Premier University

Department of Computer Science and Engineering

CSE 368 - Computer Networks Laboratory

LABORATORY MANUAL

July 2024

CONTENTS

1. Introduction to different types of networking cables

Title: Fabrication and Testing of LAN Cable.

Objective:

1. Learn about cables, different types of cable and equipment.
2. Learn about when which types of cables are needed.
3. Fabricate a LAN cable for internet connection.
4. Test if the cable is correctly working.

Prerequisite: None

Theory:

Figure 1.1: LAN Cable

LAN Cable:
A LAN cable is a conductor that connects devices in a Local Area Network (LAN) with a network connector**.**

CAT 4, 5, 6:
Cat 4, Cat 5, and Cat 6 are different categories of Ethernet cables used for networking purposes. The main difference lies in their performance capabilities and the level of data transmission they can support.

- Cat 4 cables were commonly used in the past but are now considered outdated. They can support data transmission speeds of up to 16 Mbps.
- Cat 5 cables can handle higher data transmission speeds of up to 100 Mbps.
- Cat 6 cables are the most advanced among the three. They can support data transmission speeds of up to 10 GB p/s over short distances. Cat 6 cables have better shielding and reduced crosstalk, resulting in improved performance and less interference.

RJ-45 Connector:
A registered jack (RJ) is a standardized physical network interface for connecting telecommunications or data equipment. The most common twisted-pair connector is an 8-position, 8-contact (8P8C) modular plug and jack commonly referred to as an RJ45 connector.

Figure 1.2: RJ-45 Connector

Straight Through Cable:

A straight-through cable is a type of Ethernet cable used to connect different types of devices within a network.

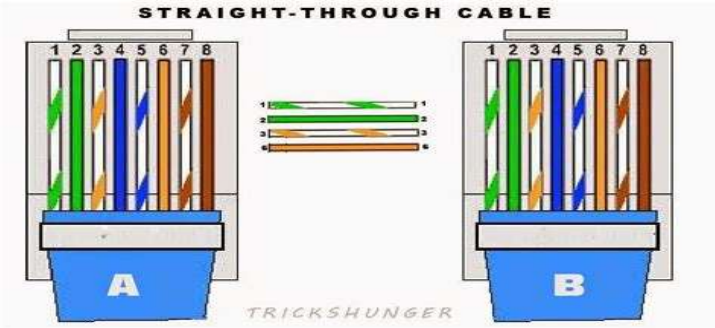


Figure 1.3: Straight-Through Cable Order Diagram

Crossover Cable:

A crossover cable is a type of Ethernet cable used to connect similar devices directly to each other without needing an intermediate network device like a switch or hub.

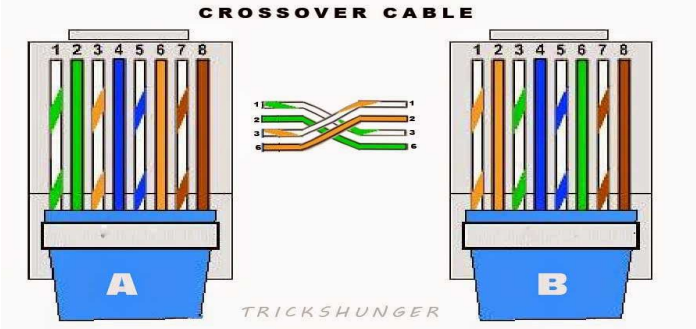


Figure 1.4: Crossover Cable Order Diagram

Methodology:

1. Cut into the plastic sheath about **one inch** from the end of the cut cable.

2. Pinch the wires between fingers and straighten them out as needed (Straight through/ Crossover) the color order is important to get correct.

3. Make a straight cut across the eight wires to shorten them to **half an inch** from the cut sleeve to the end of the wires.

4. Push all eight unstripped colored wires into the connector.

5. Carefully place the connector into the Ethernet crimper and cinch down on the handles tightly. The copper splicing tabs on the connector will pierce into each of the eight wires. There's also a locking tab that holds the blue plastic sleeve in place for a tight compression fit.

6. Test the cable using the cable tester.

Equipment:

1. CAT5e or CAT6 cable
2. RJ45 connectors
3. Crimping tool
4. Cutter/stripper
5. Cable tester

Procedure:

- Unwind the cable.

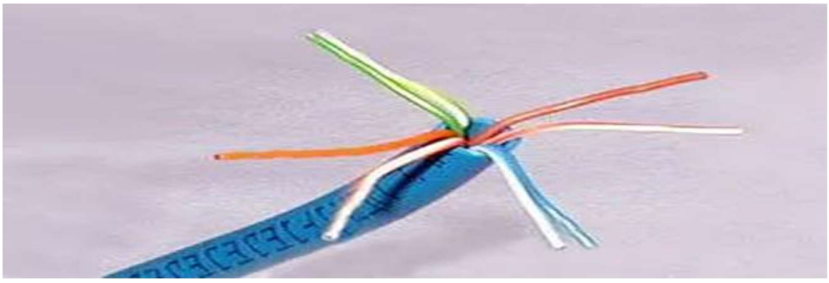


Figure 1.5: Unwind cable.

- Cut into same length while following the color scheme of straight through/ crossover.

Figure 1.6: Cable cut into same length

- Insert cable into the RJ-45 connector. And check if all 8 wire's end are visible in the connector's front side.

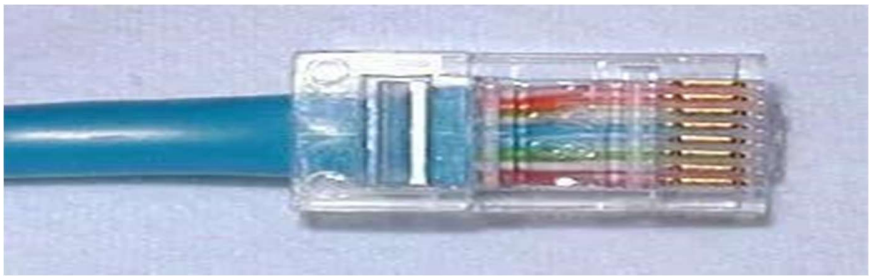


Figure 1.7: Cable inside the RJ-45 connector.

- Put the RJ-45 connector inside the clamper and carefully put pressure in the clamper. This will lock the cable with the connector permanently.

Figure 1.8: Clamping the cable.

- Now put each end of the wire into the 2 ports of the cable tester. The tester will blink lights that resembles the pattern shown in the diagram of **Figure 3, 4**. Check if the pattern matches according to the diagram.

Figure 1.9: Testing the cable.

Observation and Result:

The cable tester as shown in **Figure 9** will blink lights according to the diagram of the straight through/ crossover cable order/pattern.

Additional reading materials/ online tutorial/ References:

1. Video Tutorial: https://www.youtube.com/watch?v=Uw8FSXx4dnU
2. Article: https://www.wikihow.com/Create-an-Ethernet-Cable

2. Introduction to Cisco Packet Tracer.

Title: Introduction to CISCO Packet Tracer.

Objective:

1. Familiarize with CISCO Packet Tracer.
2. Learn basic functionality of CISCO Packet Tracer.
3. Use CISCO Packet Tracer for building and configuring a basic network.
4. Check connectivity of the network.

Prerequisite:

1. Knows basic equipment/ devices (i.e. End devices, Switch and different types of cable) and their functionality.
2. Understanding of networking, networking structure and terminology.

Theory:

IP Address:

An Internet Protocol address is a numerical label such as 192.0.2.1 that is assigned to a device connected to a computer network that uses the Internet Protocol for communication. IP addresses serve two main functions: network interface identification, and location addressing.

Gateway:

A gateway is a network node or device that connects two networks that use different transmission protocols. Gateways play an important role in connecting two networks. It works as the entry-exit point for a network because all traffic that passes across the networks must pass through the gateway.

Methodology:

1. Learn about menus, sub-menus.
2. Make a basic network.
3. Configure the network
4. Check connectivity.

Equipment:

1. PC
2. Switch (2960-24TT)
3. Wire

Procedure:

Step 1: Familiarize with CISCO Packet Tracer

- This illustrates Cisco Packet Tracer. Cisco Packet Tracer's graphical interface makes it easy to visualize network connections and understand how data flows between devices.

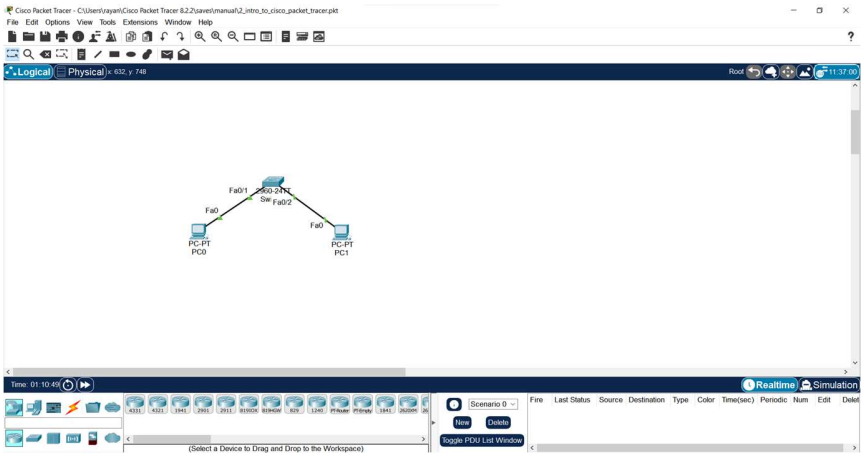


Figure 2.1: Cisco Packet Tracer Interface.

- Here in top there is the toolbar that provides quick access to various menus related to file and customization. Also there is PDU (Packet), drawing and note option available.

Figure 2.2: Cisco Packet Tracer Interface - Main Toolbar.

- Here in bottom there is the bottom toolbar that provides quick access to various tools and options for network design and simulation.

Figure 2.3: Cisco Packet Tracer Interface: Bottom Toolbar.

- In this menu every kind of components/equipment that are needed for building a network is available. Here different types of networking devices such as computer/communication devices, IOT devices, wire, miscellaneous and multiuser connection are available.

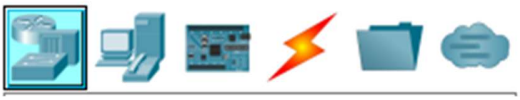


Figure 2.4: Types of component available in Cisco Packet Tracer.

- While selecting the submenus there will be component of verity model available.

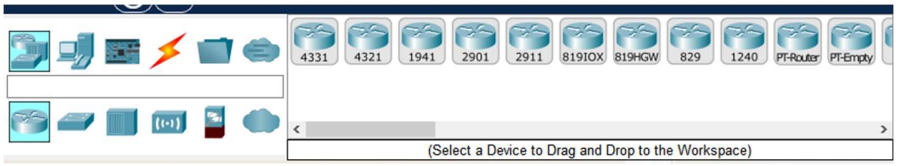


Figure 2.5: Different type of Router.

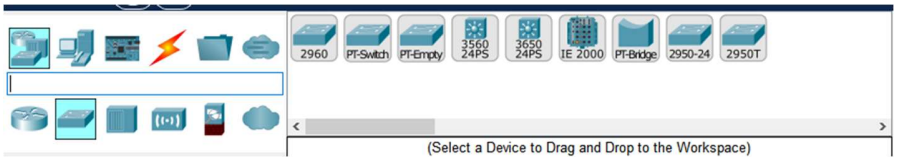


Figure 2.6: Different types of Switch.

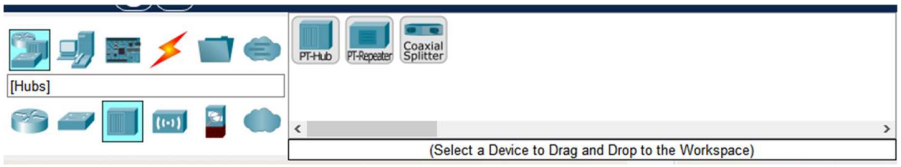


Figure 2.7: Different types of Hub.

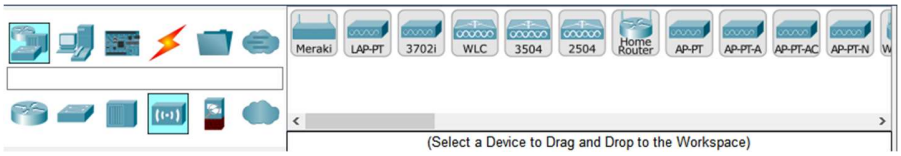


Figure 2.8: Different types of Wireless Devices.

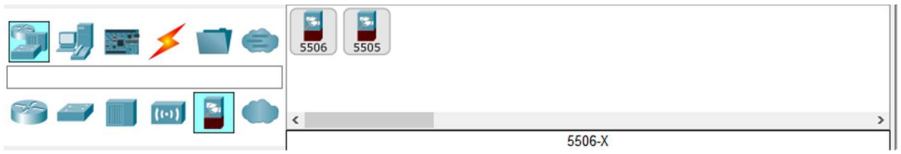


Figure 2.9: Different types of security components.

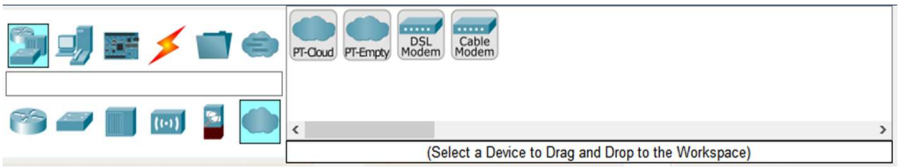


Figure 2.10: Different types of WAN Emulation.

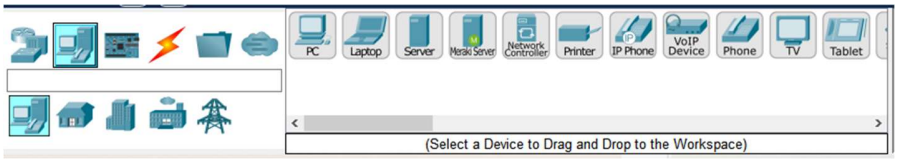


Figure 2.11: Different types of End Devices.

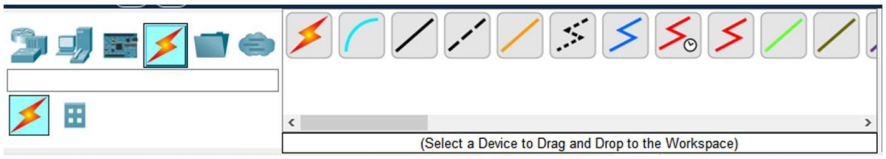


Figure 2.12: Different types of cable.

This option from the cable menu helps to connect any devices with the type of wire those devices needed automatically.

User Have to select the required components (By left mouse click) if user need more of that component then instead of selecting every times user can just (ctrl + left mouse click) to avoid this problem. When done press ESC from keyboard to dis-select the component.

[N.B: Live demonstrated should be presented in-order to help users understanding how to work with Packet Tracer software.**]**

- Double click on any component will pop open a screen where there will be menus/options for configure the component.

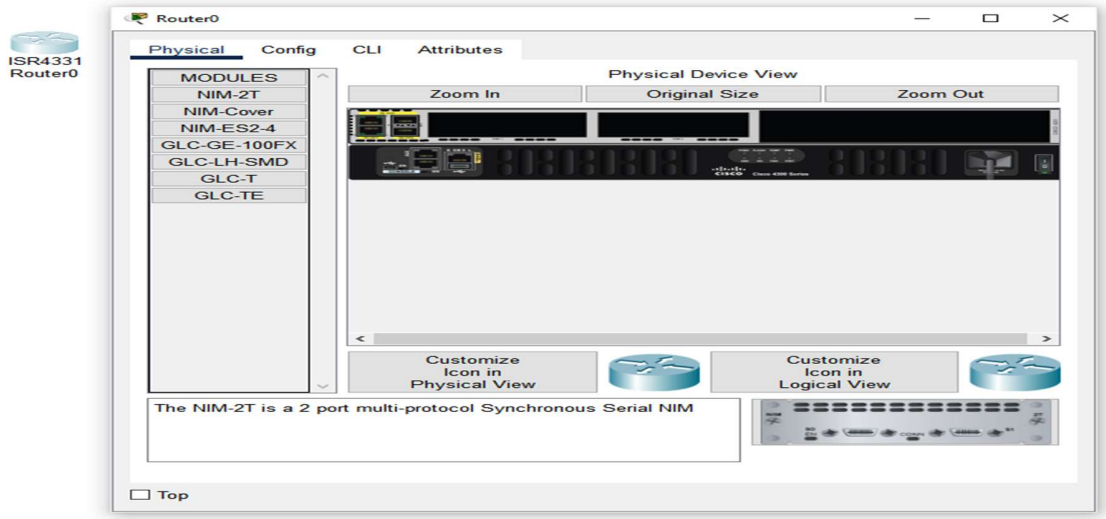


Figure 2.13: View of configuration screen of a component.

Step 2: Make and configure a basic network

- Drag and drop 1 PC,1 Laptop, 1 Switch (2960-24TT) from the bottom toolbar

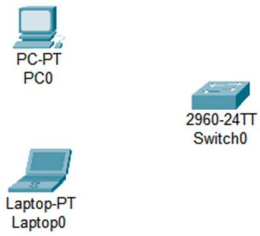


Figure 2.13: PC, Laptop, and Switch in the main window.

- Use the "Automatically Choose Connection Type" as shown in **figure 2.12** from the cable menu from the bottom toolbar. Then Select on 2 of the one by one for connection.

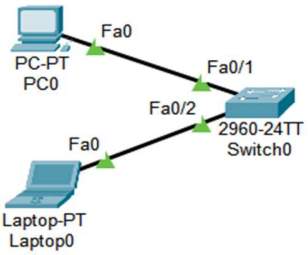


Figure 2.14: Both end devices are connected to the switch via cable.

Step 3: Configure the network

- Double click on end devices (PC, Laptop) will show a pop up screen from there go to desktop

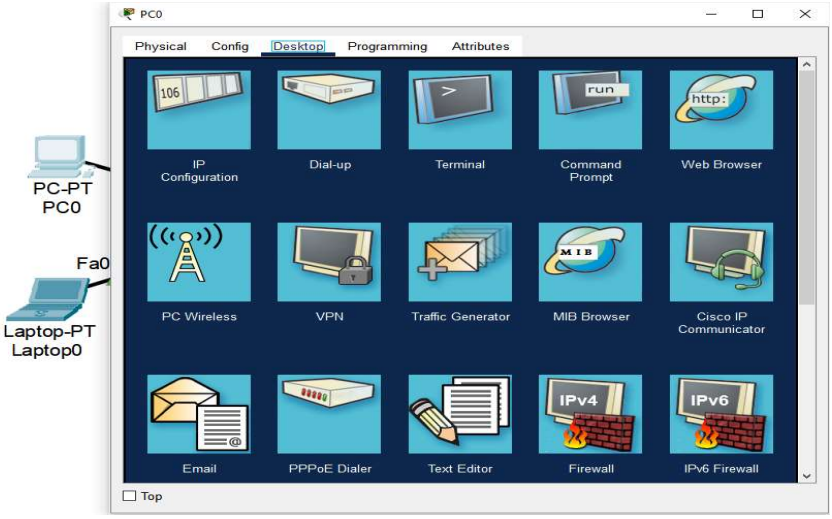


Figure 2.15: End Device configuration menus.

- Select IP Configuration from there.

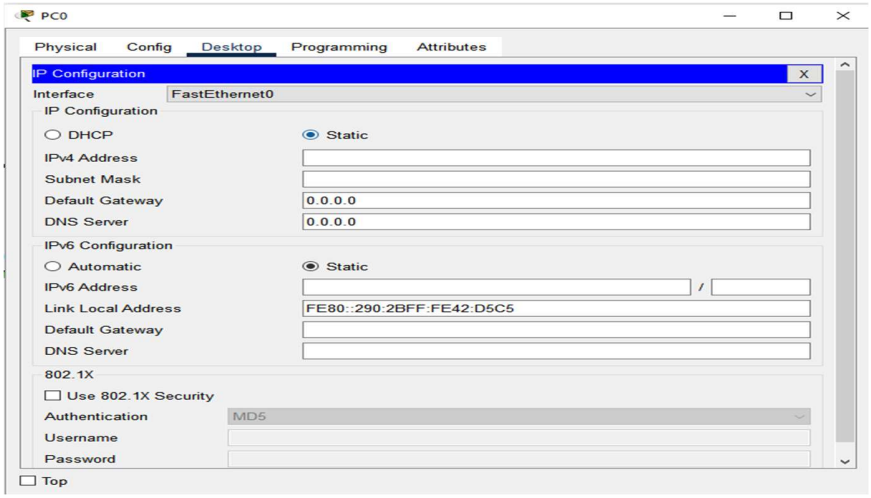


Figure 2.16: IP Configuration menu for PC0

- Set IP and Gateway for the devices.

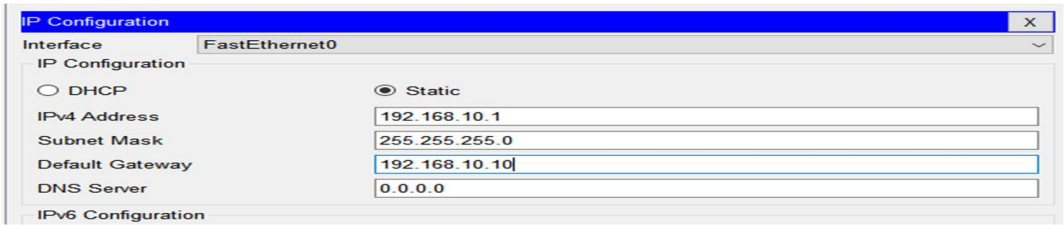


Figure 2.17: IP and Gateway for PC0

- Repeat This process for another End Device (Laptop)

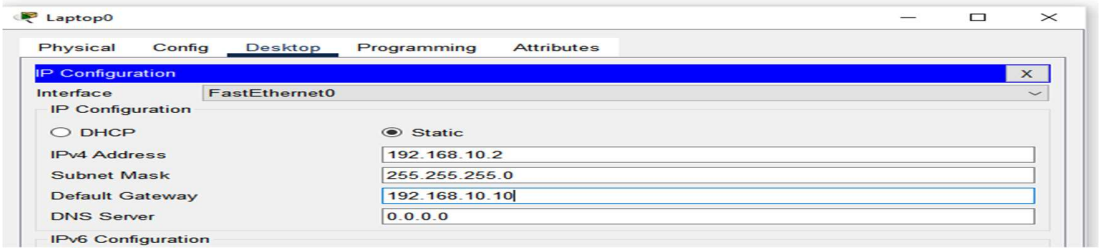


Figure 2.18: IP and Gateway for Laptop0

Step 4: Check Connectivity.

- From the top menu **Figure 2.2** select this icon and click on both 2 End Devices (PC0, Laptop0) source and destination.

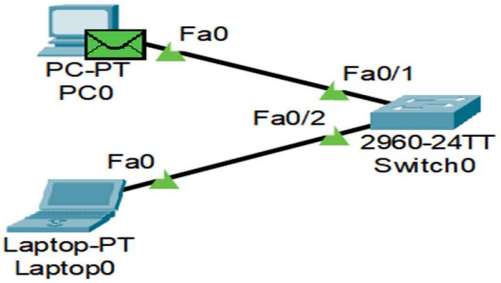


Figure 2.19: Adding simple PPU

- Click Simulation from the bottom toolbar

Figure 2.20: Simulation menu in bottom toolbar

- This will show the Simulation menu in the right side of screen. Where there will be a play button. Click that play button, the packet will transfer/travel from source to destination and every step/action will be visible.

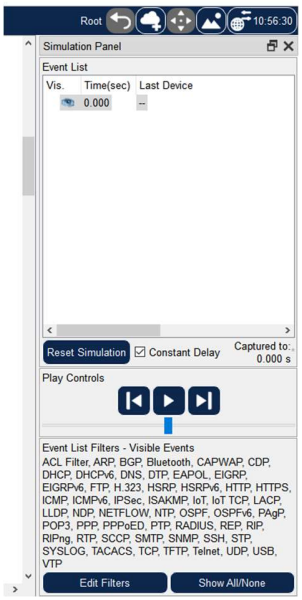


Figure 2.21: Right sidebar for simulation menus.

- If the packet is successfully transferred and received the result will be shown in the right side of the bottom toolbar.

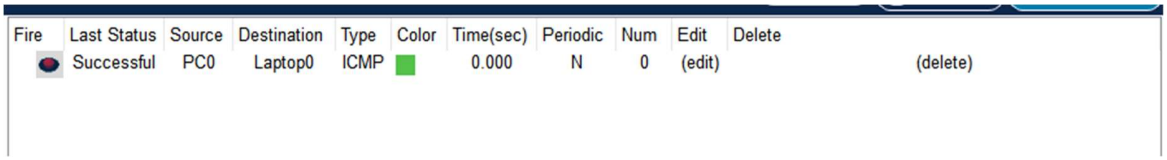


Figure 2.22: Result of successful packet transmit and receive.

This verifies that the network is properly configured and working correctly.

Observation and Result:

Here are the mostly used components that will be needed for future work. User can find, get the component, show the configuration window and connect devices using "Automatic Console Connection Type" from the cable menu. Configured the component use the simulation and successfully send and receive a packet.

Additional reading materials/ online tutorial/ References:

1. Video Tutorial: https://www.youtube.com/watch?v=frUQMHXhnvs
2. Article: https://learningnetwork.cisco.com/s/article/Using-Packet-Tracer-for-CCNA-Studys

3. Basic Router Configuration.

Title: Basic Router Configuration.

Objectives

1. Learn how to access and configure a router.
2. Understand basic commands for initial device setup.
3. Assigning name to the router and setting up domain.
4. Disabling DNS lookup.
5. Assigning console and vty password and creating a banner.
6. Configuring 3 interfaces (g0/0, g0/1 and loopback) of the router.
7. Verify network connectivity.
8. Saving the running configuration file.
9. Accessing the router form another computer using telnet.

Prerequisites

1. Knowledge of how to build network topology, configure the components.
2. Familiarity with command-line interfaces.

Theory

Router

Figure 3.1: Router

A router is a device that connects two or more packet-switched networks or subnetworks.

It serves two primary functions: managing traffic between these networks by forwarding data packets to their intended IP addresses, and allowing multiple devices to use the same Internet connection.

Switch

Figure 3.2: Switch

The Switch is a network device that is used to segment the networks into different subnetworks called subnets or LAN segments.

Switches have many ports, and when data arrives at any port, the destination address is examined first and some checks are also done and then it is processed to the devices.

VTY (Virtual Teletype)

“VTY" stands for Virtual Teletype. VTY is a virtual port used for Telnet or SSH access to the device, allowing network administrators to connect to and manage Cisco devices remotely.

Telnet (Teletype Network)

Telnet is a client/server application protocol that provides access to virtual terminals of remote systems on local area networks or the Internet.

Methodology

1. Set Up the Topology and Initialize End Devices.
2. Assigning name to the router and setting up domain.
3. Disabling DNS lookup.
4. Assigning console and vty password and creating a banner.
5. Configuring 3 interfaces (g0/0, g0/1 and loopback) of the router.
6. Verify network connectivity.
7. Saving the running configuration file.
8. Accessing the router form another computer using telnet.

Equipment

1. 2 Router (2911)
2. 1 Switch (2960-24TT)
3. 3 End Devices (2 PC, 1 Laptop)

Procedure

Step 1: Creating a network and configure the End Devices.

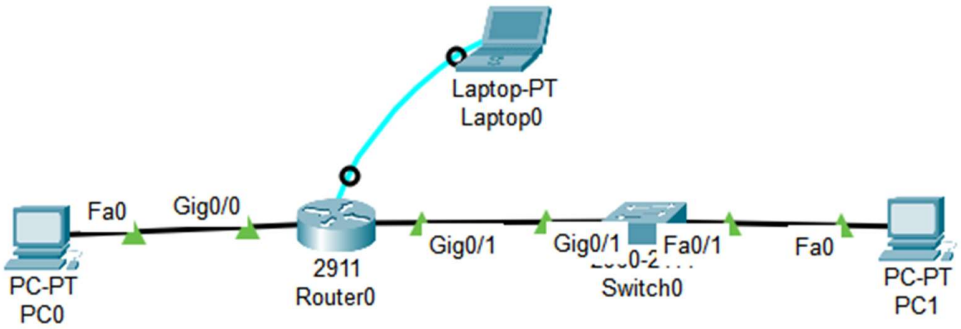


Figure 3.3: Topology of established network.

- Configure PC0 IP address 192.168.10.2, subnet mask 255.255.255.0 and default gateway 192.168.10.1
- Configure PC1 IP address 192.168.11.2, subnet mask 255.255.255.0 and default gateway 192.168.11.1

Go to the laptop's terminal and click ok that will give access to the router's CLI.

Step 2: Assigning name to the router and setting up domain.

a. Console into the router and enable privileged EXEC mode.

router> enable

b. Enter configuration mode.

router# config terminal

c. Assign a device name to the router.

router(config)# hostname R1

Step 3: Disabling DNS lookup.

a. Disable DNS lookup to prevent the router from attempting to translate incorrectly entered commands as though they were host names.

R1(config)# no ip domain lookup

Step 4: Assigning console, privileged EXEC and VTY password and creating a banner (Message).

a. Assign the console password to "ciscoConsole"

R1(config)# line console 0
R1(config-line)# password ciscoConsole
R1(config-line)# login

"line console 0" specifies the console line configuration. The 0 represents the first (and usually only) console line on the device.

b. Assign the privilege mode (Exec mode) password to "ciscoEnable"

R1(config)# enable password ciscoEnable

c. Assign "ciscoVty" as the vty password and enable login.

R1(config)# line vty 0 4
R1(config-line)# password ciscoVty
R1(config-line)# login

Here "line vty 0 4" means there can be 5 (0,1,2,3,4) users.

d. Create a banner that warns anyone accessing the device.

R1(config)# banner motd @
****
Authorized Access Only.
****
@

Step 5: Configuring 3 interfaces (g0/0, g0/1 and loopback) of the router.

a. Configure all three interfaces on the router with the IPv4 address and add description.

R1(config)# interface g0/0
R1(config-if)# ip address 192.168.10.1 255.255.255.0
R1(config-if)# description Connection to PC-0
R1(config-if)# no shutdown
R1(config-if)# exit

R1(config)# interface g0/1
R1(config-if)# ip address 192.168.11.1 255.255.255.0
R1(config-if)# description Connection to Switch-1
R1(config-if)# no shutdown
R1(config-if)# exit

R1(config)# interface loopback0
R1(config-if)# ip address 192.168.12.1 255.255.255.0
R1(config-if)# description loopback adapter
R1(config-if)# no shutdown
R1(config-if)# exit

(**N.B.** The loopback interface acts as a placeholder for the static IP address and provides default routing information.)

Step 6: Verify network connectivity.

a. Check console login
b. Check network's packet transfer to verify successful connection.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC0	PC1	ICMP		0.000	N	0	(edit)	

Figure 3.5: Packet transfer from PC0 to PC1 successful.

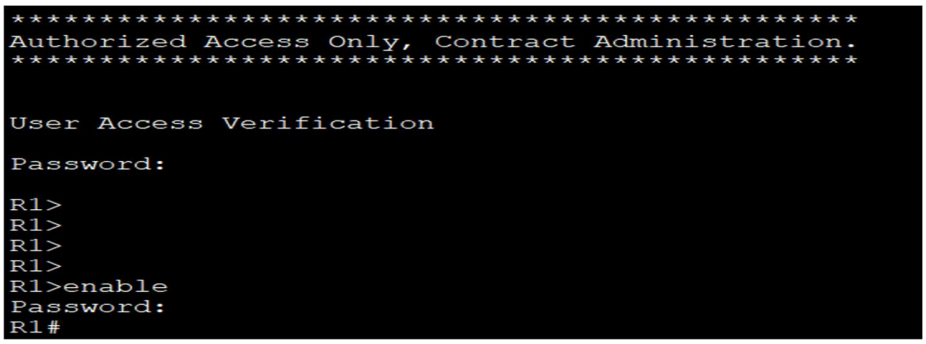


Figure 3.6: Console login and privileged EXEC mode login.

Step 7: Saving the running configuration file.

a. Save the running configuration to the startup configuration file.
R1# copy running-config startup-config

Step 8: Accessing the router form another computer using telnet.

a. From PC-A's Command Prompt
C:\>**telnet 192.168.10.1**
b. Enter password (ciscoVty) to check if it's working.

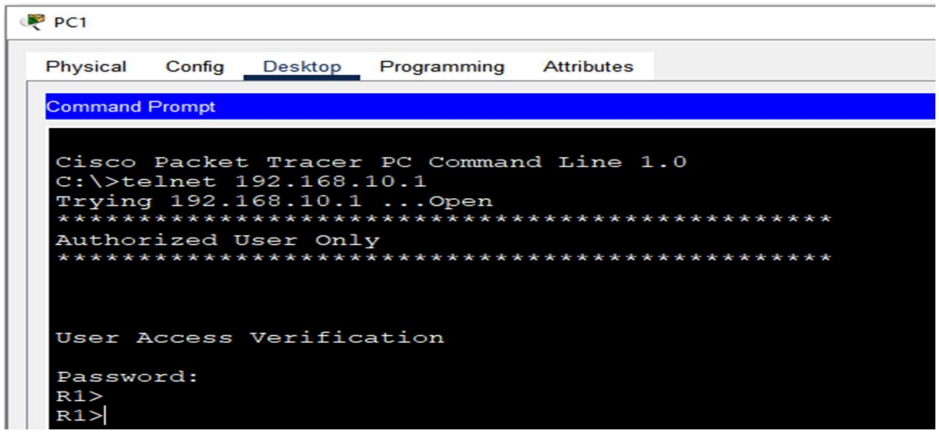


Figure 3.7: Login from PC0 to Router using Telnet.

Observation and Result

The telnet is working so does the user verification system.

Additional reading materials/ online tutorial/ References.

4. Application Layer Services (Server).

Title: Application Layer Services (Server)

Objectives

1. Learn how to setup and configure DHCP servers.
2. Learn how to setup and configure DNS servers.
3. Learn how to setup and configure HTTP servers.
4. Learn how to setup and configure Email (SMTP) servers.

Prerequisites

1. Knowledge of how to build network topology, configure the components.

Theory

Application layer Services

An application layer is an abstraction layer that specifies the shared communication protocols and interface methods used by hosts in a communications network. An application layer abstraction is specified in both the Internet Protocol Suite and the OSI model. Although both models use the same term for their respective highest-level layer.

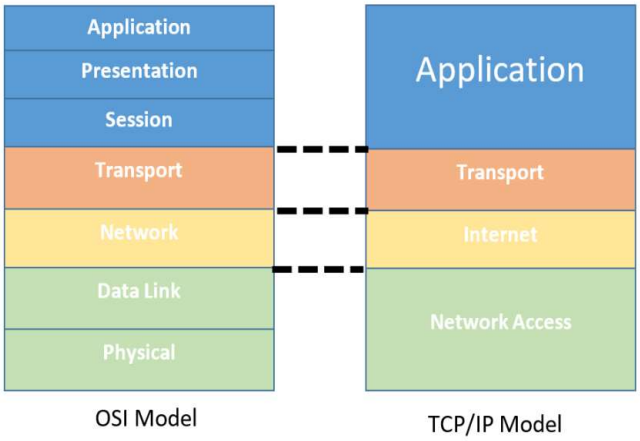


Figure 4.1: OSI and TCP/IP model

HTTP (Hypertext Transfer Protocol)

HTTP is an application layer protocol in the Internet protocol suite model for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web, where hypertext documents include hyperlinks to other resources that the user can easily access, for example by a mouse click or by tapping the screen in a web browser.

HTTP Server
HTTP server is a combination of hardware and software. Through this software and hardware, we translate and browse all the data and information. The webpages or websites that we visit while browsing, is stored within this server.

DHCP (Dynamic Host Configuration Protocol)
DHCP (Dynamic Host Configure Protocol) is a network management protocol used to dynamically assign an IP address to any device, or node, on a network so it can communicate using IP. DHCP automates and centrally manages these configurations rather than requiring network administrators to manually assign IP addresses to all network.

DNS (Domain Name System)
The Domain Name System (DNS) is the phonebook of the Internet. When user type domain names such as 'google.com' or 'nytimes.com' into web browsers, DNS is responsible for finding the correct IP address for those sites. Browsers then use those addresses to communicate with origin / main servers or to access website information.

SMTP (Simple Mail Transfer Protocol)
Simple Mail Transfer mechanism (SMTP) is a mechanism for exchanging email messages between servers. It is an essential component of the email communication process and operates at the application layer of the TCP/IP protocol stack. SMTP is a protocol for transmitting and receiving email messages. In this article, we are going to discuss every point about SMTP.

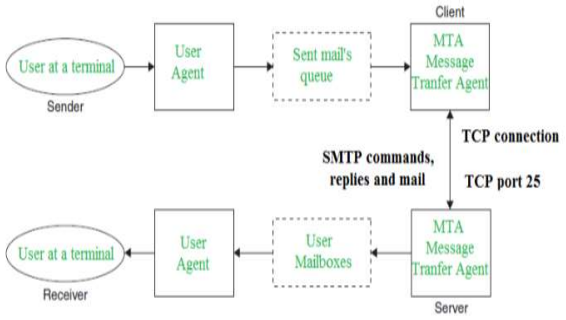


Figure 4.2: SMTP Model

Methodology

1. Set up the topology.
2. Configure IP addresses for the servers.
3. Configure DHCP server.
4. Configure IP address and default gateway automatically by DHCP.
5. Configure DNS server and verify it's working.
6. Configure HTTP server and verify it's working.
7. Configure Email (SMTP) server and verify it's working.

Equipment

1. 2 Server-PT
2. 4 Switch (2960-24TT)
3. 4 End Devices (2 PC, 1 Laptop)

Procedure

Step 1: Creating a network and configure the End Devices.

Here the IP of the PC's will not be configured manually because the DHCP server will automatically provide and configure IP for the PCs.

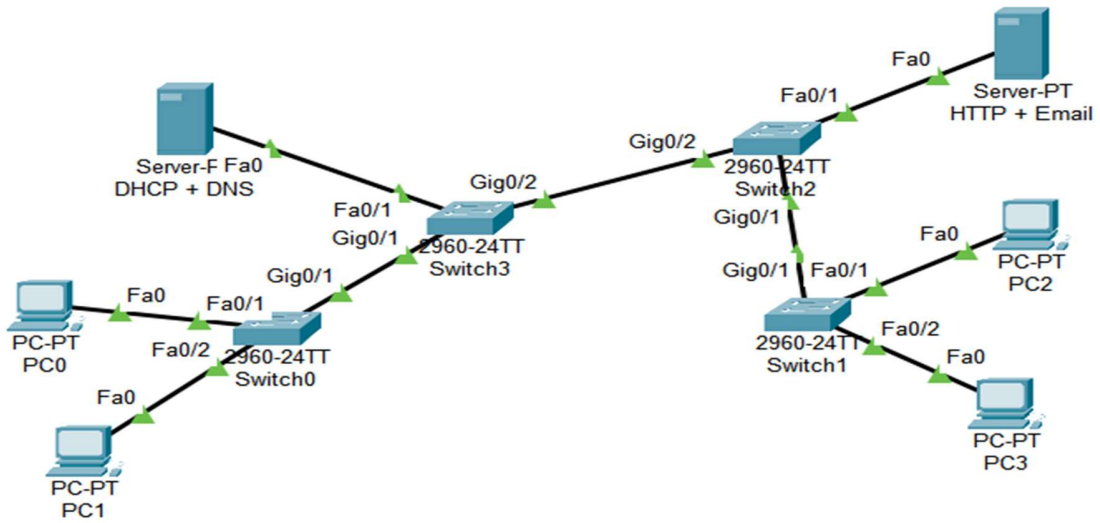


Figure 4.3: Topology of established network.

Step 2: Configure IP Addresses for the servers.

- Set 192.168.100.10 as IP address and 192.168.100.1 as default gateway for the DHCP + DNS server.

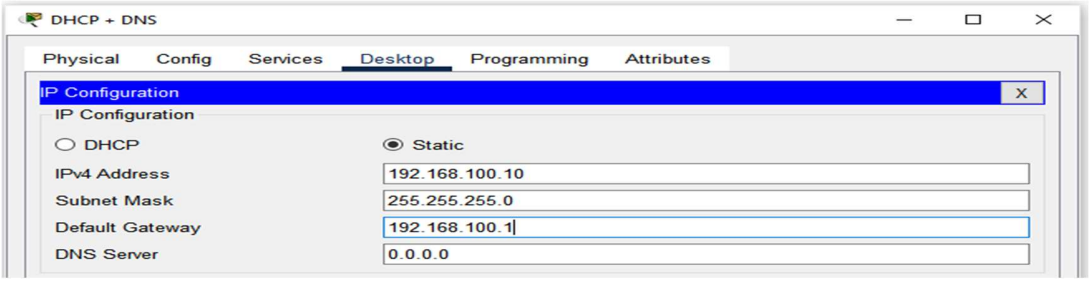


Figure 4.4: IP address and default gateway for DHCP+DNS server.

- Set 192.168.100.10 as IP address and 192.168.100.1 as default gateway for the HTTP + Email server.

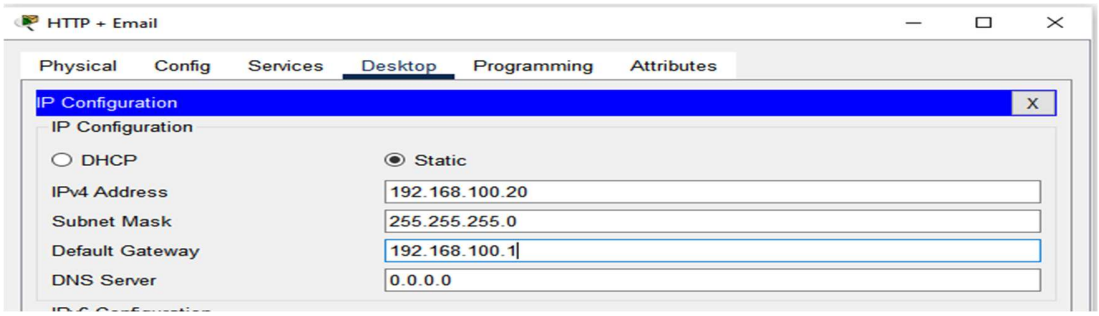


Figure 4.5: IP address and default gateway for HTTP + Email server.

Step 3: Configure DHCP Server

- Click on the DHCP + DNS server, from service tab select DHCP from the right menu.
- Turn on the server.
- Put 192.168.100.1 as default gateway.
- For DNS server put 192.168.100.10 (DHCP + DS Server's IP)
- Set start IP address 192.168.100.50
- And Define maximum user number (Which will get IP from this server automatically).
- Click Save Button

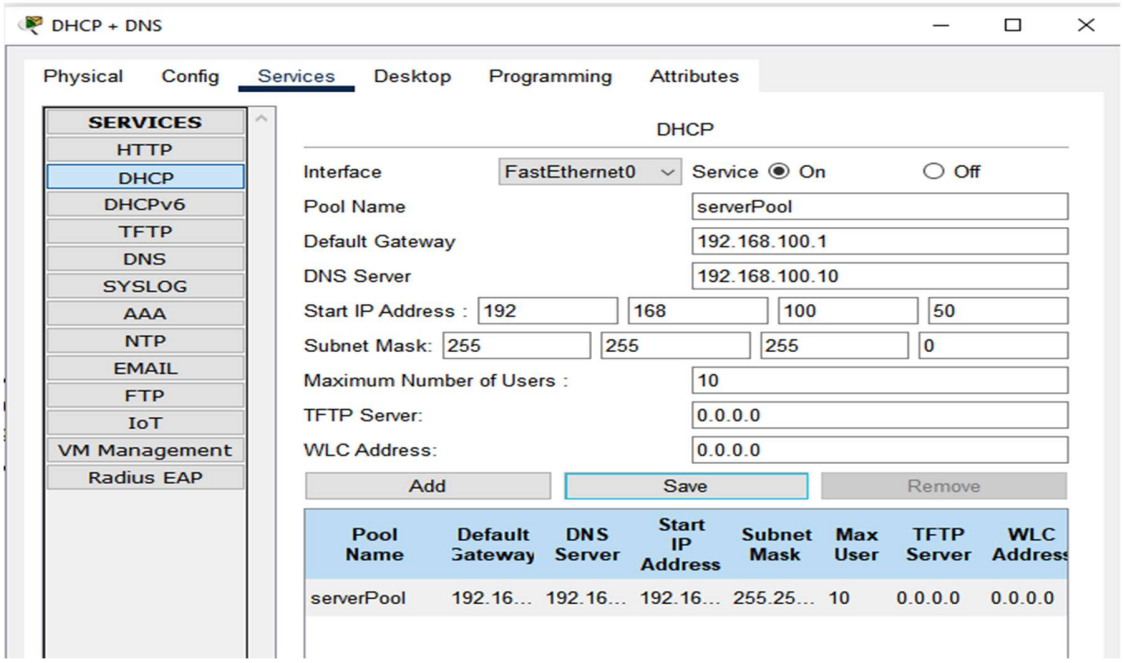


Figure 4.6: DHCP server configuration.

Step 4: Configure IP address and default gateway automatically by DHCP

- In every PC on desktop's IP configuration menu click DHCP it will request for IP address, Default gateway and when the request is successful the IP address, Subnet mask and Default gateway will automatically and dynamically initialized.

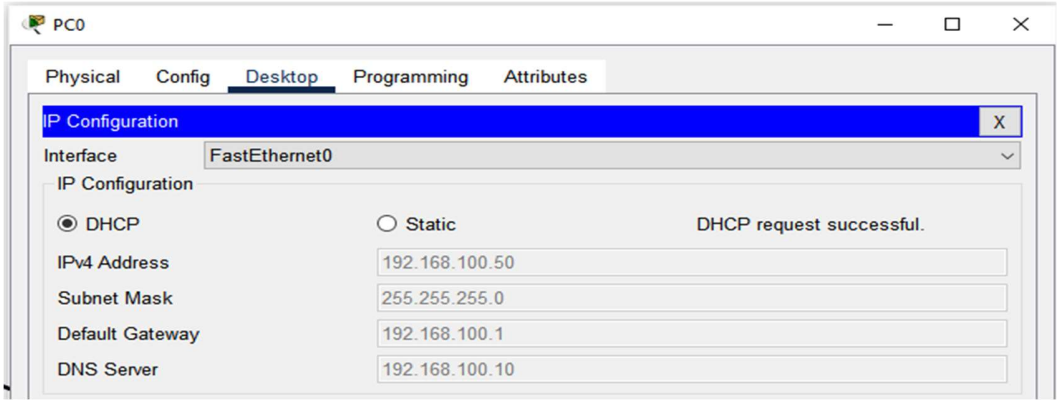


Figure 4.7: IP address, subnet mask and default gateway automatically configured on PC0.

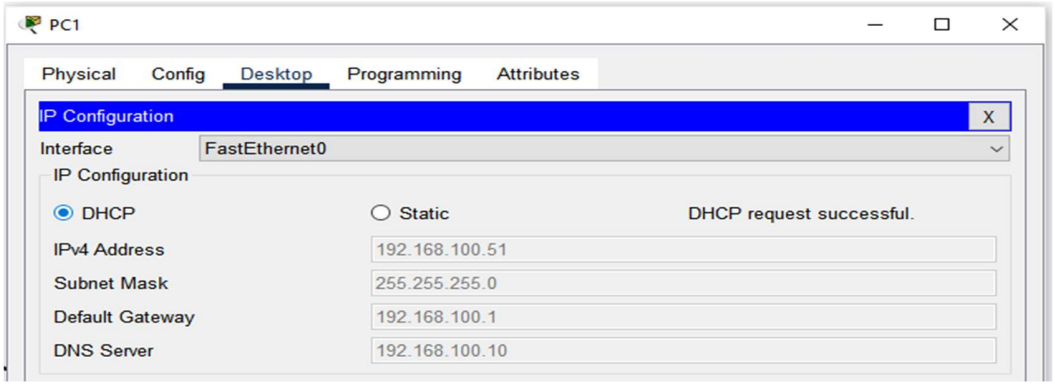


Figure 4.8: IP address, subnet mask and default gateway automatically configured on PC1.

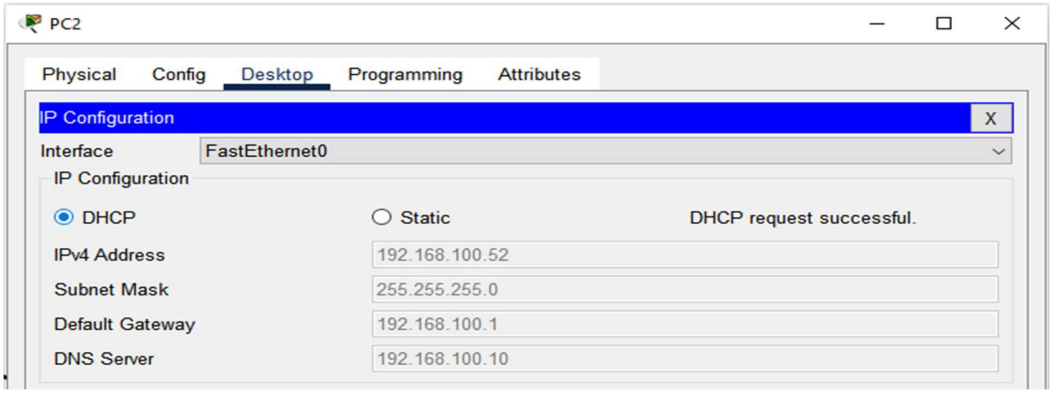


Figure 4.9: IP address, subnet mask and default gateway automatically configured on PC2.

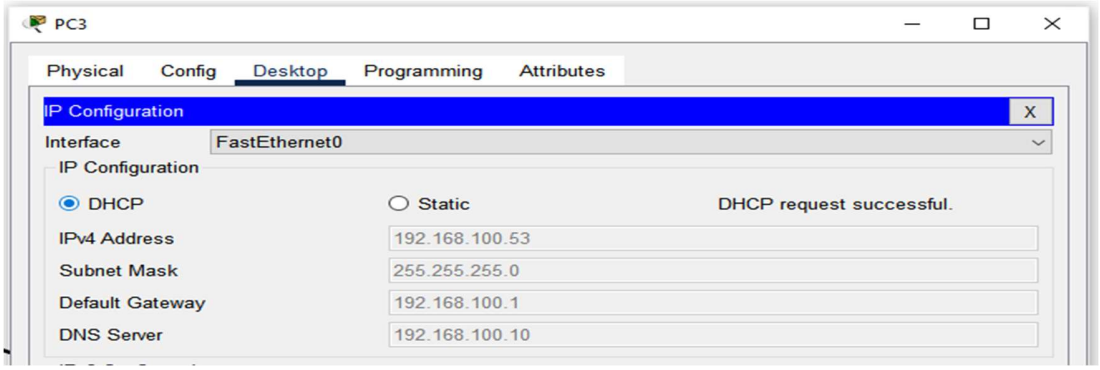


Figure 4.10: IP address, subnet mask and default gateway automatically configured on PC3.

Step 5: Configure DNS server.

- In DHCP + DNS server click DNS option from Services menu.
- Turn on the DNS service
- Put a name for an IP address
- Put the IP address
- Click save

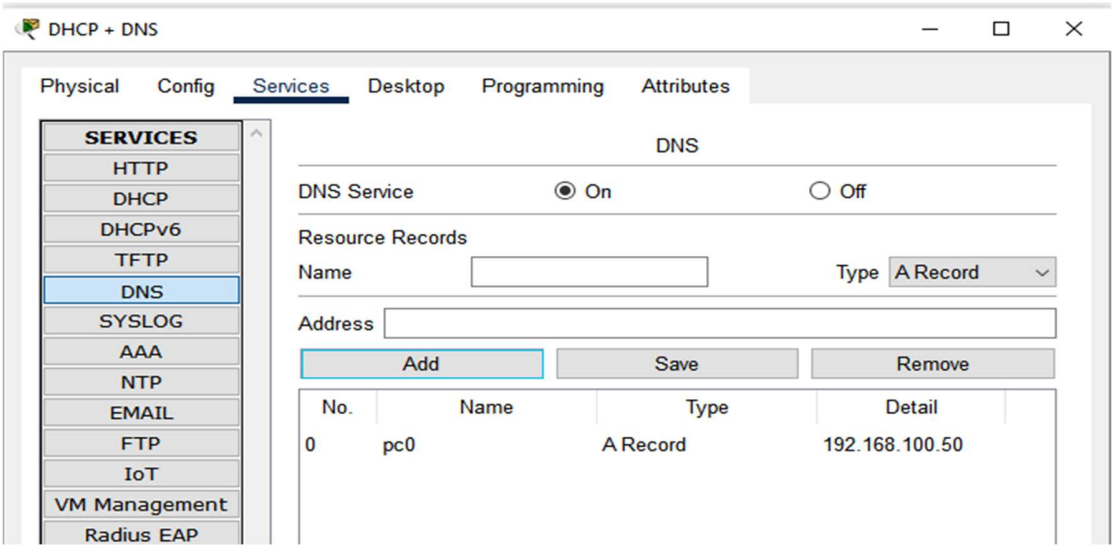


Figure 4.11: Adding record in the DNS server.

- To verify if its working, here in DNS server 192.168.100.50 (which is the IP address for PC0) is recorded as “pc0”. Open any other PC (for this case PC1) click PC1’s command prompt and write **ping pc0**. Which will send and receive 4 packets from pc0 (192.168.100.50 or PC0). If the 4 packets are received then the DNS server is working correctly.

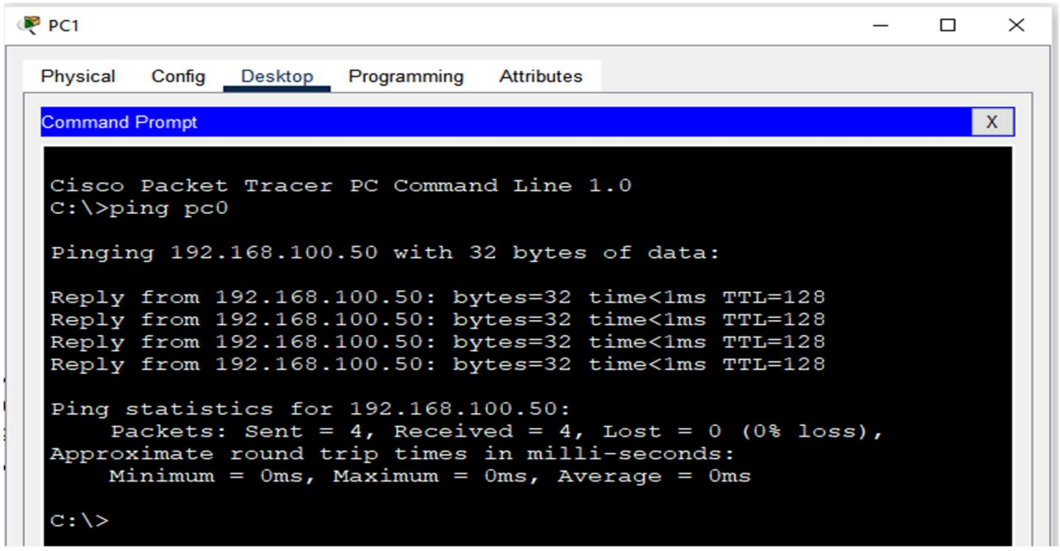


Figure 4.12: All packet are successfully send and received from PC0.

Step 6: Configure HTTP server and verify it's working.

- IN HTTP + Email server turn on HTTP and click on New File.

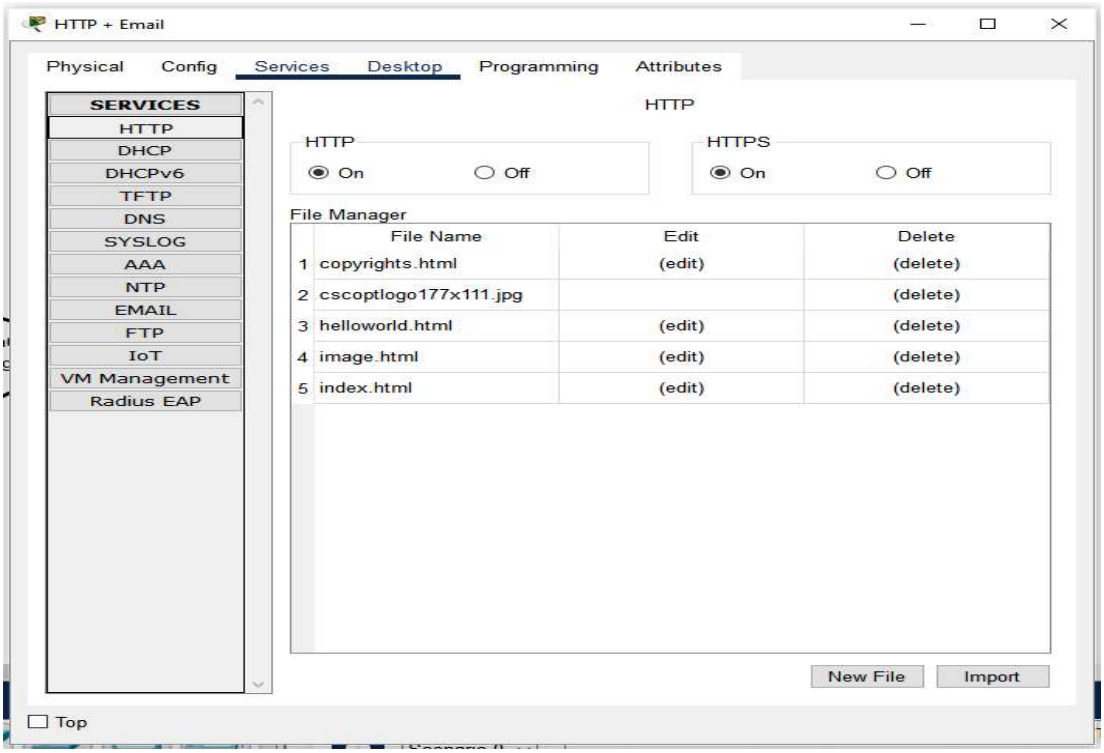


Figure 4.13: HTTP server menu.

- Put a file name and a html text for verification.

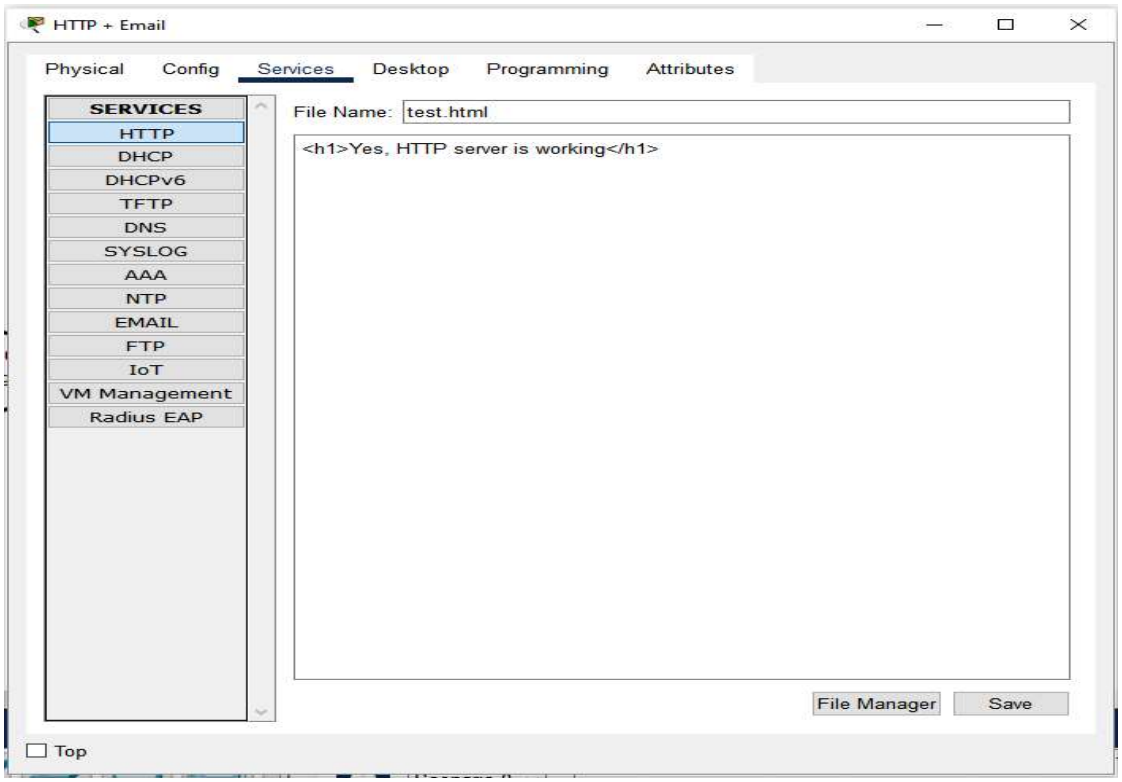


Figure 4.14: Creating a html document.

- To verify its working from any PC (in this case PC1) open its browser and type the IP address or the domain name (IF its been added into the DNS server's record) then the html document name that has been created now. In this case the link will be IP address/test.html or http://192.168.100.20/test.html in the URL field. It will show the contents from the test.html that is created above.

Figure 4.15: Contents from test.html document which is stored in HTTP server.

Step 7: Configure Email (SMTP) server and verify it's working.

- In HTTP + Email server form services select Email. Here in user field fill user name and password and add by clicking + option on right.

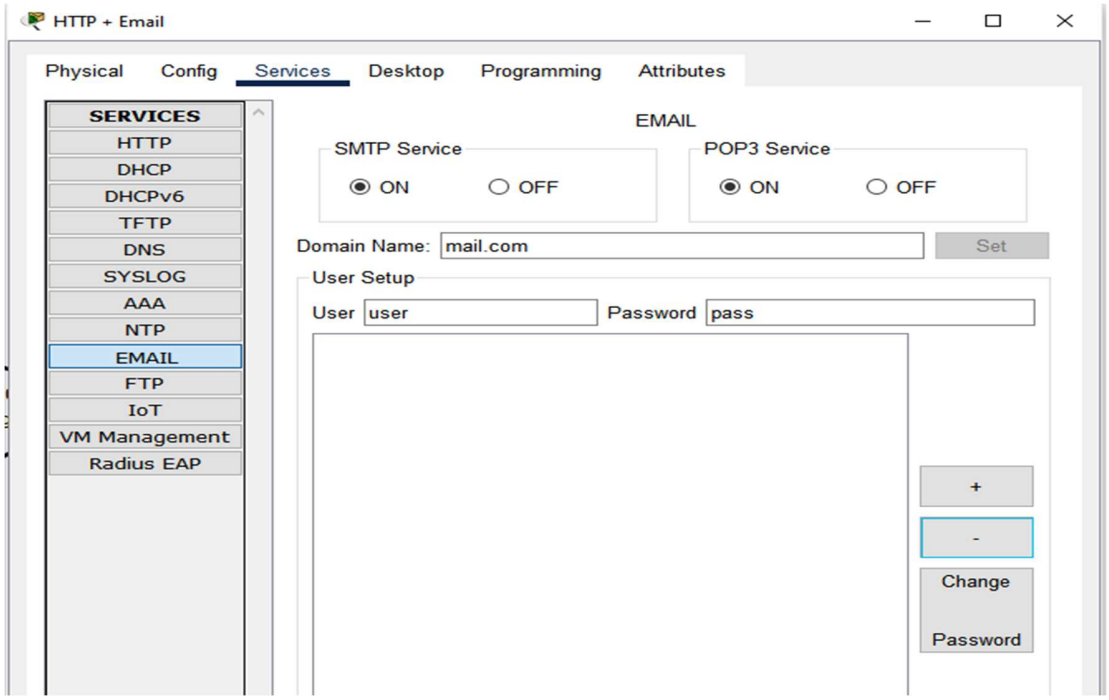


Figure 4.16: Adding users in SMTP services.

- From any PC's (in this case PC1 and PC3) select browser and fill the credential. Remember these credential should match with the added user's credential from Email services. In incoming and outgoing server put HTTP + Email server's IP address.

Figure 4.17: User 1's credential added.

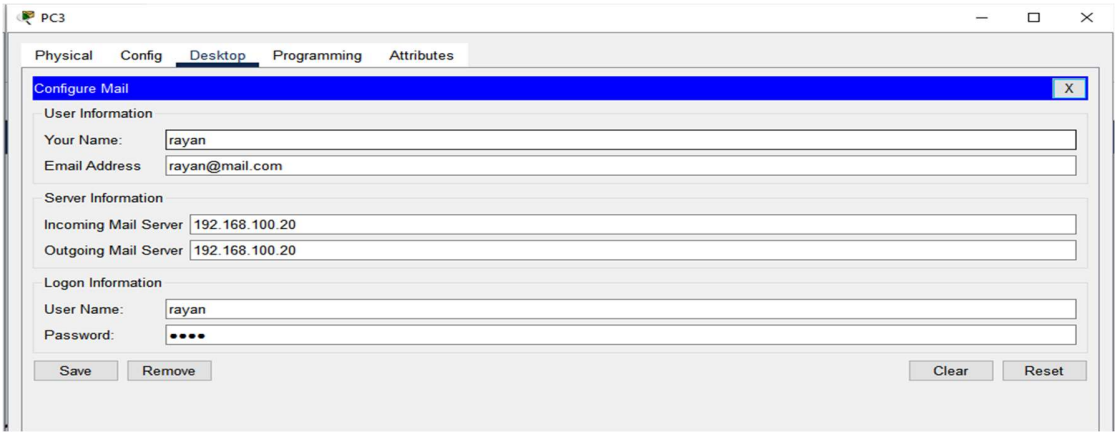


Figure 4.18: User 1's credential added.

- To test if its working compose a mail and send the mail. On the other pc if the mail is available from the receive menu then the Email/SMTP server is working correctly.

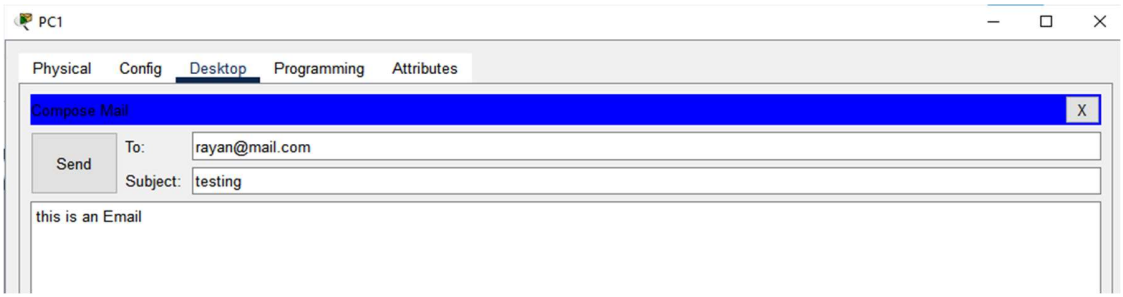


Figure 4.19: Composing an Email.

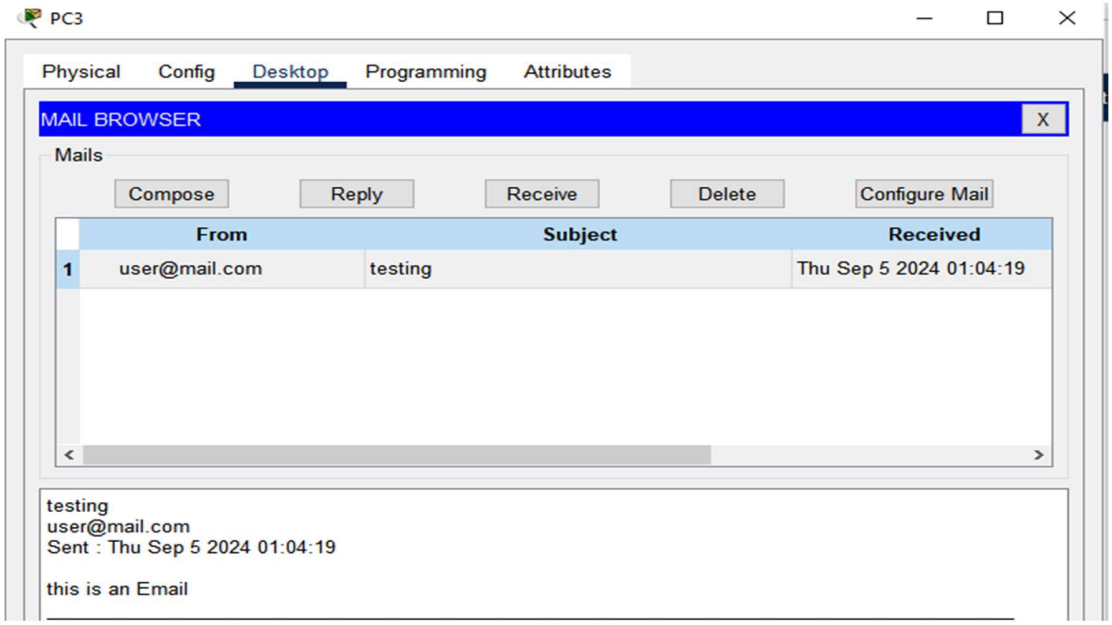


Figure 4.20: Email successfully received.

Observation and Result

All of the servers i.e. DHCP, DNS, HTTP, Email are working properly.

Additional reading materials/ online tutorial/ References.

1. DHCP server setup article: https://www.packettracerlab.com/setting-up-a-dhcp-server/
2. DNS server setup article: https://www.packettracerlab.com/dns-in-cisco-packet-tracer/
3. HTTP server setup article: https://ccnapracticallabs.com/set-up-http-server-packet-tracer/
4. Email server setup article: https://computernetworking747640215.wordpress.com/2018/07/05/configuring-a-mail-server-in-packet-tracer/

5. Sub-netting and VLSM

Title: Routing Protocol Configuration.

Objectives

1. Implement Sub-netting and VLSM

Prerequisites

1. Knowledge of how to build network topology, configure the components.
2. **Theoretical knowledge on Sub-netting, VLSM, and IP address and subnet mask calculation based on VLSM.**
3. Different class of IP,

Theory

Sub-netting

When a bigger network is divided into smaller networks, to maintain security, then that is known as Sub-netting.

Subnets make networks more efficient. Through sub-netting, network traffic can travel a shorter distance without passing through unnecessary routers to reach its destination.

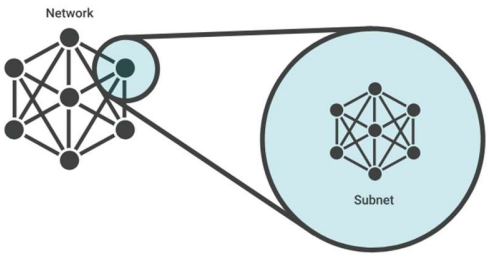


Figure 5.1: Sub-net

Approaches to sub-netting:

There are two approaches to subnetting an IP address for a network: Fixed length subnet mask (FLSM) and variable-length subnet mask (VLSM).

In FLSM subnetting, all subnets are of equal size with an equal number of host identifiers. It use the same subnet mask for each subnet, and all the subnets have the same number of addresses in them. It tends to be the most wasteful because it uses more IP addresses than are necessary.

VLSM is a subnet design strategy that allows all subnet masks to have variable sizes. In VLSM subnetting, network administrators can divide an IP address space into subnets of different sizes,

and allocate it according to the individual need on a network. This type of sub netting makes more efficient use of a given IP address range.

Subnet mask

Subnet masks are used by a computer to determine if any computer is on the same network or on a different network. An IPv4 subnet mask is a 32-bit sequence of ones (1) followed by a block of zeros (0). The ones designate the network prefix, while the trailing block of zeros designates the host identifier. In shorthand, we use /24, which simply means that a subnet mask has 24 ones, and the rest are zeros.

Methodology

1. Set up the topology.
2. Configure IP addresses for the PCs and routers.
3. Implementing OSPF in routers.
4. Check connectivity.

Equipment

1. 2 Server-PT
2. 4 Switch (2960-24TT)
3. 4 End Devices (2 PC, 1 Laptop)

Procedure:

Step 1: Setup the network Topology.

- Connect all component as shown in this figure.

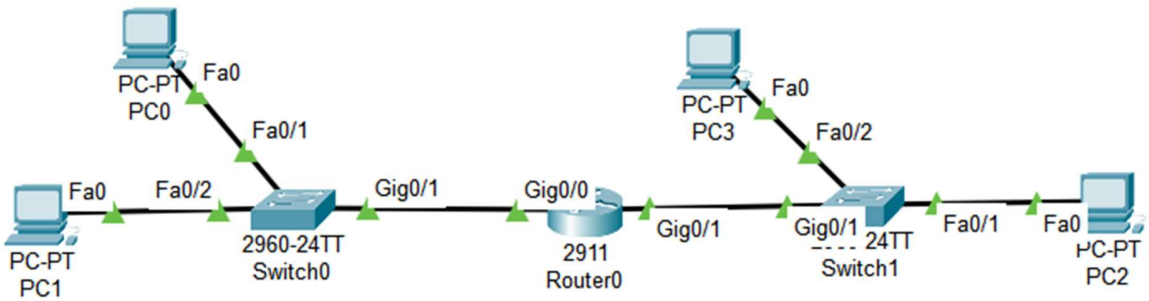


Figure 5.2: Topology of established network.

Here assume that, on left side the network need 10 valid/usable IP addresses and on the right side it needs 2 valid/usable IP address. For this requirement do the calculation of IP address range and subnet mask for the range for IP address 192.168.10.0/24 which will be needed for configuration.

Step 2: Configure IP addresses for the PCs and routers.
Here is the answer for the

- For left side of network: (PC0, PC1)
  IP address range: 192.168.10.0 – 192.168.10.15
  Subnet mask: 255.255.255.240
  Default gateway: 192.168.10.1
  Network address: 192.168.10.0
  Broadcast address: 192.168.10.15

- For right side of network: (PC2, PC3)
  IP address range: 192.168.10.16 – 192.168.10.23
  Subnet mask: 255.255.255.248
  Default gateway: 192.168.10.17
  Network address: 192.168.10.16
  Broadcast address: 192.168.10.23

- Configure IP address, subnet mask and default gateway on the PC and routers according to the calculation.

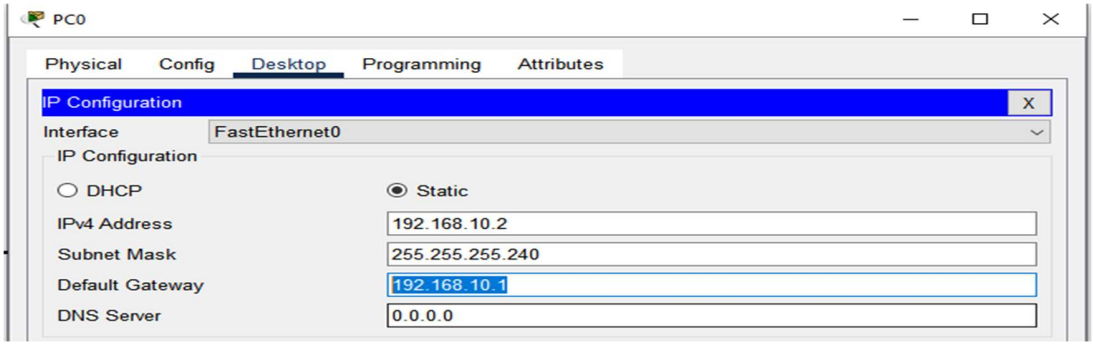


Figure 5.3: IP address, subnet mask and default gateway configured on PC0

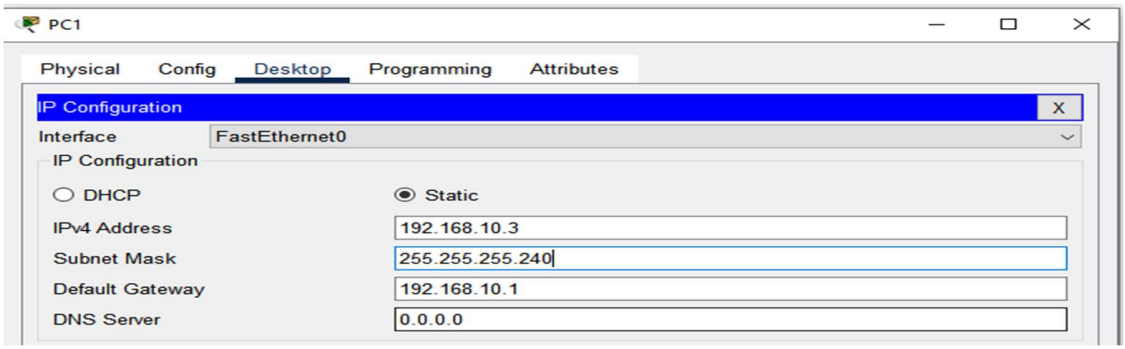


Figure 5.4: IP address, subnet mask and default gateway configured on PC1

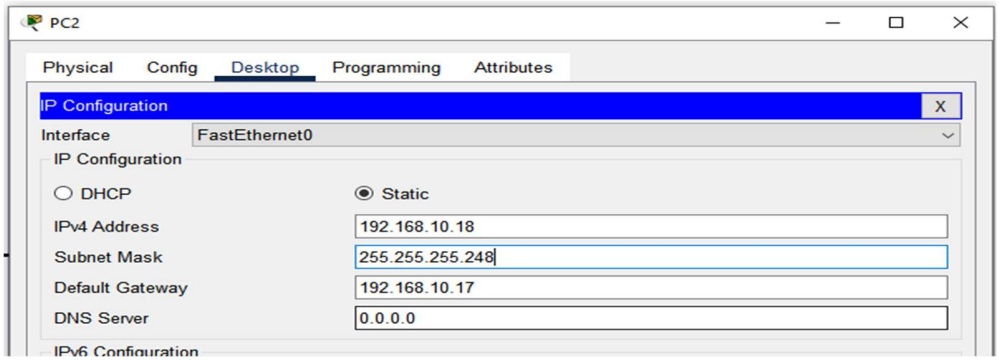


Figure 5.5: IP address, subnet mask and default gateway configured on PC2

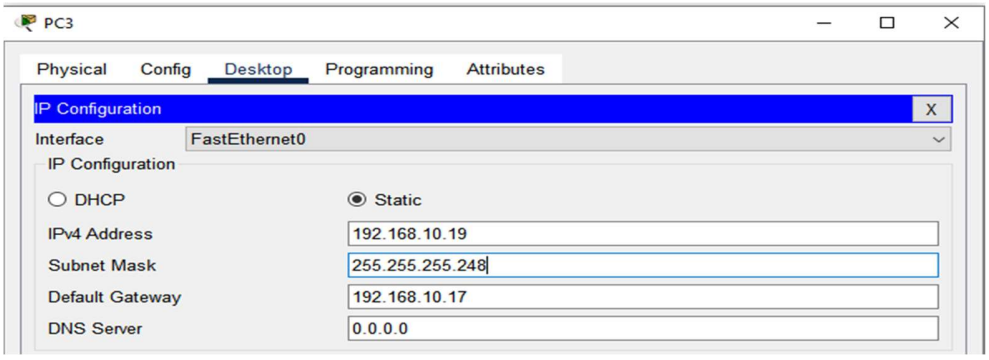


Figure 5.6: IP address, subnet mask and default gateway configured on PC3

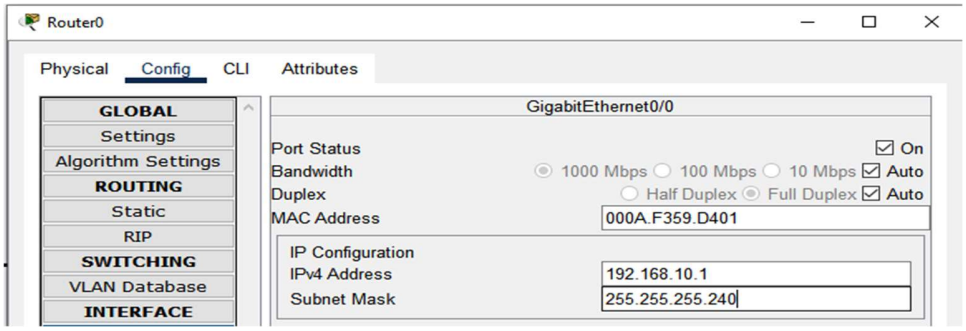


Figure 5.7: IP address, subnet mask for interface gig0/0 configured on Router

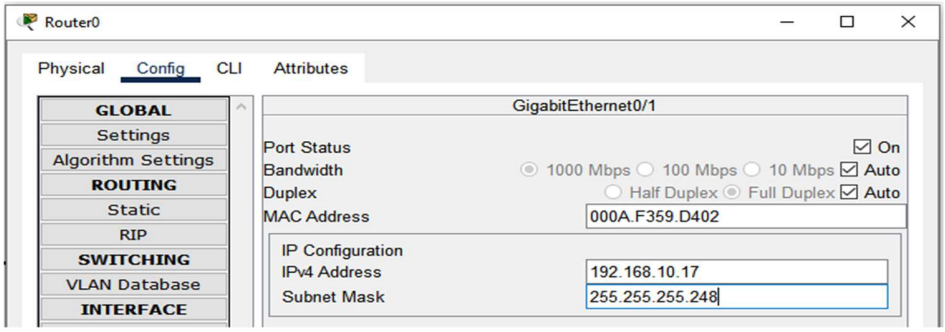
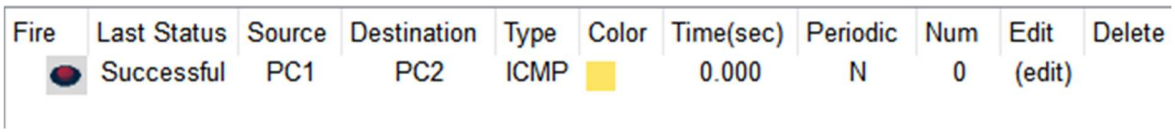


Figure 5.8: IP address, subnet mask for interface gig0/0 configured on Router

Step 3: Check connectivity

- For confirming if VLSM is working correctly transmit and receive a packet from any PC from left side sub network to the right side sub network to opposite side network PC. If its successful then the VLSM is correctly implemented.



Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC1	PC2	ICMP		0.000	N	0	(edit)	

Figure 5.9: Packet transmission and received successfully.

Observation and Result

Packet from one sub network to another sub network's end device is successfully transmitted and received, meaning VLSM is working proporly.

Additional reading materials/ online tutorial/ References.

6. Routing Protocol Configuration

Title: Routing Protocol Configuration.

Objectives

1. Configure Static routing protocol.
2. Configure OSPF routing protocol.
3. Configure RIP version 2 routing protocol

Prerequisites

1. Knowledge of how to build network topology, configure the components.

Theory

Static

Static routing is a process in which we have to manually add routes to the routing table.It enables the router to assign a specific path to each network segment and to keep track of network changes.

Network ID

The network ID is a portion of an IP address that is used to designate a specific network or host. This section of the IP address is typically found towards the beginning of an IP address.

Host ID

It is the fragment of an IP address that uniquely identifies a host within a network.

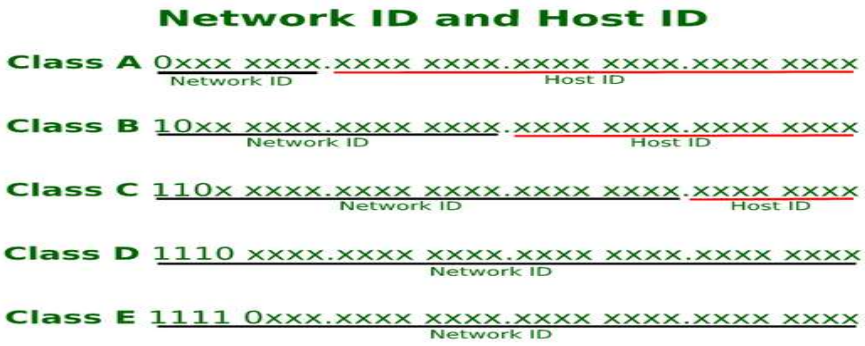


Figure 6.1: Network ID and Host ID

RIP Version 2

Routing Information Protocol (RIP) is a dynamic routing protocol that uses hop count as a routing metric to find the best path between the source and the destination network. It is a distance-vector routing protocol that has an AD value of 120 and works on the Network layer of the OSI model. RIP uses port number 520. There are three versions of routing information protocol – RIP Version1**,** RIP Version2**,** and RIPng**.**

RIP v1: Known as *Classful* Routing Protocol. It doesn't send information of subnet mask in its routing update.

RIP v2: Known as *Classless* Routing Protocol. It sends information of subnet mask in its routing update.

OSPF

Open Shortest Path First (OSPF) is a link-state routing protocol that is used to find the best path between the source and the destination router using its own Shortest Path First. It is a network layer protocol which works on protocol number 89 and uses AD value 110.

Neighbor Discovery:

OSPF routers use Hello packets to discover and establish neighbor relationships with directly connected routers. Routers must agree on certain parameters (like Hello and Dead intervals) to become neighbors.

Link-State Advertisements (LSAs):

Once neighbors are established, routers exchange Link-State Advertisements (LSAs). Which contains information about the router's links and their state.

Link-State Database (LSDB):

Each router maintains an LSDB, a database of all received LSAs. The LSDB is identical on all routers within the same OSPF area.

Shortest Path First (SPF) Algorithm

Routers use the SPF algorithm (Dijkstra's algorithm) to calculate the shortest path to each destination and The result is installed in the router's routing table.

Areas

Area 0, the backbone area, is the central area to which all other areas connect.

Route Calculation-

OSPF calculates the best path based on the cost (metric), which is typically determined by the bandwidth of the links.

Methodology

1. Set up the topology.
2. Configure IP addresses for the PCs and routers.
3. Implementing routing protocol in routers.
4. Check connectivity.

(DO these steps for all of the routing protocols)

Equipment

1. 4 Routers (2911)
2. 6 Switch (2960-24TT)
3. 6 End Devices

Procedure:

1. Static routing protocol

Step 1: Setup the network Topology.

- Connect all component as shown in this figure and for connecting routers with routers use serial cable for that use HWIC-2T module.

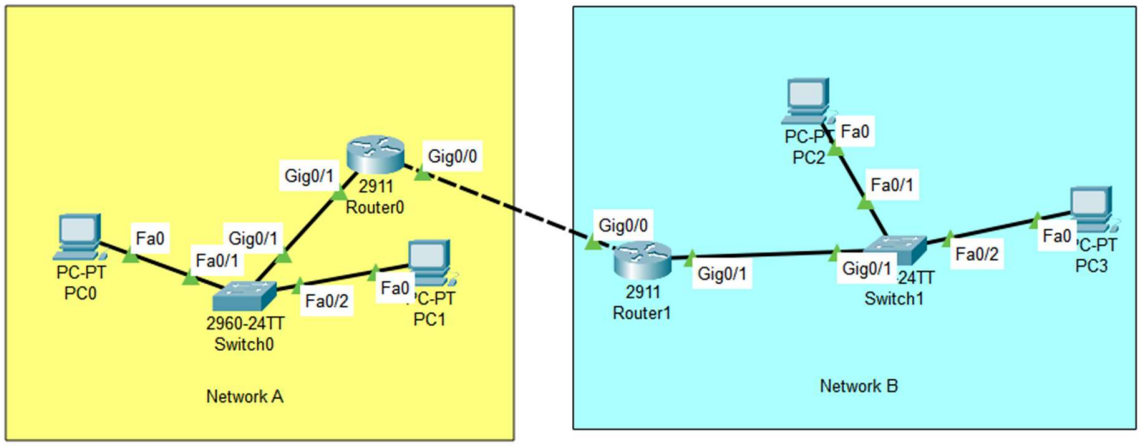


Figure 6.2: Topology of established network for static routing protocol.

Step 2: Configure IP addresses for the PCs and routers.

- Configure IP address, subnet mask and default gateway of PCs.

PC	IP Address	Subnet Mask	Default Gateway
PC 0	192.168.10.2	255.255.255.0	192.168.10.1
PC 1	192.168.10.3	255.255.255.0	192.168.10.1
PC 2	192.168.20.2	255.255.255.0	192.168.20.1
PC 3	192.168.20.3	255.255.255.0	192.168.20.1

- Turn on every router interface that is connected and configure IP address and subnet mask for all of the interfaces that are being used.

Router	Interface	IP Address	Subnet Mask
Router 0	Gigabit Ethernet 0/1	192.168.10.1	255.255.255.0
Router 0	Gigabit Ethernet 0/0	192.168.30.1	255.255.255.0
Router 1	Gigabit Ethernet 0/0	192.168.30.2	255.255.255.0
Router 1	Gigabit Ethernet 0/1	192.168.20.1	255.255.255.0

Step 3: Implementing Static routing protocol in routers.

CLI command : ip route <network id> <subnet mask><next hop>

For Network A the destination network is Network B that is 192.168.20.0 which will be available via Router 1's GigabitEthernet port 0/0 here this is considered as a hop in between.

Router(config)# ip route 192.168.20.0 255.255.255.0 192.168.30.2

And same for the Network B where in this case the destination will be Network A via Router 0's GigabitEthernet Port 0/0.

Router(config)# ip route 192.168.10.0 255.255.255.0 192.168.30.1

Step 4: Checking connectivity.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC0	PC3	ICMP		0.000	N	0	(edit)	

Figure 6.3: Packet successfully transmitted form Network A to Network B.

2. OSPF routing protocol

Step 1: Setup the network Topology.

Setup network topology like before (in Static routing protocol). Or you can copy the topology from static routing experiment before implementing the static protocol. If static protocol is already implemented then use "no" command before the static routing command to undo the implementation of Static so that now it can be used for implementing OSPF.

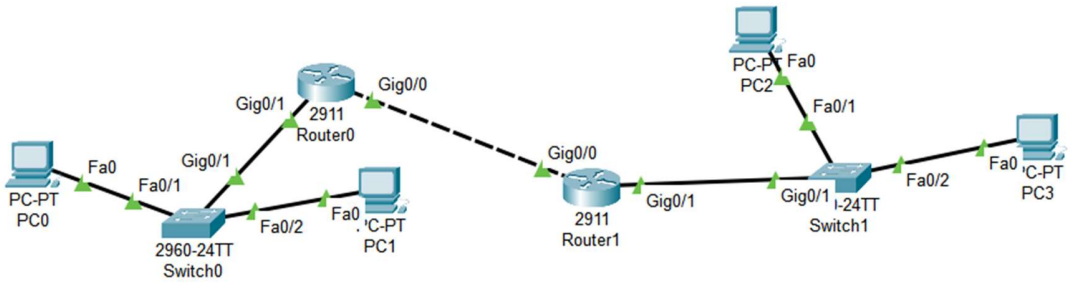


Figure 6.3: Topology of established network for OSPF routing protocol.

Step 2: Configure IP addresses for the PCs and routers.

Do this step also like the Static routing experiment, nothing is changed.

Step 3: Implementing OSPF in routers

For Router 0:

- Console into the router and enable privileged EXEC mode.
  Router> enable
- Enter configuration mode.
  Router# config terminal
- Enable OSPF.
  Router (config)# router ospf 1
- Assign Router IDs.
  Router (config-router)# router-id 1.1.1.1

(Assign a router ID to each router. This ID should be a unique IP address-like identifier for OSPF to use. It does not need to be an actual IP address from the network but should be unique)

- Define OSPF Network Statements

Router (config-router)# network 192.168.10.0 0.0.0.255 area 0
Router (config-router)# network 192.168.30.0 0.0.0.255 area 0

(Define the networks that OSPF should advertise and the associated wildcard mask. The wildcard mask is the inverse of the subnet mask. Specify the area as 0 (for a simple setup with one OSPF area).)

- Exit configuration mode and return to privileged EXEC mode.
  Router (config-router)# end
- Save configuration
  Router# write memory

For Router 1:

- Console into the router and enable privileged EXEC mode.
  Router> enable
- Enter configuration mode.
  Router# config terminal
- Enable OSPF.
  Router (config)# router ospf 1
- Assign Router IDs.
  Router (config-router)# router-id 2.2.2.2
- Define OSPF Network Statements
  Router (config-router)# network 192.168.20.0 0.0.0.255 area 0
  Router (config-router)# network 192.168.30.0 0.0.0.255 area 0
- Exit configuration mode and return to privileged EXEC mode.
  Router (config-router)# end
- Save configuration
  Router# write memory

Verify OSPF Configuration:

Verify that OSPF is running and that neighbors have been discovered in every routers.

Router# show ip ospf neighbor

```
Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           1   FULL/BDR        00:00:39    192.168.30.2    GigabitEthernet0/0
Router#
```

Figure 6.22: All connected routers with Router 0

```
Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.1           1   FULL/DR         00:00:32    192.168.30.1    GigabitEthernet0/0
Router#
```

Figure 6.23: All connected routers with Router 1

Step 4: Checking connectivity.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC0	PC3	ICMP		0.000	N	0	(edit)	

Figure 6.3: Packet successfully transmitted form Network A to Network B.

2. RIP routing protocol

Step 1: Setup the network Topology.

Setup network topology like before (in Static routing protocol).

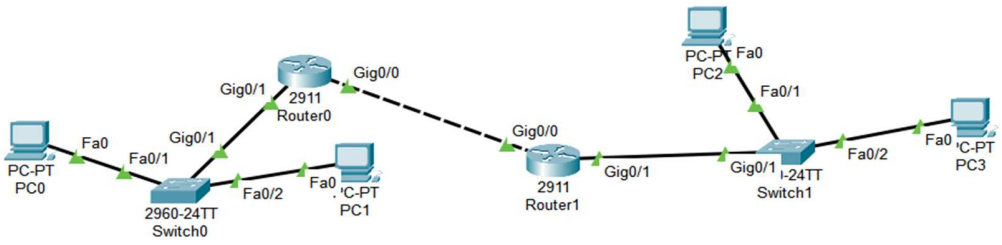


Figure 6.3: Topology of established network for RIP routing protocol.

Step 2: Configure IP addresses for the PCs and routers.

Do this step also like the Static routing experiment, nothing is changed.

Step 3: Implementing OSPF in routers

For Router 0:

- Console into the router and enable privileged EXEC mode.
  Router> enable
- Enter configuration mode.
  Router# config terminal

- Enable RIP

  Router(config)# router rip

- RIP network statement

  Router(config-router)# network 192.168.10.0
  Router(config-router)# network 192.168.30.0

- Using RIP version 2

  Router(config-router)# version 2

For Router 1:

- Console into the router and enable privileged EXEC mode.

  Router> enable
- Enter configuration mode.

  Router# config terminal
- Enable RIP

  Router(config)# router rip

- RIP network statement

  Router(config-router)# network 192.168.20.0
  Router(config-router)# network 192.168.30.0

- Using RIP version 2

  Router(config-router)# version 2

Step 4: Checking connectivity.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC0	PC3	ICMP		0.000	N	0	(edit)	

Figure 6.3: Packet successfully transmitted form Network A to Network B.

Observation and Result

Packet from one router's end device to another routers end device is successfully transmitted and received, meaning OSPF is working as routing protocol here.

Additional reading materials/ online tutorial/ References.

7. Network Address Translation

Title: Network Address Translation (NAT).

Objectives

1. To permit external access to particular internal services by setting up Destination NAT (Port Forwarding).
2. To translate private and public IP addresses one-to-one by implementing static NAT Mappings.
3. To confirm and test the setups for static NAT and port forwarding.
4. To resolve problems related to Static NAT and port forwarding.

Prerequisites

1. Knowledge of how to build network topology, configure the components.
2. Theoretical knowledge of Network address translation.

Theory

Network Address translation

Network Address Translation (NAT) is a process in which one or more local IP addresses are translated into one or more Global IP addresses and vice versa to provide Internet access to the local hosts. It also does the translation of port numbers, i.e., masks the port number of the host with another port number in the packet that will be routed to the destination. It then makes the corresponding entries of IP address and port number in the NAT table. NAT generally operates on a router or firewall.

Working of Network Address Translation (NAT)

Generally, the border router is configured for NAT i.e. the router which has one interface in the local (inside) network and one interface in the global (outside) network. When a packet traverse outside the local (inside) network, then NAT converts that local (private) IP address to a global (public) IP address. When a packet enters the local network, the global (public) IP address is converted to a local (private) IP address.

If NAT runs out of addresses, i.e., no address is left in the pool configured then the packets will be dropped and an Internet Control Message Protocol (ICMP) host unreachable packet to the destination is sent.

Types of NAT:

1. Static NAT
2. Dynamic NAT
3. PAT

Methodology

1. Set up the topology.
2. Configure IP addresses for the PCs and routers.
3. Implementing NAT in routers.
4. Check connectivity.

Equipment

1. 4 Routers (2911)
2. 6 Switch (2960-24TT)
3. 6 End Devices

Procedure:

Step 1: Setup the network Topology.

Connect all component as shown in this figure.

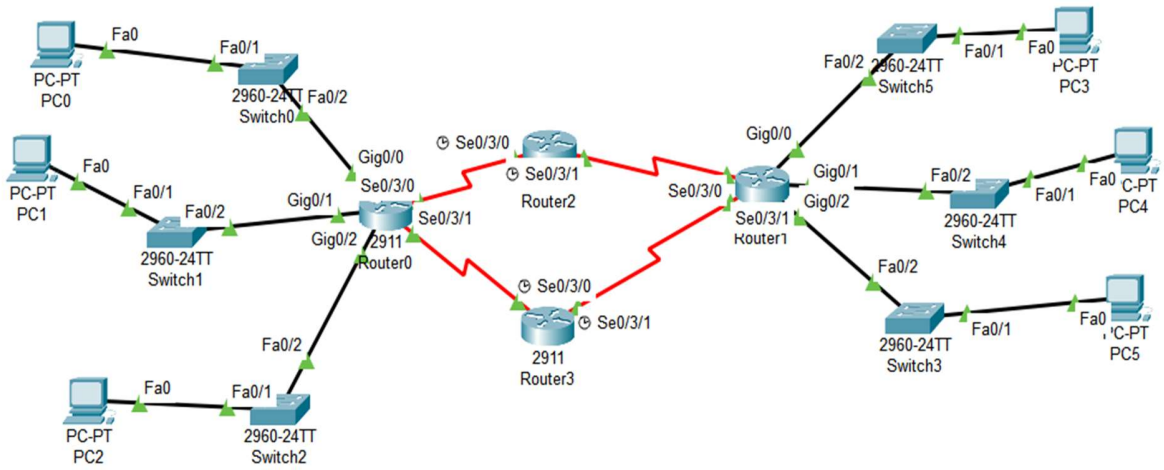


Figure 7.1: Topology of established network.

- Configure IP address, subnet mask and default gateway for the PC.

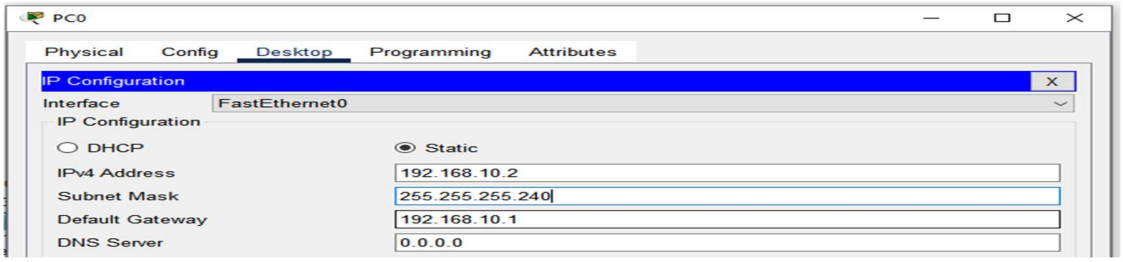


Figure 7.2: IP address, subnet mask and default gateway configured on PC0.

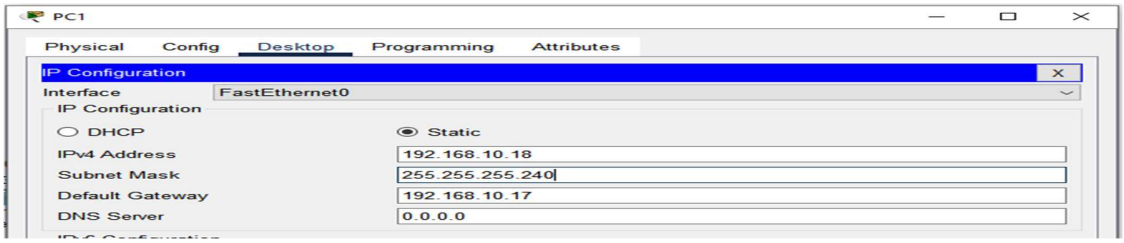


Figure 7.3: IP address, subnet mask and default gateway configured on PC1.

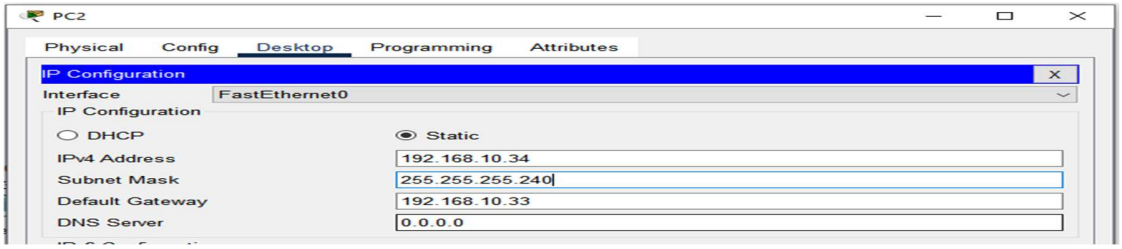


Figure 7.4: IP address, subnet mask and default gateway configured on PC2.

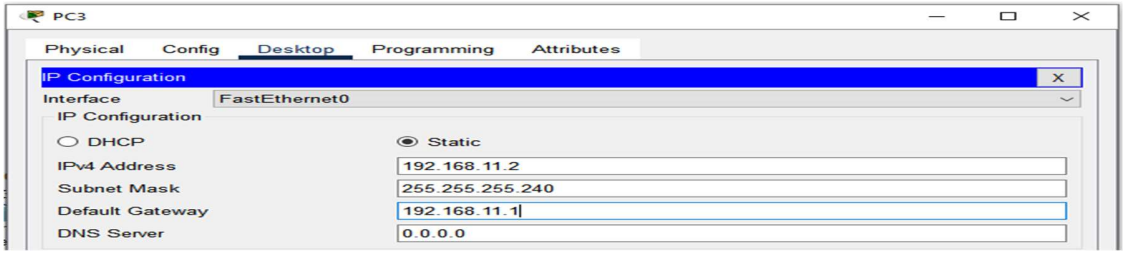


Figure 7.5: IP address, subnet mask and default gateway configured on PC3.

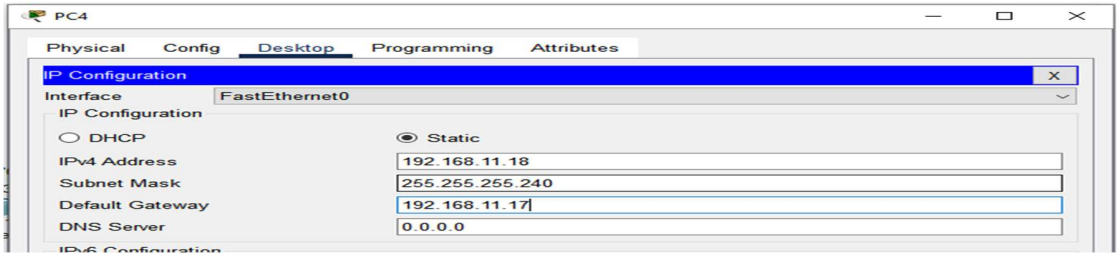


Figure 7.6: IP address, subnet mask and default gateway configured on PC4.

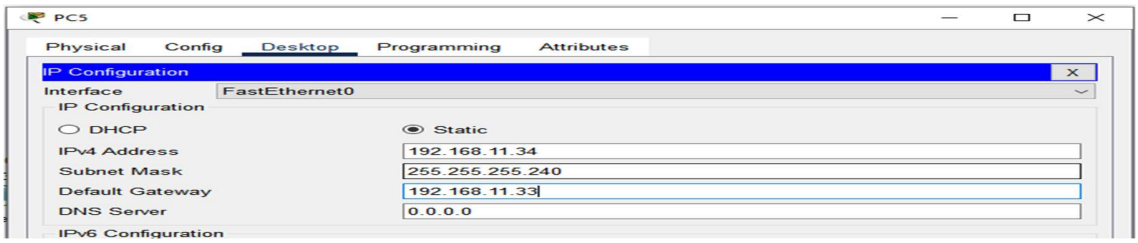


Figure 7.7: IP address, subnet mask and default gateway configured on PC5.

- Turn on every router interface that is connected and configure IP address and subnet mask for all of the interfaces that are being used.

Step 2: Configure Network Address Translation (NAT).

1. Static NAT:

Maps a single private IP address to a single public IP address. This one-to-one mapping is used when a specific internal device needs to be accessible from the outside world, such as a web server. Static NAT configuration requires three steps:

1. Define IP address mapping.
2. Define inside local interface.
3. Define inside global interface.

Since static NAT use manual translation, we have to map each inside local IP address (which needs a translation) with inside global IP address. Following command is used to map the inside local IP address with inside global IP address.

Router(config)# ip nat inside source static [inside local ip address] [inside global IP address]

For example in our lab Laptop1 is configured with IP address 192.168.10.0. To map it with 0.0.0.15 IP address we will use following command

Router(config)# ip nat inside source static 192.168.10.0 0.0.0.15

In second step we have to define which interface is connected with local the network. On both routers interface Fa0/0 is connected with the local network which need IP translation.

Following command will define interface Fa0/0 as inside local.

Router(config-if)# ip nat inside

In third step we have to define which interface is connected with the global network. On both routers serial 0/0/0 interface is connected with the global network. Following command will define interface Serial0/0/0 as inside global.

```
Router(config-if)# ip nat outside
```

Static NAT Configuration

```
Router(config)# interface fastethernet0/0

# Internal network (inside)
Router(config-if)# ip address 192.168.1.1 255.255.255.0
Router(config-if)# ip nat inside
Router(config-if)# exit

Router(config)# interface serial0/0

# Public network (outside)
Router(config-if)# ip address 192.168.10.0 0.0.0.15
Router(config-if)# ip nat outside
Router(config-if)# exit

# Static NAT configuration
Router(config)# ip nat inside source static 192.168.1.10 203.0.113.2

# To verify NAT configuration
Router# show ip nat translations
```

2. Dynamic NAT

Maps a private IP address to a public IP address selected from a pool of public addresses. This type of NAT is used when there are more internal devices than available public IP addresses.
Dynamic NAT configuration requires four steps: -

1. Create an access list of IP addresses which need translation
2. Create a pool of all IP address which are available for translation
3. Map access list with pool
4. Define inside and outside interfaces.

In first step we will create a standard access list which defines which inside local addresses are permitted to map with inside global address.

To create a standard numbered ACL following global configuration mode command is used:
Router(config)# access-list ACL_Identifier_number permit/deny matching-parameters

Let's understand this command and its options in detail.

access-list

Through this parameter we tell router that we are creating or accessing an access list.

ACL_Identifier_number

With this parameter we specify the type of access list. We have two types of access list; standard and extended. Both lists have their own unique identifier numbers. Standard ACL uses numbers range 1 to 99 and 1300 to 1999. We can pick any number from this range to tell the router that we are working with standard ACL. This number is used in groping the conditions under a single ACL. This number is also a unique identifier for this ACL in router.

permit/deny

An ACL condition has two actions; permit and deny. If we use permit keyword, ACL will allow all packets from the source address specified in next parameter. If we use deny keyword, ACL will drop all packets from the source address specified in next parameter.

Matching-parameters

This parameter allows u s to specify the contents of packet that we want to match. In a standard ACL condition it could be a single source address or a range of addresses.

We have three options to specify the source address.

Any: Any keyword is used to match all sources. Every packet compared against this condition would be matched.

Host: Host keyword is used to match a specific host. To match a particular host, type the keyword host and then the IP address of host.

A.B.C.D: Through this option we can match a single address or a range of addresses. To match a single address, simply type its address. To match a range of addresses, we need to use wildcard mask.

In second step we define a pool of inside global addresses which are available for translation. Following command is used to define the NAT pool:

Router(config)# ip nat pool [Pool Name] [Start IP address] [End IP address] netmask [Subnet mask]

This command accepts four options pool name, start IP address, end IP address and Subnet mask.

Pool Name: - This is the name of pool. We can choose any descriptive name here.

Start IP Address: - First IP address from the IP range which is available for translation.

End IP Address: - Last IP address from the IP range which is available for translation.

There is no minimum or maximum criteria for IP range for example we can have a range of single IP address or we can have a range of all IP address from a subnet.

Subnet Mask: - Subnet mask of IP range.

Let's create a pool named ccna with an IP range of two addresses.

R1(config)# ip nat pool ccna 50.0.0.1 50.0.0.2 netmask 255.0.0.0

This pool consist two class A IP address 50.0.0.1 and 50.0.0.2.

In third step we map access list with pool. Following command will map the access list with pool and configure the dynamic NAT.

Router(config)# ip nat inside source list [access list name or number] pool [pool name]

This command accepts two options.

1. Access list name or number: - Name or number the access list which we created in first step.

2. Pool Name: - Name of pool which we created in second step.

In first step we created a standard access list with number **1** and in second step we created a pool named **ccna**. To configure a dynamic NAT with these options we will use following command:

R1(config)# ip nat inside source list 1 pool ccna

Finally we have to define which interface is connected with local network and which interface is connected with global network.

To define an inside local we use following command:

Router(config-if)# ip nat inside

Following command defines inside global:

Router(config-if)# ip nat outside

Dynamic NAT Configuration

```
# Define private network
Router(config)# access-list 1 permit 192.168.1.0 0.0.0.15

# Define public IP pool
Router(config)# ip nat pool SEC_B 200.200.200.1 200.200.200.100 netmask 255.255.255.0

Router(config)# ip nat inside source list 1 pool SEC_B

Router(config)# interface fastethernet0/0
Router(config-if)# ip address 192.168.1.1 255.255.255.0
Router(config-if)# ip nat inside
Router(config-if)# exit

Router(config)# interface serial0/0
Router(config-if)# ip nat outside
Router(config-if)# exit

# To verify NAT configuration
Router# show ip nat translations
```

3. Port Address Translation (PAT)

Also known as "overloading," PAT maps multiple private IP addresses to a single public IP address using different ports. This is the most common form of NAT used in home routers and small networks.

PAT configuration requires four steps:

1. Create an access list of IP addresses which need translation
2. Create a pool of all IP address which are available for translation
3. Map access list with pool
4. Define inside and outside interfaces

In first step, we will create a standard access list which defines which inside local addresses are permitted to map with inside global address.

To create a standard numbered ACL following global configuration mode command is used:

```
Router(config)# access-list ACL_Identifier_number permit/deny matching-parameters
```

In second step, we define a pool of inside global addresses which are available for translation. Following command is used to define the NAT pool:

```
Router(config)#ip nat pool [Pool Name] [Start IP address] [End IP address] netmask [Subnet mask]
```

In third step, we map access list with pool. Following command will map the access list with pool and configure the PAT:

```
Router(config)#ip nat inside source list [access list name or number] pool [pool name]overload
```

In first step, we created a standard access list with number 1 and in second step we created a pool named ccna. To configure a PAT with these options we will use following command:

```
R1(config)#ip nat inside source list 1 pool ccna overload
```

Finally we have to define which interface is connected with local network and which interface is connected with global network.

To define an inside local we use following command:

```
Router(config-if)#ip nat inside
```

Following command defines inside global:

```
Router(config-if)#ip nat outside
```

PAT Configuration

```
# Define private network
Router(config)# access-list 1 permit 192.168.10.0 0.0.0.15

# Configure PAT (overload)
Router(config)# ip nat inside source list 1 interface serial0/0 overload

Router(config)# interface fastethernet0/0
Router(config-if)# ip address 192.168.10.0 255.255.255.0
Router(config-if)# ip nat inside
Router(config-if)# exit

Router(config)# interface serial0/0
Router(config-if)# ip nat outside
Router(config-if)# exit

# To verify NAT configuration
Router# show ip nat translations
```

Observation and Result

Transmitted a packet from PC6 to PC3 through R1:

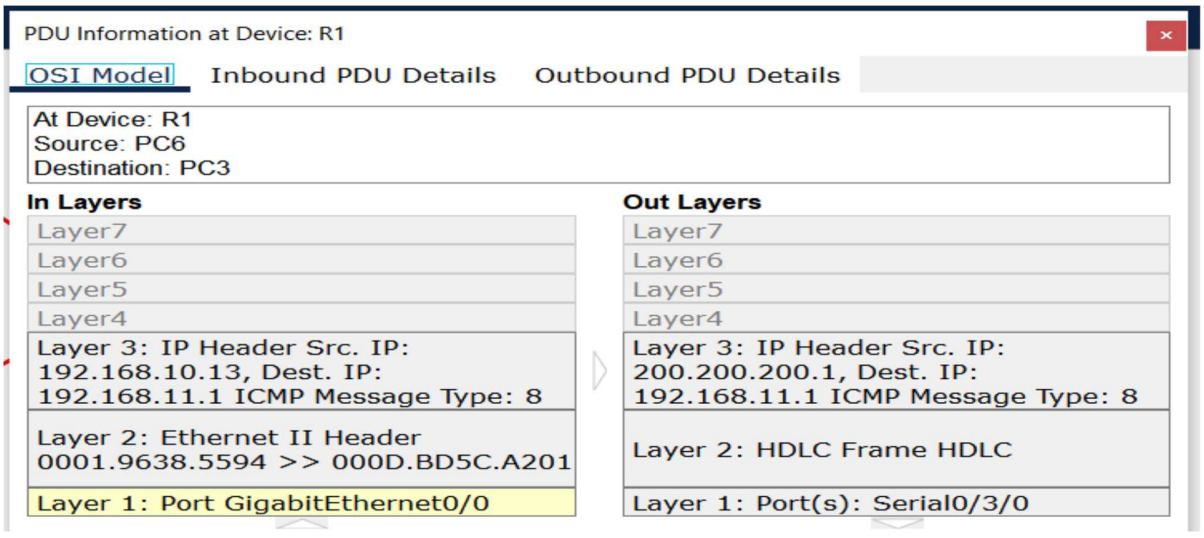


Figure 7.7: PDU Information at R1

The overall activity demonstrated successful NAT configuration with all types of NAT (Static, Dynamic, and PAT) working as expected, allowing internal network devices to access external networks through IP translation.

8. Access Control List

Title: Access Control List.

Objectives

1. To configure and verify ACLs to control traffic.
2. To verify ACLs using the logging capabilities of the router.

Prerequisites

1. Knowledge of how to build network topology, configure the components.
2. Knowledge of routing and switching..

Theory

Access Control List

Access Control Lists (ACL) are used to filter network traffic on Cisco routers. In order to filter network traffic, ACLs control if routed packets have to be forwarded or blocked at the ingress or egress router interface. The router checks each packet to determine whether to forward or drop the packet based on the criteria specified in the ACL applied to the interface.

Two types of IP ACL can be configured in Cisco Packet:

Standard ACLs: This is the oldest ACL type which can be configured on Cisco routers. Traffic is filtered based on the source IP address of IP packets. The access-list number can be any number from 1 to 99. This kind of ACL has to be placed near the destination to avoid blocking legitimate traffic from the source.

```
Router(config)# ip access-list standard ACL_name
Router(config-std-acl)# permit|deny source_IP_address [wildcard_mask]
```

Extended ACLs: The extended ACLs are more complex and allow filtering of the IP traffic based on a combination of multiple criteria: source IP address, destination IP address, TCP or UDP port, protocol, etc. In numbered ACLs, the access-list number can be any number from 100 to 199 or 2000 to 2699. Such ACLs can also be named access lists in which the ACL number is replaced by a keyword. This kind of ACL has to be placed near the source as it allows fine grained control to resources accessed. Placing the ACL near the destination will make the traffic travel through the network before being blocked, resulting in bandwidth waste.

```
access-list 1 permit ip 10.2.25.0 0.0.0.255 10.1.0.0 0.0.255.255
access-list 101 permit icmp any 10.1.0.0 0.0.255.255 echo
access-list 1 deny ip any any
```

Methodology

1. Set up the topology.
2. Configure IP addresses for the PCs and routers.
3. Applying ACL.
4. Testing the configuration.

Equipment

1. 3 Routers (2911)
2. 2 Switch (2960-24TT)
3. 4 End Devices
4. 1 Server

Procedure:

Step 1: Setup the network Topology.

- Connect all component as shown in this figure.

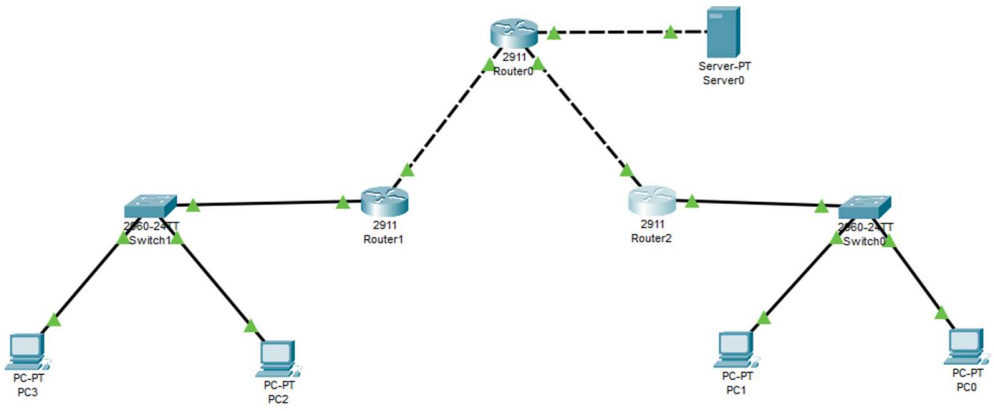


Figure 8.1: Topology of established network.

- Configure IP address, subnet mask and default gateway for the PC.

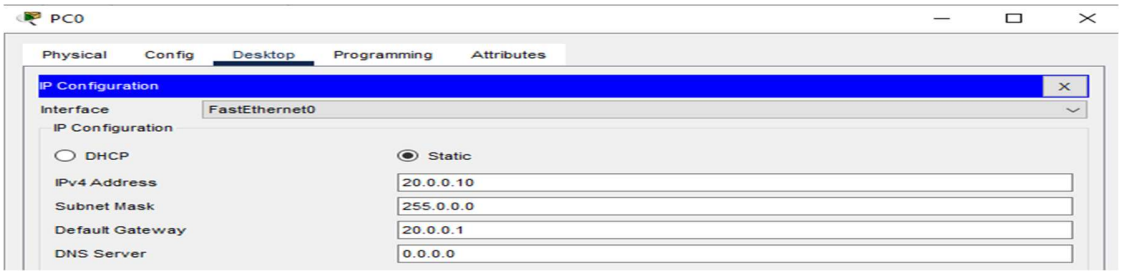


Figure 8.2: IP address, subnet mask and default gateway configured on PC0.

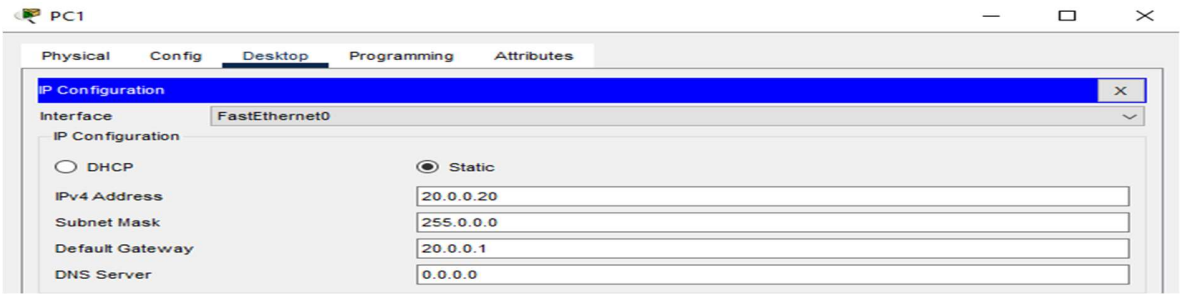


Figure 8.3: IP address, subnet mask and default gateway configured on PC1

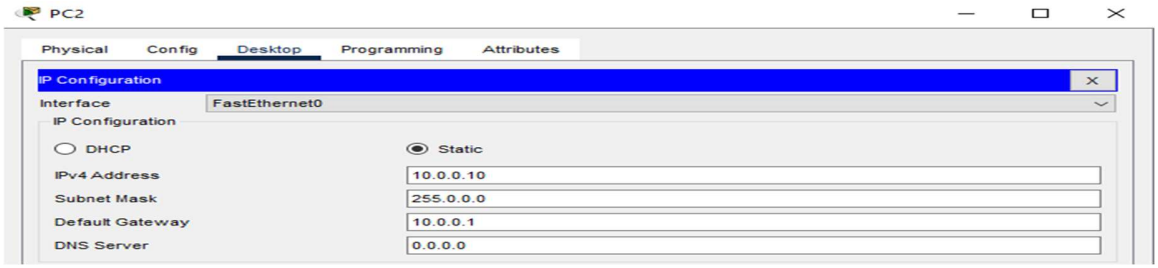


Figure 8.4: IP address, subnet mask and default gateway configured on PC2.

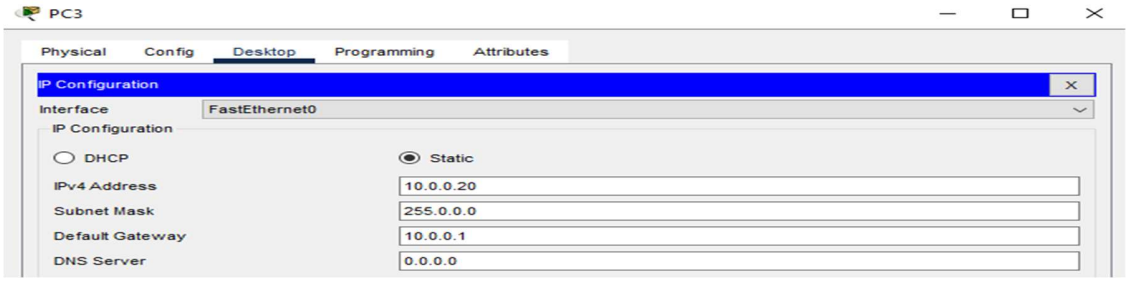


Figure 8.5: IP address, subnet mask and default gateway configured on PC3.

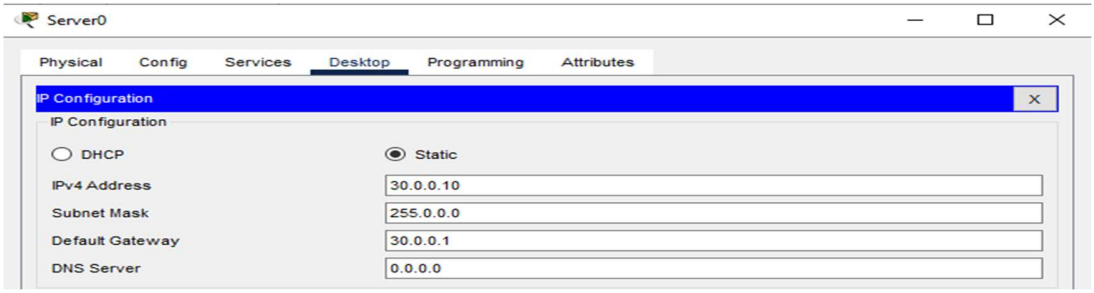


Figure 8.6: IP address, subnet mask and default gateway configured on Server0

Step 2: Check connectivity.

- There are two networks: one is the student network, and the other is the teacher network. PC2 and PC3 are devices on the student network, while PC0 and PC1 are devices on the teacher network. We are pinging server0 from both networks to check connectivity.

Check connectivity between PC3 to Server0(Student network):

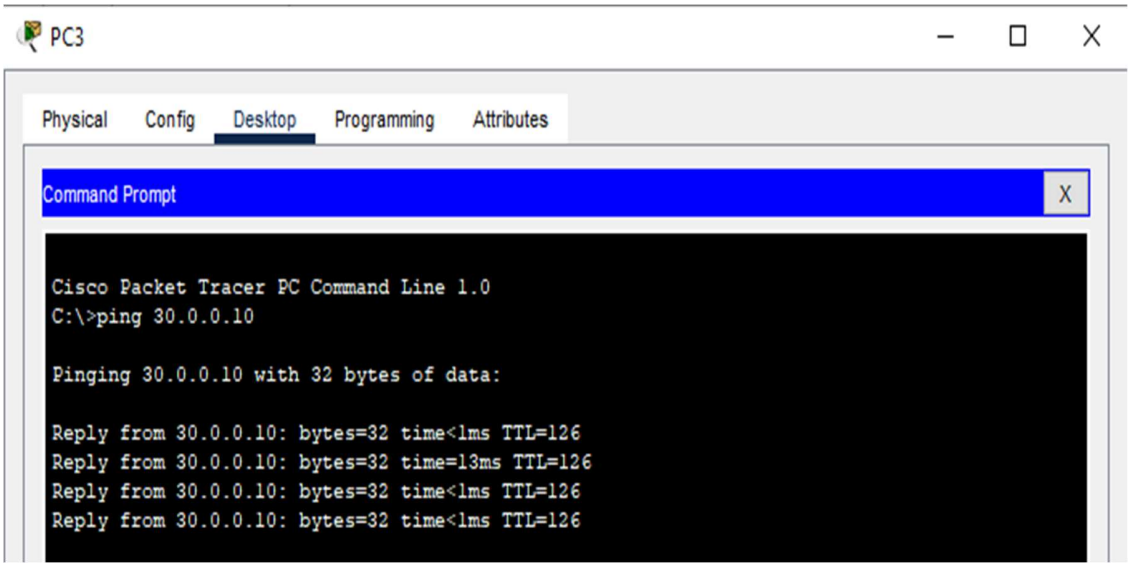


Figure 8.7: Ping Server0 from PC3.

- Check connectivity between PC0 to Server0(Teacher Network):

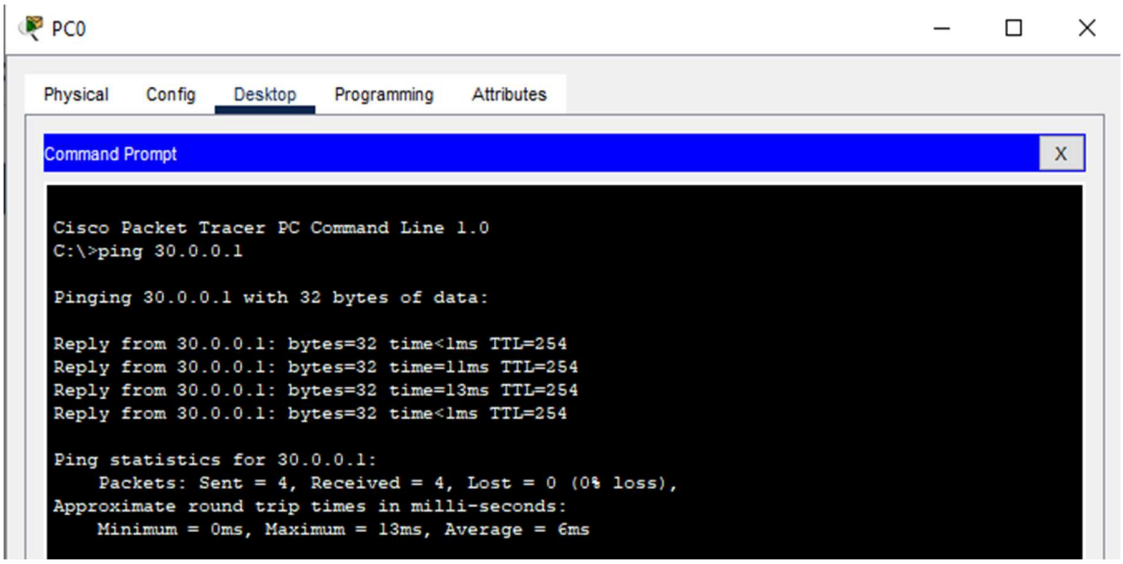


Figure 8.7: Ping Server0 from PC0.

Step 3: Create & verify Access Control List.

1. Standard ACL:

Creating a standard ACL

```
Router>
Router>enable
Router#configure terminal
Router(config)#ip access-list standard BlockStudents
Router(config-std-nacl)#deny 10.0.0.0 0.255.255.255
Router(config-std-nacl)#permit any
Router(config-std-nacl)#exit
Router(config)#interface gigabitethernet 0/0
Router(config-if)#ip access-group BlockStudents out
Router(config-if)#exit
Router(config)#exit
```

"ip access-list standard BlockStudents" creates a standard ACL named **BlockStudents**. In ACL configuration mode, we added two statements. The first statement denies all traffic from the 10.0.0.0/8 subnet. The second statement allows all other traffic. The next command applies the **BlockStudents** ACL in the out direction of interface gigabitEthernet 0/0.

Verifying

To verify the ACL, we can test connectivity between sections. The Students section should not be able to access the Server section but it should be able to access the Teachers section. The Teachers section should be able to access both the Server and the Students section.

Student Network:

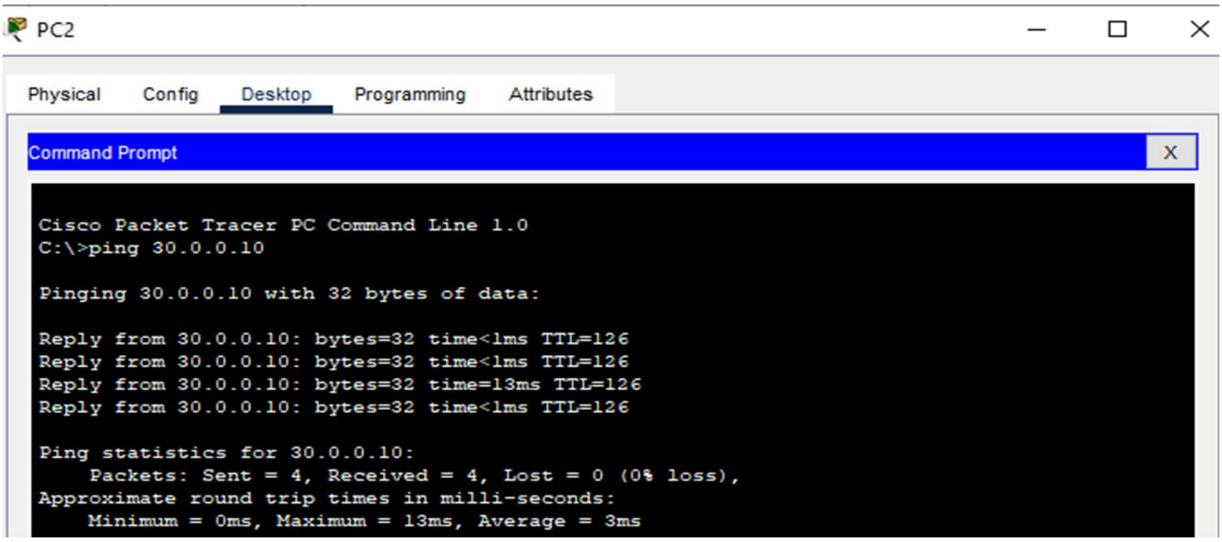


Figure 8.8: Ping Server0 from PC2 before applying ACL.

```
C:\>ping 30.0.0.10

Pinging 30.0.0.10 with 32 bytes of data:

Reply from 192.168.1.2: Destination host unreachable.
Reply from 192.168.1.2: Destination host unreachable.
Reply from 192.168.1.2: Destination host unreachable.
Reply from 192.168.1.2: Destination host unreachable.

Ping statistics for 30.0.0.10:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 8.8: Ping Server0 from PC2 after applying ACL.

Teacher Network:

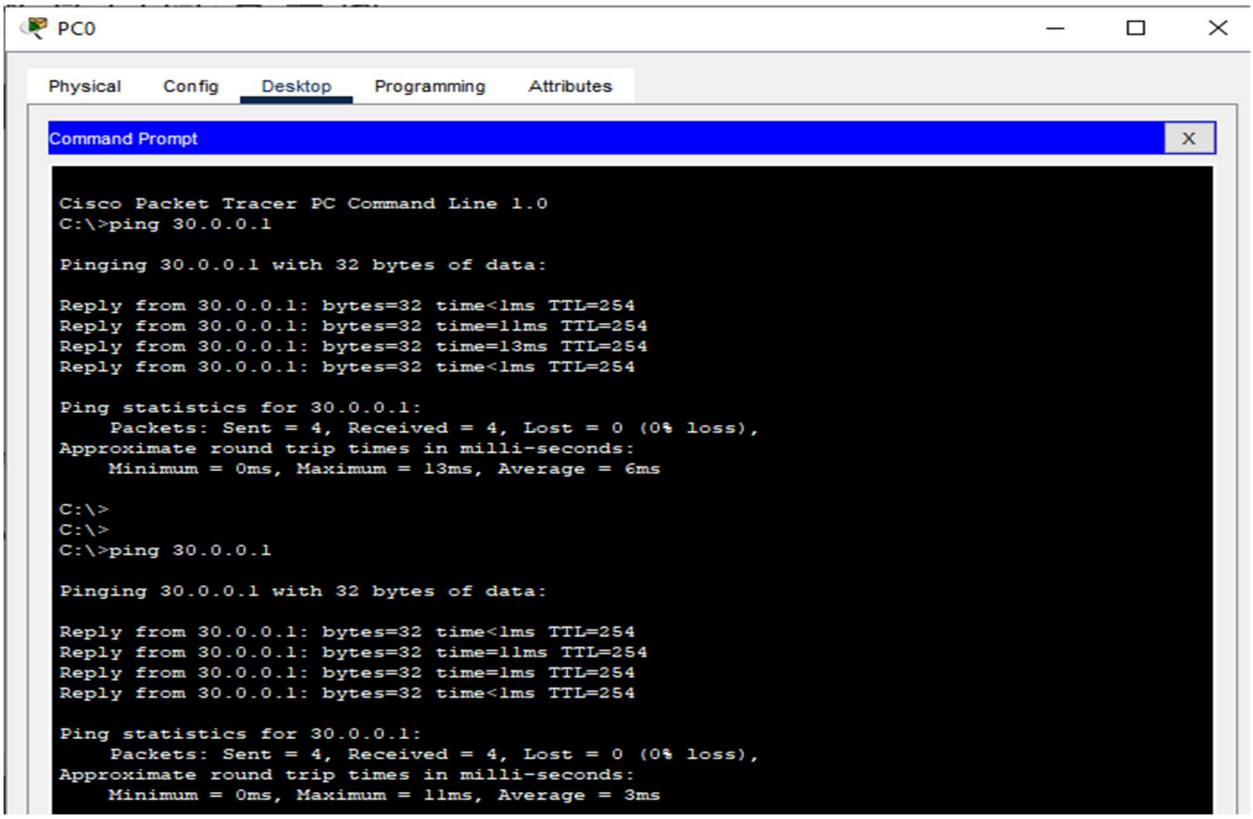


Figure 8.8: Ping Server0 from PC0 before & after applying ACL.

The '**show access-lists**' command to view the sequence number of the statement:

```
Router#show access-lists
Standard IP access list BlockStudents
    10 deny 10.0.0.0 0.255.255.255 (8 match(es))
    20 permit any
```

Now instead of blocking the entire subnet we only want to block a single host (10.0.0.10/8) from the Students section:

```
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#ip access-list standard BlockStudents
Router(config-std-nacl)#no 10
Router(config-std-nacl)#10 deny 10.0.0.10 0.0.0.0
Router(config-std-nacl)#exit
Router(config)#exit
```

Verifying

Since the ACL is already active on the interface, the interface begins using the new statement as soon as it is added.

```
Router#show access-lists
Standard IP access list BlockStudents
    10 deny host 10.0.0.10
    20 permit any
```

To verify the change, send ping requests again from both the blocked host and the allowed host PC2 & PC3 from student network:

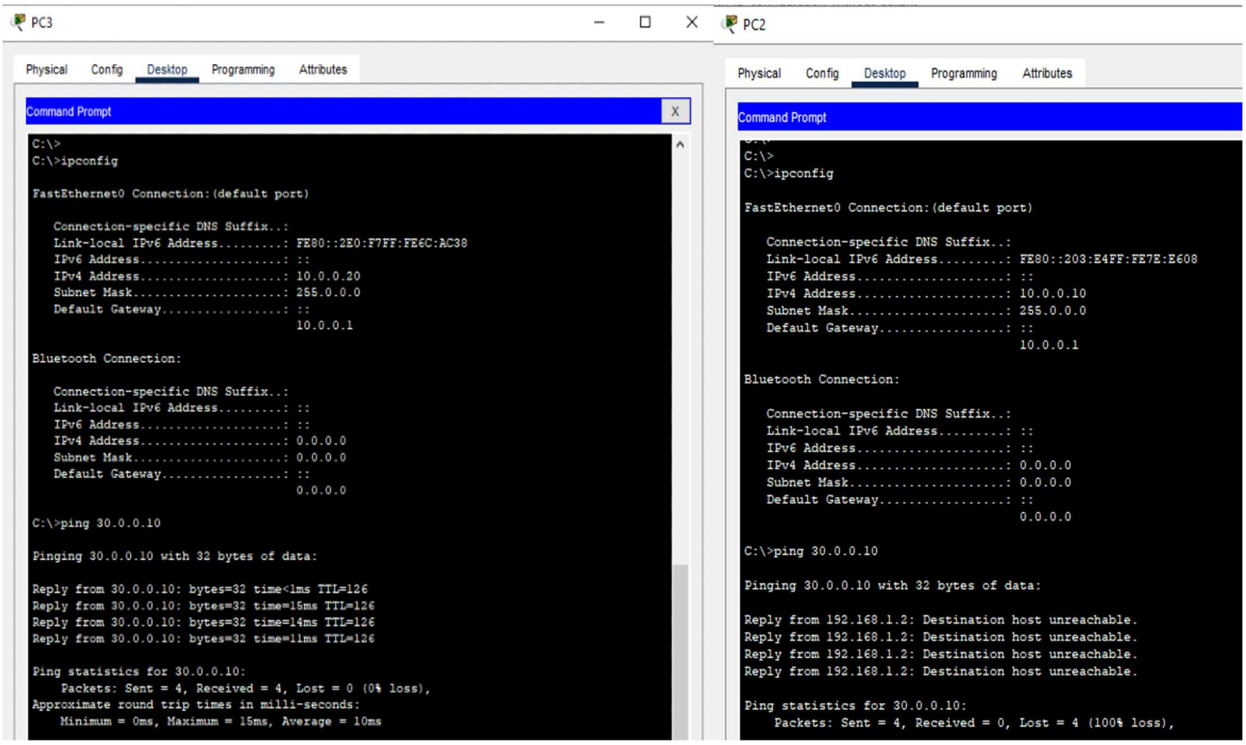


Figure 8.9: Ping Server0 from PC2 & PC3 after modifying standard ACL.

Deleting a standard ACL

The following command deletes the BlockStudents ACL.

```
Router(config# no ip access-list standard BlockStudents
```

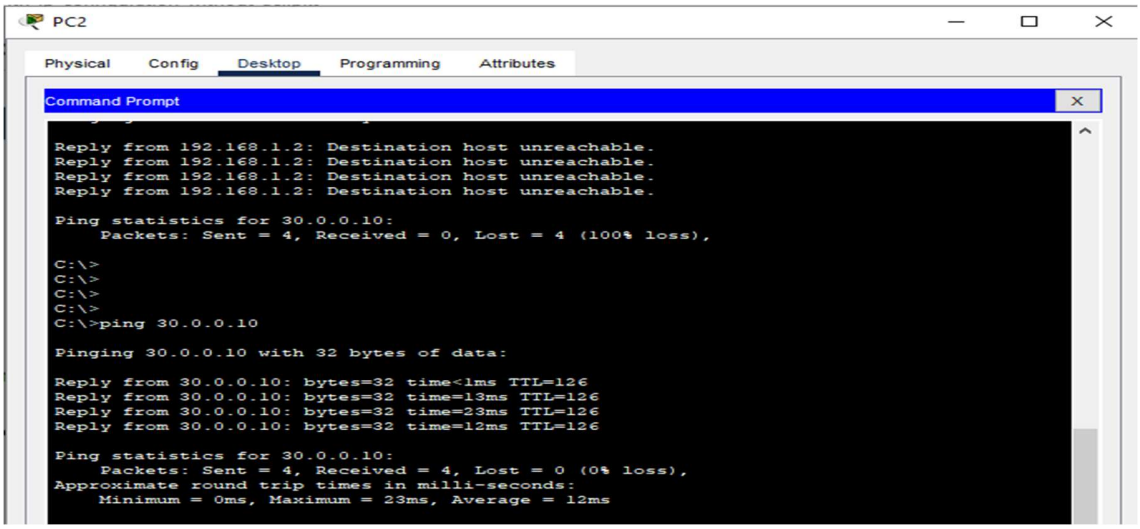


Figure 8.9: Ping Server0 from PC2 after deleting standard ACL.

2. Extended ACL

Creating a standard ACL

```
Router>enable
Router#configure terminal

Router(config)# ip access-list extended BlockStudent
Router(config-ext-nacl)# deny ip 10.0.0.0 0.255.255.255 any Router(config-ext-nacl)# permit ip
20.0.0.0 0.255.255.255 any
Router(config-ext-nacl)# exit

Router(config)# interface gigabitEthernet 0/0
Router(config-if)# ip access-group BlockStudent in
Router(config-if)# exit
```

Verifying

There are two networks: one is the student network, and the other is the teacher network. PC2 and PC3 are devices on the student network, while PC0 and PC1 are devices on the teacher network.

Figure 8.10: PC2 access webpage from Server0 before applying extended ACL.

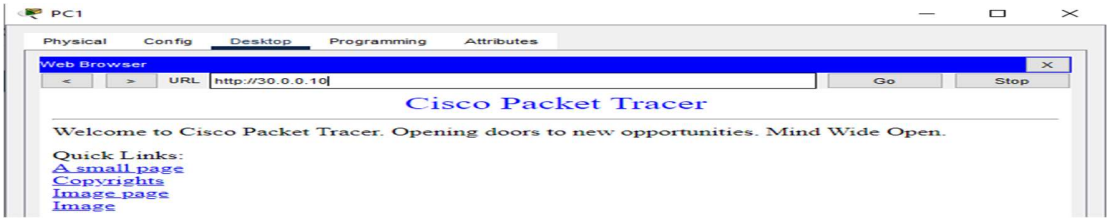


Figure 8.11: PC1 access webpage from Server0 before applying extended ACL.

To verify the ACL, we can test connectivity between sections. The Students section should not be able to access the Server section but it should be able to access the Teachers section. The Teachers section should be able to access both the Server and the Students section.

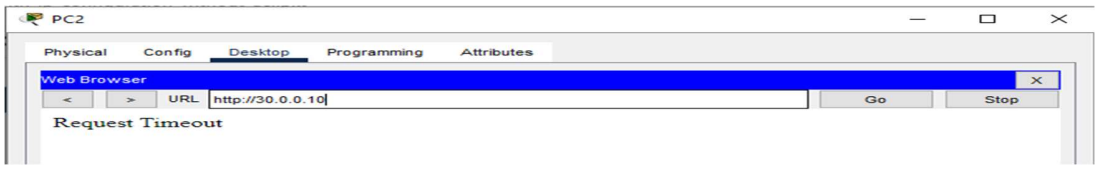


Figure 8.12: PC2 can't access webpage from Server0 after applying extended ACL.

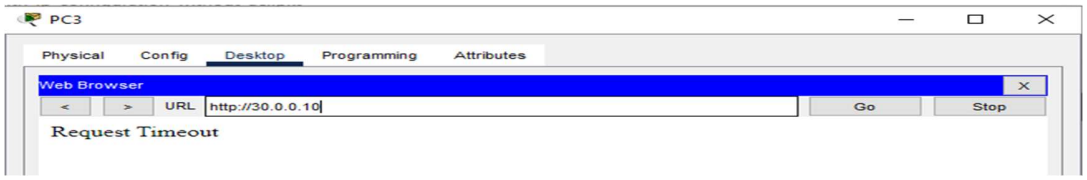


Figure 8.13: PC3 can't access webpage from Server0 after applying extended ACL.

Figure 8.14: PC1 can access webpage from Server0 after applying extended ACL.

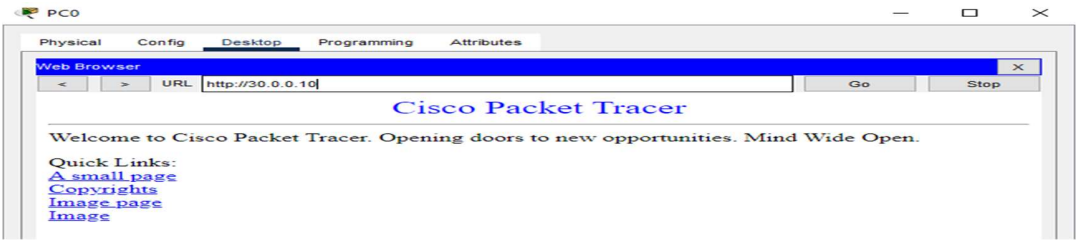


Figure 8.15: PC0 access webpage from Server0 after applying extended ACL.

Now instead of blocking the entire subnet we only want to block a single host (10.0.0.10/8) from the Students section:

```
Router(config)# ip access-list extended BlockOneStudent
Router(config-ext-nacl)# deny ip host 10.0.0.10 any
Router(config-ext-nacl)# permit ip any any
Router(config-ext-nacl)# exit
Router(config)# interface gigabitEthernet 0/0
Router(config-if)# ip access-group BlockOneStudent in
Router(config-if)# exit
```

To verify the change, access webpage from Server0 again from both the blocked host and the allowed host PC2 & PC3 from student network:

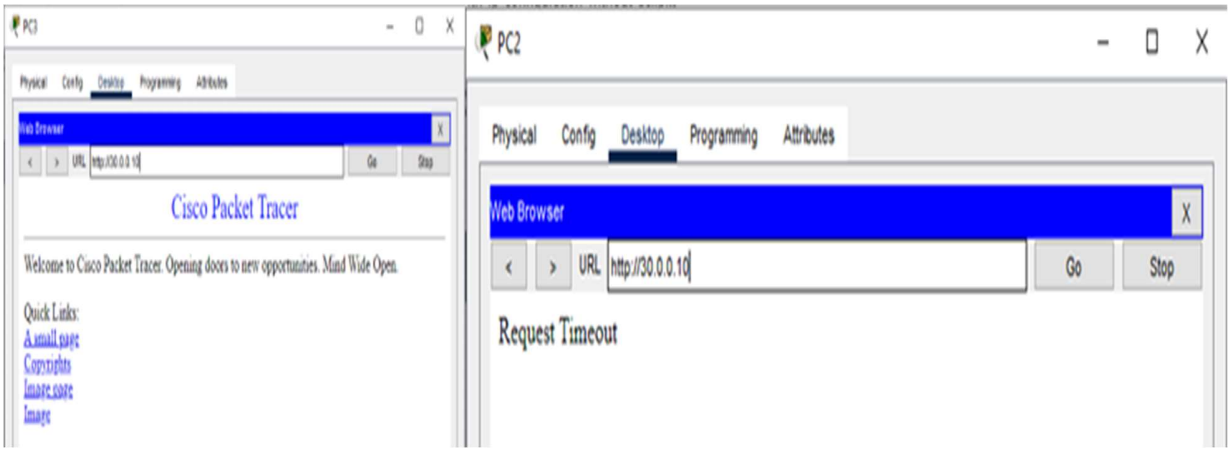


Figure 8.16: PC2 & PC3 access webpage after modifying extended ACL.

To delete the extended ACL named BlockOneHost:

```
Router(config)# no ip access-list extended BlockOneStudent
```

Additional reading materials/ online tutorial/ References

1. Standard & Extended ACL: https://www.computernetworkingnotes.com/ccna-study-guide/

9. Inter-VLAN routing

Title: Inter-VLAN routing.

Objectives

1. To build the network and configure basic device settings.
2. To create VLANs and assign switch port.
3. To maintain VLANs port assignments and the VLAN database.
4. To configure a Trunk between the switches.

Prerequisites

1. Knowledge of how to build network topology, configure the components.
2. Theoretical knowledge of Inter-VLAN.

Theory

VLAN

A Virtual LAN (VLAN) is simply a logical LAN, just as its name suggests. VLANs have similar characteristics to those of physical LANs, only that with VLANs, you can logically group hosts even if they are physically located on separate LAN segments.

We treat each VLAN as a separate subnet or broadcast domain. For this reason, to move packets from one VLAN to another, we have to use a router or a layer 3 switch.

VLANs are configured on switches by placing some interfaces into one broadcast domain and some interfaces into another. For this tutorial, we'll configure 2 VLANs on a switch. We will then proceed and configure a router to enable communication between the two VLANs.

Methodology

1. Set up the topology.
2. Configure IP addresses for the PCs.
3. Create VLANs on the switch.
4. Assign switch ports to the VLANs.
5. Configure inter-VLAN routing on the router.
6. Testing the connectivity.

Equipment

1. 1 Routers (2911)
2. 3 Switch (2960-24TT)
3. 4 End Devices

Procedure:

Step 1: Setup the network Topology.
Connect all component as shown in this figure.

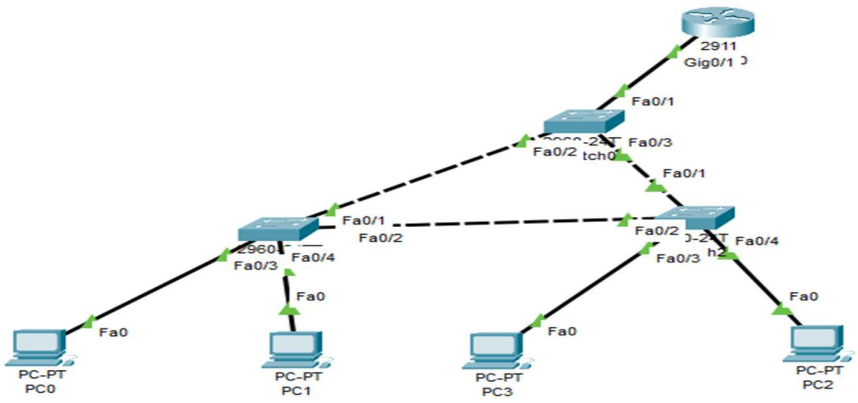


Figure 9.1: Topology of established network.

Step 2: Configure IP address, subnet mask and default gateway for the PC.

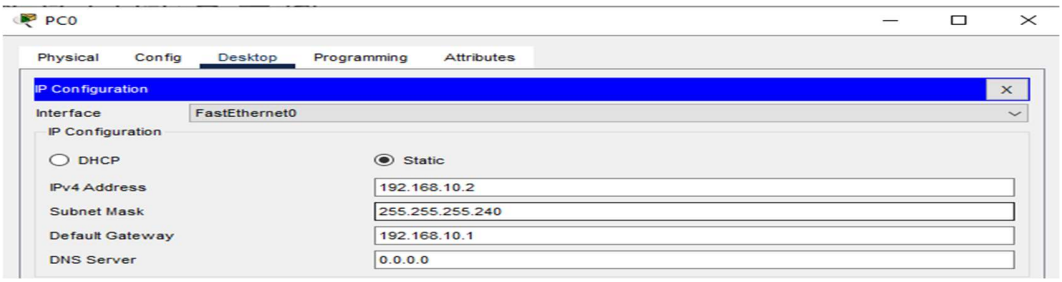


Figure 9.2: IP address, subnet mask and default gateway configured on PC0.

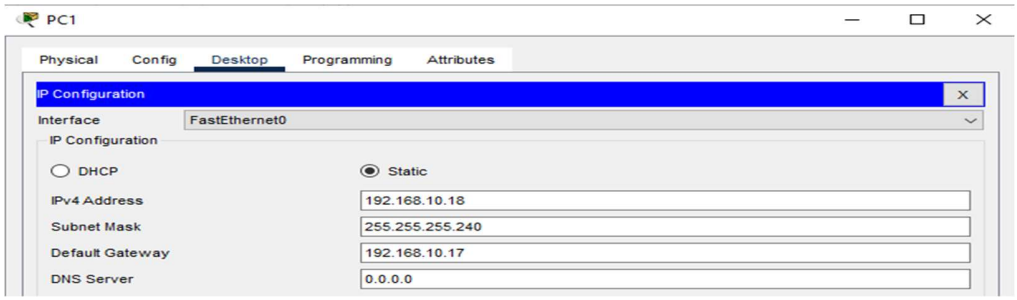


Figure 9.2: IP address, subnet mask and default gateway configured on PC1.

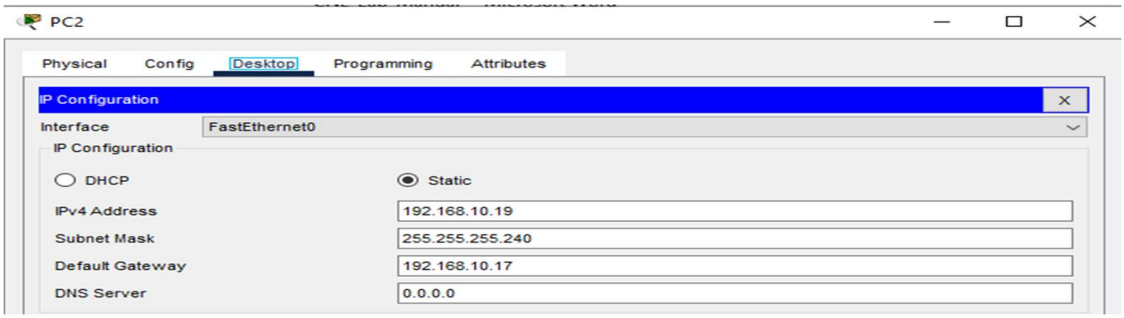


Figure 9.2: IP address, subnet mask and default gateway configured on PC2.

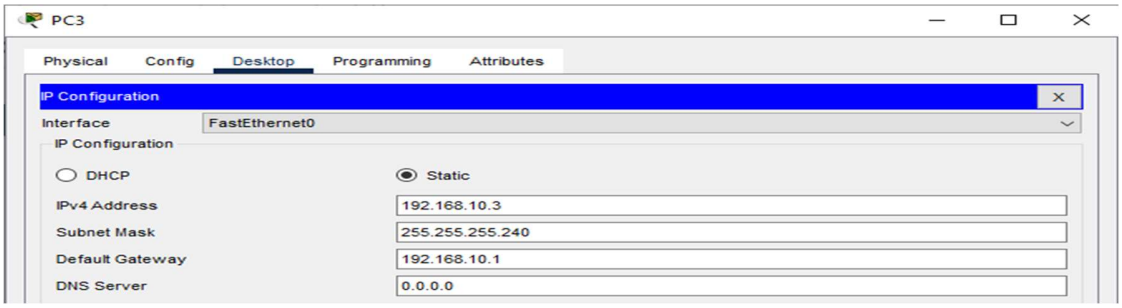


Figure 9.2: IP address, subnet mask and default gateway configured on PC3.

Step 3: Create 3 VLANs on the switch:

- VLAN 10, VLAN 20, and VLAN 99. We can give them names student, faculty and native respectively.

Creating VLAN in switch0,switch1, switch2

```
Switch>enable
Switch#config terminal

Switch(config)#vlan 10
Switch(config-vlan)#name student
Switch(config-vlan)#exit

Switch(config)#vlan 20
Switch(config-vlan)#name faculty
Switch(config-vlan)#exit

Switch(config)#vlan 99
Switch(config-vlan)#name native
Switch(config-vlan)#exit
```

Assign switch ports to the VLANs. Remember each VLAN is viewed as a separate broadcast domain. And just before we configure, keep in mind that switch ports could be either access or trunk.

- An access port is assigned to a single VLAN. These ports are configured for switch ports that connect to devices with a normal network card, for example, a PC in a network.
- A trunk port on the other hand is a port that can be connected to another switch or router. This port can carry traffic of multiple VLANs.

Configuring access and trunk ports in switche0, switch1, switch2

```
Switch(config)#interface fastEthernet 0/3
Switch(config-if)#switchport access vlan 10
Switch(config-if)#exit

Switch(config)#interface fastEthernet 0/1
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk native vlan 99
Switch(config-if)#exit
```

In our topology, we will configure switch interfaces fa0/3 to VLAN10. This access ports Switch port fa0/1 will be configured as a trunk which will carry traffic between VLANs (10, 20).

Show command for verification of VLAN database

```
Switch>
Switch>show vlan brief

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

Show command trunk port configuration

```
Switch>show interfaces trunk
Port      Mode         Encapsulation        Status         Native vlan
Fa0/1     on           802.1q               trunking       99
Fa0/2     on           802.1q               trunking       99
Fa0/3     on           802.1q               trunking       99
Port                                        Vlans allowed on trunk
Fa0/1                                       1-1005
Fa0/2                                       1-1005
Fa0/3                                       1-1005

Port                                        Vlans allowed and active in management domain
Fa0/1                                       1,10,20,99
Fa0/2                                       1,10,20,99
Fa0/3                                       1,10,20,99

Port          Vlans in spanning tree forwarding state and not pruned
Fa0/1                                       1,10,20,99
Fa0/2                                       1,10,20,99
Fa0/3                                       1,10,20,99

Switch>
```

Step 4: Configure inter-VLAN routing on the router

Router configuration (Router on stick)

Now we will configure the router so that it will enable communication between the two VLANs via a single physical interface. We'll divide the single physical interface on the router into logical interfaces (sub-interfaces). Each sub-interface will then serve as a default gateway for each of the VLANs. This scenario is called a router on a stick (R.O.A.S) and will allow the VLANs to communicate through a single physical interface.

Worth noting: We can't assign an IP address to the router's physical interface that we have subdivided into logical sub-interfaces. We'll instead assign IP addresses to the sub-interfaces.

Configuring interface in the router

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface gigabitEthernet 0/1.10
Router(config-subif)#encapsulation dot1Q 10
Router(config-subif)#ip address 192.168.10.1 255.255.255.240
Router(config-subif)#exit
Router(config)#interface gigabitEthernet 0/1.20
Router(config-subif)#encapsulation dot1Q 20
Router(config-subif)#ip address 192.168.10.17 255.255.255.240
Router(config-subif)#exit
Router(config)#interface gigabitEthernet 0/1.99
Router(config-subif)#encapsulation dot1Q 99 native
Router(config-subif)#exit
```

Show command for verification of sub-interfaces

```
Router>show ip interface brief
Interface              IP-Address      OK? Method  Status          Protocol
GigabitEthernet0/0     unassigned      YES unset   administratively down  down
GigabitEthernet0/1     unassigned      YES unset   up              up
GigabitEthernet0/1.10  192.168.10.1    YES manual  up              up
GigabitEthernet0/1.20  192.168.10.17   YES manual  up              up
GigabitEthernet0/1.99  unassigned      YES unset   up              up
GigabitEthernet0/2     unassigned      YES unset   administratively down  down
Vlan1                  unassigned      YES unset   administratively down  down

Router>
Router>show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter-area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     192.168.10.0/24 is variably subnetted, 4 subnets, 2 masks
C          192.168.10.0/28 is directly connected, GigabitEthernet0/1.10
L          192.168.10.1/32 is directly connected, GigabitEthernet0/1.10
C          192.168.10.16/28 is directly connected, GigabitEthernet0/1.20
L          192.168.10.17/32 is directly connected, GigabitEthernet0/1.20

Router>
```

Step 5: Test inter-VLAN connectivity

- VLAN 10 : 192.168.10.0/28 , Gateway-192.168.10.1,
  PC0- 192.168.10.2, PC3- 192.168.10.3.
- VLAN 20 : 192.168.0.16/28, Gateway-192.168.0.17,
  PC1- 192.168.10.18, PC2-192.168.10.19.

```
Router>
C:\>ipconfig /all

FastEthernet0 Connection:(default port)

Connection-specific DNS Suffix..:
  Physical Address.....: 0090.2B26.724B
  Link-local IPv6 Address.....: FE80::290:2BFF:FE26:724B
  IPv6 Address.....: ::
  IPv4 Address.....: 192.168.10.2
  Subnet Mask.....: 255.255.255.240
  Default Gateway.....: ::
                                        192.168.10.1
  DHCP Servers.....: 0.0.0.0
  DHCPv6 IAID.....:
```

```
  DHCPv6 Client DUID.....: 00-01-00-01-5B-CB-84-55-00-90-2B-26-72-4B
  DNS Servers.....: ::
                              0.0.0.0

Bluetooth Connection:

  Connection-specific DNS Suffix..:
  Physical Address.....: 0090.0CC9.1D36
  Link-local IPv6 Address.....: ::
  IPv6 Address.....: ::
  IPv4 Address.....: 0.0.0.0
  Subnet Mask.....: 0.0.0.0
  Default Gateway.....: ::
                              0.0.0.0
  DHCP Servers.....: 0.0.0.0
  DHCPv6 IAID.....:
  DHCPv6 Client DUID.....: 00-01-00-01-5B-CB-84-55-00-90-2B-26-72-4B
  DNS Servers.....: ::
                              0.0.0.0


C:\>
C:\>ping 192.168.10.19

Pinging 192.168.10.19 with 32 bytes of data:

Request timed out.
Reply from 192.168.10.19: bytes=32 time=1ms TTL=127
Reply from 192.168.10.19: bytes=32 time<1ms TTL=127
Reply from 192.168.10.19: bytes=32 time<1ms TTL=127

Ping statistics for 192.168.10.19:
        Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>ping 192.168.10.3

Pinging 192.168.10.3 with 32 bytes of data:

Reply from 192.168.10.3: bytes=32 time<1ms TTL=128
Reply from 192.168.10.3: bytes=32 time=1ms TTL=128
Reply from 192.168.10.3: bytes=32 time<1ms TTL=128
Reply from 192.168.10.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.10.3:
        Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>
```

We have successfully pinged the PC in VLAN 10 and VLAN 20 from VLAN 10. If everything is well configured, then we will see a ping reply as shown above.

Additional reading materials/ online tutorial/ References

1. https://medium.com/@minwork/
2. https://computernetworking747640215.wordpress.com/

11. Wireless Router Configuration

Title: Wireless router configuration

Objectives

1. To configure wireless router.
2. To check connectivity.

Prerequisites

1. Knowledge of how to build network topology, configure the components.
2. Theoretical knowledge of wireless communication.

Theory

Wireless routing

Wireless router configuration is a process that involves setting up and managing the settings of a wireless router to enable wireless network connectivity for devices like smartphones, laptops, and other Wi-Fi-enabled devices. This configuration process is essential for establishing a secure and efficient wireless network in homes, offices, or public space.

Network Name (SSID)

The SSID (Service Set Identifier) is the name of the Wi-Fi network. Configuring this helps in identifying the network from other nearby wireless networks.

Wireless Security

Setting up security protocols such as WPA3, WPA2, or WPA to protect the network from unauthorized access. This often involves setting a strong password.

Network Mode and Channel

Users can choose the network mode (e.g., 802.11ac, 802.11n) and select a channel to minimize interference from other networks.

DHCP Server

The router can be configured to act as a DHCP server, automatically assigning IP addresses to devices on the network.

Port Forwarding and DMZ

Advanced configurations like port forwarding and setting up a DMZ (Demilitarized Zone) for specific devices can be done for better control over network traffic and security.

Procedure:

Step 1: Create a network topology

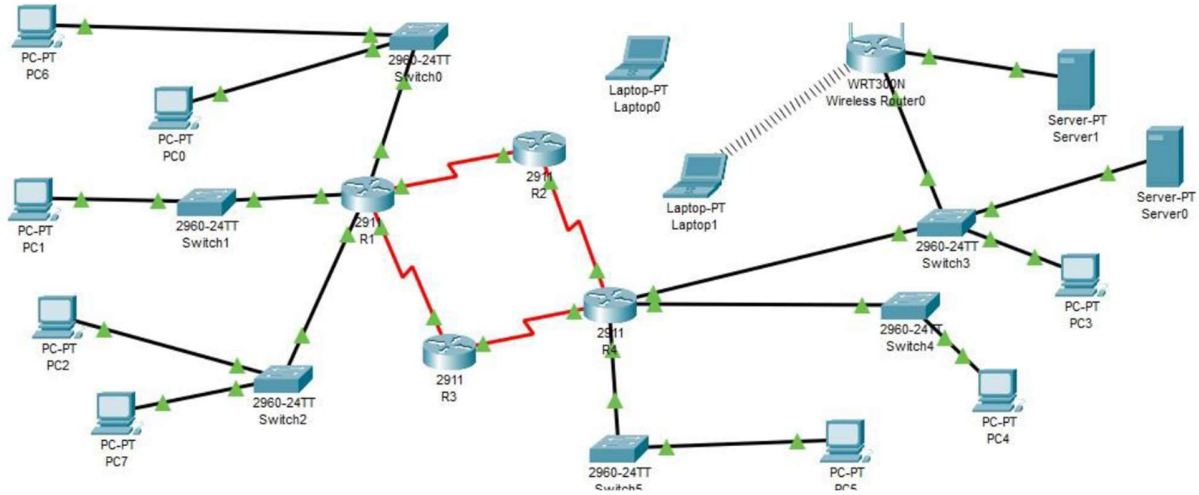


Figure 11.1: Topology of established network

- Now configure DHCP server for gaining IP address.

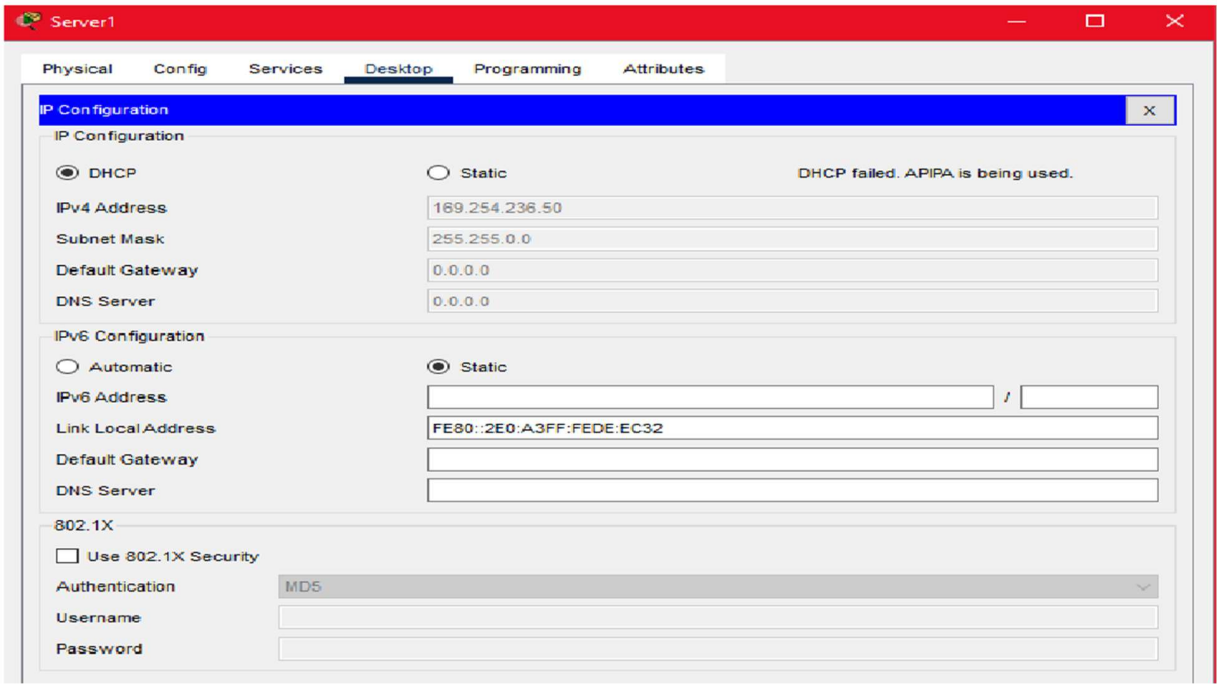


Figure 11.2: Configuring the DHCP server.

- From Laptop 0 configure the router wirelessly

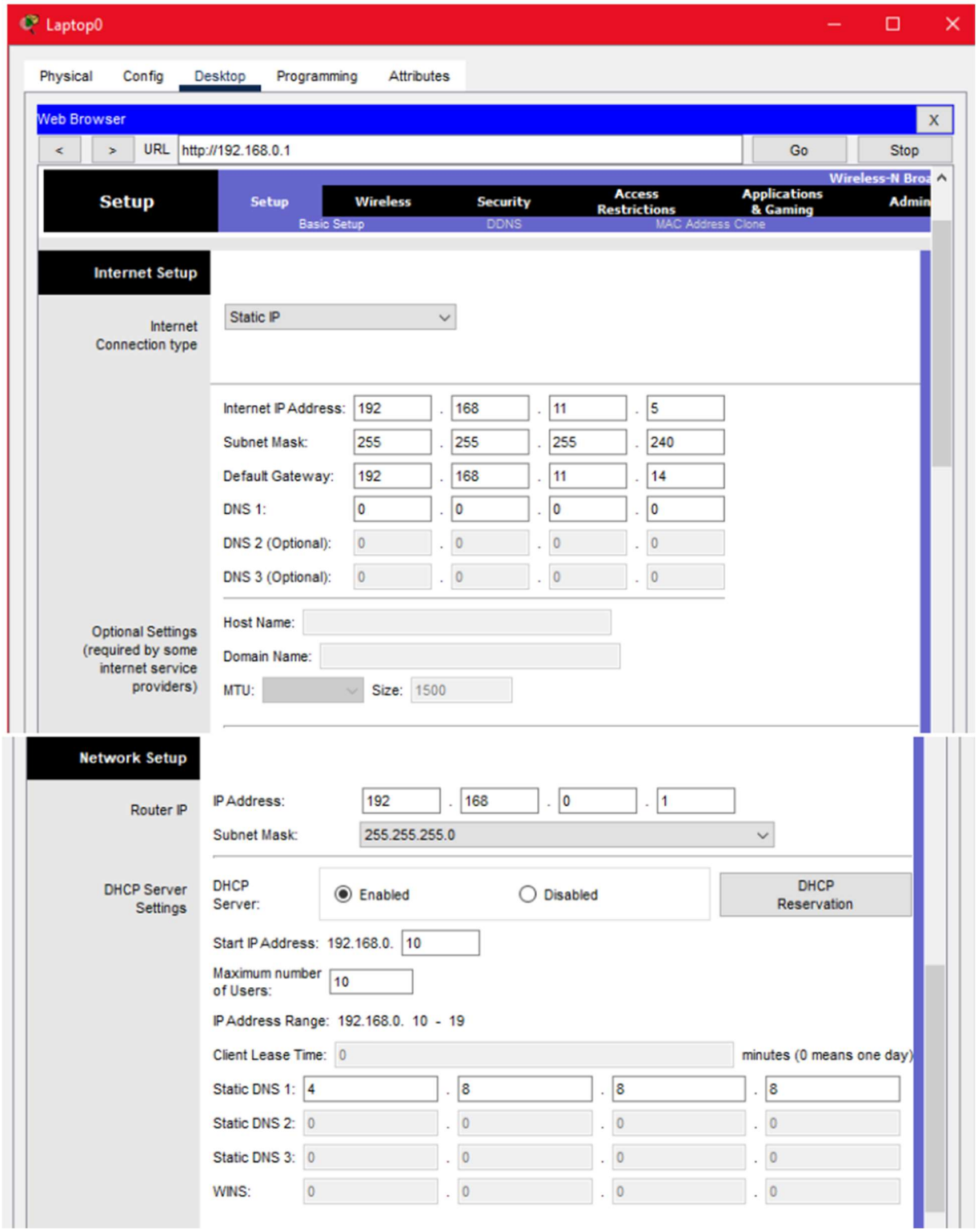


Figure 11.3: Wireless router configuration

- Configure HTTP port

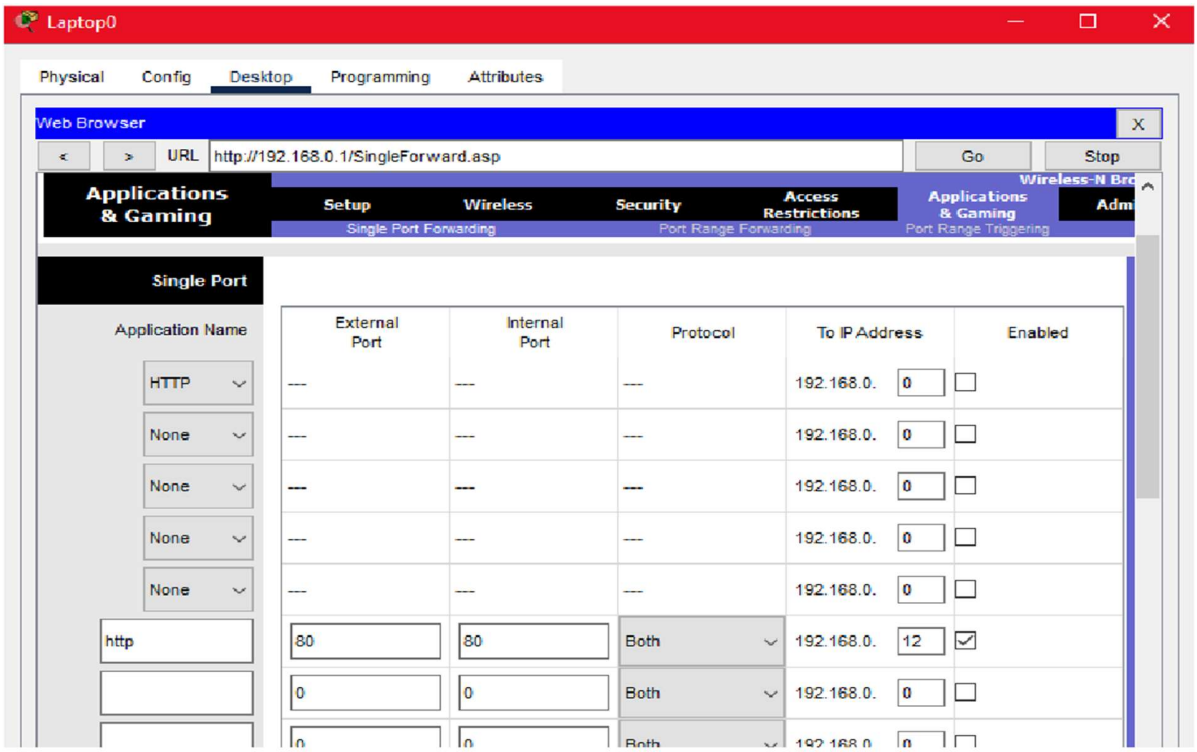


Figure 11.4: HTTP port configuration.

- Configure SSID on the router

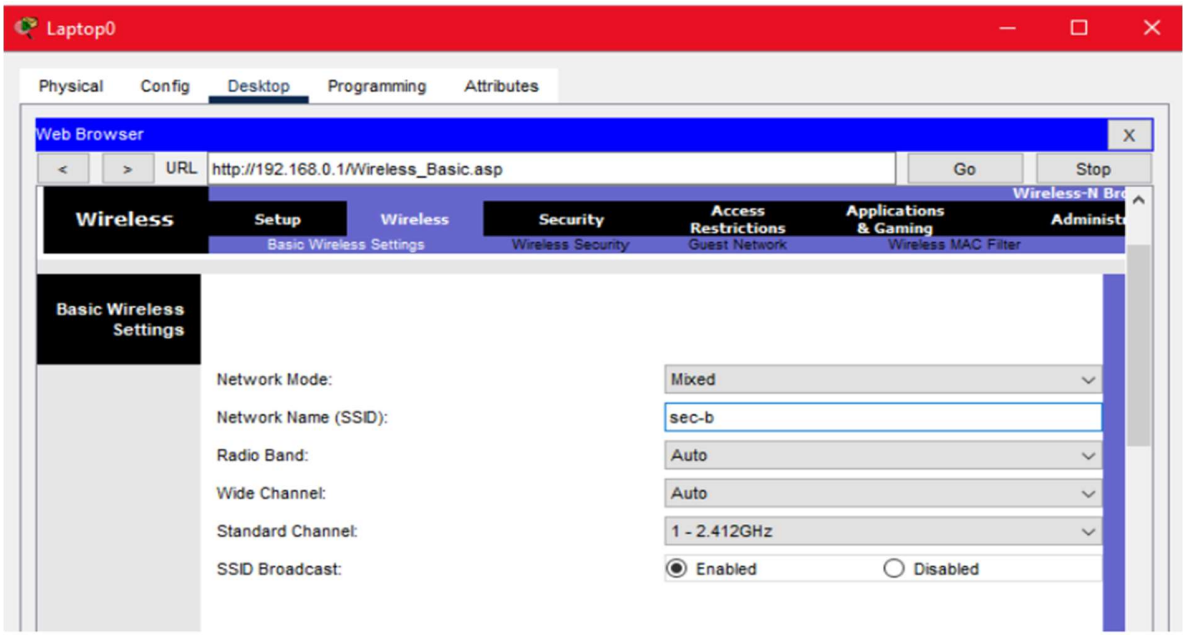


Figure 11.5: SSID configuration.

- For using wireless network the laptop needs an wireless module.

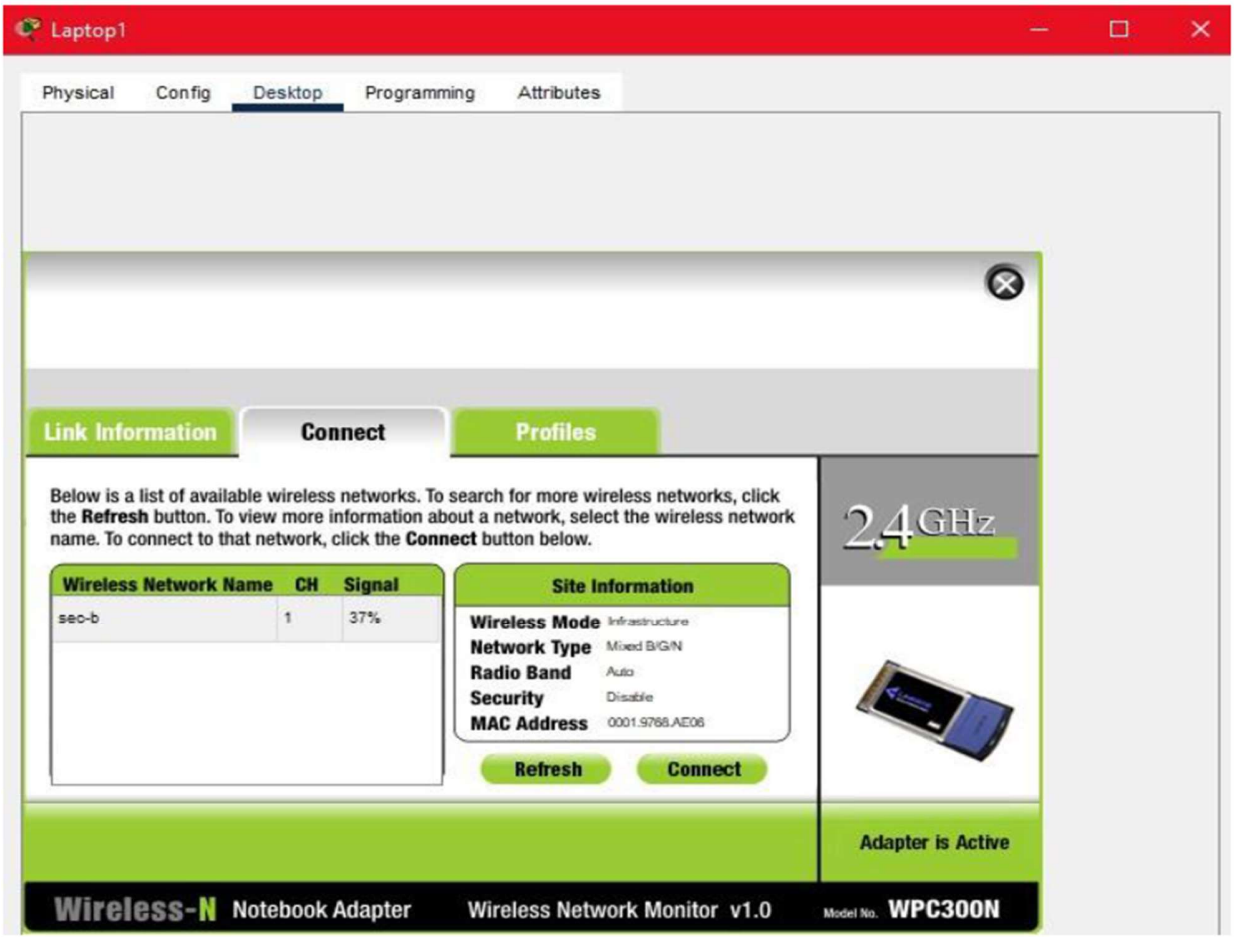


Figure 11.6: Wireless module for laptop to use wireless network

Additional reading materials/ online tutorial/ References

11. IPv6

Title: IPv6.

Objectives

1. To configure IPv6.
2. To check IPv6 connectivity.

Prerequisites

1. Knowledge of how to build network topology, configure the components.
2. Theoretical knowledge of IPv6.

Theory

IPv6

The newest version of the IP protocol, which people often refer to as 'the next- generation Internet Protocol'. It was developed as an answer to many deficiencies of IPv4, most notably the problem of IPv4 address exhaustion. IPv6 is used for the same general functions as IPv4, though with a different implementation.

- There is a very large number of IPv6 addresses, since IPv6 addresses are 128-bit , unlike IPv4 addresses which are 32-bit.
- IPv6 has a simpler header – because there are no checksum bits as used in an IPv4 header. For this reason, routers don't need to calculate the checksum for every packet.
- There is Stateless address autoconfiguration in IPv6: – Hosts autoconfigure themselves with IPv6 addresses.
- There is no need for NAT, since each device in an IPV6 network has a globally unique IPv6 address.

Methodology

1. Set up the topology.
2. Configuring Ipv6
3. Testing connectivity.

Equipment

- 1 Routers (2911)
- Switch (2960-24TT)
- 3 End Devices

Procedure:

Step 1: Create a network topology and an IPv6 addressing table based on the given specifications:

S.NO	Device Name	Link-Local-Address	Default-Gateway
1.	PC	FE80::207:ECFF:FEA3:EB56	FE80::1
2.	Switch(2960-24TT)	FE80::207:ECFF:FEA3:EB56	FE80::1
3.	Router(2911)	FE80::250:FFF:FE6C:B21	FE80::1
4.	Cable	nil	nil

- Connect all component as shown in this figure.

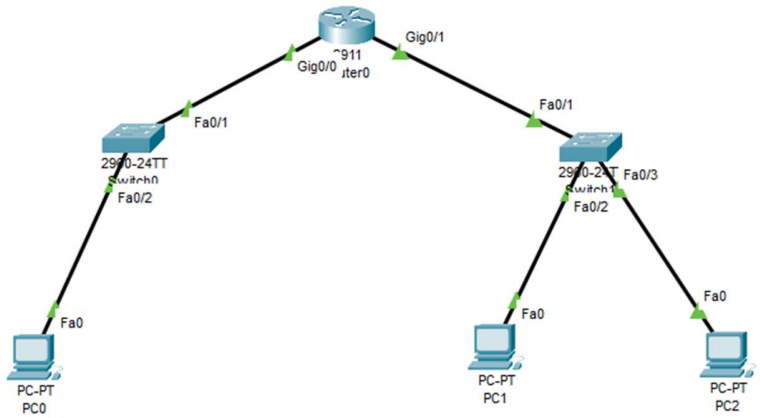


Figure 11.1: Topology of established network.

Step 2: Configuring the Gigabit Ethernet Interfaces.

Configuring the GigabitEthernet0/0 using CLI

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ipv6 unicast-routing
Router(config)#int Gig0/0
Router(config-if)#ipv6 address FE80::1 link
Router(config-if)#ipv6 address FE80::1 link-local
Router(config-if)#no shut
```

Configuring the GigabitEthernet0/1 using CLI

```
Router(config-if)#interface  gigabitEthernet0/1
Router(config-if)#ipv6 address FE80::1 link-local
Router(config-if)#no shutdown
```

Step 3: Configuring Ipv6

CLI commands to configure IPv6 address in GigabitEthernet0/0 and GigabitEthernet0/1 ports:

```
Router#enable
Router#configure terminal
Enter configuration commands, one per line.End with CNTL/Z.
Router(config)#interface gigabitEthernet0/0
Router(config-if)#ipv6 address 2001:DB8:AAAA:A::1/64
Router(config-if)#no shutdown
Router(config-if)#interface gigabitEthernet0/1
Router(config-if)#ipv6 address 2001:DB8:AAAA:B::1/64
Router(config-if)#no shutdown
Router(config-if)#
```

IP Addressing Table:

S.NO	Interface	IPv6 Address
1.	Gig0/0	2001:DB:AAAA:A::1/64
2.	Gig0/1	2001:DB:AAAA:B::1/64

- First, click on PC0 and go to desktop then IP configuration.
- Now find the IPv6 configuration.
- Change the settings from static to automatic and then after a few seconds, the IPv6 address and default gateway are displayed.

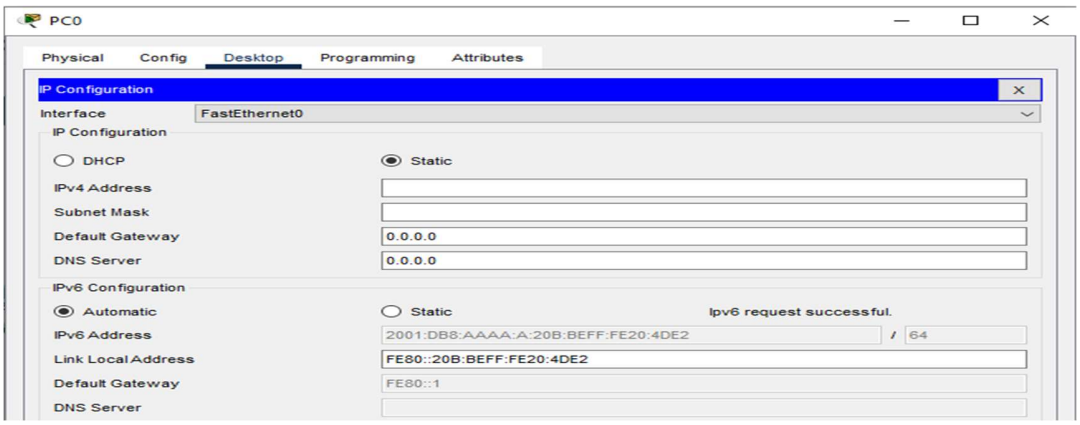


Figure 11.2: IPv6 address of PC0.

- Similarly, repeat this procedure with PC1 and PC2:

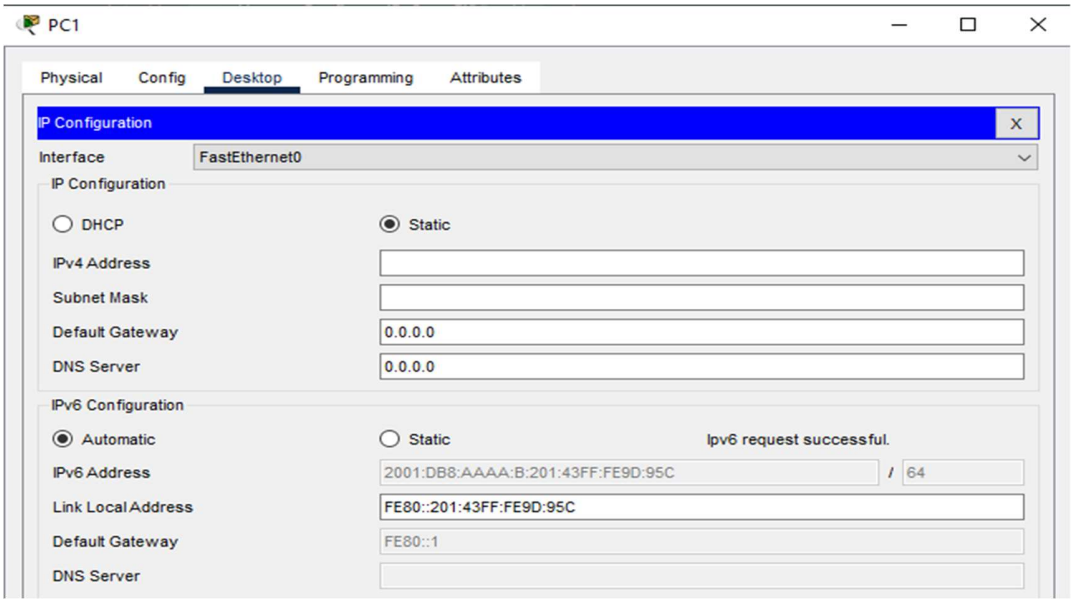


Figure 11.3: IPv6 address of PC1.

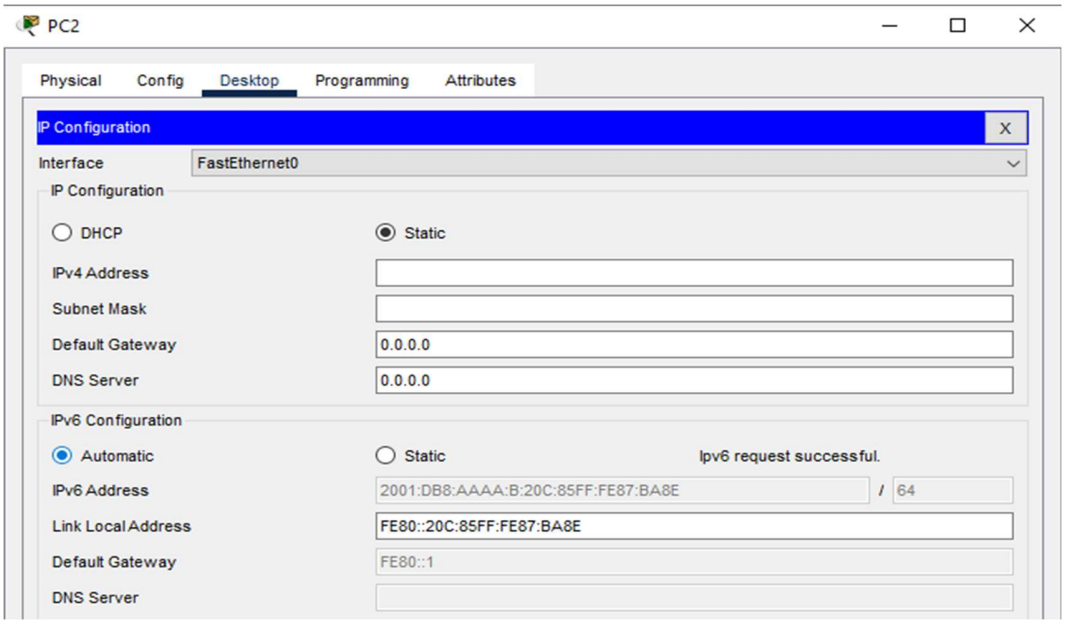


Figure 11.4: IPv6 address of PC2.

Step 5: Check connectivity.

Now we have to verify the connection by pinging the IPv6 address of PC0 in PC1.

- First, click on PC1 and go to the command prompt, and type ping <ipv6 address>

command: ping 2001:DB8:AAAA:A:20D:BDFF:FE1A:D121

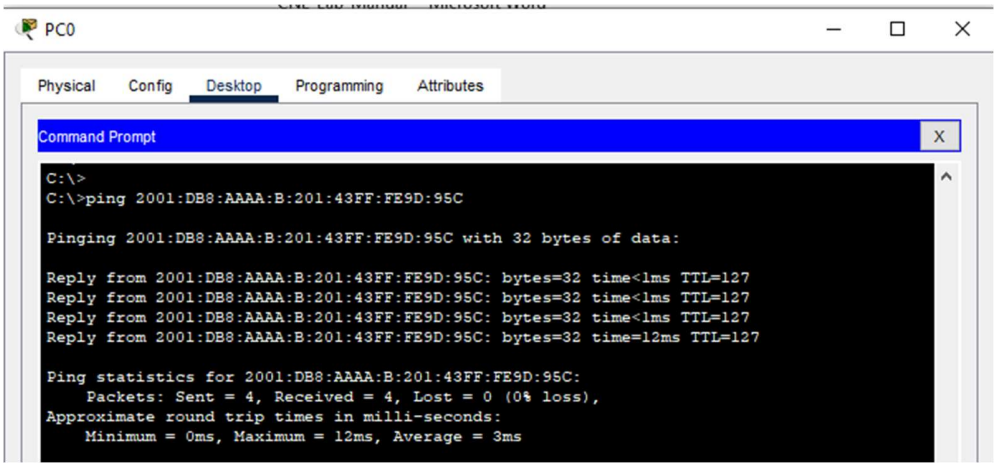


Figure 11.5: Pingin PC1 from PC0.

Now we have to verify the connection by pinging the IPv6 address of PC2 in PC0.

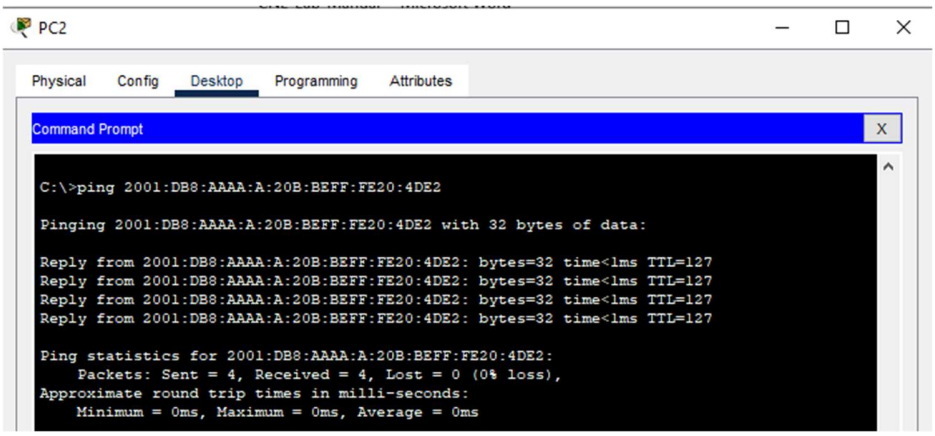


Figure 11.6: Pingin PC0 from PC2.

As we can see in these figure, getting replies from PC0 & PC1 means the connection is established successfully.

Additional reading materials/ online tutorial/ References

1. https://www.geeksforgeeks.org/
2. https://ipcisco.com/
3. https://computernetworking747640215.wordpress.com/

12. UDP and TCP Socket Programming

Title: UDP and TCP Socket Programming.

Objectives

1. To configure TCP Socket Programming.
2. To configure UDP Socket Programming.

Prerequisites

1. Knowledge of Socket programming.
2. Knowledge of Python.

Theory

Sockets are endpoints in bi-directional communications between two processes.
Sockets allow us to connect, send, and receive messages across a network. The network can be logical, local, or external. A socket connects and then uses the read() and write() commands the same way that file-descriptors use files and pipes.
Python has a library for socket programming that provides an interface to the **Berkeley sockets API**. You can import the socket library by writing:

```
import socket
```

Some fundamental python socket API methods are:

- socket()
- bind()
- listen()
- accept()
- connect()
- send()
- recv()
- close()

TCP and UDP sockets

Before we can use any other socket function, we need to create a socket. To create a socket, we use the socket() method. It is up to the user to create either a TCP socket or a UDP socket.

Compared to a UDP socket, a TCP socket is more reliable and provides in-order data delivery. However, UDP sockets deliver messages faster. It is advised that you use a TCP socket to achieve best-effort delivery from the underlying network.

Creating a TCP socket:

```
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

Creating a UDP socket:

```
sock = socket.socket(socket.AF_INET, socket.SOCK_ DGRAM)
```

socket.AF_INET is used to designate the type of addresses our socket can communicate with which, in this case, is ipv4 addresses. socket.SOCK_STREAM specifies a TCP socket and socket.SOCK_DGRAM specifies a UDP socket.

TCP Sockets

we're going to create a socket object using socket.socket(), specifying the socket type as socket.SOCK_STREAM. When we do that, the default protocol that's used is the Transmission Control Protocol (TCP). This is a good default and probably what we want.
The Transmission Control Protocol (TCP):

- **Is reliable:** Packets dropped in the network are detected and retransmitted by the sender.
- **Has in-order data delivery:** Data is read by our application in the order it was written by the sender.

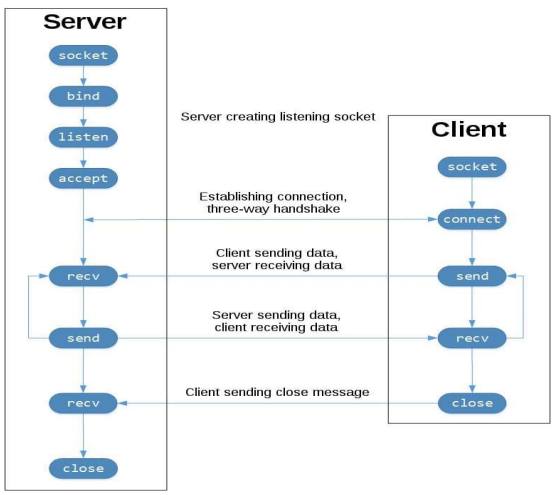
In contrast, User Datagram Protocol (UDP) sockets created with socket.SOCK_DGRAM aren't reliable, and data read by the receiver can be out-of-order from the sender's writes.

The sequence of socket API calls and data flow for TCP:

The left-hand column represents the server. On the right-hand side is the client.
Starting in the top left-hand column, note the API calls that the server makes to set up a "listening" socket:

- socket()
- bind()
- listen()
- accept()

A listening socket does just what its name suggests. It listens for connections from clients. When a client connects, the server calls .accept() to accept,



or complete, the connection.
The client calls .connect() to establish a connection to the server and initiate the three-way handshake. The handshake step is important because it ensures that each side of the connection is reachable in the network, in other words that the client can reach the server and vice-versa. It may be that only one host, client, or server can reach the other.
In the middle is the round-trip section, where data is exchanged between the client and server using calls to .send() and .recv().

TCP Client and Server

Now that you've gotten an overview of how the client and server communicate:

Server.py

```
from socket import *
import socket
import threading

def handle_client(client_socket):
    try:
        message = connectionSocket.recv(1024).decode()
        #if not message:
         #   connectionSocket.close()
          #  continue
        filename = message.split()[1]
        print('Filename requested is ', filename)
        f = open(filename[1:])
        outputdata = f.read()
        # Send one HTTP header line into socket
        sendMessage = 'HTTP/1.1 200 OK \r\n\r\n'
        connectionSocket.send(sendMessage.encode())
        #connectionSocket.sendAll(outputdata.encode())
        # Send the content of the requested file to the client
        for i in range(0, len(outputdata)):
            connectionSocket.send(outputdata[i].encode())
        connectionSocket.send("\r\n".encode())
        print('Success File sent !')
        connectionSocket.close()

    except IOError:
        # Send response message for file not found
        sendMessage = 'HTTP/1.1 404 Not Found\r\n\r\n'
        connectionSocket.send(sendMessage.encode())
        print('Requested File not available.')

        # Close client socket
```

```
        connectionSocket.close()
if __name__ == '__main__':
    serverSocket = socket.socket(AF_INET, SOCK_STREAM)
    #Prepare a sever socket
    serverPort = 65000
    #serverName = ("192.168.0.165")
    serverSocket.bind(('192.168.0.113', serverPort))
    serverSocket.listen()
    while True:
        # Establish the connection
        print('The Server ' + socket.gethostname() + ' is running ...')
        connectionSocket, addr = serverSocket.accept()
        trd = threading.Thread(target=handle_client, args=(connectionSocket, ))
        print('Request from client {} on port {} ...'.format(addr, trd.name))
        trd.start()
   #serverSocket.close()
    #sys.exit()
```

server.py consists of a Server class that takes care of the server-side implementation of the chat app. There are three methods (including the constructor):

- __init__: Here, the server's socket is initialized and, by default, bound to address localhost and port 65000.
- forward_message: This method forwards the message to the recipient that server receives from the sender.
- start: This method is called upon at the start of the code. This method saves the port and address of the client if it receives a join signal, and forwards the message to the recipient if it receives a send_message signal from a the sender.

Server.py

```
import socket
import sys
def main():

    if len(sys.argv) != 4:
        print("Usage: client.py server_host server_port filename")
        return

    server_host = sys.argv[1]
    server_port = int(sys.argv[2])
    filename = sys.argv[3]

    request = f"GET /{filename} HTTP/1.1\r\nHost: {server_host}\r\n\r\n"
```

```
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect((server_host, server_port))
    client_socket.send(request.encode('utf-8'))

    response = client_socket.recv(1024)
    print(response.decode('utf-8'))

    client_socket.close()

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

client.py consists of a Client class that takes care of the client-side implementation of the chat app. There are five methods (including the constructor):

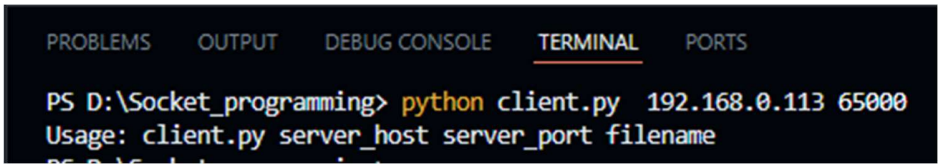
- __init__: Here, the server's port and address are initialized. Moreover, the client's socket is initialized and bound to a random port.
- join: This method sends the join-message to the server when the client code is run for the first time.
- send_message: This method sends the message entered by the client to the server – the server then forwards it to the recipient.
- start: This method is called at the start of the code. It calls either the join method or the send_message method depending on the input signal.
- receive_handler: This method receives the message from the server and prints it on the terminal. It is called using a thread object in order to run it parallel to the start method.

Running the TCP Client and Servers

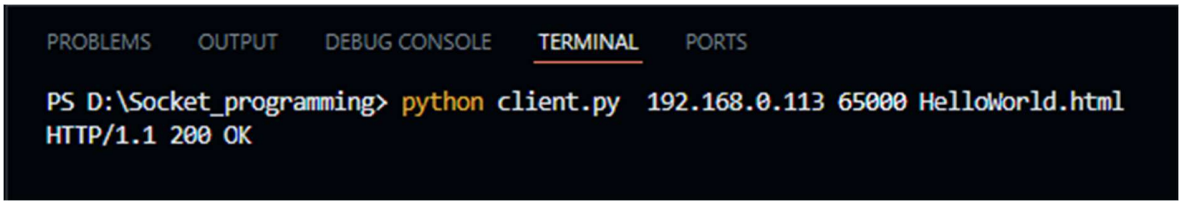
Open a terminal or command prompt, navigate to the directory that contains our scripts, ensure that you have Python, then run the server:

It’s waiting for a client connection. Now, open another terminal window or command prompt and run the client:

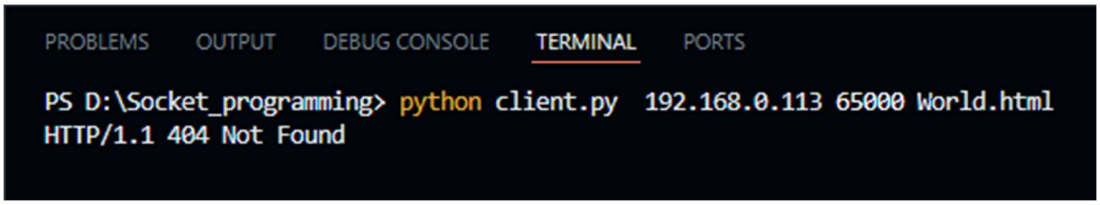
- If any argument missing in the command, we should see something like:



- If everything is correct, we should see something like:



- If something is incorrect, we should see something like:



Notice that the server continues running and will establish a new connection every time you run the client and append any new output.

UDP Sockets

User Datagram Protocol (UDP) sockets created with socket.SOCK_DGRAM aren’t reliable, and data read by the receiver can be out-of-order from the sender’s writes.

UDP Client and Server

Now that you’ve gotten an overview of how the client and server communicate:

Server.py

```
from socket import *

class Server:
    def __init__(self, port=65000):
        # Initialize server socket and bind to the port
        self.port = port
        self.socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.socket.bind(("", self.port))
```

```
        print(f"Server listening on port {self.port}...")

    def listen(self):
        while True:
            # Receive data from the client
            data, addr = self.socket.recvfrom(1024)
            print(f"Received message from {addr}: {data.decode('utf-8')}")

if __name__ == "__main__":
    server = Server()
    server.listen()
```

server.py consists of a Server class that takes care of the server-side implementation of the chat app. There are two methods (including the constructor):

- __init__: Initializes the server's UDP socket and binds it to the specified port (defaults to 65000). The server is now ready to listen for incoming messages.
- listen: This method runs an infinite loop, where the server listens for incoming messages using recvfrom(). When a message is received, it prints the message and the address of the client that sent it.

Server.py

```
import socket

class Client:
    def __init__(self, host="192.168.0.113", port=65000):
        # Initialize client socket
        self.host = host
        self.port = port
        self.socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    def send_message(self, message):
        # Encode and send the message to the server
        self.socket.sendto(message.encode('utf-8'), (self.host, self.port))

if __name__ == "__main__":
    client = Client()
    client.send_message('Prova protocollo UDP')
```

client.py consists of a Client class that takes care of the client-side implementation of the chat app. There are five methods (including the constructor):

- __init__: This method initializes the client socket and sets the host and port to which the client will send messages. By default, it sends to localhost on port 65000.

- send_message: This method takes a message, encodes it to UTF-8, and sends it to the server using sendto().

Running the UDP Client and Servers

Open a terminal or command prompt, navigate to the directory that contains our scripts, ensure that you have Python, then run the server:

It's waiting for a client connection. Now, open another terminal window or command prompt and run the client:

Main Differences in UDP:

- **Connectionless protocol**: Unlike TCP, we don't have to establish a connection. Instead, you simply send data using sendto() and receive data using recvfrom().
- **No accept() in the server**: The server waits for incoming datagrams (packets) and processes each one without establishing a persistent connection.
- **No connect() in the client**: The client just sends a request to the server and waits for a response.

Additional reading materials/ online tutorial/ References

1. https://realpython.com/python-sockets/
2. https://www.educative.io/
3. https://python-forum.io/
4. https://medium.com/