Table of Contents

Name of Experiment: Configuring a server & connecting to a device to create a LAN

1. Introduction

A Local Area Network (LAN) connects devices within a limited area, enabling resource sharing and communication. This lab demonstrates configuring a server as the central node and connecting it to devices like PCs and switches to establish a functional LAN.

2. Objective

The purpose of this lab is to configure a server and connect it to multiple devices (e.g., PCs, switches) to create a Local Area Network (LAN). The goal is to ensure successful communication between all connected devices.

3. Equipment & Software Used

- Software: Cisco Packet Tracer
- Devices:1 Server,2 PCs,1 Switch
- Cables: Ethernet cables (Straight-through)

4. Network Topology

This section explains the network setup where a server is connected to multiple devices through a switch, creating a LAN.

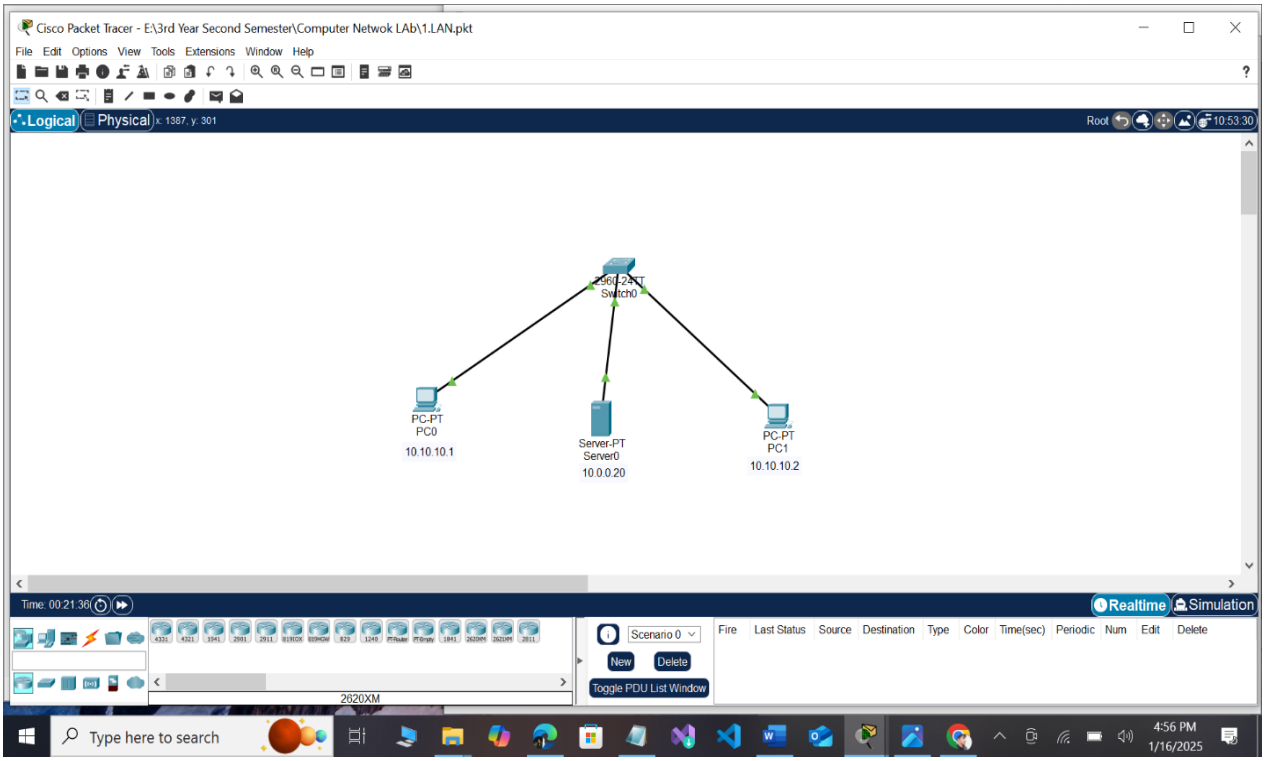


Figure: Topology of the main network

5. Task Details

Below are the steps for configuring the server and connecting devices. Each step includes an explanation and the corresponding screenshot.

Task 1: Adding Devices to the Workspace

- **Objective:** Add a server, a switch, and PCs to the Packet Tracer workspace.
- **Steps Performed:**
    1. Drag and drop a server from the “End Devices” menu.
    2. Drag and drop a switch from the “Switches” menu.
    3. Drag and drop PCs from the “End Devices” menu.

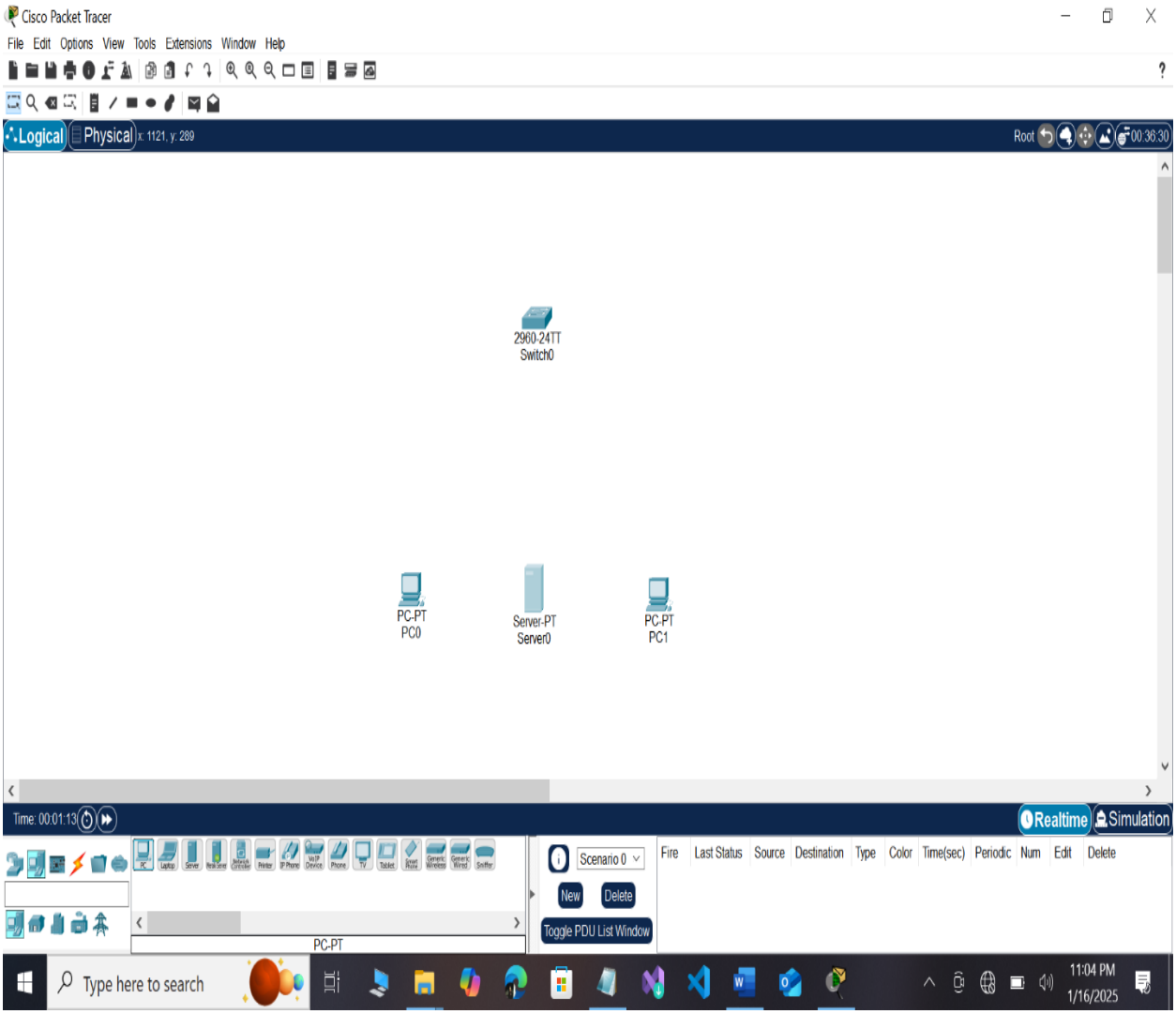


Figure: Adding Devices to the Workspace

Task 2: Connecting Devices Using Cables

- **Objective:** Connect all devices to the switch using Ethernet cables
- **Steps Performed:**
    1. Select the “Connections” tool and choose the straight-through cable.
    2. Connect the server to the switch.
    3. Connect each PC to the switch.

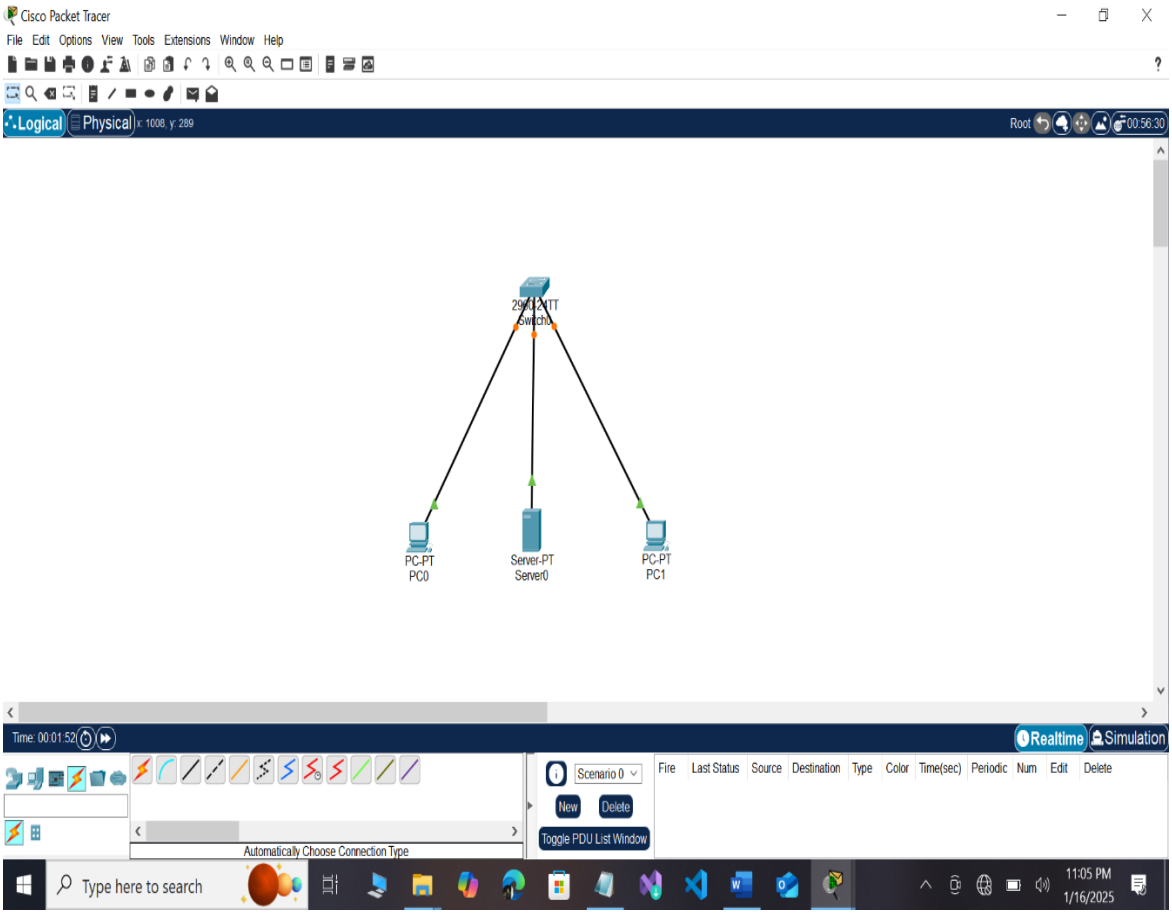


Figure: Connecting Devices Using Cables

Task 3: Configuring the Server

- **Objective:** Assign an IP address and services to the server.
- **Steps Performed:**
    1. Click on the server to open its configuration panel.
    2. Navigate to the “Config” tab and assign an IP address (e.g., 10.0.0.20) and subnet mask (e.g., 255.0.0.0).

3. Enable necessary services (HTTP).

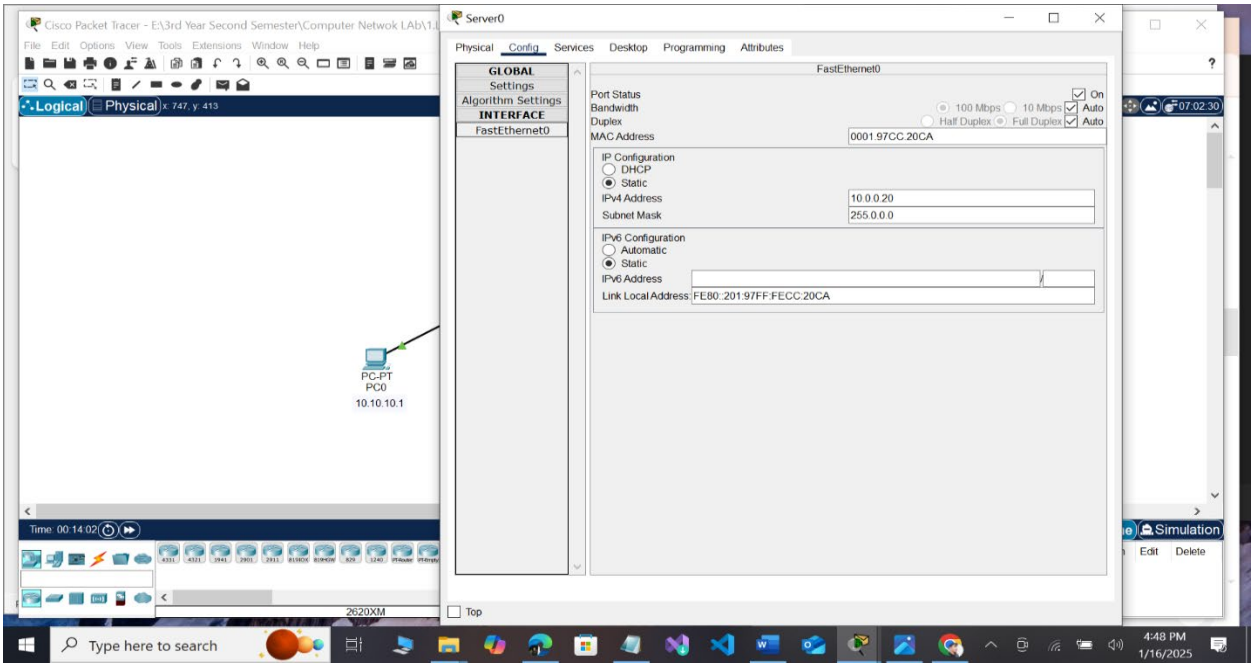


Figure: Assign an IP address

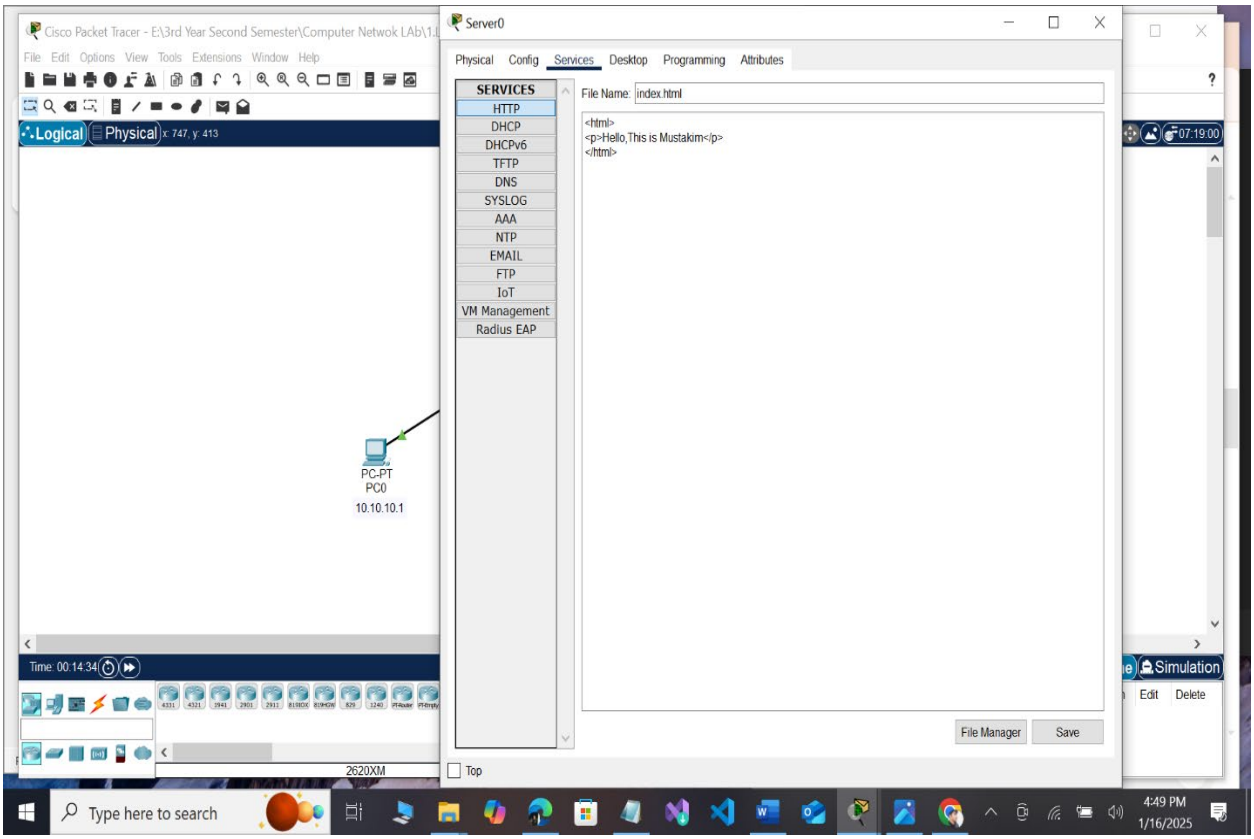


Figure: Assign services to the server.

Task 4: Configuring IP Addresses on PCs

- **Objective:** Assign static IP addresses to the PCs.
- **Steps Performed:**
    1. Click on each PC and navigate to the “Desktop” tab.
    2. Open the “IP Configuration” tool.
    3. Assign static IPs (10.10.10.1, 10.10.10.2,) or enable DHCP if configured on the server.

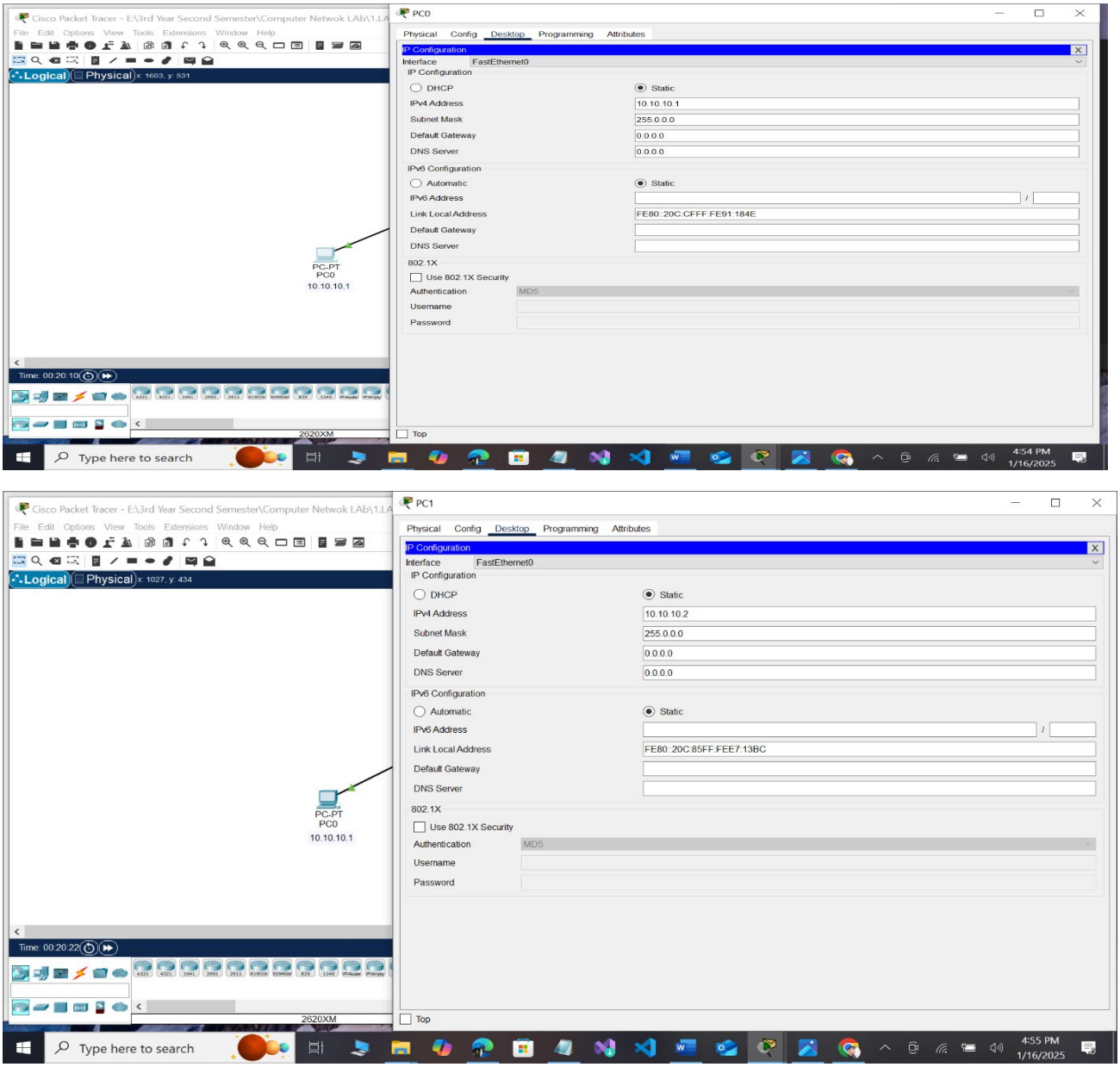


Figure: Assign static IP addresses to the PCs.

Task 5: Verifying Connectivity

- **Objective:** Test communication between devices using ping commands.
- **Steps Performed:**
    1. Open the command prompt on one PC.
    2. Ping the server and other PCs to verify connectivity.

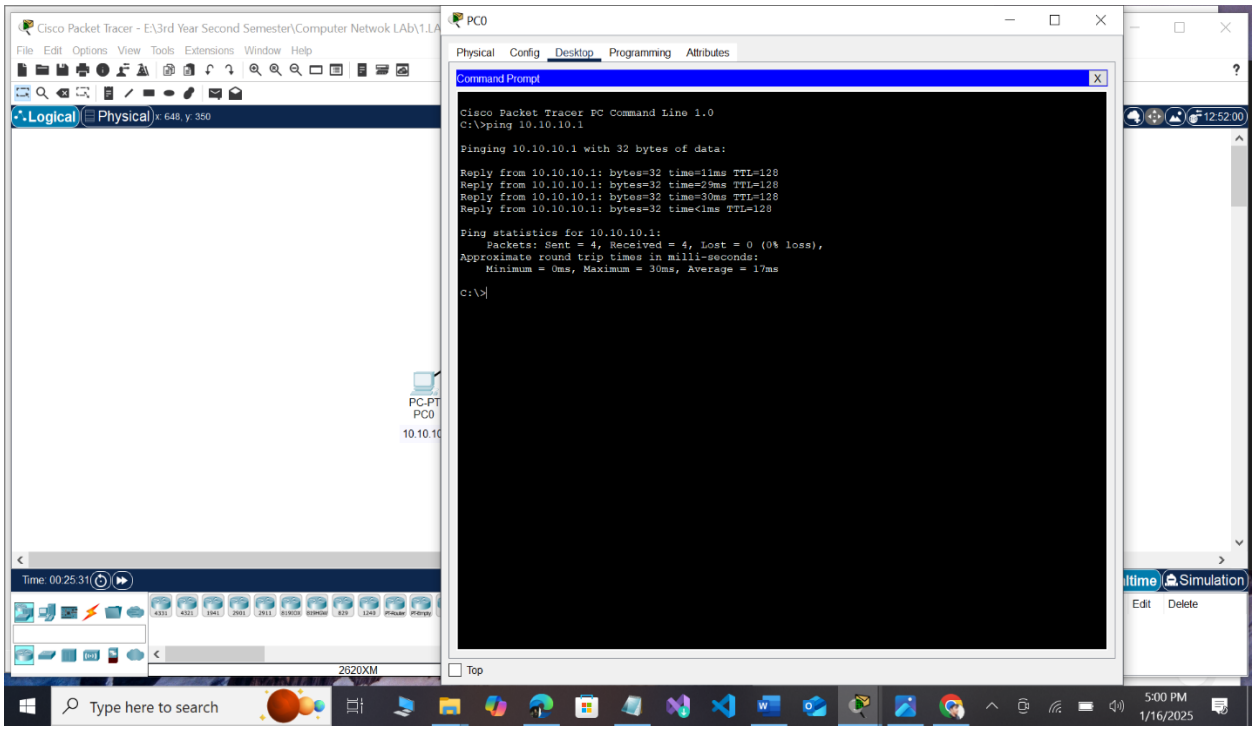


Figure Test communication between devices using ping commands.

6. Observations & Results

- All devices were successfully connected to the server via the switch.
- IP addresses were configured, and the network was verified to be operational.
- The ping results showed successful communication between the server and devices, confirming LAN connectivity.

7. Conclusion

In this lab, a server was configured and connected to devices through a switch to establish a Local Area Network (LAN). The objectives were met as the devices successfully communicated with each other. This task demonstrated basic network configuration and connectivity troubleshooting in Cisco Packet Tracer.

Name of Experiment: Configuring HTTP Service in a Server

1. Introduction

This lab focuses on setting up and configuring the HTTP service on a server, allowing it to host a webpage. A PC is connected to access the hosted webpage using a browser. Additionally, a DNS server is implemented to resolve domain names to IP addresses, ensuring smoother communication between devices in the network.

2. Objective

The purpose of this lab is to configure the HTTP service on a server and ensure that a PC can access a webpage hosted on the web server. A DNS server is used to resolve domain names to IP addresses.

3. Equipment & Software Used

- **Software:** Cisco Packet Tracer
- **Devices:**1 PC,1 Web Server,1 DNS Server
- **Cables:** Ethernet cables (Straight-through)

4. Network Topology

This section explains the network setup where a PC, web server, and DNS server are connected to facilitate HTTP communication.

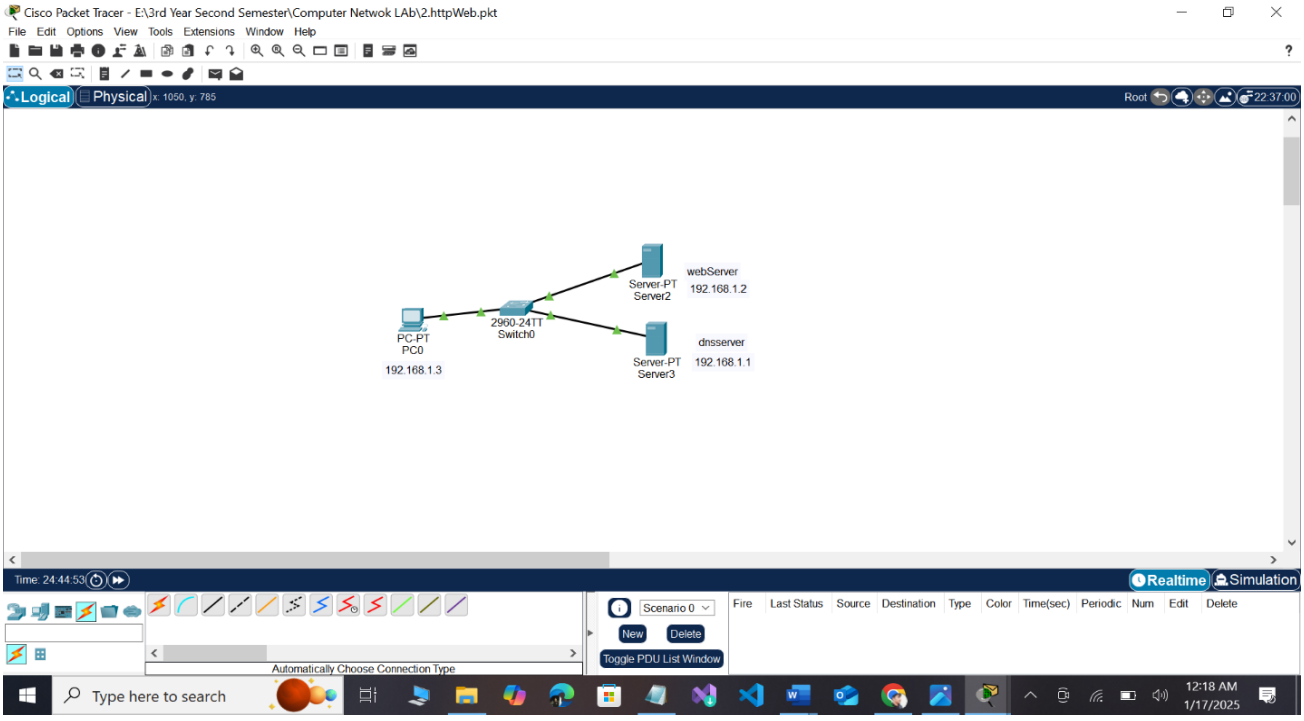


Figure: Mainn topology of the network

5. Task Details

Below are the steps for configuring the HTTP service and connecting devices. Each step includes an explanation and the corresponding screenshot.

Task 1: Adding Devices to the Workspace

- **Objective:** Add a PC, a web server, and a DNS server to the Packet Tracer workspace.
- **Steps Performed:**
    1. Drag and drop a PC from the “End Devices” menu.
    2. Drag and drop a web server and a DNS server from the “Servers” menu.

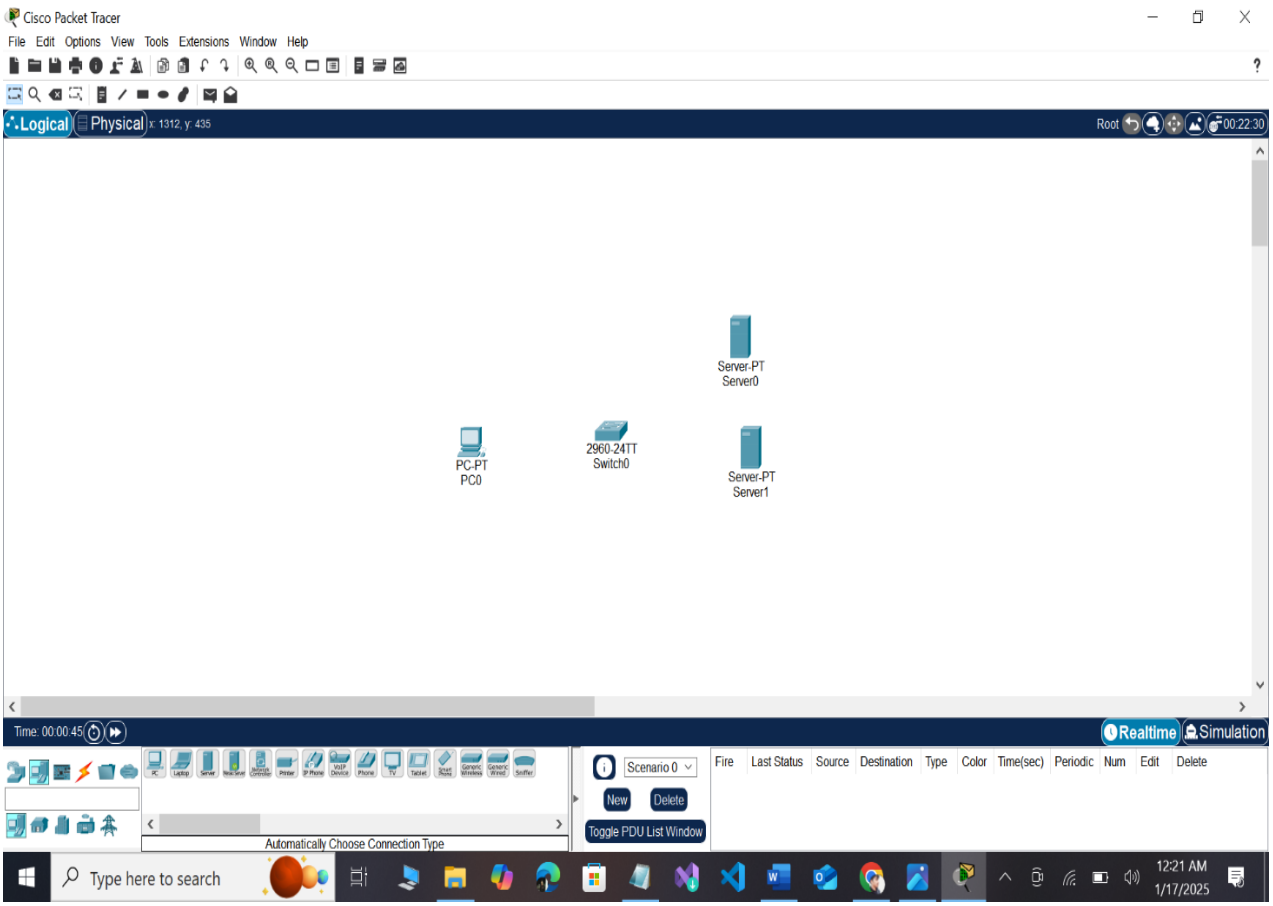


Figure: Adding Devices to the Workspace

Task 2: Connecting Devices Using Cables

- **Objective:** Connect all devices using Ethernet cables.
- **Steps Performed:**
    1. Select the “Connections” tool and choose the straight-through cable.
    2. Connect the PC to the web server and the DNS server.

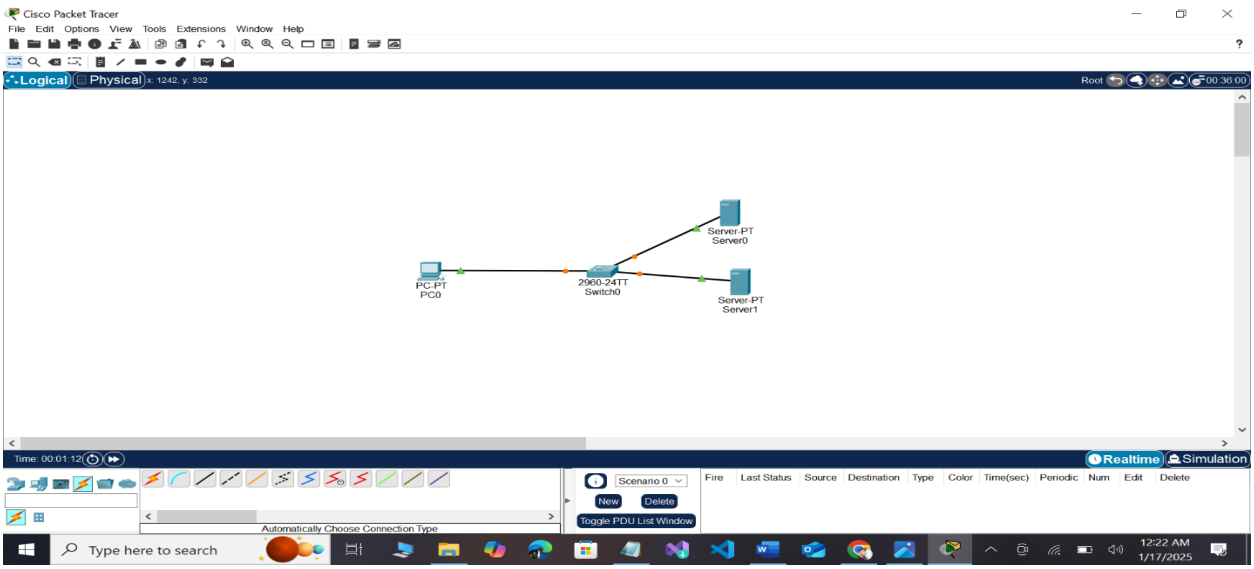


Figure: Connecting Devices Using Cables

Task 3: Configuring the Web Server

- **Objective:** Enable the HTTP service on the web server and upload a sample webpage.
- **Steps Performed:**
    1. Click on the web server to open its configuration panel.
    2. Navigate to the “Services” tab and enable the HTTP service.
    3. Upload a sample webpage by adding content to the HTTP settings.

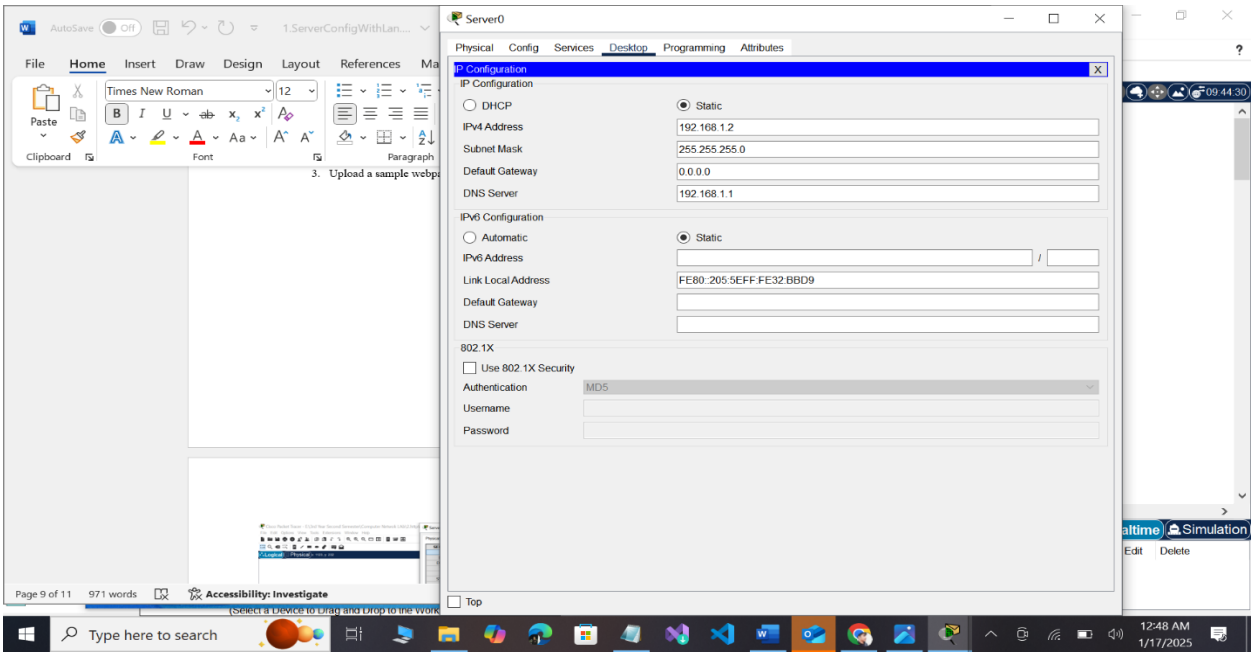


Figure: Configuring IP address

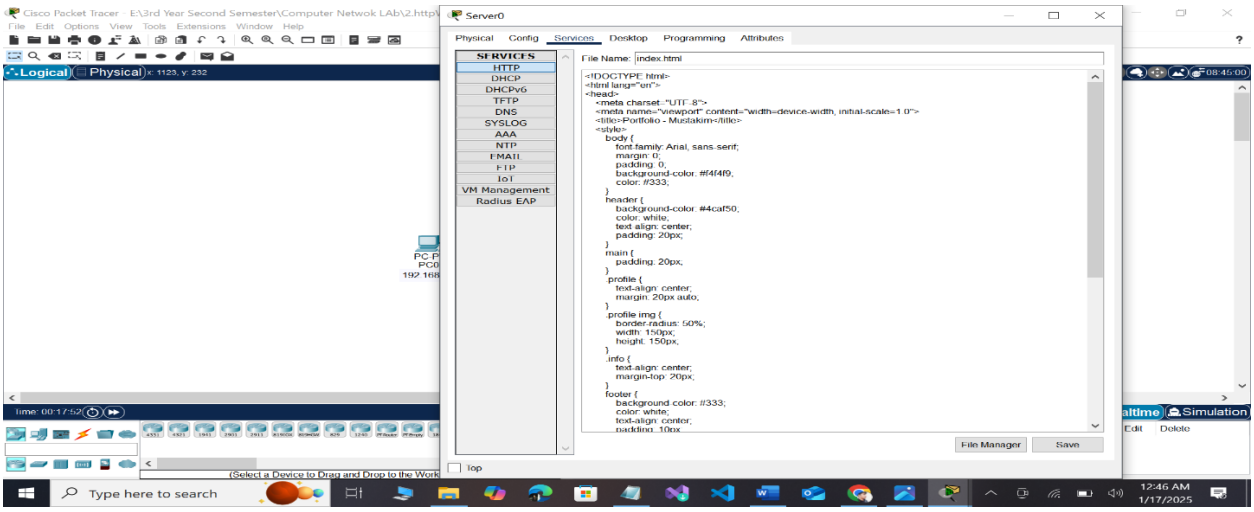


Figure: HTTP service setting modification

Task 4: Configuring the DNS Server

- **Objective:** Set up the DNS server to resolve the domain name of the web server.
- **Steps Performed:**
    1. Click on the DNS server to open its configuration panel.
    2. Navigate to the “Services” tab and enable the DNS service.
    3. Add a DNS record mapping the domain name (e.g., www.mustakim.com) to the web server’s IP address.

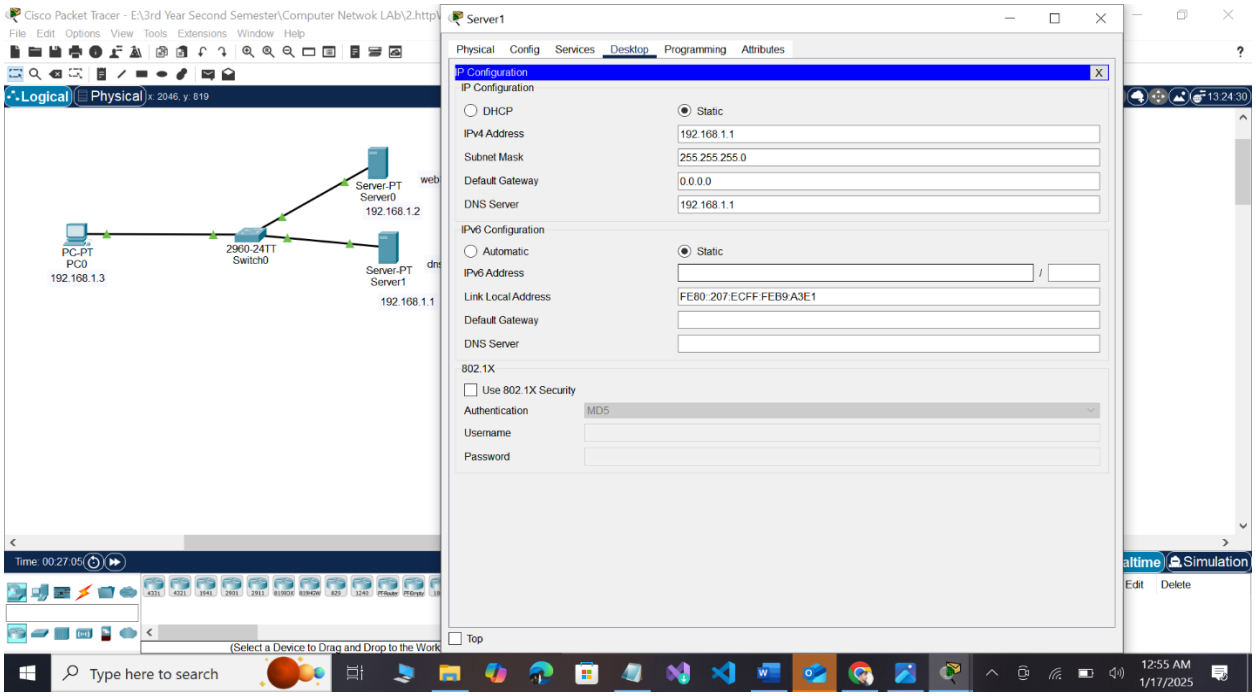


Figure: IP configuration for DNS server

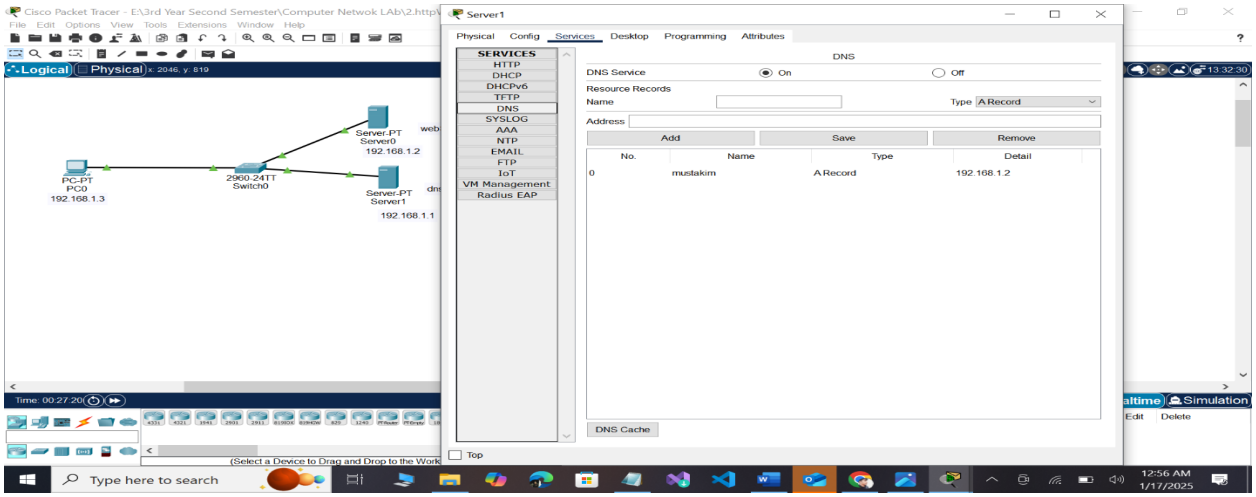


Figure: DNS service setting modification

Task 5: Testing HTTP Access from the PC

- **Objective:** Access the hosted webpage using the PC's browser.
- **Steps Performed:**
    1. Open the PC and navigate to the "Desktop" tab.
    2. Open the web browser and enter the domain name (e.g., www.mustakim.com).
    3. Verify that the webpage loads successfully.

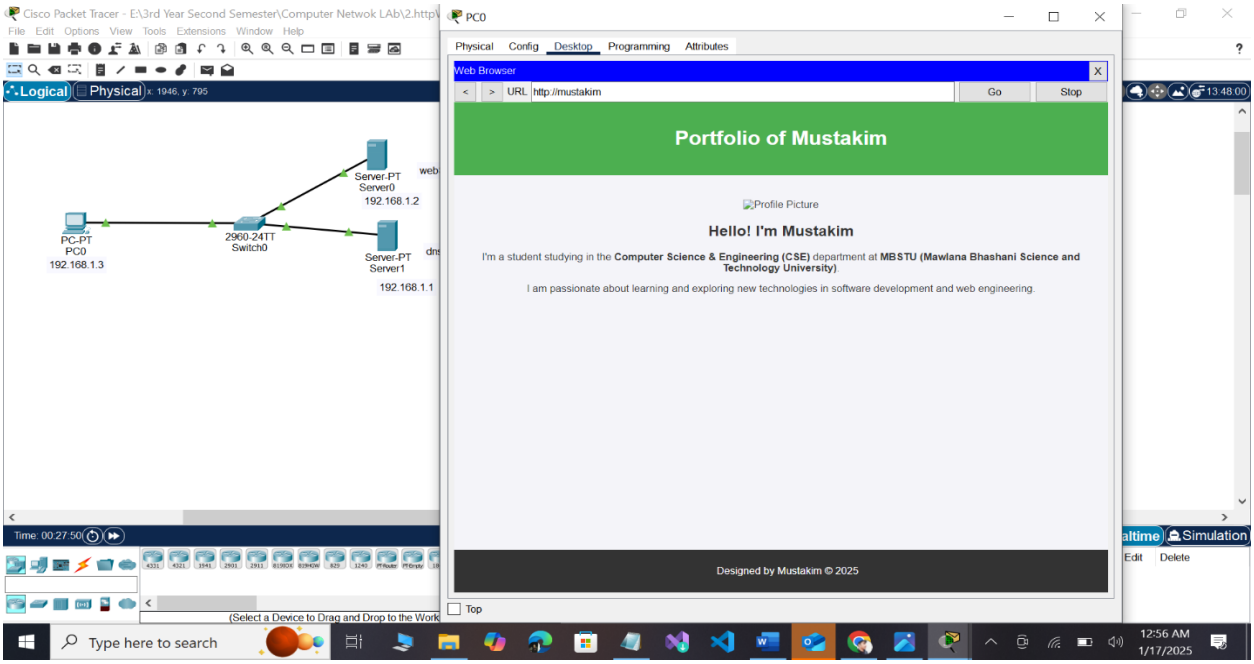


Figure: The webpage displayed on the PC's browser.

6. Observations & Results

- The web server successfully hosted the webpage.
- The DNS server resolved the domain name to the correct IP address.
- The PC was able to access the webpage using the domain name, confirming that the HTTP service was correctly configured.

7. Conclusion

In this lab, the HTTP service was configured on a web server, and a DNS server was set up to resolve domain names. The PC successfully accessed the hosted webpage, demonstrating the functionality of HTTP communication within the network. All objectives were met, and the network operated as expected.

Name of Experiment: Connecting three different LANs and sharing HTTP services.

1. Introduction

This lab demonstrates how to connect three different LANs (MBSTU, DU, and JU) using routers to enable communication across the networks. Each LAN includes a PC, a switch, and a server, with the servers hosting HTTP services. The configuration ensures that devices in one LAN can access resources, such as web pages, hosted in other LANs.

2. Objective

The purpose of this lab is to connect three different LANs (MBSTU, DU, and JU) using routers. Each LAN consists of a PC, a switch, and a server. The goal is to configure the network such that devices from one LAN can communicate with devices in other LANs.

3. Equipment & Software Used

- **Software:** Cisco Packet Tracer
- **Devices:**
    - 3 PCs (one for each LAN)
    - 3 Switches (one for each LAN)
    - 3 Servers (one for each LAN)
    - 3 Routers (one for each LAN)
- **Cables:** Ethernet cables (Straight-through and Cross-over as needed)

4. Network Topology

This section explains the setup where each LAN (MBSTU, DU, JU) is connected to a router, and routers are interconnected to enable communication between the LANs.

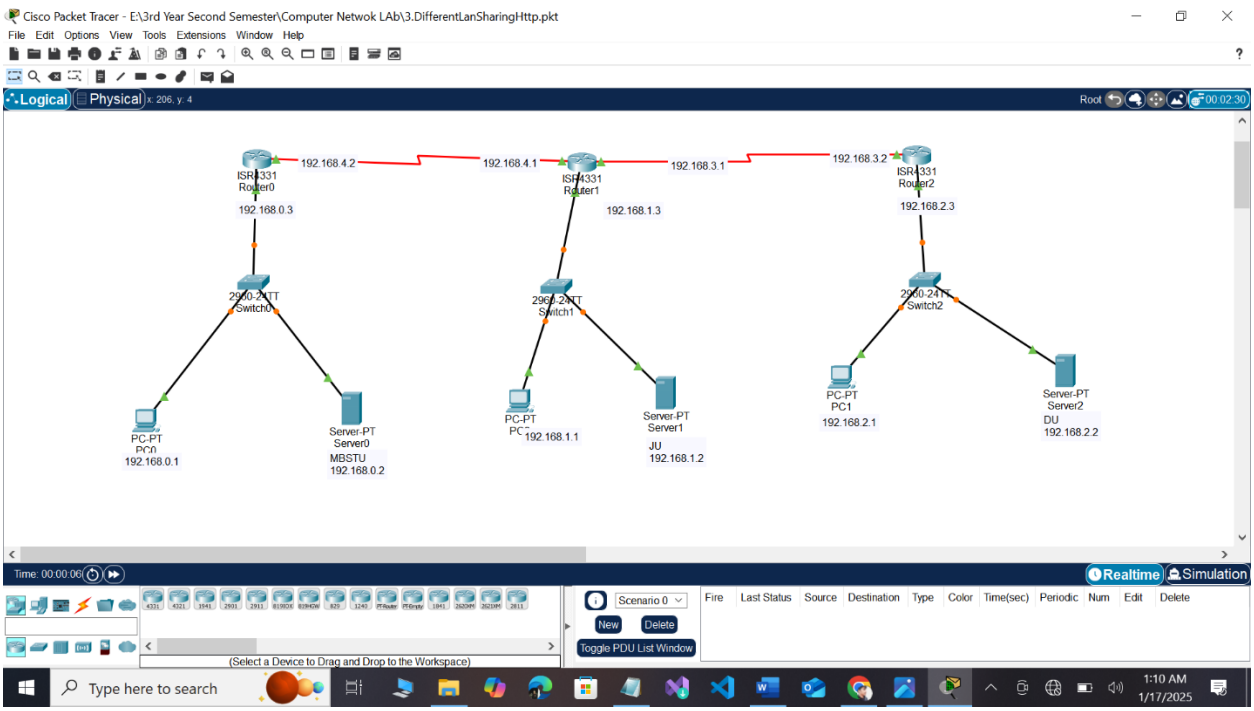


Figure: Connecting three different LANs and sharing HTTP services

5. Configuring IP Addresses on PCs

- **Objective:** Assign static IP addresses to the PCs.
- **Steps Performed:**
    1. Click on each PC and navigate to the “Desktop” tab.
    2. Open the “IP Configuration” tool.
    3. Assign static IPs

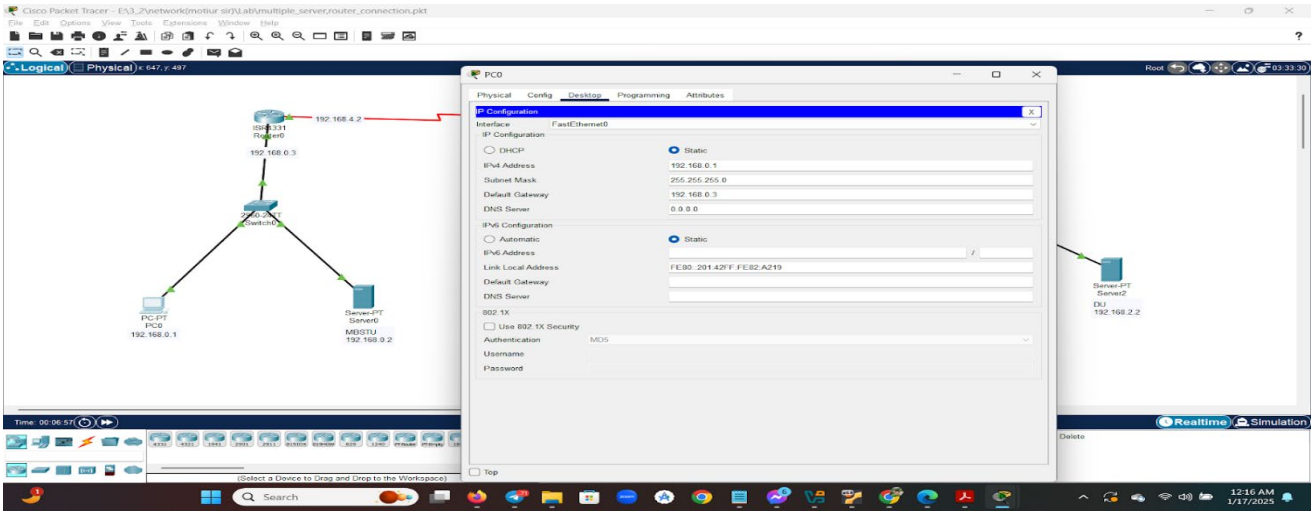


Figure: Set the IP Address of all the PC

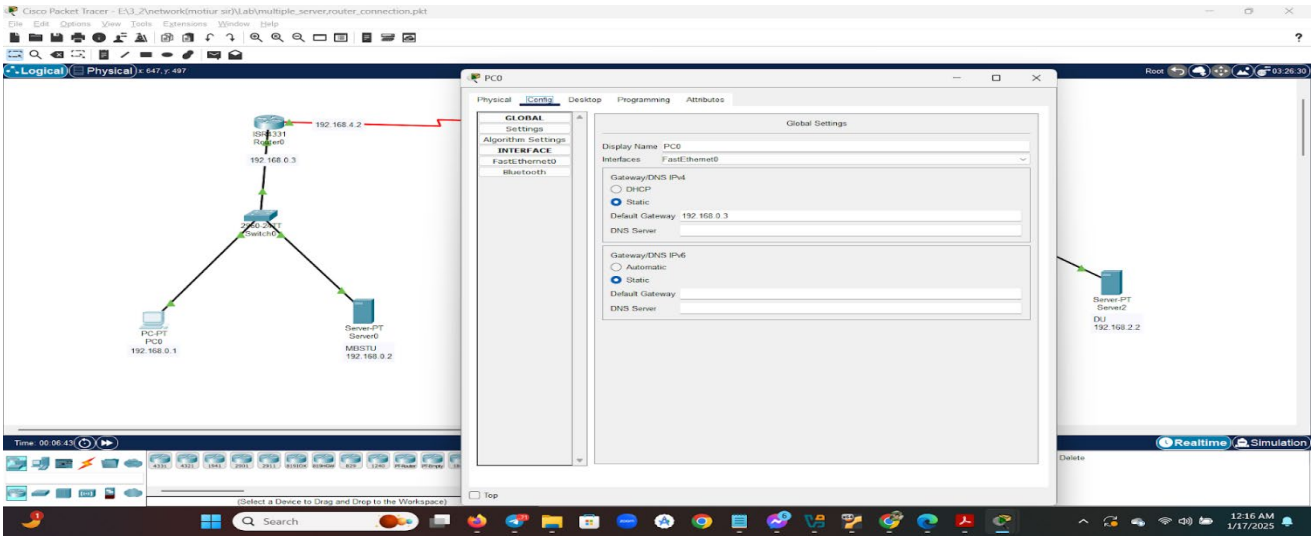


Figure: Set the Getaway in all PC

6. Configuring the Server

- **Objective:** Assign an IP address and services to the server.
- **Steps Performed:**
    1. Click on the server to open its configuration panel.
    2. Navigate to the “Config” tab and assign an IP address and subnet mask.
    3. Enable necessary services (HTTP).

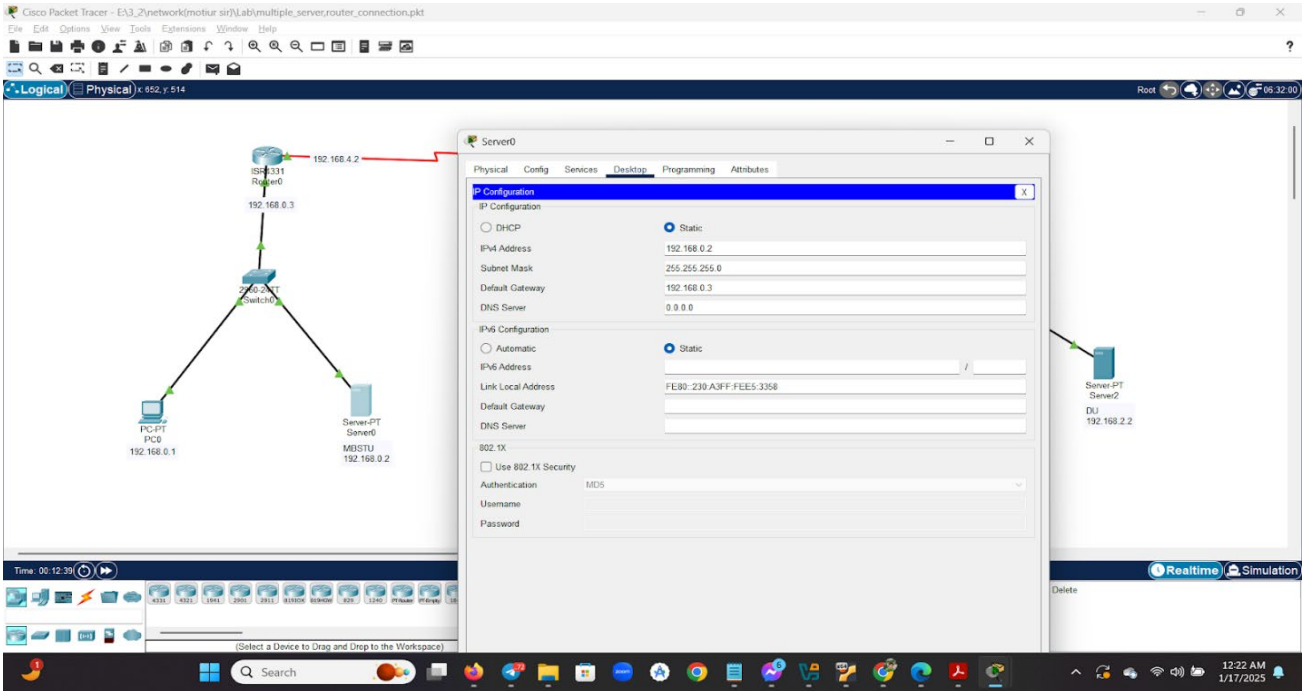


Figure: Set the IP address for all server

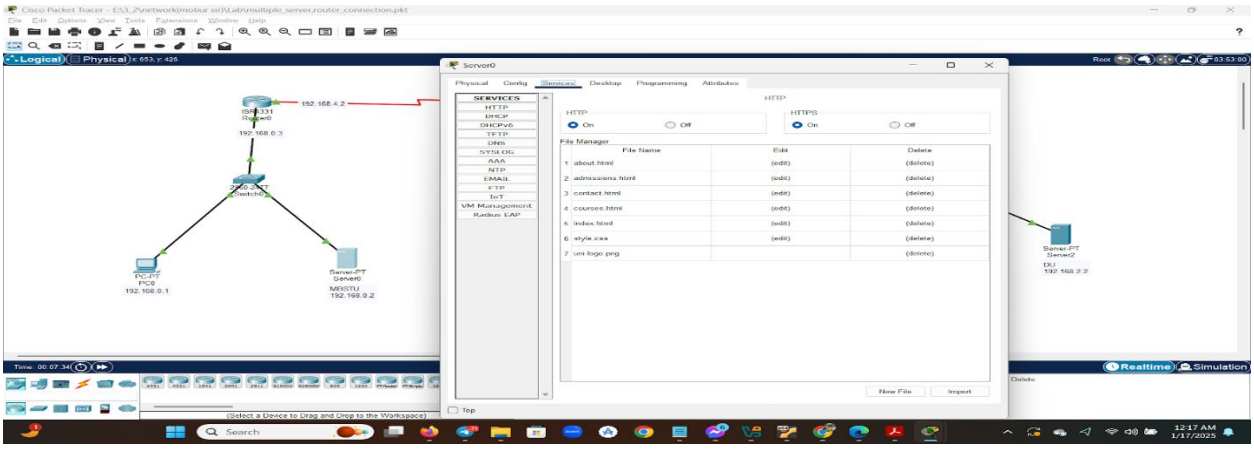
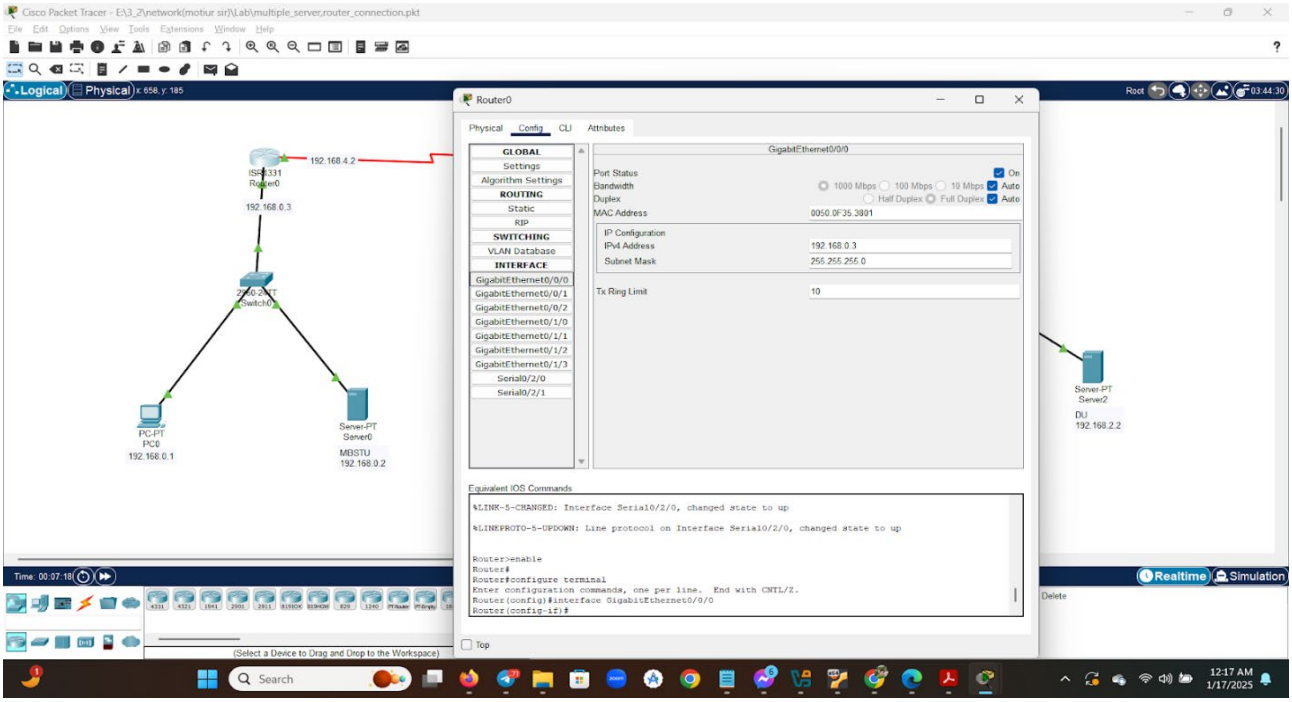
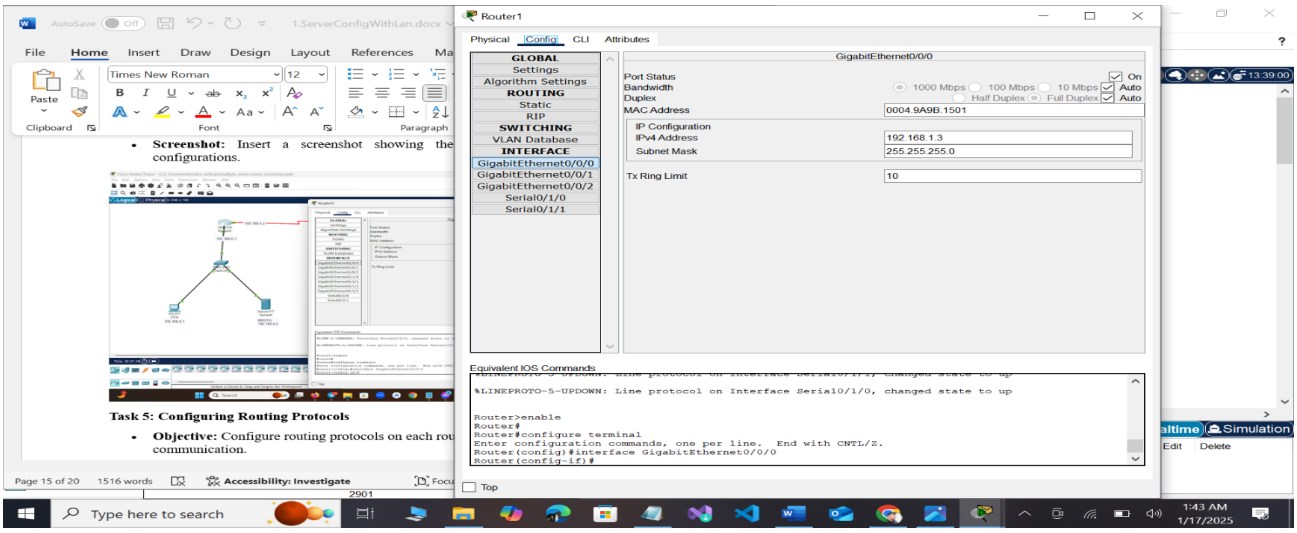


Figure: Edit the HTML code

7. Connecting the Routers

- **Objective:** Connect the routers of the three LANs to enable inter-LAN communication.
- **Steps Performed:**
    1. Connect each LAN's switch to its respective router using a straight-through cable.
    2. Interconnect the routers using cross-over cables to form a network backbone.
    3. Assign IP addresses to router interfaces (e.g., 192.168.0.3 for MBSTU, 192.168.2.3 for DU, and 192.168.1.3 for JU).





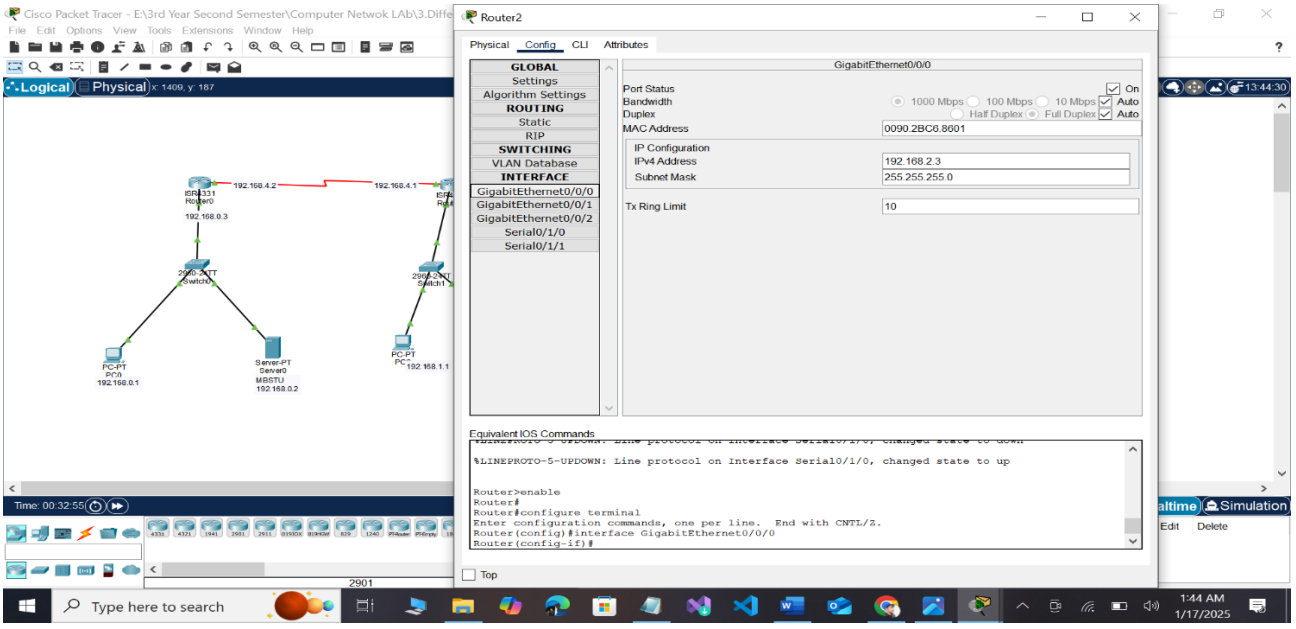


Figure: Configuration of Router

8. Testing Connectivity

- **Objective:** Test communication between devices in different LANs using ping commands.
- **Steps Performed:**
    1. Open the command prompt on a PC in one LAN.
    2. Ping the server or PC in another LAN.
    3. Verify successful communication.

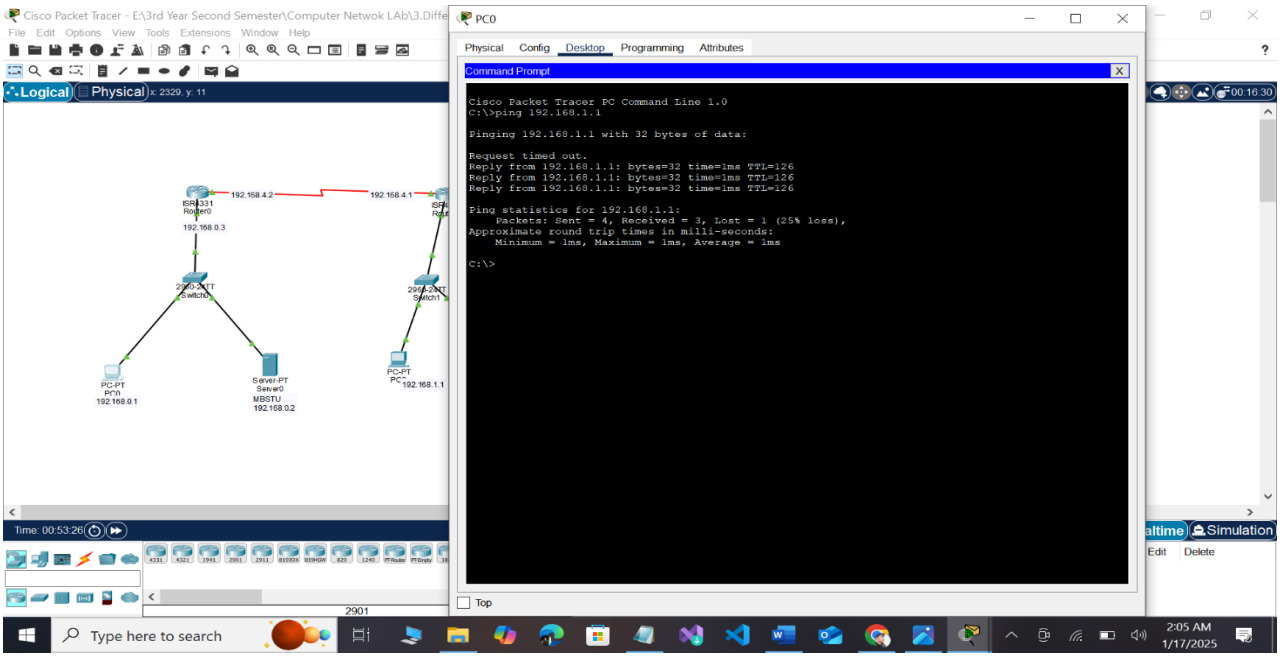


Figure: Pinging pc with between different LAN

9. Observations & Results

- All three LANs (MBSTU, DU, JU) were successfully configured.
- Routers enabled communication between the LANs through proper routing configuration.
- Ping results confirmed inter-LAN connectivity

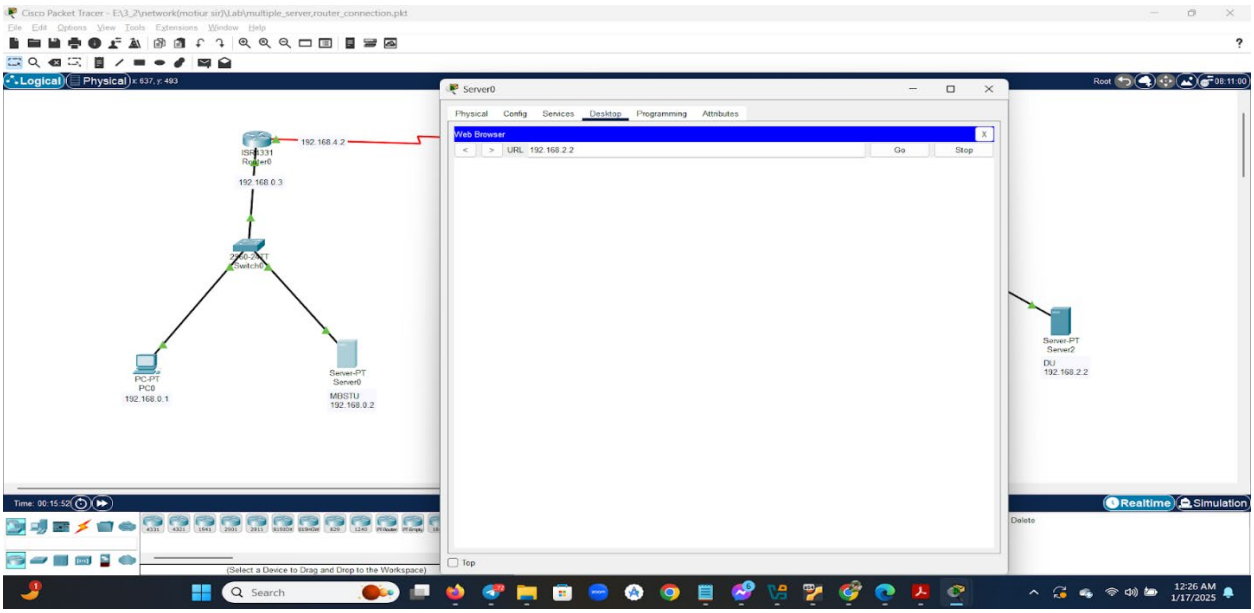


Figure: Open the server web browser and run with IP address of another server

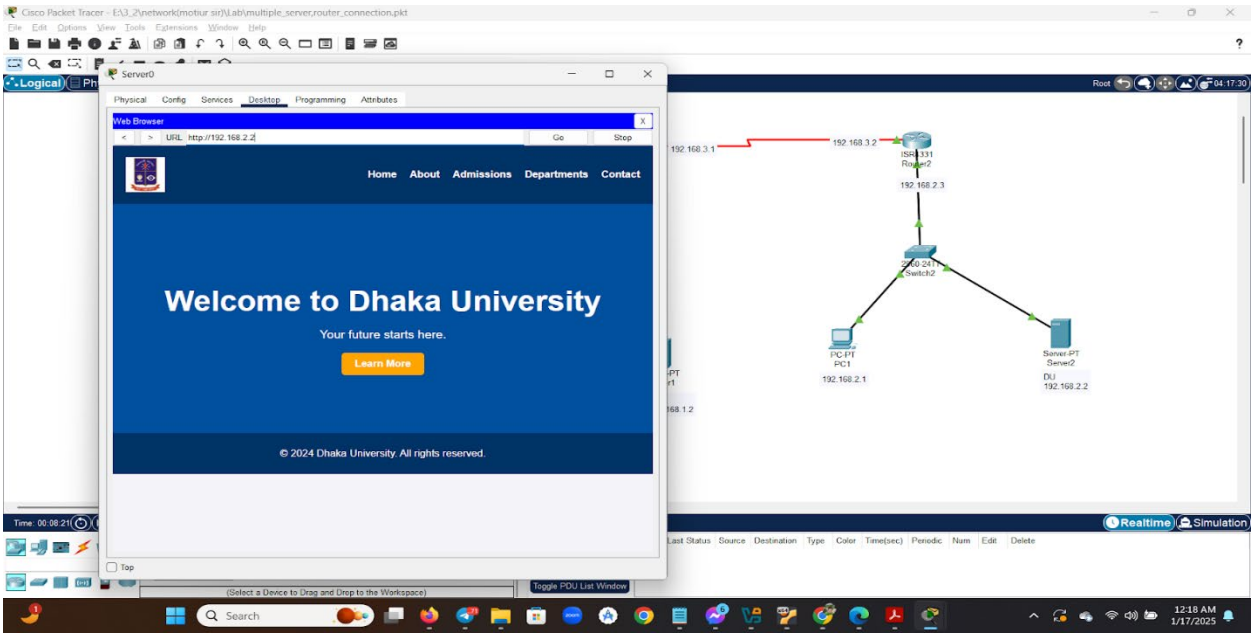


Figure: With IP address the server call another server

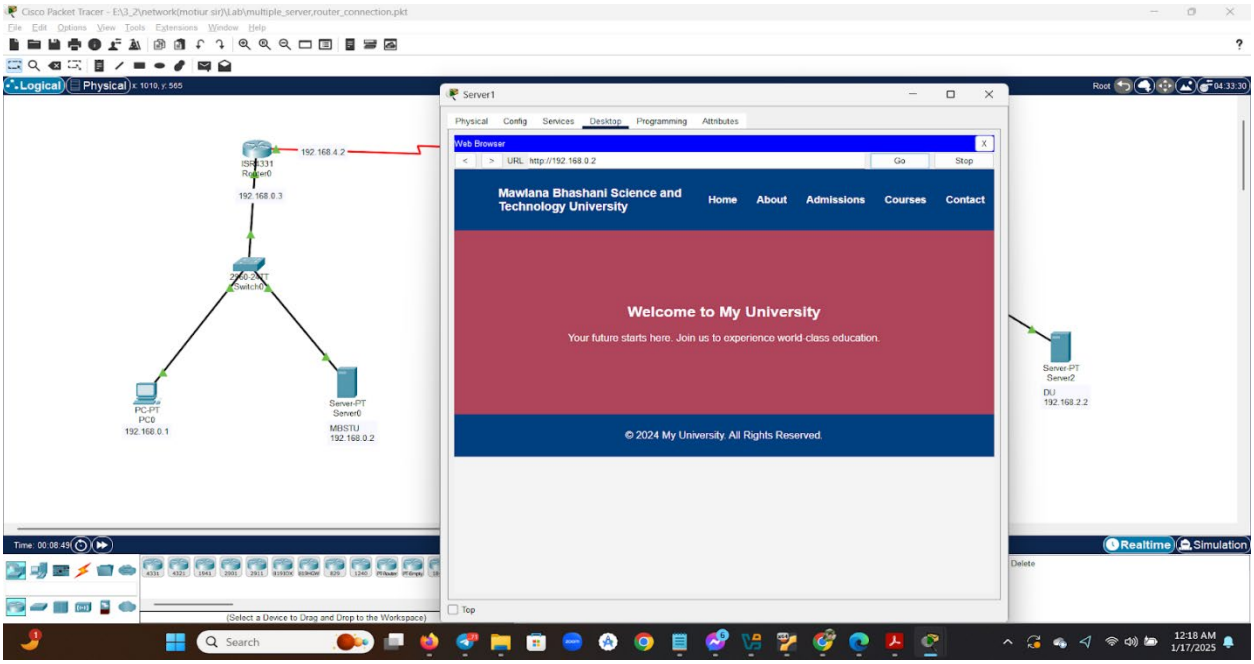


Figure: With IP address the server call another server

10. Conclusion

In this lab, three separate LANs were created and interconnected using routers. The setup demonstrated the implementation of inter-network communication using routing protocols. All objectives were met, and the network operated as expected.

Name of Experiment: Configuring FTP services in a server.

1. Introduction

File Transfer Protocol (FTP) is a standard network protocol used for transferring files between a client and server on a computer network. This lab demonstrates the process of configuring FTP services on a Cisco server using Packet Tracer.

2. Objectives

- Configure a basic network topology with a server and clients
- Set up FTP services on the server
- Create user accounts with appropriate permissions
- Test FTP connectivity and file transfer capabilities
- Verify security measures and access controls

3. Equipment Required

- Cisco Packet Tracer Software
- Server PT
- 2x PC clients
- 1x Switch
- Ethernet cables

4. Diagram

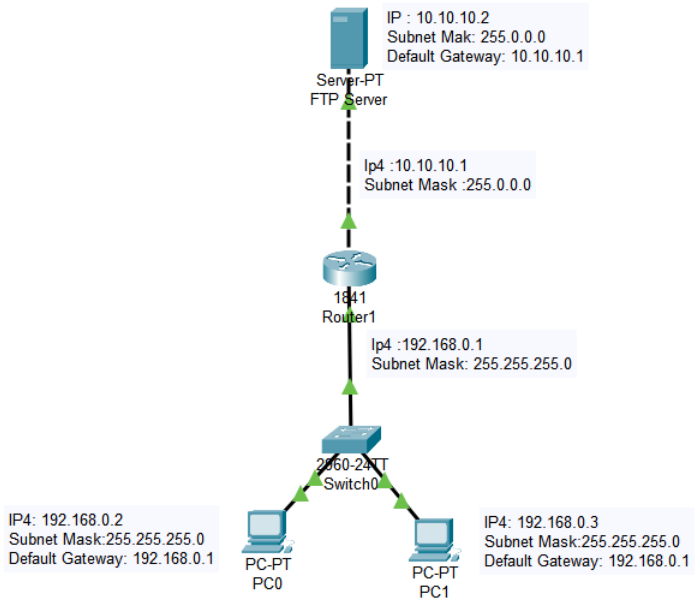


Figure: Configuring FTP services in a server

5. Procedure

i. Set Up the Network

- Drag and drop devices (e.g., PCs, servers, switch/router) into the workspace.
- Connect devices using copper straight-through cables.

ii. Configure the Router

- Assign FastEthernet0/0 IP address and Subnet mask
- Assign FastEthernet0/1 IP address and Subnet mask

iii. Configure the FTP Server

- Click on the server.
- Go to the Services tab and enable the FTP service.
  - Assign IP address, Default Gateway and Subnet mask.
- Add user accounts with appropriate credentials.
  - Set Username and password in Services tab.

iv. Assign IP Addresses

- Open each device's configuration tab and assign static IPs based on your network design.
  - Assign IP address, Default Gateway and Subnet mask.

v. Test Connectivity

- Use the Command Prompt on PCs to ping the server to ensure network connectivity.

vi. Create Files

- Create a text file using Text Editor option on one of the Device (PC0)

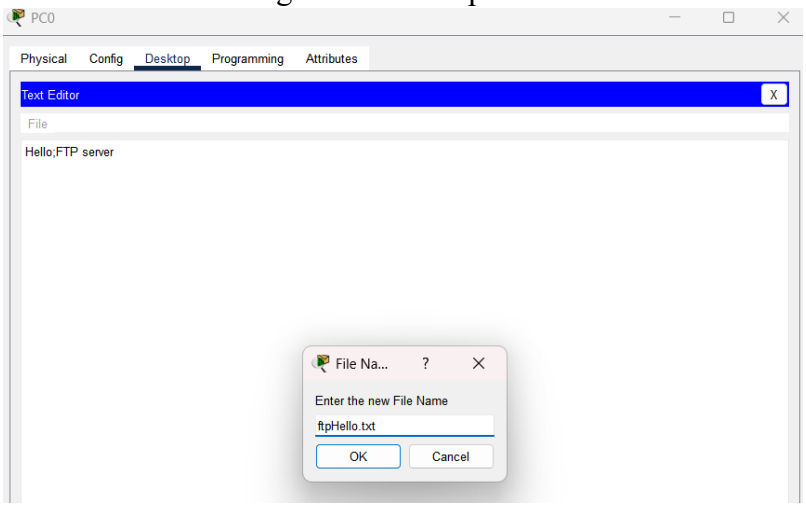


Figure: create a ftpHello.txt file

vii. Upload file on Server

- On a PC, use the Command Prompt and type: ftp [server IP address].
- Log in with the configured credentials.
- Use FTP commands put to upload files.

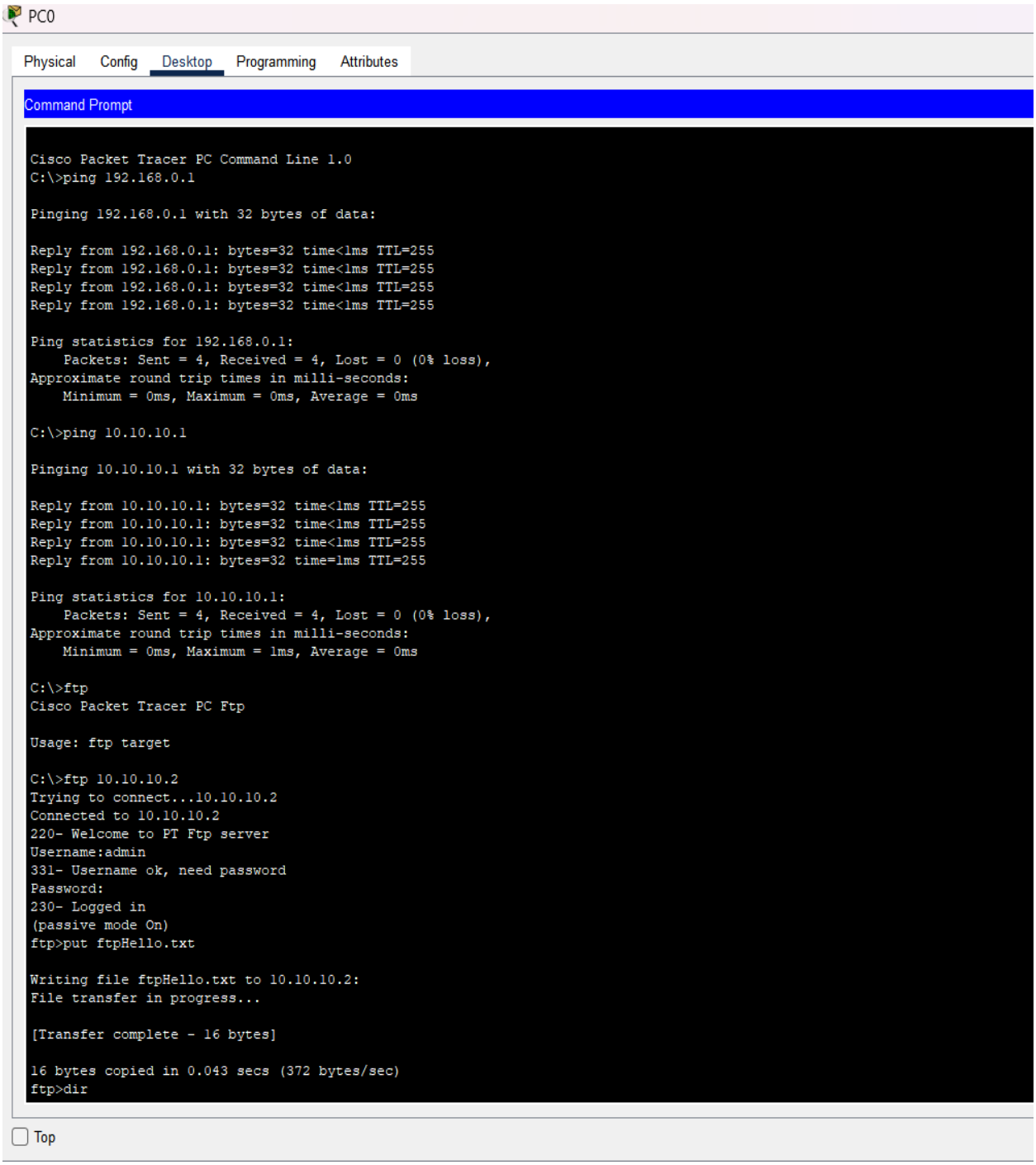


Figure: Upload file on Server

viii. Access Files

- On a PC, use the Command Prompt and type: ftp [server IP address].
- Log in with the configured credentials.
- Use FTP commands get to download files.

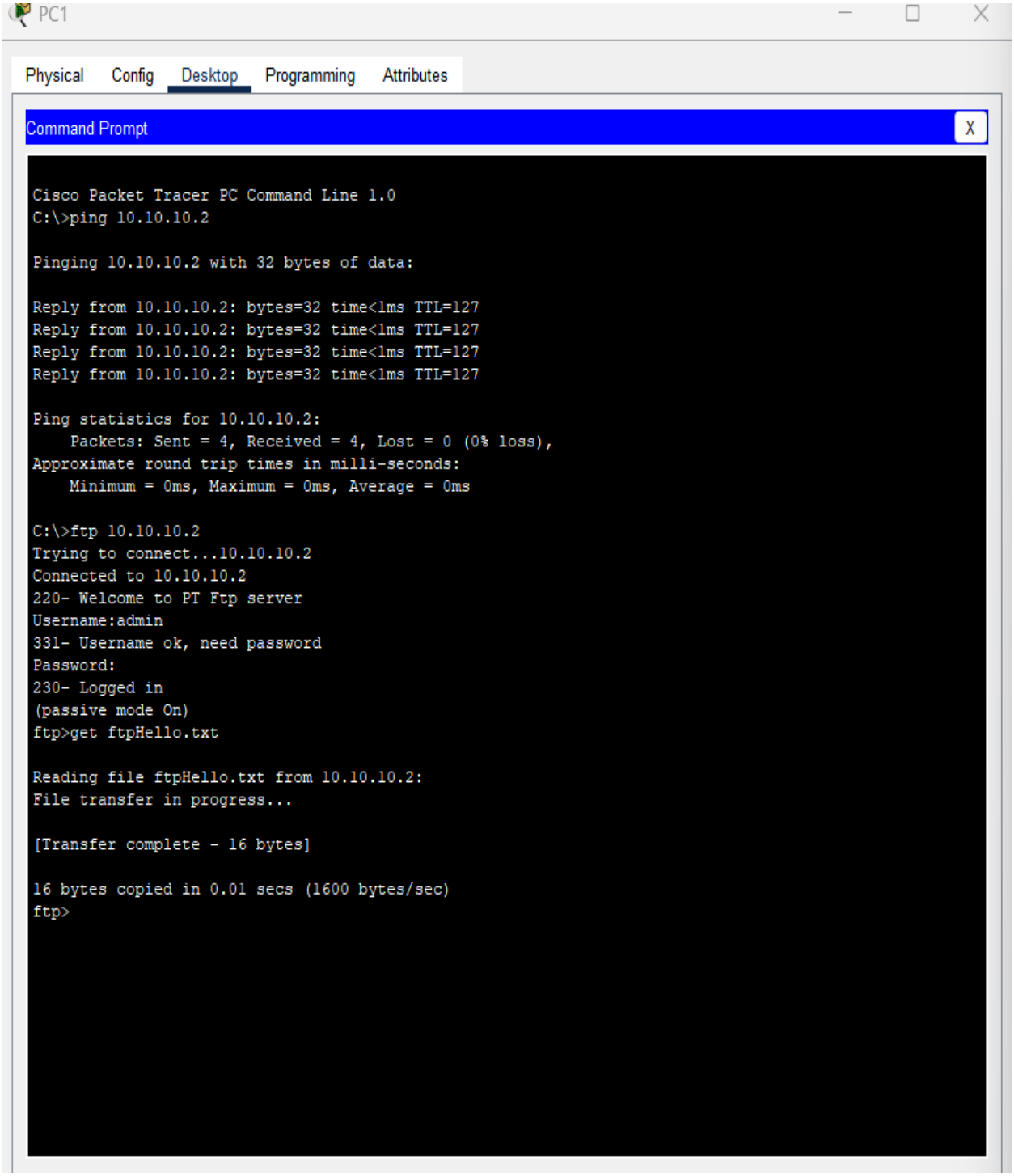


Figure: Access file on Server

6. Challenges Faced and Resolutions

Common issues and solutions:

- Connection timeout
  - Verify IP configuration
  - Check network connectivity
  - Confirm FTP service is running
- Authentication failures
  - Verify username and password
  - Check permission settings
  - Confirm user account status

7. Result Discussion

The configuration of FTP services on the server was successfully completed, and file transfers between the server and clients were verified. The network demonstrated how FTP allows users to upload and download files seamlessly, highlighting its role in managing and sharing data over a network**.**

8. Conclusion

This lab successfully demonstrated the configuration of FTP services on a Cisco server using Packet Tracer. The implementation provided secure file transfer capabilities while maintaining appropriate access controls.

Name of Experiment: Configuring SMTP/E-Mail services in a server.

1. Introduction

SMTP stands for Simple Mail Transfer Protocol—it's an application used by mail servers to send, receive, and relay outgoing email between senders and receivers. It's also known as an outgoing mail server.

2. Objectives

- Set up a basic email server architecture.
- Configure SMTP services on the server.
- Create email user accounts and mailboxes.
- Test email delivery functionality.
- Implement basic security measures.
- Verify email routing and delivery.

3. Equipment Required

- Cisco Packet Tracer Software
- Server PT
- 2x PC clients
- 1x Switch
- Ethernet cables

4. Diagram

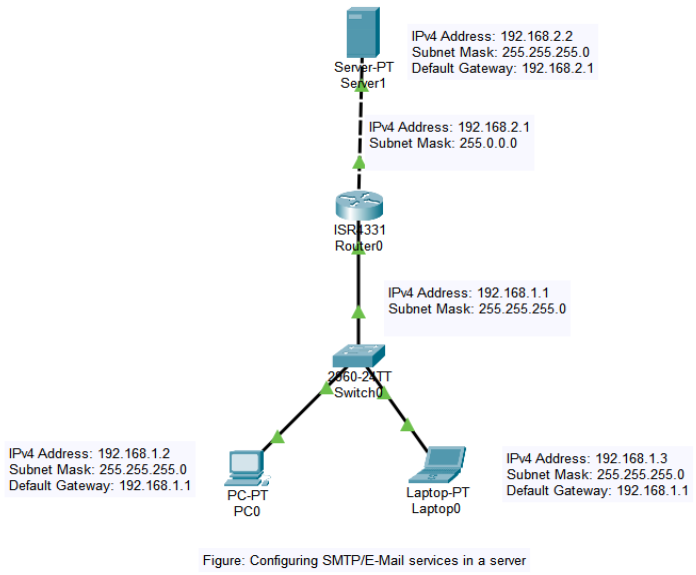


Figure: Configuring SMTP/E-Mail services

5. Procedure

i. Set Up the Network

- Drag and drop devices (e.g., PCs, servers, switch/router) into the workspace.
- Connect devices using copper straight-through cables.

ii. Configure the Router

- Assign GigaBitEthernet0/0/0 IP address and Subnet mask
- Assign GigaBitEthernet0/0/1 IP address and Subnet mask

iii. Configure the Server

- Click on the server.
- Go to the Services tab and select Email.
- Enable the SMTP and POP3/IMAP services.
- Add user accounts with unique usernames and passwords.

iv. Assign IP Addresses

- Open each device's configuration tab and assign static IPs based on your network design.
  - Assign IP address, Default Gateway and Subnet mask.

v. Configure the Client Email Settings

- On each PC, open the Email client application.
- Configure the email account.
  - Incoming mail server: [Server IP address] (POP3/IMAP).
  - Outgoing mail server: [Server IP address] (SMTP).
  - Enter username and password created on the server.

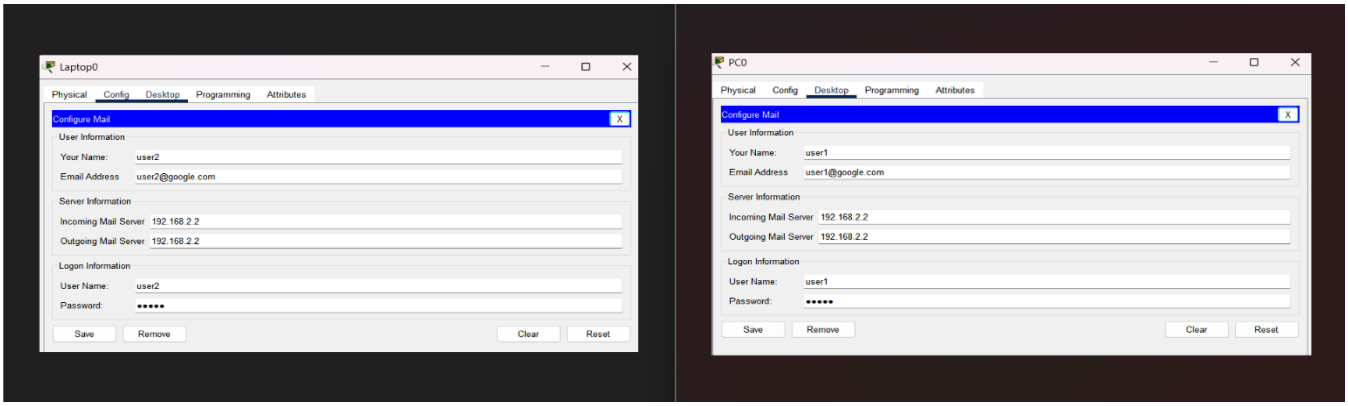


Figure: configure Email

vi. Test Connectivity

- Use the Command Prompt on PCs to ping the server to ensure network connectivity.

vii. Test Email Communication

- Use one client to send an email to another client.

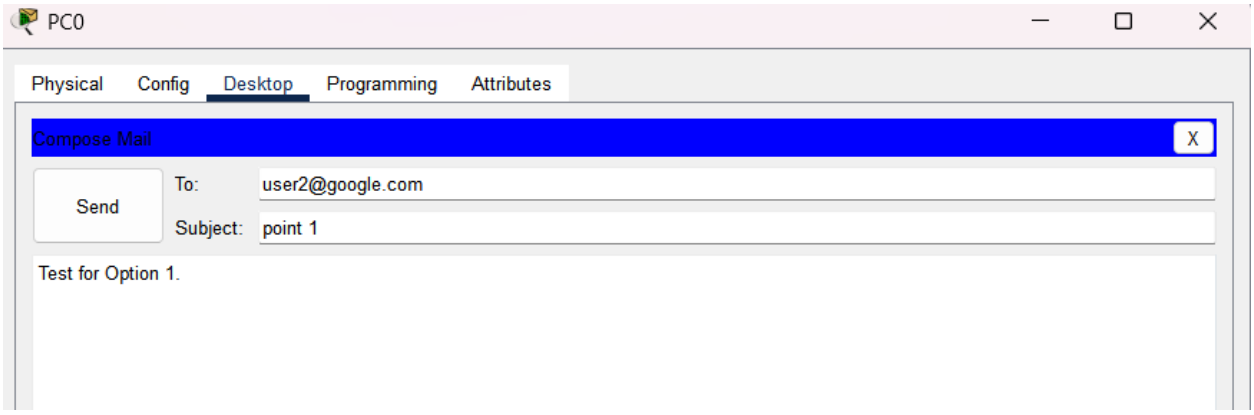


Figure: Compose Email

- Verify the email is received on the recipient’s client.

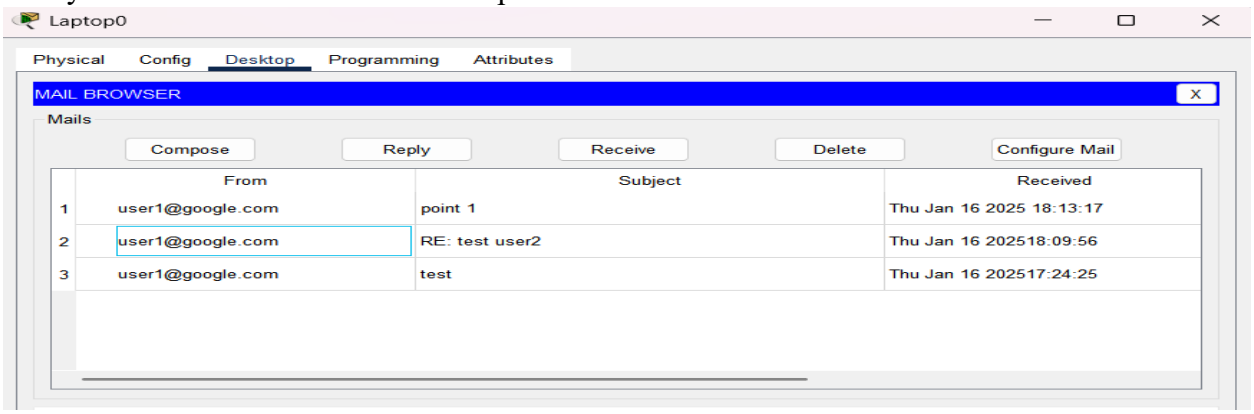


Figure: Verify received on the recipient's

6. Result Discussion

The configuration of SMTP and E-Mail services on the server was successfully completed. The network topology was functional, and the email communication between clients was validated through successful email transfers.

7. Challenges Faced and Resolutions

The configuration of SMTP and E-Mail services on the server was successfully completed. The network topology was functional, and the email communication between clients was validated through successful email transfers.

Challenges Faced and Resolutions

- Connection Problems:
  - Verify network connectivity
  - Check firewall settings
  - Confirm DNS configuration
- Authentication Issues:
  - Verify user credentials
  - Confirm permissions settings
- Mail Delivery Delays:
  - Check mail queue
  - Monitor server resources

8. Conclusion

The successful configuration of SMTP services enables reliable email communication within the network environment. The implementation includes necessary security measures and demonstrates proper email routing functionality.

Name of Experiment: Configuring multiple servers with HTTP services (e.g. making their respective websites) and connecting them to their respective LANs, creating a WAN by connecting all the LANs and sharing their services with each other.

1. Introduction

This lab involves configuring HTTP servers on separate LANs, connecting them via a WAN, and enabling service sharing. It demonstrates practical networking concepts, including LAN/WAN setup, routing, and HTTP service management, simulating real-world distributed systems.

2. Objectives

- Configure HTTP services on multiple servers.
- Establish LANs for each server and connect clients.
- Create a WAN to link all LANs.
- Configure routing for inter-LAN communication.
- Implement web services on each server.
- Test connectivity and service sharing across networks.

3. Equipment Required

- Cisco Packet Tracer Software
- For Each LAN (×2):
    - 1× Cisco Router (2911)
    - 1× Layer 2 Switch (2960)
    - 1× Web Server
    - 3× Client PCs
    - Ethernet cables
- For Wan Connections
    - Serial cables for router interconnection
    - WAN interface modules for routers

4. Diagram

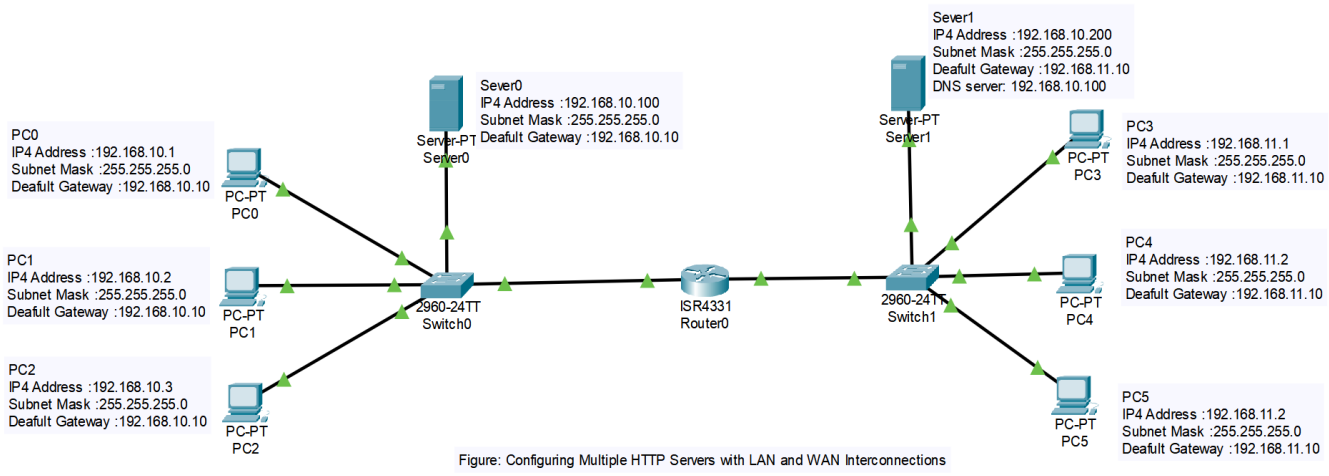


Figure: Configuring Multiple HTTP Servers with LAN and WAN Interconnections

5. Procedure

i. Set Up the Network

- Drag and drop devices (e.g., PCs, servers, switch/router) into the workspace.
- Connect devices using copper straight-through cables.
- Create separate LANs for each server with its clients

ii. Configure HTTP Services on Servers

On each Server

- Click on the server.
- Go to the Services tab and enable HTTP.
- Create a unique website for each server by modifying the Index.html file in the HTTP service.

iii. Assign IP Addresses

- Open each device's configuration tab and assign static IPs based on your network design.
  - Assign IP address, Default Gateway and Subnet mask also DNS server address.

iv. Connect LANs to WAN

- Use routers to connect the LANs.
- Configure serial interfaces on routers for WAN connections.

v. Test Connectivity

- Use the Command Prompt on PCs to ping the server to ensure network connectivity.
- Test WAN connectivity by pinging between devices in different LANs.

vi. Access HTTP Services

- From a client PC in one LAN, open a web browser and access the websites hosted on servers in other LANs using their respective IP addresses.

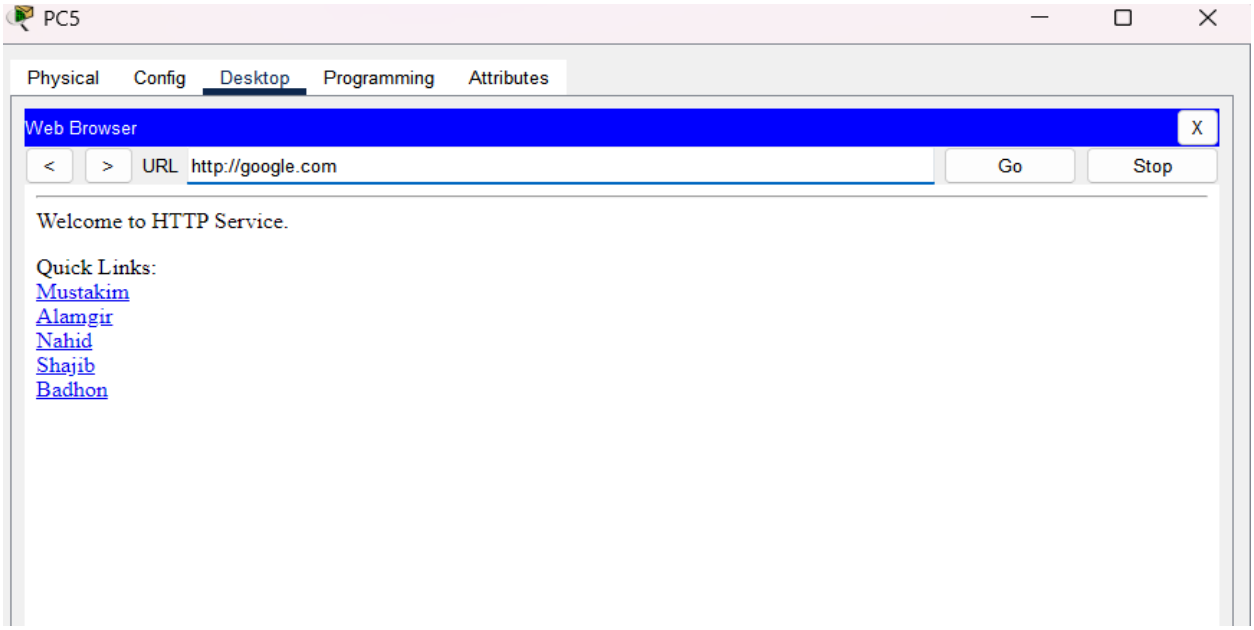


Figure: Access HTTP services

6.Result Discussion

- The configuration of HTTP servers on separate LANs and their interconnection through a WAN was successfully completed.
- Each server hosted a unique website, which was accessible from devices across the WAN.

7. Challenges Faced and Resolutions

- Routing Misconfigurations: Resolved by correctly setting up routing protocols like RIP or OSPF.
- Connectivity Issues: Fixed by verifying IP addresses and network connections.

8. Conclusion

The configuration demonstrates successful implementation of multiple HTTP servers across interconnected LANs, enabling organization-wide access to web services while maintaining network segregation and security.

Name of Experiment: DNS Packet Analysis in Cisco Packet Tracer

Objective: To analyze DNS packets using Cisco Packet Tracer and understand how DNS queries and responses are structured and transmitted in a simulated network environment.

Materials Required

- Cisco Packet Tracer software
- Network topology including:
  - End devices (PCs)
  - DNS Server
  - Router/Switch
  - Internet cloud (simulated)
- Configured DNS settings on Packet Tracer

Theory/Background: The Domain Name System (DNS) is responsible for translating human-readable domain names into IP addresses. When a user requests a website, a DNS query is sent to a DNS server, which then responds with the corresponding IP address. Cisco Packet Tracer allows for the simulation of DNS resolution by setting up a DNS server and observing packet exchanges between devices.

Working Procedure:

1. Open Cisco Packet Tracer and create a new network topology.
2. Add a DNS server, router, switch, and end devices (PCs).

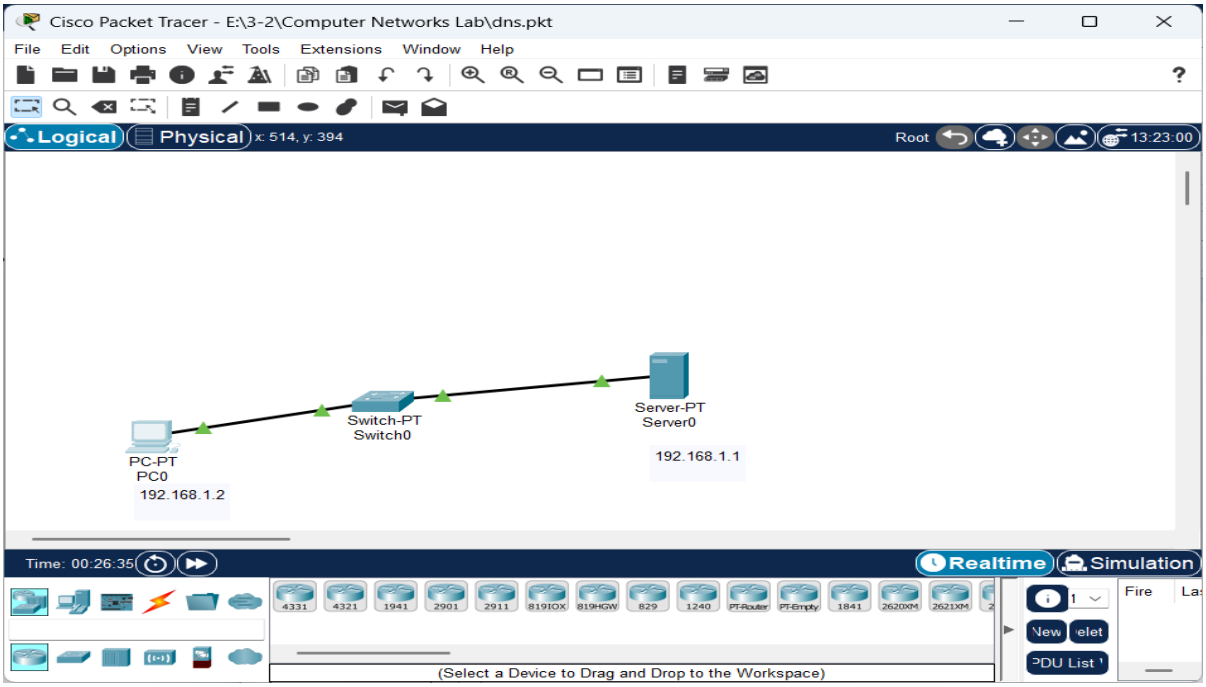


Figure: DNS server configuration

3. Configure the DNS server with domain names and corresponding IP addresses.
4. Assign appropriate IP addresses and subnet masks to all devices.
5. Configure the end devices to use the DNS server's IP address.

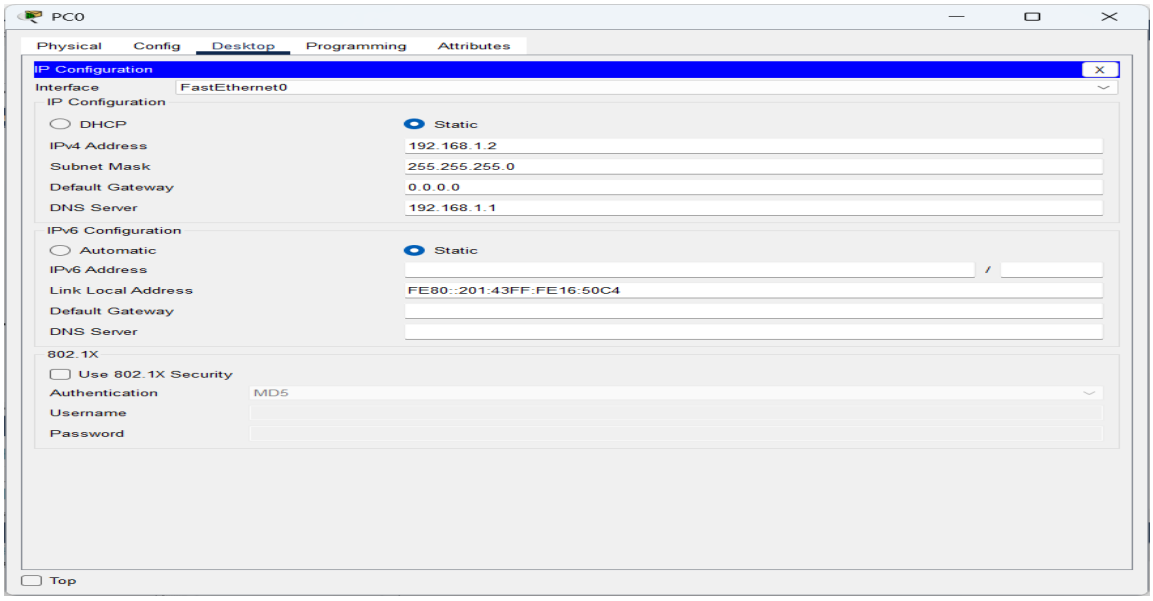


Figure: PC 0 IP configuration

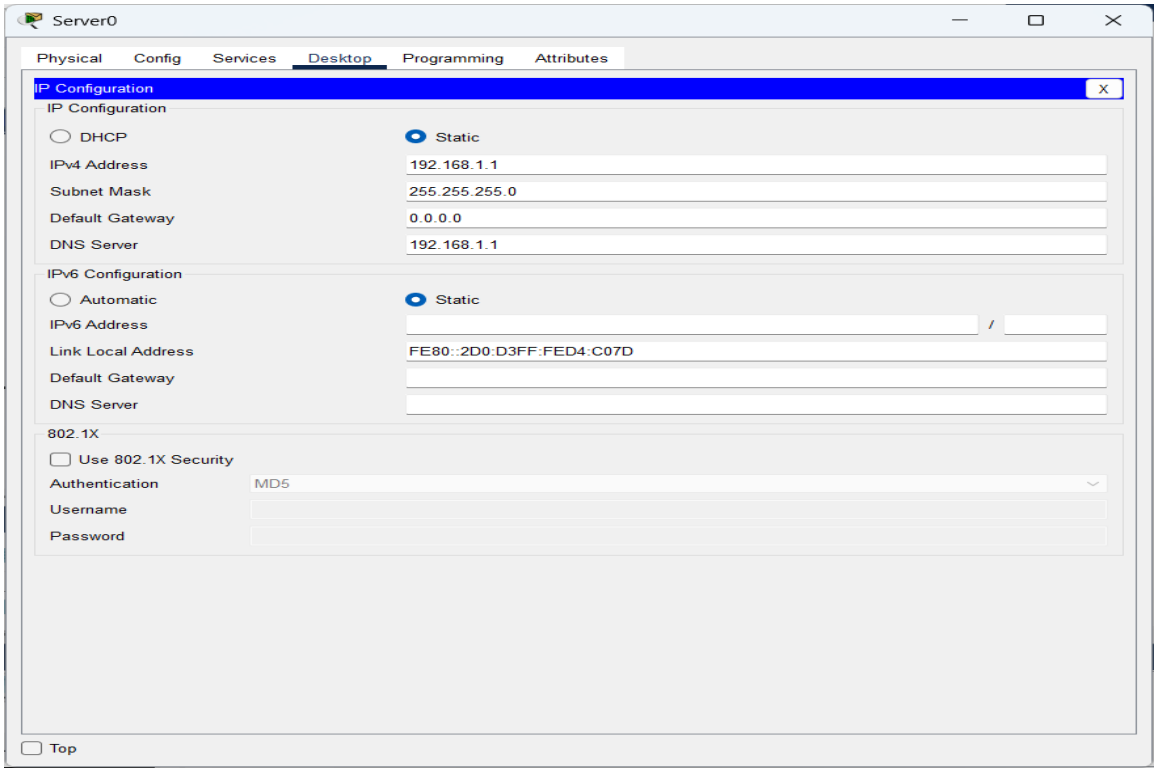


Figure: DNS server IP configuration

6. Open the command prompt on a PC and use the `nslookup` command to query domain names.

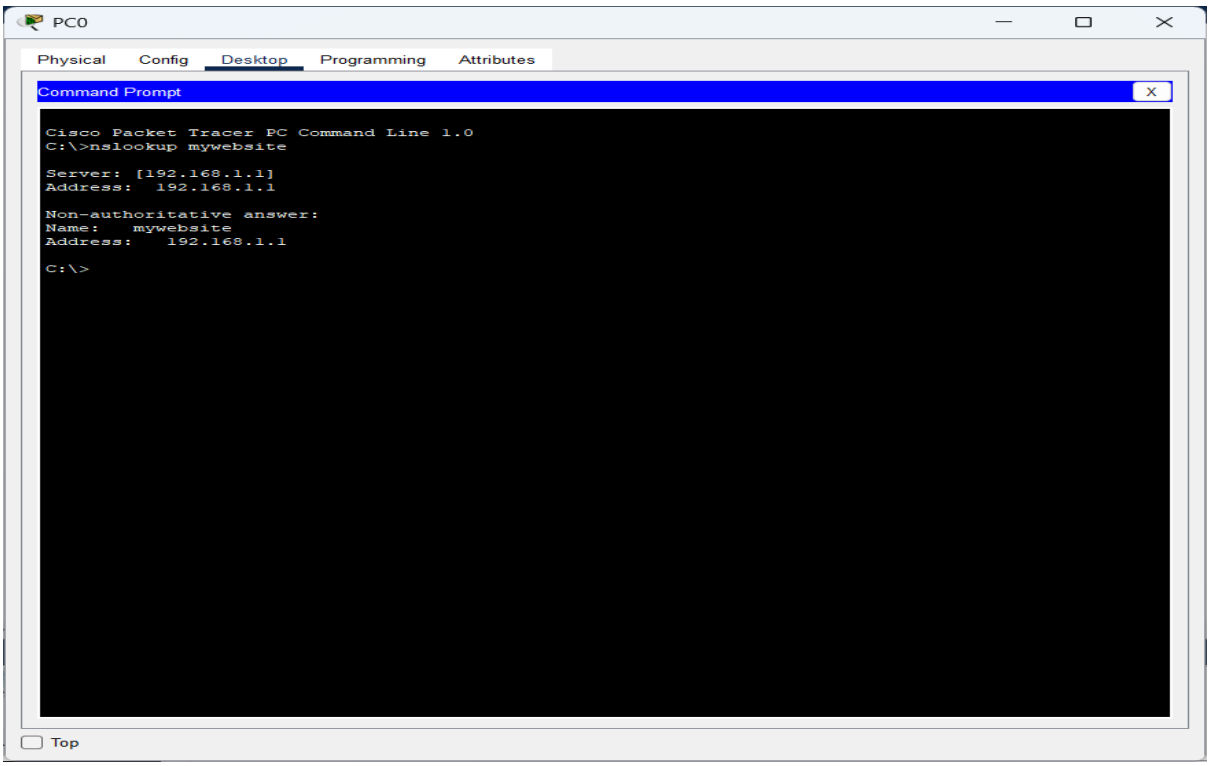


Figure: nslookup Query

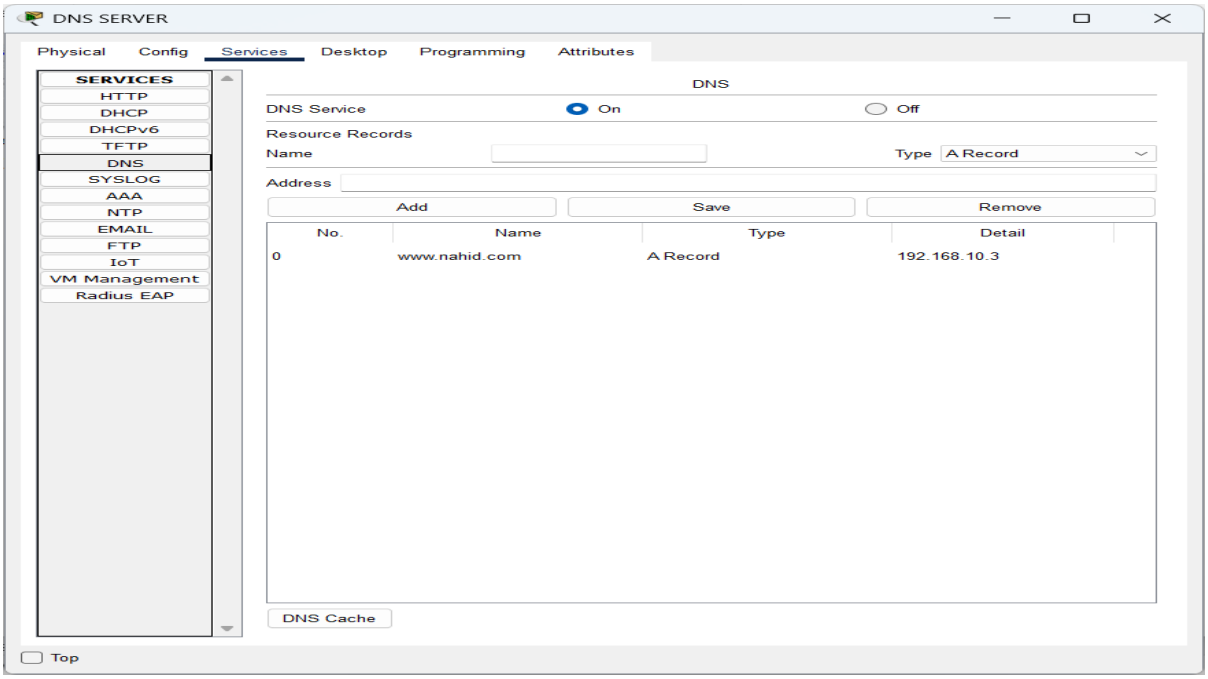


Figure: DNS service configuration

7. Monitor packet flow using the simulation mode in Packet Tracer.
8. Capture and analyse DNS request and response packets.

Observations:

- DNS queries are sent using UDP on port 53.
- The DNS server successfully resolves domain names to IP addresses.
- Simulated network traffic follows the expected path through switches and routers.
- Configured caching allows for faster subsequent queries.

Result Analysis:

- The analysis showed that DNS queries and responses follow the expected communication model.
- Packet Tracer's simulation mode effectively visualized the request-response process.
- Response times depend on network latency and server availability in the simulation.

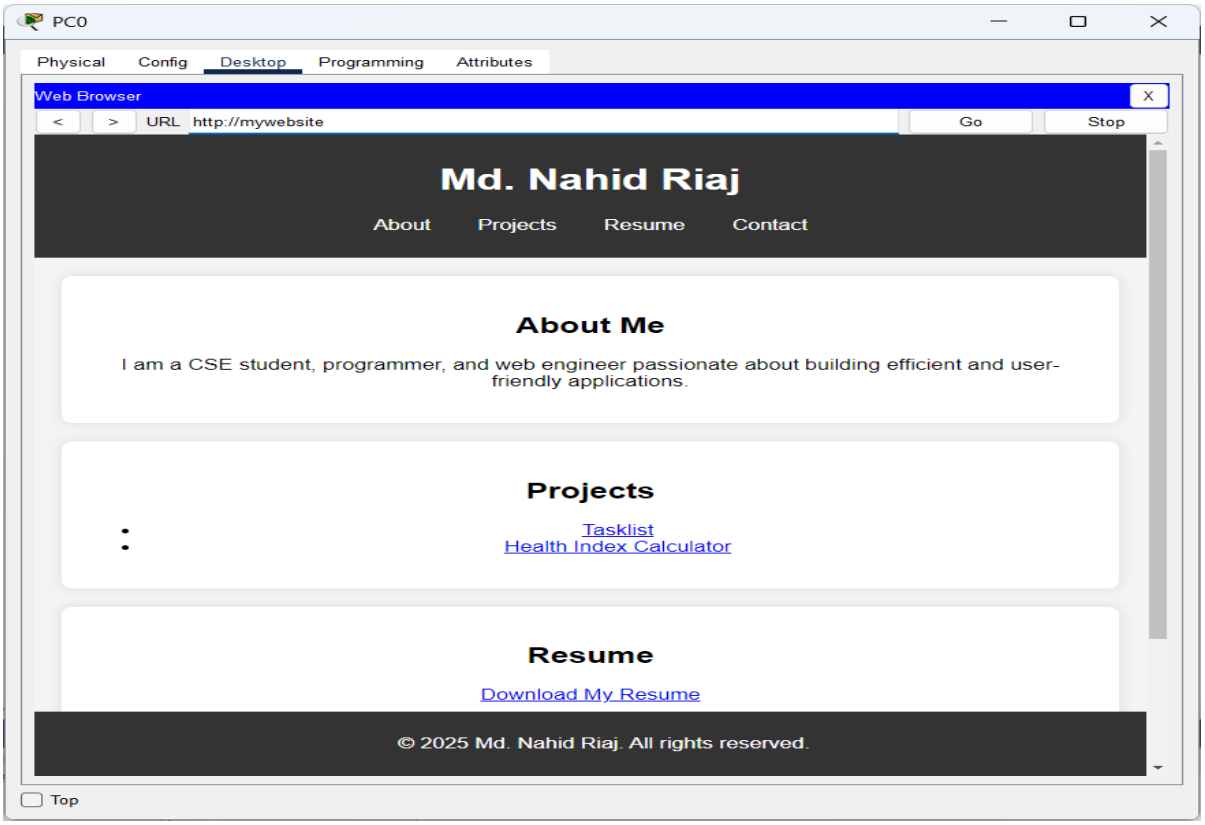


Figure: Using DNS server IP 192.168.10.3 set to website

Conclusion: The experiment successfully demonstrated DNS packet structures and response behaviours within Cisco Packet Tracer. This simulation helped in understanding real-world DNS operations and network configurations.

Name of Experiment: DHCP Packet Analysis in Cisco Packet Tracer

Objective: To analyze DHCP packets using Cisco Packet Tracer and understand how the DHCP lease process assigns IP addresses dynamically to network devices.

Materials Required:

- Cisco Packet Tracer software
- Network topology including:
  - End devices (PCs)
  - DHCP Server
  - Router/Switch
  - Internet cloud (simulated)
- Configured DHCP settings on Packet Tracer

Theory/Background: Dynamic Host Configuration Protocol (DHCP) automates the assignment of IP addresses, subnet masks, gateways, and DNS configurations to network devices. The DHCP lease process follows the DORA sequence: Discover, Offer, Request, and Acknowledge. Cisco Packet Tracer enables the simulation of DHCP operations and provides visualization of packet exchanges.

Working Procedure:

1. Open Cisco Packet Tracer and create a new network topology.
2. Add a DHCP server, router, switch, and end devices (PCs).

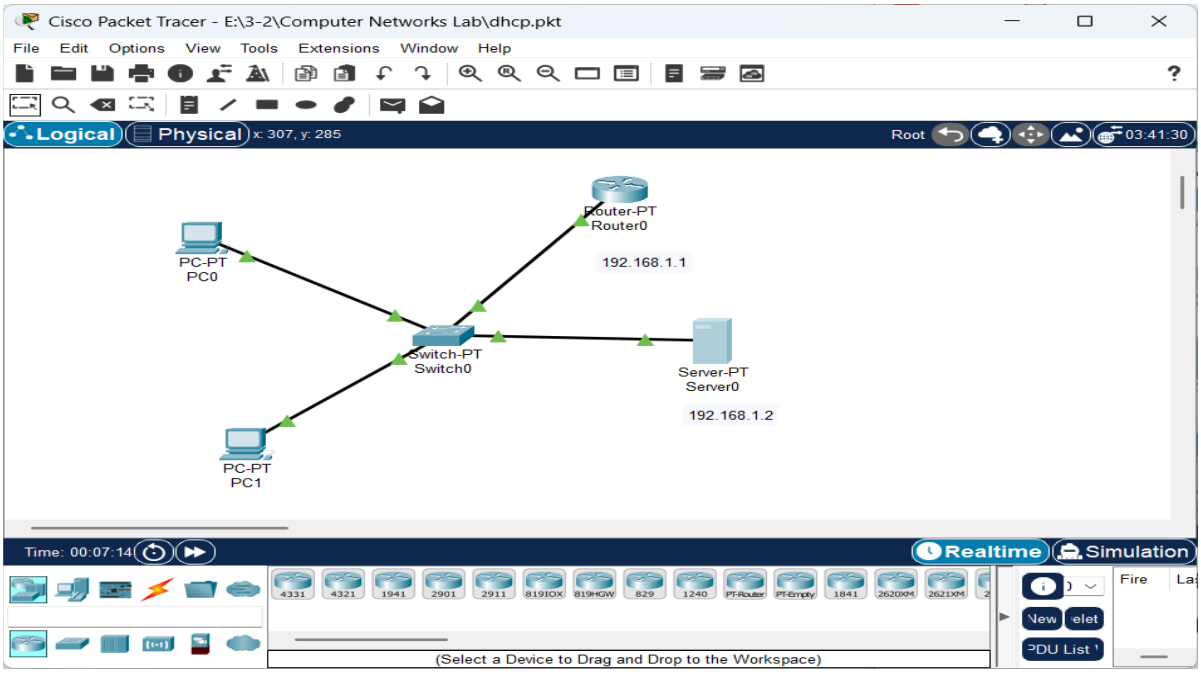


Figure: DHCP network configuration

3. Configure the DHCP server to assign dynamic IP addresses, subnet masks, gateways, and DNS details.

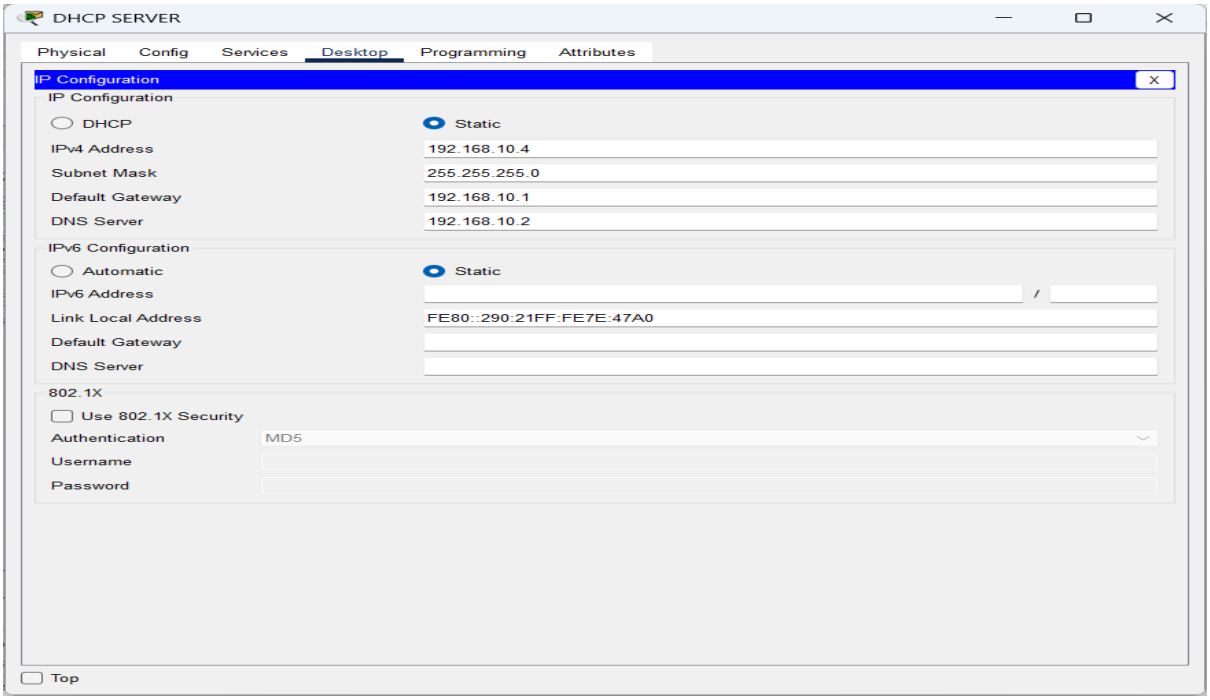


Figure: DHCP server IP configuration

4. Assign static IP addresses to the router and servers.

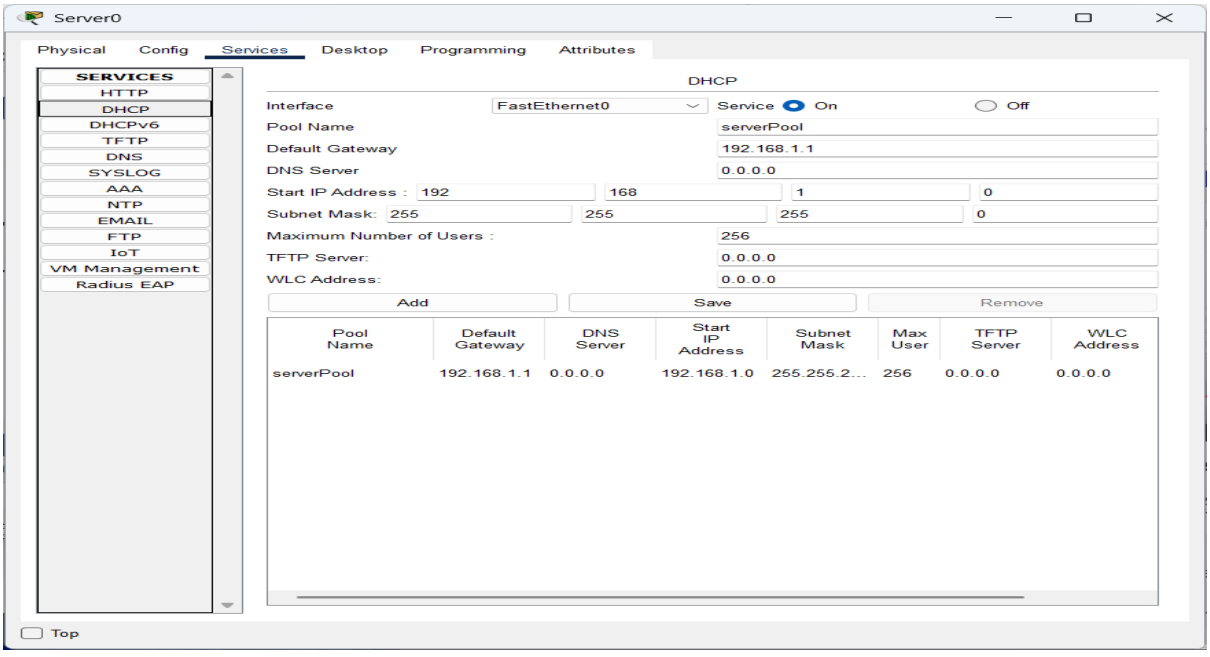


Figure: Assign static IP to router and server

5. Configure end devices to obtain IP addresses dynamically from the DHCP server.
6. Open the command prompt on a PC and use the `ipconfig` command to verify the assigned IP configuration.

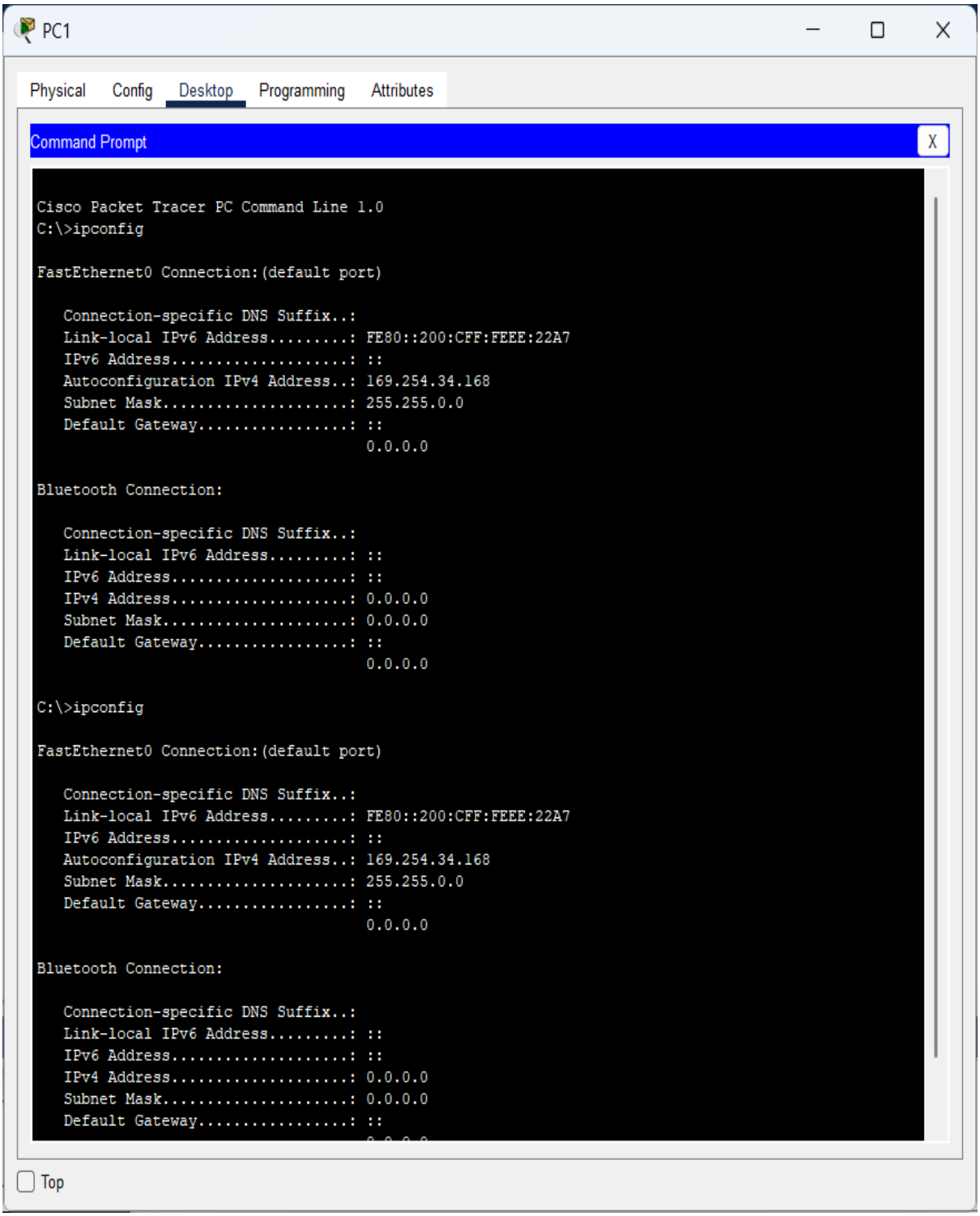


Figure: Show IP for PC1

7. Monitor packet flow using the simulation mode in Packet Tracer.
8. Capture and analyze DHCP lease packets.

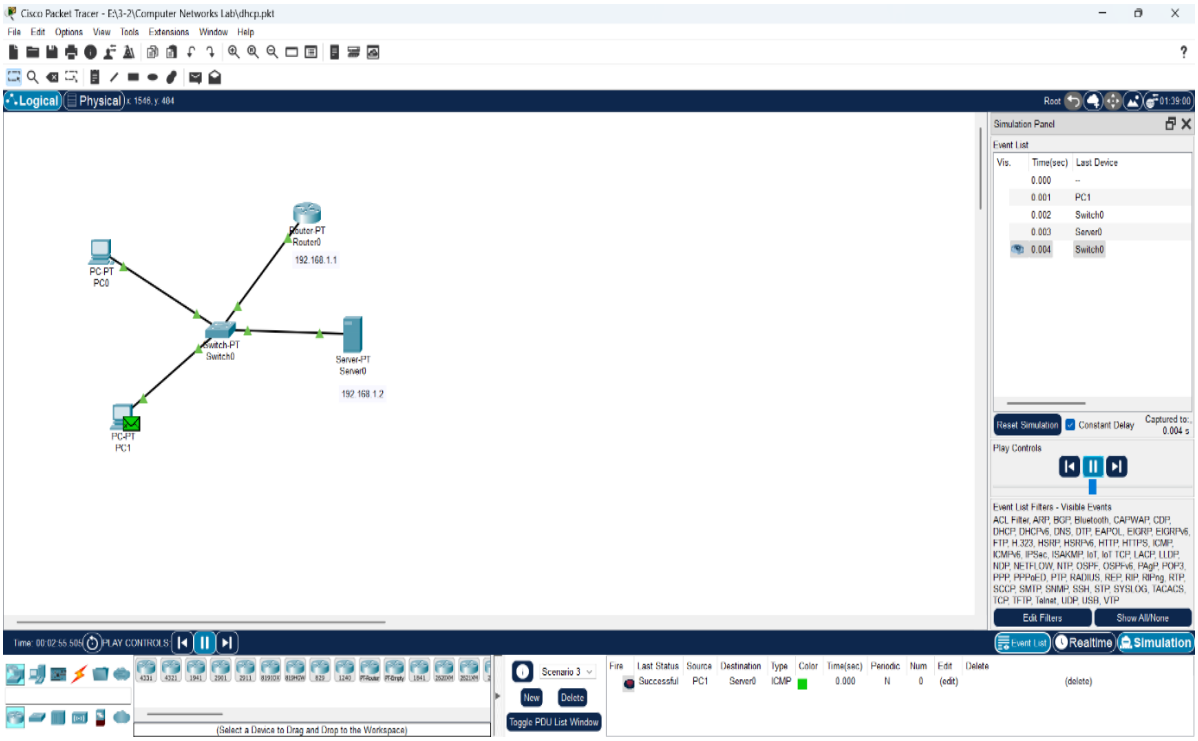


Figure: How DHCP work

Observations:

- DHCP lease process follows the DORA sequence:
  - **Discover:** Client broadcasts a request for an IP address.
  - **Offer:** DHCP server responds with an available IP address.
  - **Request:** Client requests the offered IP address.
  - **Acknowledge:** DHCP server confirms the assignment.
- DHCP packets use UDP ports 67 (server) and 68 (client).
- Devices successfully obtain IP addresses dynamically.
- Lease expiration and renewal processes work as expected.

Result Analysis:

- The analysis confirmed the expected behaviour of DHCP communication.
- Packet Tracer's simulation mode effectively visualized DHCP packet exchanges.
- Automatic IP configuration reduces administrative workload and avoids IP conflicts.

Conclusion: The experiment successfully demonstrated DHCP packet structures and response behaviours within Cisco Packet Tracer. This simulation provided a clear understanding of the DHCP lease process and its importance in network management.

Name of Experiment: FTP, HTTP, DHCP, and DNS Packet Analysis in Cisco Packet Tracer

Objective: To analyze FTP, HTTP, DHCP, and DNS packets using Cisco Packet Tracer and understand how these network services communicate and transmit data in a simulated environment.

Materials Required:

- Cisco Packet Tracer software
- Network topology including:
  - End devices (PCs)
  - FTP Server
  - HTTP Server
  - DHCP Server
  - DNS Server
  - Router/Switch
- Configured FTP, HTTP, DHCP, and DNS settings on Packet Tracer

Theory/Background:

1. **File Transfer Protocol (FTP):** Used for transferring files between a client and a server over a network.
2. **Hypertext Transfer Protocol (HTTP):** Used for requesting and transferring web pages.
3. **Dynamic Host Configuration Protocol (DHCP):** Automatically assigns IP addresses to network devices.
4. **Domain Name System (DNS):** Translates domain names into IP addresses to facilitate web browsing.

Working Procedure:

1. Open Cisco Packet Tracer and create a new network topology.
2. Add and configure:
   - FTP Server for file transfer.
   - HTTP Server for web page hosting.
   - DHCP Server for automatic IP allocation.
   - DNS Server for domain name resolution.

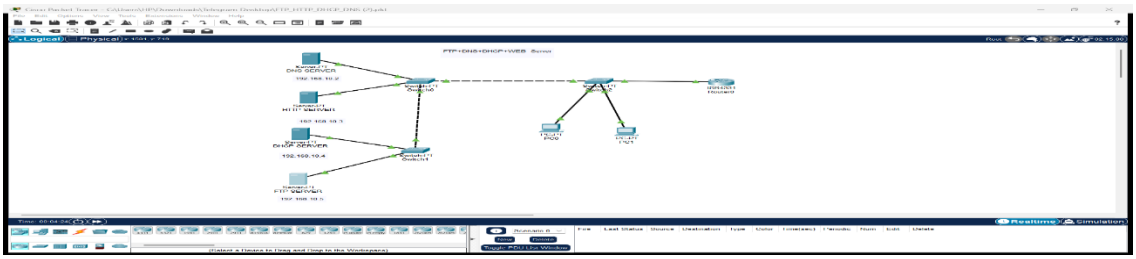


Figure: Configuration of FTP, HTTP, DHCP, and DNS server

3. Configure the router and switches for network communication.

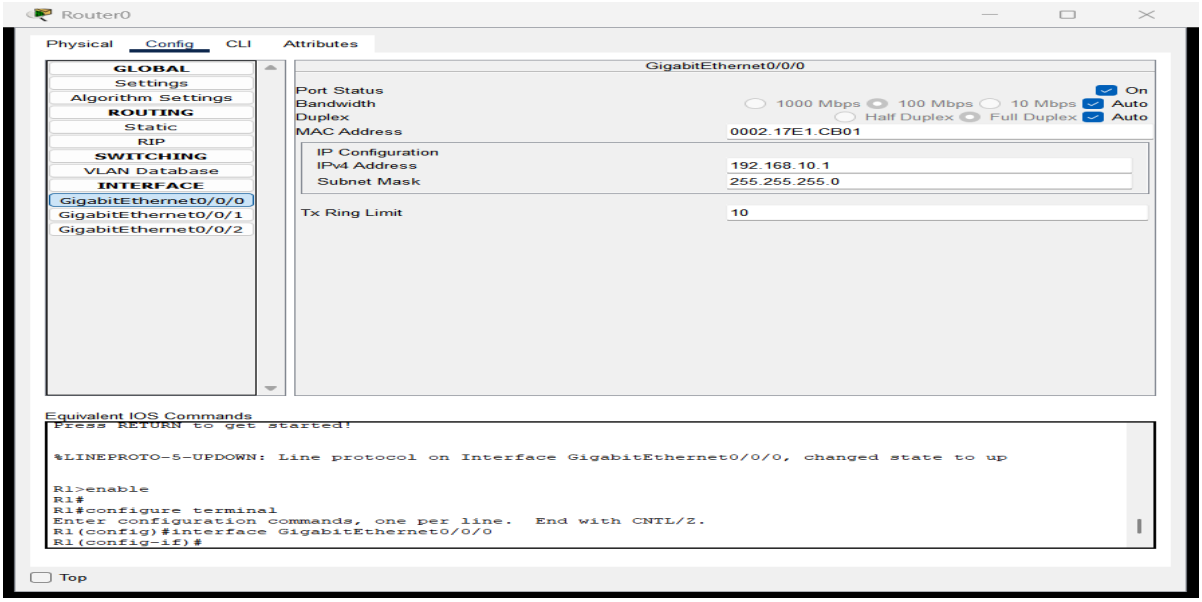


Figure: Configuration of Router

4. Assign static IP addresses to servers and enable DHCP for clients.

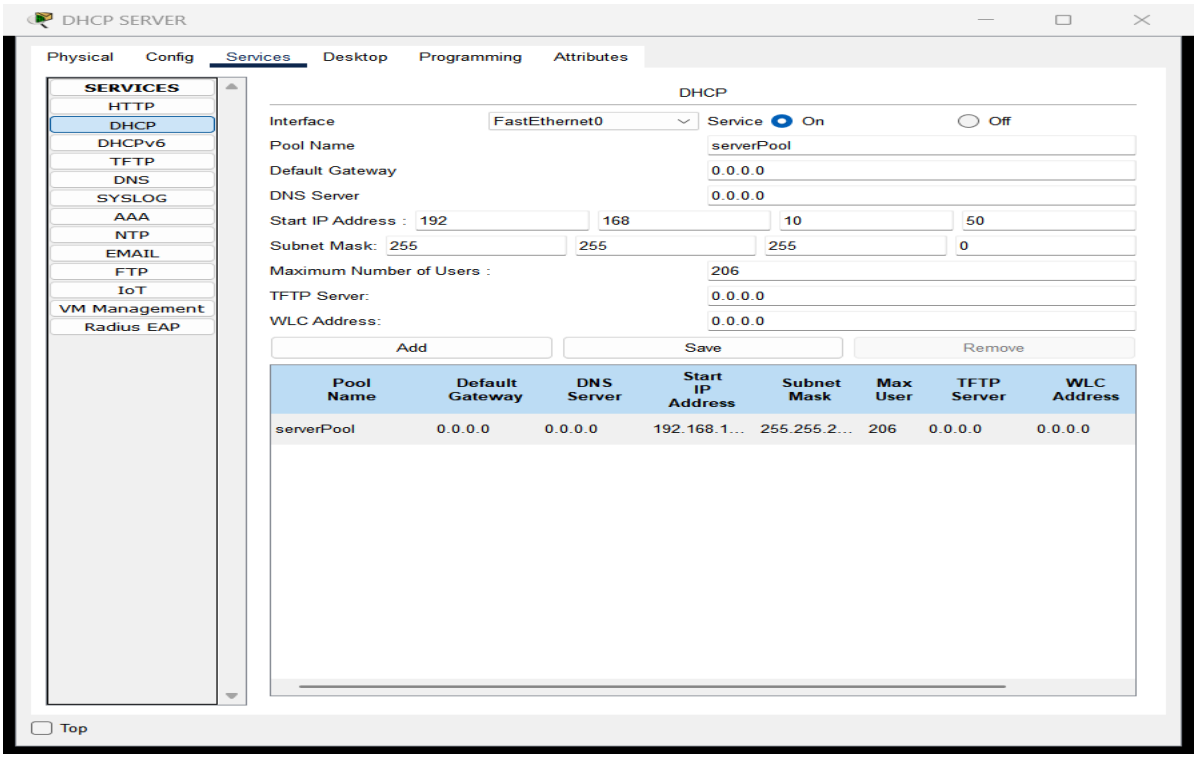


Figure: IP assign to DHCP server for client

5. Set up domain names and corresponding IP addresses on the DNS server.

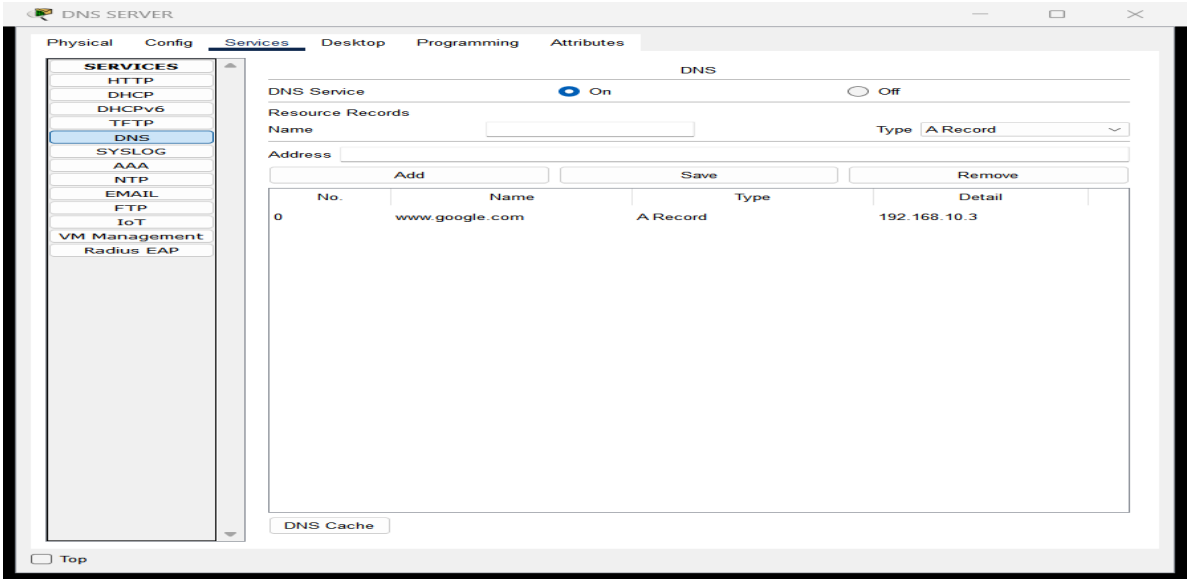


Figure: Assign domain name to IP address

6. Open a PC and:
   - Use a web browser to access the HTTP server.
   - Use the `ftp` command to connect to the FTP server.
   - Use `ipconfig` to check DHCP-assigned configurations.
   - Use `nslookup` to resolve domain names.

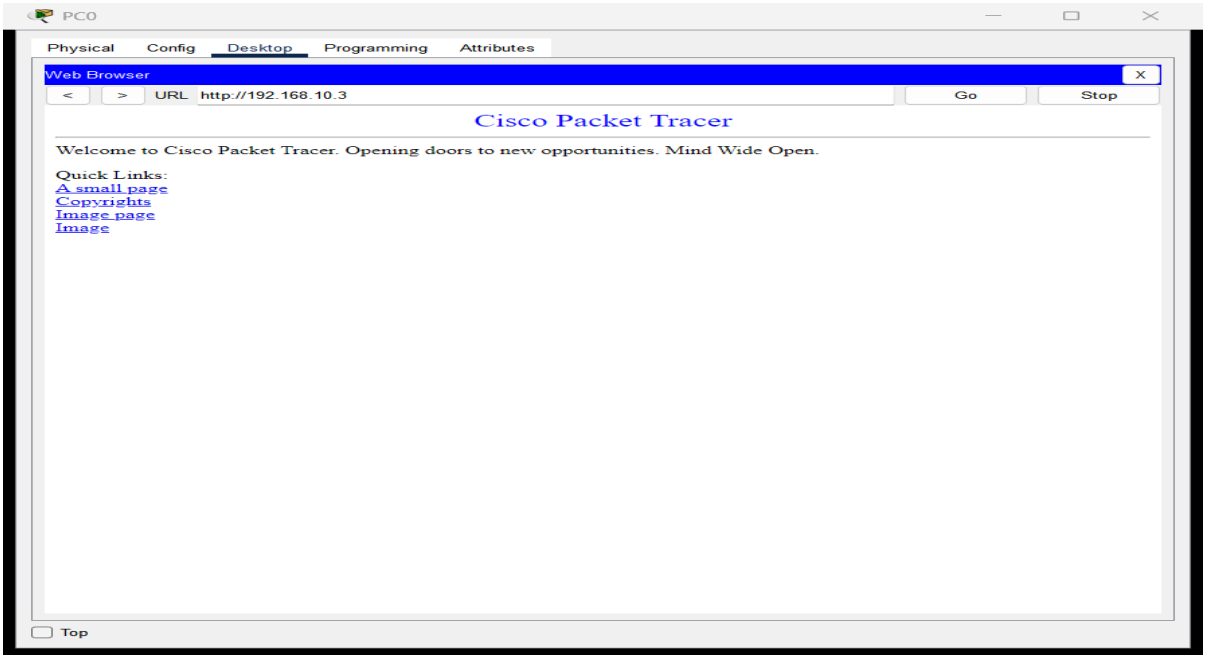


Figure: Access HTTP server from a PC

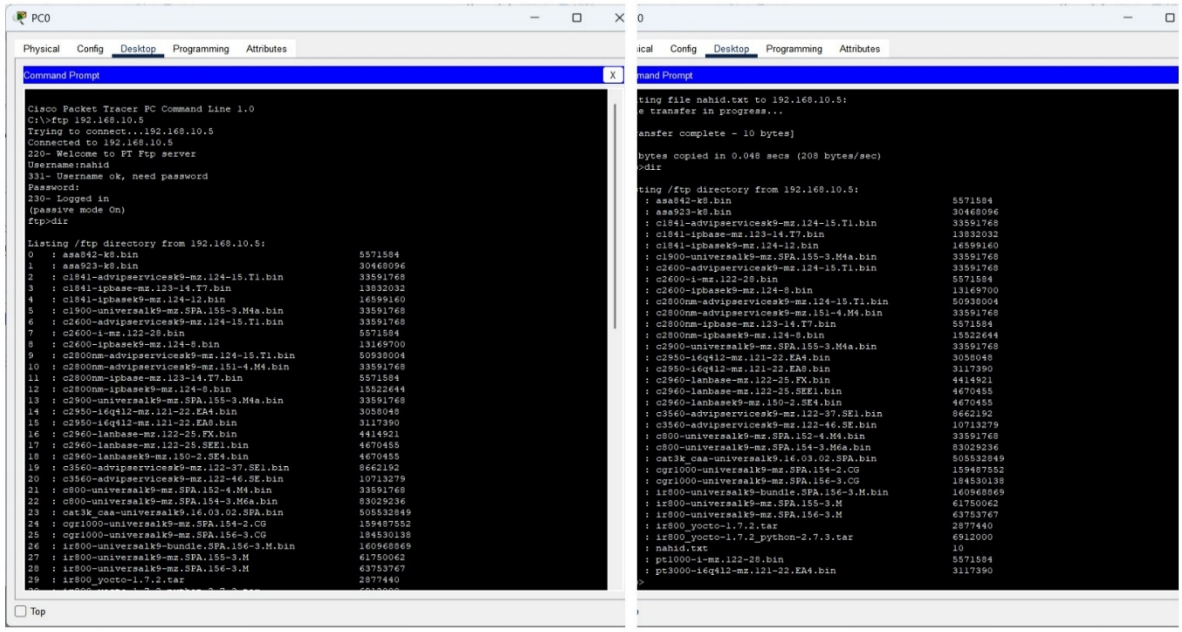


Figure: Put a file into FTP server

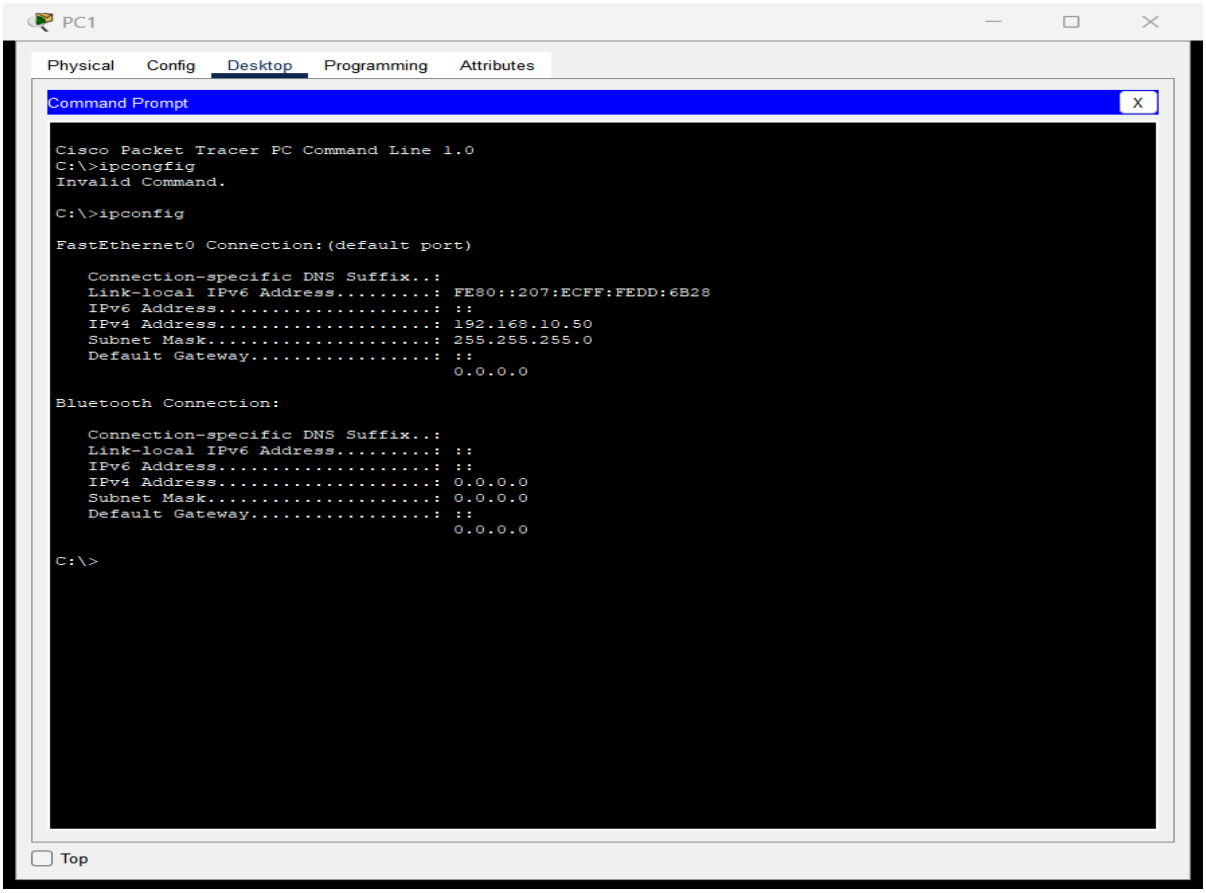


Figure: DHCP-assigned configurations

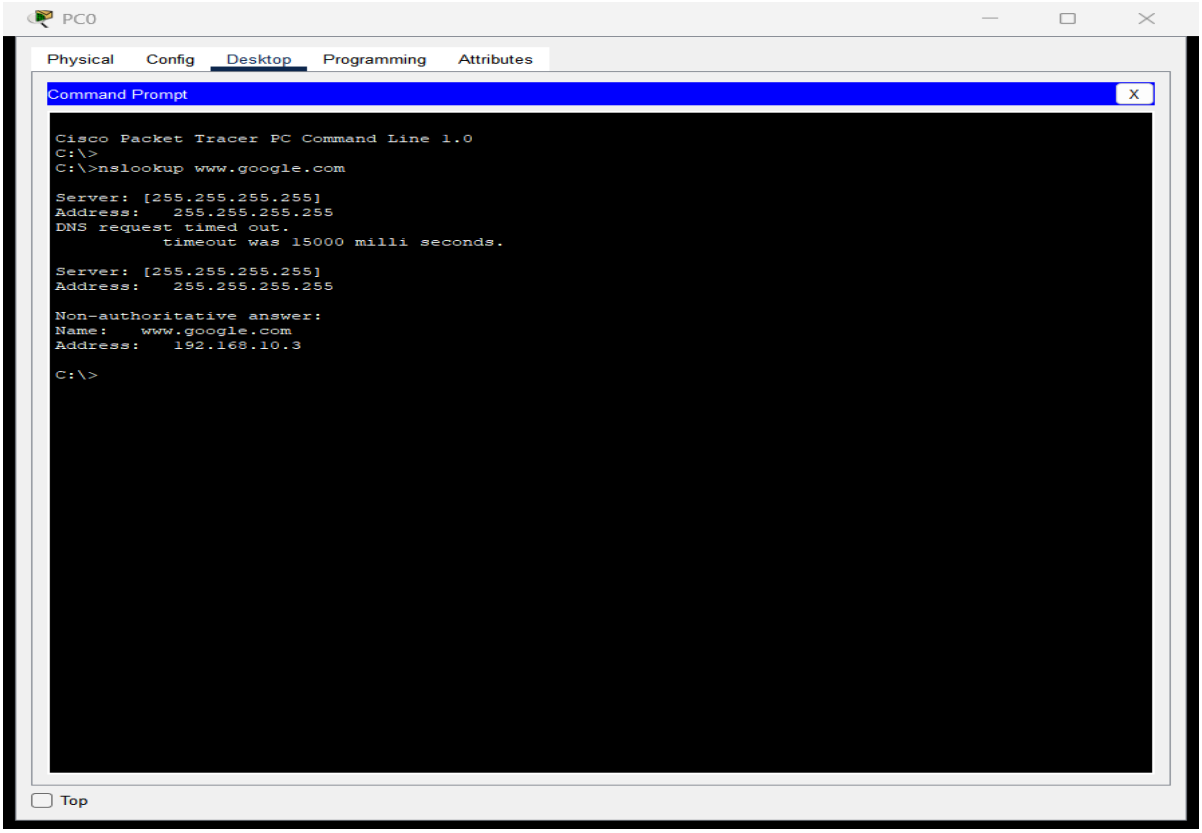


Figure: nslookup Query

7. Monitor packet flow using the simulation mode in Packet Tracer.
8. Capture and analyze FTP, HTTP, DHCP, and DNS packets.

Observations:

- **FTP:** Uses TCP port 21 for connection and data transfer.
- **HTTP:** Uses TCP port 80 to fetch web pages.
- **DHCP:** Follows the DORA process using UDP ports 67 and 68.
- **DNS:** Resolves domain names using UDP port 53.
- All configured services successfully responded to client requests.

Result Analysis:

- The analysis confirmed expected behaviours for all network protocols.
- Packet Tracer's simulation mode effectively visualized protocol interactions.
- Each protocol functioned correctly in the assigned topology.

Conclusion: The experiment successfully demonstrated FTP, HTTP, DHCP, and DNS packet structures and their respective communication processes. This simulation helped in understanding real-world network operations and troubleshooting techniques.

Name of Experiment: IoT based smart Home using WPA2 security & Radius server in Cisco Packet Tracer.

1. Introduction

Creating a smart home simulation in Cisco Packet Tracer (CPT) involves using IoT (Internet of Things) devices to replicate smart home functionalities. Cisco Packet Tracer supports a range of IoT devices, such as smart lights, fans, motion sensors, and cameras. Here's a step-by-step guide to building a smart home simulation.

2. Objectives

- To create a networked smart home using IoT devices.
- To configure devices for automation and control.
- To simulate real-world scenarios such as motion detection and temperature control.
- To test and evaluate the functionality of the smart home.

3. Setup the Environment

- Go to the IoT Devices section in the device list.
- Drag and drop a Home Gateway to the workspace. This acts as the central hub for IoT devices.
- Add a Router and connect it to the Home Gateway using a wired or wireless connection.

4. Diagram

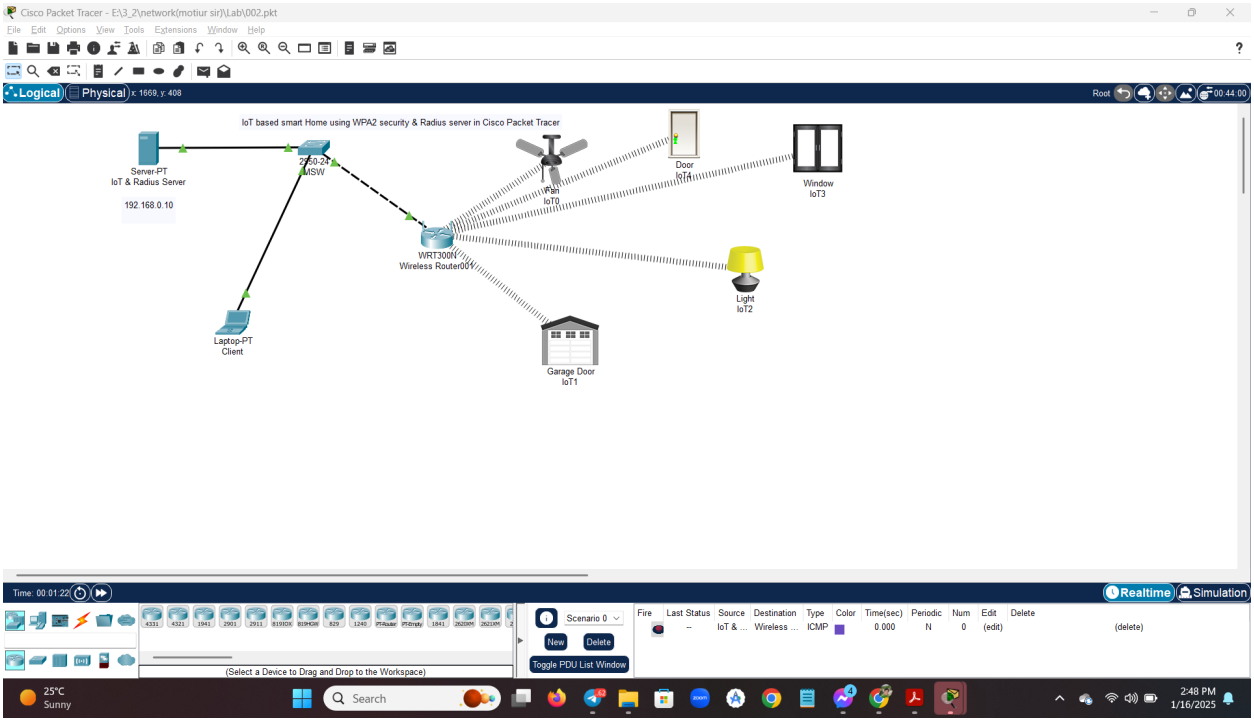


Figure: IoT based smart Home using WPA2 security & Radius server in Cisco Packet Tracer.

5. Add IoT Devices

- Add devices like Smart Lights, Smart Fans, Smart Door Sensors, Smart Window, and Garage Door.
- Place these devices around the workspace to represent different rooms.
- Select each device, click on the Config tab, and assign it to the Home Gateway.
- Provide a unique name and optionally adjust settings like lighting intensity or motion sensitivity.

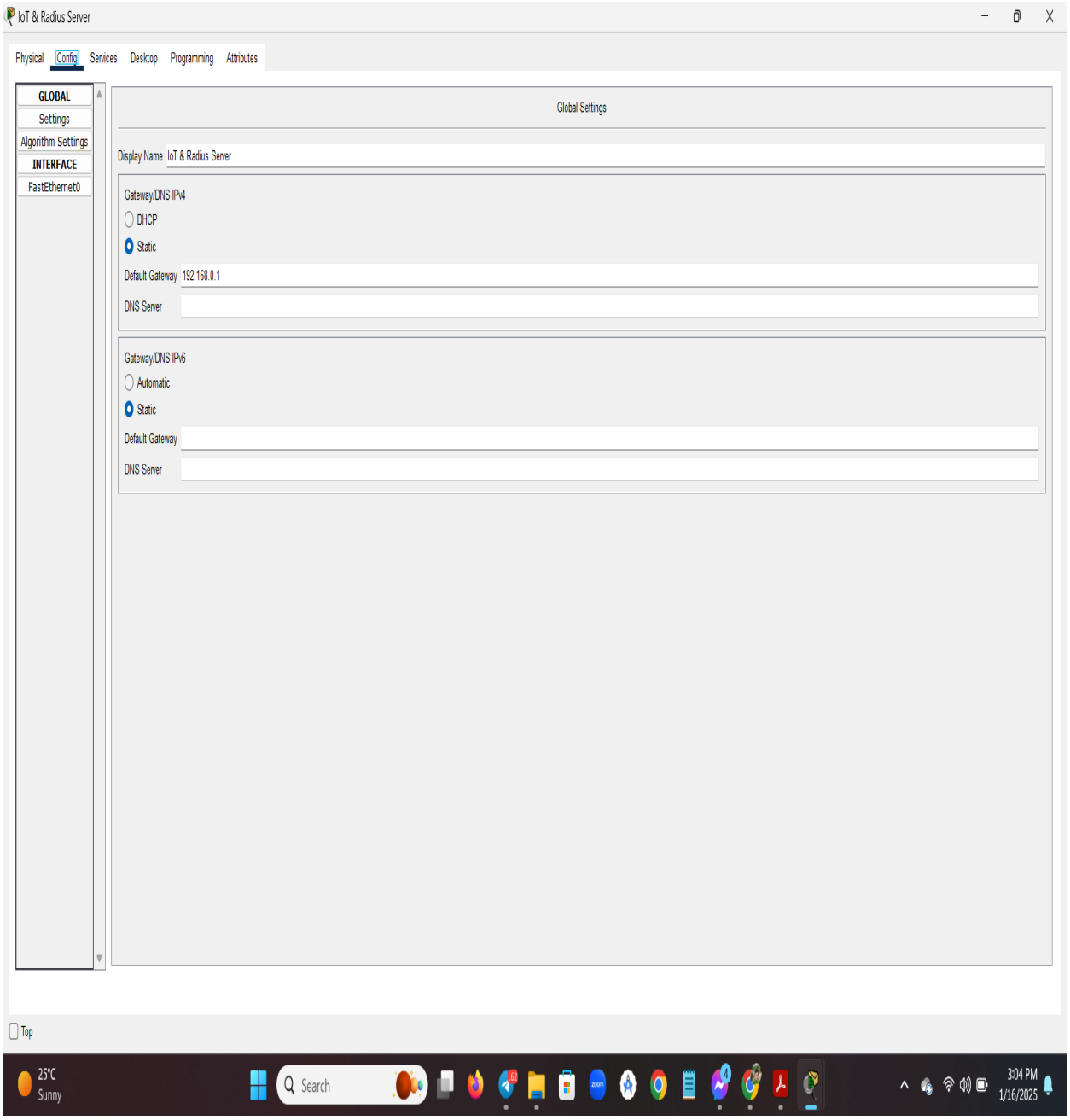


Figure: In IoT & Radius Server Config

6. Program the Devices

Select the Home Gateway, navigate to the Programming tab (or IoT Monitor), and set up conditions for devices.

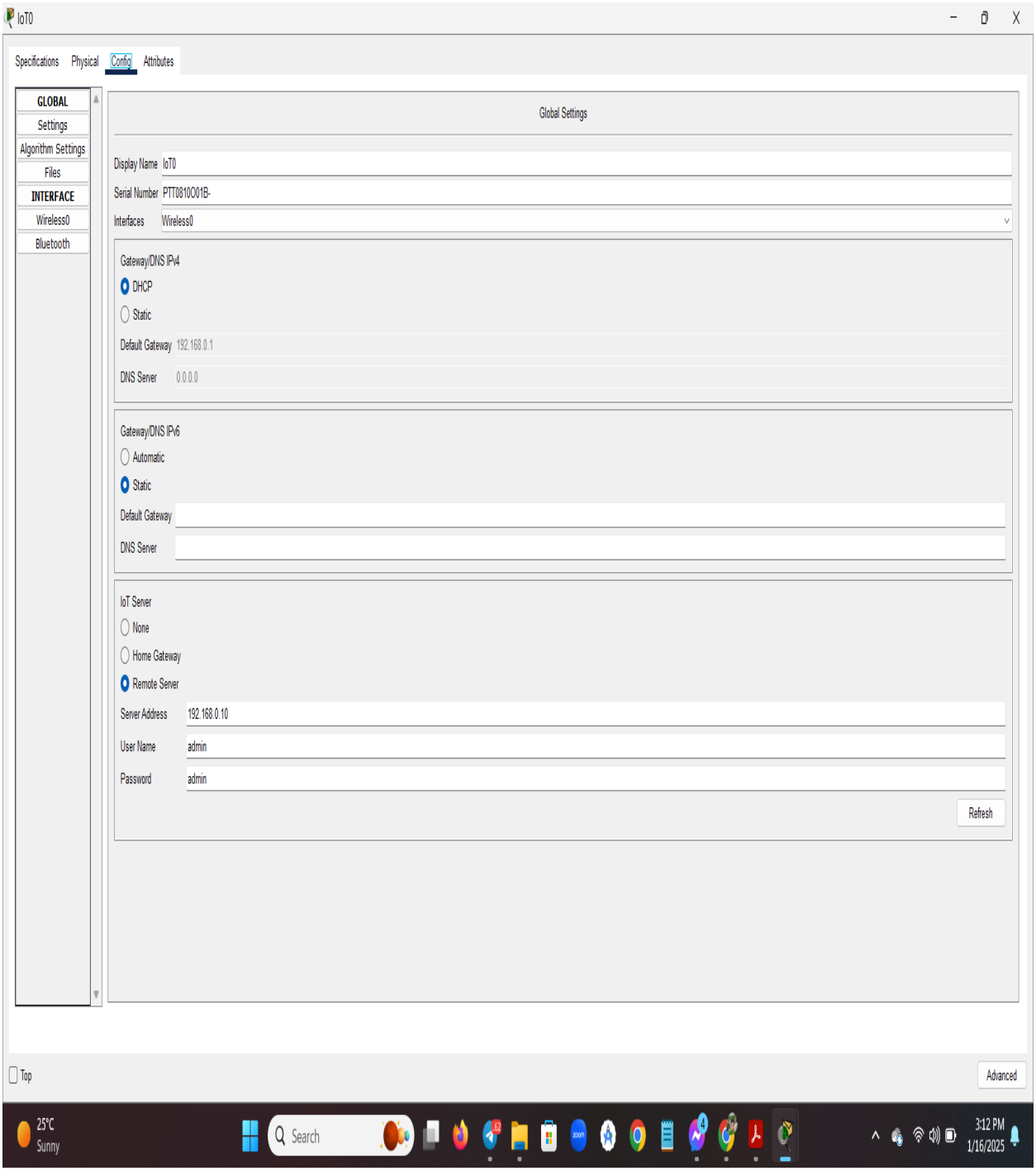


Figure: In Smart Fans config

7. Add User Interaction

- Use Smartphones or Tablets (from the End Devices tab) as controllers.
- Connect them to the Home Gateway's network (Wi-Fi).
- Open the IoT Monitor app on the smartphone and test remote control of devices.

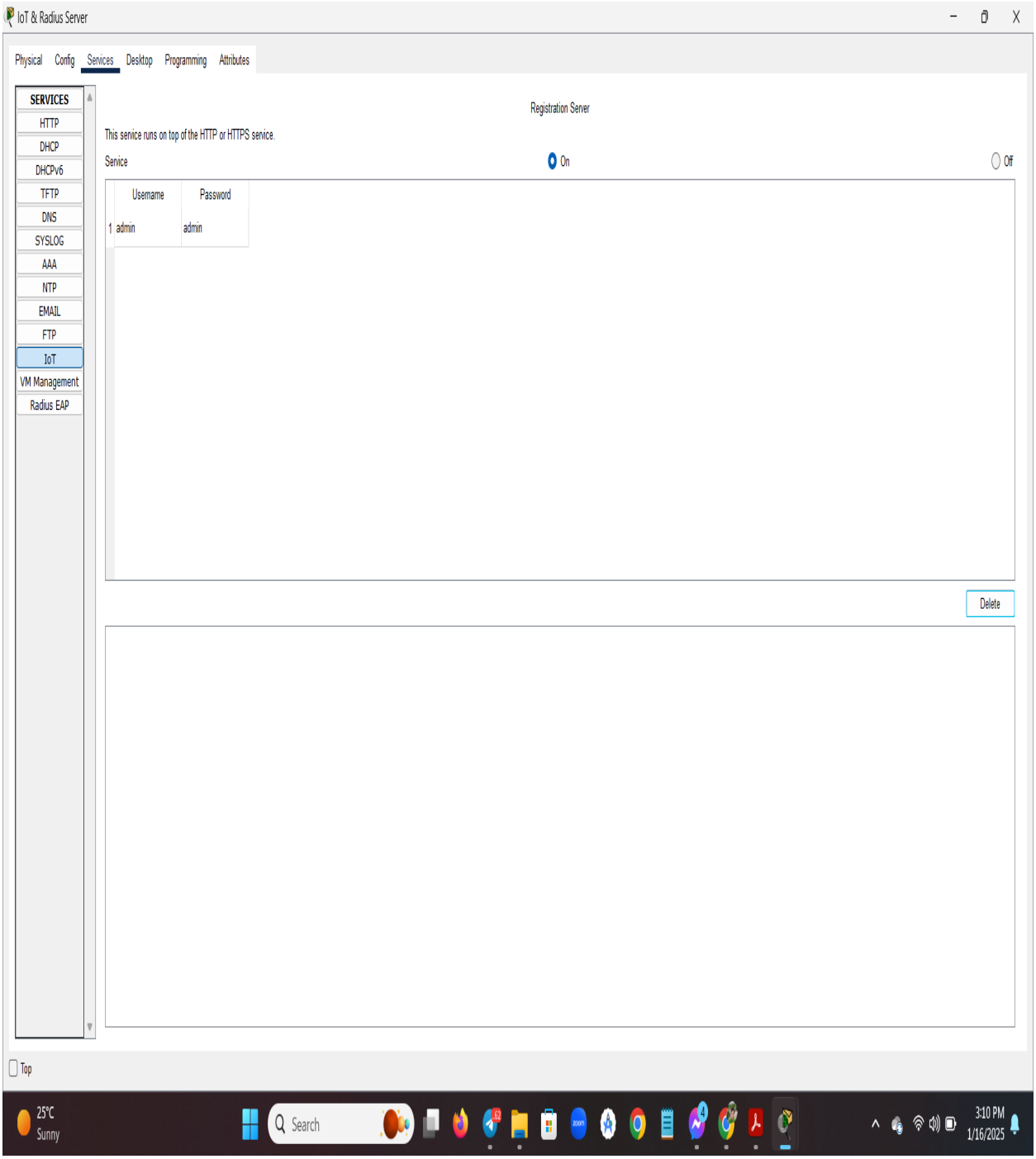


Figure: In IoT & Radius Server IoT config

8. Test and Debug

- Use the Real-Time Mode and simulate various conditions (e.g., motion, time of day).
- Switch to Simulation Mode to visualize data packets between devices.

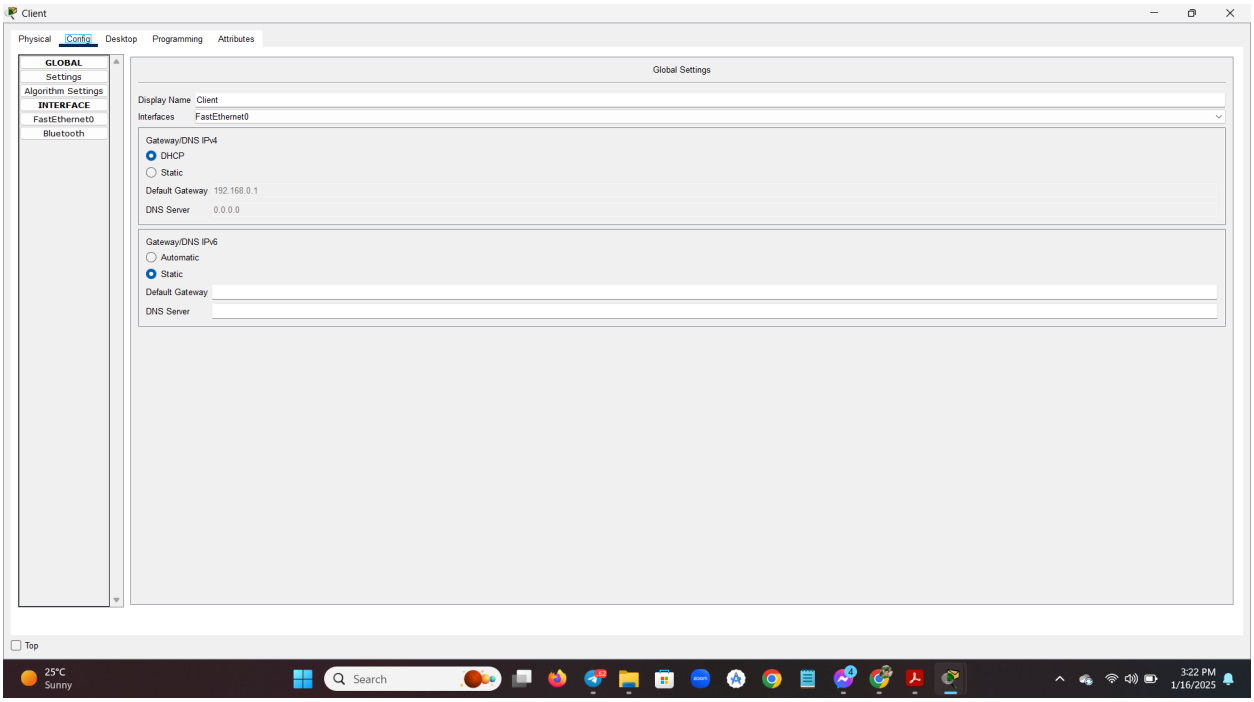


Figure: In Client Config

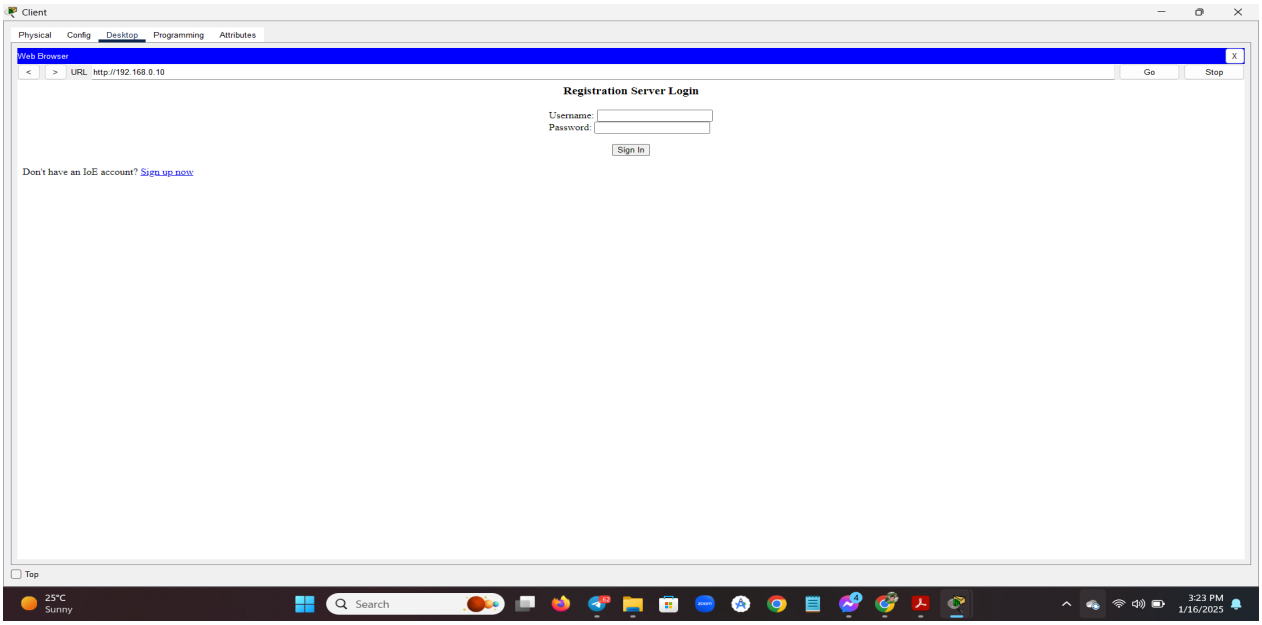


Figure: In Client web browser for browsing and give the username and password for login

9. Example Scenario

- Add a Motion Detector in the living room.
- Configure it to turn on the Smart Light when motion is detected.

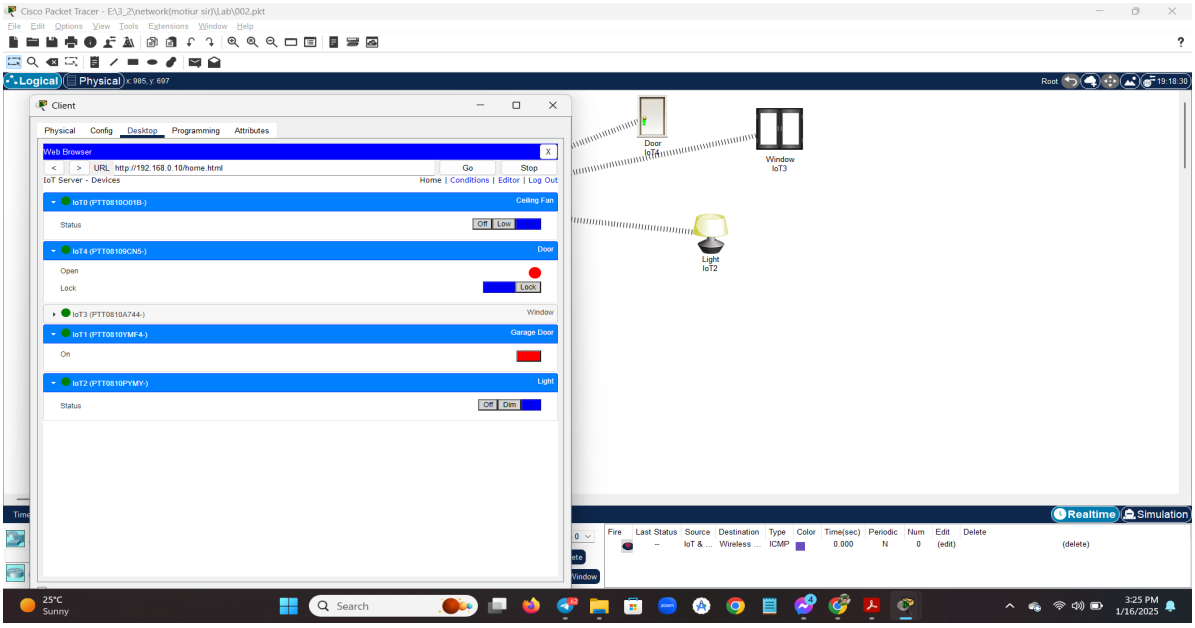


Figure: After login it shows all the connections for IoT based smart Home.

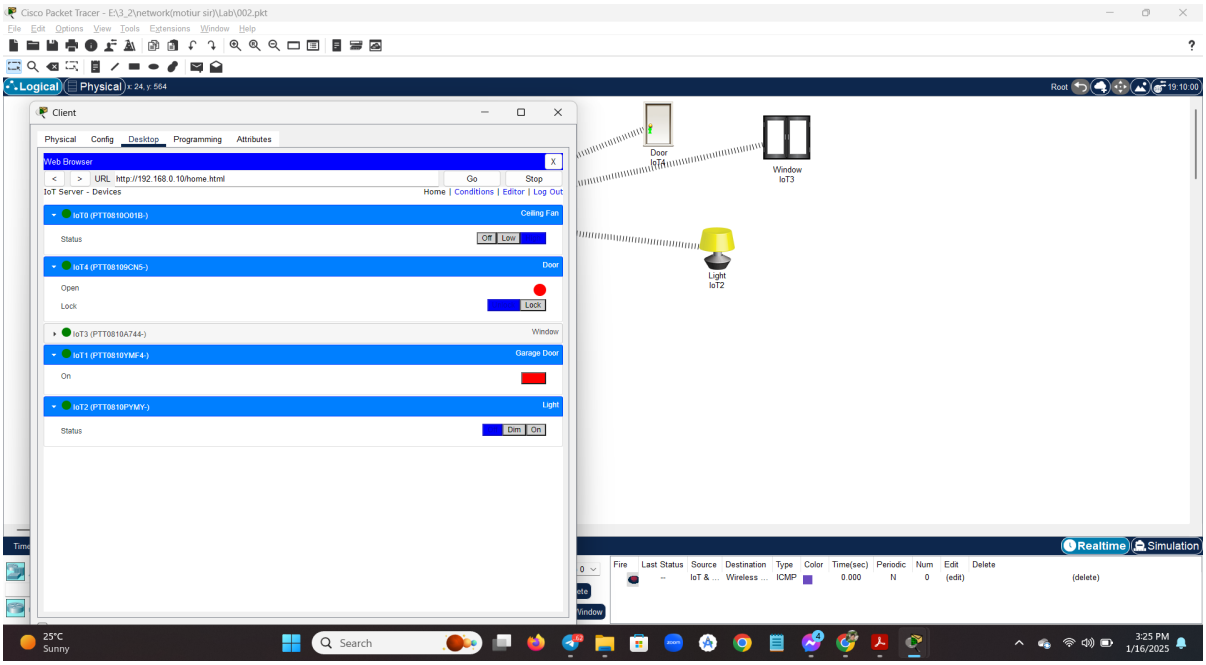


Figure: work for light off to On

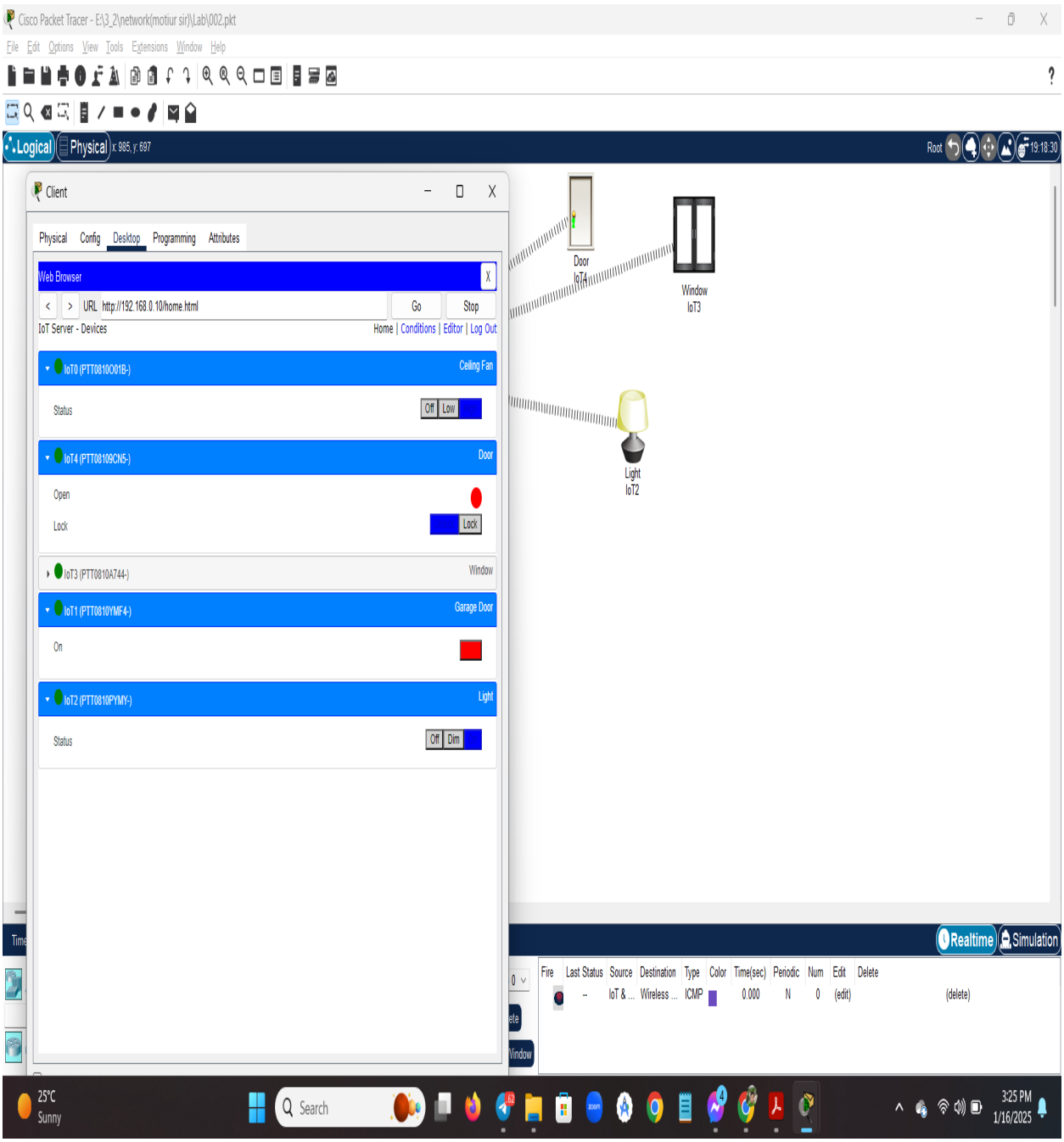


Figure: The light Work

The smart home simulation successfully demonstrated the integration and automation of IoT devices in a virtual environment. The use of Cisco Packet Tracer allowed for effective testing of device interactions, automation scenarios, and network functionality. Future work could explore advanced security features and integration with voice-controlled assistants.

Name of Experiment: Cisco SDN implementation with REST API.

1. Introduction

Cisco's SDN solutions support interaction via REST APIs, enabling automation, programmability, and integration with external systems. Key Cisco platforms like Cisco Application Centric Infrastructure (ACI) and Cisco Digital Network Architecture (DNA) expose REST APIs for developers to manage and automate network configurations, policies, and monitoring.

2. Objectives

- Explore Cisco SDN REST API Features.
- Implement Network Automation.
- Develop Custom Applications.
- Integrate with Existing Tools.

3. Setup the Environment

Cisco ACI (Application Centric Infrastructure)

- Automating network policies.
- Configuring application profiles and tenants.
- Monitoring fabric health and performance.

Cisco DNA (Digital Network Architecture)

- Intent-based networking.
- Automating device provisioning and configuration.
- Monitoring and troubleshooting network issues with AI/ML insights.

4. Diagram

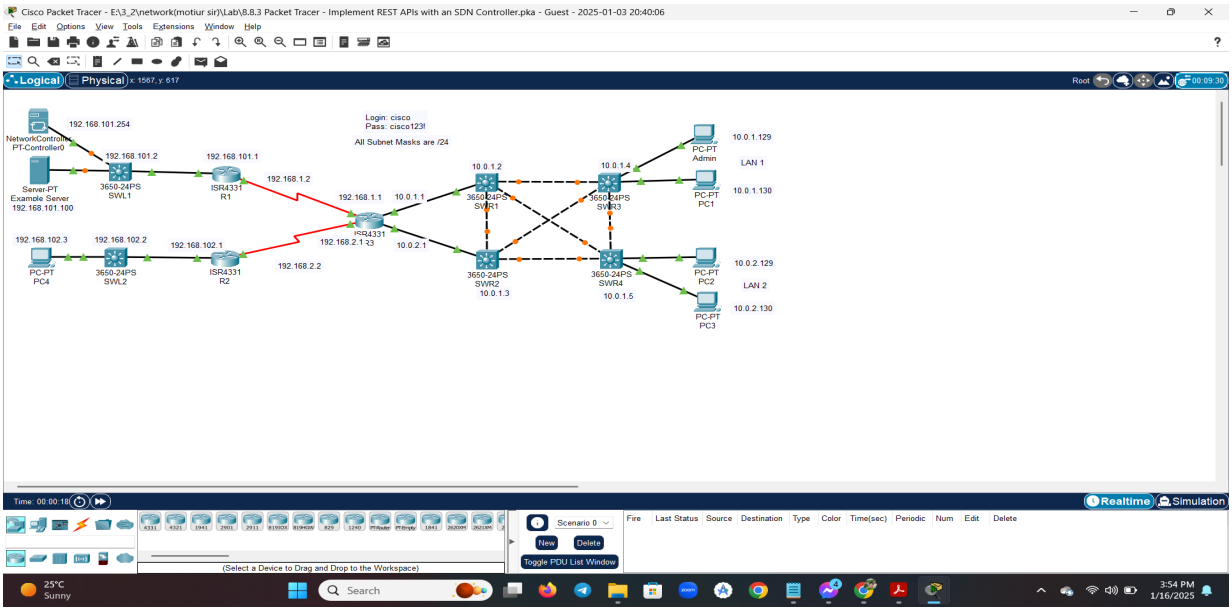


Fig: Cisco SDN implementation with REST API

5. Choose the Appropriate SDN Controller

- Cisco APIC for ACI environments.
- Cisco DNA Center for campus and WAN SDN.
- Open SDN Controllers (e.g. OpenDaylight) for integration with Cisco devices in custom SDN environments.

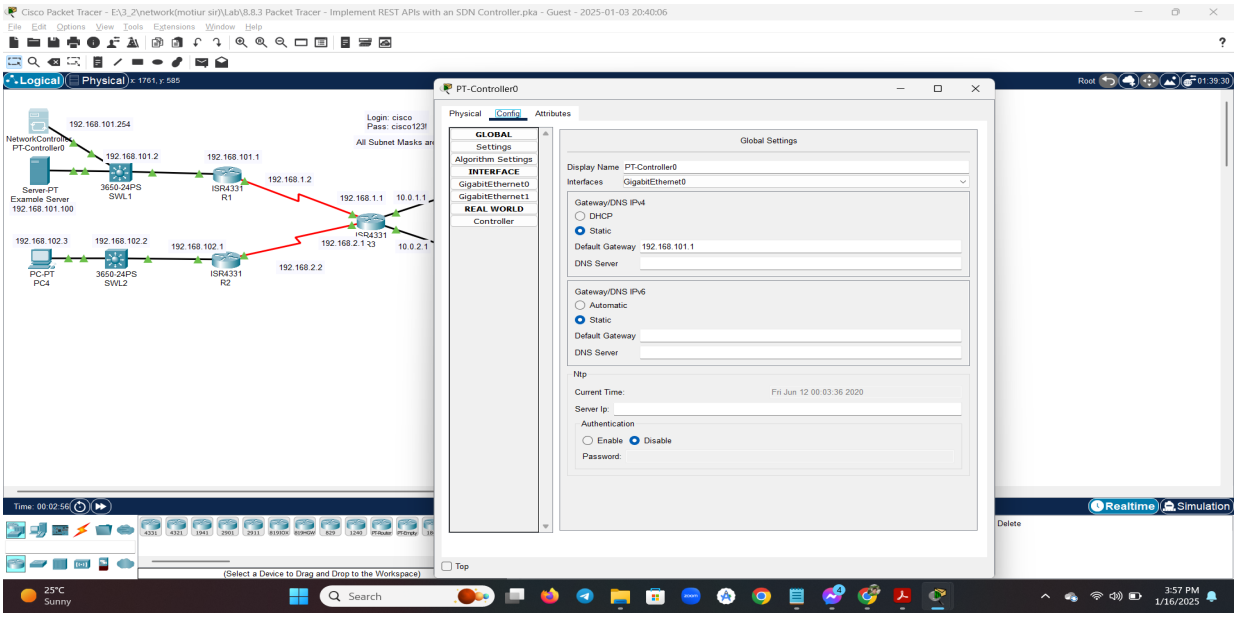


Fig: Setup the Gateway in Config from Network Controller

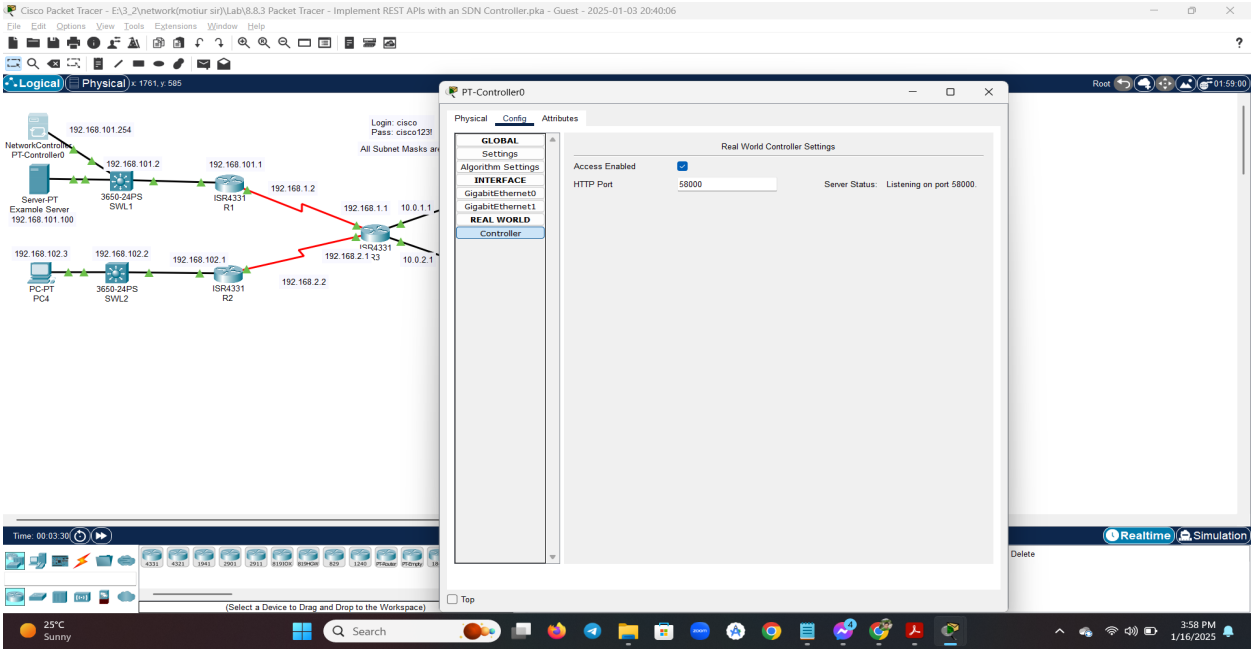


Fig: Setup the HTTP Port in Network Controller

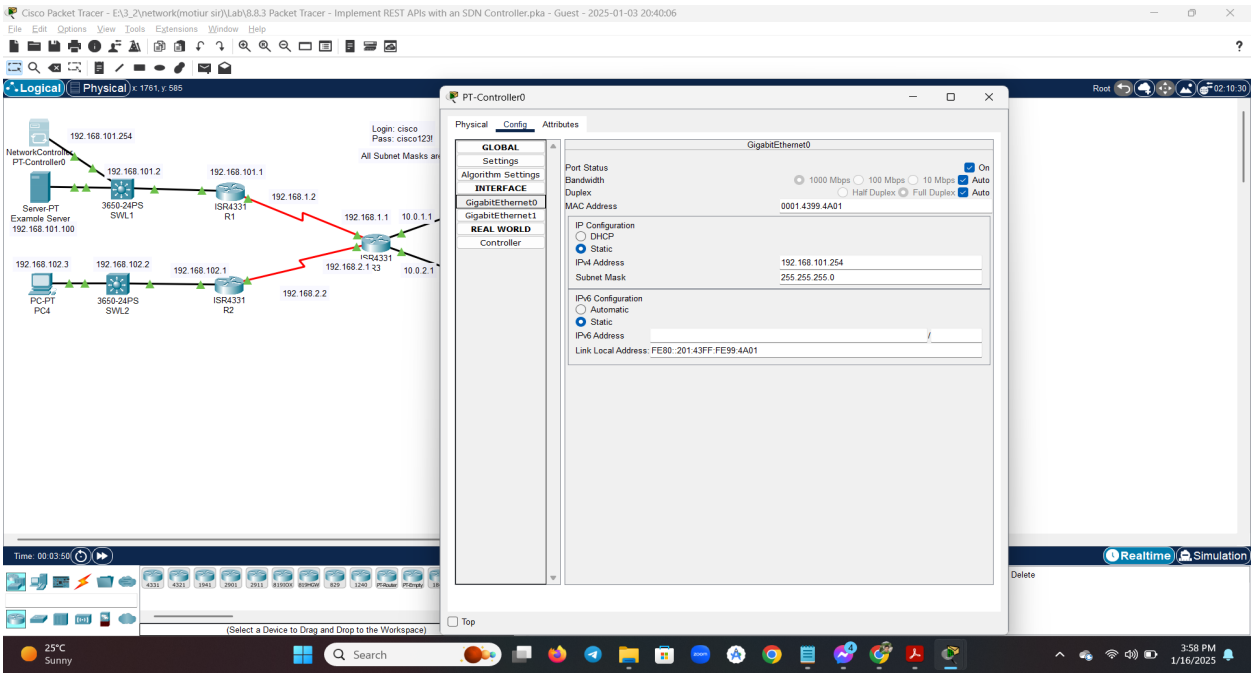


Figure: Setup the Config controller in Network Controller

6. Connect Devices to the SDN Controller

- Configure network devices to communicate with the SDN controller.
- Enable protocols like OpenFlow, Netconf/YANG, or proprietary APIs (Cisco uses NX-API for Nexus devices).

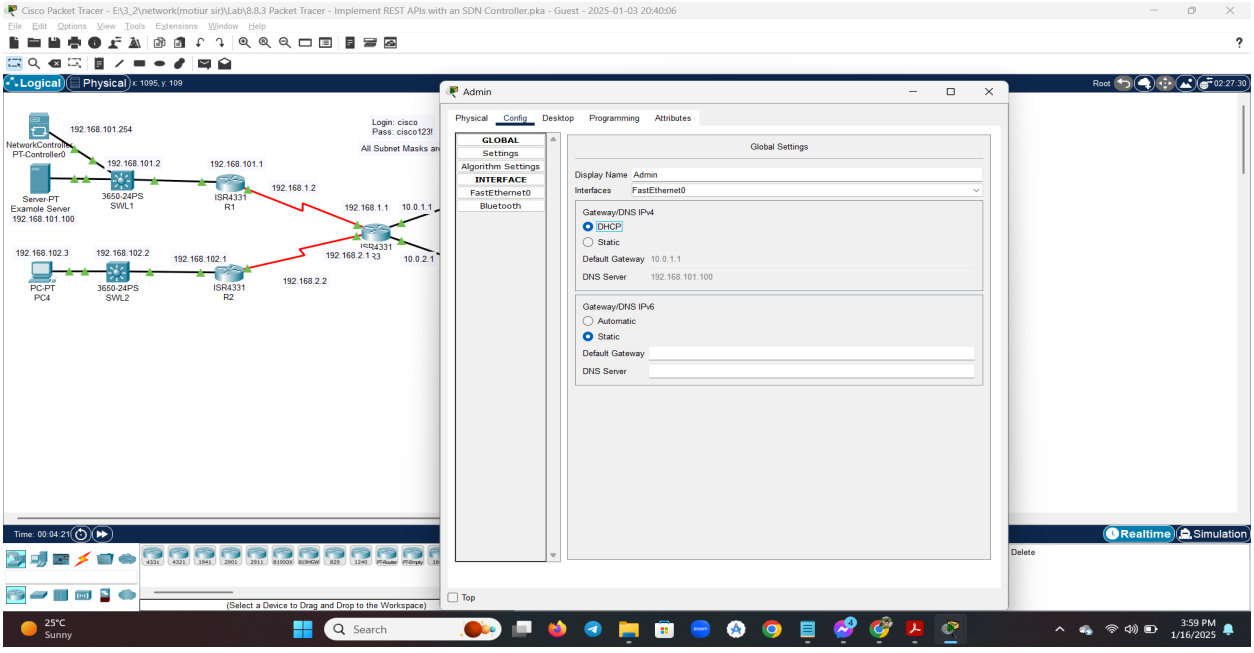


Figure: Setup the Admin Config DHCP

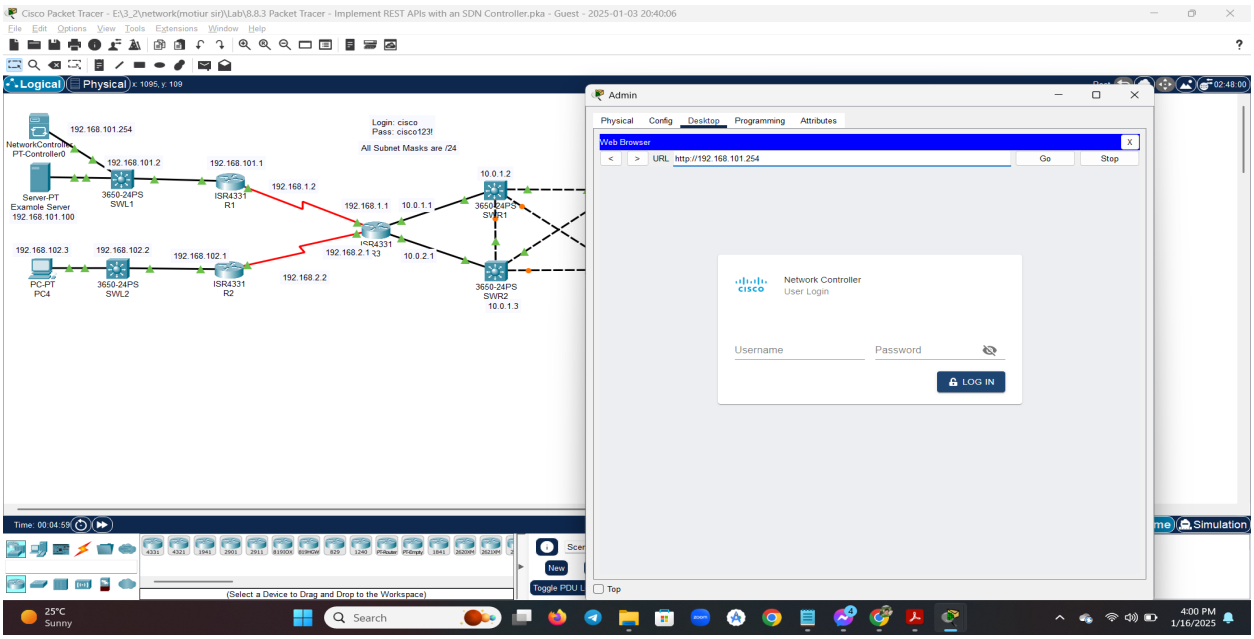


Figure: Going Admin Web Browser and Run with IP address and Give Username and Password

7. Test and Validate

- Simulate traffic and test policies to ensure they work as intended.
- Validate the network's ability to scale and adapt to changes.

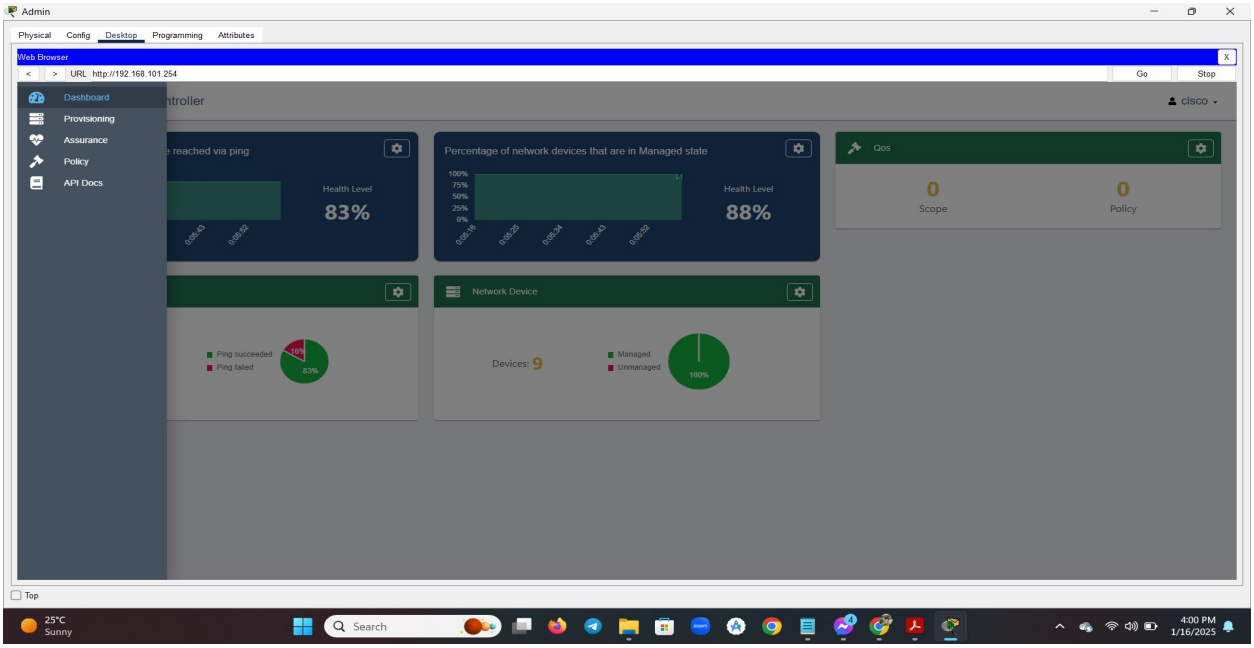


Figure: After login open the control page

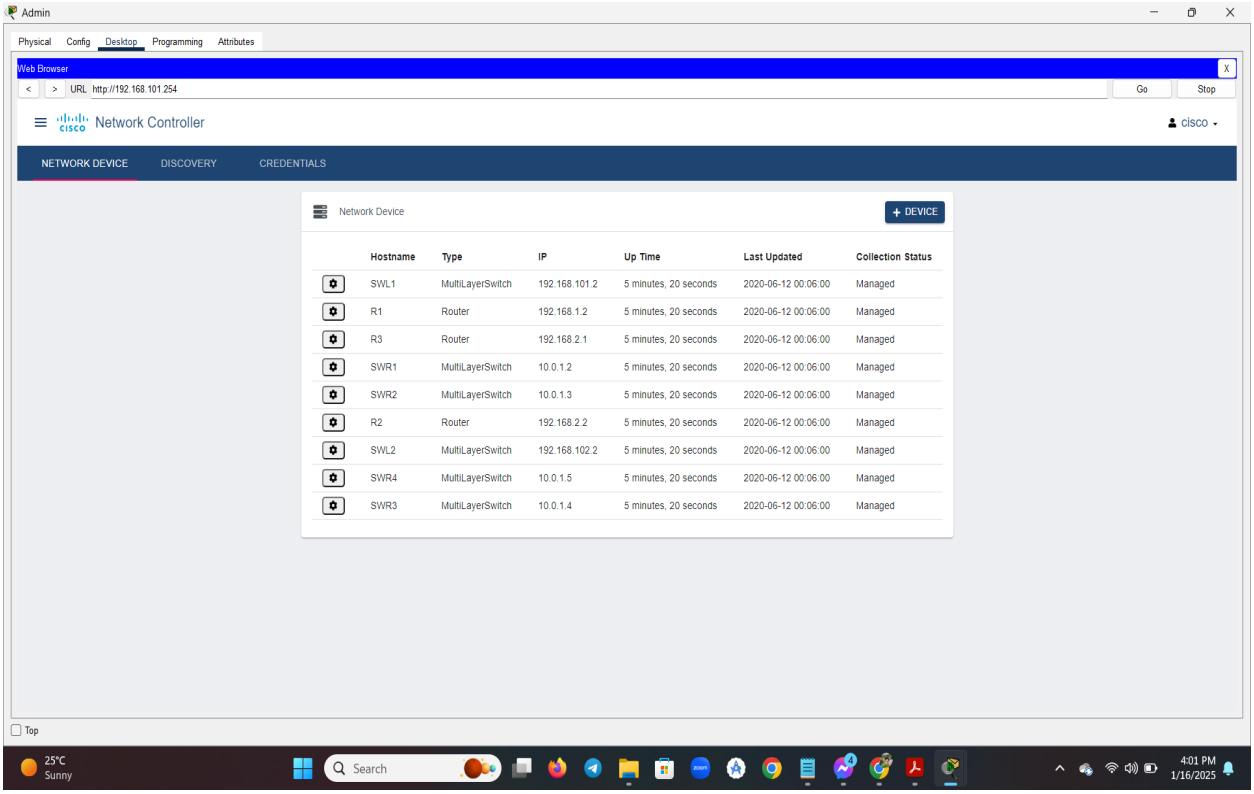


Figure: Going to Provisioning for show all the server

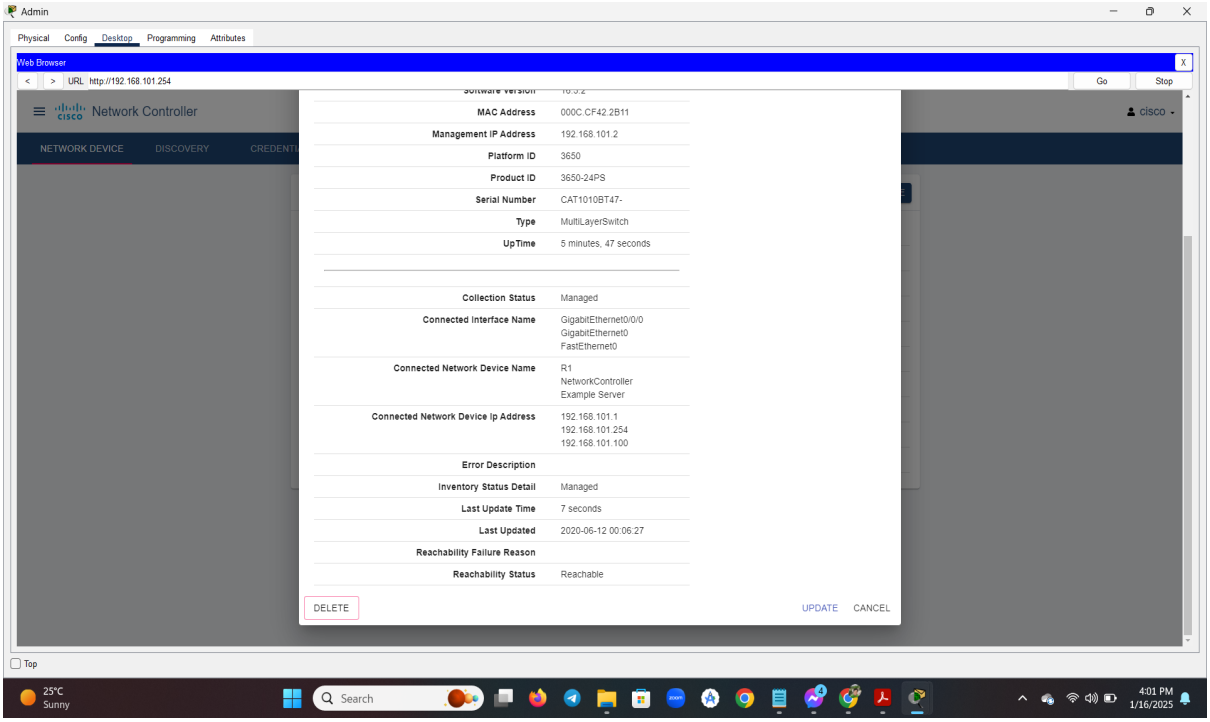


Figure: Show the config of any server which can be delete or update

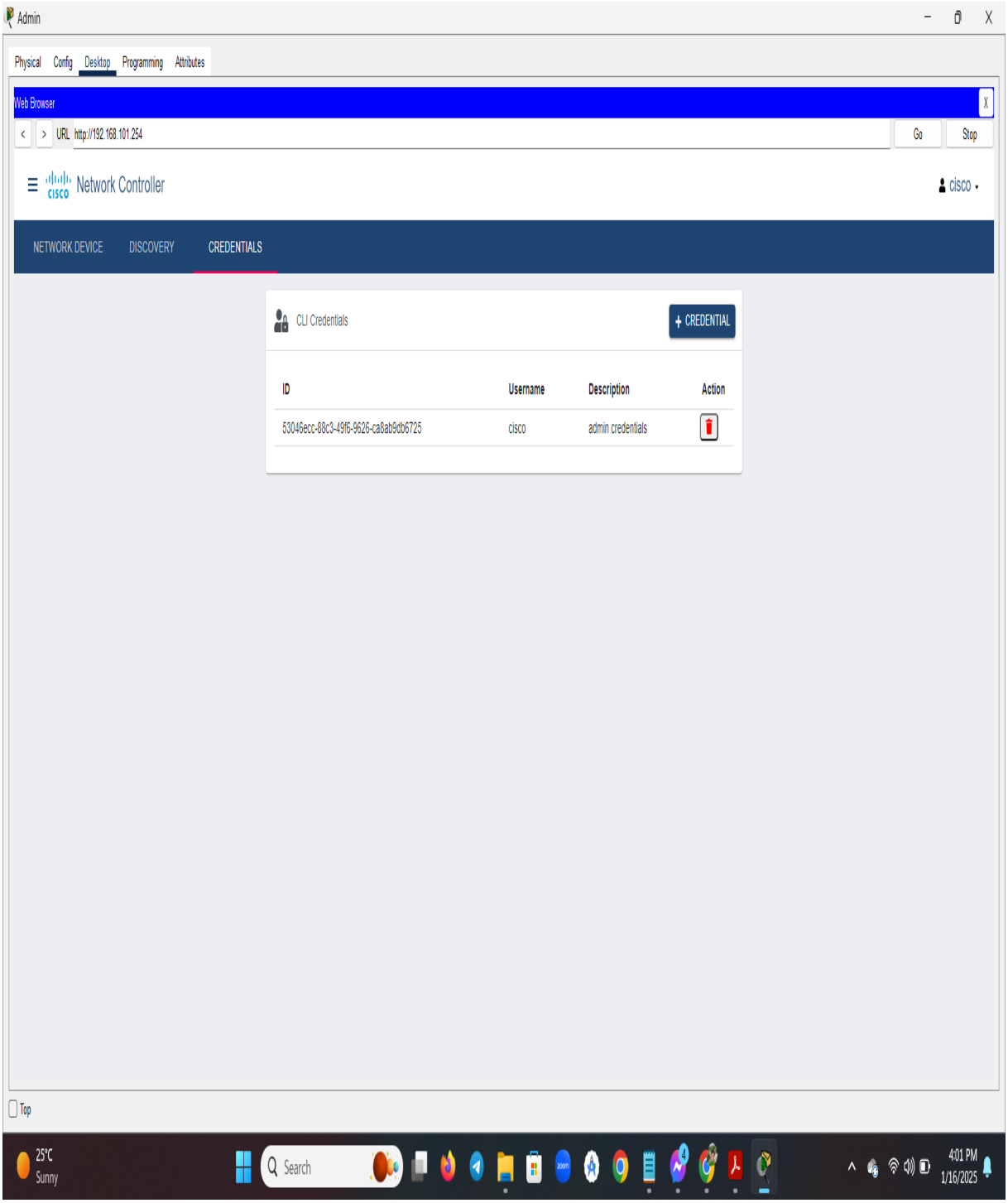


Figure: Going Credentials for setup the username and password

The lab reinforced the significance of SDN in modern network management and showcased how REST APIs can be leveraged to automate, monitor, and optimize network operations. This approach is essential for building efficient, future-proof networking solutions in today's dynamic IT environments.

Name of Experiment: Creating a Mininet Topology.

1. Introduction

This lab demonstrates the creation of a simple network topology using Mininet to simulate two LANs interconnected by a router. Each LAN consists of two hosts and a switch, with the router providing inter-LAN communication. Mininet's flexibility is utilized to configure IP addresses, routing, and forwarding rules, showcasing a practical implementation of basic networking concepts.

2. Objectives

- To design and simulate a network topology with two LANs connected via a router using Mininet.
- To configure IP addresses and routing rules for hosts and the router to enable inter-LAN communication.
- To understand and apply network concepts such as IP forwarding, default gateways, and routing table configuration.
- To evaluate Mininet's functionality in creating and testing software-defined network topologies.

3. Setup the Environment

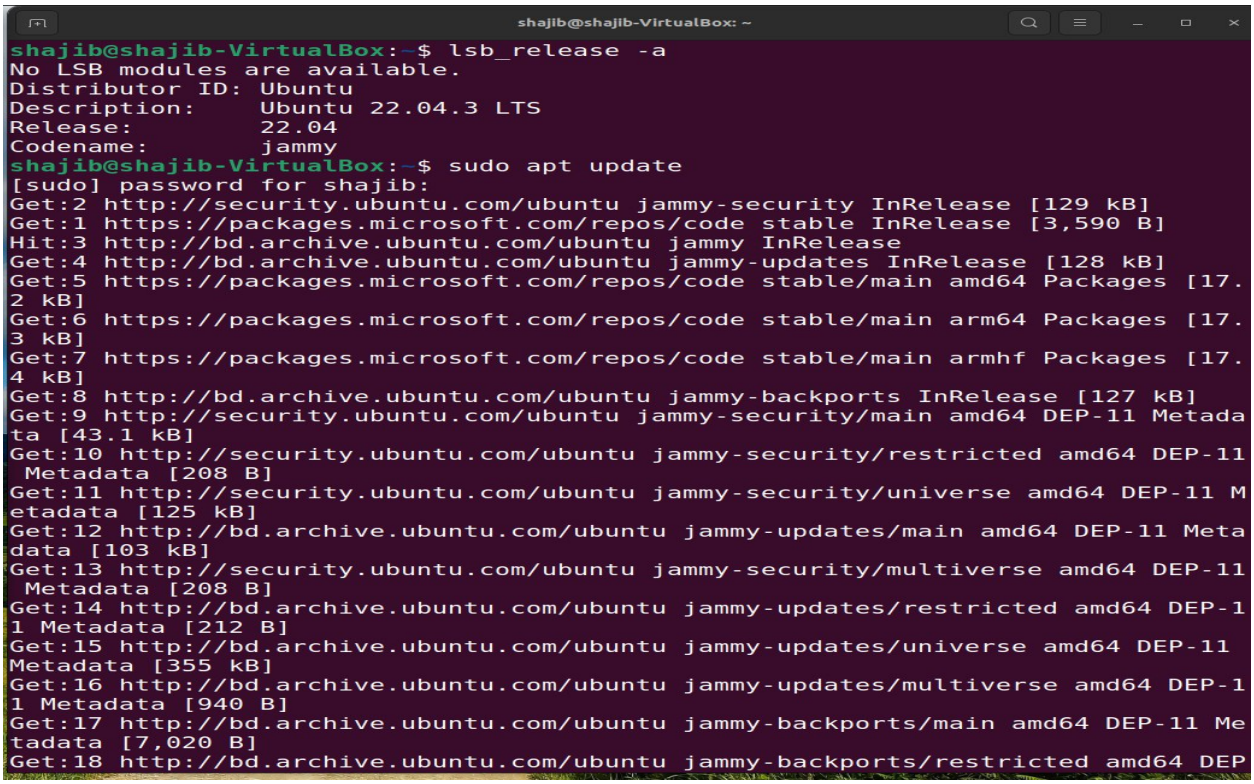


Figure: Check version and Update Ubuntu

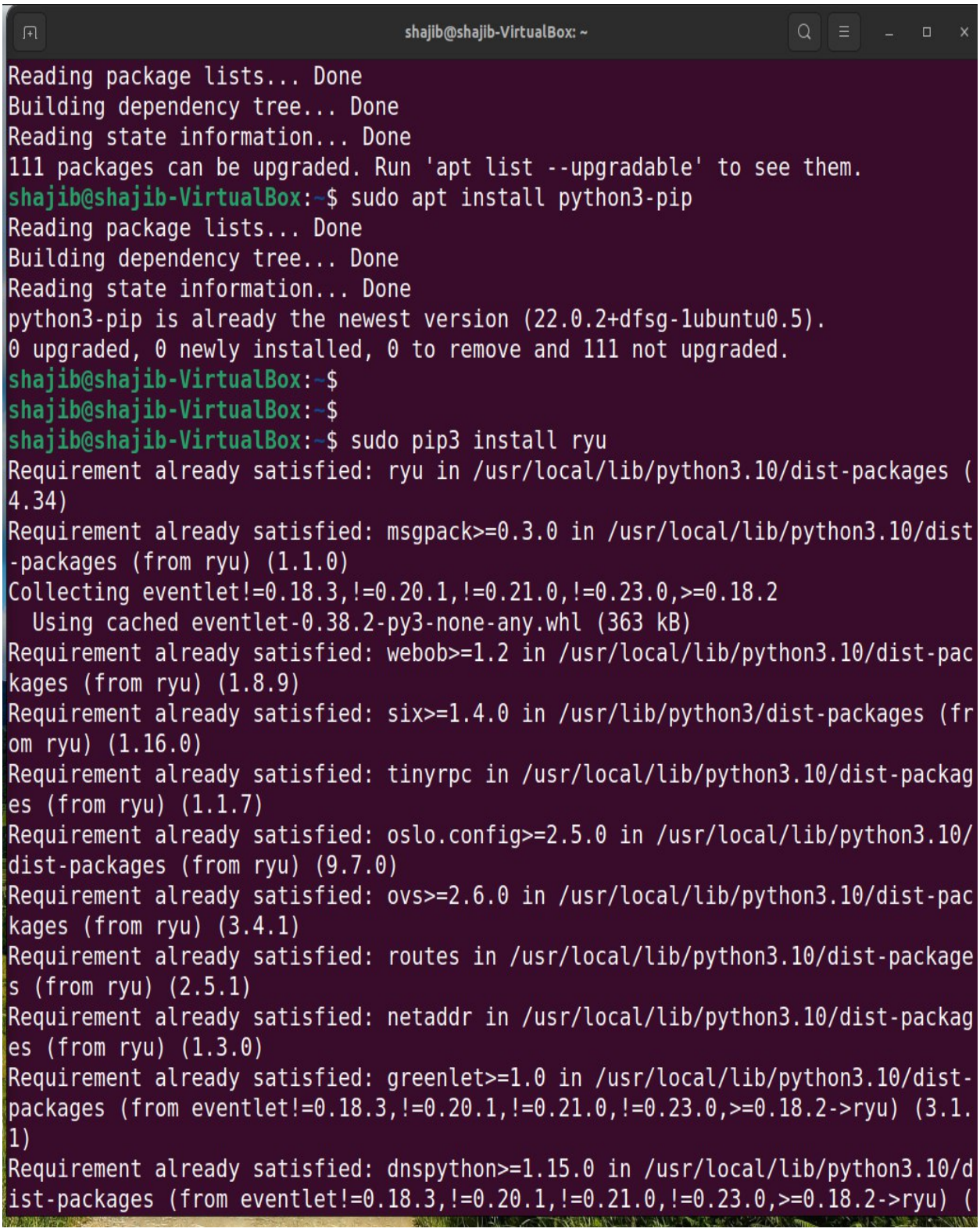


Figure: Install python3-pip and pip3 ryu

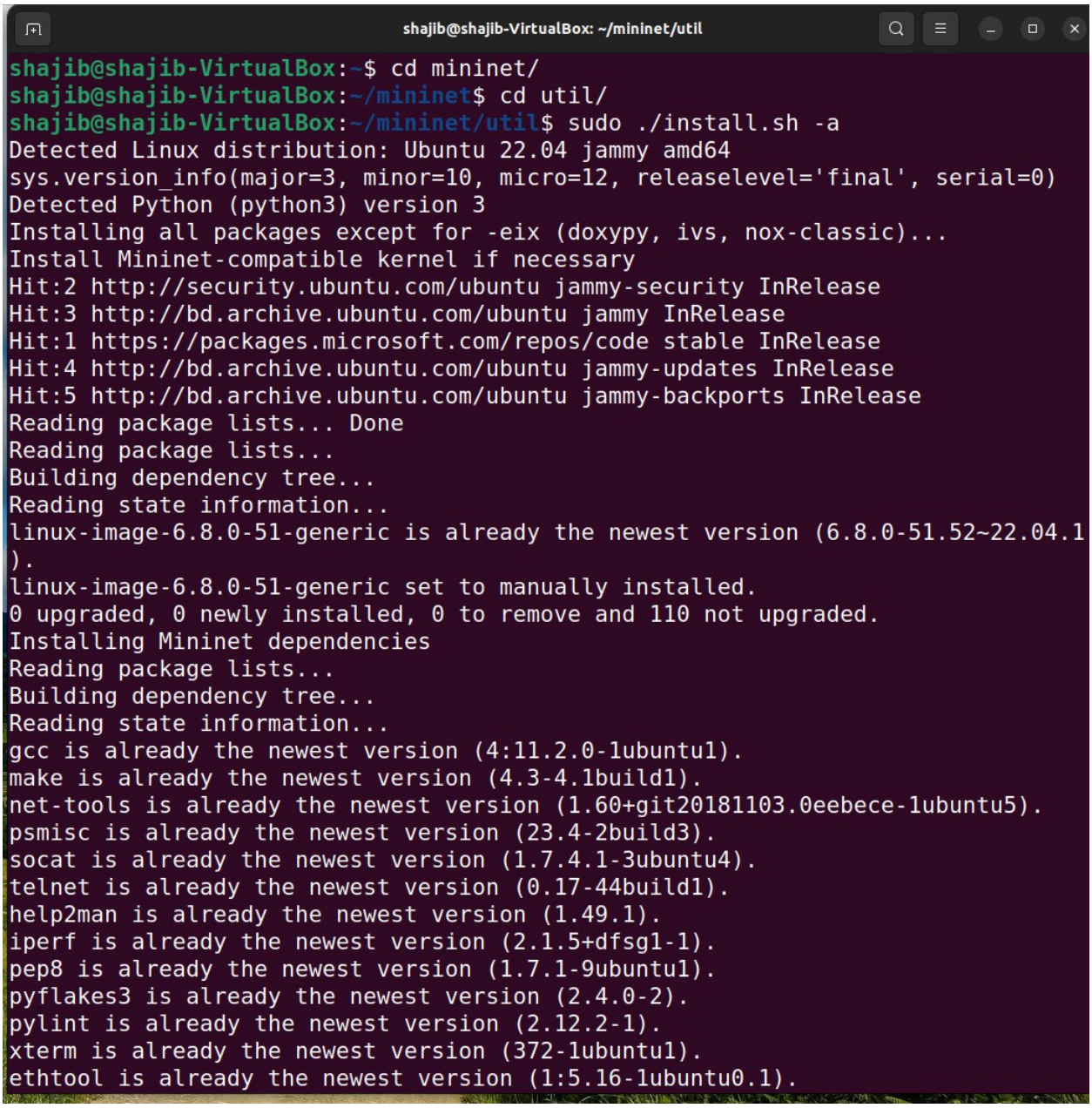


Figure: install.sh -a on mininet/util

4. Source Code (Python)

```
from mininet.net import Mininet
from mininet.node import Controller, OVSKernelSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink

def createTopology():
    net = Mininet(controller=Controller, link=TCLink, switch=OVSKernelSwitch)
```

```
info('* Adding controller\n')
net.addController('c0')

info('* Adding hosts and switches\n')
# LAN 1
h1 = net.addHost('h1', ip='10.0.1.1/24')
h2 = net.addHost('h2', ip='10.0.1.2/24')
s1 = net.addSwitch('s1')

# LAN 2
g1 = net.addHost('g1', ip='10.0.2.1/24')
g2 = net.addHost('g2', ip='10.0.2.2/24')
s2 = net.addSwitch('s2')

# Router
r1 = net.addHost('r1')

info('* Creating links\n')
# LAN 1 Links
net.addLink(h1, s1)
net.addLink(h2, s1)
net.addLink(s1, r1, intfName2='r1-eth1', params2={'ip': '10.0.1.254/24'})

# LAN 2 Links
net.addLink(g1, s2)
net.addLink(g2, s2)
net.addLink(s2, r1, intfName2='r1-eth2', params2={'ip': '10.0.2.254/24'})

info('* Starting network\n')
net.start()

# Configure router
info('* Configuring router\n')
r1.cmd('sysctl -w net.ipv4.ip_forward=1')
r1.cmd('ifconfig r1-eth1 10.0.1.254/24')
r1.cmd('ifconfig r1-eth2 10.0.2.254/24')

# Configure routes
info('* Configuring routes\n')
h1.cmd('ip route add default via 10.0.1.254')
h2.cmd('ip route add default via 10.0.1.254')
g1.cmd('ip route add default via 10.0.2.254')
g2.cmd('ip route add default via 10.0.2.254')

info('* Running CLI\n')
CLI(net)
```

```
    info('* Stopping network\n')
    net.stop()

if _name_ == '_main_':
    setLogLevel('info')
    createTopology()
```

5. Screenshot of Work Procedure

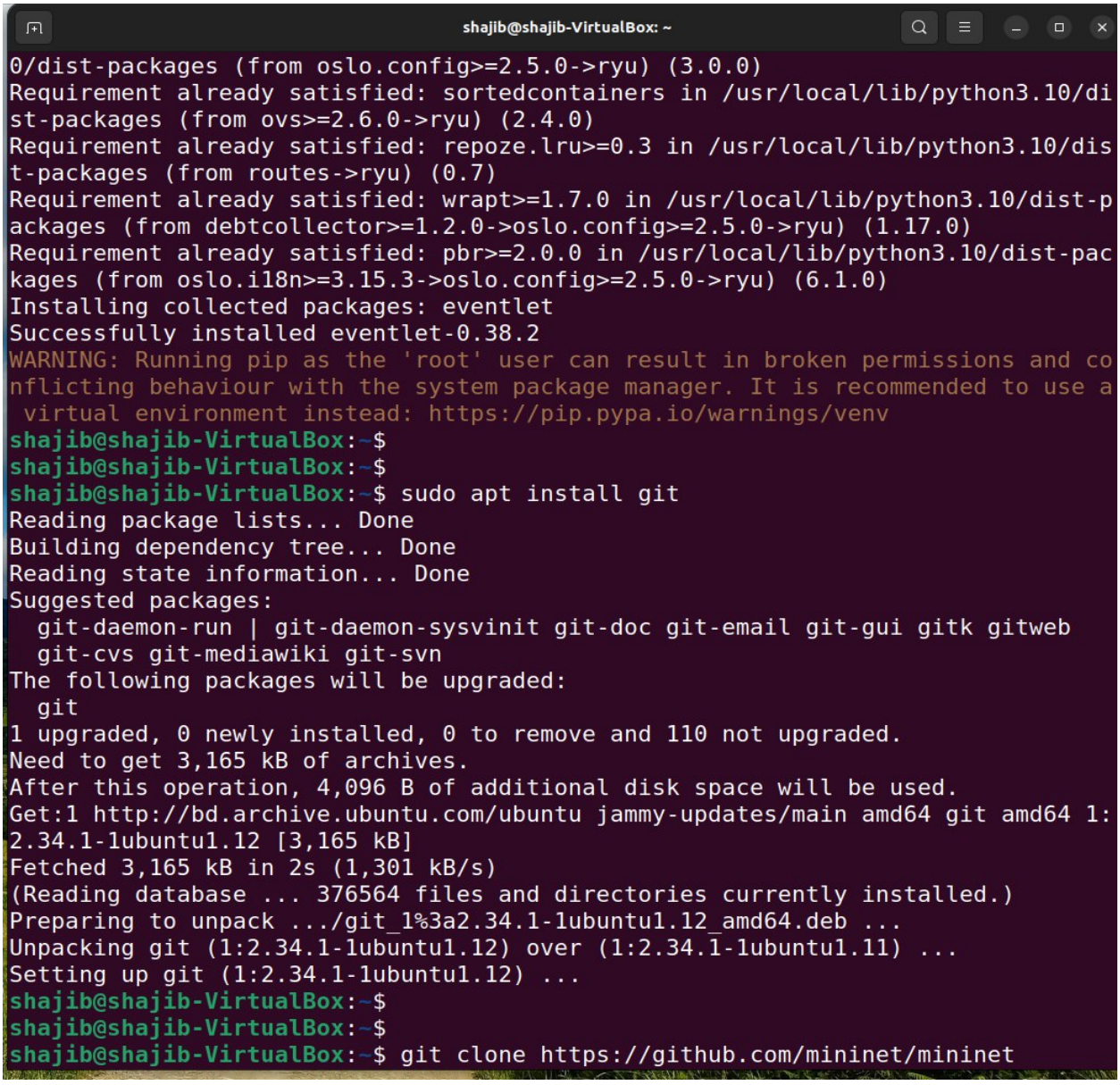


Figure: Install git and clone mininet from github

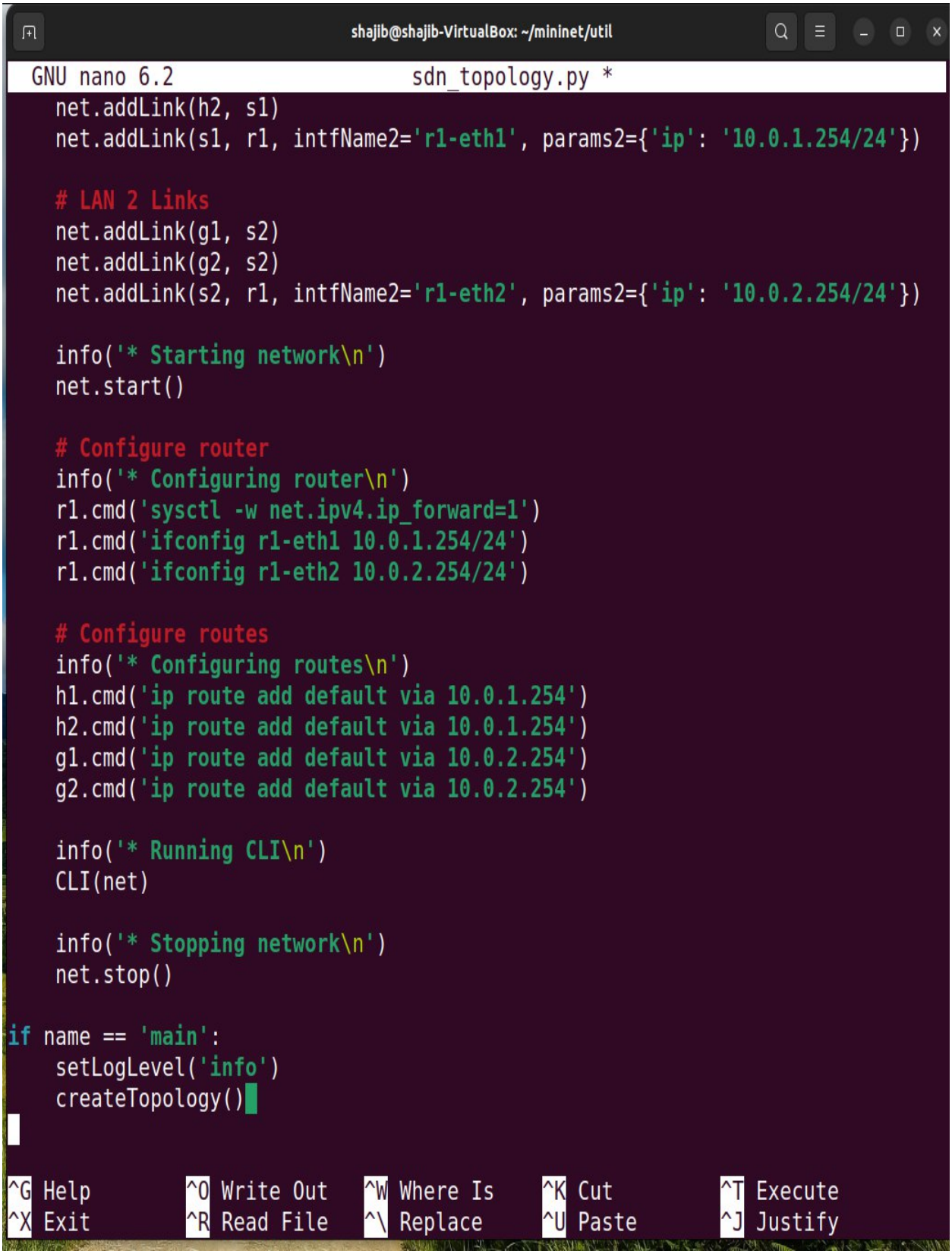


Figure: Write python code on sdn_topology.py

6. Result Analysis

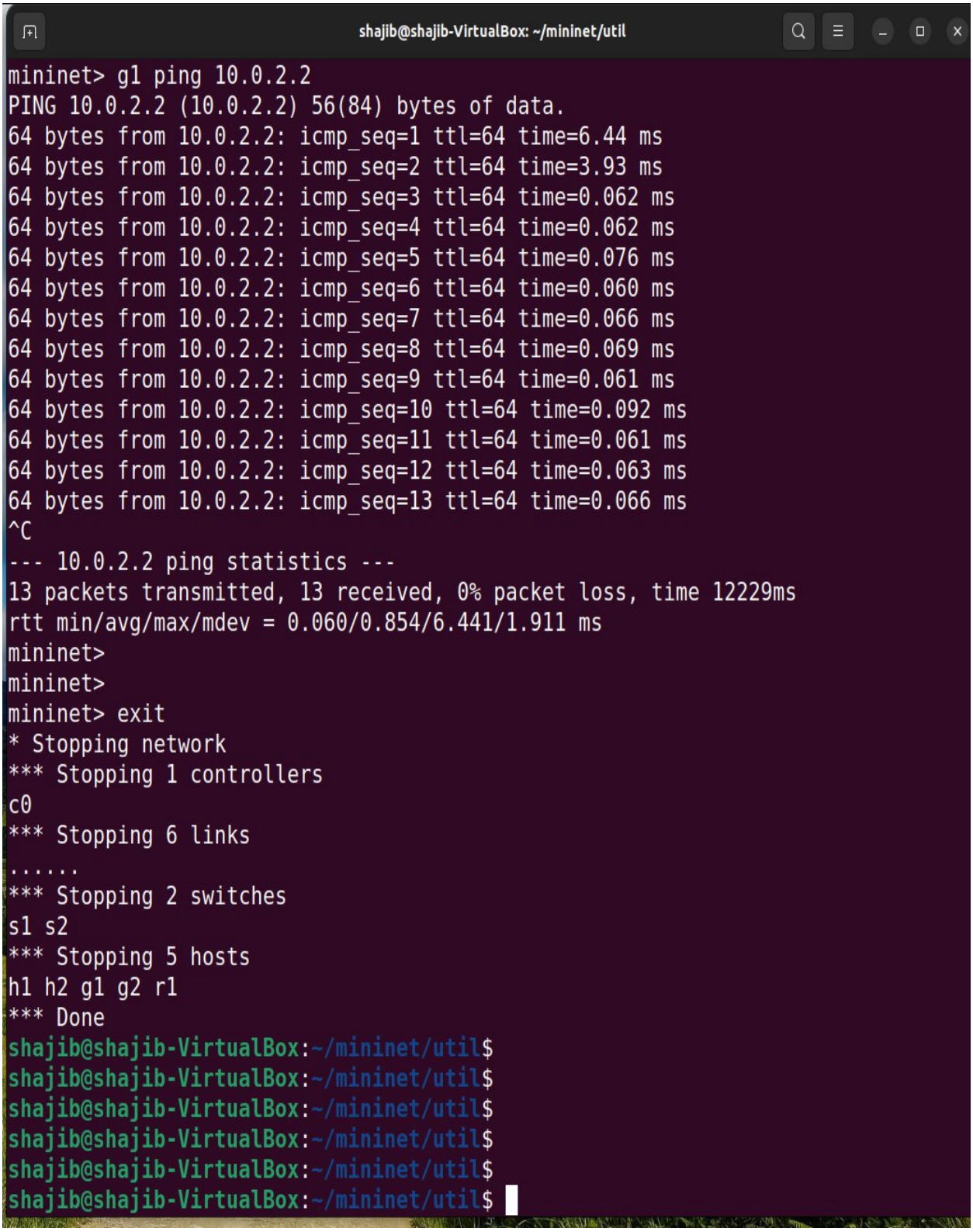


Figure: Transfer data from one LAN to another LAN

Name of Experiment: P2P Topology using NS3.

1. Introduction

In this experiment, we explore the implementation of a Point-to-Point (P2P) network topology using Network Simulation-3 (NS-3) on an Ubuntu system. The primary objective is to simulate a simple network consisting of two nodes connected by a dedicated link with specific bandwidth and delay characteristics. NS-3 is a widely-used open-source discrete-event network simulator that provides robust capabilities for network simulation, analysis, and visualization.

2. Objectives

- To simulate a P2P network topology connecting two nodes with a dedicated 50 Mbps bandwidth and a 5ms delay link using NS-3.
- To configure and install the Internet Protocol (IP) stack on the nodes and assign appropriate IP addresses for communication.
- To set up and demonstrate a UDP Echo Server on the first node and a UDP Echo Client on the second node for testing packet transmission and reception.
- To transmit 10 UDP echo packets with a size of 512 bytes from the client to the server at an interval of 1 second.
- To collect network trace data and analyze the performance of the P2P network based on packet delivery and delay metrics.
- To visualize the network communication and node interactions using NetAnim for better understanding and analysis.

3. Setup the Environment

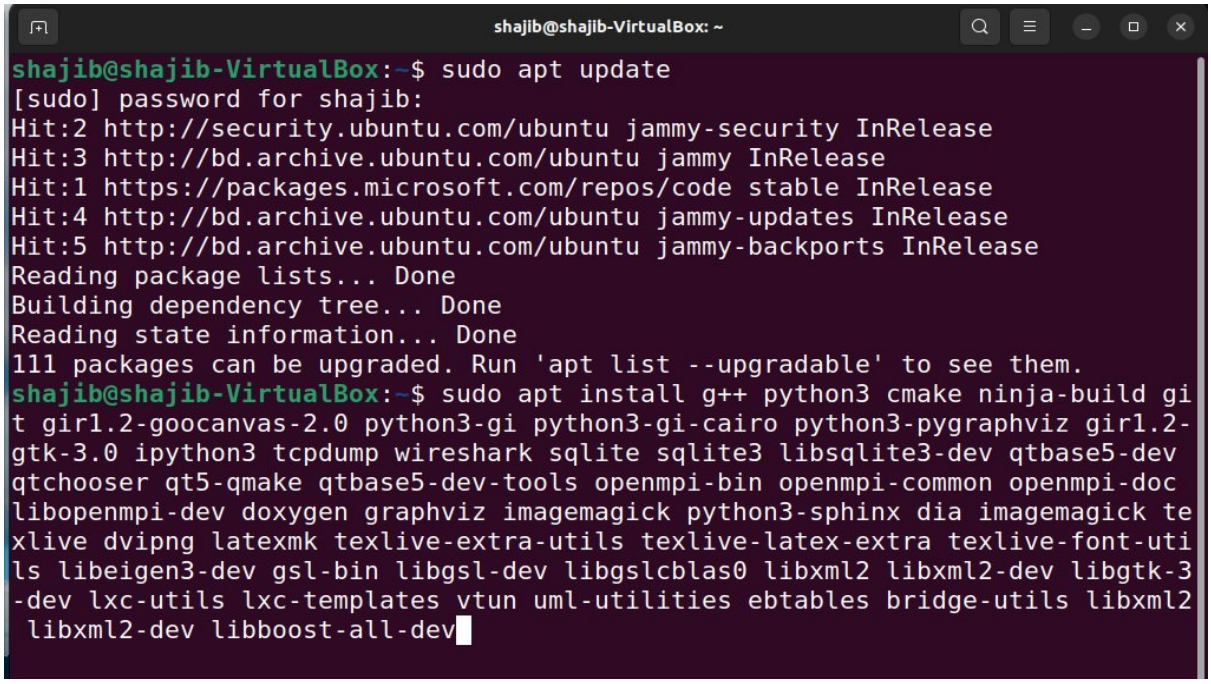


Figure: NS3 installation on Ubuntu 22.04.3 LTS

4. Source Code (C++)

```
#include "ns3/applications-module.h"
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/network-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/netanim-module.h"

using namespace ns3;

NS_LOG_COMPONENT_DEFINE("FirstScriptExample");

int main(int argc, char* argv[]){
   CommandLine cmd(_FILE_);
   cmd.Parse(argc, argv);

   Time::SetResolution(Time::NS);
   LogComponentEnable("UdpEchoClientApplication", LOG_LEVEL_INFO);
   LogComponentEnable("UdpEchoServerApplication", LOG_LEVEL_INFO);

   // Create two nodes
   NodeContainer nodes;
   nodes.Create(2);

   // Configure the Point-to-Point link
   PointToPointHelper pointToPoint;
   pointToPoint.SetDeviceAttribute("DataRate", StringValue("50Mbps"));
   pointToPoint.SetChannelAttribute("Delay", StringValue("5ms"));

   // Set up devices
   NetDeviceContainer devices;
   devices = pointToPoint.Install(nodes);

   // Install the Internet stack
   InternetStackHelper stack;
   stack.Install(nodes);

   // Set base IP address and assign addresses to devices
   Ipv4AddressHelper address;
   address.SetBase("10.1.1.0", "255.255.255.0");
   Ipv4InterfaceContainer interfaces = address.Assign(devices);

   // Configure the UDP Echo Server on node 0
   UdpEchoServerHelper echoServer(8080);
   ApplicationContainer serverApps = echoServer.Install(nodes.Get(0));
   serverApps.Start(Seconds(1.0));
```

```
    serverApps.Stop(Seconds(20.0));


  // Configure the UDP Echo Client on node 1
    UdpEchoClientHelper echoClient(interfaces.GetAddress(0), 8080);
    echoClient.SetAttribute("MaxPackets", UintegerValue(10));
    echoClient.SetAttribute("Interval", TimeValue(Seconds(1.0)));
    echoClient.SetAttribute("PacketSize", UintegerValue(512));

    ApplicationContainer clientApps = echoClient.Install(nodes.Get(1));
    clientApps.Start(Seconds(2.0));
    clientApps.Stop(Seconds(20.0));

    // Enable ASCII tracing
    AsciiTraceHelper ascii;
    pointToPoint.EnableAsciiAll(ascii.CreateFileStream("512.tr"));

    // Enable NetAnim
    AnimationInterface anim("p2p.xml");

    Simulator::Run();
    Simulator::Destroy();

    return 0;
}
```

5. Screenshot of Work Procedure

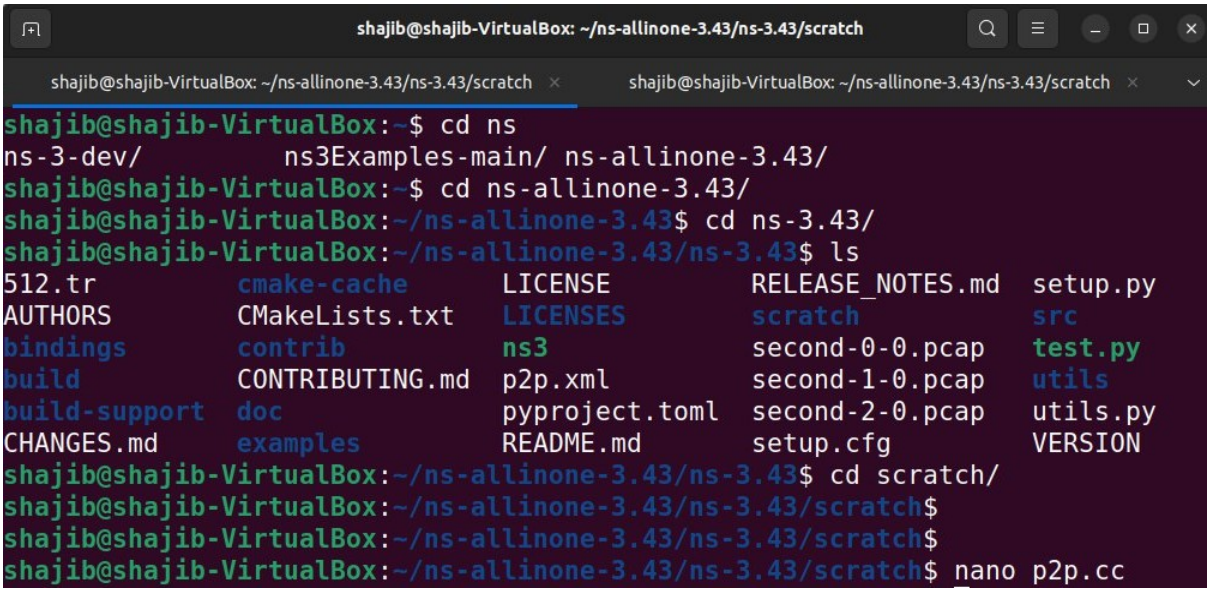


Figure: Create p2p.cc file on scratch folder

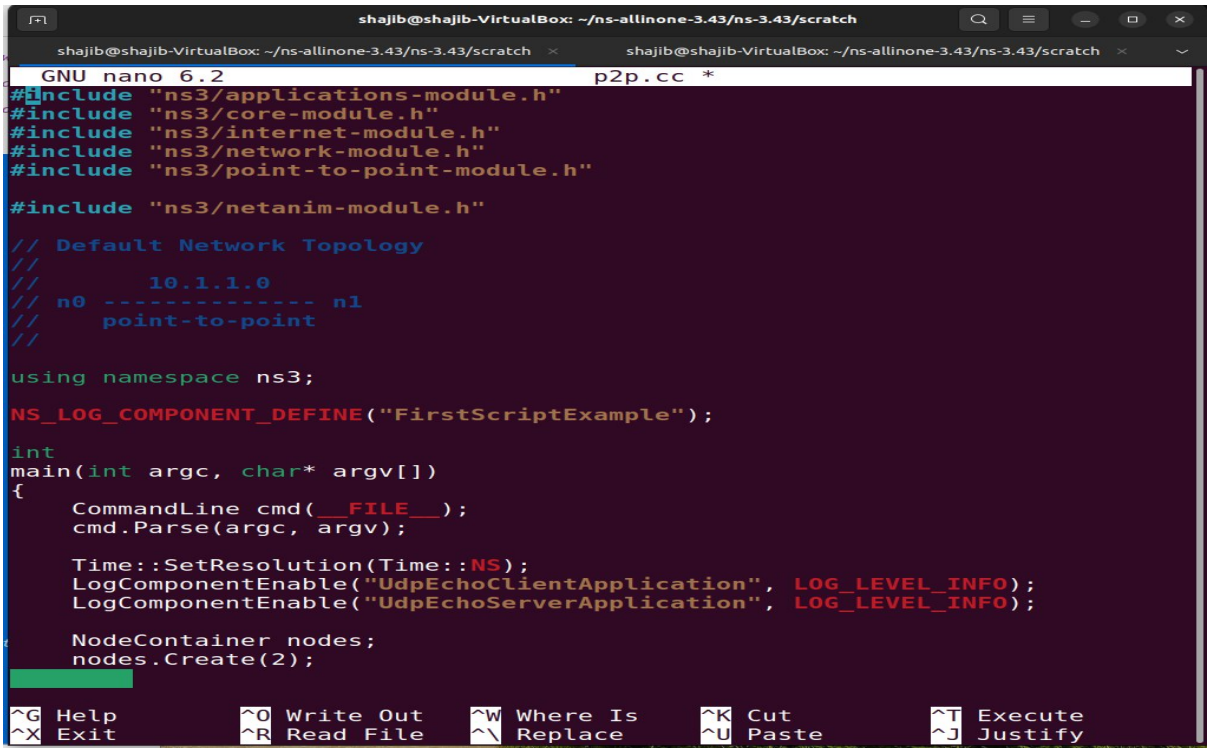


Figure: Write p2p.cc code

6. Result Analysis

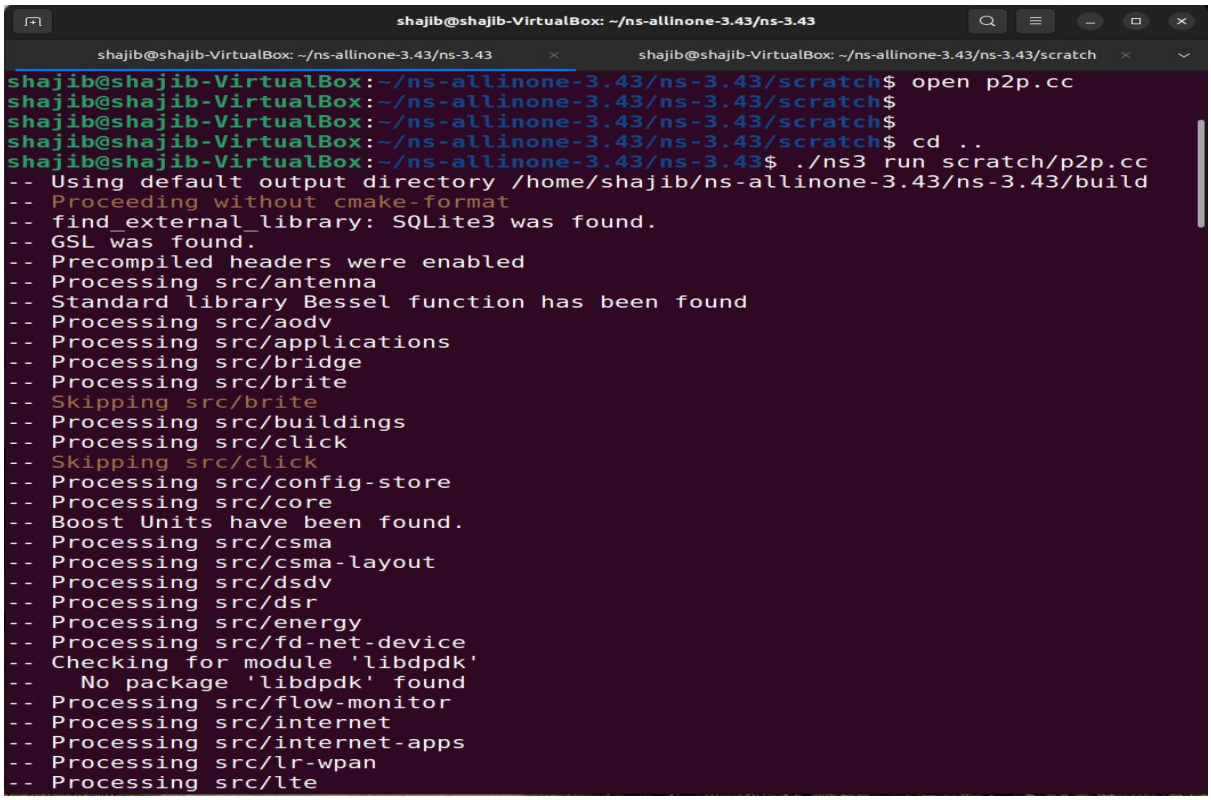


Figure: Run p2p.cc file

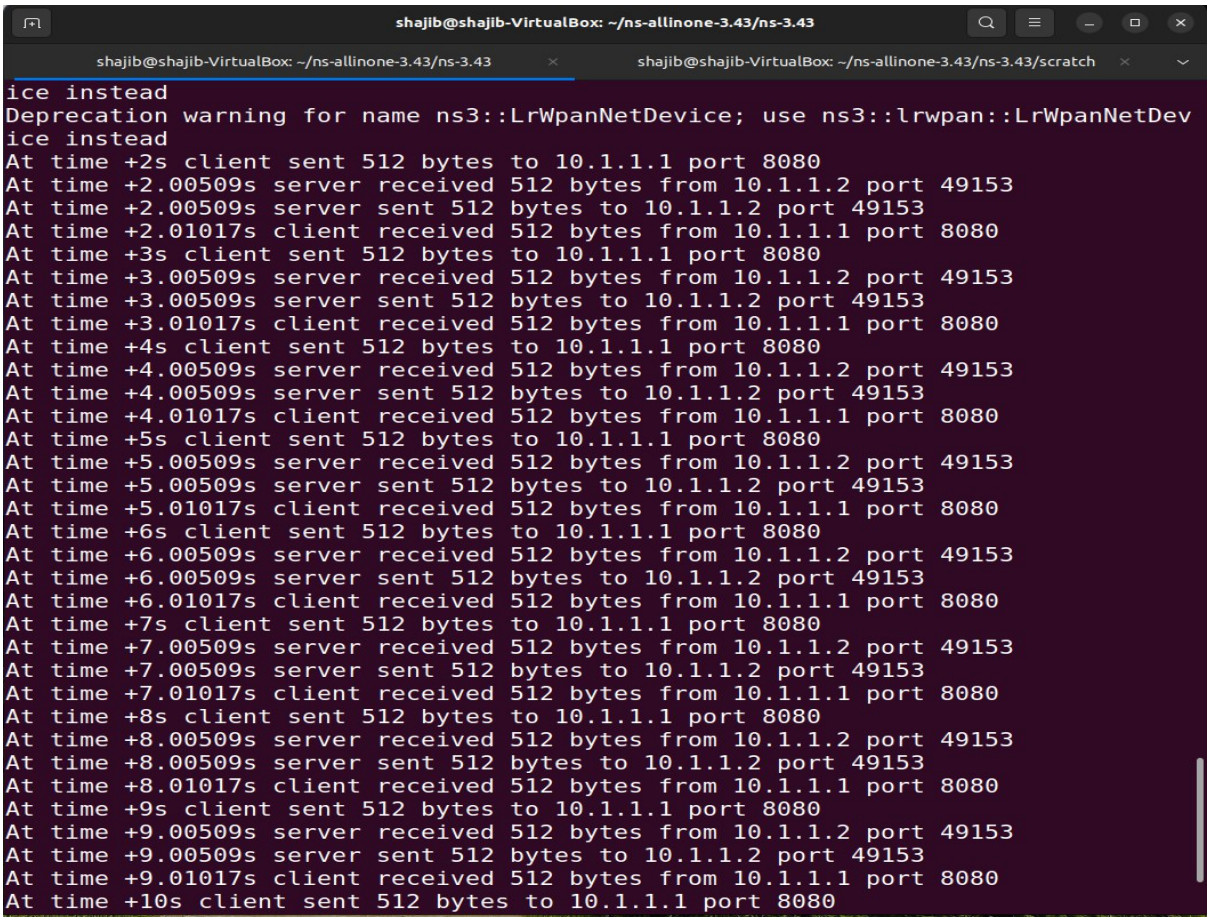


Figure: Transfer data from one node to another

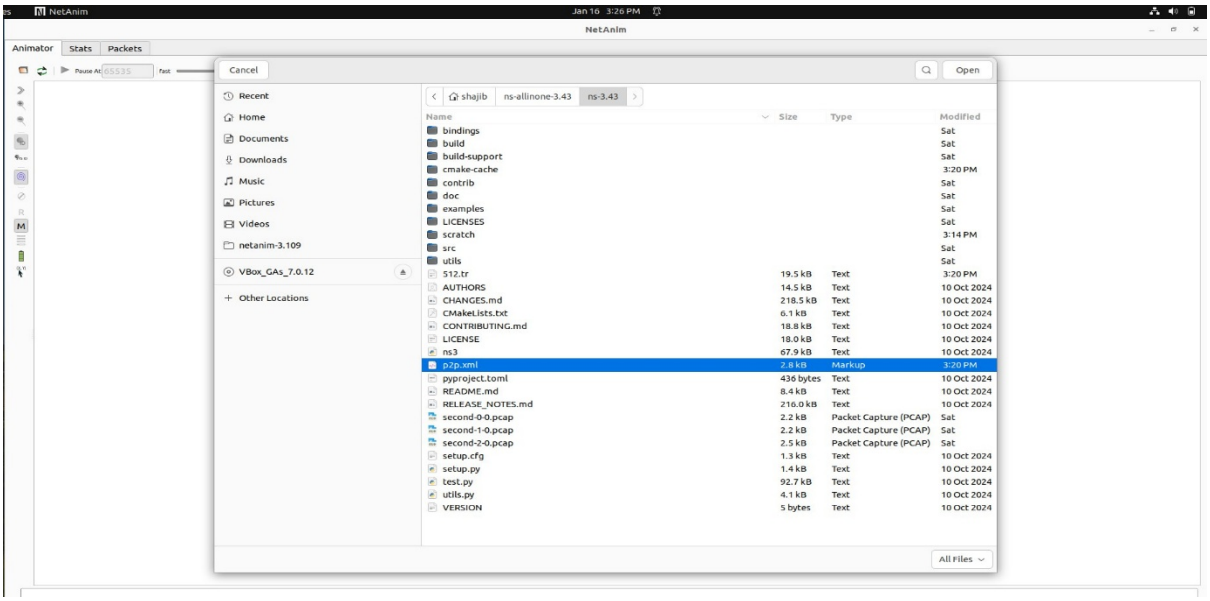
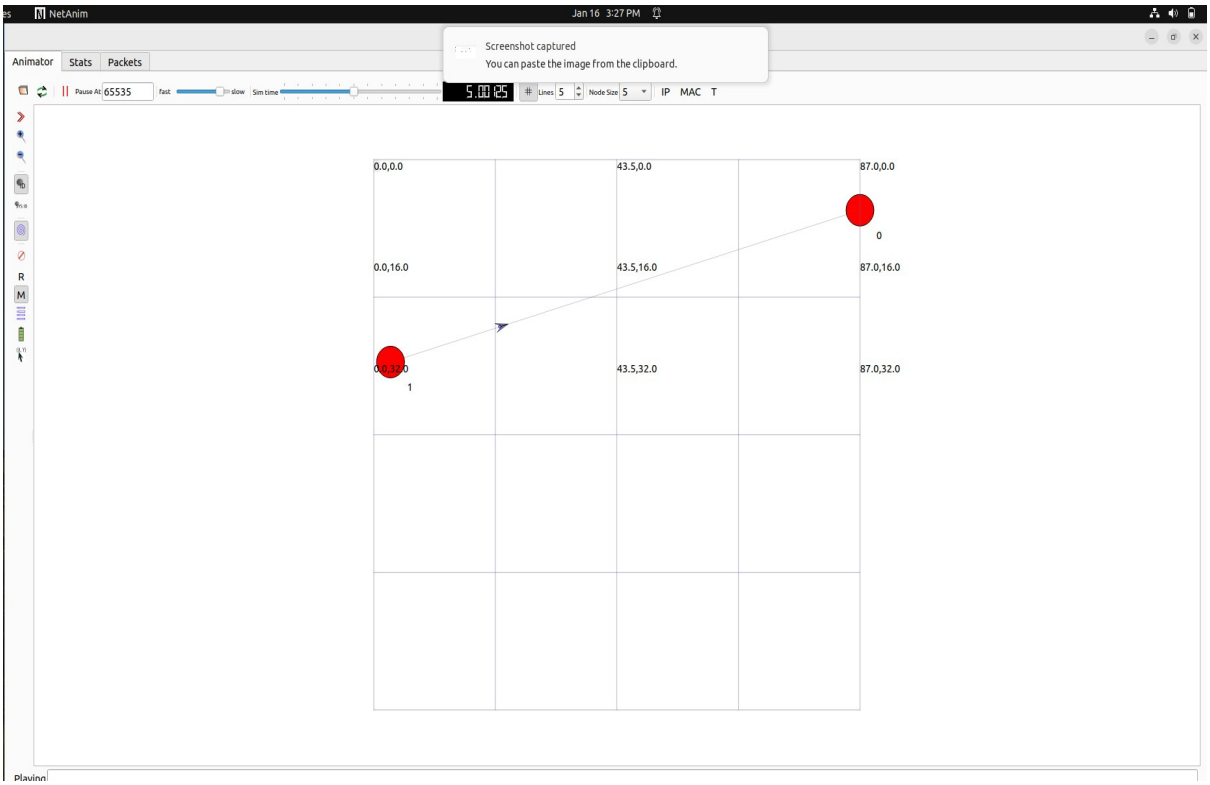


Figure: Open p2p.xml file



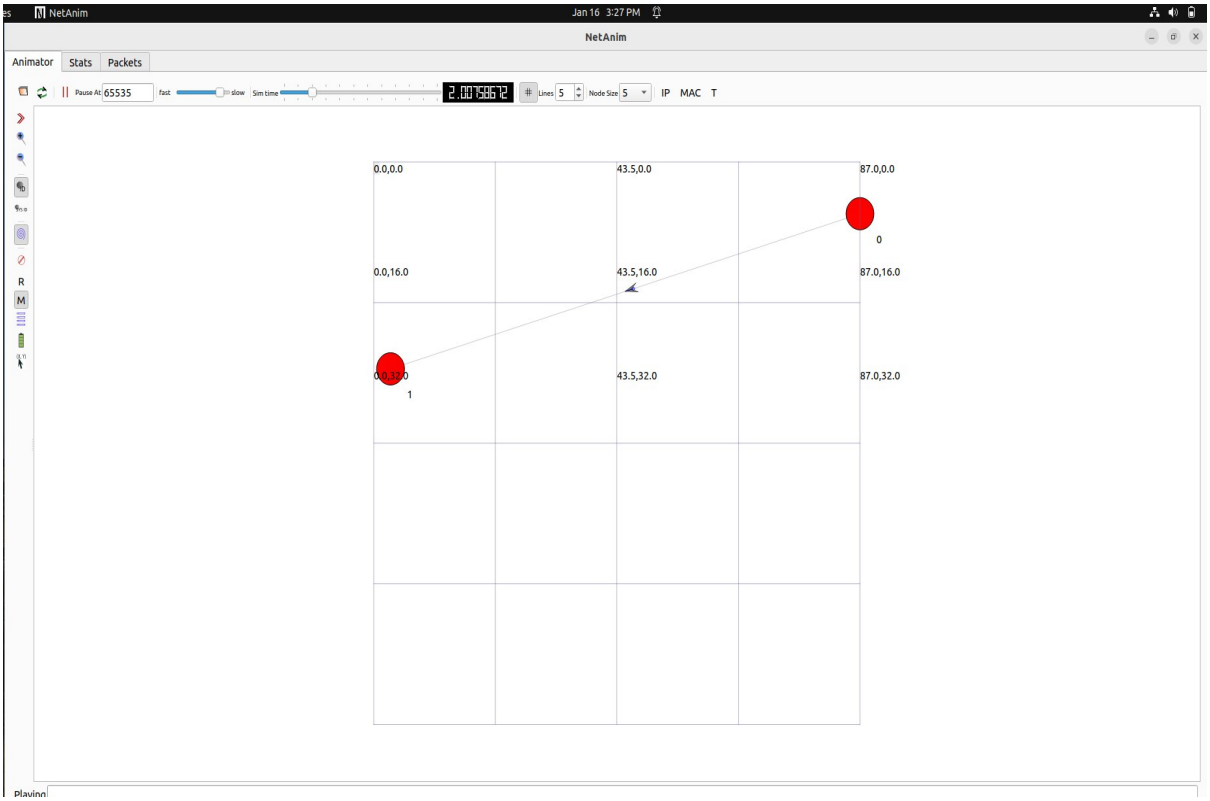


Figure: NetAnim Visualization of P2P Connection

Name of Experiment: WAN Network Creation using NS3.

1. Introduction

In this experiment, we simulate a Wide Area Network (WAN) using the NS-3 simulator to understand the fundamental concepts of interconnecting multiple Local Area Networks (LANs) through routers. This simulation involves creating two LANs, each consisting of two nodes (PCs), connected via two routers that form the backbone of the WAN. The topology aims to mimic real-world scenarios where distant networks communicate over a wider network infrastructure. The simulation demonstrates UDP-based communication between nodes in different LANs, visualized using NetAnim.

2. Objectives

- To simulate a WAN connecting two LANs using routers.
- To implement UDP communication between devices in different networks.
- To understand IP address allocation, routing, and data flow across WANs.
- To analyze the behavior of network communication using NS-3 tools like NetAnim and ASCII trace.
- To understand IP address allocation, routing, and data flow across WANs.
- To analyze the behavior of network communication using NS-3 tools like NetAnim and ASCII trace.

3. Setup the Environment

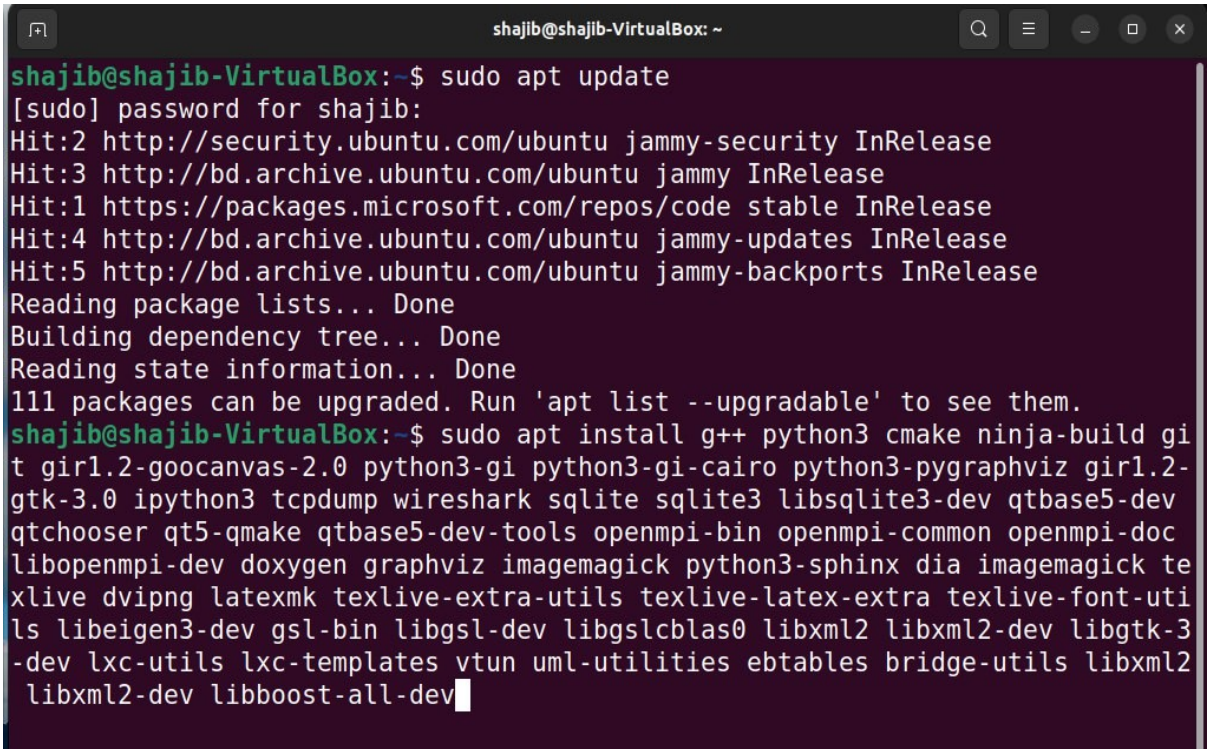


Figure: NS3 installation on Ubuntu 22.04.3 LTS

4. Source Code (C++)

```
#include "ns3/applications-module.h"
#include "ns3/core-module.h"
#include "ns3/internet-module.h"
#include "ns3/network-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/netanim-module.h"

using namespace ns3;
NS_LOG_COMPONENT_DEFINE("WANExample");

int main(int argc, char* argv[]){
   CommandLine cmd(_FILE_);
   cmd.Parse(argc, argv);

   // Set time resolution
   Time::SetResolution(Time::NS);
   LogComponentEnable("UdpEchoClientApplication", LOG_LEVEL_INFO);
   LogComponentEnable("UdpEchoServerApplication", LOG_LEVEL_INFO);

   // Create nodes for two LANs and routers
   NodeContainer lan1Nodes, lan2Nodes, routers;
   lan1Nodes.Create(2); // PCs in LAN 1
   lan2Nodes.Create(2); // PCs in LAN 2
   routers.Create(2);   // Routers: r0 and r1

   // Point-to-Point Helper for LAN 1 (n0 to r0, n1 to r0)
   PointToPointHelper lanHelper;
   lanHelper.SetDeviceAttribute("DataRate", StringValue("50Mbps"));
   lanHelper.SetChannelAttribute("Delay", StringValue("10ms"));

   NetDeviceContainer lan1Devices;
   lan1Devices.Add(lanHelper.Install(lan1Nodes.Get(0), routers.Get(0))); // n0 to r0
   lan1Devices.Add(lanHelper.Install(lan1Nodes.Get(1), routers.Get(0))); // n1 to r0

   // Point-to-Point Helper for LAN 2 (n2 to r1, n3 to r1)
   NetDeviceContainer lan2Devices;
   lan2Devices.Add(lanHelper.Install(lan2Nodes.Get(0), routers.Get(1))); // n2 to r1
   lan2Devices.Add(lanHelper.Install(lan2Nodes.Get(1), routers.Get(1))); // n3 to r1

   // Point-to-Point Helper for inter-router link (r0 to r1)
   PointToPointHelper routerLinkHelper;
   routerLinkHelper.SetDeviceAttribute("DataRate", StringValue("100Mbps"));
   routerLinkHelper.SetChannelAttribute("Delay", StringValue("20ms"));

   NetDeviceContainer routerDevices;
```

```
    routerDevices.Add(routerLinkHelper.Install(routers.Get(0), routers.Get(1))); // r0 to r1

    // Install Internet Stack on all nodes and routers
    InternetStackHelper stack;
    stack.Install(lan1Nodes);
    stack.Install(lan2Nodes);
    stack.Install(routers);

    // Assign IP addresses to all devices
    Ipv4AddressHelper address;

    // Assign IP to LAN 1
    address.SetBase("10.1.1.0", "255.255.255.0");
    Ipv4InterfaceContainer lan1Interfaces = address.Assign(lan1Devices);

    // Assign IP to LAN 2
    address.SetBase("10.1.2.0", "255.255.255.0");
    Ipv4InterfaceContainer lan2Interfaces = address.Assign(lan2Devices);

    // Assign IP to inter-router link
    address.SetBase("10.1.3.0", "255.255.255.0");
    Ipv4InterfaceContainer routerInterfaces = address.Assign(routerDevices);

    // Create UDP Echo Server on node n0 in LAN 1 (PC in LAN 1)
    UdpEchoServerHelper echoServer(8080); // Port 8080
    ApplicationContainer serverApps = echoServer.Install(lan1Nodes.Get(0)); // n0 in LAN
1
    serverApps.Start(Seconds(1.0));
    serverApps.Stop(Seconds(20.0));

    // Create UDP Echo Client on node n3 in LAN 2 for continuous packet transfer
    UdpEchoClientHelper echoClient(lan1Interfaces.GetAddress(0), 8080);
    echoClient.SetAttribute("MaxPackets", UintegerValue(100));
    echoClient.SetAttribute("Interval", TimeValue(Seconds(0.1)));
    echoClient.SetAttribute("PacketSize", UintegerValue(512));

    ApplicationContainer clientApps = echoClient.Install(lan2Nodes.Get(1)); // n3 in LAN 2
    clientApps.Start(Seconds(2.0)); // Start after 2 seconds
    clientApps.Stop(Seconds(20.0)); // Stop after 20 seconds

    // Enable ASCII trace for debugging
    AsciiTraceHelper ascii;
    lanHelper.EnableAsciiAll(ascii.CreateFileStream("wan.tr"));

    // Enable NetAnim for visualization
    AnimationInterface anim("wan.xml");
```

```
    // Set node positions for visualization
    anim.SetConstantPosition(lan1Nodes.Get(0), 10.0, 10.0); // PC1 (n0)
    anim.SetConstantPosition(lan1Nodes.Get(1), 10.0, 50.0); // PC2 (n1)
    anim.SetConstantPosition(lan2Nodes.Get(0), 80.0, 10.0); // PC3 (n2)
    anim.SetConstantPosition(lan2Nodes.Get(1), 80.0, 50.0); // PC4 (n3)
    anim.SetConstantPosition(routers.Get(0), 20.0, 30.0);    // Router r0
    anim.SetConstantPosition(routers.Get(1), 70.0, 30.0);    // Router r1

    // Label nodes in the animation
    anim.UpdateNodeDescription(lan1Nodes.Get(0), "PC 1 (n0)");
    anim.UpdateNodeDescription(lan1Nodes.Get(1), "PC 2 (n1)");
    anim.UpdateNodeDescription(lan2Nodes.Get(0), "PC 3 (n2)");
    anim.UpdateNodeDescription(lan2Nodes.Get(1), "PC 4 (n3)");
    anim.UpdateNodeDescription(routers.Get(0), "Router r0");
    anim.UpdateNodeDescription(routers.Get(1), "Router r1");

    // Start the simulation
    Simulator::Run();
    Simulator::Destroy();
    return 0;
}
```

5. Screenshot of Work Procedure

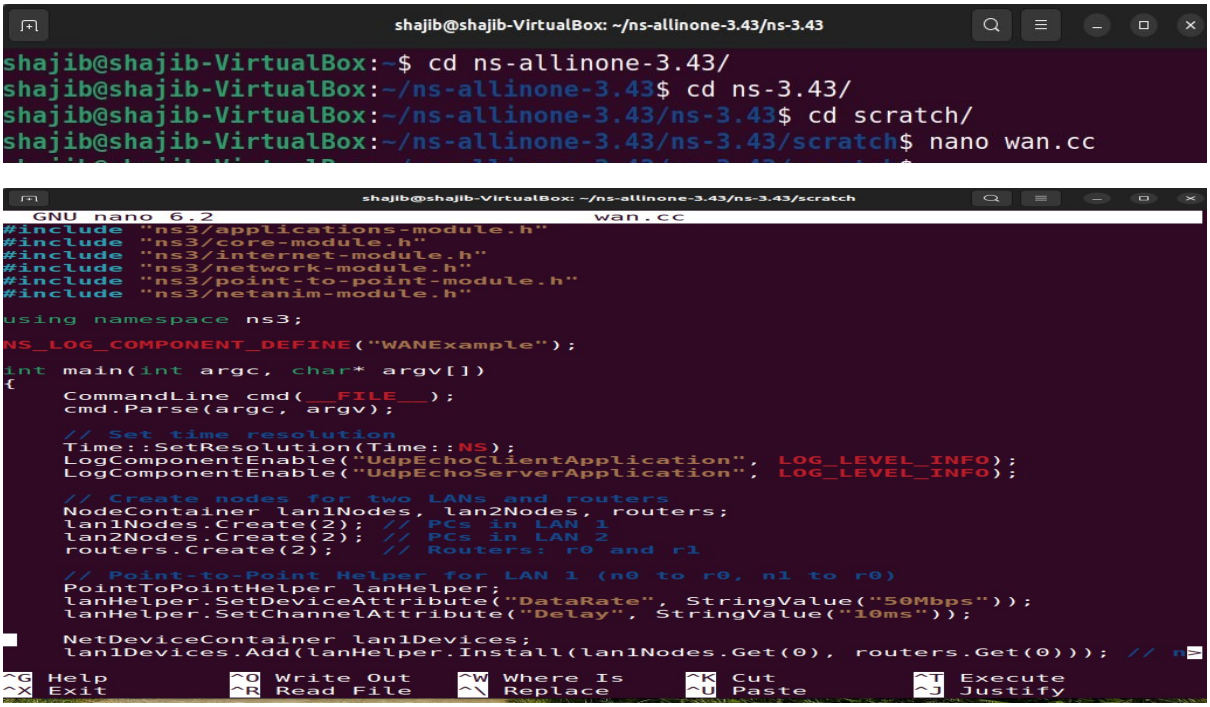


Figure: Write wan.cc file

6. Result Analysis

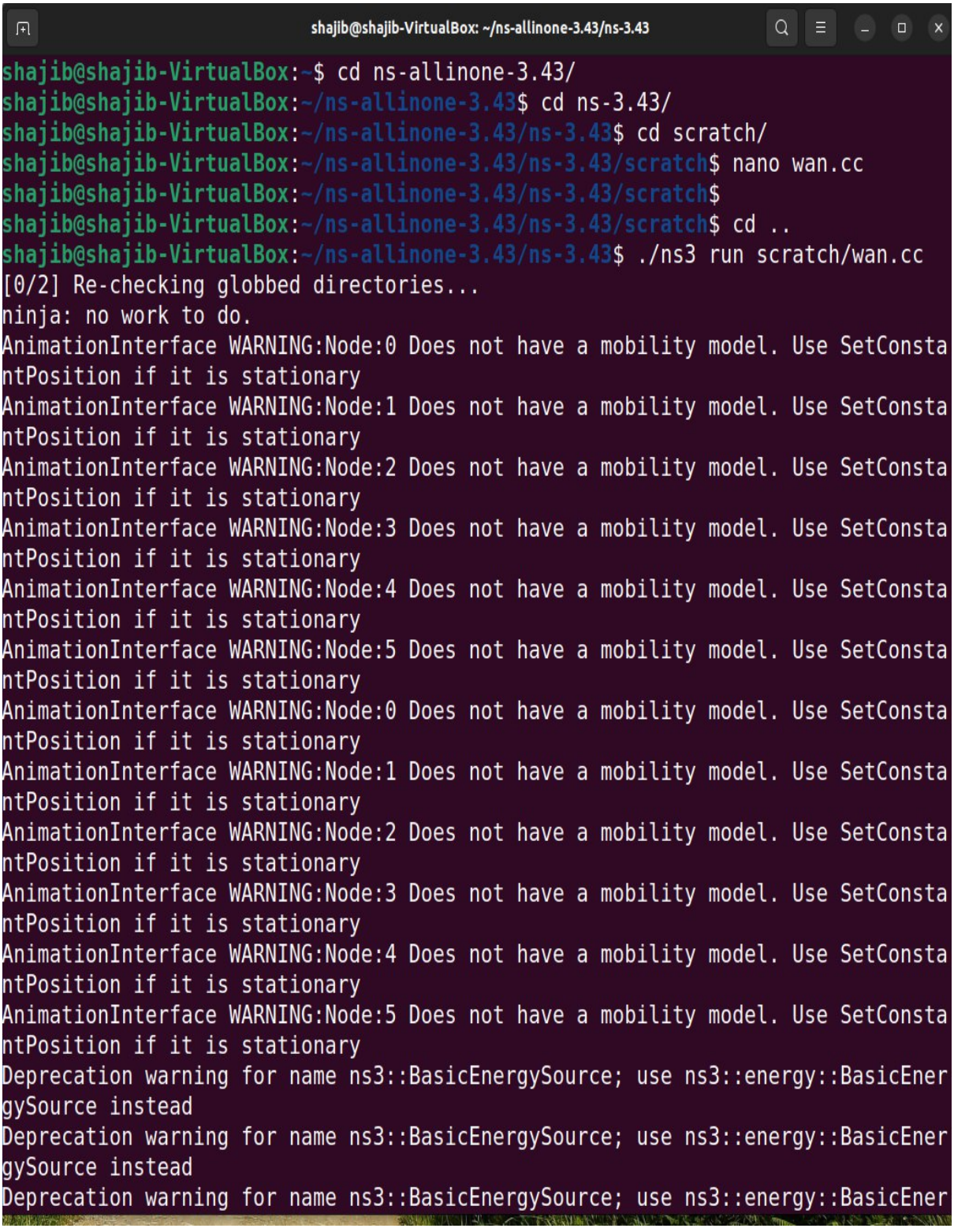


Figure: Run wan.cc file

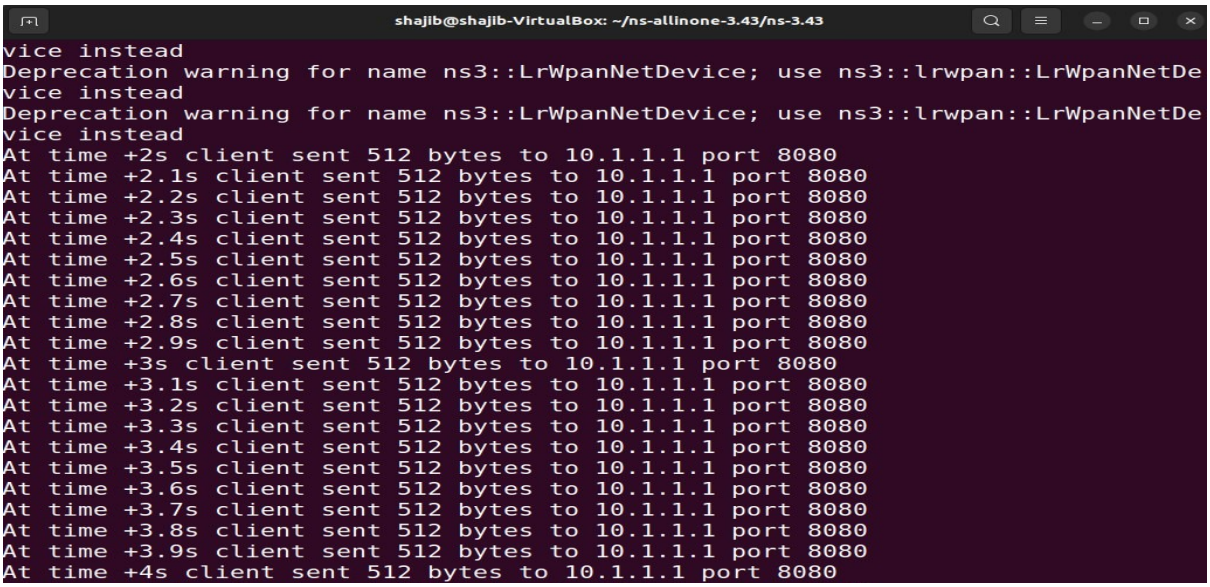


Figure: Transfer data from one LAN to another LAN

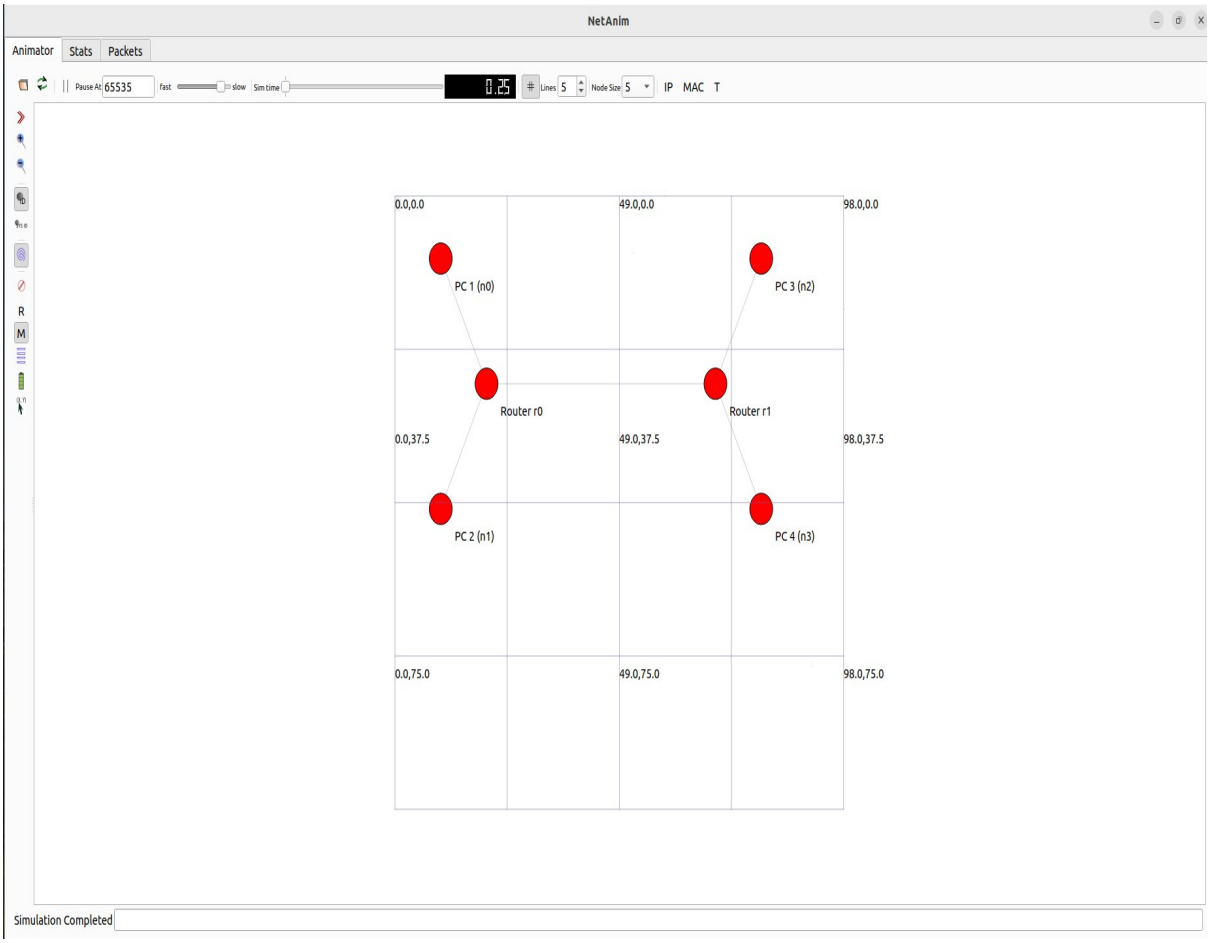


Figure: NetAnim Visualization of WAN Network

References

1. **Cisco Packet Tracer, Cisco Networking Academy**
   https://www.netacad.com/courses/packet-tracer

2. **Mininet from GitHub**
   https://github.com/mininet/mininet

3. **Network Simulator 3 (NS3)**
   https://www.nsnam.org/