path | Native path |
|---|---|
| / | %<home> |

Mount options

Access mode: Read + Write

☑ Apply permissions to subdirectories

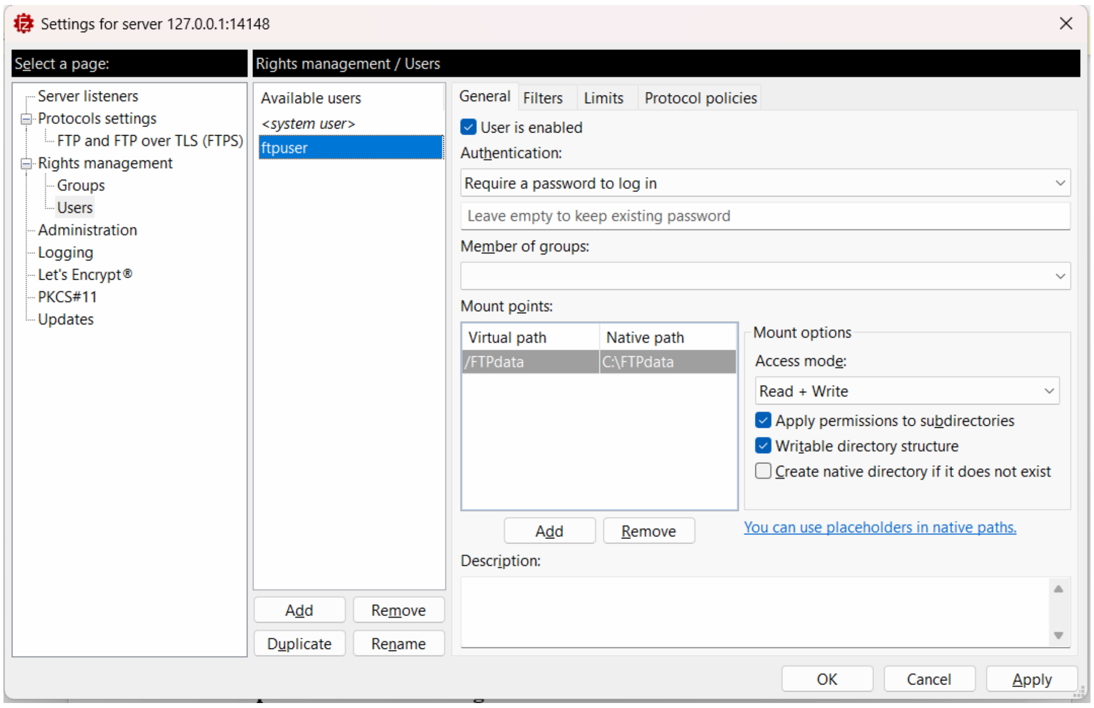
☑ Writable directory structure

☐ Create native directory if it does not exist

You can use placeholders in native paths.

Description: This user can impersonate any system user.

2. Click **"Add"** to create a new user (e.g., `ftpuser`)



Settings for server 127.0.0.1:14148

Select a page:
- Server listeners
- Protocols settings
  - FTP and FTP over TLS (FTPS)
- Rights management
  - Groups
  - Users
- Administration
- Logging
- Let's Encrypt®
- PKCS#11
- Updates

Rights management / Users

Available users: *<system user>*, ftpuser

General | Filters | Limits | Protocol policies

☑ User is enabled

Authentication: Require a password to log in

Leave empty to keep existing password

Member of groups:

Mount points:

| Virtual path | Native path |
|---|---|
| /FTPdata | C:\FTPdata |

Mount options

Access mode: Read + Write

☑ Apply permissions to subdirectories

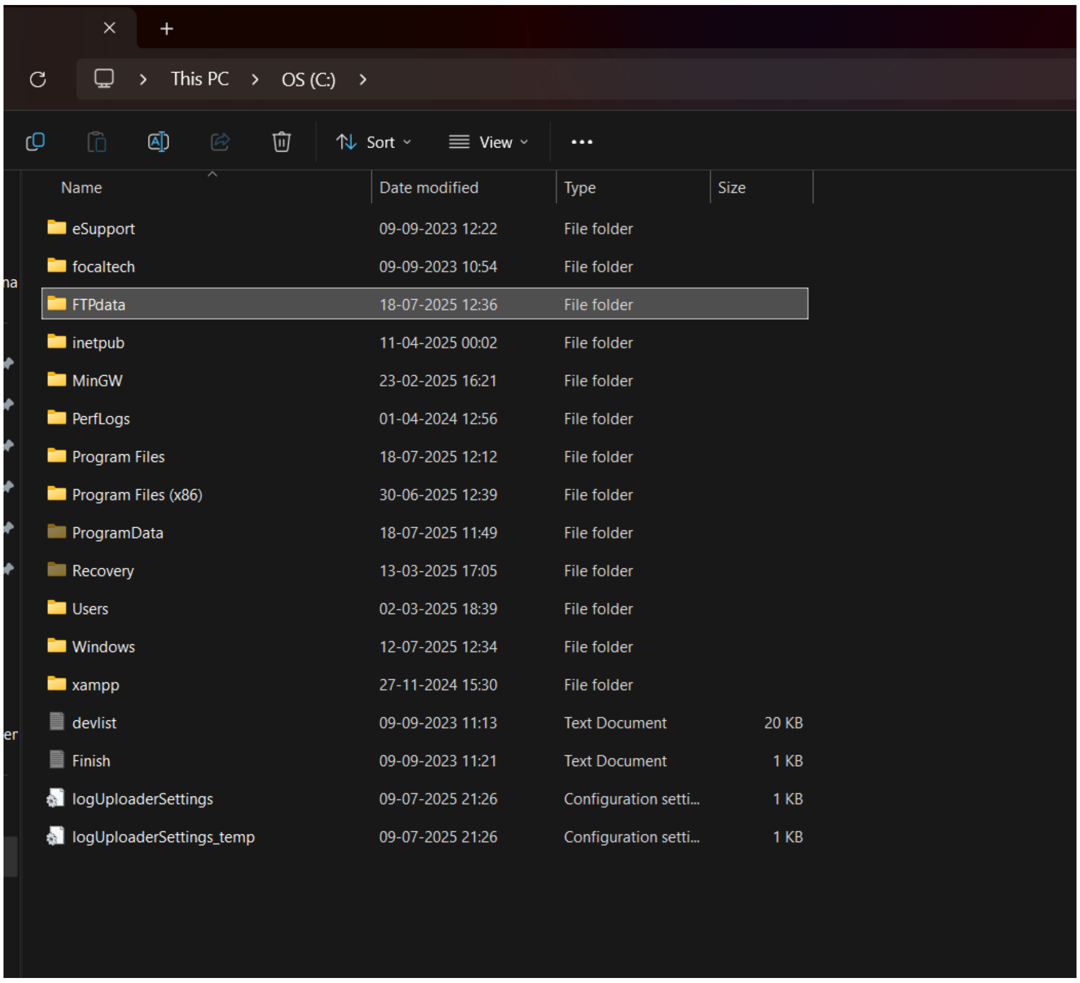
☑ Writable directory structure

☐ Create native directory if it does not exist

You can use placeholders in native paths.

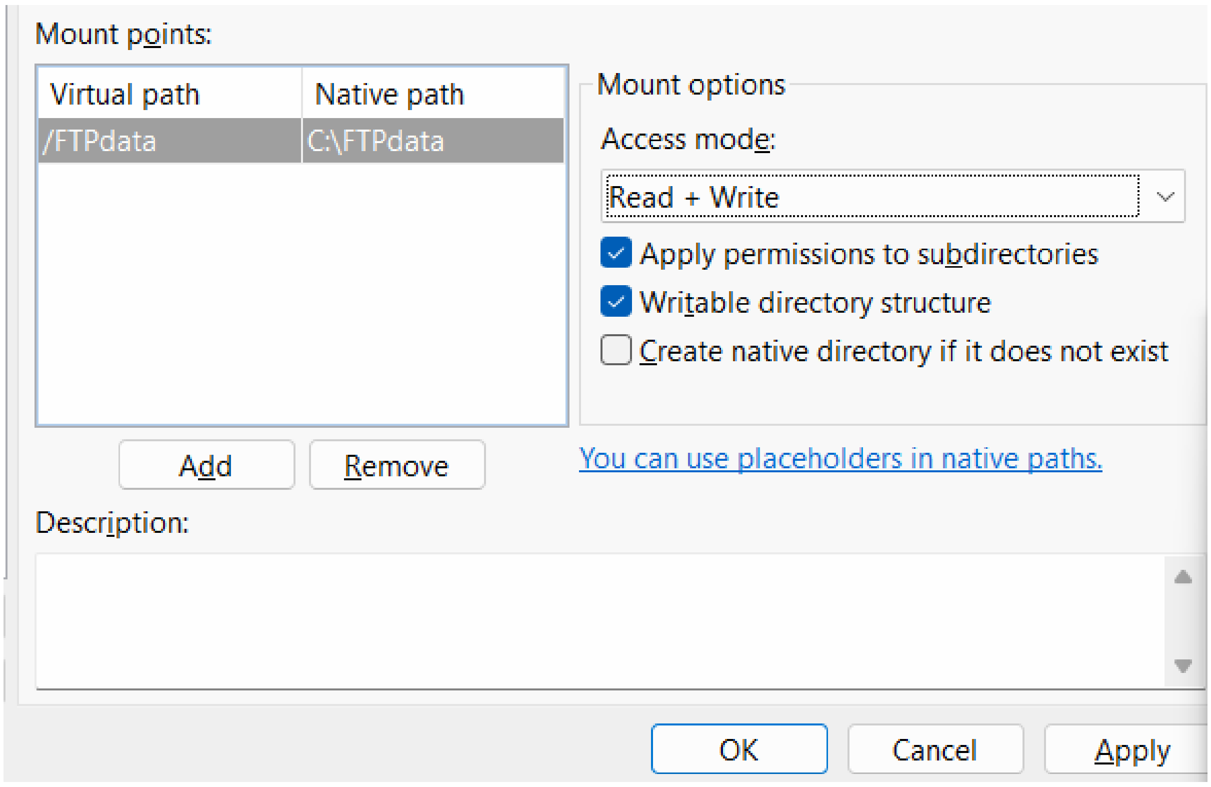
Description:

3. Click **"Shared folders"**


This PC > OS (C:) >

| Name | Date modified | Type | Size |
|---|---|---|---|
| eSupport | 09-09-2023 12:22 | File folder | |
| focaltech | 09-09-2023 10:54 | File folder | |
| FTPdata | 18-07-2025 12:36 | File folder | |
| inetpub | 11-04-2025 00:02 | File folder | |
| MinGW | 23-02-2025 16:21 | File folder | |
| PerfLogs | 01-04-2024 12:56 | File folder | |
| Program Files | 18-07-2025 12:12 | File folder | |
| Program Files (x86) | 30-06-2025 12:39 | File folder | |
| ProgramData | 18-07-2025 11:49 | File folder | |
| Recovery | 13-03-2025 17:05 | File folder | |
| Users | 02-03-2025 18:39 | File folder | |
| Windows | 12-07-2025 12:34 | File folder | |
| xampp | 27-11-2024 15:30 | File folder | |
| devlist | 09-09-2023 11:13 | Text Document | 20 KB |
| Finish | 09-09-2023 11:21 | Text Document | 1 KB |
| logUploaderSettings | 09-07-2025 21:26 | Configuration setti... | 1 KB |
| logUploaderSettings_temp | 09-07-2025 21:26 | Configuration setti... | 1 KB |

4. Click **"Add"** to select a folder for FTP access (e.g., `C:\FTP_Shared`)


Mount points:

| Virtual path | Native path |
|---|---|
| /FTPdata | C:\FTPdata |

Mount options

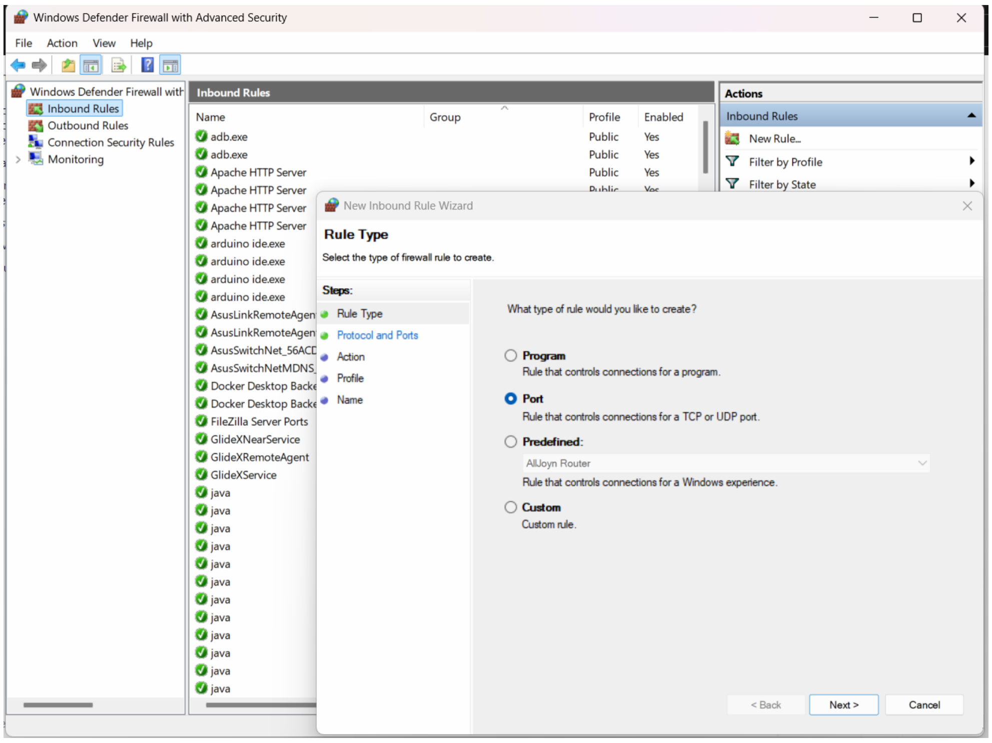
Access mode: Read + Write

- [✓] Apply permissions to subdirectories
- [✓] Writable directory structure
- [ ] Create native directory if it does not exist

5. Assign permissions like Read, Write, Delete, etc.

## Step 4: Allow FTP Through Firewall

1. Open **Control Panel → Windows Defender Firewall**



Inbound Rules

| Name | Group | Profile | Enabled |
|---|---|---|---|
| adb.exe | | Public | Yes |
| adb.exe | | Public | Yes |
| Apache HTTP Server | | Public | Yes |
| Apache HTTP Server | | Public | Yes |
| Apache HTTP Server | | | |
| Apache HTTP Server | | | |
| arduino ide.exe | | | |
| arduino ide.exe | | | |
| arduino ide.exe | | | |
| arduino ide.exe | | | |
| AsusLinkRemoteAgen | | | |
| AsusLinkRemoteAgen | | | |
| AsusSwitchNet_56ACD | | | |
| AsusSwitchNetMDNS_ | | | |
| Docker Desktop Backe | | | |
| Docker Desktop Backe | | | |
| FileZilla Server Ports | | | |
| GlideXNearService | | | |
| GlideXRemoteAgent | | | |
| GlideXService | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |
| java | | | |

New Inbound Rule Wizard

**Rule Type**

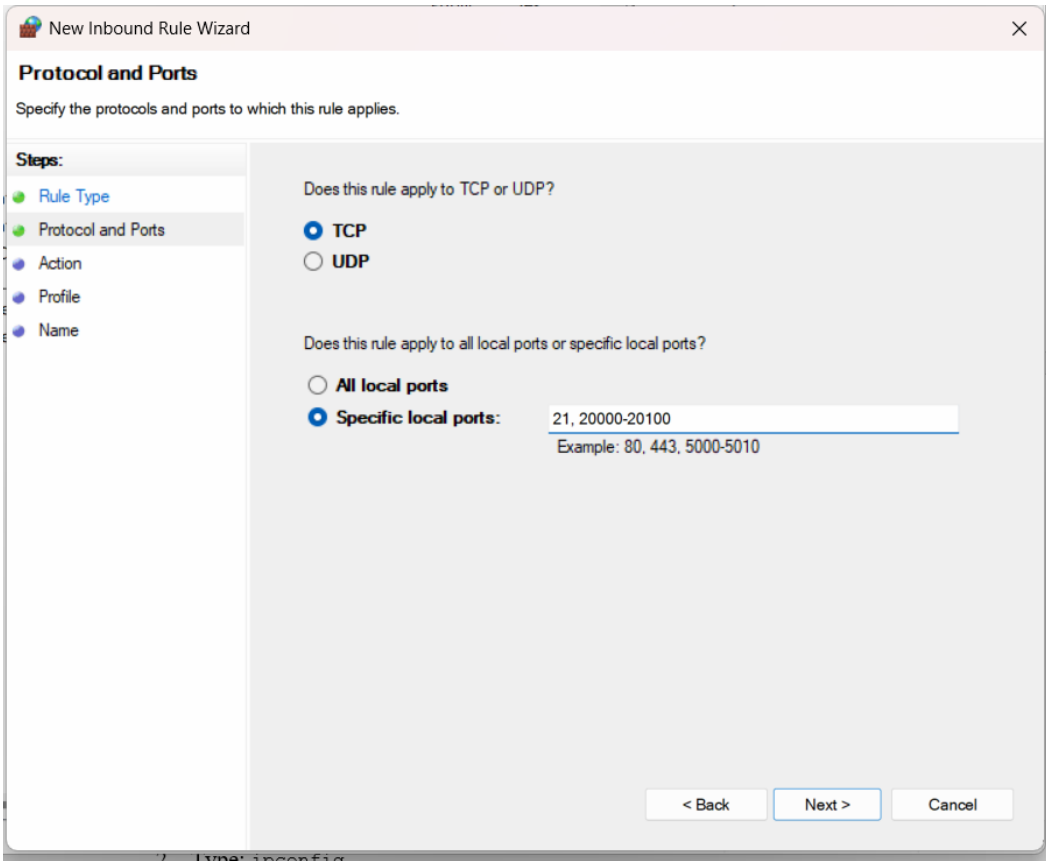
Select the type of firewall rule to create.

Steps:
- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

What type of rule would you like to create?

- ○ **Program** — Rule that controls connections for a program.
- [✓ **Port**] — Rule that controls connections for a TCP or UDP port.
- ○ **Predefined:** AllJoyn Router — Rule that controls connections for a Windows experience.
- ○ **Custom** — Custom rule.

2. Click on **"Allow an app through Windows Firewall"**
3. Add **FileZilla Server** to the list.



**Protocol and Ports**

Specify the protocols and ports to which this rule applies.

Does this rule apply to TCP or UDP?

[✓ TCP] UDP

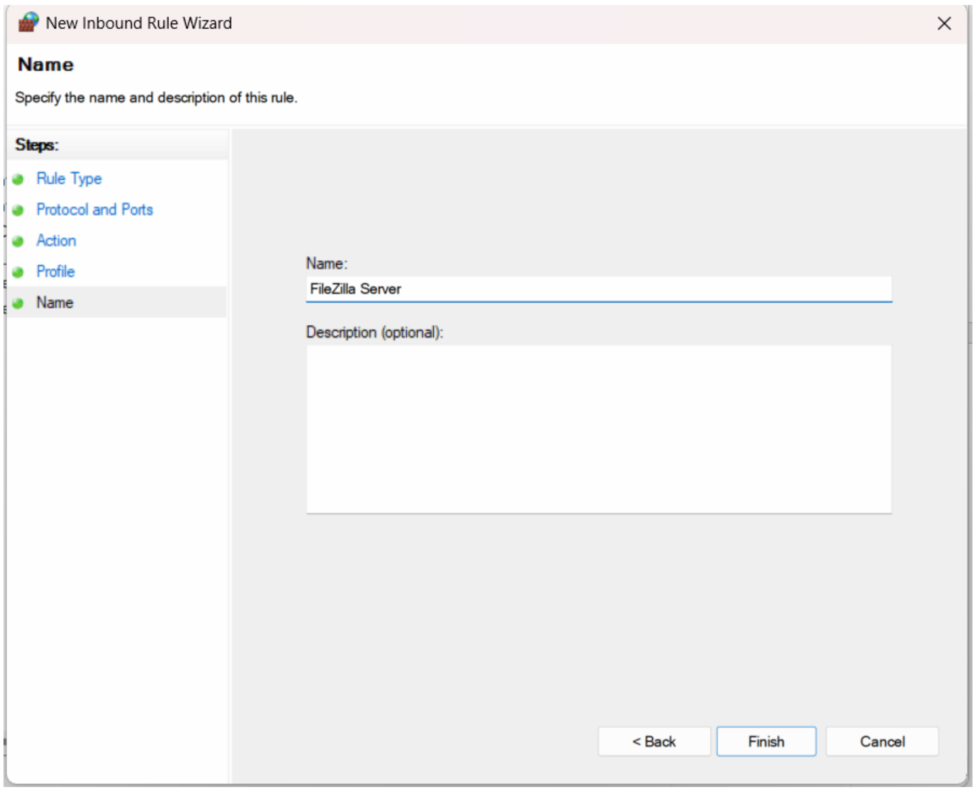
Does this rule apply to all local ports or specific local ports?

All local ports

[✓ Specific local ports:] 21, 20000-20100

Example: 80, 443, 5000-5010

4. Also, allow port **21** (default FTP port) through firewall if not already allowed.



**Name**

Specify the name and description of this rule.

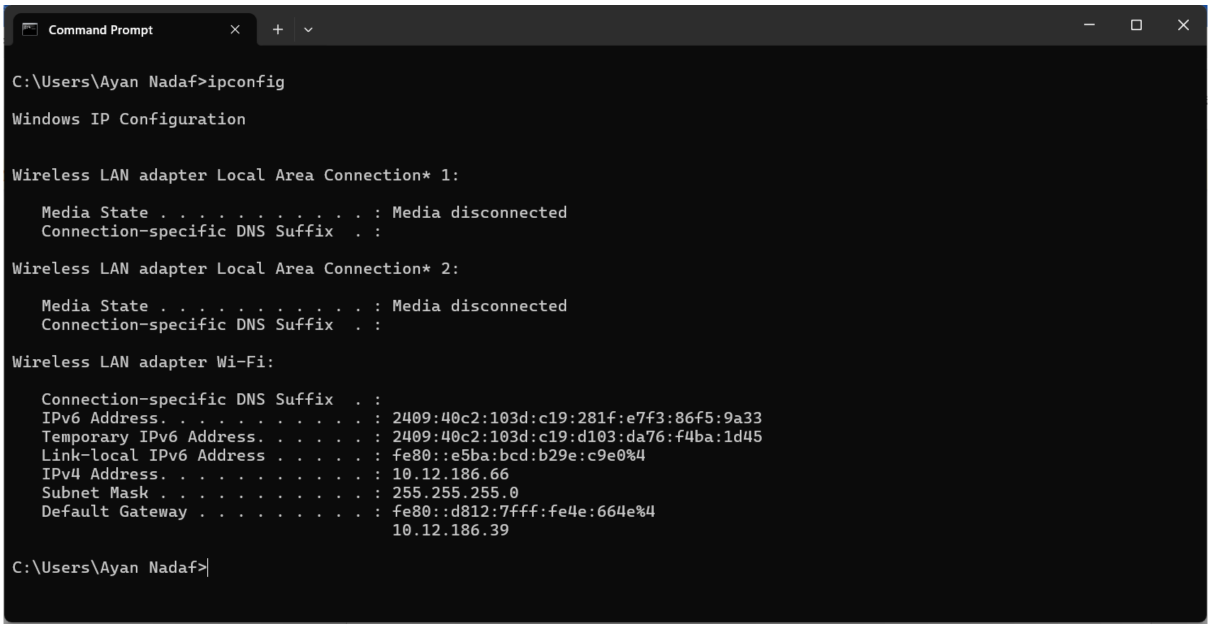
Name:

FileZilla Server

Description (optional):

### Step 5: Get the Server IP Address

1. Open **Command Prompt** on the server PC.
2. Type: `ipconfig`
3. Note the IPv4 address (e.g., `192.168.1.10`)

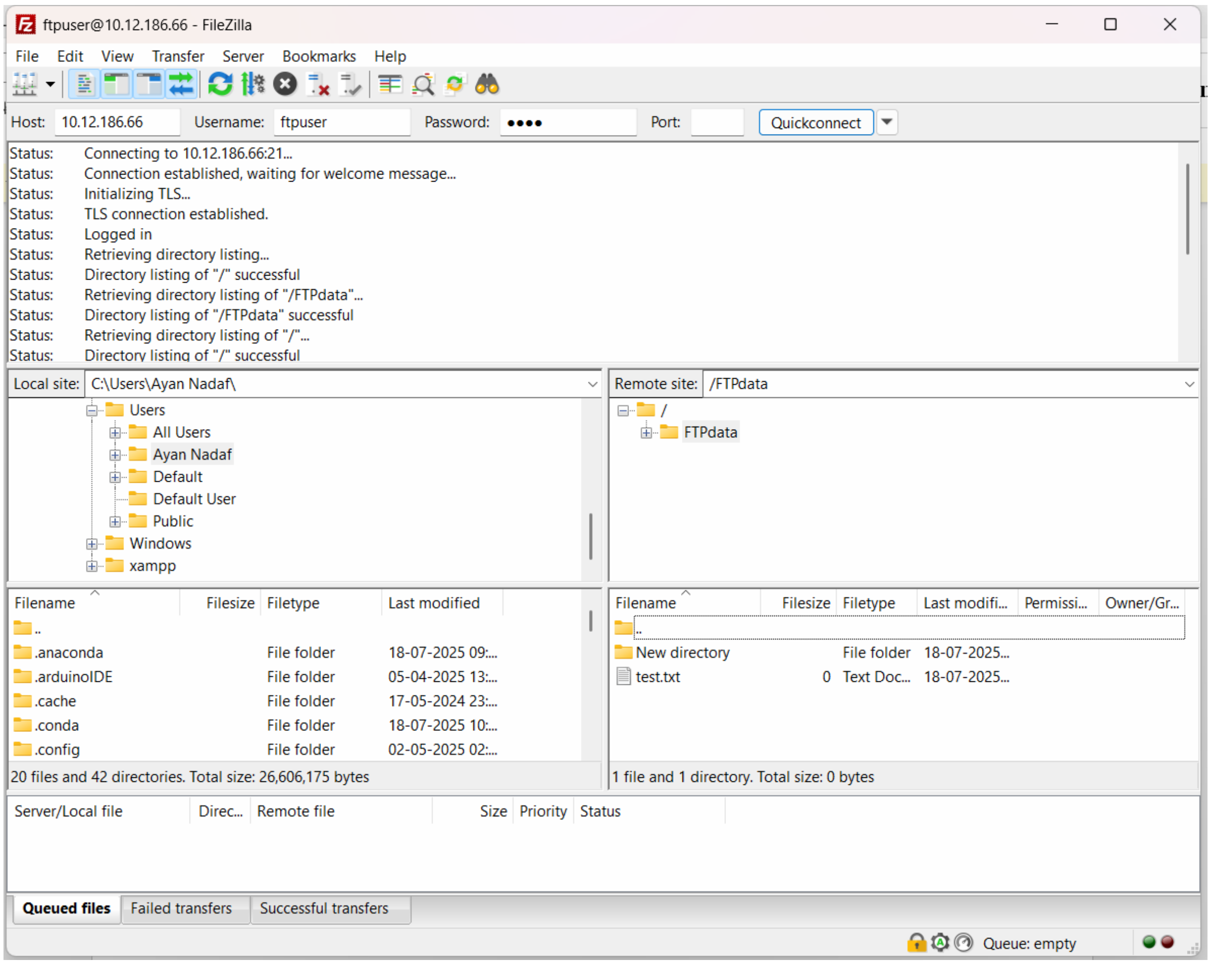


```
C:\Users\Ayan Nadaf>ipconfig

Windows IP Configuration


Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2409:40c2:103d:c19:281f:e7f3:86f5:9a33
   Temporary IPv6 Address. . . . . . : 2409:40c2:103d:c19:d103:da76:f4ba:1d45
   Link-local IPv6 Address . . . . . : fe80::e5ba:bcd:b29e:c9e0%4
   IPv4 Address. . . . . . . . . . . : 10.12.186.66
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::d812:7fff:fe4e:664e%4
                                       10.12.186.39

C:\Users\Ayan Nadaf>
```

## PART B: Access FTP Server from Client

### Option 1: Use FileZilla FTP Client

1. Download and install **FileZilla Client** on a different PC (client).
2. Open FileZilla Client and enter:
   - Host: 18.12.186.66 (server IP)
   - Username: `ftpuser`
   - Password: root (leave blank or use the one you created)
   - Port: `21`
3. Click **Quickconnect**


ftpuser@10.12.186.66 - FileZilla

Host: 10.12.186.66 Username: ftpuser Password: •••• Port: Quickconnect

```
Status:     Connecting to 10.12.186.66:21...
Status:     Connection established, waiting for welcome message...
Status:     Initializing TLS...
Status:     TLS connection established.
Status:     Logged in
Status:     Retrieving directory listing...
Status:     Directory listing of "/" successful
Status:     Retrieving directory listing of "/FTPdata"...
Status:     Directory listing of "/FTPdata" successful
Status:     Retrieving directory listing of "/"...
Status:     Directory listing of "/" successful
```

Local site: C:\Users\Ayan Nadaf\

- Users
  - All Users
  - Ayan Nadaf
  - Default
  - Default User
  - Public
- Windows
- xampp

Remote site: /FTPdata

- /
  - FTPdata

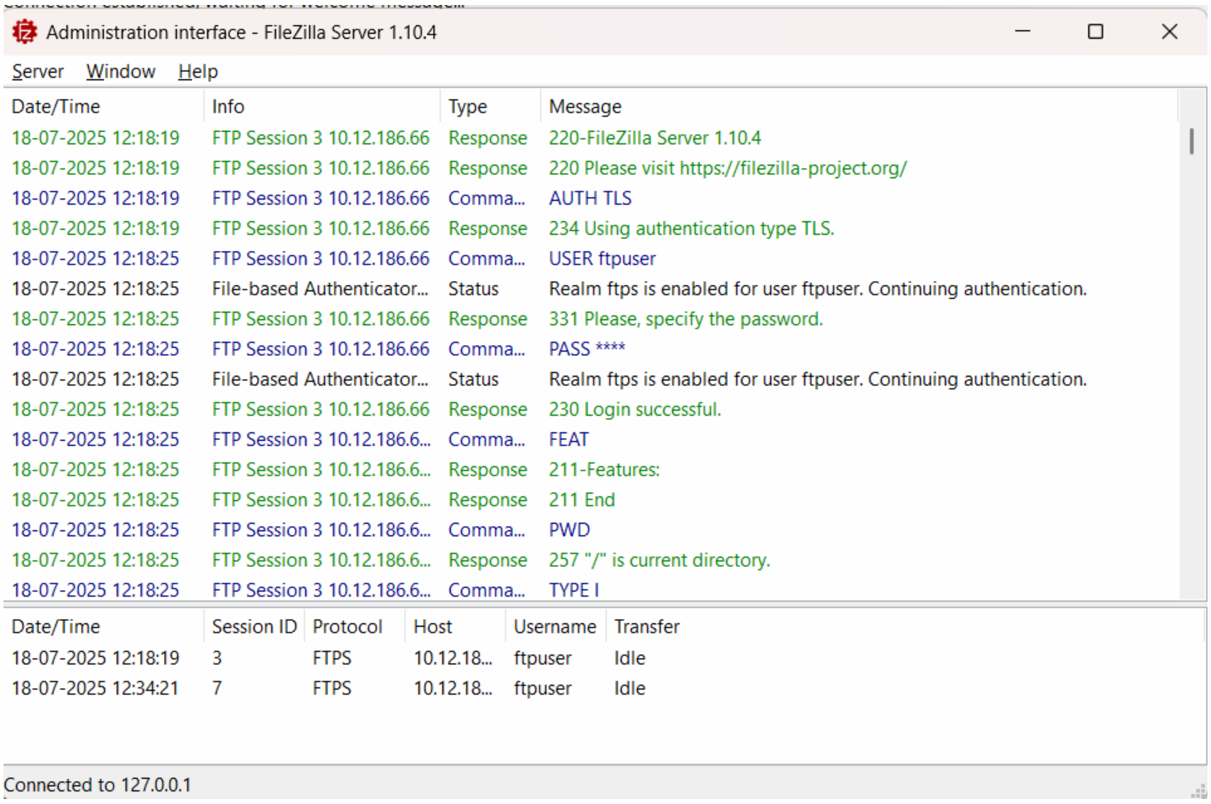
| Filename | Filesize | Filetype | Last modified |
|---|---|---|---|
| .. | | | |
| .anaconda | | File folder | 18-07-2025 09:... |
| .arduinoIDE | | File folder | 05-04-2025 13:... |
| .cache | | File folder | 17-05-2024 23:... |
| .conda | | File folder | 18-07-2025 10:... |
| .config | | File folder | 02-05-2025 02:... |

20 files and 42 directories. Total size: 26,606,175 bytes

| Filename | Filesize | Filetype | Last modifi... | Permissi... | Owner/Gr... |
|---|---|---|---|---|---|
| .. | | | | | |
| New directory | | File folder | 18-07-2025... | | |
| test.txt | 0 | Text Doc... | 18-07-2025... | | |

1 file and 1 directory. Total size: 0 bytes

| Server/Local file | Direc... | Remote file | Size | Priority | Status |
|---|---|---|---|---|---|


Administration interface - FileZilla Server 1.10.4

```
Date/Time             Info                          Type       Message
18-07-2025 12:18:19   FTP Session 3 10.12.186.66    Response   220-FileZilla Server 1.10.4
18-07-2025 12:18:19   FTP Session 3 10.12.186.66    Response   220 Please visit https://filezilla-project.org/
18-07-2025 12:18:19   FTP Session 3 10.12.186.66    Comma...   AUTH TLS
18-07-2025 12:18:19   FTP Session 3 10.12.186.66    Response   234 Using authentication type TLS.
18-07-2025 12:18:25   FTP Session 3 10.12.186.66    Comma...   USER ftpuser
18-07-2025 12:18:25   File-based Authenticator...   Status     Realm ftps is enabled for user ftpuser. Continuing authentication.
18-07-2025 12:18:25   FTP Session 3 10.12.186.66    Response   331 Please, specify the password.
18-07-2025 12:18:25   FTP Session 3 10.12.186.66    Comma...   PASS ****
18-07-2025 12:18:25   File-based Authenticator...   Status     Realm ftps is enabled for user ftpuser. Continuing authentication.
18-07-2025 12:18:25   FTP Session 3 10.12.186.66    Response   230 Login successful.
18-07-2025 12:18:25   FTP Session 3 10.12.186.6...  Comma...   FEAT
18-07-2025 12:18:25   FTP Session 3 10.12.186.6...  Response   211-Features:
18-07-2025 12:18:25   FTP Session 3 10.12.186.6...  Response   211 End
18-07-2025 12:18:25   FTP Session 3 10.12.186.6...  Comma...   PWD
18-07-2025 12:18:25   FTP Session 3 10.12.186.6...  Response   257 "/" is current directory.
18-07-2025 12:18:25   FTP Session 3 10.12.186.6...  Comma...   TYPE I
```

| Date/Time | Session ID | Protocol | Host | Username | Transfer |
|---|---|---|---|---|---|
| 18-07-2025 12:18:19 | 3 | FTPS | 10.12.18... | ftpuser | Idle |
| 18-07-2025 12:34:21 | 7 | FTPS | 10.12.18... | ftpuser | Idle |

Connected to 127.0.0.1

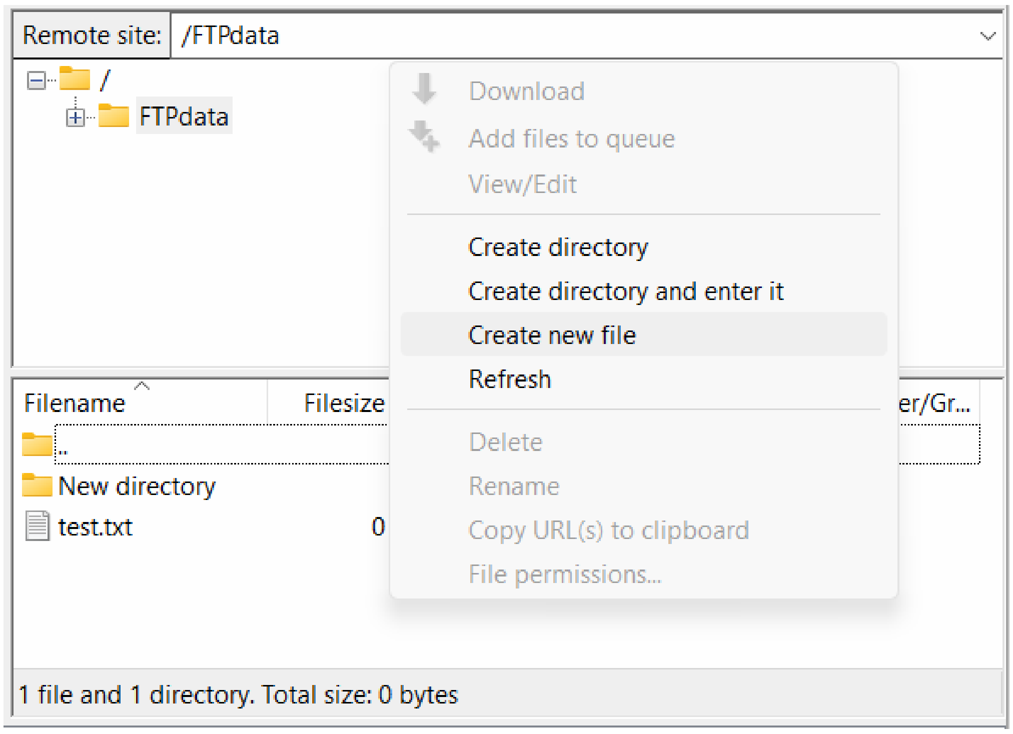
### Option 2: Use Web Browser

1. Open any browser.
2. In the address bar, type:
   `ftp://` 18.12.186.66
3. When prompted, enter username and password.

## PART C: Upload and Download Files

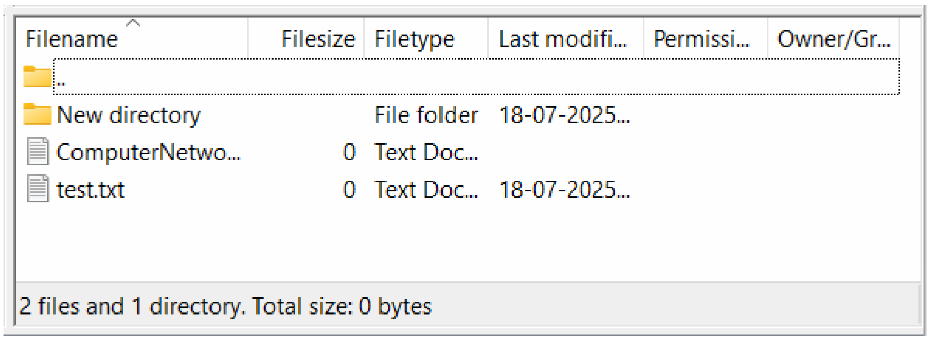
### To Upload:

1. On FileZilla Client, locate your local files (left panel)



| Filename | Filesize |
|---|---|
| .. | |
| New directory | |
| test.txt | 0 |

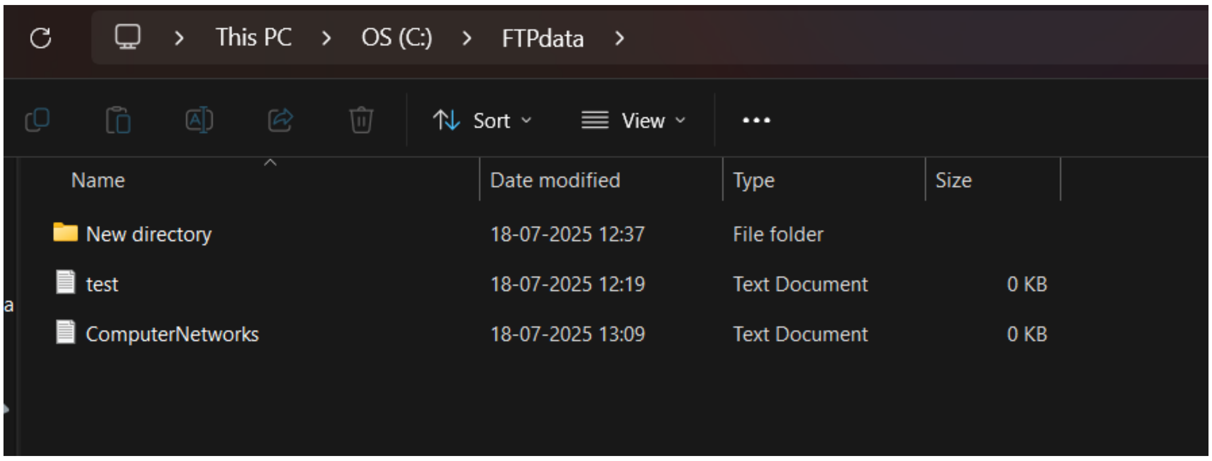
1 file and 1 directory. Total size: 0 bytes



| Filename | Filesize | Filetype | Last modifi... | Permissi... | Owner/Gr... |
|---|---|---|---|---|---|
| .. | | | | | |
| New directory | | File folder | 18-07-2025... | | |
| ComputerNetwo... | 0 | Text Doc... | | | |
| test.txt | 0 | Text Doc... | 18-07-2025... | | |

2 files and 1 directory. Total size: 0 bytes

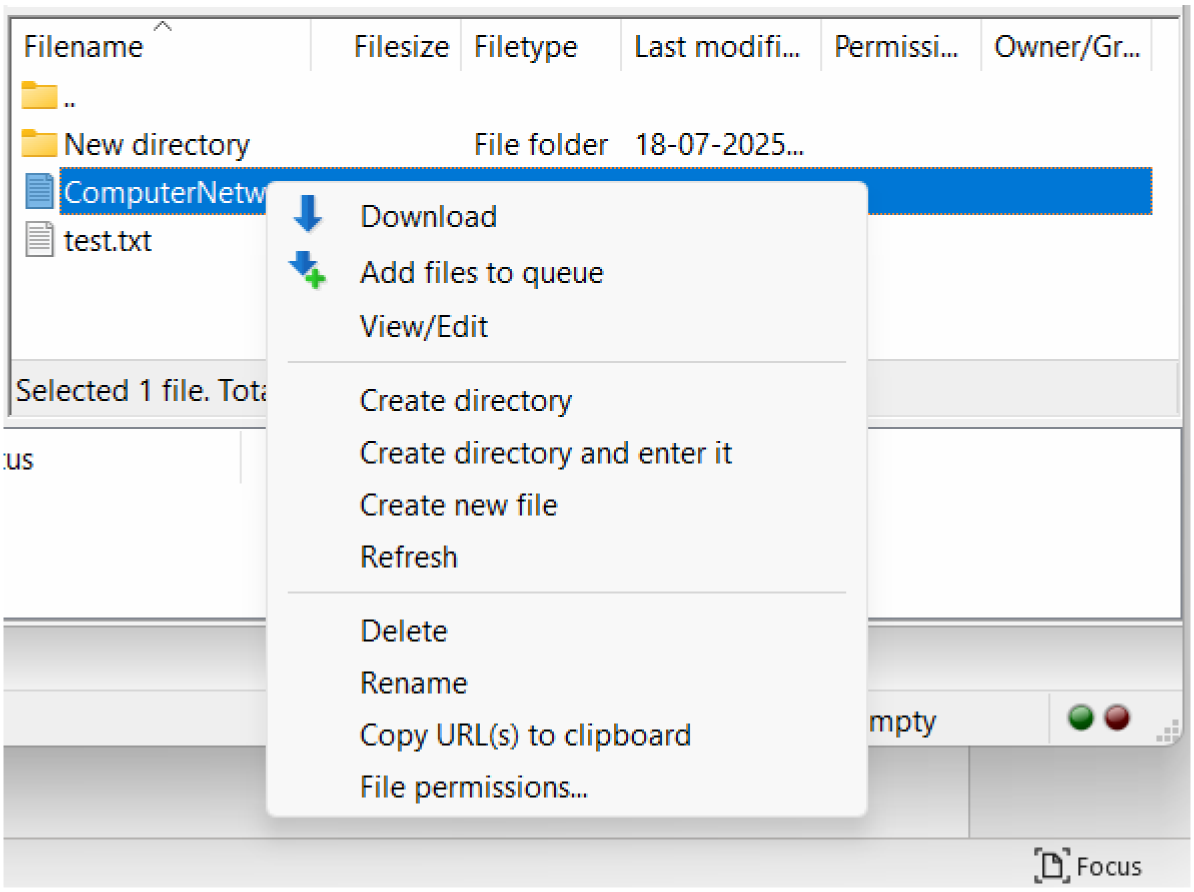
2. Drag and drop files into the remote folder (right panel)


This PC > OS (C:) > FTPdata >

| Name | Date modified | Type | Size |
|---|---|---|---|
| New directory | 18-07-2025 12:37 | File folder | |
| test | 18-07-2025 12:19 | Text Document | 0 KB |
| ComputerNetworks | 18-07-2025 13:09 | Text Document | 0 KB |

## To Download:

1. On the right panel (remote server), right-click any file
2. Choose **Download** to save it to the client system



| Filename | Filesize | Filetype | Last modifi... | Permissi... | Owner/Gr... |
|---|---|---|---|---|---|
| .. | | | | | |
| New directory | | File folder | 18-07-2025... | | |
| ComputerNetw | | | | | |
| test.txt | | | | | |

Selected 1 file. Tota

## Screenshots to Include in Assignment Report

1. FileZilla Server interface
2. User account setup
3. Shared folder configuration
4. FTP client connection window
5. Upload/download file confirmation
6. Successful connection logs

## Conclusion:

In this experiment, we successfully **installed, configured, and used an FTP (File Transfer Protocol) server** using **FileZilla Server** to simulate file sharing between a client and a server over a network. We created user accounts, defined access permissions, and transferred files using both **GUI-based FTP client** and **command-line utilities**.

This experiment provided practical understanding of:

- Setting up a real-world **FTP server environment**
- Using **port 21** for FTP communication
- Understanding how **file permissions and directories** are managed
- Performing basic file transfer operations such as **upload (PUT)** and **download (GET)**

By completing this task, we demonstrated the use of FTP as an **application layer protocol**, learned how clients and servers interact using predefined commands, and understood the importance of user authentication and network accessibility in file-sharing systems.

## Experiment No: 04

**Experiment Title:**

**Send/receive email via SMTP and POP3 by using telnet or a mail client.**

**Objective**

To configure and simulate the **sending and receiving of emails** using **SMTP** (for sending) and **POP3** (for receiving) through **Telnet** or a mail client in a simulated network.

**Course Outcome Mapped**

- **CO2**: Apply application and transport layer protocols to implement networking services.

## Tools Required

- Cisco Packet Tracer
- Email Server (built-in)
- PCs (at least 2: sender and receiver)
- Switch
- Copper Straight-through Cables
- Telnet Client (built-in in Packet Tracer)

## Step-by-Step Execution in Cisco Packet Tracer

### Step 1: Set Up the Devices

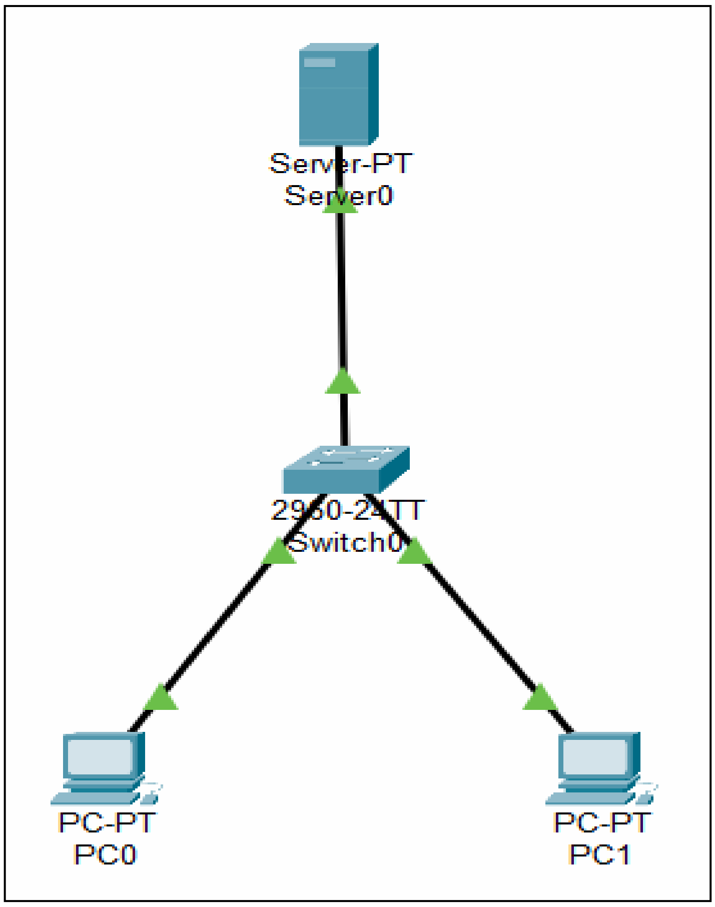
Add the following to the workspace:

- **1 Server**
- **2 PCs (PC0 = sender, PC1 = receiver)**
- **1 Switch**
- **Cables** (Copper Straight-through)

### Step 2: Connect the Devices

Use cables to connect:

- PC0 → Switch
- PC1 → Switch
- Server → Switch

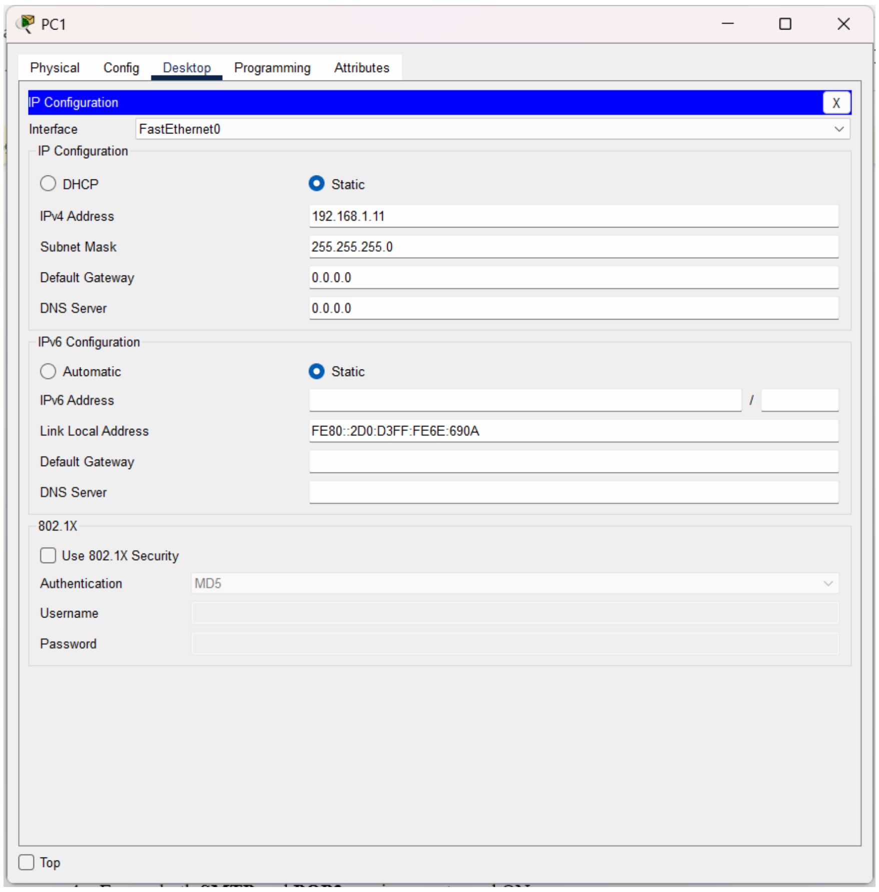


## Step 3: Assign IP Addresses

| Device | IP Address | Subnet Mask |
|---|---|---|
| PC0 | 192.168.1.10 | 255.255.255.0 |
| PC1 | 192.168.1.11 | 255.255.255.0 |
| Server | 192.168.1.5 | 255.255.255.0 |

Configure each device using:

**Desktop > IP Configuration**


PC1

Physical | Config | Desktop | Programming | Attributes

IP Configuration

| | |
|---|---|
| Interface | FastEthernet0 |

IP Configuration

DHCP [✓ Static]

| | |
|---|---|
| IPv4 Address | 192.168.1.11 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 0.0.0.0 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

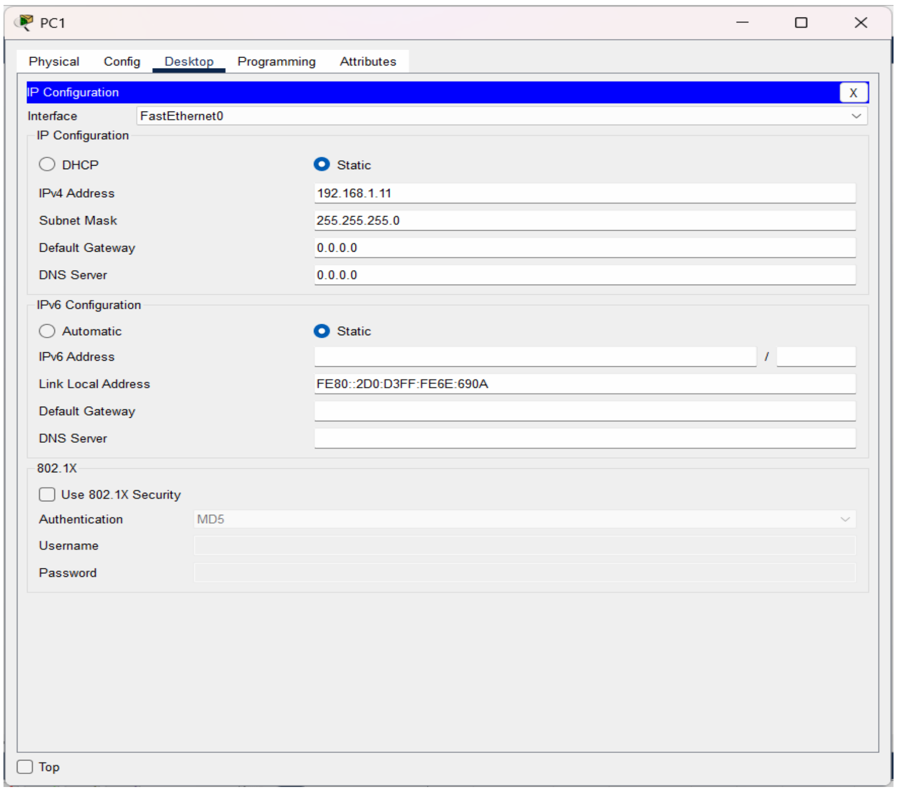
| | |
|---|---|
| IPv6 Address | / |
| Link Local Address | FE80::2D0:D3FF:FE6E:690A |
| Default Gateway | |
| DNS Server | |

802.1X

☐ Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

☐ Top


PC1

Physical | Config | Desktop | Programming | Attributes

IP Configuration

| | |
|---|---|
| Interface | FastEthernet0 |

IP Configuration

DHCP [✓ Static]

| | |
|---|---|
| IPv4 Address | 192.168.1.11 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 0.0.0.0 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

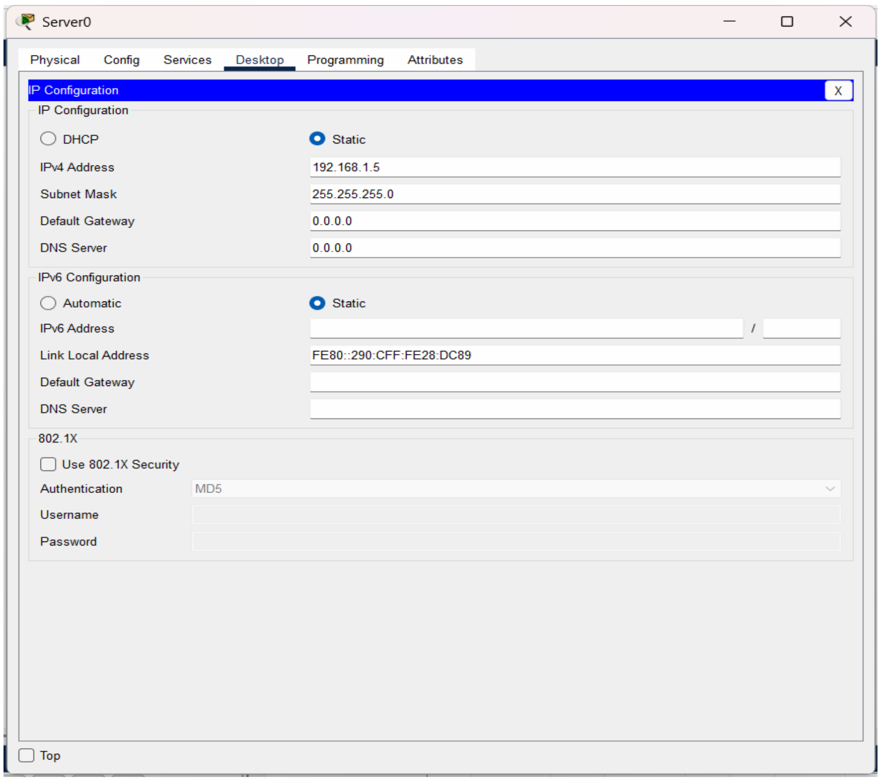
| | |
|---|---|
| IPv6 Address | / |
| Link Local Address | FE80::2D0:D3FF:FE6E:690A |
| Default Gateway | |
| DNS Server | |

802.1X

☐ Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

☐ Top



IP Configuration

○ DHCP ● Static

IPv4 Address: 192.168.1.5
Subnet Mask: 255.255.255.0
Default Gateway: 0.0.0.0
DNS Server: 0.0.0.0

IPv6 Configuration

○ Automatic ● Static

IPv6 Address: /
Link Local Address: FE80::290:CFF:FE28:DC89
Default Gateway:
DNS Server:

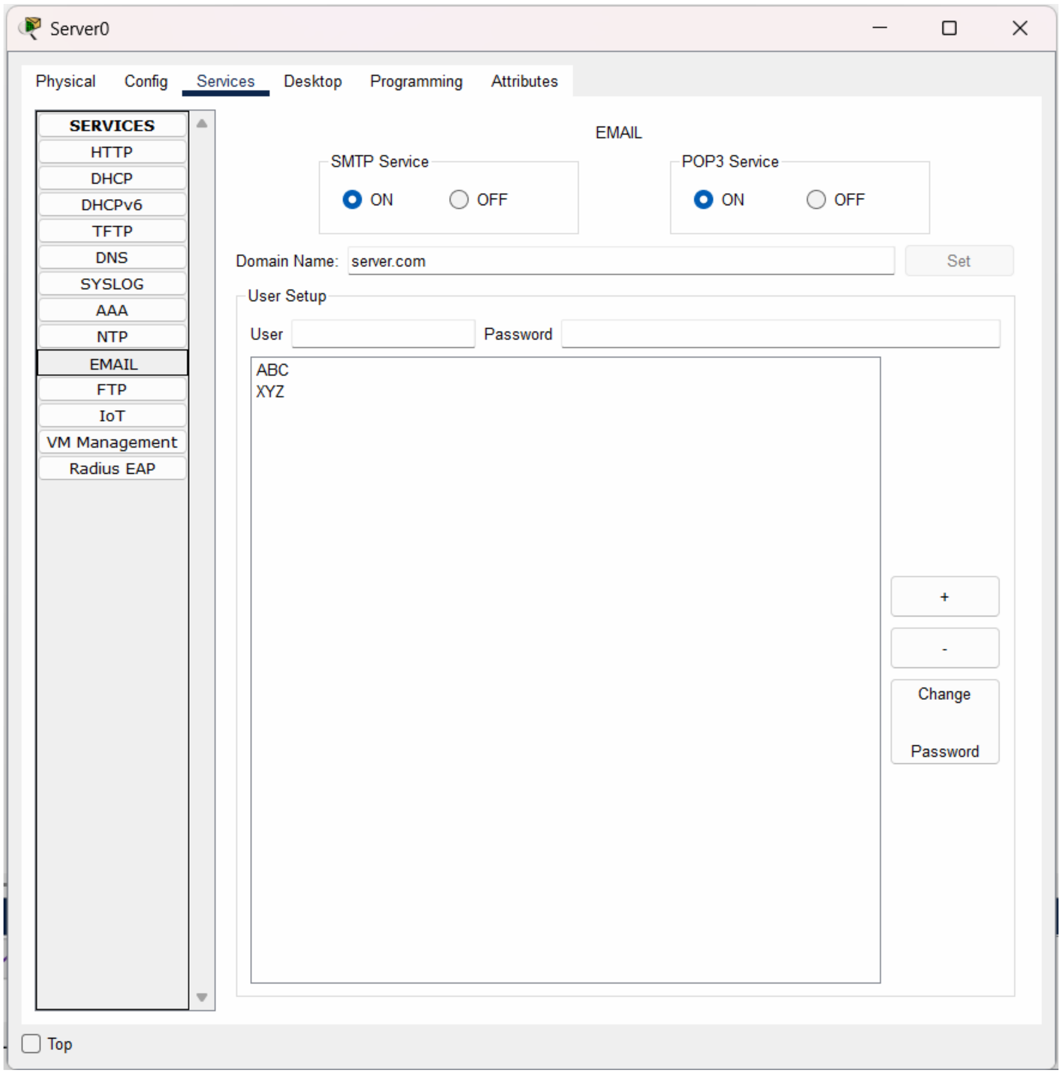
802.1X

☐ Use 802.1X Security
Authentication: MD5
Username:
Password:

## Step 4: Configure the Email Server

1. Click on the **Server**
2. Go to the **Services** tab
3. Click on **Email**
4. Ensure both **SMTP** and **POP3** services are turned ON
5. Add users:
    - Username: `ABC` | Password: `1234` | Email: `abc@server.com`
    - Username: `XYZ` | Password: `6789` | Email: `xyz@server.com`

Now, the server is ready to send and receive emails.



Server0

Physical | Config | Services | Desktop | Programming | Attributes

SERVICES: HTTP, DHCP, DHCPv6, TFTP, DNS, SYSLOG, AAA, NTP, EMAIL, FTP, IoT, VM Management, Radius EAP

EMAIL

SMTP Service: [✓ ON] OFF

POP3 Service: [✓ ON] OFF

Domain Name: server.com

User Setup

User | Password

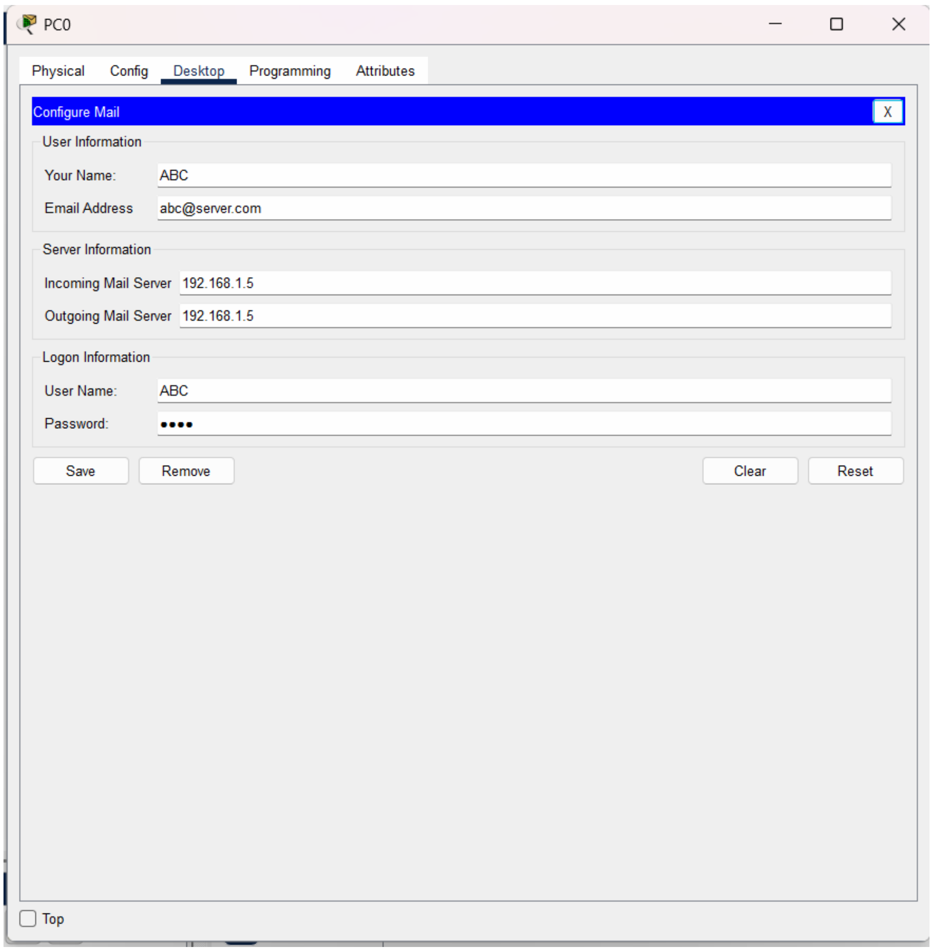
ABC
XYZ

# PART A: Send Email Using a Mail Client

## Step 1: Configure Email Client on PC0 (Sender)

1. Click **PC0 > Desktop > Email**
2. Fill out:
   - Display Name: ABC
   - Email: `abc@server.com`
   - Incoming Mail Server: **`192.168.1.5`**
   - Outgoing Mail Server: **`192.168.1.5`**
   - **Username: `ABC`**
   - **Password: `1234`**

Click **"Set"** to confirm configuration.



User Information
Your Name: ABC
Email Address: abc@server.com

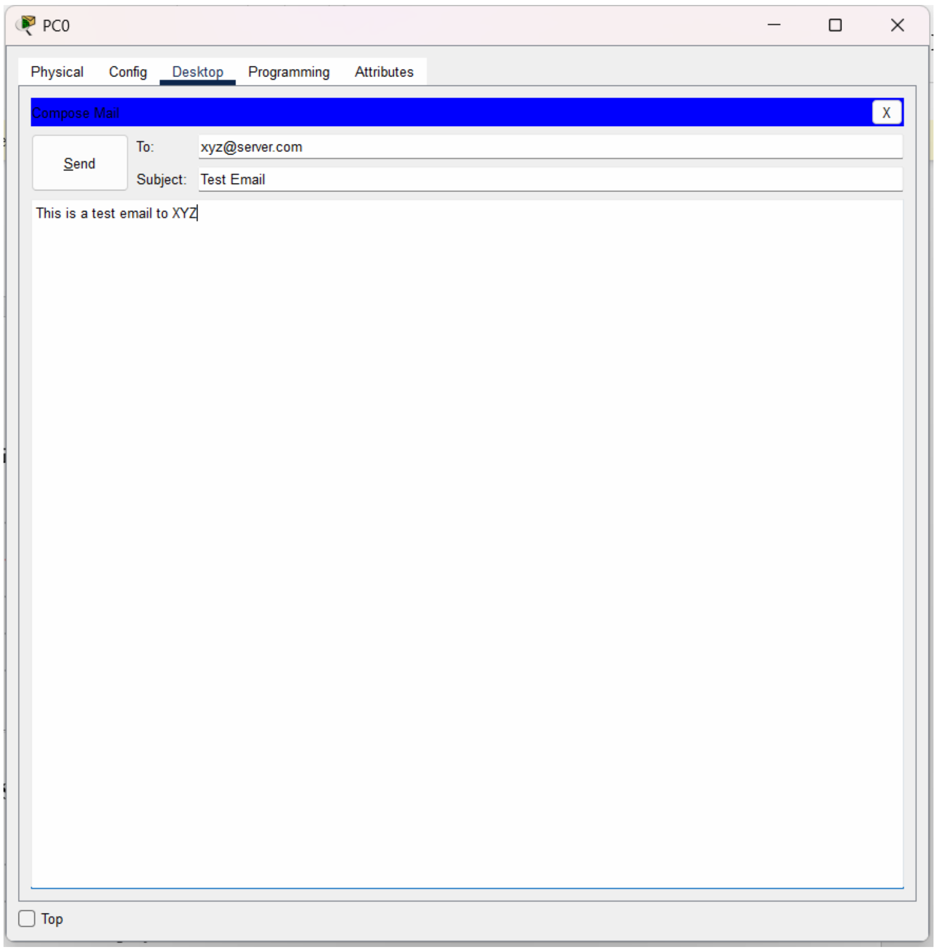
Server Information
Incoming Mail Server: 192.168.1.5
Outgoing Mail Server: 192.168.1.5

Logon Information
User Name: ABC
Password: ••••

## Step 2: Send an Email from ABC to XYZ

Still in **PC0 > Email** tab:

1. Click **Compose**
2. To: `xyz@server.com`
3. Subject: `Test Email`
4. Body: `This is a test email to XYZ`
5. Click **Send**



To: xyz@server.com
Subject: Test Email

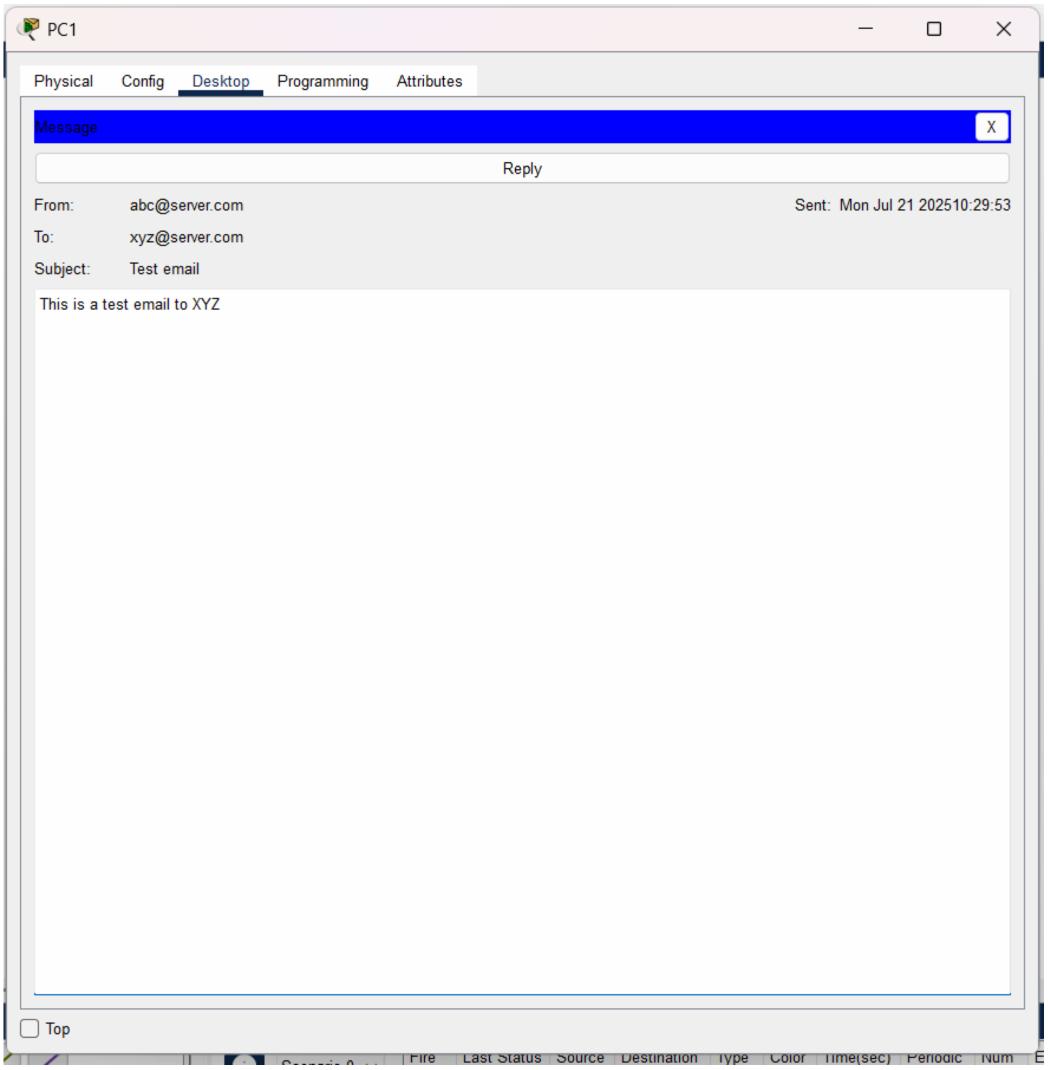
This is a test email to XYZ

## Step 3: Receive Email on PC1

1. Go to **PC1 > Desktop > Email**
2. Configure with:
   - Display Name: XYZ
   - Email: `xyz@server.com`

- **Incoming Mail Server:** `192.168.1.5`
    - **Outgoing Mail Server:** `192.168.1.5`
    - **Username:** `xyz`
    - **Password:** `6789`
3. Click **“Receive”** to fetch the email

The message from ABC will appear in the inbox.



From: abc@server.com
Sent: Mon Jul 21 202510:29:53
To: xyz@server.com
Subject: Test email

This is a test email to XYZ

## PART B: Use Telnet to Simulate SMTP & POP3

## Step 4: Use Telnet for Sending Email (SMTP)

1. On PC0 → Open **Command Prompt**
2. Connect to the SMTP server:

```
telnet 192.168.1.5 25
```

Now type SMTP commands:

```
HELO server.com
MAIL FROM:<b=abc@server.com>
RCPT TO:<xyz@server.com>
DATA
Subject: Hello XYZ
This is a test message using Telnet.
.
QUIT
```

Email sent to XYZ manually using SMTP commands.

### Step 5: Use Telnet for Receiving Email (POP3)

1. On PC1 → Open **Command Prompt**

```
telnet 192.168.1.5 110
```

Now type SMTP commands:

```
Sql

USER XYZ
PASS 6789
LIST
RETR 1
QUIT
```

Email from ABC retrieved manually using POP3 commands.

## Conclusion:

In this experiment, we successfully demonstrated how **emails are sent and received over a network** using two essential application layer protocols: **SMTP (Simple Mail Transfer Protocol)** and **POP3 (Post Office Protocol 3)**. We configured an email server, created user accounts, and simulated the complete email exchange process using both a **GUI-based mail client** and **Telnet commands**.

This experiment enhanced our understanding of:

- The roles of **SMTP** (for sending mail) and **POP3** (for receiving mail)

- The working of email services over **TCP/IP**
- How to manually connect and interact with servers using **Telnet**
- Verifying connectivity and mail delivery through **real-time simulation**

By completing this experiment, we gained valuable insight into **email protocol communication** and the practical aspects of configuring and troubleshooting **mail services** in both manual and client-based scenarios.

# Assignment No: - 5

**Aim:** Write a client-server program in Java or Python using TCP sockets.

## Software/Tools Required:

IDE or Text Editor (e.g., VS Code, PyCharm, Eclipse), Terminal or Command Prompt.

## Theory:

Python's socket module is a powerful tool for creating network applications. The socket is the endpoint of a bidirectional communications channel between the server and the client. Sockets may communicate within a process, between processes on the same machine, or between processes on different machines.

Steps to communicate both server and client model:

1. Python socket server program executes at first and waits for any request
2. Python socket client program will initiate the conversation at first.
3. Then server program will response accordingly to client requests.
4. Client program will terminate if user enters "bye" message. Server program will also terminate when client program terminates, this is optional and we can keep server program running indefinitely or terminate with some specific command in client request.

To use python socket connection, we need to import **socket** module. Then, sequentially we need to perform some tasks to establish connection between server and client. We can obtain host address by using socket.gethostname() function. The user port address should above 1024 because port number lesser than 1024 are reserved for standard internet protocol.

**Python socket server code:**

```
import socket

def server_program():
    # get the hostname
    host = socket.gethostname()
    port = 5000  # initiate port no above 1024

    server_socket = socket.socket()  # get instance
    # look closely. The bind() function takes tuple as argument
    server_socket.bind((host, port))  # bind host address and port together

    # configure how many client the server can listen simultaneously
    server_socket.listen(2)
    conn, address = server_socket.accept()  # accept new connection
    print("Connection from: " + str(address))
    while True:
        # receive data stream. it won't accept data packet greater than 1024 bytes
        data = conn.recv(1024).decode()
        if not data:
            # if data is not received break
            break
        print("from connected user: " + str(data))
        data = input(' -> ')
        conn.send(data.encode())  # send data to the client

    conn.close()  # close the connection


if __name__ == '__main__':
    server_program()
```

**Python Socket Client:**

```
import socket


def client_program():
    host = socket.gethostname()  # as both code is running on same pc
    port = 5000  # socket server port number

    client_socket = socket.socket()  # instantiate
    client_socket.connect((host, port))  # connect to the server

    message = input(" -> ")  # take input

    while message.lower().strip() != 'bye':
        client_socket.send(message.encode())  # send message
        data = client_socket.recv(1024).decode()  # receive response

        print('Received from server: ' + data)  # show in terminal

        message = input(" -> ")  # again take input

    client_socket.close()  # close the connection


if __name__ == '__main__':
    client_program()
```

**https://www.digitalocean.com/community/tutorials/python-socket-programming-server-client**

**Result:**

The client-server program was successfully executed using **TCP sockets** in Python (or Java). The client was able to **establish a connection** with the server, **send a message**, and receive a **response** from the server over a **reliable TCP connection**. This demonstrated basic socket communication and connection-oriented networking.

**Conclusion:**

This experiment demonstrated how to implement a **connection-oriented communication model** using **TCP sockets**. It provided a practical understanding of how **servers listen for client connections**, and how **clients initiate communication** over a specific port. The experiment reinforced core networking concepts such as socket creation, binding, listening, accepting, sending, and receiving messages using the **TCP protocol**. This forms the foundation of many real-world networked applications, including web servers, chat systems, and file transfer utilities.

**Oral Questions:**

1. What is a socket in networking?
2. What is the difference between TCP and UDP sockets?
3. What is the role of the server and client in socket communication?
4. What are the typical port numbers used for socket programming?
5. Why is TCP called a connection-oriented protocol?
6. Which function is used to create a socket in Python/Java?
7. What is the use of `bind()` in server-side programming?
8. What does the `listen()` function do?
9. What is the role of `accept()` on the server side?
10. How does the client connect to the server?

# Assignment No: - 6

**Aim:** Implement a TCP-IP client-server application. The server accepts a string as a request and sends the capitalized version back.

## Software/Tools Required:

Python, Terminal/Command Prompt, Text editor or IDE (VS Code, PyCharm, etc.)

.

## Theory:

In computer networks, communication between systems (hosts) is carried out using protocols—rules that govern data transmission. The TCP/IP model, also known as the Internet Protocol Suite, is the foundation of modern networking.

The TCP/IP model consists of four layers:

1. Application Layer – Supports network applications (e.g., HTTP, FTP, SMTP).
2. Transport Layer – Provides reliable data transfer using TCP or faster, connectionless transfer using UDP.
3. Internet Layer – Handles logical addressing (IP addresses) and routing.
4. Network Access Layer – Deals with physical transmission of data over network media.

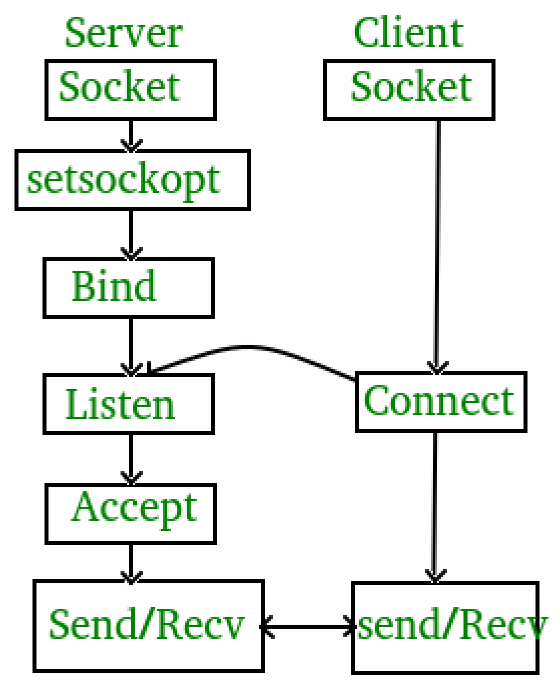
**Client-Server Architecture** involves two components:

- **Client:** Initiates communication by sending a request.
- **Server:** Waits for incoming requests and sends back responses.

**TCP (Transmission Control Protocol)** is connection-oriented and ensures reliable delivery of data.

**Socket Programming:** Involves using APIs to send/receive data over a network.

The entire process can be broken down into the following steps:



**TCP Server -**

1. Using create(), Create TCP socket.
1. Using bind(), Bind the socket to server address.
1. Using listen(), put the server socket in a passive mode, where it waits for the client to approach the server to make a connection
1. Using accept(), At this point, connection is established between client and server, and they are ready to transfer data.
1. Go back to Step 3.

**TCP Client -**

1. Create TCP socket.
1. Connect newly created client socket to server.

**Python socket server code:**

```
import socket

# Create a TCP/IP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind the socket to localhost and a port
server_socket.bind(('localhost', 12345))
```

```
# Listen for incoming connections
server_socket.listen(1)
print("Server is listening on port 12345...")

# Accept connection
conn, addr = server_socket.accept()
print(f"Connected by {addr}")

# Receive data
data = conn.recv(1024).decode()

# Process and send back response
capitalized_data = data.upper()
conn.send(capitalized_data.encode())

# Close connection
conn.close()
```

**Python Socket Client:**

```
import socket

# Create a socket

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Connect to server

client_socket.connect(('localhost', 12345))

# Send input

message = input("Enter a string: ")

client_socket.send(message.encode())

# Receive response

response = client_socket.recv(1024).decode()

print("Capitalized string from server:", response)

# Close socket

client_socket.close()
```

**Procedure:**

1. Open a terminal and run server.py.
2. Open another terminal and run client.py.
3. Enter a string in the client terminal.

4. Observe the capitalized string returned by the server.

**Result:**
The TCP/IP client-server application was successfully implemented. The server received a string input from the client, processed it by converting it into uppercase letters, and sent the transformed string back to the client. This demonstrated reliable, bidirectional communication using TCP sockets, validating the concepts of socket programming, client-server architecture, and TCP/IP protocol operations.

**Conclusion:**
In this lab experiment, we successfully developed a TCP/IP-based client-server application using socket programming. The client sent a string to the server, which processed the input by converting it to uppercase and returned the modified string.

**Oral Questions:**
1. What is the main objective of this experiment?
2. What did you conclude from this experiment?
3. Why was TCP used instead of UDP in this application?
4. What happens if the server is not running when the client tries to connect?
5. How does the server know when to stop receiving data?
6. What is the role of bind(), listen(), and accept() functions in the server program?
7. What does socket.AF_INET and socket.SOCK_STREAM mean in Python socket programming?
8. Can this server handle multiple clients? Why or why not?
9. What is the importance of encoding and decoding messages in socket communication?
10. What changes would you make to convert this from a single-use to a multi-client server?

**Experiment No: 07**

## 1) Experiment Title

**Setup and Use an FTP Server (PT Server) with a Client (PC) in a Single-LAN Topology**

## 2) Aim

Design a simple LAN in Cisco Packet Tracer, configure an FTP server and an FTP client, and verify file upload/download. Observe the FTP control and data connections using Simulation Mode.

## 3) Outcomes (CO Mapping)

After completing this experiment, the student will be able to:

1. Design a small switched LAN and assign IPv4 addressing.
2. Configure Packet Tracer **Server** as an **FTP** server (enable service, create users).
3. Use a **PC FTP Client** to connect, authenticate, upload, and download files.
4. Validate connectivity with **ICMP (ping)** and analyze FTP using **Simulation Mode** (TCP 21 control, TCP 20 data in active mode).
5. Troubleshoot common FTP issues (IP addressing, service status, credentials).

## 4) Prerequisites / Theory Snapshot

- **FTP (File Transfer Protocol)** runs on TCP; default **control port 21**, **data port 20** (active mode).
- **Authentication**: Username/password (avoid Anonymous for controlled access).
- **Packet Tracer Server** provides a simplified FTP service; it is sufficient for functional demonstration (not full FTPS/SFTP security).
- **Active vs Passive**: Packet Tracer models basic/active behavior; focus on port 21 (control) and server-initiated data connection (port 20) for this lab.

## 5) Lab Topology & Addressing

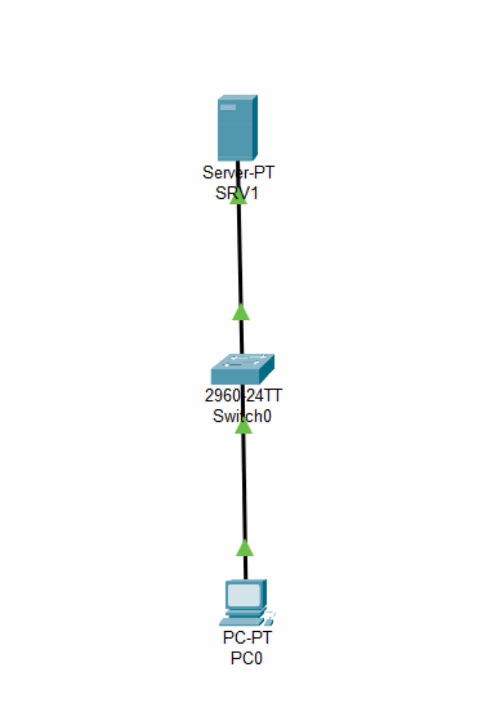
**Devices**

- 1 × Switch (e.g., **2960**).
- 1 × PC (Client) → **PC0**.

- 1 × **Server** (PT Server) → **SRV1**.

**Connections**

- **Copper Straight-Through** cables from PC0 → Switch, and Server → Switch.



**IPv4 Plan**

| Device | Interface | IP Address | Subnet Mask | Default GW |
|---|---|---|---|---|
| PC0 (Client) | FastEthernet0 | 192.168.1.1 | 255.255.255.0 | (blank) |
| SRV1 (Server) | FastEthernet0 | 192.168.1.2 | 255.255.255.0 | (blank) |

*No router is required for a same-subnet LAN demo; therefore, Gateway can remain blank.*

## 6) Step-by-Step Procedure

### A) Build the Physical Topology

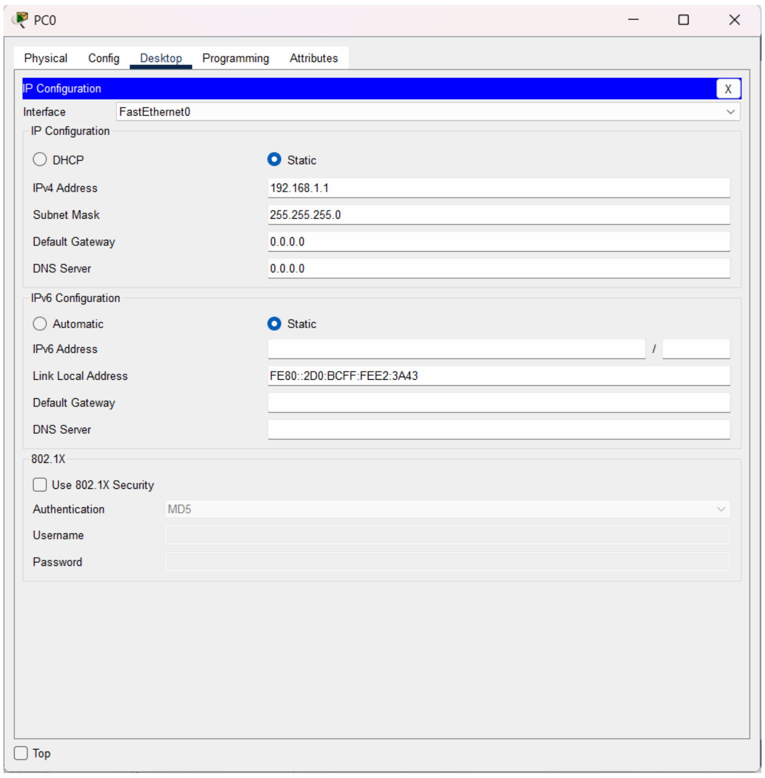
1. **Open Cisco Packet Tracer** → New workspace.
2. From **End Devices**, drag **PC** (PC0) and **Server** (SRV1) to the workspace.
3. From **Switches**, drag **2960** to the workspace.
4. Click **Connections (lightning bolt)** → choose **Copper Straight-Through**.
   - Click **PC0** → select **FastEthernet0** → then click **Switch** → choose any **FastEthernet** port (e.g., Fa0/1).

    - Click **SRV1** → **FastEthernet0** → **Switch** → (e.g., Fa0/2).
5. Wait for link LEDs to turn **green** (up/up). If amber, give it a moment to negotiate.

## B) Configure IP Addressing

### On PC0:

1. Click **PC0** → **Desktop** → **IP Configuration**.
2. Set **IP Address** = `192.168.1.1`, **Subnet Mask** = `255.255.255.0`.
3. Leave **Default Gateway** blank. Close.



Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

| | |
|---|---|
| IPv4 Address | 192.168.1.1 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 0.0.0.0 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

| | |
|---|---|
| IPv6 Address | / |
| Link Local Address | FE80::2D0:BCFF:FEE2:3A43 |
| Default Gateway | |
| DNS Server | |

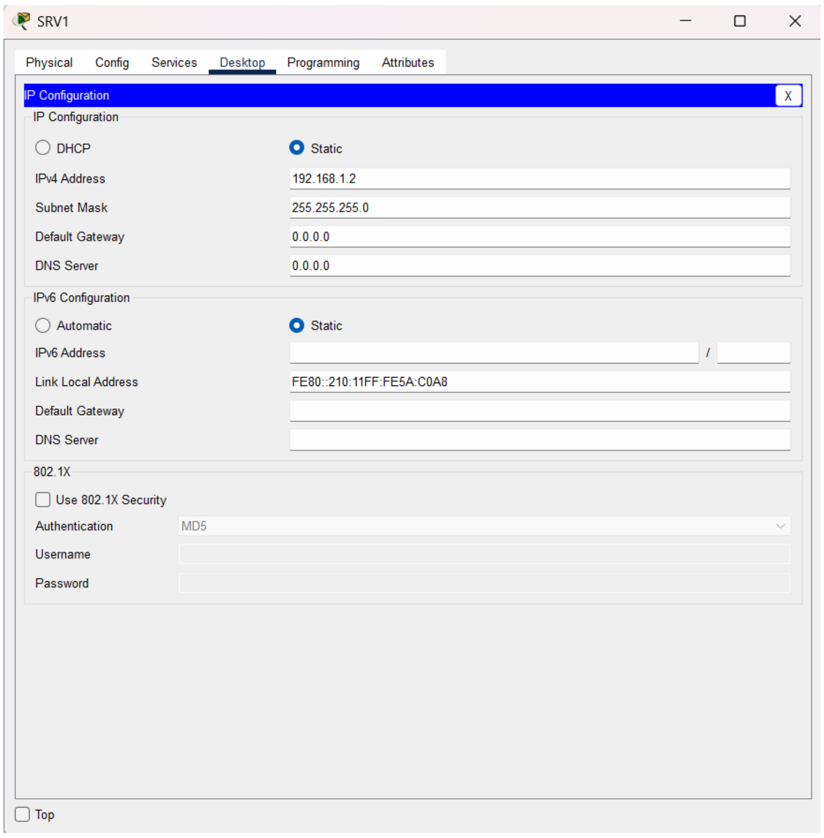
802.1X

Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

### On SRV1 (Server):

1. Click **Server** → **Desktop** → **IP Configuration**.
2. Set **IP Address** = `192.168.1.2`, **Subnet Mask** = `255.255.255.0`.
3. Leave **Default Gateway** blank. Close.


SRV1

Physical | Config | Services | Desktop | Programming | Attributes

IP Configuration

IP Configuration

DHCP [✓ Static]

| | |
|---|---|
| IPv4 Address | 192.168.1.2 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 0.0.0.0 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

| | |
|---|---|
| IPv6 Address | / |
| Link Local Address | FE80::210:11FF:FE5A:C0A8 |
| Default Gateway | |
| DNS Server | |

802.1X

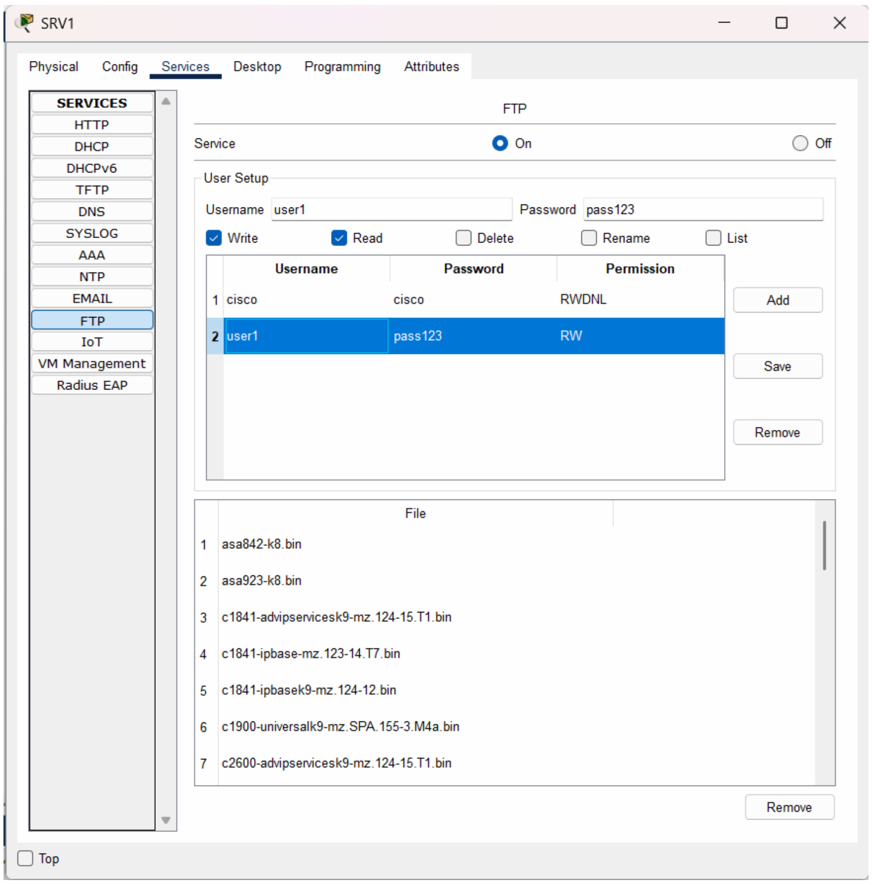
☐ Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

☐ Top

## C) Enable & Configure FTP Service on the Server

1. Click **Server** → go to the **Services** tab.
2. In the left pane, click **FTP**.
3. Toggle **FTP: On** (ensure the service status shows **On**).
4. **Users section**: Add a managed user for authentication:
    - Click **Add** → enter **Username**: `user1` → **Password**: `pass123`.
    - Optionally uncheck **Anonymous** if present, to prevent anonymous access.
    - Ensure **Read** and **Write** permissions are allowed for `user1`.


SRV1

Physical | Config | Services | Desktop | Programming | Attributes

SERVICES: HTTP, DHCP, DHCPv6, TFTP, DNS, SYSLOG, AAA, NTP, EMAIL, FTP, IoT, VM Management, Radius EAP

FTP

Service [✓ On] Off

User Setup

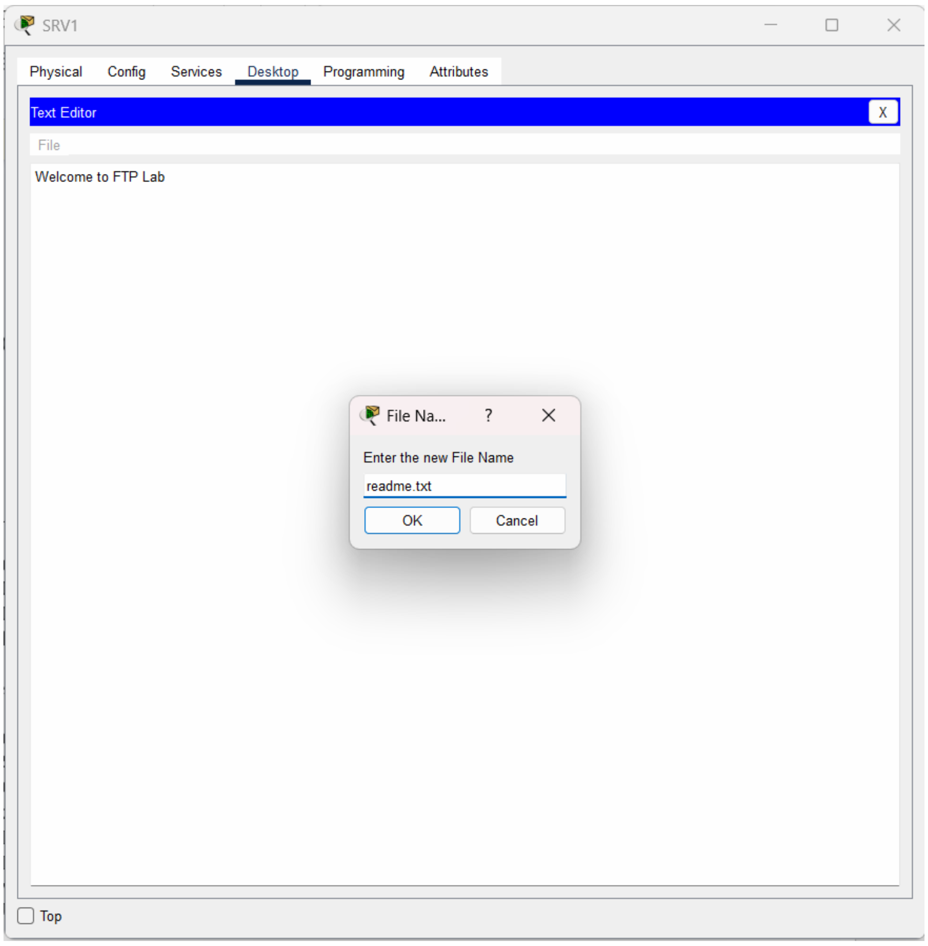
Username user1 Password pass123

[✓ Write] [✓ Read] Delete Rename List

| | Username | Password | Permission |
|---|---|---|---|
| 1 | cisco | cisco | RWDNL |
| 2 | user1 | pass123 | RW |

| | File |
|---|---|
| 1 | asa842-k8.bin |
| 2 | asa923-k8.bin |
| 3 | c1841-advipservicesk9-mz.124-15.T1.bin |
| 4 | c1841-ipbase-mz.123-14.T7.bin |
| 5 | c1841-ipbasek9-mz.124-12.bin |
| 6 | c1900-universalk9-mz.SPA.155-3.M4a.bin |
| 7 | c2600-advipservicesk9-mz.124-15.T1.bin |

5. (Optional) Prepare a server-side file for download:
   - Go to **Server** → **Desktop** → **Text Editor** → create sample text (e.g., "Welcome to FTP lab") → **Save** as `readme.txt`.
   - If needed, use **Desktop** → **File Manager** to ensure the file is in the server's default FTP directory (Packet Tracer maps it internally; saving under the Server's file system is sufficient for this lab).



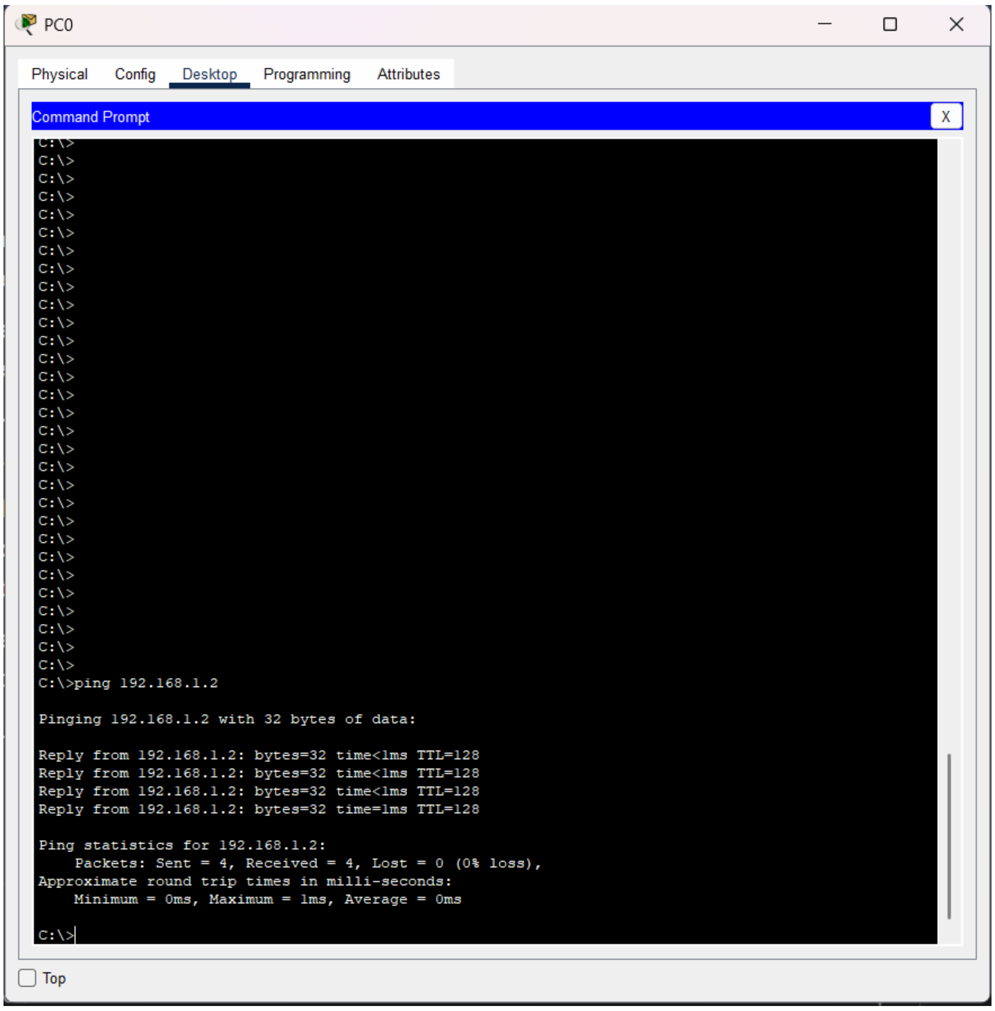
Welcome to FTP Lab

Enter the new File Name

readme.txt

## D) Validate Layer-3 Connectivity (Ping)

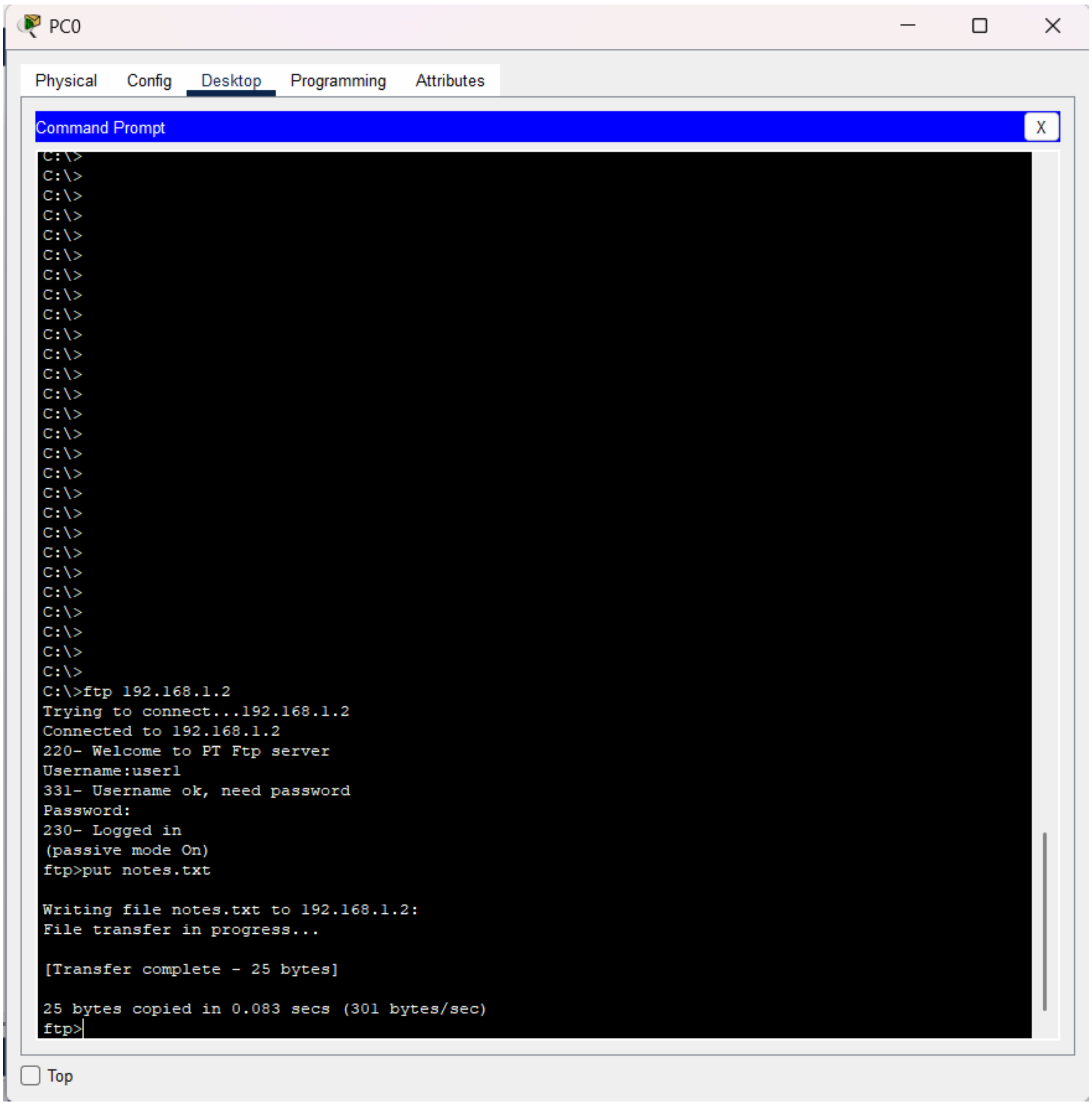
1. On **PC0** → **Desktop** → **Command Prompt**.
2. Run: `ping 192.168.1.2`
3. Expect **Reply from 192.168.1.2** with 0% loss. If it fails, re-check IPs, cabling, and link LEDs.



```
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
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 = 1ms, Average = 0ms

C:\>
```

## E) Use the FTP Client (on PC0)

1. On **PC0** → **Desktop** → **FTP Client**.
2. **Server**: `192.168.1.2` | **Username**: `user1` | **Password**: `pass123` | **Port**: `21.`
3. Click **Connect** → observe the **status/log** window: look for `220` (service ready) and `230` (user logged in).
4. **Download test**: If `readme.txt` exists on the server side, select it and click **Download**. Confirm it appears in PC0's local directory (shown on the left pane), or verify using **Desktop** → **File Manager** on PC0.
5. **Upload test**:
   - On **PC0** → **Desktop** → **Text Editor**, create `notes.txt` (e.g., "FTP upload from PC0") → **Save**.
   - In **FTP Client**, navigate to local file list, select `notes.txt` → click **Upload**.
   - Verify on the server: **Server** → **Desktop** → **File Manager** (or re-open FTP Client on PC0 and check remote listing).



```
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>
C:\>ftp 192.168.1.2
Trying to connect...192.168.1.2
Connected to 192.168.1.2
220- Welcome to PT Ftp server
Username:user1
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>put notes.txt

Writing file notes.txt to 192.168.1.2:
File transfer in progress...

[Transfer complete - 25 bytes]

25 bytes copied in 0.083 secs (301 bytes/sec)
ftp>
```

### F) Observe FTP in Simulation Mode

1. Click the **Simulation** tab (bottom right).
2. Click **Edit Filters** → enable **TCP**, **ICMP**, and **FTP** (if listed).
3. Initiate an FTP action (e.g., reconnect, list, upload).
4. Use **Auto Capture/Play** or **Capture/Forward** to step through packets.
5. Observe:
   - **TCP 3-way handshake** between PC0 and SRV1.
   - **Control channel** to **port 21** (USER, PASS, LIST, STOR/RETR).
   - **Data transfer** (active mode) indicated by a separate TCP session (server data port 20).

## 7) Verification & Expected Results

- `ping 192.168.1.2` from PC0 succeeds.
- FTP Client log shows: `220 Service ready`, `331 User name okay`, `230 User logged in`, `226 Transfer complete` for uploads/downloads.
- Uploaded file `notes.txt` appears on server; downloaded `readme.txt` appears on PC0.
- In Simulation Mode, you can identify **TCP 21** control packets and data transfer packets with different sequence numbers.

## 8) Troubleshooting Guide

| Symptom | Likely Cause | Fix |
|---|---|---|
| Ping fails | Wrong IP, mask, or cable/port | Re-check IPs, ensure /24 mask, correct Fa0/port, link LEDs are green |
| FTP connect timeout | FTP service off | Server → Services → FTP → set **On** |
| 530 Login incorrect | Wrong username/password | Recreate user `user1/pass123` on Server → FTP Users |
| Upload not visible | Saved in wrong local path | Confirm file saved on PC0 via Desktop → File Manager; refresh FTP client listing |
| Cannot download | File absent on server | Create `readme.txt` on server (Desktop → Text Editor), verify via File Manager |
| Control works but no data | Simulation step not continued | In Simulation, keep stepping until data session completes |

## Experiment No: 08

## Experiment Title:

## Design and implement subnetting schemes for a given IP address.

### Objective

To design and implement a subnetting scheme for a given IP address, assign IP addresses to devices, and verify network connectivity using **Cisco Packet Tracer**.

### Course Outcome Mapped

- **CO1**: Describe fundamental concepts of Computer Networks & role of network models.
- **CO3**: Discuss skills in configuring network devices and implementing IP addressing.

### Tools Required

- Cisco Packet Tracer (v7.x or later)
- Networking devices: PCs, Switches, Routers
- Ethernet cables (Copper Straight-Through)

### Theory

**Subnetting** is the process of dividing a larger network into smaller sub networks to optimize IP address usage and improve network performance and security.

Key terms:

- **IP Address**: A unique identifier for a device in a network.
- **Subnet Mask**: Defines the network and host portions of an IP address.
- **CIDR Notation**: A shorthand representation of the subnet mask (e.g., `/26`).

### What is Subnetting

Subnetting is the process of **dividing a large IP network** into **smaller, more manageable subnetworks** (subnets).
It's done by **borrowing bits** from the host portion of an IP address to create additional **network addresses**.

**Why we do it:**

- Better IP address **utilization**
- Improved **network performance** (less broadcast traffic)
- Enhanced **security** by isolating network segments
- Logical grouping of devices (departments, floors, buildings)

## IP Address Basics

An **IPv4 address** is **32 bits** long, written in **dotted decimal** form:

Example:

```
CopyEdit
192.168.10.25 → 11000000.10101000.00001010.00011001
```

The IP address is split into:

- **Network portion** → Identifies the network
- **Host portion** → Identifies a device in that network

The **Subnet Mask** determines where the split happens.

## Subnet Mask

A subnet mask is a **32-bit number** that **masks** (marks) the network part as `1`s and the host part as `0`s.

Example:

```
yaml
CopyEdit
IP:          192.168.10.25
Subnet Mask: 255.255.255.0
Binary:      11111111.11111111.11111111.00000000
```

Here:

- First **24 bits** (`255.255.255`) → Network
- Last **8 bits** (`0`) → Hosts

## CIDR Notation

CIDR (**Classless Inter-Domain Routing**) is a shorthand for subnet masks.

Example:

```
255.255.255.0 → /24
255.255.255.192 → /26
```

`/26` means **first 26 bits** are network bits, and remaining bits are for hosts.

## How Subnetting Works

We **borrow host bits** to create more networks.

**Example Problem**:
Network: `192.168.10.0/24` → Need **4 subnets**.

### Step 1: Find required subnet bits

Formula:

```
Number of subnets = 2^n
```

Where `n` = number of borrowed bits.

For 4 subnets:

```
2^n = 4 → n = 2 bits
```

### Step 2: New Subnet Mask

Original mask: `/24` → 255.255.255.0
Borrow 2 bits → `/26` → 255.255.255.192

### Step 3: Number of hosts per subnet

Formula:

```
Hosts per subnet = 2^h - 2
```

Where `h` = number of host bits.

Host bits: `32 - 26 = 6` → 2^6 – 2 = **62 hosts** per subnet.

### Step 4: Subnet Ranges

**Increment** = 256 – 192 = **64**

| Subnet No | Network Address | First Host | Last Host | Broadcast |
|---|---|---|---|---|
| 1 | 192.168.10.0 | 192.168.10.1 | 192.168.10.62 | 192.168.10.63 |
| 2 | 192.168.10.64 | 192.168.10.65 | 192.168.10.126 | 192.168.10.127 |
| 3 | 192.168.10.128 | 192.168.10.129 | 192.168.10.190 | 192.168.10.191 |
| 4 | 192.168.10.192 | 192.168.10.193 | 192.168.10.254 | 192.168.10.255 |

## Subnetting Formulas Summary

1. **Subnet mask** = Network bits + Borrowed bits
2. **Number of subnets** = 2^n (n = borrowed bits)
3. **Number of hosts per subnet** = 2^h – 2 (h = host bits)
4. **Subnet increment** = 256 – last octet of mask

## Example Scenario

Given Network: `192.168.10.0/24`
Requirement: 4 subnets, each with up to 50 hosts.

### Step 1: Subnet Calculation

1. Hosts required: 50
2. Formula:

   ```
   Hosts = 2^n - 2
   Where n = number of host bits
   ```

   For 50 hosts:

   ```
   2^6 - 2 = 62 hosts → 6 host bits
   ```

3. New Subnet Mask:

   ```
   32 - 6 = 26 → 255.255.255.192 (/26)
   ```

4. Subnets:
   - Subnet 1: `192.168.10.0/26` (Hosts: .1 - .62, Broadcast: .63)
   - Subnet 2: `192.168.10.64/26` (Hosts: .65 - .126, Broadcast: .127)
   - Subnet 3: `192.168.10.128/26` (Hosts: .129 - .190, Broadcast: .191)
   - Subnet 4: `192.168.10.192/26` (Hosts: .193 - .254, Broadcast: .255)

## Procedure in Packet Tracer

1. **Open Packet Tracer** and create a new workspace.
2. **Place Devices**:
   - 4 PCs
   - 1 Router
   - 1 Switch per subnet
3. **Connect Devices**:
   - Use Copper Straight-Through cables for PC–Switch and Switch–Router connections.
4. **Assign IP Addresses**:
   - PC1: `192.168.10.1/26`, Gateway: `192.168.10.62`
   - PC2: `192.168.10.65/26`, Gateway: `192.168.10.126`
   - PC3: `192.168.10.129/26`, Gateway: `192.168.10.190`
   - PC4: `192.168.10.193/26`, Gateway: `192.168.10.254`
5. **Configure Router Interfaces**:

```
Router> enable
Router# configure terminal
Router(config)# interface fastEthernet 0/0
Router(config-if)# ip address 192.168.10.62 255.255.255.192
Router(config-if)# no shutdown

Router(config)# interface fastEthernet 0/1
Router(config-if)# ip address 192.168.10.126 255.255.255.192
Router(config-if)# no shutdown
...
```

6. **Verify Connectivity**:
   - Use **Command Prompt** on each PC:

```
ping <IP of another subnet host>
```

## Observation Table

| Subnet | Network Address | First Host | Last Host | Broadcast Address |
|---|---|---|---|---|
| 1 | 192.168.10.0 | 192.168.10.1 | 192.168.10.62 | 192.168.10.63 |
| 2 | 192.168.10.64 | 192.168.10.65 | 192.168.10.126 | 192.168.10.127 |
| 3 | 192.168.10.128 | 192.168.10.129 | 192.168.10.190 | 192.168.10.191 |
| 4 | 192.168.10.192 | 192.168.10.193 | 192.168.10.254 | 192.168.10.255 |

## Expected Output

- All devices in the **same subnet** can communicate directly.
- Devices in **different subnets** can communicate through the router.
- **Ping tests** between subnets are successful.

**Practical: Configure Routing Tables and Enable Routing using Packet Tracer or GNS3**

**Objective**

To understand and implement routing by configuring routing tables on routers and enabling communication between different networks using Packet Tracer or GNS3.

**Background / Theory**

Routing is the process of selecting the best path for data packets to travel across interconnected networks. A routing table is a data table stored in a router or networked computer that lists the routes to particular network destinations. When a packet arrives at a router, the router checks its routing table to determine the next hop (outgoing interface or router) for the packet.

There are two types of routing:

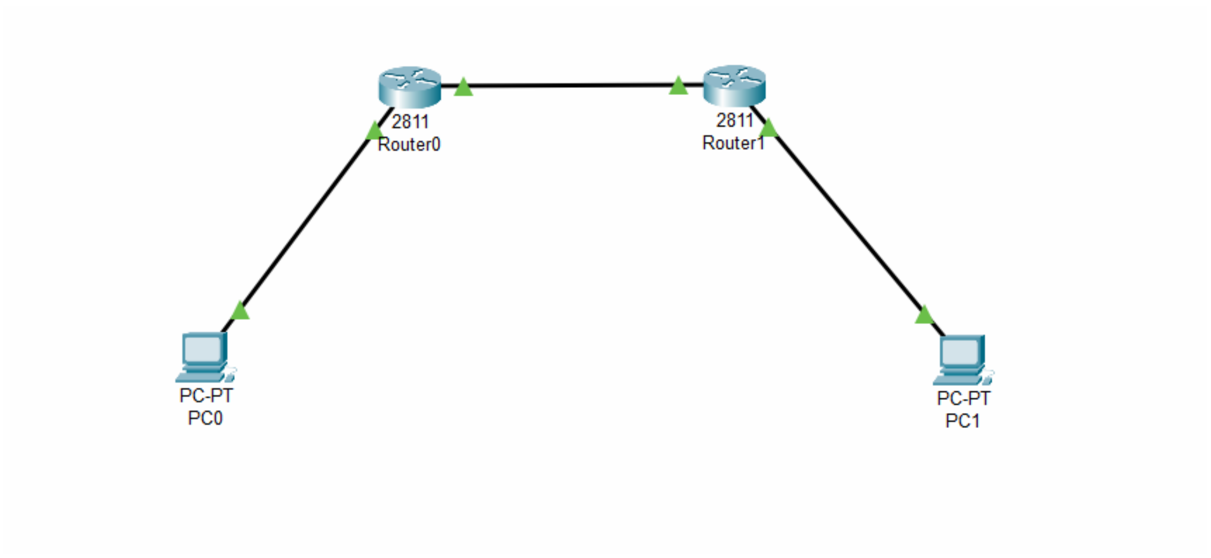
1. **Static Routing** – Routes are manually configured by the network administrator.
2. **Dynamic Routing** – Routes are learned and updated automatically using routing protocols like RIP, OSPF, or EIGRP.

In this practical, static routing is used to manually configure routes in the routing table so that routers can forward packets correctly between different networks.

**Experiment Steps**

**Step 1 — Build the topology in Packet Tracer**

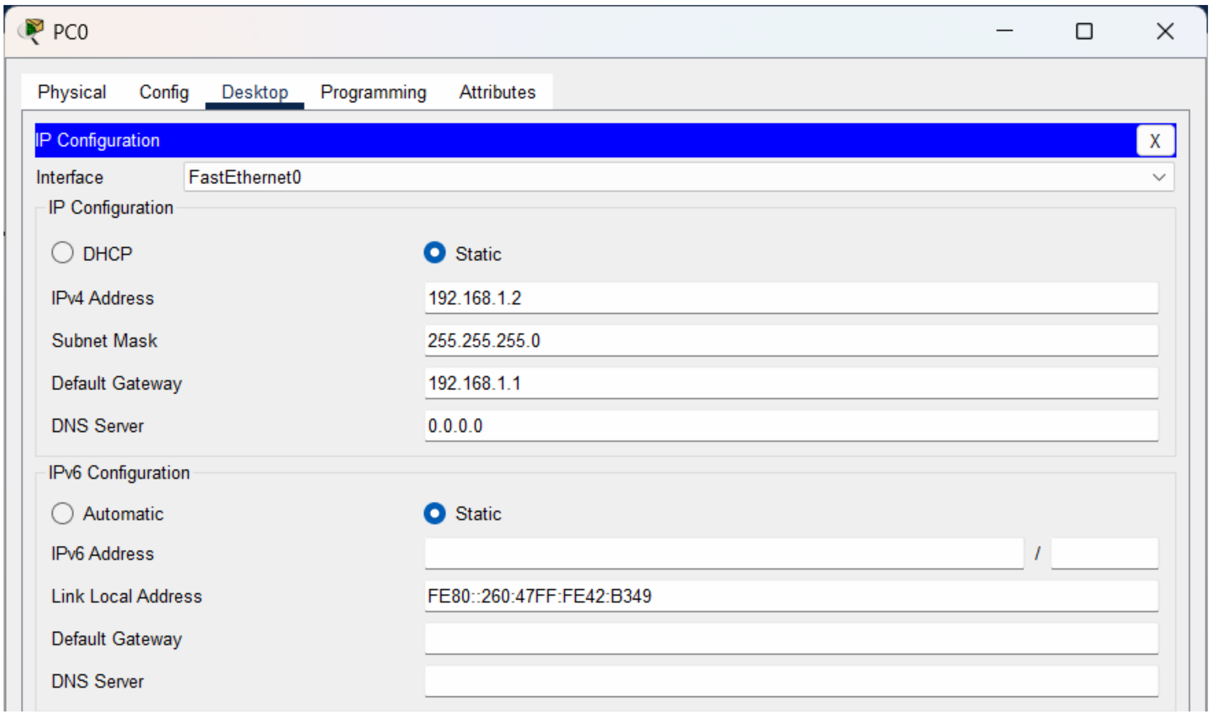
1. Open Cisco Packet Tracer.
2. From Routers, drag 2 routers (e.g., 2811) onto the workspace. Name them R1 and R2 (you can rename later).
3. From End Devices, drag 2 PCs onto the workspace: PC0 and PC1.
4. Connect devices:
   - Click the Connections (lightning) icon → choose Copper Straight-Through.
   - Connect PC0 → R1 FastEthernet0/0.
   - Connect R1 FastEthernet0/1 → R2 FastEthernet0/1.
   - Connect R2 FastEthernet0/0 → PC1.



**Step 2 — Assign IP addresses and enable router interfaces**

**Configure PCs:**

Click PC0 → Desktop → IP Configuration → enter IP: 192.168.1.2, Subnet: 255.255.255.0, Gateway: 192.168.1.1.



IP Configuration

| | |
|---|---|
| Interface | FastEthernet0 |

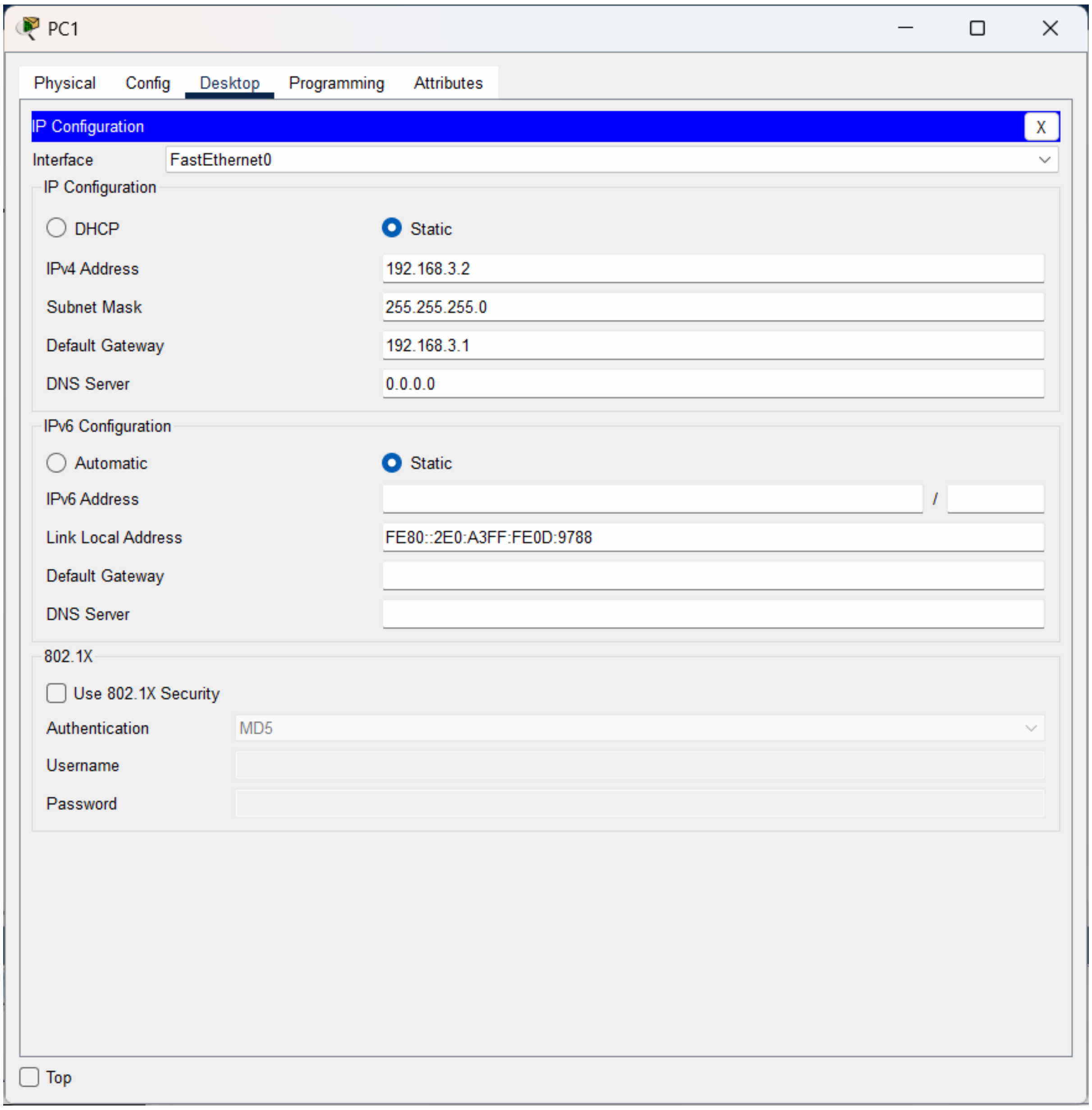
IP Configuration

| | |
|---|---|
| DHCP | [✓ Static] |
| IPv4 Address | 192.168.1.2 |
| Subnet Mask | 255.255.255.0 |
| Default Gateway | 192.168.1.1 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

| | |
|---|---|
| Automatic | [✓ Static] |
| IPv6 Address | / |
| Link Local Address | FE80::260:47FF:FE42:B349 |
| Default Gateway | |
| DNS Server | |

Do the same for PC1 with IP: 192.168.3.2, Subnet: 255.255.255.0, Gateway: 192.168.3.1.



Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

IPv4 Address: 192.168.3.2

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.3.1

DNS Server: 0.0.0.0

IPv6 Configuration

Automatic [✓ Static]

IPv6 Address: /

Link Local Address: FE80::2E0:A3FF:FE0D:9788

Default Gateway:

DNS Server:

802.1X

Use 802.1X Security

Authentication: MD5

Username:

Password:

**Configure Routers (example for R1):**

Click R1 → CLI. Type:

```
Router> enable
Router# configure terminal
Router(config)# interface fastEthernet 0/0
Router(config-if)# ip address 192.168.1.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# interface fastEthernet 0/1
Router(config-if)# ip address 192.168.2.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
```

**Do the same on R2:**

```
Router> enable
Router# configure terminal
Router(config)# interface fastEthernet 0/0
Router(config-if)# ip address 192.168.3.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
Router(config)# interface fastEthernet 0/1
Router(config-if)# ip address 192.168.2.2 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
```

**Step 3 — Configure Routing Tables**

We’ll use **Static Routing** for simplicity (later you can try RIP, OSPF, etc.).

**On Router R1:**

```
Router> enable
Router# configure terminal
Router(config)# ip route 192.168.3.0 255.255.255.0 192.168.2.2
```

**On Router R2:**

Router> enable

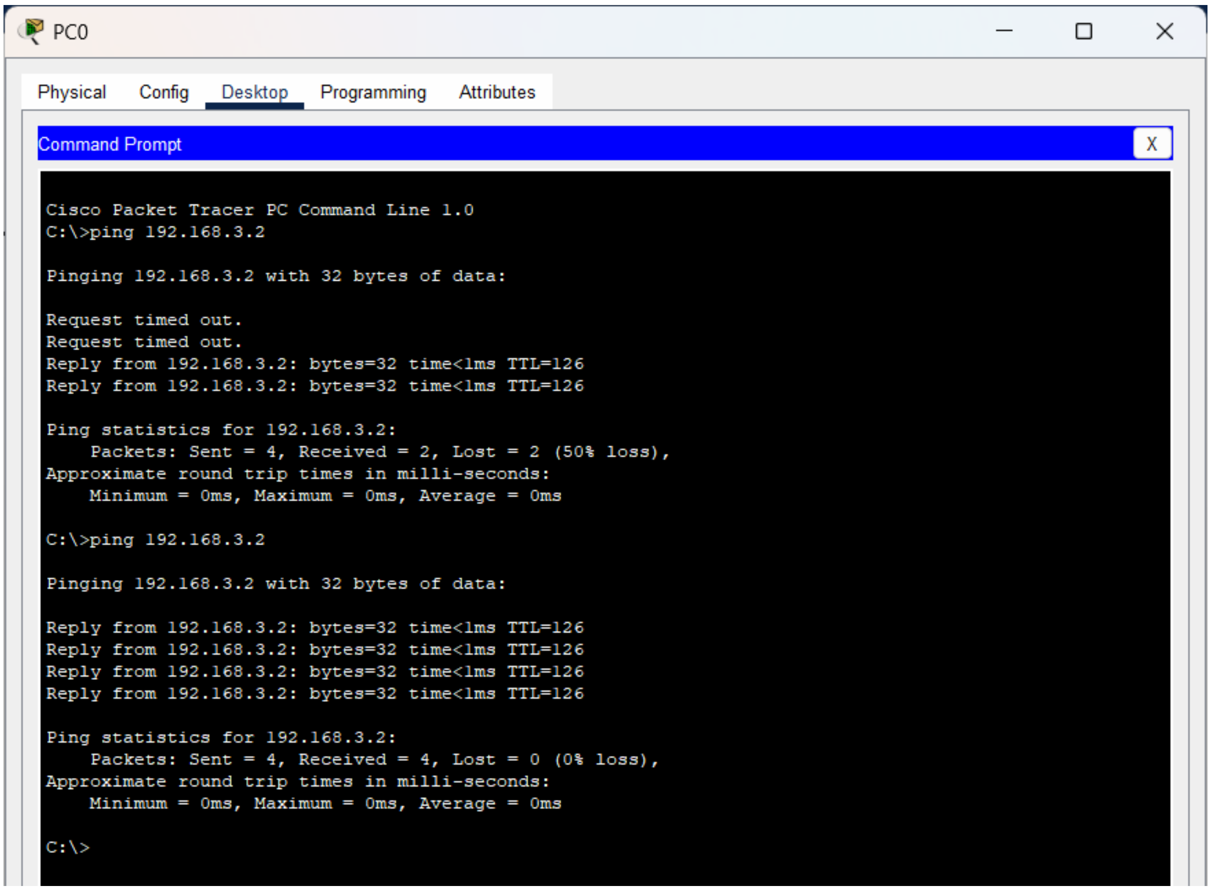
Router# configure terminal

Router(config)# ip route 192.168.1.0 255.255.255.0 192.168.2.1

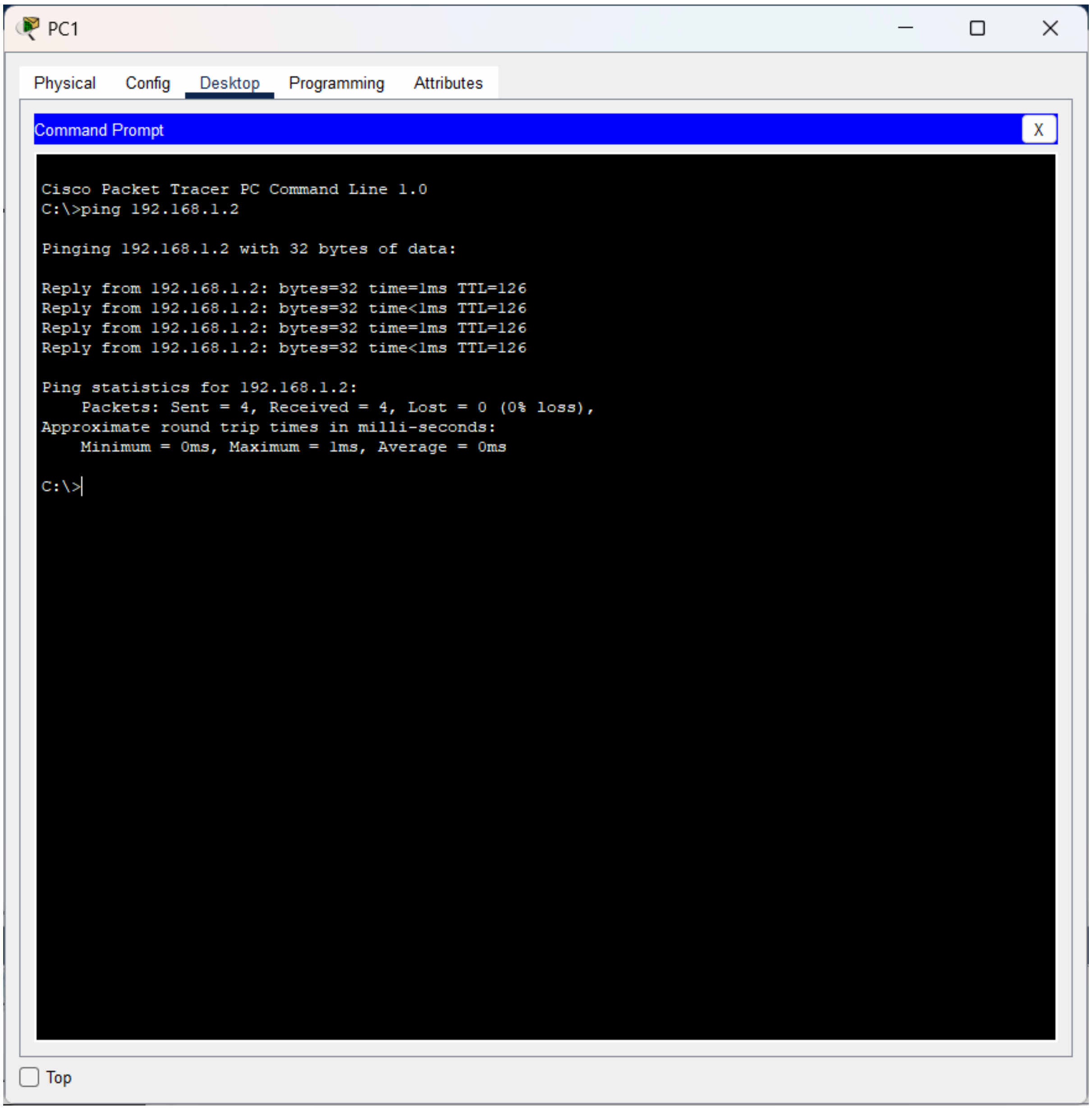
**Step 4 — Test Connectivity**

Now that routing is enabled, let's test if **PC0 can reach PC1**.

1. Go to **PC0 → Desktop → Command Prompt**.
2. Type: ping 192.168.3.2
3. You should see **Reply from 192.168.3.2** messages. If successful → routing is working!
4. For confirmation, test from **PC1 to PC0**:
5. Go to **PC1 → Desktop → Command Prompt**.
   - Type: ping 192.168.1.2



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.3.2

Pinging 192.168.3.2 with 32 bytes of data:

Request timed out.
Request timed out.
Reply from 192.168.3.2: bytes=32 time<1ms TTL=126
Reply from 192.168.3.2: bytes=32 time<1ms TTL=126

Ping statistics for 192.168.3.2:
    Packets: Sent = 4, Received = 2, Lost = 2 (50% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 192.168.3.2

Pinging 192.168.3.2 with 32 bytes of data:

Reply from 192.168.3.2: bytes=32 time<1ms TTL=126
Reply from 192.168.3.2: bytes=32 time<1ms TTL=126
Reply from 192.168.3.2: bytes=32 time<1ms TTL=126
Reply from 192.168.3.2: bytes=32 time<1ms TTL=126

Ping statistics for 192.168.3.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```



```
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=126
Reply from 192.168.1.2: bytes=32 time<1ms TTL=126
Reply from 192.168.1.2: bytes=32 time=1ms TTL=126
Reply from 192.168.1.2: bytes=32 time<1ms TTL=126

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 = 1ms, Average = 0ms

C:\>
```

If both pings succeed, you’ve successfully configured routing in Packet Tracer.

**Title: Configure IPv6 Addressing on Virtual/Physical Machines**

**Objective**

To configure IPv6 addressing on network devices (routers, PCs, or servers) and verify IPv6 connectivity on virtual or physical machines.

**Background / Theory**

IPv6 (Internet Protocol version 6) is the successor to IPv4, developed to overcome the problem of IPv4 address exhaustion. It uses 128-bit addresses, allowing for a vastly larger address space. IPv6 also introduces simplified header format, hierarchical addressing, better security features (IPsec built-in), and auto-configuration capabilities.

An IPv6 address is represented in hexadecimal and divided into eight groups of four hex digits, separated by colons (e.g., 2001:0db8:85a3::8a2e:0370:7334).
Types of IPv6 addresses include:

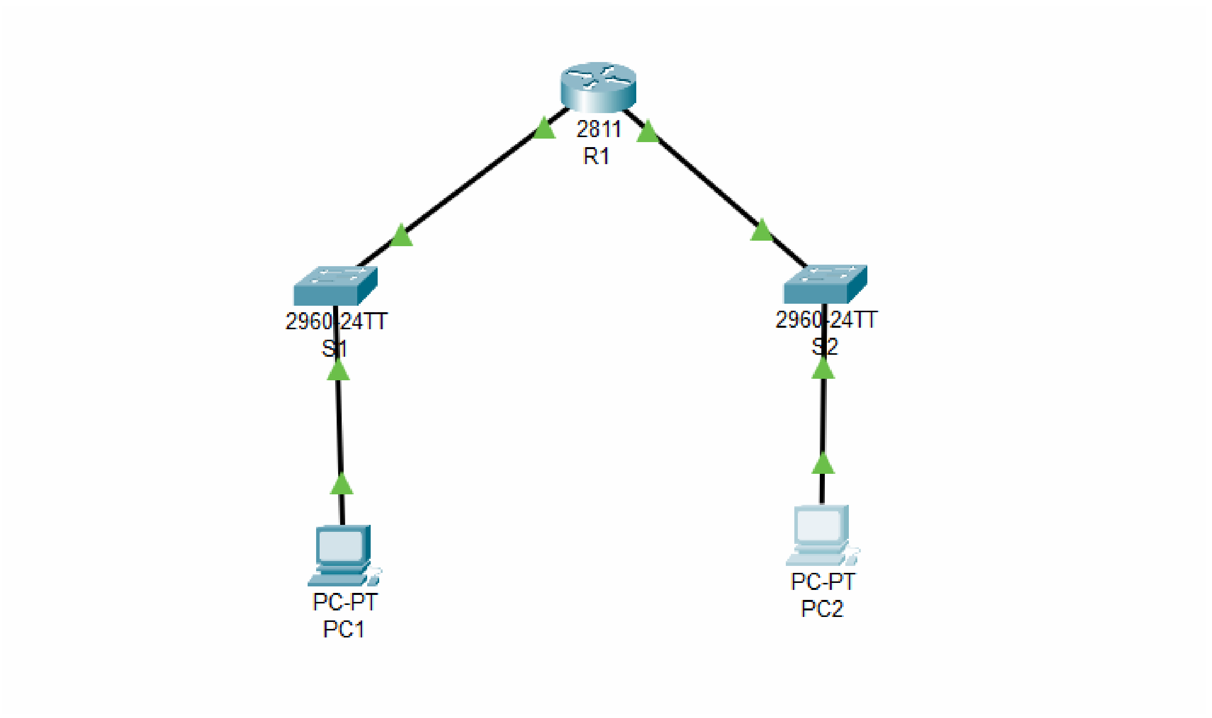
- **Unicast**: Identifies a single interface.
- **Multicast**: Identifies multiple interfaces.
- **Anycast**: Assigned to multiple interfaces; the nearest one is chosen.

In this practical, we configure IPv6 addresses on PCs and routers, then verify end-to-end communication.

**Experiment Steps:**

**Step 1 — Build the physical topology in Packet Tracer (prepare the lab)**

1. Open Cisco Packet Tracer.
2. From the device panel add:
   - **1 Router** (e.g., 2811 or 1941) — name it **R1**.
   - **2 Switches** (e.g., 2960) — name them **S1** and **S2**.
   - **2 PCs** (PC-PT) — name them **PC1** and **PC2**.
3. Arrange them on the canvas like this:



4. Connect devices using the **Connections** tool:
   - Choose **Copper Straight-Through**.
   - Connect **PC1 → S1**: PC1 FastEthernet0 → S1 Fa0/2
   - Connect **PC2 → S2**: PC2 FastEthernet0 → S2 Fa0/2
   - Connect **R1 → S1**: R1 Fa0/0 → S1 Fa0/1
   - Connect **R1 → S2**: R1 Fa0/1 → S2 Fa0/1

**Step 2 — Configure IPv6 addresses on router interfaces**

1. Click on **R1** → go to **CLI tab**, Type:

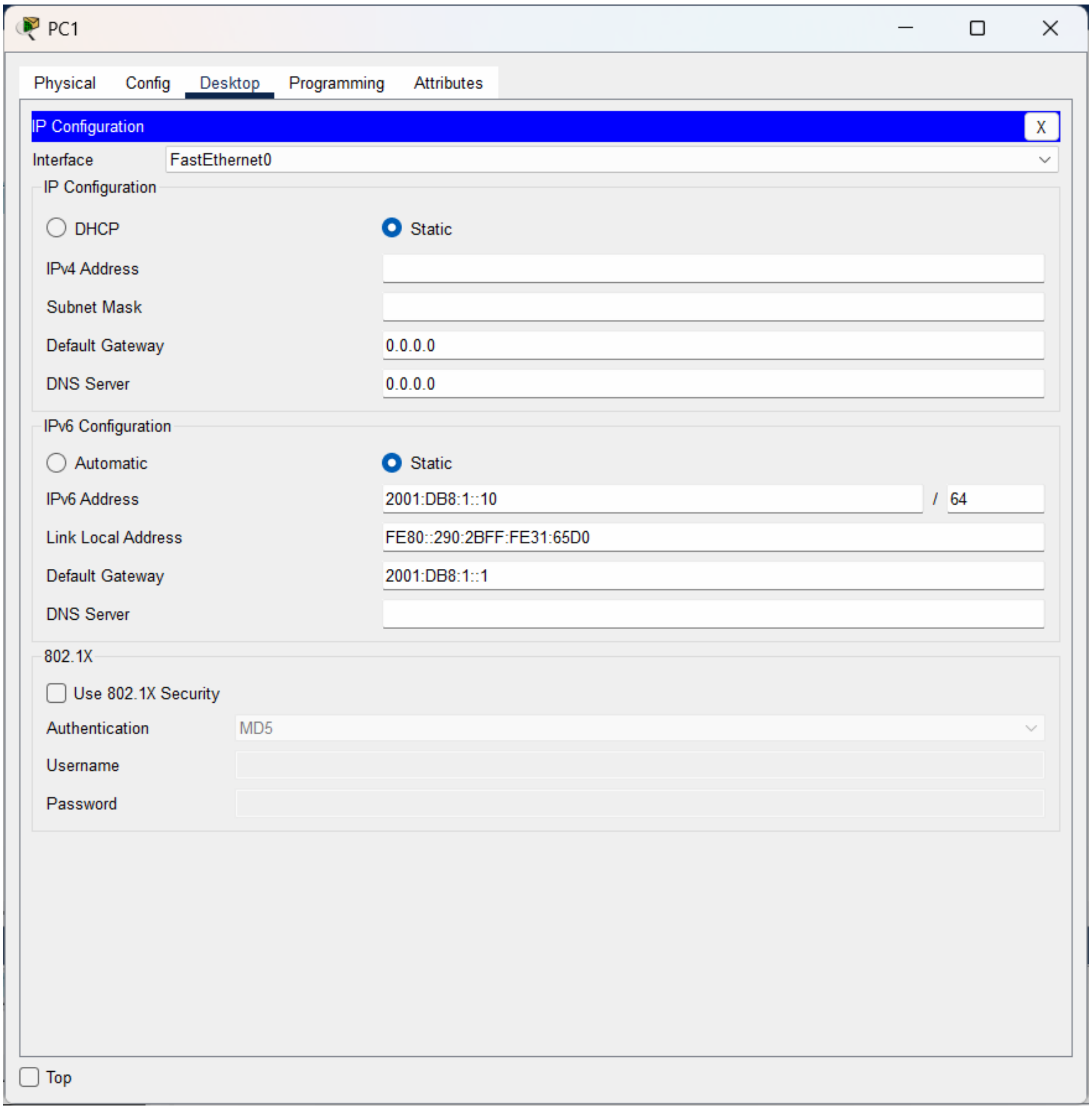
```
Router> enable

Router# configure terminal

Router(config)# interface FastEthernet0/0

Router(config-if)# ipv6 address 2001:db8:1::1/64

Router(config-if)# no shutdown

Router(config-if)# exit

Router(config)# interface FastEthernet0/1

Router(config-if)# ipv6 address 2001:db8:2::1/64

Router(config-if)# no shutdown
```

Router(config-if)# exit

Router(config)# ipv6 unicast-routing

**Step 3 — Configure IPv6 addressing on PCs**

1. Click on **PC1** → go to the **Desktop tab** → select **IP Configuration**.
2. In the **IPv6 Configuration** section:
   - IPv6 Address: 2001:db8:1::10
   - Subnet Prefix Length: 64
   - Default Gateway: 2001:db8:1::1
3. Close PC1's config window.



Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

| | |
|---|---|
| IPv4 Address | |
| Subnet Mask | |
| Default Gateway | 0.0.0.0 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

| | |
|---|---|
| IPv6 Address | 2001:DB8:1::10 / 64 |
| Link Local Address | FE80::290:2BFF:FE31:65D0 |
| Default Gateway | 2001:DB8:1::1 |
| DNS Server | |

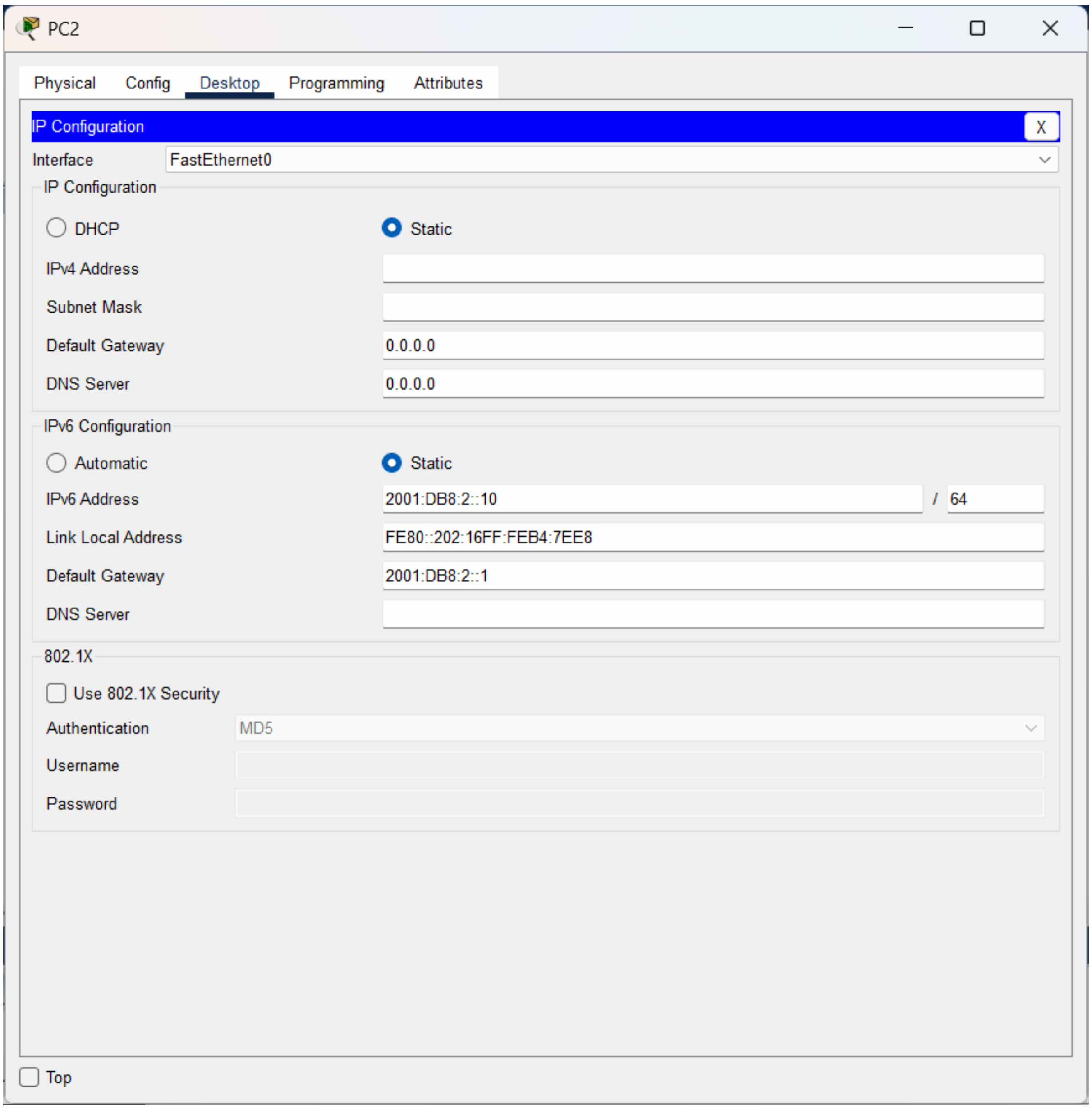
802.1X

☐ Use 802.1X Security

| | |
|---|---|
| Authentication | MD5 |
| Username | |
| Password | |

☐ Top

4. Do the same for **PC2**:
   - IPv6 Address: 2001:db8:2::10
   - Subnet Prefix Length: 64
   - Default Gateway: 2001:db8:2::1



PC2

Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

|  |  |
|---|---|
| IPv4 Address |  |
| Subnet Mask |  |
| Default Gateway | 0.0.0.0 |
| DNS Server | 0.0.0.0 |

IPv6 Configuration

Automatic [✓ Static]

|  |  |
|---|---|
| IPv6 Address | 2001:DB8:2::10 / 64 |
| Link Local Address | FE80::202:16FF:FEB4:7EE8 |
| Default Gateway | 2001:DB8:2::1 |
| DNS Server |  |

802.1X

☐ Use 802.1X Security

|  |  |
|---|---|
| Authentication | MD5 |
| Username |  |
| Password |  |

☐ Top

**Step 4 — Verify IPv6 connectivity**

1. On **PC1** → go to **Desktop tab** → open **Command Prompt**.

   Try pinging PC2's IPv6 address:

   ```
   ping 2001:db8:2::10
   ```

If everything is configured correctly, you should see replies.

2. Also, test by pinging the **router gateways**:

   - From PC1 →

     ping 2001:db8:1::1

     ping 2001:db8:2::1

   - From PC2 →

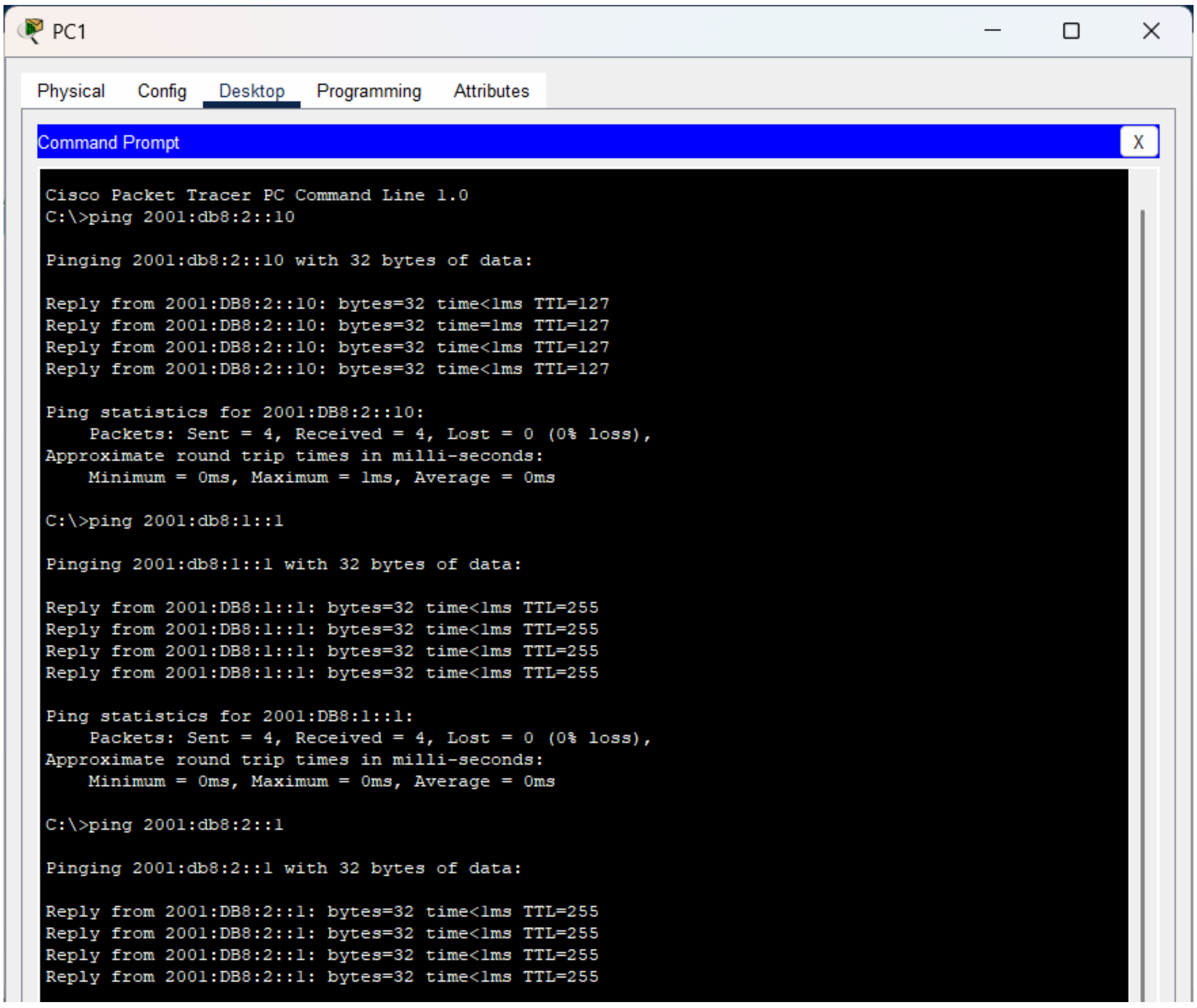
     ping 2001:db8:2::1

     ping 2001:db8:1::1

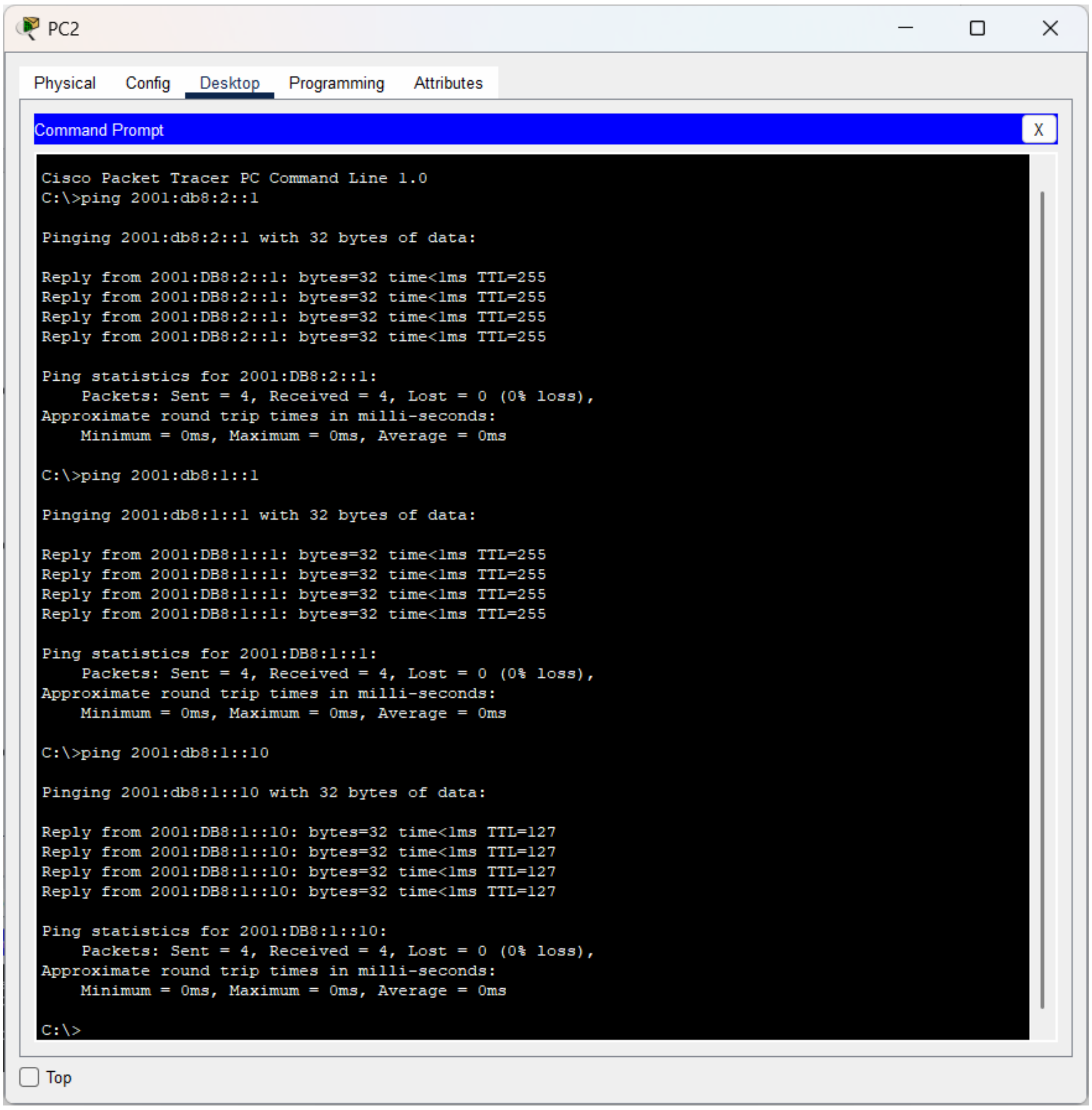
3. Finally, test **end-to-end** again:

   - From PC2 →

     ping 2001:db8:1::10



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 2001:db8:2::10

Pinging 2001:db8:2::10 with 32 bytes of data:

Reply from 2001:DB8:2::10: bytes=32 time<1ms TTL=127
Reply from 2001:DB8:2::10: bytes=32 time=1ms TTL=127
Reply from 2001:DB8:2::10: bytes=32 time<1ms TTL=127
Reply from 2001:DB8:2::10: bytes=32 time<1ms TTL=127

Ping statistics for 2001:DB8:2::10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>ping 2001:db8:1::1

Pinging 2001:db8:1::1 with 32 bytes of data:

Reply from 2001:DB8:1::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:1::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:1::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:1::1: bytes=32 time<1ms TTL=255

Ping statistics for 2001:DB8:1::1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 2001:db8:2::1

Pinging 2001:db8:2::1 with 32 bytes of data:

Reply from 2001:DB8:2::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:2::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:2::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:2::1: bytes=32 time<1ms TTL=255
```



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 2001:db8:2::1

Pinging 2001:db8:2::1 with 32 bytes of data:

Reply from 2001:DB8:2::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:2::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:2::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:2::1: bytes=32 time<1ms TTL=255

Ping statistics for 2001:DB8:2::1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 2001:db8:1::1

Pinging 2001:db8:1::1 with 32 bytes of data:

Reply from 2001:DB8:1::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:1::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:1::1: bytes=32 time<1ms TTL=255
Reply from 2001:DB8:1::1: bytes=32 time<1ms TTL=255

Ping statistics for 2001:DB8:1::1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 2001:db8:1::10

Pinging 2001:db8:1::10 with 32 bytes of data:

Reply from 2001:DB8:1::10: bytes=32 time<1ms TTL=127
Reply from 2001:DB8:1::10: bytes=32 time<1ms TTL=127
Reply from 2001:DB8:1::10: bytes=32 time<1ms TTL=127
Reply from 2001:DB8:1::10: bytes=32 time<1ms TTL=127

Ping statistics for 2001:DB8:1::10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

If you get successful replies you have completed the **IPv6 addressing practical**.

**Title: Configure a Basic Wireless Network using 802.11 Standards in Packet Tracer**

## Objective

To configure a basic wireless network using 802.11 standards in Packet Tracer and verify wireless connectivity between devices.

## Background / Theory

A wireless network allows devices to communicate without physical cables, using radio waves. The **IEEE 802.11** family of standards defines wireless LAN (WLAN) communication, including several versions:

- **802.11a** – Operates at 5 GHz, maximum speed 54 Mbps.
- **802.11b** – Operates at 2.4 GHz, maximum speed 11 Mbps.
- **802.11g** – Operates at 2.4 GHz, maximum speed 54 Mbps.
- **802.11n** – Operates at 2.4/5 GHz, maximum speed 600 Mbps.
- **802.11ac** – Operates at 5 GHz, high throughput and range.

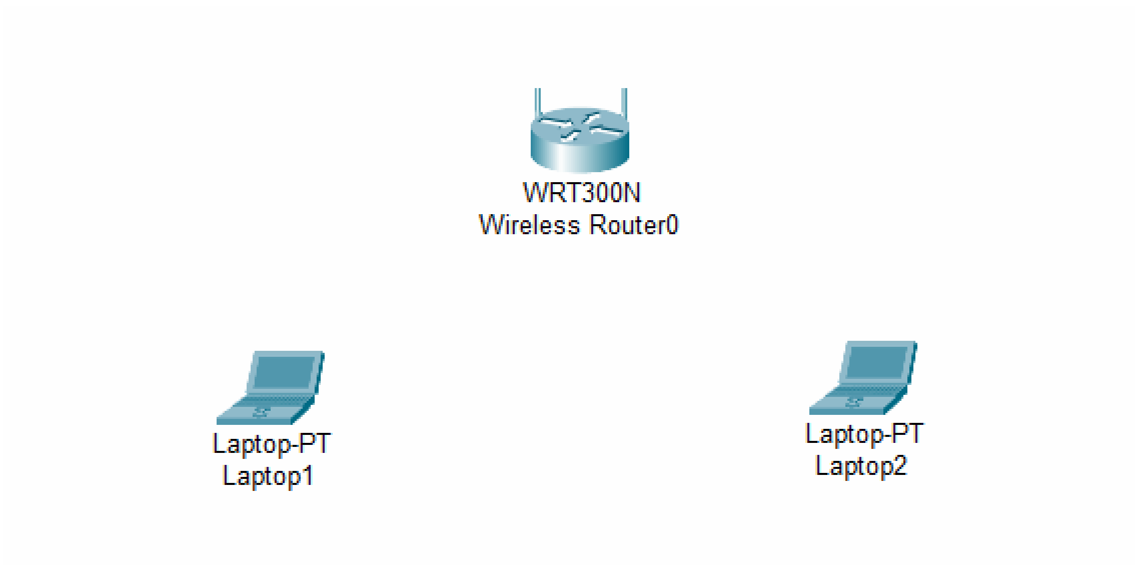
In wireless networks, an **Access Point (AP)** serves as a central device that connects wireless clients (laptops, PCs, smartphones) to the network. Wireless devices communicate using SSIDs (network names) and optional security keys (WPA/WPA2).

## Experiment Steps:

### Step 1 — Add wireless networking devices

1. From the bottom-left device menu, click on Wireless Devices (antenna icon).
2. Drag a Wireless Router (e.g., WRT300N) into the workspace.
3. From End Devices, drag at least two laptops into the workspace.
4. Place them near the wireless router in the logical view.

This sets up the base devices for your wireless LAN.



**Step 2 — Connect devices wirelessly**

1. Select each **Laptop** one by one.
2. Go to the **Physical tab** → turn off the device (power button).
3. Remove the existing wired **FastEthernet NIC**.
4. From the **Modules list**, add a **Wireless WPC300N card**.
5. Turn the device back on.

Now your end devices can connect wirelessly to the router.

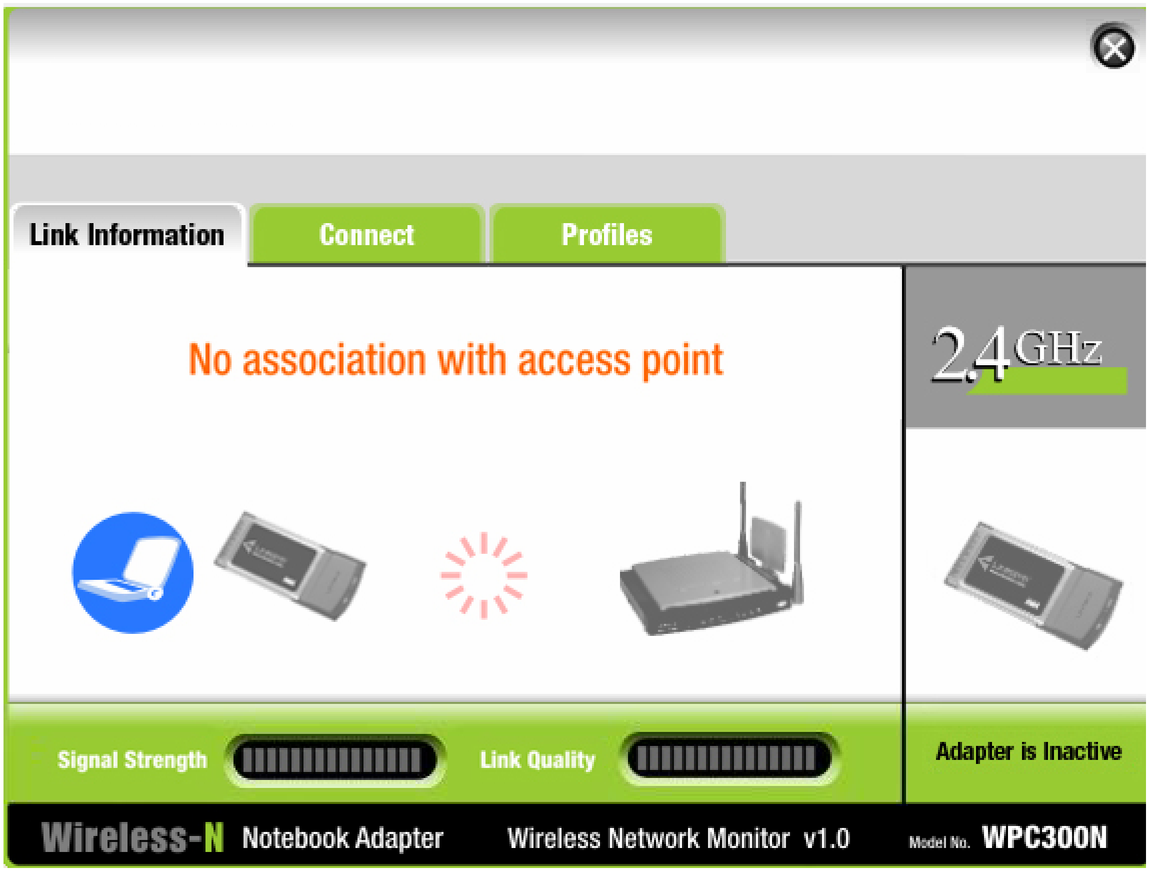
**Step 3 — Configure the wireless router (SSID & security)**

1. Click on the **Wireless Router**.
2. Go to the **Config tab**.
3. From the left menu, select the **Wireless interface**:
   - Set **Network Name (SSID)** → e.g., My_WiFi.
   - Choose **Authentication** → e.g., WPA2-PSK.
   - Enter a passphrase (e.g., cisco1234).

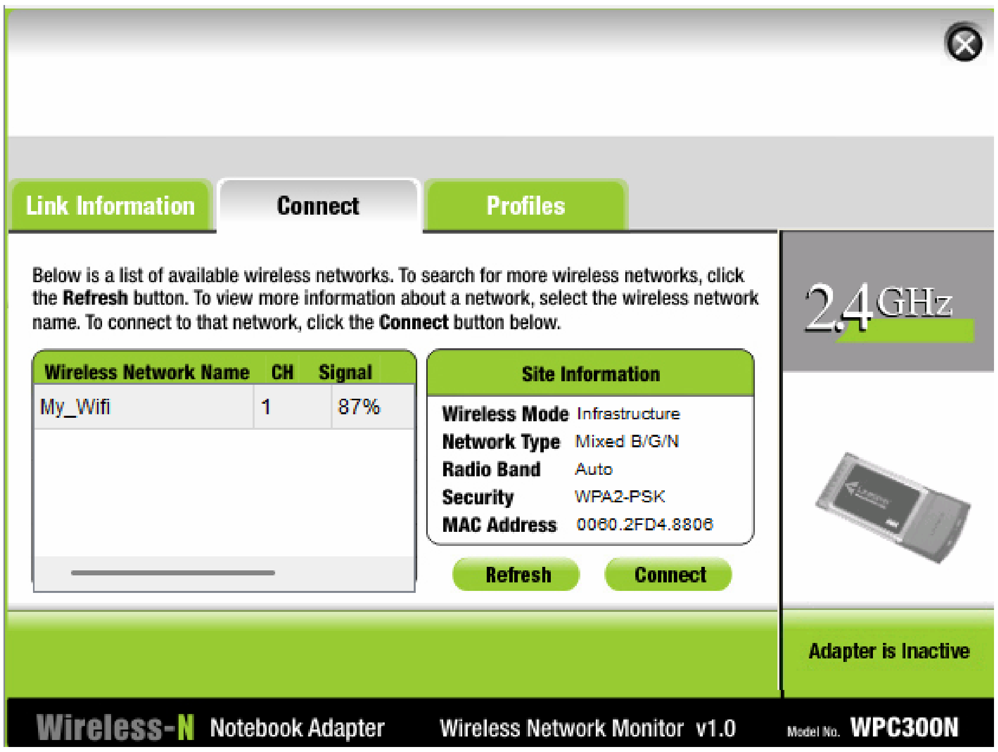
This creates your wireless network with 802.11 standards and security.

**Step 4 — Connect laptops to the Wi-Fi**

1. Click on a **Laptop**.
2. Go to the **Desktop tab** → open **PC Wireless**.



3. Click **Connect** tab → you'll see available wireless networks.
4. Select your SSID (e.g., My_WiFi).



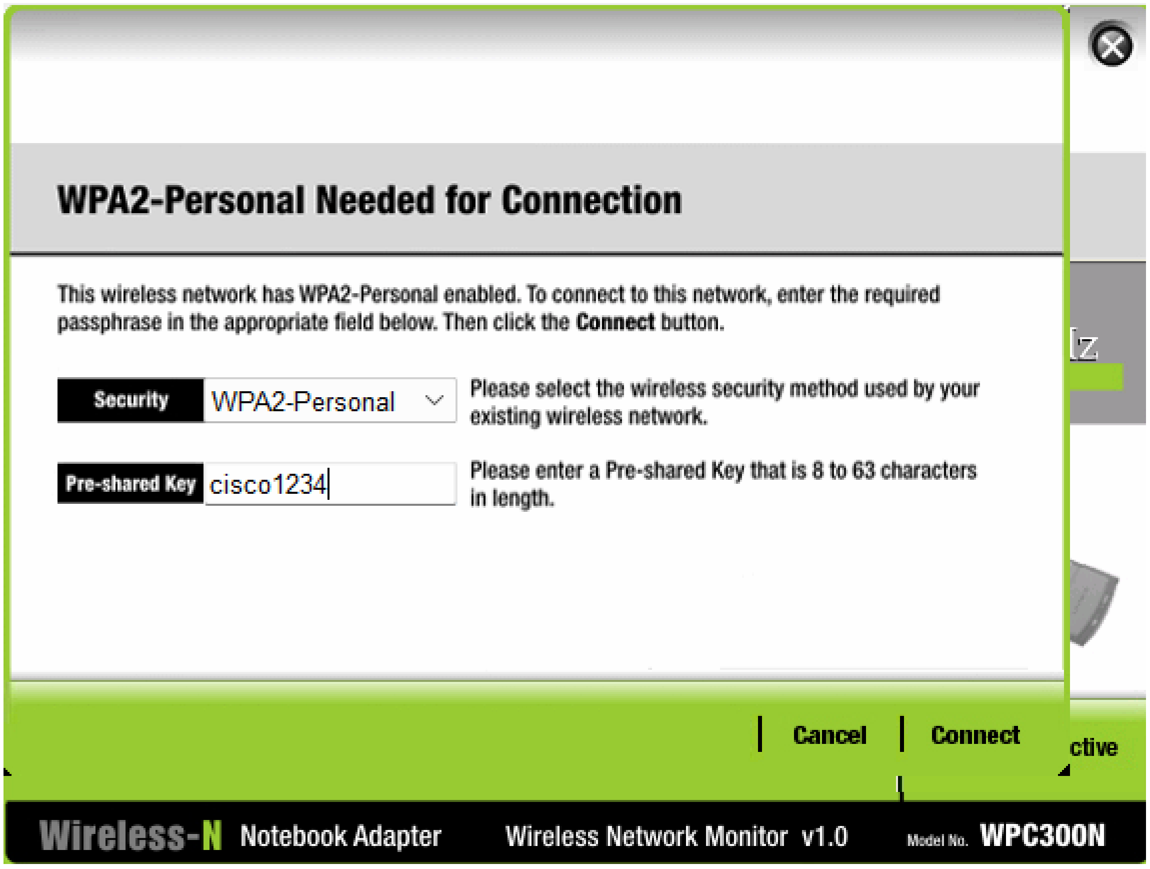
Below is a list of available wireless networks. To search for more wireless networks, click the **Refresh** button. To view more information about a network, select the wireless network name. To connect to that network, click the **Connect** button below.

| Wireless Network Name | CH | Signal |
|---|---|---|
| My_Wifi | 1 | 87% |

**Site Information**

| | |
|---|---|
| **Wireless Mode** | Infrastructure |
| **Network Type** | Mixed B/G/N |
| **Radio Band** | Auto |
| **Security** | WPA2-PSK |
| **MAC Address** | 0060.2FD4.8806 |

5. Enter the password you set (e.g., cisco1234) → click **Connect**.



**WPA2-Personal Needed for Connection**

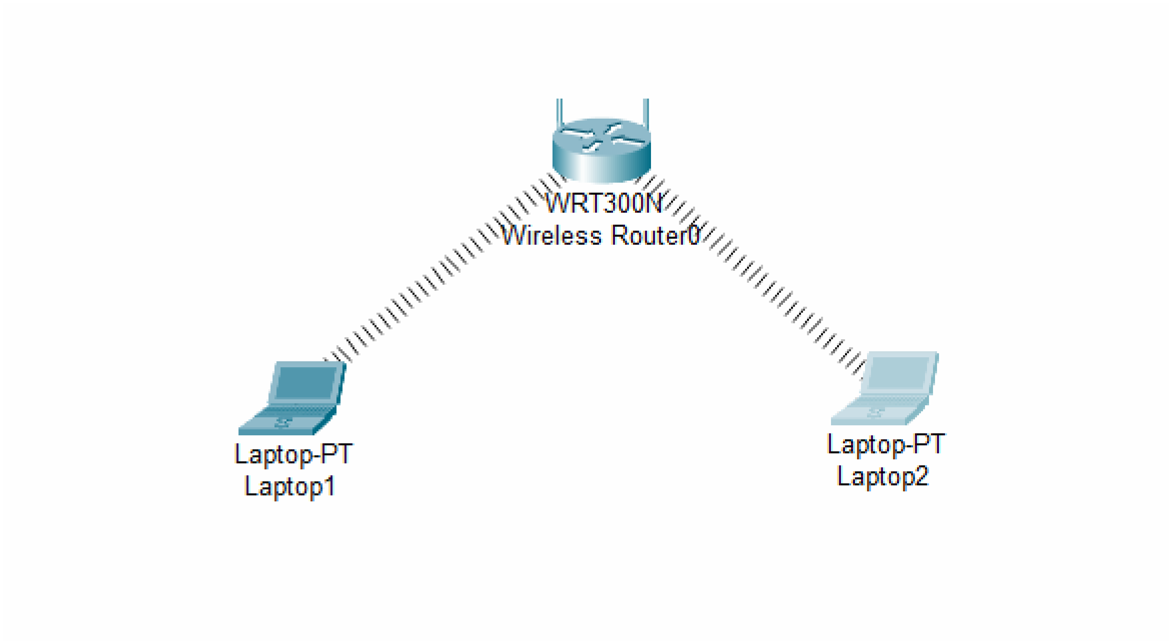
This wireless network has WPA2-Personal enabled. To connect to this network, enter the required passphrase in the appropriate field below. Then click the **Connect** button.

Security: WPA2-Personal — Please select the wireless security method used by your existing wireless network.

Pre-shared Key: cisco1234 — Please enter a Pre-shared Key that is 8 to 63 characters in length.

6. Repeat for the second Laptop.

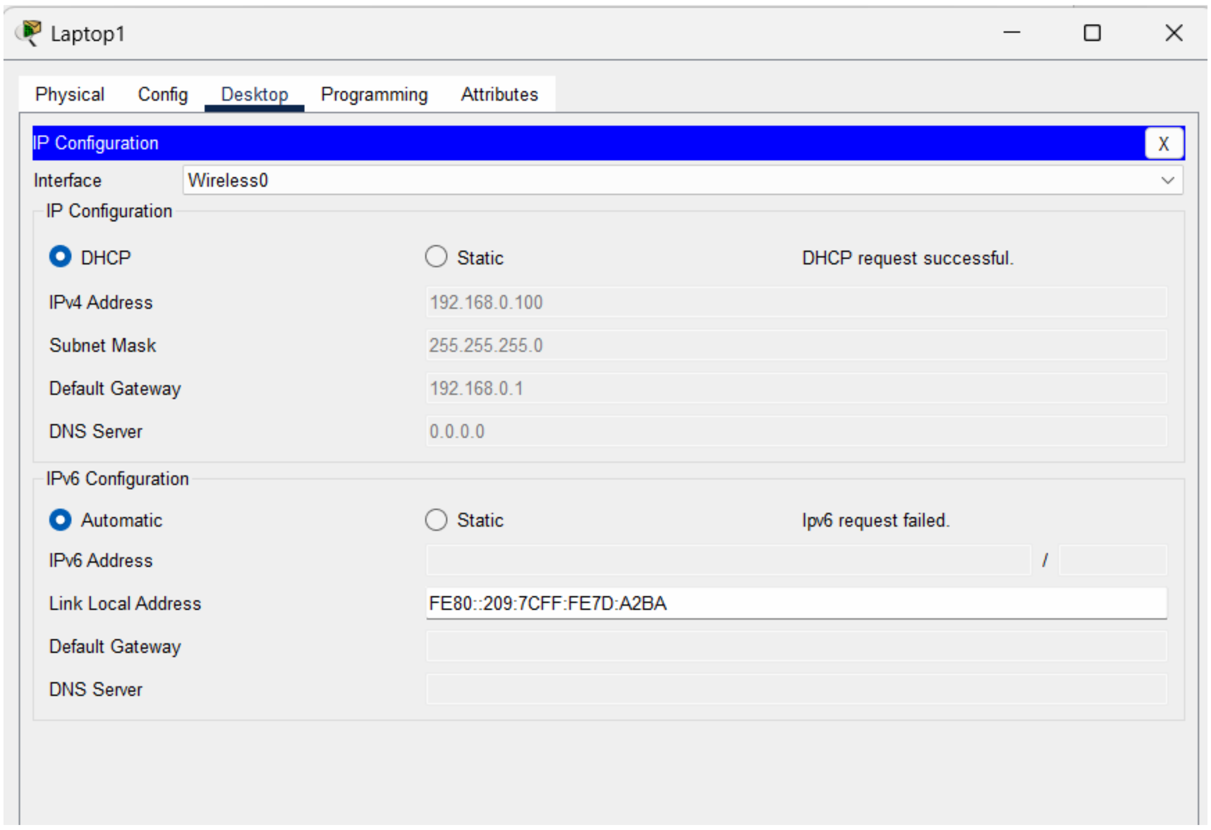
Now both laptops should be wirelessly associated with the router.



**Step 5 – Verification through ping commands**

1. On Laptop1, go to **Desktop -> IP Configuration.**

2. You will see that **IPv4 address** is already assigned to the laptop (e.g., 192.168.0.100), **note it down.**



Laptop1

Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: Wireless0

IP Configuration

DHCP [✓] Static — DHCP request successful.

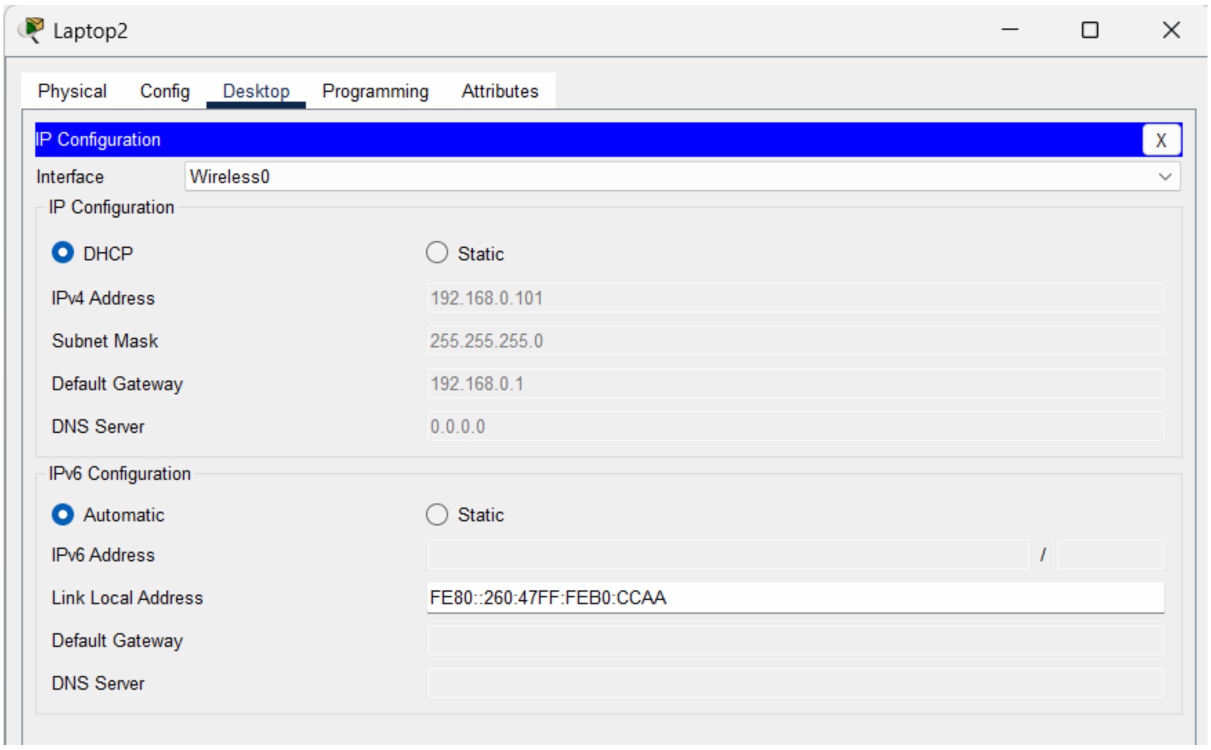
IPv4 Address: 192.168.0.100
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.0.1
DNS Server: 0.0.0.0

IPv6 Configuration

Automatic [✓] Static — Ipv6 request failed.

IPv6 Address: /
Link Local Address: FE80::209:7CFF:FE7D:A2BA
Default Gateway:
DNS Server:

3. Repeat the same for **Laptop2**.



Laptop2

Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: Wireless0

IP Configuration

DHCP [✓] Static

IPv4 Address: 192.168.0.101
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.0.1
DNS Server: 0.0.0.0

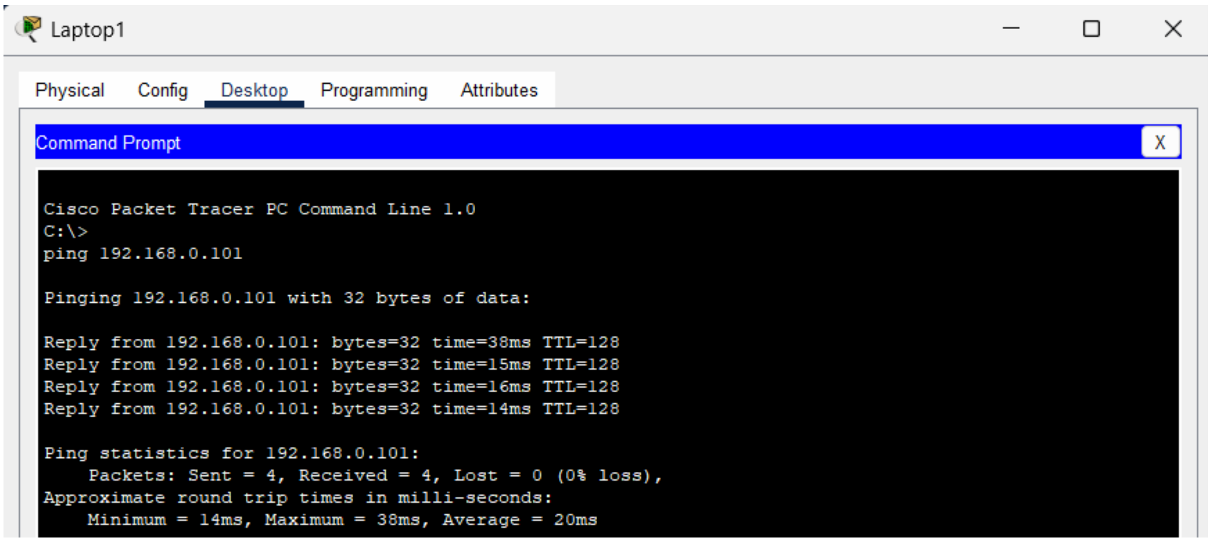
IPv6 Configuration

Automatic [✓] Static

IPv6 Address: /
Link Local Address: FE80::260:47FF:FEB0:CCAA
Default Gateway:
DNS Server:

4. Now on Laptop1, go to **Desktop -> Command Prompt**, ping with the **IP Address of Laptop2** that you have noted (e.g., 192.168.0.101):

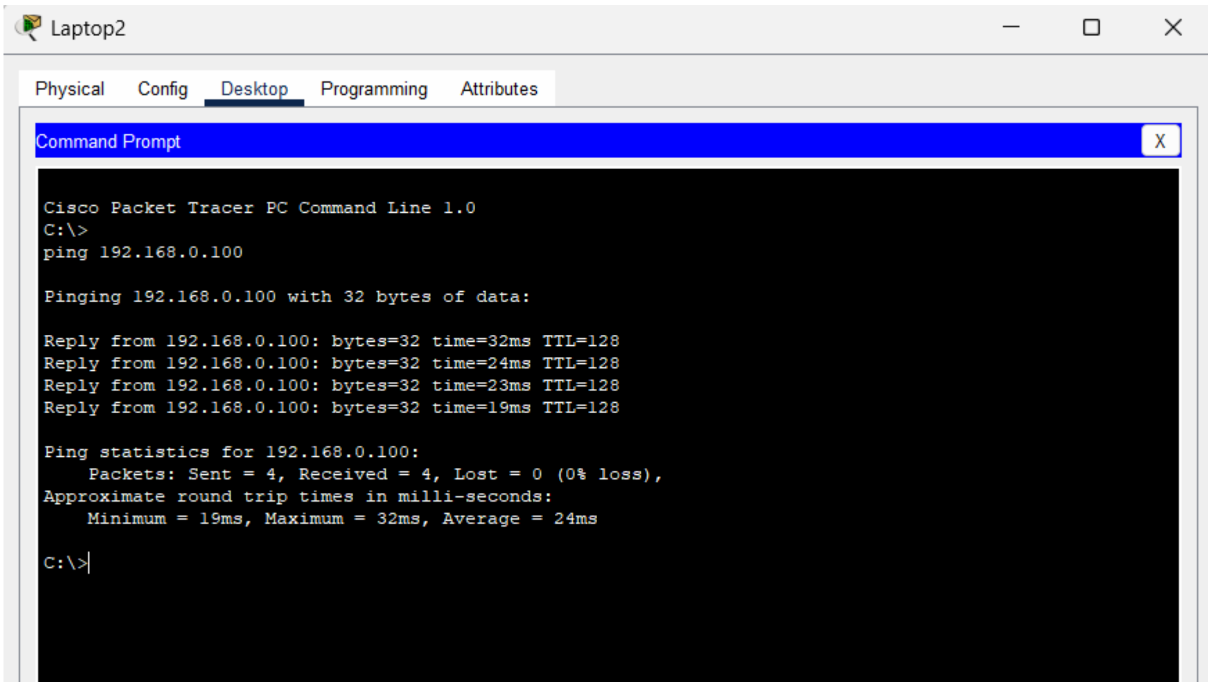
   ping 192.168.0.101



```
Cisco Packet Tracer PC Command Line 1.0
C:\>
ping 192.168.0.101

Pinging 192.168.0.101 with 32 bytes of data:

Reply from 192.168.0.101: bytes=32 time=38ms TTL=128
Reply from 192.168.0.101: bytes=32 time=15ms TTL=128
Reply from 192.168.0.101: bytes=32 time=16ms TTL=128
Reply from 192.168.0.101: bytes=32 time=14ms TTL=128

Ping statistics for 192.168.0.101:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 14ms, Maximum = 38ms, Average = 20ms
```

5. On Laptop2, ping with the **IP Address of Laptop1** (e.g., 192.168.0.100):

   ping 192.168.0.100



```
Cisco Packet Tracer PC Command Line 1.0
C:\>
ping 192.168.0.100

Pinging 192.168.0.100 with 32 bytes of data:

Reply from 192.168.0.100: bytes=32 time=32ms TTL=128
Reply from 192.168.0.100: bytes=32 time=24ms TTL=128
Reply from 192.168.0.100: bytes=32 time=23ms TTL=128
Reply from 192.168.0.100: bytes=32 time=19ms TTL=128

Ping statistics for 192.168.0.100:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 19ms, Maximum = 32ms, Average = 24ms

C:\>
```

If both pings succeed, you've **successfully configured a Wireless LAN in Packet Tracer**.