

Computing Networks

Laboratory No. 6

Application Layer Protocols, Base

Platform, and Network Layer

Students:

Andrea Camila Torres González

Jorge Andrés Gamboa Sierra

Presented to:

Fabian Eduardo Sierra Sánchez

Semester 2024-2

Contend

Objective.....	6
Tools to be used.....	6
Abstract.....	6
Theoretical framework.....	6
Introduction.....	8
Simulations	8
1. Router Simulation	8
Base Software Installation.....	37
1. Web Service Installation.....	37
1.1. Solaris	37
1.2. Linux Slackware	41
1.3. Windows Server	54
2. Hosting Service Configuration	69
Conclusions.....	76
References	76
Figure 1 Initial configuration.....	9
Figure 2 Router0 configuration	9
Figure 3 Router1 configuration	9
Figure 4 connection between routers.....	10
Figure 5 connection to port RS 232.....	10
Figure 6 connection to console port	11
Figure 7 Console cable connection.....	11
Figure 8 configuration of Router	11
Figure 9 Terminal configuration.....	12
Figure 10 login configuration.....	12
Figure 11 interface and Ip configuration	13
Figure 12 configure interface Router0.....	13
Figure 13 enable password secret.....	13
Figure 14 saving changes	13
Figure 15 connection console port	15
Figure 16 connection RS 232 port.....	15
Figure 17 configuration router1.....	15
Figure 18 initial configuration router1	16
Figure 19 configuration connection to PC2.....	16
Figure 20 interfaces configuration.....	16
Figure 21 interface to router0.....	16
Figure 22 saving changes router1	17

UNIVERSIDAD

Figure 23 PC0 Settings	18
Figure 24 PC1 Settings	19
Figure 25 PC2 Settings	19
Figure 26 PC3 Settings	20
Figure 27 general view of the network	20
Figure 28 successful ping	21
Figure 29 successful ping	21
Figure 30 failed ping	21
Figure 31 failed ping	22
Figure 32 successful ping	22
Figure 33 successful ping	22
Figure 34 failed ping	23
Figure 35 successful ping	23
Figure 36 successful ping	23
Figure 37 failed ping	24
Figure 38 failed ping	24
Figure 39 successful ping	24
Figure 40 successful ping	25
Figure 41 failed ping	25
Figure 42 failed ping	25
Figure 43 successful ping	26
Figure 44 route configuration	26
Figure 45 static route configuration	26
Figure 46 successful ping	27
Figure 47 trace view	27
Figure 48 successful ping	27
Figure 49 trace view	27
Figure 50 general network view	28
Figure 51 peer module	28
Figure 52 multi-user configuration	29
Figure 53 multi-user configuration	30
Figure 54 Incoming connection	31
Figure 55 connection with peer module	31
Figure 56 multiuser outgoing	32
Figure 57 connection to another network	32
Figure 58 configuration interface	33
Figure 59 configuration interface	33
Figure 60 general view	33
Figure 61 static route configuration	33
Figure 62 static route configuration	34
Figure 63 tracer successful	34
Figure 64 tracer successful	34
Figure 65 tracer successful	34
Figure 66 tracer successful	35
Figure 67 configuration interface Router1 RED_Camila	35
Figure 68 configuration static route RED_Camila Router0	36
Figure 69. Initializing Apache	37

UNIVERSIDAD

Figure 70. Moving into htdocs directory	37
Figure 71. Editing the main file on Apache web server	38
Figure 72. Verifying that Solaris web server is working.....	39
Figure 73. Configuring zone file gamboa.com.it to resolve the domain name of the web server (www.gamboa.com.it)	40
Figure 74. Accessing the Solaris web server by domain name	41
Figure 75. Downloading nginx.tar.gz file from SlackBuilds repository	41
Figure 76. Decompressing nginx.tar.gz.....	42
Figure 77. Downloading nginx version	42
Figure 78. Decompressing nginx version	43
Figure 79. Configuring nginx	44
Figure 80. Installing PCRE.....	45
Figure 81. Installing libxslt.....	46
Figure 82. Installing gd.....	47
Figure 83. Configuring nginx with the necessary libraries.....	48
Figure 84. Compiling nginx package	49
Figure 85. Installing nginx version 1.26.2.....	50
Figure 86. Running web service on Nginx	50
Figure 87. Customizing Slackware website with nginx	51
Figure 88. Configuring web service when the slackware boots	52
Figure 89. Verifying that Slackware web server is working	52
Figure 90. Configuring zone file torres.org.uk.hosts to resolve the domain name of the web server (www.torres.org.uk)	53
Figure 91. Accessing the Slackware web server by domain name.....	53
Figure 92. Opening server configuration.....	54
Figure 93. Selecting operating system.....	55
Figure 94. Selecting web server for Windows Server	56
Figure 95. Adding Features	57
Figure 96. Selecting Role Services.....	58
Figure 97. Starting the web server installation on Windows.....	59
Figure 98. Installing web service in Windows	60
Figure 99. Opening wwwroot directory to store websites	61
Figure 100. Creating index.html into wwwroot	62
Figure 101. Customizing index.html file in windows web server.....	63
Figure 102. Saving index.html file into Windows Web Service	64
Figure 103. Opening Windows services configuration	65
Figure 104. Searching web service in Windows services configuration	66
Figure 105. Configuring web server to start when Windows boots	67
Figure 106. Verifying that Windows web server is working.....	68
Figure 107. Configuring gamboa.com.it to access the Windows web server.....	68
Figure 108. Configuring torres.org.uk to access the Windows web server.....	69
Figure 109. Accessing the Windows web server by domain name	69
Figure 110. Opening httpd.conf file	69
Figure 111. Configuring Virtual Host service	70
Figure 112. Creating Virtual Host directories	70
Figure 113. Customizing network.camila.com.co page	71
Figure 114. Customizing security.jorge.org.jp page.....	71

UNIVERSIDAD

Figure 115. Adding new zones in /etc/named.conf zones file	72
Figure 116. Configuring zone file Jorge.org.jp	73
Figure 117. Configuring zone file camila.com.co	73
Figure 118. Verifying the domain jorge.org.jp is responding	74
Figure 119. Verifying the domain camila.com.co is responding	74
Figure 120. Configuring DNS server on the client machine	75
Figure 121. Verifying that the Virtual Host of network.camila.com.co is working	75
Figure 122. Verifying that the Virtual Host of security.jorge.org.jp is working	76

Objective

Install and configure basic software for web servers.

Tools to be used

Computers
Internet Access

Abstract

This lab focuses on installing and configuring web servers within an IT infrastructure. Students will simulate router configurations using Packet Tracer, establishing static routes and testing connectivity between networks. They will also install and configure web servers, including Apache on Solaris and Nginx on Slackware, while ensuring proper functionality through basic web pages. Additionally, DNS services will be configured for domain name resolution. The lab culminates in configuring virtual hosts on Solaris to support multiple domain webpages, demonstrating essential skills in web service management.

Theoretical framework

1. Network Infrastructure

Network infrastructure is the collection of interconnected components that enable communication and data exchange within an organization. It includes:

- **Workstations:** Devices that allow users to interact with the network, run applications, and access services. They can be physical or virtual computers.
- **Servers:** Machines that provide resources and services to workstations. These can include web servers, email servers, database servers, and more, available in both physical and virtual forms.
- **Network Devices:**
 - **Layer 2 Switches:** Connect multiple devices within a local area network (LAN) and direct data traffic based on MAC addresses.
 - **Layer 3 Routers:** Route traffic between different networks and management functions such as NAT (Network Address Translation) and ACL (Access Control Lists).
 - **NAT (Network Address Translation):** A technique used to modify the IP address in the packet headers while traversing a router or firewall. NAT allows multiple devices on a local network to use a single public IP address to access the Internet, helping to conserve IP addresses and providing an additional layer of security by hiding internal network addresses.
 - **ACL (Access Control List):** A list that defines who can access certain resources within a network, specifying access permissions for users or groups. In routers and switches, ACLs are used to filter network traffic, allowing or denying the passage of packets based on IP addresses, protocols, or ports, which enhances security and traffic management.

2. Cisco Packet Tracer Components

Cisco Packet Tracer is a network simulation tool that allows users to create and manage virtual networks. Some of its components include:

- **Multiuser Peer Module:** Enables real-time collaboration between multiple users in the simulation, facilitating group work.
- **PC-PT:** Simulates personal computers, allowing network configurations, connectivity tests (such as ping), and application execution.

UNIVERSIDAD

- **Router 1941:** A router model that enables routing configurations, NAT, and other advanced functions, helping to understand data flow within a network.

3. Application Layer Protocols

Application layer protocols are essential for communication between different services on a network. They define the rules and conventions for data exchange at the highest level of the OSI model, ensuring that applications can communicate effectively. The most relevant protocols for this lab include:

- **HTTP (Hypertext Transfer Protocol)** is the foundation of data communication for the World Wide Web, used for transmitting web pages from servers to clients. Operating over TCP, typically on port 80, it follows a request-response model where the client sends an HTTP request, and the server responds with the requested resource. HTTP is stateless, meaning each request is independent of previous ones, which simplifies the protocol but can lead to inefficiencies that are addressed in more advanced versions like HTTP/2. While HTTP itself lacks built-in security features, it can be secured using HTTPS.
- **HTTPS (HTTP Secure):** is the secure version of HTTP, providing encrypted communication and secure identification of a network web server. Utilizing SSL (Secure Sockets Layer) or TLS (Transport Layer Security) protocols, HTTPS operates over TCP on port 443. The encryption ensures data privacy and integrity between clients and servers, protecting against eavesdropping and man-in-the-middle attacks. The security of HTTPS relies on digital certificates issued by trusted Certificate Authorities (CAs), which verify the authenticity of the websites and establish a secure connection.
- **DNS (Domain Name System)** Translates human-readable domain names into IP addresses, allowing users to access resources on the Internet without needing to remember numerical IP addresses. It operates over UDP, usually on port 53, providing a hierarchical and distributed database system for name resolution. DNS queries and responses involve multiple servers, including recursive resolvers, root name servers, and authoritative name servers, to efficiently map domain names to their corresponding IP addresses. This process is crucial for the functionality of the Internet, enabling seamless access to websites and online services.

4. Installation and Configuration of Web Servers

In this lab, web servers will be installed and configured, including:

- **Apache:** An open-source web server widely used for its versatility. It supports multiple domains through virtual host configuration.
- **Nginx:** A web server known for its high performance and ability to handle multiple simultaneous connections, also used as a reverse proxy and load balancer.
- **Windows Server:** A Microsoft operating system that provides web server services through IIS (Internet Information Services), allowing management of web applications in a Windows environment.

5. Hosting Service Configuration

Hosting service configuration allows multiple websites to be hosted on a single server. This is typically achieved using virtual hosts, a feature provided by web servers like Apache.

Virtual Hosts in Apache

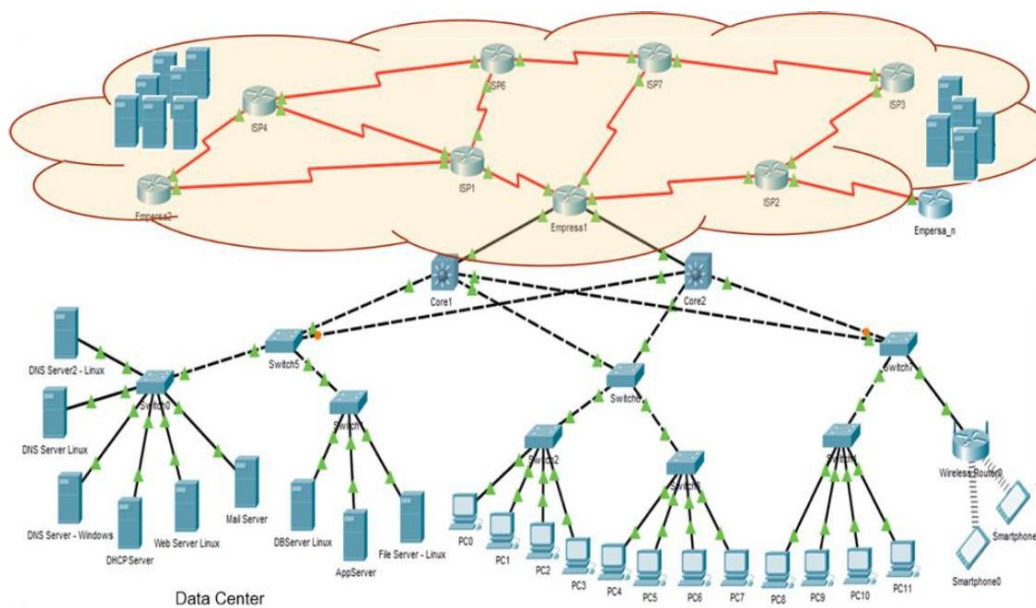
Virtual Hosts in Apache enable a single server to host multiple websites or domains, allowing efficient use of resources. This is accomplished by defining different configurations for each website within the server's settings. There are two main types of virtual hosts:

1. **Name-Based Virtual Hosts:** This type allows multiple websites to share a single IP address, distinguishing between them based on the domain name included in the request. This is the most common approach used by hosting providers.
2. **IP-Based Virtual Hosts:** In this scenario, each website has its own unique IP address. This method is less common due to IP address limitations but may be necessary for specific configurations, particularly in scenarios requiring separate SSL certificates for each domain.

By effectively utilizing virtual hosts, Apache can serve multiple websites seamlessly, making it an essential

Introduction

We are continuing to work on the infrastructure of a company that typically hosts several IT infrastructure services. This infrastructure includes both wired and wireless user workstations, as well as physical and virtualized servers, all interconnected through Layer 2 and Layer 3 switches, wireless devices, and routers that provide Internet access. Additionally, it is common to have cloud infrastructures from which resources are provisioned based on the organization's needs. Among the servers, you can find web services, DNS, email, databases, storage, and applications, among others. Let's recall the base configuration we are using:



In this lab, we will focus on server infrastructure.

Simulations

Conduct the following tests individually or in groups, as instructed by the professor, and document your experience.

1. Router Simulation

First, we place the devices indicated in the diagram, along with their respective PCs and routers, along with their connections in the network configurations of each one.

UNIVERSIDAD

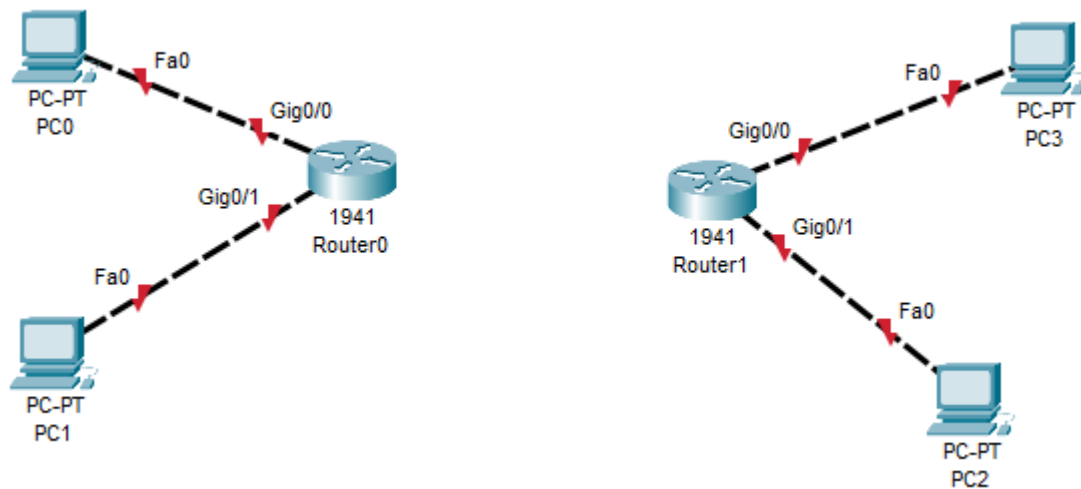


Figure 1 Initial configuration

For the serial connection between the two routers, we open the configuration of each one, turn it off, and connect the HWIC-2T module to be able to connect the serial cable later.

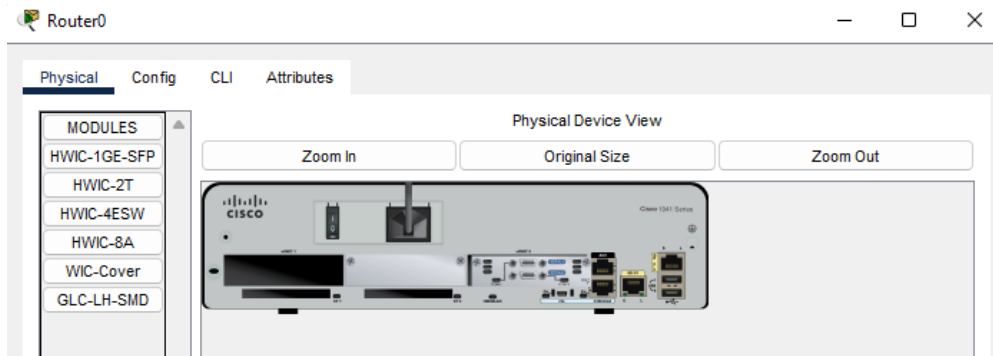


Figure 2 Router0 configuration

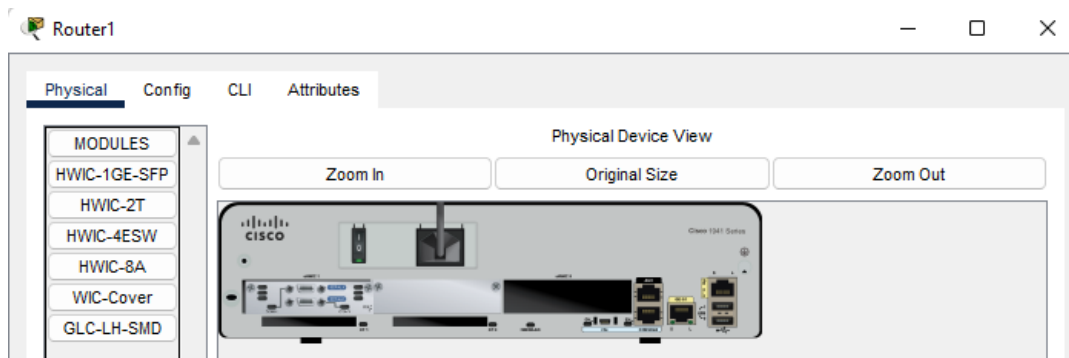


Figure 3 Router1 configuration

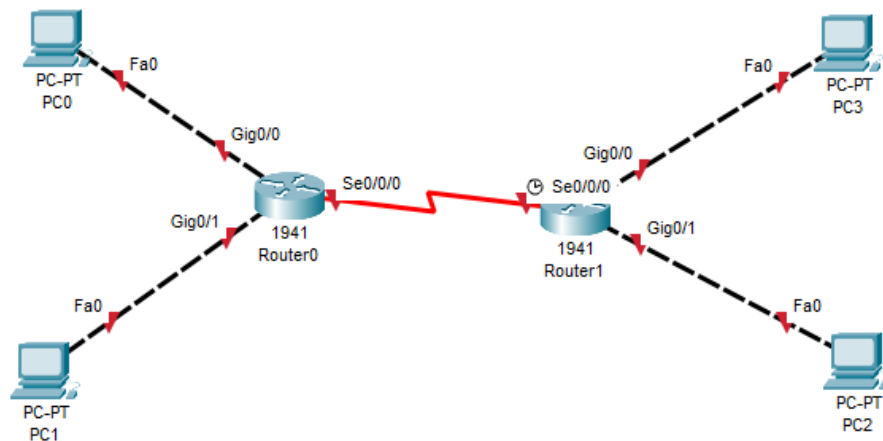


Figure 4 connection between routers

Now, we configure the IPs of each device respectively, according to each student's configuration. In this case, the networks will be: Student 01: 132.18.0.0/16, 132.19.0.0/16, 132.20.0.0/16, 132.21.0.0/16.

First, we will start with the configuration of routers 0 and 1.

we connect a console cable to a computer and to the router we want to configure. The computer must be connected through the RS-232 port and the router through the console port to configure it.

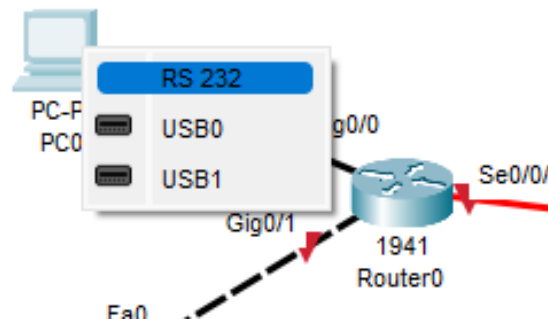


Figure 5 connection to port RS 232

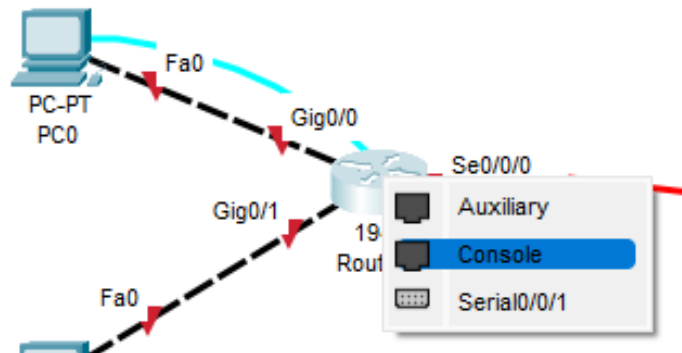


Figure 6 connection to console port

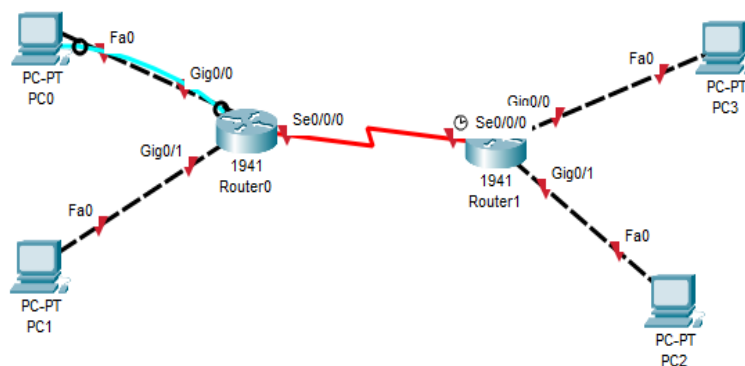


Figure 7 Console cable connection.

The terminal is open on the computer to which we connect the console cable.

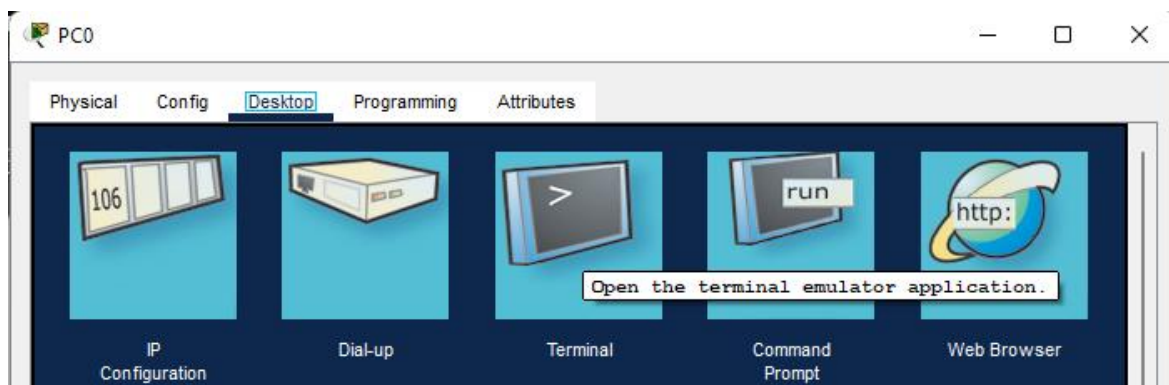


Figure 8 configuration of Router

we select OK and continue

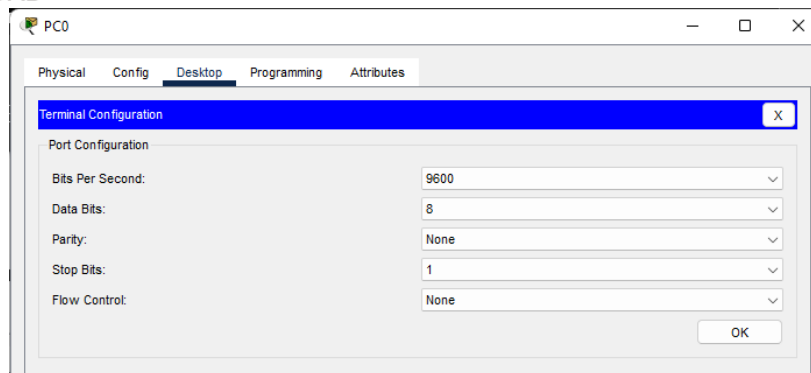


Figure 9 Terminal configuration

Now we must execute the following commands in the terminal

```
Would you like to enter the initial configuration dialog? [yes/no]: n

Press RETURN to get started!

Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#hostname Jorge
Jorge(config)#banner motd "Exclusive use for RECO students"
Jorge(config)#line console 0
Jorge(config-line)#logging synchronous
Jorge(config-line)#password Clave_C
Jorge(config-line)#login
Jorge(config-line)#exit
Jorge(config)#line vty 0 15
Jorge(config-line)#logging synchronous
Jorge(config-line)#password Clave_T
Jorge(config-line)#login
Jorge(config-line)#exit
Jorge(config)#no ip domain-lookup
Jorge(config)#interface GigabitEthernet 0/0
```

Figure 10 login configuration

UNIVERSIDAD

```
Jorge(config-if)#description "connection Router0 with PC0"
Jorge(config-if)#ip address 132.18.0.1 255.255.0.0
Jorge(config-if)#no shutdown

Jorge(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state to up

Jorge(config-if)#interface GigabitEthernet 0/1
Jorge(config-if)#description "connection router0 to PC1"
Jorge(config-if)#ip address 132.19.0.1 255.255.0.0
Jorge(config-if)#no shutdown

Jorge(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to up

Jorge(config-if)#exit
```

Figure 11 interface and Ip configuration

```
Jorge(config)#interface Serial0/0/0
Jorge(config-if)#ip address 20.0.0.1 255.0.0.0
Jorge(config-if)#description "configure interface router0"
Jorge(config-if)#no shutdown
```

Figure 12 configure interface Router0

```
Jorge(config-if)#exit
Jorge(config)#enable secret Clave_E
Jorge(config)#exit
Jorge#
%SYS-5-CONFIG_I: Configured from console by console

Jorge#
```

Figure 13 enable password secret

```
Jorge# copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
```

Figure 14 saving changes

- **enable:** Switches to privileged mode in the device's CLI.
- **configure terminal:** Enters global configuration mode to make changes.
- **hostname Jorge:** Assigns the name "Jorge" to the device for easier identification.
- **banner motd "Exclusive use for RECO students":** Configures a welcome message displayed at login.
- **copy running-config startup-config:** saves the current configuration to startup memory.

Console Configuration:

- **line console 0:** Configures the console line.
- **logging synchronous:** Prevents system messages from interrupting command input.
- **password Clave_C:** Sets the password for the console.
- **login:** Requires authentication to access the console.

UNIVERSIDAD

- **exit:** Exits console line configuration mode.

Remote Access Configuration:

- **line vty 0 15:** Configures the virtual terminal lines (vty).
- **password Clave_T:** Sets the password for remote access.
- **login:** Requires authentication on the vty lines.
- **exit:** Exits the vty line configuration mode.
- **no ip domain-lookup:** Disables DNS lookup to avoid delays due to command errors.

Network Interface Configuration:

- **interface GigabitEthernet 0/0:** Enters configuration for interface 0/0.
- **Description "connection Router0 with PC0":** Assigns a description to the interface.
- **ip address 132.18.0.1 255.255.0.0:** Configures the IP address and subnet mask.
- **no shutdown:** Activates the interface.
- **exit:** Exits the interface configuration mode.
- **interface GigabitEthernet 0/1:** Repeats the process for interface 0/1.
- **description "connection to router0 with PC2":** Assigns a description to the interface.
- **ip address 132.19.0.1 255.255.0.0:** Configures the IP address.
- **no shutdown:** Activates the interface.
- **exit:** Exits the interface configuration mode.
- **interface Serial0/0/0:** Repeats the process for interface 0/0/0.
- **description "configure interface router0":** Assigns a description to the interface.
- **ip address 20.0.0.1 255.0.0.0:** Configures the IP address.
- **no shutdown:** Activates the interface.
- **exit:** Exits the interface configuration mode.
- **enable secret Clave_E:** Sets a secret password for privileged access.
- **exit:** Ends the global configuration session.

The configuration for router 1 will be the same as that of router 0, with the only difference being the network configurations, which will be different. The rest of the procedure and commands will be the same.

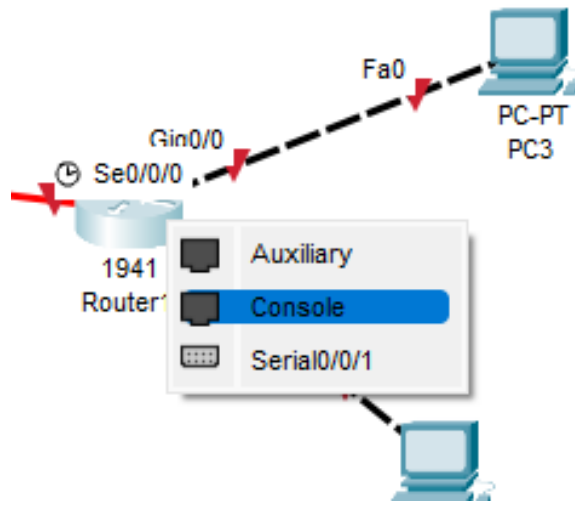


Figure 15 connection console port

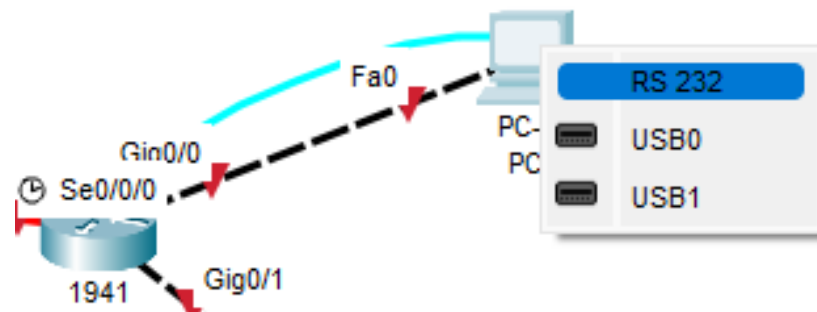


Figure 16 connection RS 232 port

We enter the terminal of the computer connected to router 1, PC3, and then we enter the following commands:

```

--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: n

Press RETURN to get started!

Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#hostname Gamboa
  
```

Figure 17 configuration router1

UNIVERSIDAD

```
Gamboa(config)#banner motd "Exclusive user for RECO sstudents"|
Gamboa(config)#line console 0
Gamboa(config-line)#logging synchronous
Gamboa(config-line)#password Clave_C
Gamboa(config-line)#login
Gamboa(config-line)#exit
Gamboa(config)#line vty 0 15
Gamboa(config-line)#logging synchronous
Gamboa(config-line)#password Clave_T
Gamboa(config-line)#login
Gamboa(config-line)#exit
Gamboa(config)#no ip domain-lookup
```

Figure 18 initial configuration router1

```
Gamboa(config)#interface GigabitEthernet 0/0
Gamboa(config-if)#description "connection to PC2"
```

Figure 19 configuration connection to PC2

```
Gamboa(config-if)#ip address 132.20.0.1 255.255.0.0
Gamboa(config-if)#no shutdown

Gamboa(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state
to up

Gamboa(config-if)#exit
Gamboa(config)#interface GigabitEthernet 0/1
Gamboa(config-if)#description "connection to Router1 with PC3"
Gamboa(config-if)#ip address 132.21.0.1 255.255.0.0
Gamboa(config-if)#no shutdown

Gamboa(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state
to up
```

Figure 20 interfaces configuration

```
Gamboa(config-if)#exit
Gamboa(config)#interface Serial0/0/0
Gamboa(config-if)#ip address 20.0.0.2 255.0.0.0
Gamboa(config-if)#description "configure interface router1"
Gamboa(config-if)#no shutdown
```

Figure 21 interface to router0

UNIVERSIDAD

```
Gamboa(config-if)#exit
Gamboa(config)#enable secret Clave_E
Gamboa(config)#exit
Gamboa#
%SYS-5-CONFIG_I: Configured from console by console

Gamboa#copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
Gamboa#
```

Figure 22 saving changes router1

- **enable:** Switches to privileged mode in the device's CLI.
- **configure terminal:** Enters global configuration mode to make changes.
- **Hostname Gamboa:** Assigns the name "Gamboa" to the device for easier identification.
- **banner motd "Exclusive use for RECO students":** Configures a welcome message displayed at login.
- **copy running-config startup-config:** saves the current configuration to startup memory.

Console Configuration:

- **line console 0:** Configures the console line.
- **logging synchronous:** Prevents system messages from interrupting command input.
- **password Clave_C:** Sets the password for the console.
- **login:** Requires authentication to access the console.
- **exit:** Exits console line configuration mode.

Remote Access Configuration:

- **line vty 0 15:** Configures the virtual terminal lines (vty).
- **password Clave_T:** Sets the password for remote access.
- **login:** Requires authentication on the vty lines.
- **exit:** Exits the vty line configuration mode.
- **no ip domain-lookup:** Disables DNS lookup to avoid delays due to command errors.

Network Interface Configuration:

- **interface GigabitEthernet 0/0:** Enters configuration for interface 0/0.
- **Description " connection to router1 with PC2":** Assigns a description to the interface.
- **ip address 132.20.0.1 255.255.0.0:** Configures the IP address and subnet mask.
- **no shutdown:** Activates the interface.
- **exit:** Exits the interface configuration mode.

UNIVERSIDAD

- **interface GigabitEthernet 0/1:** Repeats the process for interface 0/1.
- **description "connection to router1 with PC3":** Assigns a description to the interface.
- **ip address 132.21.0.1 255.255.0.0:** Configures the IP address.
- **no shutdown:** Activates the interface.
- **exit:** Exits the interface configuration mode.
- **interface Serial0/0/1:** Repeats the process for interface 0/0/0.
- **description "configure interface router1":** Assigns a description to the interface.
- **ip address 20.0.0.2 255.0.0.0:** Configures the IP address.
- **no shutdown:** Activates the interface.
- **exit:** Exits the interface configuration mode.
- **enable secret Clase_E:** Sets a secret password for privileged access.
- **exit:** Ends the global configuration session.

Now we enter the respective IP configurations on each PC. We assign each computer its respective part of the network and set it to end with .2 since .0 is the network identifier and .1 is the gateway configured on the router.

- PC0

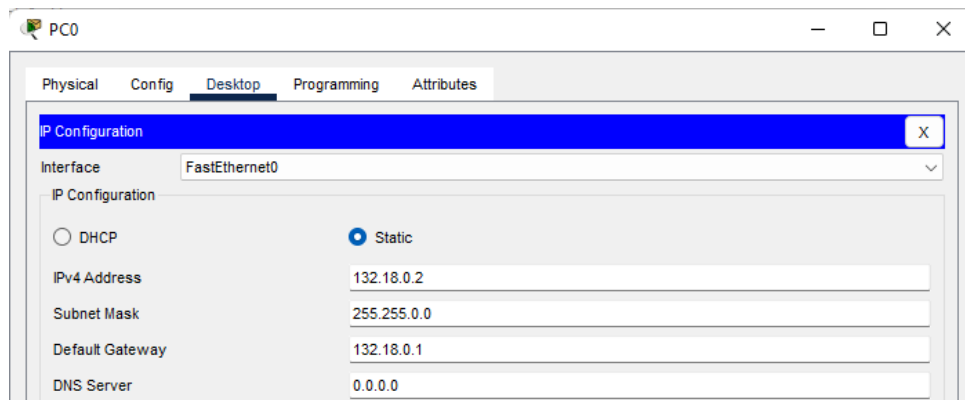


Figure 23 PC0 Settings

- PC1

UNIVERSIDAD

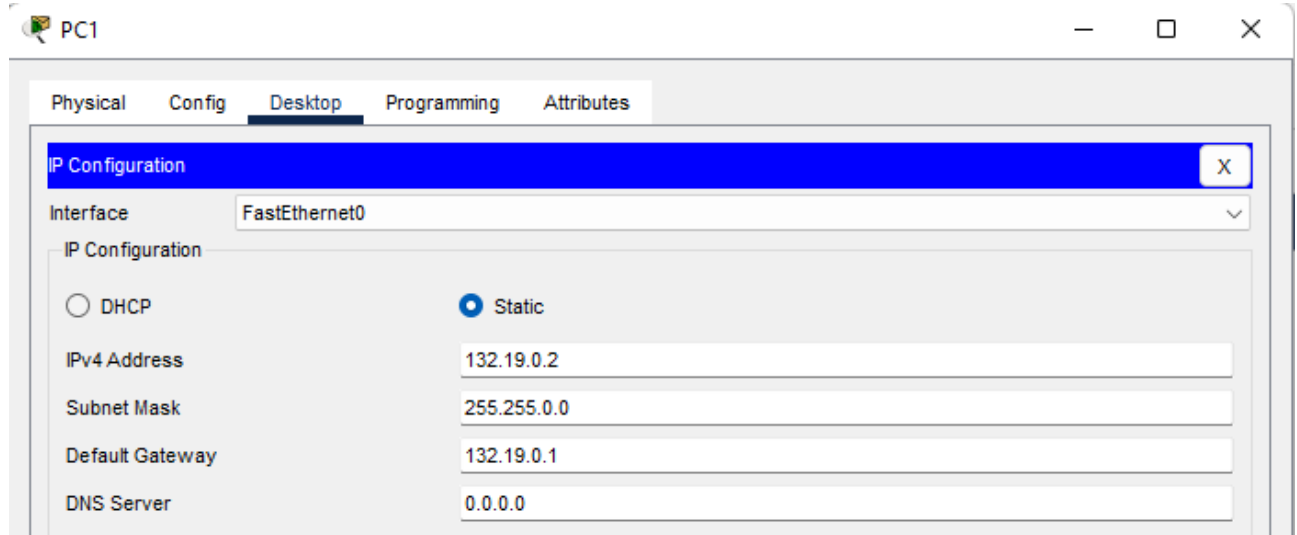


Figure 24 PC1 Settings

- PC2

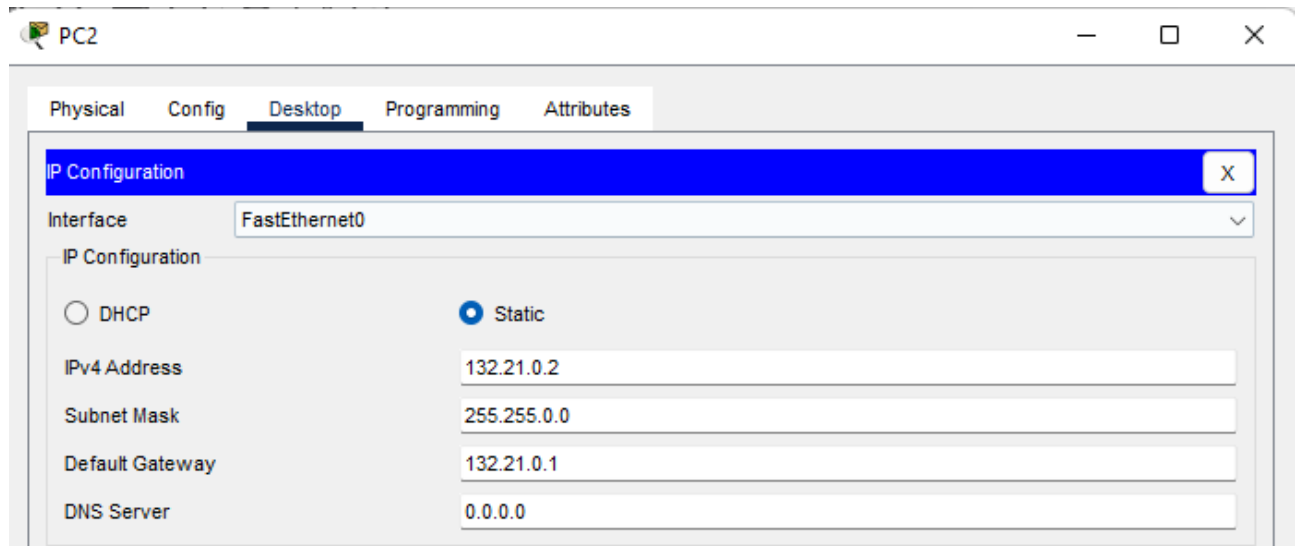


Figure 25 PC2 Settings

- PC3

UNIVERSIDAD

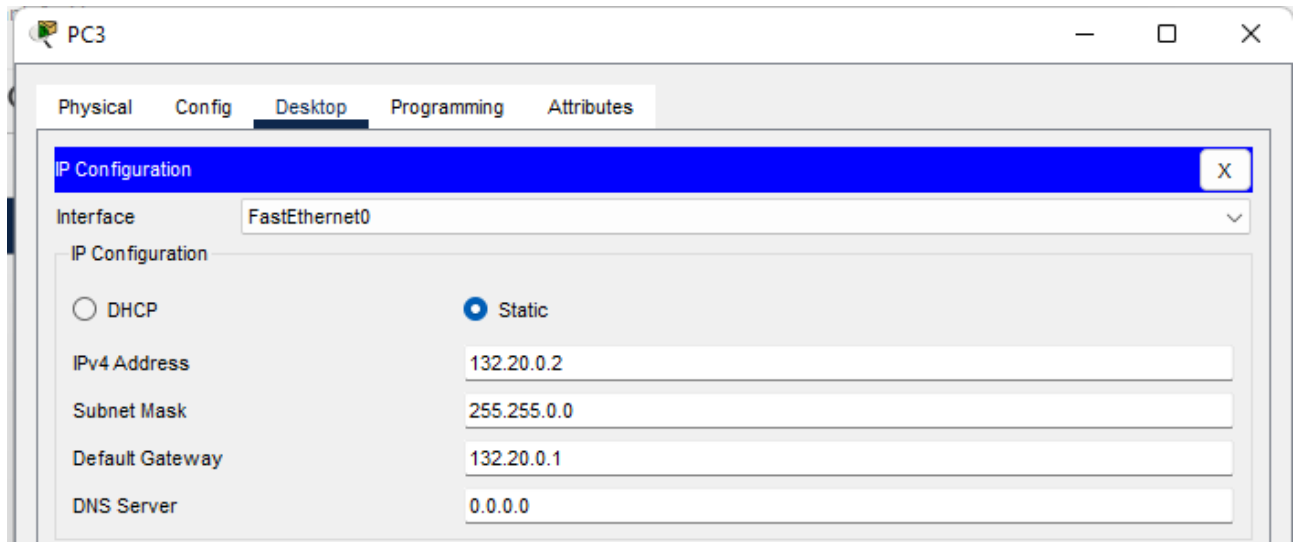


Figure 26 PC3 Settings

The network configuration result is as follows:

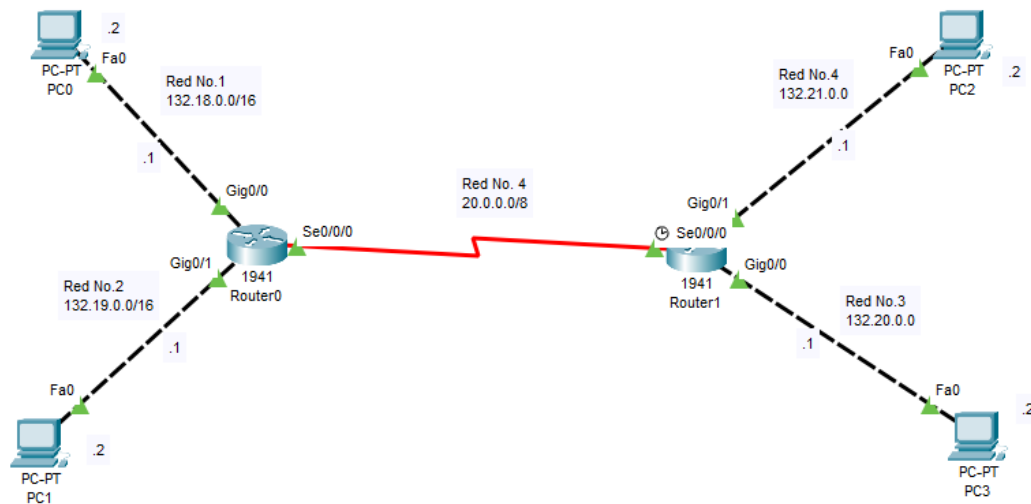


Figure 27 general view of the network

Test connectivity between the PCs on the same LAN and to different networks. Which ones work and which ones don't?

Only the connections between devices within the same network and their respective gateways function properly, specifically those that are closest to the router.

Connectivity tests within the same LAN.

1. **PC0 (IP: 132.18.0.2):**
 - Test ping 132.18.0.1 (Router0)

```
C:\>ping 132.18.0.1

Pinging 132.18.0.1 with 32 bytes of data:

Reply from 132.18.0.1: bytes=32 time<1ms TTL=255
Reply from 132.18.0.1: bytes=32 time<1ms TTL=255
Reply from 132.18.0.1: bytes=32 time<1ms TTL=255
Reply from 132.18.0.1: bytes=32 time<1ms TTL=255

Ping statistics for 132.18.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 28 successful ping

- Test ping 132.19.0.2 (PC1)

```
Pinging 132.19.0.1 with 32 bytes of data:

Reply from 132.19.0.1: bytes=32 time<1ms TTL=255
Reply from 132.19.0.1: bytes=32 time<1ms TTL=255
Reply from 132.19.0.1: bytes=32 time<1ms TTL=255
Reply from 132.19.0.1: bytes=32 time<1ms TTL=255

Ping statistics for 132.19.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 29 successful ping

- Test ping 132.21.0.2 (PC2)

```
C:\>ping 132.21.0.2

Pinging 132.21.0.2 with 32 bytes of data:

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

Ping statistics for 132.21.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 30 failed ping

- Test ping 132.20.0.2 (PC3)

```
C:\>ping 132.20.0.2

Pinging 132.20.0.2 with 32 bytes of data:

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

Ping statistics for 132.20.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 31 failed ping

2. PC1 (IP: 132.19.0.2):

- Test ping 132.19.0.1 (Router0)

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 132.19.0.1

Pinging 132.19.0.1 with 32 bytes of data:

Reply from 132.19.0.1: bytes=32 time<1ms TTL=255
Reply from 132.19.0.1: bytes=32 time<1ms TTL=255
Reply from 132.19.0.1: bytes=32 time<1ms TTL=255
Reply from 132.19.0.1: bytes=32 time<1ms TTL=255

Ping statistics for 132.19.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 32 successful ping

- Test ping 132.18.0.2 (PC0)

```
C:\>ping 132.18.0.2

Pinging 132.18.0.2 with 32 bytes of data:

Reply from 132.18.0.2: bytes=32 time<1ms TTL=127
Reply from 132.18.0.2: bytes=32 time<1ms TTL=127
Reply from 132.18.0.2: bytes=32 time<1ms TTL=127
Reply from 132.18.0.2: bytes=32 time=13ms TTL=127

Ping statistics for 132.18.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 13ms, Average = 3ms
```

Figure 33 successful ping

- Test ping 132.20.0.2 (PC2)

```
C:\>ping 132.20.0.2

Pinging 132.20.0.2 with 32 bytes of data:

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

Ping statistics for 132.20.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 34 failed ping

- Test ping 132.21.0.2 (PC3)

```
C:\>ping 132.21.0.2

Pinging 132.21.0.2 with 32 bytes of data:

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

Ping statistics for 132.21.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 35 successful ping

3. **PC2 (IP: 132.21.0.2):**

- Test ping 132.21.0.1 (Router1)

```
C:\>ping 132.21.0.1

Pinging 132.21.0.1 with 32 bytes of data:

Reply from 132.21.0.1: bytes=32 time<1ms TTL=255
Reply from 132.21.0.1: bytes=32 time<1ms TTL=255
Reply from 132.21.0.1: bytes=32 time<1ms TTL=255
Reply from 132.21.0.1: bytes=32 time<1ms TTL=255

Ping statistics for 132.21.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 36 successful ping

- Test ping 132.19.0.2 (PC0)

```
C:\>ping 132.19.0.1

Pinging 132.19.0.1 with 32 bytes of data:

Reply from 132.21.0.1: Destination host unreachable.
Request timed out.
Reply from 132.21.0.1: Destination host unreachable.
Reply from 132.21.0.1: Destination host unreachable.

Ping statistics for 132.19.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 37 failed ping

- Test ping 132.18.0.2 (PC1)

```
C:\>ping 132.18.0.1

Pinging 132.18.0.1 with 32 bytes of data:

Reply from 132.21.0.1: Destination host unreachable.
Reply from 132.21.0.1: Destination host unreachable.
Request timed out.
Reply from 132.21.0.1: Destination host unreachable.

Ping statistics for 132.18.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 38 failed ping

- Test ping 132.20.0.2 (PC3)

```
C:\>ping 132.20.0.2

Pinging 132.20.0.2 with 32 bytes of data:

Reply from 132.20.0.2: bytes=32 time<1ms TTL=127
Reply from 132.20.0.2: bytes=32 time<1ms TTL=127
Reply from 132.20.0.2: bytes=32 time<1ms TTL=127
Reply from 132.20.0.2: bytes=32 time<1ms TTL=127

Ping statistics for 132.20.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 39 successful ping

4. PC3 (IP: 132.20.0.2):

- Test ping 132.20.0.1 (Router1)


```
C:\>ping 132.20.0.1

Pinging 132.20.0.1 with 32 bytes of data:

Reply from 132.20.0.1: bytes=32 time<1ms TTL=255
Reply from 132.20.0.1: bytes=32 time<1ms TTL=255
Reply from 132.20.0.1: bytes=32 time<1ms TTL=255
Reply from 132.20.0.1: bytes=32 time<1ms TTL=255

Ping statistics for 132.20.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 40 successful ping

- Test ping 132.19.0.2 (PC0)

```
C:\>ping 132.19.0.2

Pinging 132.19.0.2 with 32 bytes of data:

Reply from 132.20.0.1: Destination host unreachable.
Reply from 132.20.0.1: Destination host unreachable.
Reply from 132.20.0.1: Destination host unreachable.
Reply from 132.20.0.1: Destination host unreachable.

Ping statistics for 132.19.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 41 failed ping

- Test ping 132.18.0.2 (PC1)

```
C:\>ping 132.18.0.2

Pinging 132.18.0.2 with 32 bytes of data:

Reply from 132.20.0.1: Destination host unreachable.
Reply from 132.20.0.1: Destination host unreachable.
Reply from 132.20.0.1: Destination host unreachable.
Reply from 132.20.0.1: Destination host unreachable.

Ping statistics for 132.18.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figure 42 failed ping

- Test ping 132.21.0.2 (PC2)

```
C:\>ping 132.21.0.2

Pinging 132.21.0.2 with 32 bytes of data:

Reply from 132.21.0.2: bytes=32 time<1ms TTL=127
Reply from 132.21.0.2: bytes=32 time<1ms TTL=127
Reply from 132.21.0.2: bytes=32 time<1ms TTL=127
Reply from 132.21.0.2: bytes=32 time<1ms TTL=127

Ping statistics for 132.21.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 43 successful ping

Request timed out: Indicates that the request packet did not receive a response within the timeout period. This is a general issue of not receiving any response, which can be due to various factors such as the target device being unavailable, a firewall blocking the traffic, or physical connectivity issues. In this case, the packets do not arrive because the static route is not configured.

Destination host unreachable: Specifically indicates that a router or intermediate device does not have a route to reach the destination. The router that does not know how to forward the packet sends an error message back to the source. This is also due to the static route not being configured.

Now we will configure the static routes on each router to enable ping between the different devices. On each router, we will include the routes needed to reach networks that are not directly connected.

In router0

```
Jorge>enable
Jorge#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Jorge(config)#ip route 132.20.0.0 255.255.0.0 20.0.0.2
Jorge(config)#ip route 132.21.0.0 255.255.0.0 20.0.0.2
Jorge(config)#exit
Jorge#
%SYS-5-CONFIG_I: Configured from console by console

Jorge#write memory
Building configuration...
[OK]
```

Figure 44 route configuration

In router1

```
Gamboa>enable
Gamboa#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Gamboa(config)#ip route 132.18.0.0 255.255.0.0 20.0.0.1
Gamboa(config)#ip route 132.19.0.0 255.255.0.0 20.0.0.1
Gamboa(config)#exit
Gamboa#
%SYS-5-CONFIG_I: Configured from console by console

Gamboa#write memory
Building configuration...
[OK]
```

Figure 45 static route configuration

Proceed to test if the static connection between the devices was made correctly.

UNIVERSIDAD

- Ping between PC0 and PC2.

```
C:\>ping 132.21.0.2

Pinging 132.21.0.2 with 32 bytes of data:

Reply from 132.21.0.2: bytes=32 time=1ms TTL=126
Reply from 132.21.0.2: bytes=32 time=1ms TTL=126
Reply from 132.21.0.2: bytes=32 time=20ms TTL=126
Reply from 132.21.0.2: bytes=32 time=3ms TTL=126

Ping statistics for 132.21.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 20ms, Average = 6ms
```

Figure 46 successful ping

```
C:\>tracert 132.21.0.2

Tracing route to 132.21.0.2 over a maximum of 30 hops:

  1  0 ms    0 ms    0 ms    132.18.0.1
  2  5 ms    0 ms    2 ms    20.0.0.2
  3 12 ms    1 ms    4 ms    132.21.0.2

Trace complete.
```

Figure 47 trace view

- Ping between PC3 and PC1.

```
C:\>ping 132.19.0.2

Pinging 132.19.0.2 with 32 bytes of data:

Reply from 132.19.0.2: bytes=32 time=24ms TTL=126
Reply from 132.19.0.2: bytes=32 time=14ms TTL=126
Reply from 132.19.0.2: bytes=32 time=16ms TTL=126
Reply from 132.19.0.2: bytes=32 time=1ms TTL=126

Ping statistics for 132.19.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 24ms, Average = 13ms
```

Figure 48 successful ping

```
C:\>tracert 132.19.0.2

Tracing route to 132.19.0.2 over a maximum of 30 hops:

  1  0 ms    0 ms    0 ms    132.20.0.1
  2  0 ms    1 ms    2 ms    20.0.0.1
  3  6 ms    0 ms    0 ms    132.19.0.2

Trace complete.
```

Figure 49 trace view

For the network configuration of the next Packet Tracer, the exact same process is followed, but the networks 132.18.0.0, 132.19.0.0, 132.20.0.0, and 132.21.0.0 are replaced by the following:

UNIVERSIDAD

- 72.0.0.0/8
- 73.0.0.0/8
- 74.0.0.0/8
- 75.0.0.0/8

The result is as follows:

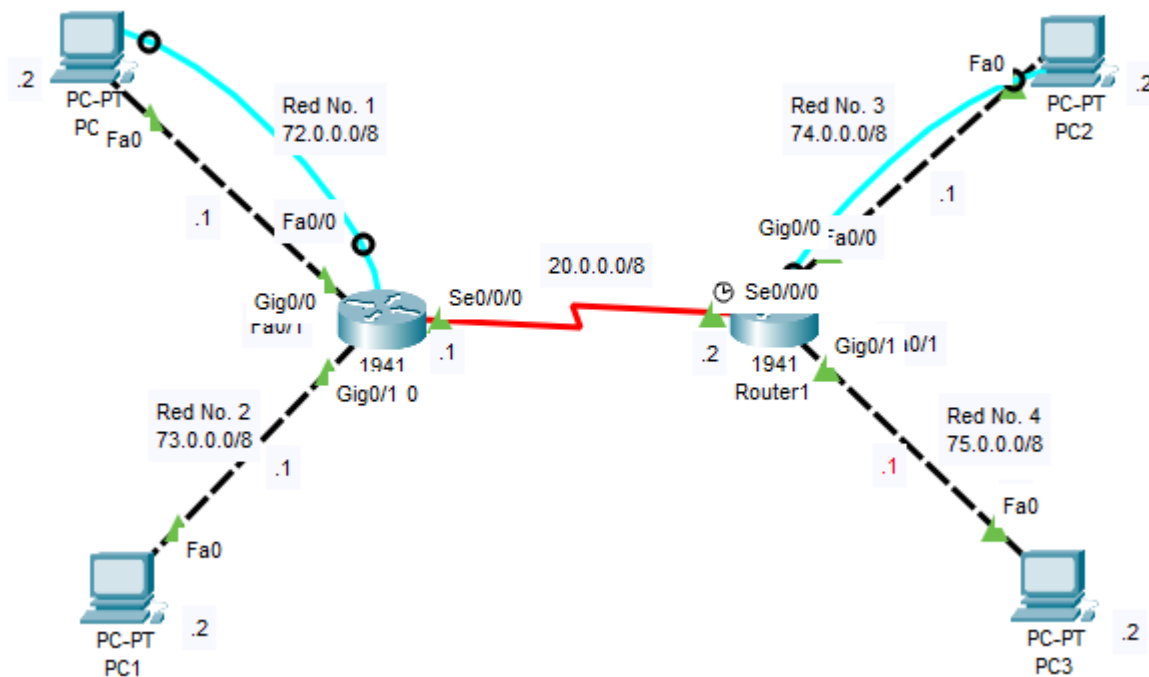


Figure 50 general network view

Now we connect the two Packet Tracers using the multi-user option.

We will start with the configuration file that has the IPs starting with 132.x.x.x.

First, we place a peer module and rename it to RED_Jorge.



RED_Jorge

Figure 51 peer module

UNIVERSIDAD

Then we go to the multiuser option and select Listen.

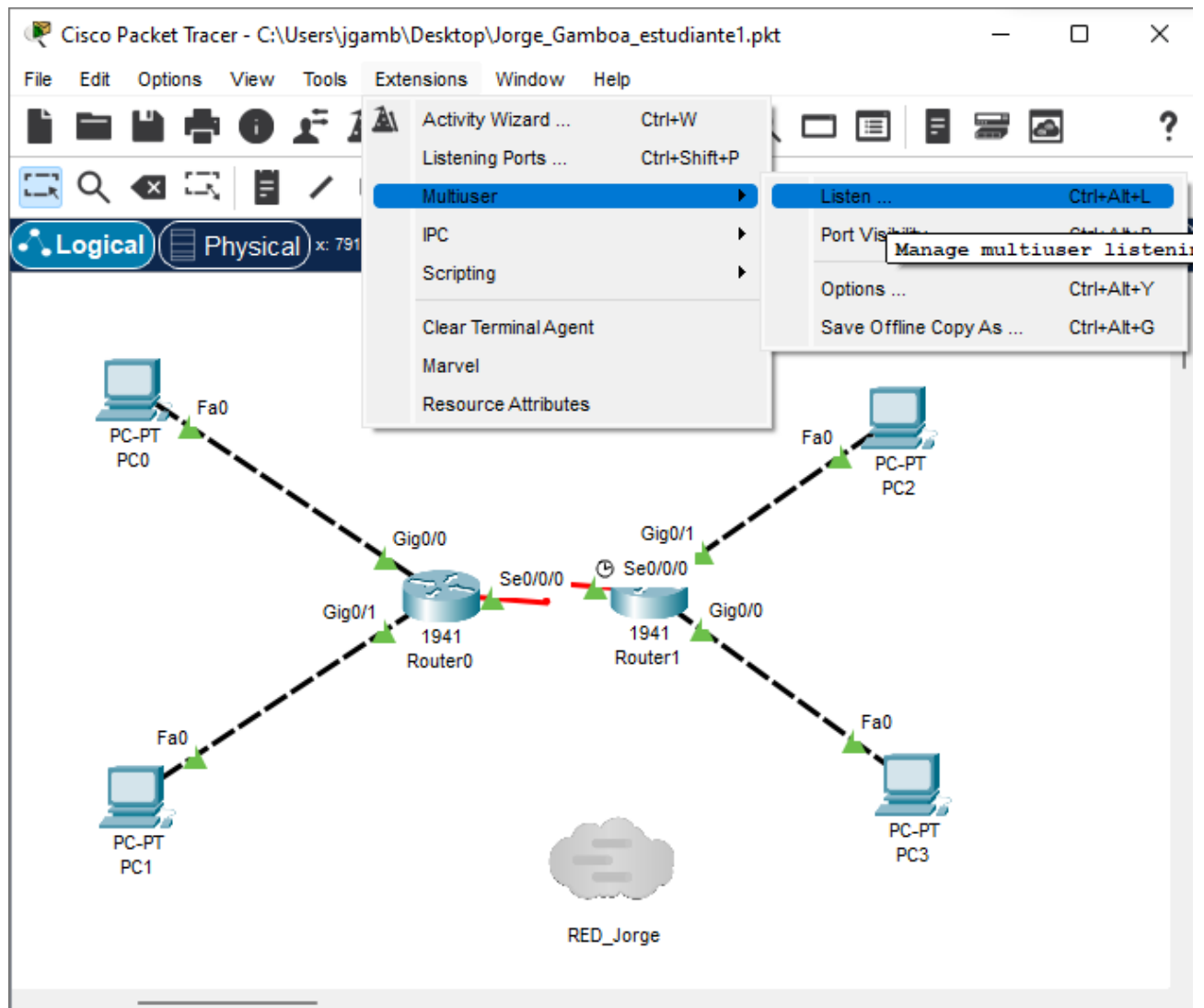


Figure 52 multi-user configuration

We set the password to cisco, and then in existing remote networks, we select Always accept, and in new remote networks, we also set it to Always accept.

Multiuser Listen

Local Listening Address:

192.168.56.1:38000
192.168.1.3:38000
127.0.0.1:38000

Port Number 38000

Password

Existing Remote Networks

☒ Always Accept
☐ Always Deny
☐ Prompt

New Remote Networks

☒ Always Accept
☐ Always Deny
☐ Prompt

Stop Listening OK Cancel

Figure 53 multi-user configuration

Then we click on the peer module and set the incoming option.

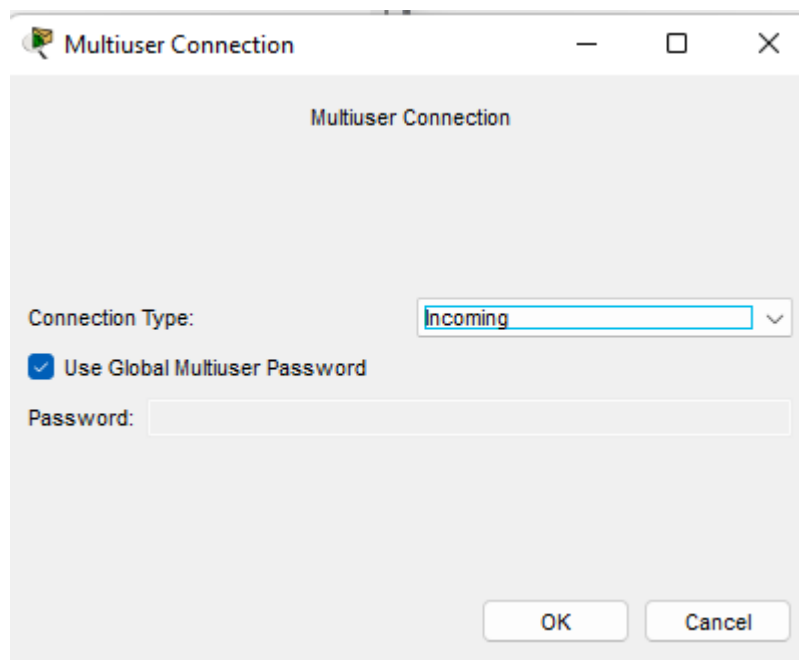


Figure 54 Incoming connection

Then we create a serial connection between Router1 and the peer module RED_Jorge. The connection will be between serial0/0/1 on the router, and we create a new connection on the peer module.

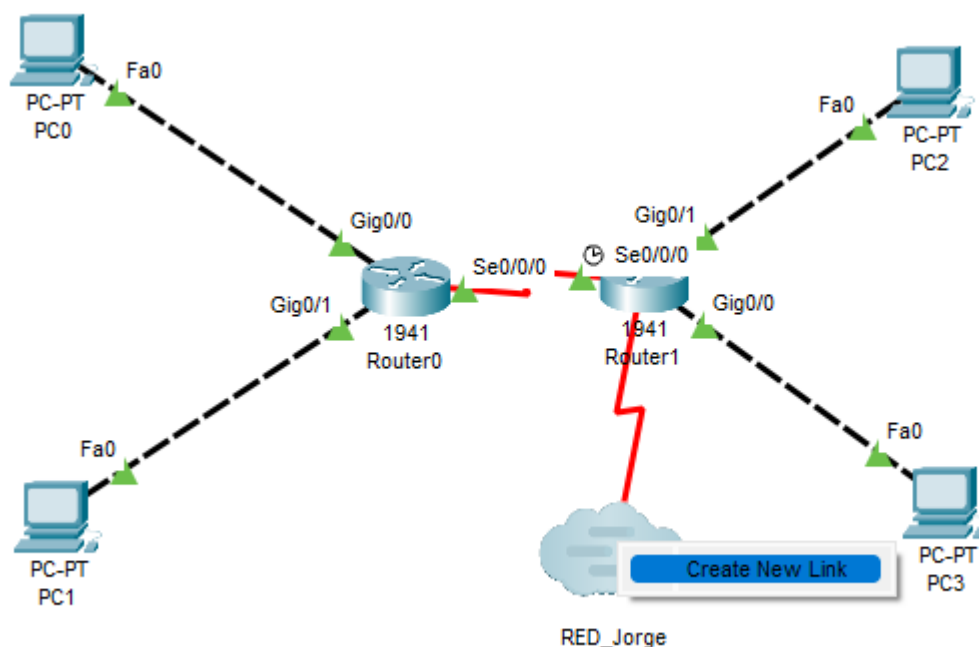


Figure 55 connection with peer module

We proceed to do the same in the other Packet Tracer file, setting up a peer module called RED_Camila and following the same configuration as before, but instead of setting the incoming option, we select the Outgoing option and enter the information for the module we want to access.

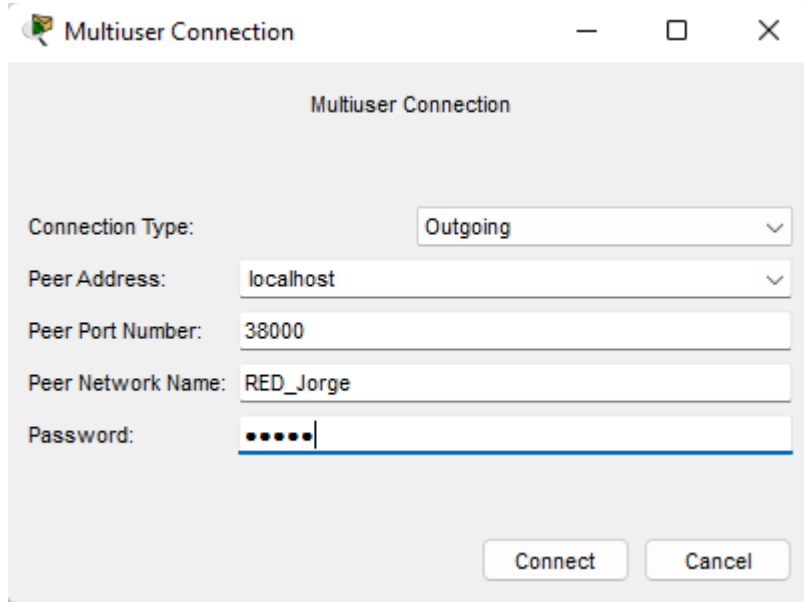


Figure 56 multiuser outgoing

On the peer module, we create a connection between Router1 through the serial0/0/1 connection and the peer module named RED_Camila via Link 0 (Router1 Serial0/0/1).

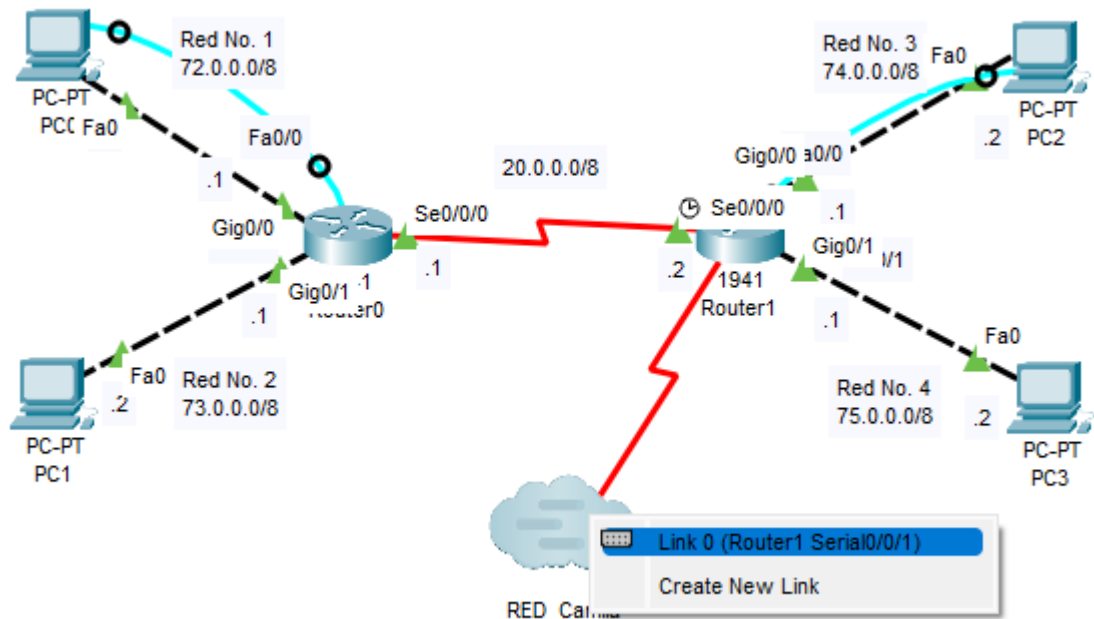


Figure 57 connection to another network

UNIVERSIDAD

We configure the network 30.0.0.0 for the connection between Router1 in both Packet Tracer files.

In router1 Gamboa

```
Gamboa(config)#interface Serial0/0/1
Gamboa(config-if)#description "configure peer module"
Gamboa(config-if)#ip address 30.0.0.4 255.0.0.0
Gamboa(config-if)#no shutdown
```

Figure 58 configuration interface

In router1 Torres

```
Torres(config)#interface serial 0/0/1
Torres(config-if)#ip address 30.0.0.3 255.0.0.0
Torres(config-if)#no shutdown
```

Figure 59 configuration interface

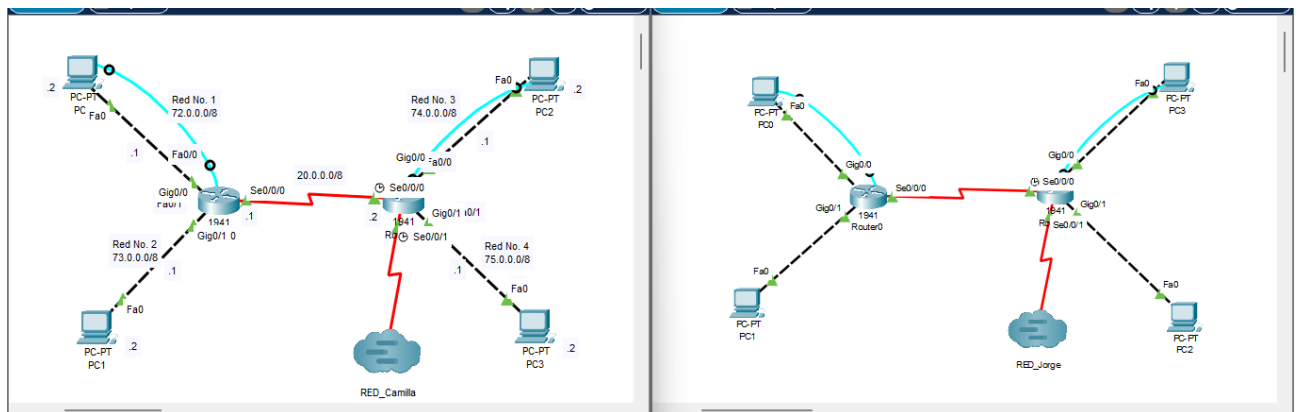


Figure 60 general view

We configure the static routes on Router0 in each Packet Tracer file.

```
Jorge>enable
Password:
Jorge#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Jorge(config)#ip route 132.18.0.0 255.255.0.0 72.0.0.1
Jorge(config)#ip route 132.18.0.0 255.255.0.0 20.0.0.2
Jorge(config)#ip route 132.19.0.0 255.255.0.0 20.0.0.2
Jorge(config)#ip route 132.20.0.0 255.255.0.0 20.0.0.2
Jorge(config)#ip route 132.21.0.0 255.255.0.0 20.0.0.2
Jorge(config)#exit
Jorge#
%SYS-5-CONFIG_I: Configured from console by console
Jorge#
```

Figure 61 static route configuration

UNIVERSIDAD

```
Jorge#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Jorge(config)#ip route 75.0.0.0 255.0.0.0 20.0.0.2
Jorge(config)#ip route 74.0.0.0 255.0.0.0 20.0.0.2
Jorge(config)#ip route 73.0.0.0 255.0.0.0 20.0.0.2
Jorge(config)#ip route 72.0.0.0 255.0.0.0 20.0.0.2
Jorge(config)#
```

Figure 62 static route configuration

Now we will test the connection between the two networks.

- 72.0.0.2 to 132.18.0.2

```
C:\>tracert 132.18.0.2

Tracing route to 132.18.0.2 over a maximum of 30 hops:

  1  0 ms    0 ms    0 ms    72.0.0.1
  2  7 ms    1 ms    1 ms    20.0.0.2
  3  44 ms   29 ms   31 ms   30.0.0.4
  4  *        *        *        Request timed out.
  5  21 ms   24 ms   36 ms   132.18.0.2

Trace complete.
```

Figure 63 tracer successful

- 132.20.0.2 to 75.0.0.2

```
C:\>tracert 75.0.0.2

Tracing route to 75.0.0.2 over a maximum of 30 hops:

  1  0 ms    0 ms    0 ms    132.20.0.1
  2  59 ms   28 ms   78 ms   30.0.0.3
  3  *        15 ms   23 ms   75.0.0.2

Trace complete.
```

Figure 64 tracer successful

- 132.19.0.2 to 73.0.0.2

```
C:\>tracert 73.0.0.2

Tracing route to 73.0.0.2 over a maximum of 30 hops:

  1  0 ms    0 ms    0 ms    132.19.0.1
  2  0 ms    1 ms    10 ms   20.0.0.2
  3  24 ms   37 ms   29 ms   30.0.0.3
  4  *        *        *        Request timed out.
  5  21 ms   32 ms   23 ms   73.0.0.2

Trace complete.
```

Figure 65 tracer successful

- 74.0.0.2 to 132.21.0.2

```
Tracing route to 132.21.0.2 over a maximum of 30 hops:

  1  0 ms      0 ms      0 ms      74.0.0.1
  2  19 ms     25 ms     26 ms     30.0.0.4
  3  *          21 ms     27 ms     132.21.0.2

Trace complete.
```

Figure 66 tracer successful

To confirm that the interface configurations were correct, the command show ip interface brief is used on each router

```
Camila>show ip interface brief
Interface          IP-Address      OK? Method Status        Protocol
GigabitEthernet0/0 72.0.0.1        YES NVRAM  up            up
GigabitEthernet0/1 73.0.0.1        YES NVRAM  up            up
Serial0/0/0         20.0.0.1        YES manual up            up
Serial0/0/1         unassigned      YES manual down          down
Vlan1               unassigned      YES unset  administratively down down
Camila>
```

Figure 67 configuration interface Router0 RED_Camila

Breakdown of the columns:

Interface: This column shows the names of device interfaces. In this case, it includes Gigabit Ethernet, Serial, and Vlan interfaces.

IP-Address: Displays the IP address assigned to each interface. For example:

- GigabitEthernet0/0 has the IP 72.0.0.1.
- Serial0/0/1 and Vlan1 do not have an IP address assigned (indicated as unassigned).

OK?: Indicates whether the IP address configuration is correct (YES means the configuration is valid).

Method: Shows how the IP address was configured:

- **NVRAM:** The IP was configured from NVRAM memory, usually during initial setup.
- **manual:** The IP was configured manually.
- **unset:** No IP address has been configured.

Status: Indicates the operational status of the interface:

- **Up:** The interface is active and functions correctly.
- **Down:** The interface is not active.
- **administratively down:** The interface has been disabled administratively (typically by a command like shutdown).

Protocol: Displays the status of the interface protocol, determining whether the interface can send and receive data:

- **up:** The protocol is active.

UNIVERSIDAD

- **down:** The protocol is not active.

To confirm that the static routes were configured correctly, the command `show ip route` is used on each router.

```
Camila>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

 20.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C    20.0.0.0/8 is directly connected, Serial0/0/0
L    20.0.0.1/32 is directly connected, Serial0/0/0
 72.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C    72.0.0.0/8 is directly connected, GigabitEthernet0/0
L    72.0.0.1/32 is directly connected, GigabitEthernet0/0
 73.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C    73.0.0.0/8 is directly connected, GigabitEthernet0/1
L    73.0.0.1/32 is directly connected, GigabitEthernet0/1
S    74.0.0.0/8 [1/0] via 20.0.0.2
S    75.0.0.0/8 [1/0] via 20.0.0.2
S   132.18.0.0/16 [1/0] via 20.0.0.2
--More--
```

Figure 68 configuration static route RED_Camila Router0

Summary of the show ip route Output:

- **Routing Codes:** Indicates the types of routes, such as:
 - C: Connected
 - S: Static
 - L: Local
- **Gateway of Last Resort:** Not set, meaning no default route is configured.

Route Table Entries:

1. **20.0.0.0/8:**
 - C: Directly connected to **Serial0/0/0**
 - L: Local IP **20.0.0.1** on **Serial0/0/0**
2. **72.0.0.0/8:**
 - C: Directly connected to **GigabitEthernet0/0**
 - L: Local IP **72.0.0.1** on **GigabitEthernet0/0**

UNIVERSIDAD

3. **73.0.0.0/8:**

- **C:** Directly connected to **GigabitEthernet0/1**
- **L:** Local IP **73.0.0.1** on **GigabitEthernet0/1**

4. **Static Routes:**

- **S:** 74.0.0.0/8, 75.0.0.0/8, and 132.18.0.0/16 are static routes pointing to next hop **20.0.0.2**.

Base Software Installation

Part of the foundational platform of an organization's IT infrastructure involves web services, which can be hosted within the company's data center or on a cloud server. These services store the organization's web pages and are used by various clients. In this lab, we will implement this service.

1. Web Service Installation

1.1. Solaris

- We don't have to install Apache because it was included when we set up the machine. So, we start the service with 'svcadm -v enable /network/http:apache24' command

```
root@solaris:~# svcadm -v enable /network/http:apache24
svc:/network/http:apache24 activado.
```

Figure 69. Initializing Apache

- We navigate to the **/var/apache2/2.4/htdocs** directory, where the web server's HTML file is stored

```
root@solaris:~# cd /var/apache2/2.4/htdocs
root@solaris:/var/apache2/2.4/htdocs# ls
index.html
root@solaris:/var/apache2/2.4/htdocs# █
```

Figure 70. Moving into htdocs directory

- We edit the index.html file

Modificado

```

<html>
<head>
  <title>Solaris Web Server </tittle>
</head>
<body>
<h1>Hello! Solaris Web Server!</h1>
<h2> We are: </h2>
<ol>
<li>Camila</li>
<li>Jorge</li>
</ol>
</body>
</html>

```

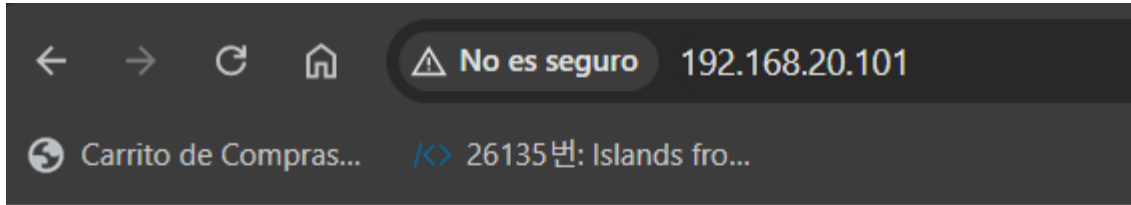
[1 línea leída]

^G Ver ayuda
^O Guardar
^W Bu

^X Salir
^R Leer fich.
^_ Reemplazar
^U Pegar txt
^T Ortografía
^_ Ir a línea

Figure 71. Editing the main file on Apache web server

- We open the browser and enter the IP address of the Solaris machine



Hello! Solaris Web Server!

We are:

1. Camila
2. Jorge

Figure 72. Verifying that Solaris web server is working

UNIVERSIDAD

- To configure the DNS so that the page is accessible by the domain name, we edit the zone file that we set up in Lab 3 (gamboa.com.it.hosts) and assign the corresponding IP address of the web server to www

Modificado

```

;
; /etc/DNS/named.soa file
;
; name server SOA file
;

@ IN SOA ns1.gamboa.com.it. root.gamboa.com.it. (
2020050101 ; serial
43200 ; refresh
3600 ; retry
432000 ; expire
86400 ; minimum time-to-live
)

gamboa.com.it. IN NS ns1.gamboa.com.it.
ns1             IN      A      10.2.77.194
ns2             IN      A      10.2.77.193

www             IN      A      10.2.77.194
mail            IN      A      10.2.77.200
servicios       IN      A      10.2.77.201
windows IN      A      10.2.77.196

juegos          IN      AAAA    ::ffff:0a02:4dcb
reco            IN      AAAA    ::ffff:0a02:4dca

correoX         IN CNAME      mail.gamboa.com.it.
onix            IN CNAME      servicios.gamboa.com.it.
sol             IN CNAME      reco.gamboa.com.it.
^G Ver ayuda
^X Salir
^R Leer fich.
^N Reemplazar
^U Pegar txt
^T OrtografÃ-a
Ir a lÃ-nea
de formato DOS) ]

```

Figure 73. Configuring zone file gamboa.com.it to resolve the domain name of the web server (www.gamboa.com.it)

- We open the browser and enter the domain name of Solaris

Hello! Solaris Web Server!

We are:

1. Camila
2. Jorge

Figure 74. Accessing the Solaris web server by domain name

1.2. Linux Slackware

- We download the .tar.gz file from the [Slackbuilds](https://slackbuilds.org/slackbuilds/15.0/network/nginx.tar.gz) repository using 'wget <https://slackbuilds.org/slackbuilds/15.0/network/nginx.tar.gz>'

```
root@andrea:~# wget https://slackbuilds.org/slackbuilds/15.0/network/nginx.tar.gz
--2024-10-27 15:16:41-- https://slackbuilds.org/slackbuilds/15.0/network/nginx.tar.gz
Resolving slackbuilds.org (slackbuilds.org)... 66.85.79.67, 2604:5800:0:90::67
Connecting to slackbuilds.org (slackbuilds.org)|66.85.79.67|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5710 (5.6K) [application/x-gzip]
Saving to: 'nginx.tar.gz'

nginx.tar.gz          100%[=====>]  5.58K  --.-KB/s    in 0s

2024-10-27 15:16:46 (1.44 GB/s) - 'nginx.tar.gz' saved [5710/5710]

root@andrea:~# ls
Makefile      find_file.sh*  libunistring-0.9.10/  newgroup.sh*  test.c
archivo.txt*  find_file_word.sh*  libunistring-0.9.10.tar.gz  newuser.sh*  test_output*
ejemplo.txt   find_word.sh*    list_files.sh*         nginx.tar.gz  total_lines.sh*
file_menu.sh* first_lines.sh*  log_menu.sh*          process_menu.sh*
find_dir.sh*  last_lines.sh*  menu_search.sh*        test*
```

Figure 75. Downloading nginx.tar.gz file from SlackBuilds repository

- We decompress the file and navigate to the nginx directory

UNIVERSIDAD

```
root@andrea:~# tar -xzvf nginx.tar.gz
nginx/
nginx/doinst.sh
nginx/nginx.info
nginx/nginx.logrotate
nginx/slack-desc
nginx/rc.nginx
nginx/README
nginx/nginx.SlackBuild
root@andrea:~# _
```

Figure 76. Decompressing nginx.tar.gz

- We download the version of the nginx required by Slackbuild (version 1.26.2) using **wget** <http://nginx.org/download/nginx-1.26.2.tar.gz>

```
root@andrea:~# cd nginx
root@andrea:~/nginx# wget http://nginx.org/download/nginx-1.26.2.tar.gz
--2024-10-27 15:21:17-- http://nginx.org/download/nginx-1.26.2.tar.gz
Resolving nginx.org (nginx.org)... 52.58.199.22, 3.125.197.172, 2a05:d014:5c0:2600::6, ...
Connecting to nginx.org (nginx.org)|52.58.199.22|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1244789 (1.2M) [application/octet-stream]
Saving to: 'nginx-1.26.2.tar.gz'

nginx-1.26.2.tar.gz      100%[=====>] 1.19M 1.16MB/s in 1.0s
2024-10-27 15:21:24 (1.16 MB/s) - 'nginx-1.26.2.tar.gz' saved [1244789/1244789]
root@andrea:~/nginx#
```

Figure 77. Downloading nginx version

- We decompress the downloaded file using the command '**-xzvf nginx-1.26.2**'

UNIVERSIDAD

```
nginx-1.26.2/auto/lib/libxslt/  
nginx-1.26.2/auto/lib/perl/  
nginx-1.26.2/auto/lib/geoip/  
nginx-1.26.2/auto/lib/zlib/  
nginx-1.26.2/auto/lib/google-perftools/  
nginx-1.26.2/auto/lib/make  
nginx-1.26.2/auto/lib/pcre/  
nginx-1.26.2/auto/lib/openssl/  
nginx-1.26.2/auto/lib/conf  
nginx-1.26.2/auto/lib/openssl/makefile.bcc  
nginx-1.26.2/auto/lib/openssl/makefile.msuc  
nginx-1.26.2/auto/lib/openssl/make  
nginx-1.26.2/auto/lib/openssl/conf  
nginx-1.26.2/auto/lib/pcre/makefile.bcc  
nginx-1.26.2/auto/lib/pcre/makefile.msuc  
nginx-1.26.2/auto/lib/pcre/make  
nginx-1.26.2/auto/lib/pcre/conf  
nginx-1.26.2/auto/lib/pcre/makefile.owc  
nginx-1.26.2/auto/lib/google-perftools/conf  
nginx-1.26.2/auto/lib/zlib/makefile.bcc  
nginx-1.26.2/auto/lib/zlib/makefile.msuc  
nginx-1.26.2/auto/lib/zlib/make  
nginx-1.26.2/auto/lib/zlib/conf  
nginx-1.26.2/auto/lib/zlib/makefile.owc  
nginx-1.26.2/auto/lib/geoip/conf  
nginx-1.26.2/auto/lib/perl/make  
nginx-1.26.2/auto/lib/perl/conf  
nginx-1.26.2/auto/lib/libxslt/conf  
nginx-1.26.2/auto/lib/libatomic/make  
nginx-1.26.2/auto/lib/libatomic/conf  
nginx-1.26.2/auto/lib/libgd/conf  
nginx-1.26.2/auto/types/typedef  
nginx-1.26.2/auto/types/value  
nginx-1.26.2/auto/types/uinptr_t  
nginx-1.26.2/auto/types/sizeof  
nginx-1.26.2/man/nginx.8  
root@andrea:~/nginx# _
```

Figure 78. Decompressing nginx version

- We configure nginx package using **./configure**

UNIVERSIDAD

```
checking for pread() ... found
checking for pwrite() ... found
checking for pwritev() ... found
checking for strerrordesc_np() ... found
checking for localtime_r() ... found
checking for clock_gettime(CLOCK_MONOTONIC) ... found
checking for posix_memalign() ... found
checking for memalign() ... found
checking for mmap(MAP_ANON|MAP_SHARED) ... found
checking for mmap("/dev/zero", MAP_SHARED) ... found
checking for System V shared memory ... found
checking for POSIX semaphores ... found
checking for struct msghdr.msg_control ... found
checking for ioctl(FIONBIO) ... found
checking for ioctl(FIONREAD) ... found
checking for struct tm.tm_gmtoff ... found
checking for struct dirent.d_namlen ... not found
checking for struct dirent.d_type ... found
checking for sysconf(_SC_NPROCESSORS_ONLN) ... found
checking for sysconf(_SC_LEVEL1_DCACHE_LINESIZE) ... found
checking for openat(), fstatat() ... found
checking for getaddrinfo() ... found
configuring additional modules
checking for PCRE2 library ... not found
checking for PCRE library ... not found
checking for PCRE library in /usr/local/ ... not found
checking for PCRE library in /usr/include/pcre/ ... not found
checking for PCRE library in /usr/pkg/ ... not found
checking for PCRE library in /opt/local/ ... not found
checking for PCRE library in /opt/homebrew/ ... not found

./configure: error: the HTTP rewrite module requires the PCRE library.
You can either disable the module by using --without-http_rewrite_module
option, or install the PCRE library into the system, or build the PCRE library
statically from the source with nginx by using --with-pcre=<path> option.

root@andrea:~/nginx#
```

Figure 79. Configuring nginx

- However, as we saw in *Image 11*, we encountered a compilation error because the required libraries for Nginx were not installed. Therefore, we install the following libraries using **slackpkg**:
 - ✓ PCRE
 - ✓ Libxslt
 - ✓ Gd

UNIVERSIDAD

```
[820296/820296]
      Downloading http://ftp.slackware-brasil.com.br/slackware64-current/./slackwa
re64/l/pcre-8.45-x86_64-1.txz.asc...
--2024-10-27 15:30:04--  http://ftp.slackware-brasil.com.br/slackware64-current/slackware64/l/pcre-8
.45-x86_64-1.txz.asc
Resolving ftp.slackware-brasil.com.br (ftp.slackware-brasil.com.br)... 200.137.217.134
Connecting to ftp.slackware-brasil.com.br (ftp.slackware-brasil.com.br)|200.137.217.134|:80... conne
cted.
HTTP request sent, awaiting response... 200 OK
Length: 163 [text/plain]
Saving to: ■//var/cache/packages/./slackware64/l/pcre-8.45-x86_64-1.txz.asc■

//var/cache/packages/./s 100%[=====>]      163  --.-KB/s   in 0s

2024-10-27 15:30:10 (24.7 MB/s) - ■//var/cache/packages/./slackware64/l/pcre-8.45-x86_64-1.txz.asc■
saved [163/163]

      Package pcre-8.45-x86_64-1.txz is already in cache - not downloading
      Installing pcre-8.45-x86_64-1...
Verifying package pcre-8.45-x86_64-1.txz.
Installing package pcre-8.45-x86_64-1.txz:
PACKAGE DESCRIPTION:
# pcre (Perl-compatible regular expression library)
#
# The PCRE library is a set of functions that implement regular
# expression pattern matching using the same syntax and semantics as
# Perl 5, with just a few differences (documented in the man page).
#
# Homepage: https://www.pcre.org
#
Executing install script for pcre-8.45-x86_64-1.txz.
Package pcre-8.45-x86_64-1.txz installed.
Searching for NEW configuration files...
      No .new files found.

root@andrea:~#
```

Figure 80. Installing PCRE

UNIVERSIDAD

```
//var/cache/packages/./s 100%[=====>] 236.18K 182KB/s in 1.3s

2024-10-27 15:32:03 (182 KB/s) - ■//var/cache/packages/./slackware64/l/libxslt-1.1.42-x86_64-1.txz■
saved [241848/241848]

        Downloading http://ftp.slackware-brasil.com.br/slackware64-current/./slackw
re64/l/libxslt-1.1.42-x86_64-1.txz.asc...
--2024-10-27 15:32:03-- http://ftp.slackware-brasil.com.br/slackware64-current/slackware64/l/libxs
t-1.1.42-x86_64-1.txz.asc
Resolving ftp.slackware-brasil.com.br (ftp.slackware-brasil.com.br)... 200.137.217.134
Connecting to ftp.slackware-brasil.com.br (ftp.slackware-brasil.com.br)|200.137.217.134|:80... conn
cted.
HTTP request sent, awaiting response... 200 OK
Length: 195 [text/plain]
Saving to: ■//var/cache/packages/./slackware64/l/libxslt-1.1.42-x86_64-1.txz.asc■

//var/cache/packages/./s 100%[=====>] 195 --.-KB/s in 0s

2024-10-27 15:32:09 (27.6 MB/s) - ■//var/cache/packages/./slackware64/l/libxslt-1.1.42-x86_64-1.txz
asc■ saved [195/195]

        Package libxslt-1.1.42-x86_64-1.txz is already in cache - not downloading
        Installing libxslt-1.1.42-x86_64-1...
Verifying package libxslt-1.1.42-x86_64-1.txz.
Installing package libxslt-1.1.42-x86_64-1.txz:
PACKAGE DESCRIPTION:
# libxslt (XML transformation library)
#
# XSLT support for libxml2. (XSLT is a language used for transforming
# XML documents)
#
Executing install script for libxslt-1.1.42-x86_64-1.txz.
Package libxslt-1.1.42-x86_64-1.txz installed.
Searching for NEW configuration files...
        No .new files found.

root@andrea:~# _
```

Figure 81. Installing libxslt

UNIVERSIDAD

```
2024-10-27 15:33:22 (19.4 MB/s) - ■//var/cache/packages/./slackware64/l/gd-2.3.3-x86_64-3.txz.asc■ s
aved [163/163]

Package aspell-gd-0.1.1_1-x86_64-5.txz is already in cache - not downloading
Installing aspell-gd-0.1.1_1-x86_64-5...
Verifying package aspell-gd-0.1.1_1-x86_64-5.txz.
grep: /var/lib/pkgtools/setup/tmp/scand6b0778d6a2060a0513bbb880a312d79/install/slack-desc: binary fi
le matches
Installing package aspell-gd-0.1.1_1-x86_64-5.txz:
PACKAGE DESCRIPTION:
grep: /var/lib/pkgtools/setup/tmp/scand6b0778d6a2060a0513bbb880a312d79/install/slack-desc: binary fi
le matches
# aspell-gd
#
Package aspell-gd-0.1.1_1-x86_64-5.txz installed.
Package gd-2.3.3-x86_64-3.txz is already in cache - not downloading
Installing gd-2.3.3-x86_64-3...
Verifying package gd-2.3.3-x86_64-3.txz.
Installing package gd-2.3.3-x86_64-3.txz:
PACKAGE DESCRIPTION:
# gd (a graphics library)
#
# gd is a graphics library. It allows your code to quickly draw images
# complete with lines, arcs, text, multiple colors, cut and paste from
# other images, and flood fills, and write out the result as a PNG or
# JPEG file. This is particularly useful in web applications, where
# PNG and JPEG are two of the formats accepted for inline images by
# most browsers. The gd library was written by Thomas Boutell.
#
# Homepage: https://www.libgd.org
#
Executing install script for gd-2.3.3-x86_64-3.txz.
Package gd-2.3.3-x86_64-3.txz installed.
Searching for NEW configuration files...
No .new files found.

root@andrea:~# _
```

Figure 82. Installing gd

- Before that, we run the **./configure** command again

UNIVERSIDAD

```
checking for POSIX semaphores ... found
checking for struct msghdr.msg_control ... found
checking for ioctl(FIONBIO) ... found
checking for ioctl(FIONREAD) ... found
checking for struct tm.tm_gmtoff ... found
checking for struct dirent.d_namlen ... not found
checking for struct dirent.d_type ... found
checking for sysconf(_SC_NPROCESSORS_ONLN) ... found
checking for sysconf(_SC_LEVEL1_DCACHE_LINESIZE) ... found
checking for openat(), fstatat() ... found
checking for getaddrinfo() ... found
checking for PCRE2 library ... not found
checking for PCRE library ... found
checking for PCRE JIT support ... found
checking for zlib library ... found
creating objs/Makefile

Configuration summary
+ using system PCRE library
+ OpenSSL library is not used
+ using system zlib library

nginx path prefix: "/usr/local/nginx"
nginx binary file: "/usr/local/nginx/sbin/nginx"
nginx modules path: "/usr/local/nginx/modules"
nginx configuration prefix: "/usr/local/nginx/conf"
nginx configuration file: "/usr/local/nginx/conf/nginx.conf"
nginx pid file: "/usr/local/nginx/logs/nginx.pid"
nginx error log file: "/usr/local/nginx/logs/error.log"
nginx http access log file: "/usr/local/nginx/logs/access.log"
nginx http client request body temporary files: "client_body_temp"
nginx http proxy temporary files: "proxy_temp"
nginx http fastcgi temporary files: "fastcgi_temp"
nginx http uwsgi temporary files: "uwsgi_temp"
nginx http scgi temporary files: "scgi_temp"

root@andrea:~/nginx/nginx-1.26.2# _
```

Figure 83. Configuring nginx with the necessary libraries

- Then, we use the ‘**make**’ command to compile the package

UNIVERSIDAD

```
objs/src/http/modules/nginx_http_static_module.o \
objs/src/http/modules/nginx_http_autoindex_module.o \
objs/src/http/modules/nginx_http_index_module.o \
objs/src/http/modules/nginx_http_mirror_module.o \
objs/src/http/modules/nginx_http_try_files_module.o \
objs/src/http/modules/nginx_http_auth_basic_module.o \
objs/src/http/modules/nginx_http_access_module.o \
objs/src/http/modules/nginx_http_limit_conn_module.o \
objs/src/http/modules/nginx_http_limit_req_module.o \
objs/src/http/modules/nginx_http_geo_module.o \
objs/src/http/modules/nginx_http_map_module.o \
objs/src/http/modules/nginx_http_split_clients_module.o \
objs/src/http/modules/nginx_http_referer_module.o \
objs/src/http/modules/nginx_http_rewrite_module.o \
objs/src/http/modules/nginx_http_proxy_module.o \
objs/src/http/modules/nginx_http_fastcgi_module.o \
objs/src/http/modules/nginx_http_uwsgi_module.o \
objs/src/http/modules/nginx_http_scgi_module.o \
objs/src/http/modules/nginx_http_memcached_module.o \
objs/src/http/modules/nginx_http_empty_gif_module.o \
objs/src/http/modules/nginx_http_browser_module.o \
objs/src/http/modules/nginx_http_upstream_hash_module.o \
objs/src/http/modules/nginx_http_upstream_ip_hash_module.o \
objs/src/http/modules/nginx_http_upstream_least_conn_module.o \
objs/src/http/modules/nginx_http_upstream_random_module.o \
objs/src/http/modules/nginx_http_upstream_keepalive_module.o \
objs/src/http/modules/nginx_http_upstream_zone_module.o \
objs/nginx_modules.o \
-lcrypt -lpcrc -lz \
-Wl,-E
sed -e "s|%%PREFIX%%|usr/local/nginx|" \
    -e "s|%%PID_PATH%%|usr/local/nginx/logs/nginx.pid|" \
    -e "s|%%CONF_PATH%%|usr/local/nginx/conf/nginx.conf|" \
    -e "s|%%ERROR_LOG_PATH%%|usr/local/nginx/logs/error.log|" \
    < man/nginx.8 > objs/nginx.8
make[1]: Leaving directory '/root/nginx/nginx-1.26.2'
root@andrea:~/nginx/nginx-1.26.2#
```

Figure 84. Compiling nginx package

- Finally, we complete the installation with ‘**make install**’ command

UNIVERSIDAD

```
cp objs/nginx '/usr/local/nginx/sbin/nginx'
test -d '/usr/local/nginx/conf' \
    || mkdir -p '/usr/local/nginx/conf'
cp conf/koi-win '/usr/local/nginx/conf'
cp conf/koi-utf '/usr/local/nginx/conf'
cp conf/win-utf '/usr/local/nginx/conf'
test -f '/usr/local/nginx/conf/mime.types' \
    || cp conf/mime.types '/usr/local/nginx/conf'
cp conf/mime.types '/usr/local/nginx/conf/mime.types.default'
test -f '/usr/local/nginx/conf/fastcgi_params' \
    || cp conf/fastcgi_params '/usr/local/nginx/conf'
cp conf/fastcgi_params \
    '/usr/local/nginx/conf/fastcgi_params.default'
test -f '/usr/local/nginx/conf/fastcgi.conf' \
    || cp conf/fastcgi.conf '/usr/local/nginx/conf'
cp conf/fastcgi.conf '/usr/local/nginx/conf/fastcgi.conf.default'
test -f '/usr/local/nginx/conf/uwsgi_params' \
    || cp conf/uwsgi_params '/usr/local/nginx/conf'
cp conf/uwsgi_params \
    '/usr/local/nginx/conf/uwsgi_params.default'
test -f '/usr/local/nginx/conf/scgi_params' \
    || cp conf/scgi_params '/usr/local/nginx/conf'
cp conf/scgi_params \
    '/usr/local/nginx/conf/scgi_params.default'
test -f '/usr/local/nginx/conf/nginx.conf' \
    || cp conf/nginx.conf '/usr/local/nginx/conf/nginx.conf'
cp conf/nginx.conf '/usr/local/nginx/conf/nginx.conf.default'
test -d '/usr/local/nginx/logs' \
    || mkdir -p '/usr/local/nginx/logs'
test -d '/usr/local/nginx/logs' \
    || mkdir -p '/usr/local/nginx/logs'
test -d '/usr/local/nginx/html' \
    || cp -R html '/usr/local/nginx'
test -d '/usr/local/nginx/logs' \
    || mkdir -p '/usr/local/nginx/logs'
make[1]: Leaving directory '/root/nginx/nginx-1.26.2'
root@andrea:~/nginx/nginx-1.26.2# _
```

Figure 85. Installing nginx version 1.26.2

- We start the service with `/usr/local/nginx/sbin/nginx` command then, we navigate to `/usr/local/nginx/html` and open the `index.html` file

```
root@andrea:~# /usr/local/nginx/sbin/nginx
root@andrea:~# nano /usr/local/nginx/html/index.html _
```

Figure 86. Running web service on Nginx

- We edit the `index.html` file to customize our website

UNIVERSIDAD

```
GNU nano 6.0 /usr/local/nginx/html/index.html
<!DOCTYPE html>
<html>
<head>
<title>Welcome to slackware web service!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to our page Jorge and Camila!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p><em>We are using Slackware 15.0!</em></p>
</body>
</html>

root@andrea:~#
```

Figure 87. Customizing Slackware website with nginx

- To start the service when the system boots, we must open **/etc/rc.d/rc.local** and place the command that we entered earlier to start the web service (*Image 18*)

UNIVERSIDAD

```
GNU nano 6.0 /etc/rc.d/rc.local
#!/bin/bash
#
# /etc/rc.d/rc.local: Local system initialization script.
#
# Put any local startup commands in here. Also, if you have
# anything that needs to be run at shutdown time you can
# make an /etc/rc.d/rc.local_shutdown script and put those
# commands in there.
#ifconfig eth1 10.2.77.193 netmask 255.255.0.0
#route add default gw 10.2.65.1
named
/usr/sbin/named
/etc/rc.d/rc.ntpd start
/etc/rc.d/rc.postgresql start
/usr/local/nginx/sbin/nginx
```

Figure 88. Configuring web service when the slackware boots

- We open the browser and look for the IP address of the Slackware machine

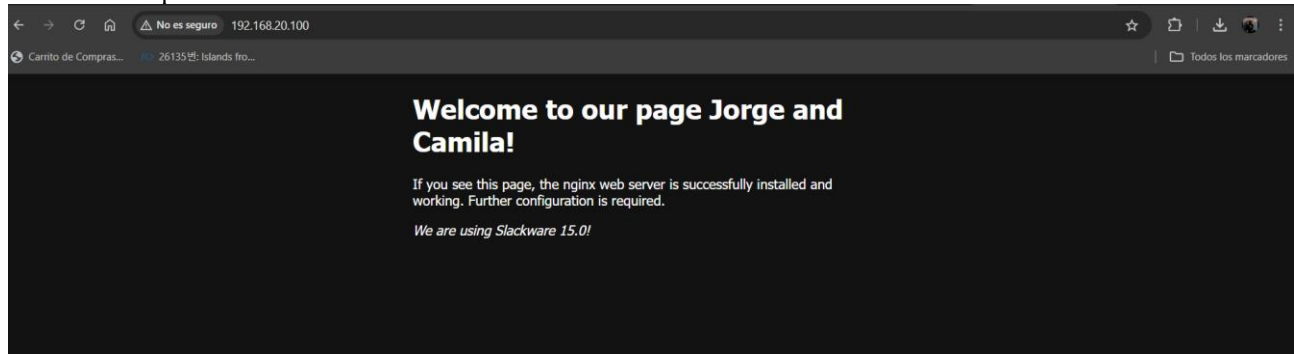


Figure 89. Verifying that Slackware web server is working

- To configure the DNS so that the page is accessible by the domain name, we edit the zone file that we set up in Lab 3 (torres.org.uk.hosts) and assign the corresponding IP address of the web server to www

```

UNIVERSIDAD
GNU nano 6.0 /etc/DNS/torres.org.uk.hosts Modified
;
; zone torres.org.uk configuration

$ IN SOA ns1.torres.org.uk. root.torres.org.uk. (
2020050101 ; serial
43200 ; refresh
3600 ; retry
432000 ; expire
86400 ; minimum time-to-live
)

torres.org.uk. IN NS ns1.torres.org.uk.
ns1 IN A 10.2.77.193
ns2 IN A 10.2.77.194

; Ipv4
www IN A 10.2.77.193
mail IN A 10.2.77.200
servicios IN A 10.2.77.201
windows IN A 10.2.77.196

; Ipv6
juegos IN AAAA ::ffff:0a02:4dcb
reco IN AAAA ::ffff:0a02:4dca

; alias
correoX IN CNAME mail.torres.org.uk.
onix IN CNAME servicios.torres.org.uk.
sol IN CNAME reco.torres.org.uk.

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location ^M-U Undo
^X Exit ^R Read File ^_ Replace ^U Paste ^J Justify ^_ Go To Line ^M-E Redo
  
```

Figure 90. Configuring zone file torres.org.uk.hosts to resolve the domain name of the web server (www.torres.org.uk)

- We open the browser and enter the domain name of Slackware

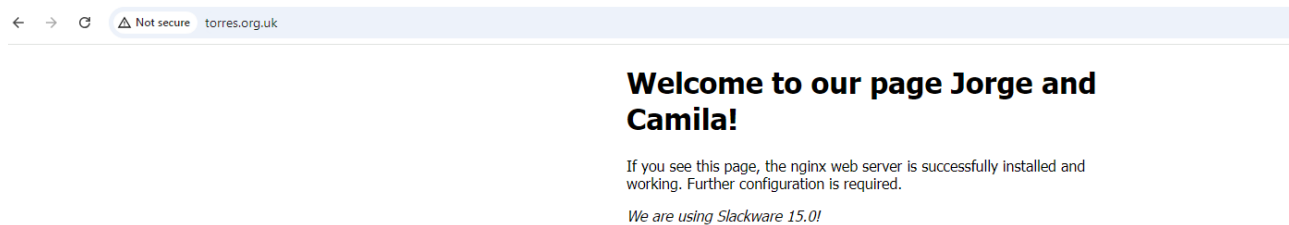


Figure 91. Accessing the Slackware web server by domain name

UNIVERSIDAD

1.3. Windows Server

- We open 'Server Manager', in tools tab click on 'Add Servers'

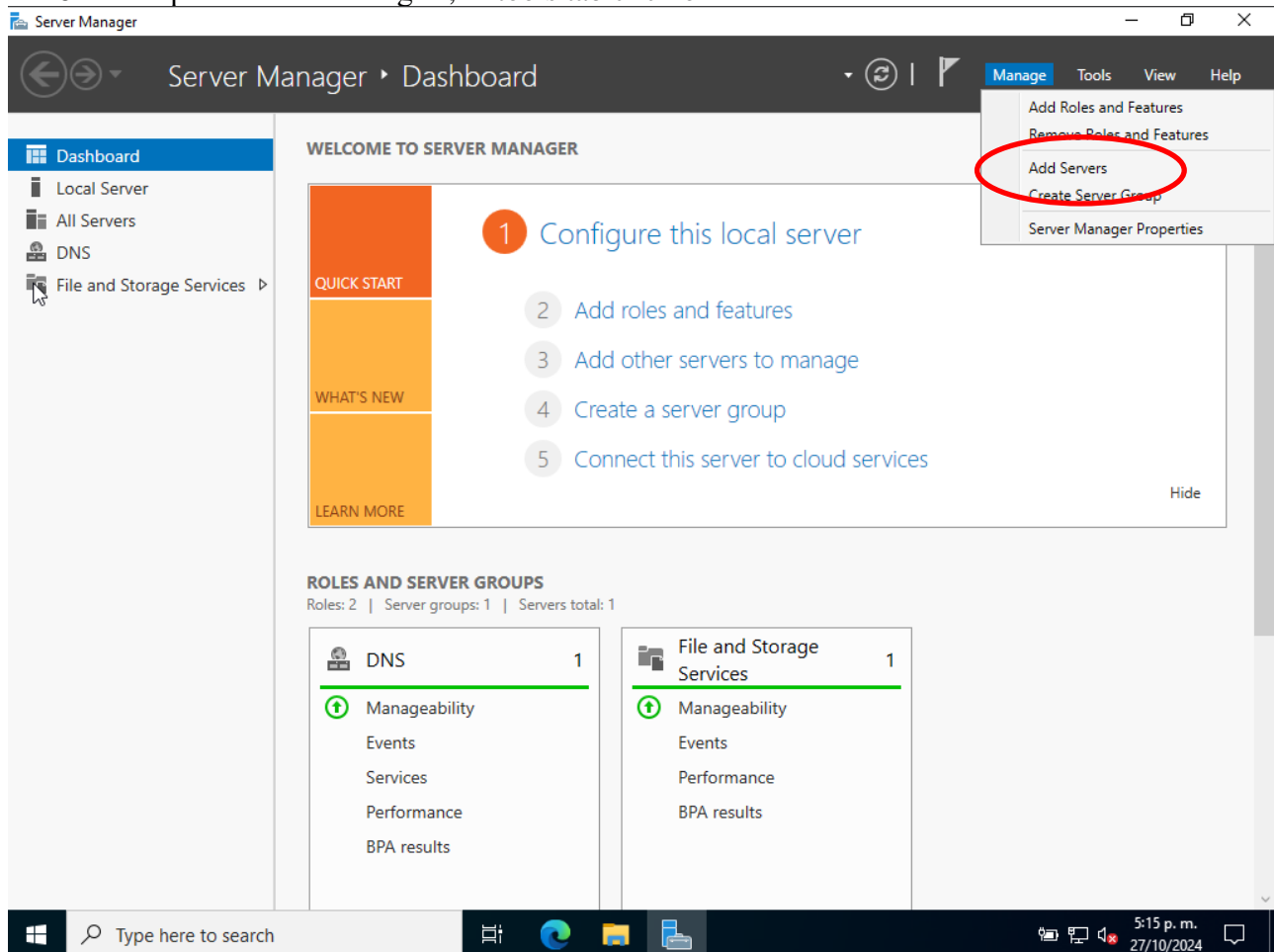


Figure 92. Opening server configuration

- In 'Server Selection' we select the current device then, we select 'Next'

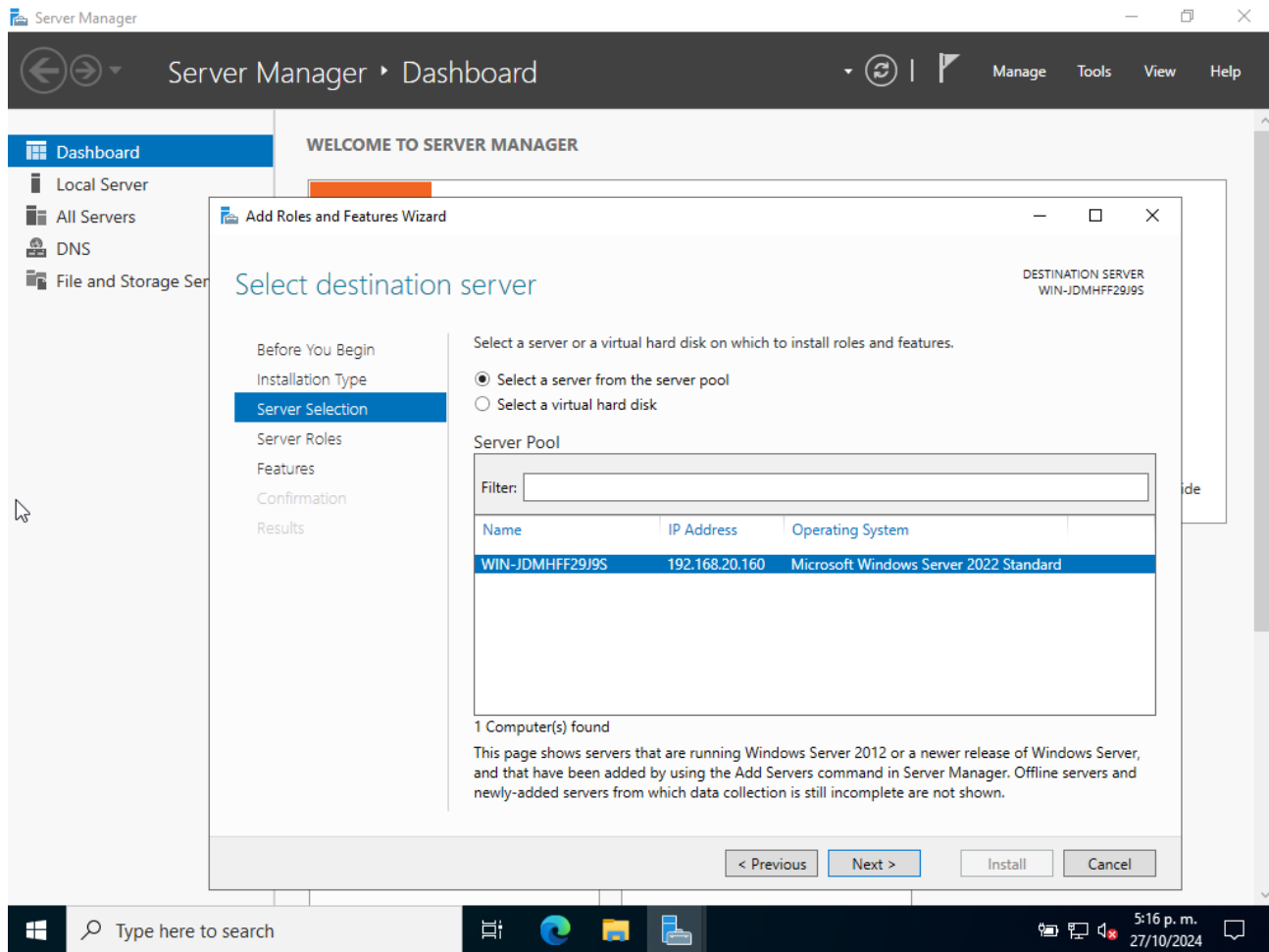


Figure 93. Selecting operating system

- Then we go to 'Server Roles' and select where it says 'Web Service (IIS)' then click Next

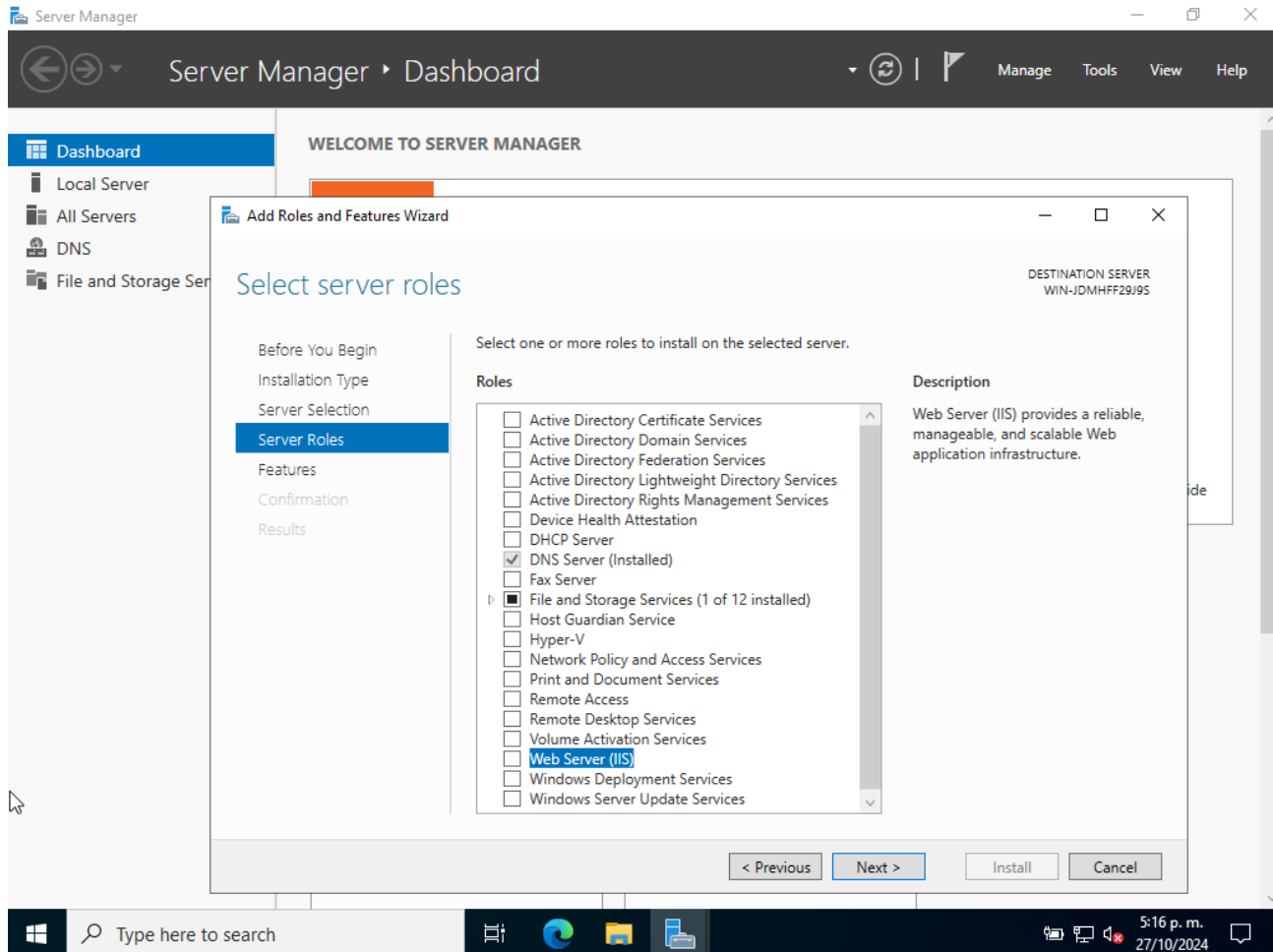


Figure 94. Selecting web server for Windows Server

- A window will open to add Roles and Features, click on 'Add Features'

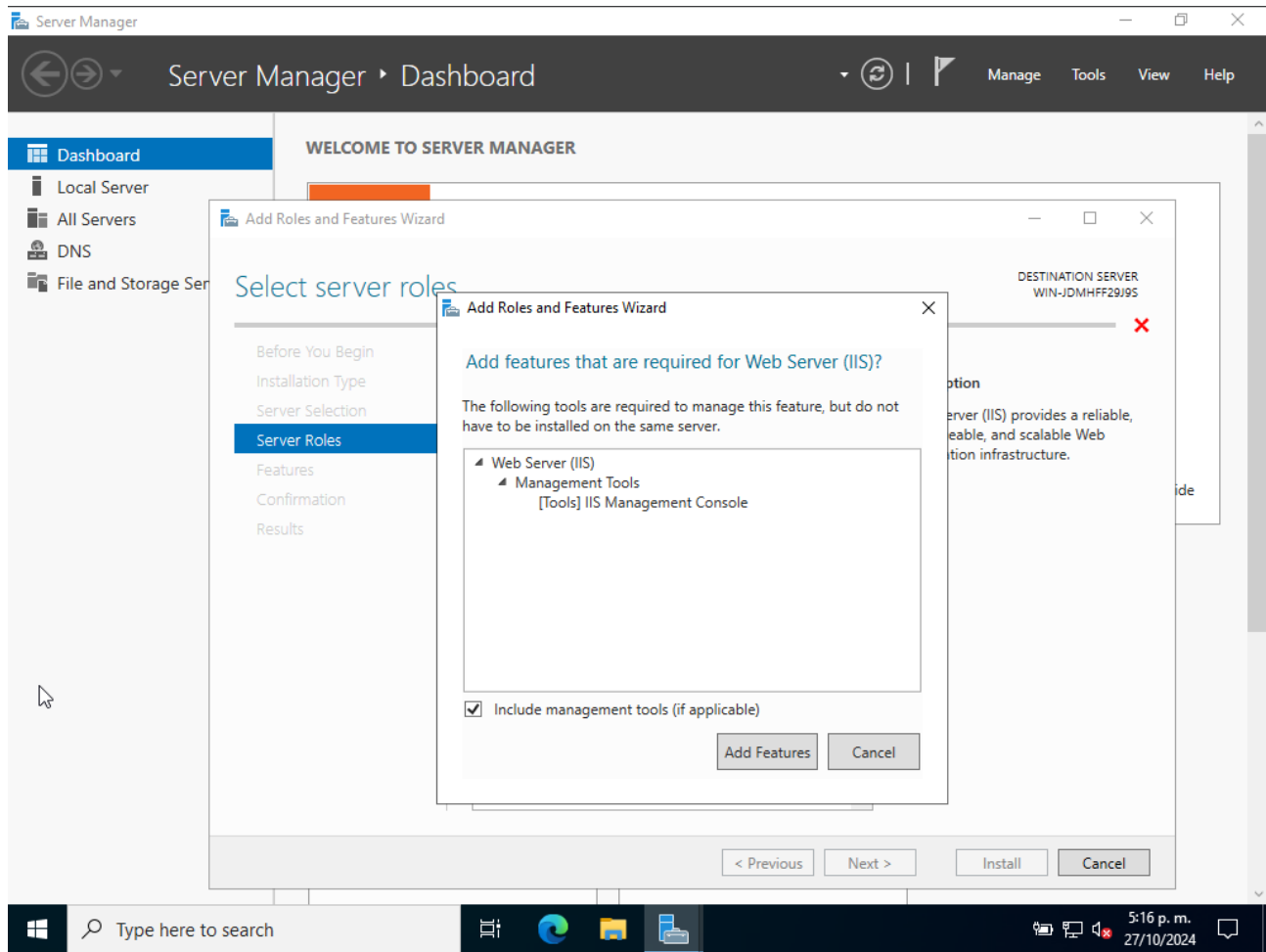


Figure 95. Adding Features

- We select all the Role Services and click on Next

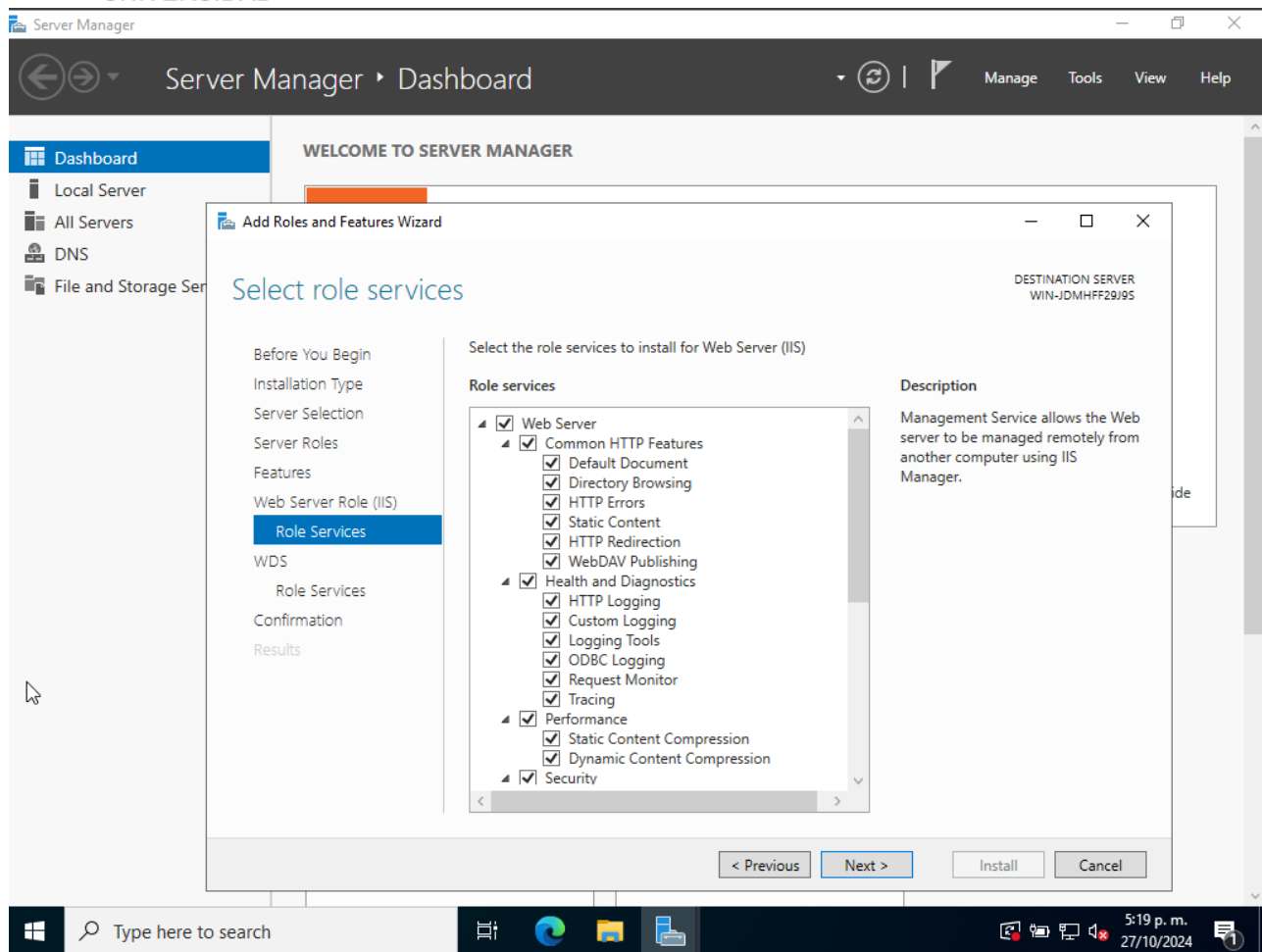


Figure 96. Selecting Role Services

- We review that everything is correct and proceed to start the installation

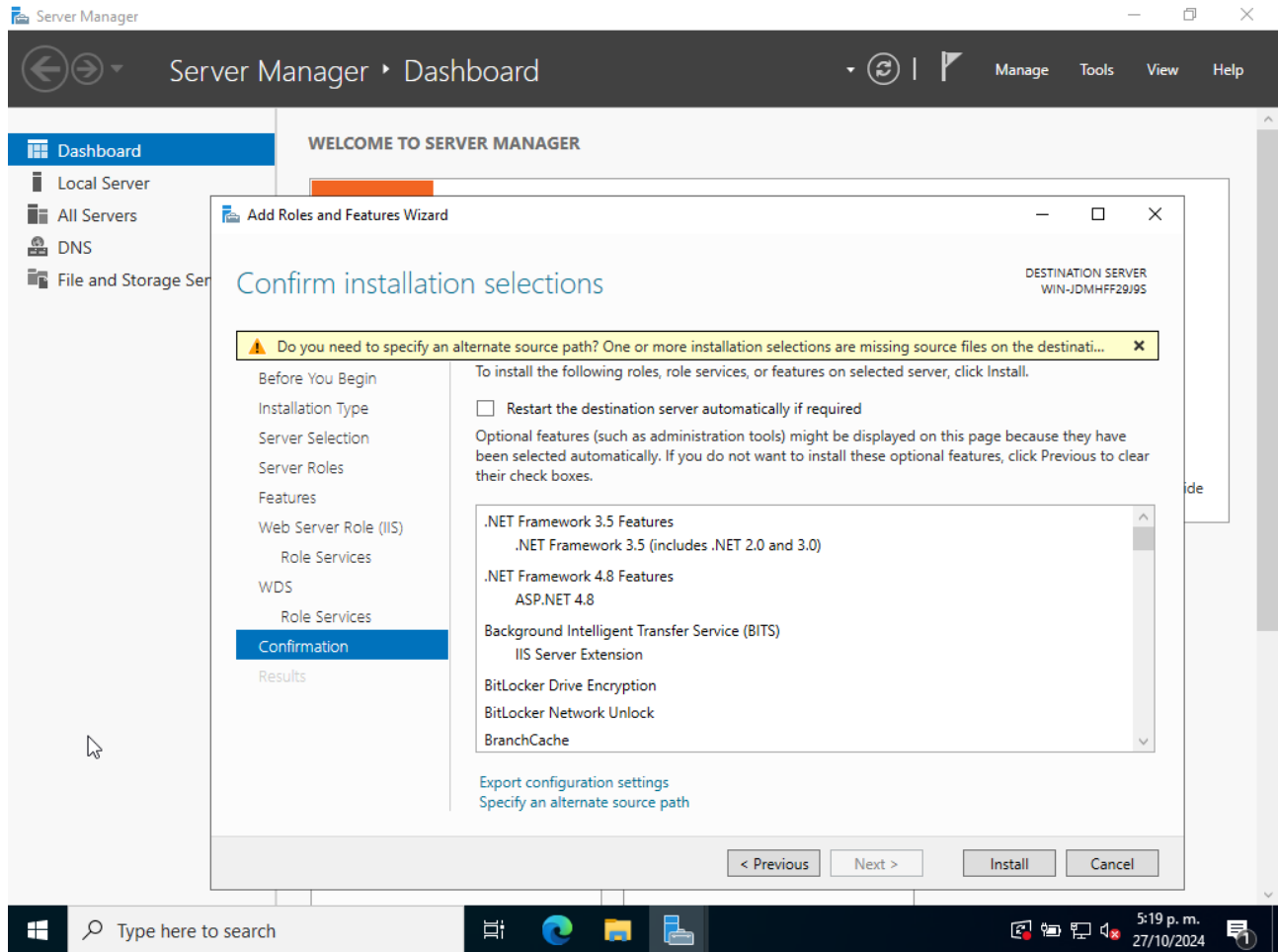


Figure 97. Starting the web server installation on Windows

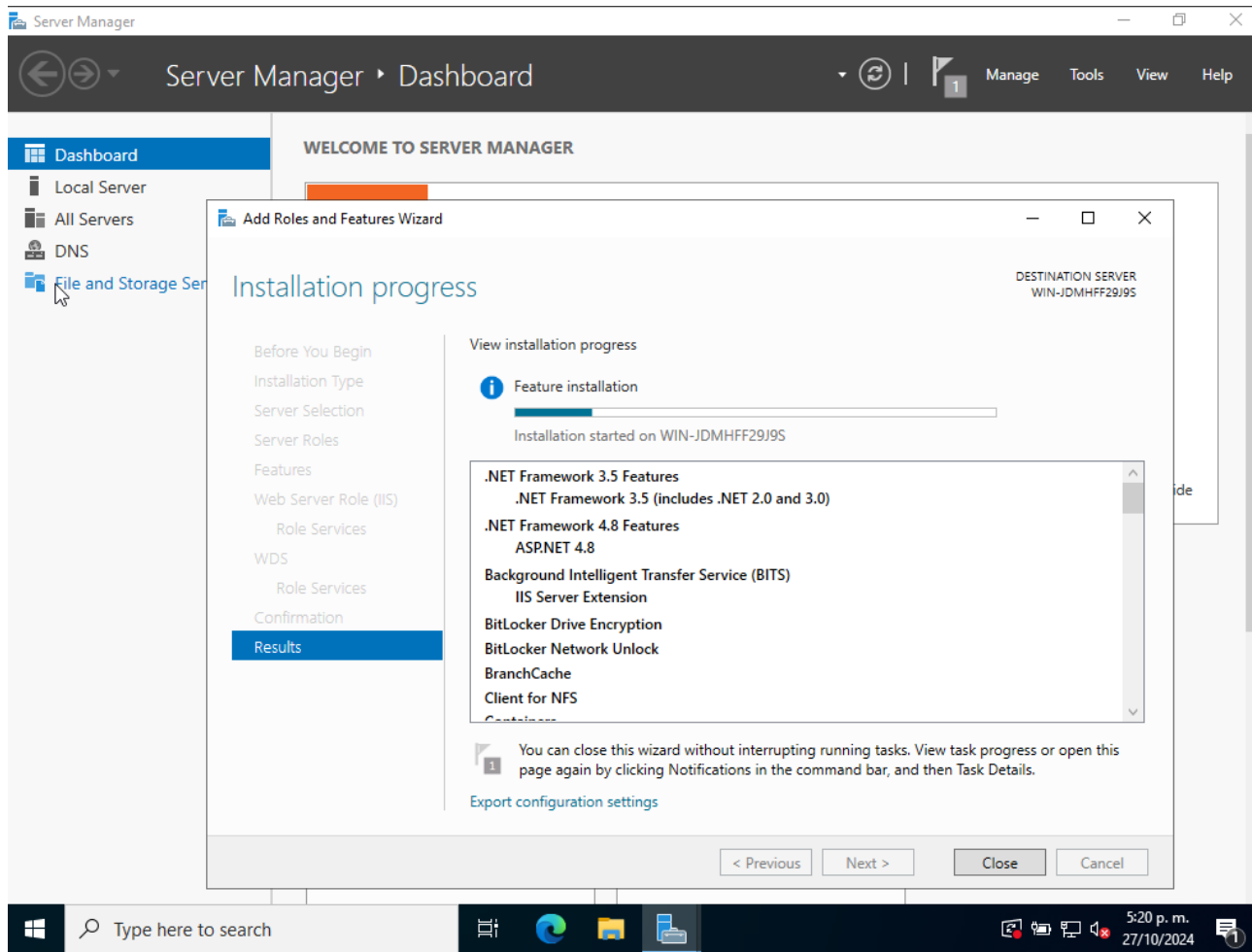


Figure 98. Installing web service in Windows

- When the installation is complete, we open **wwwroot** directory where the web pages are stored

UNIVERSIDAD

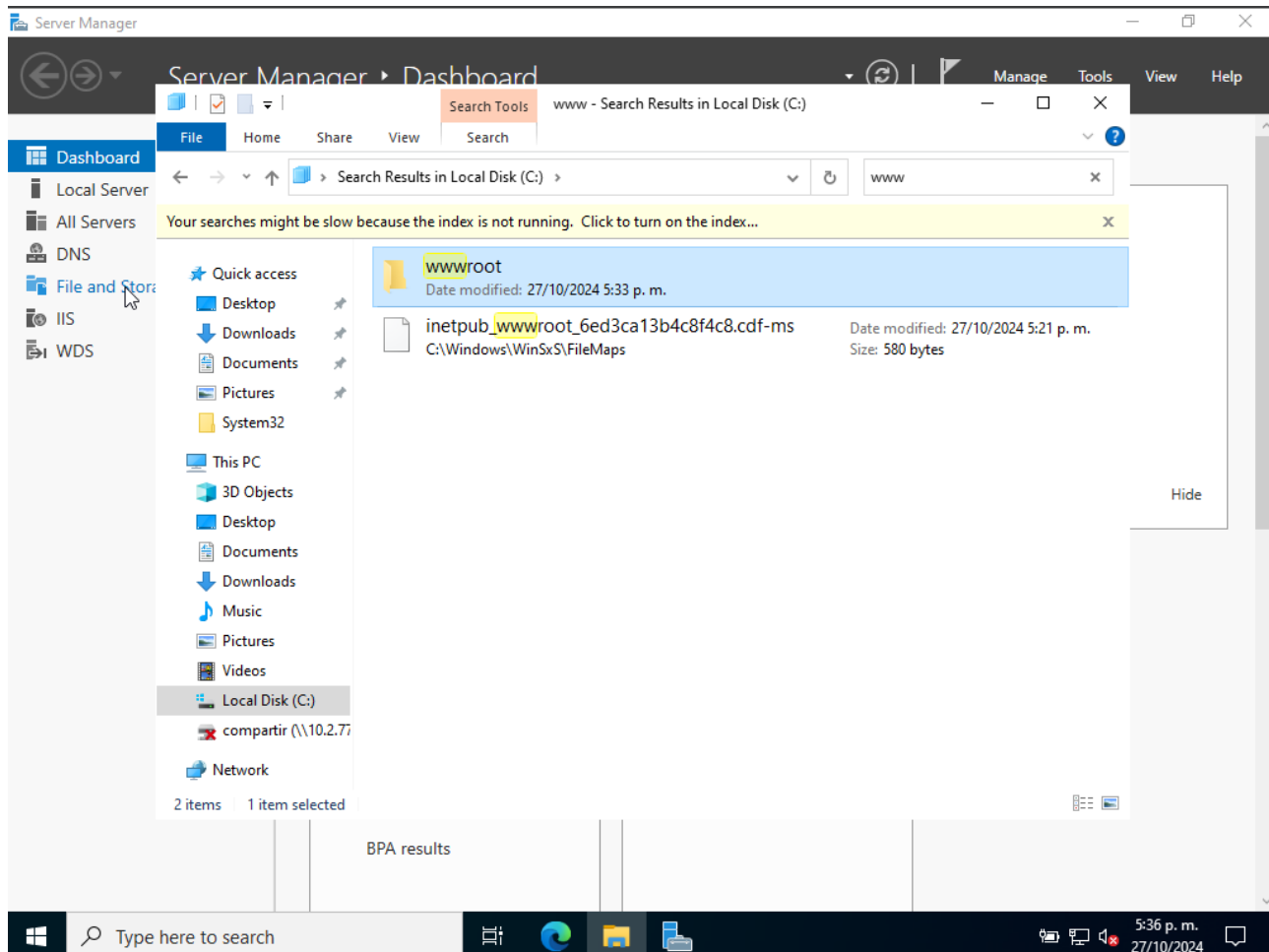


Figure 99. Opening wwwroot directory to store websites

- We create an index.html into this directory

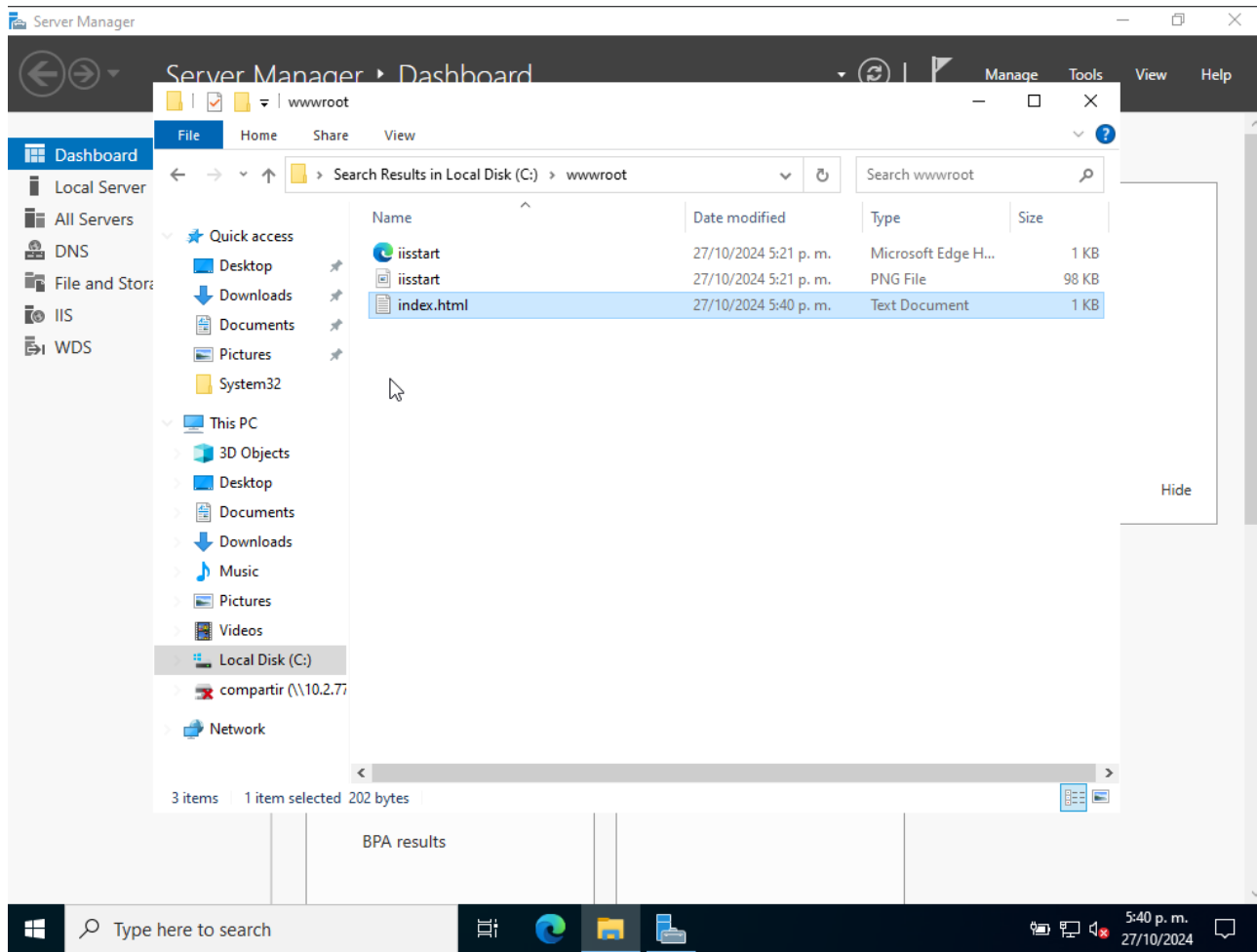


Figure 100. Creating index.html into wwwroot

- We add HTML tags to customize our website

UNIVERSIDAD

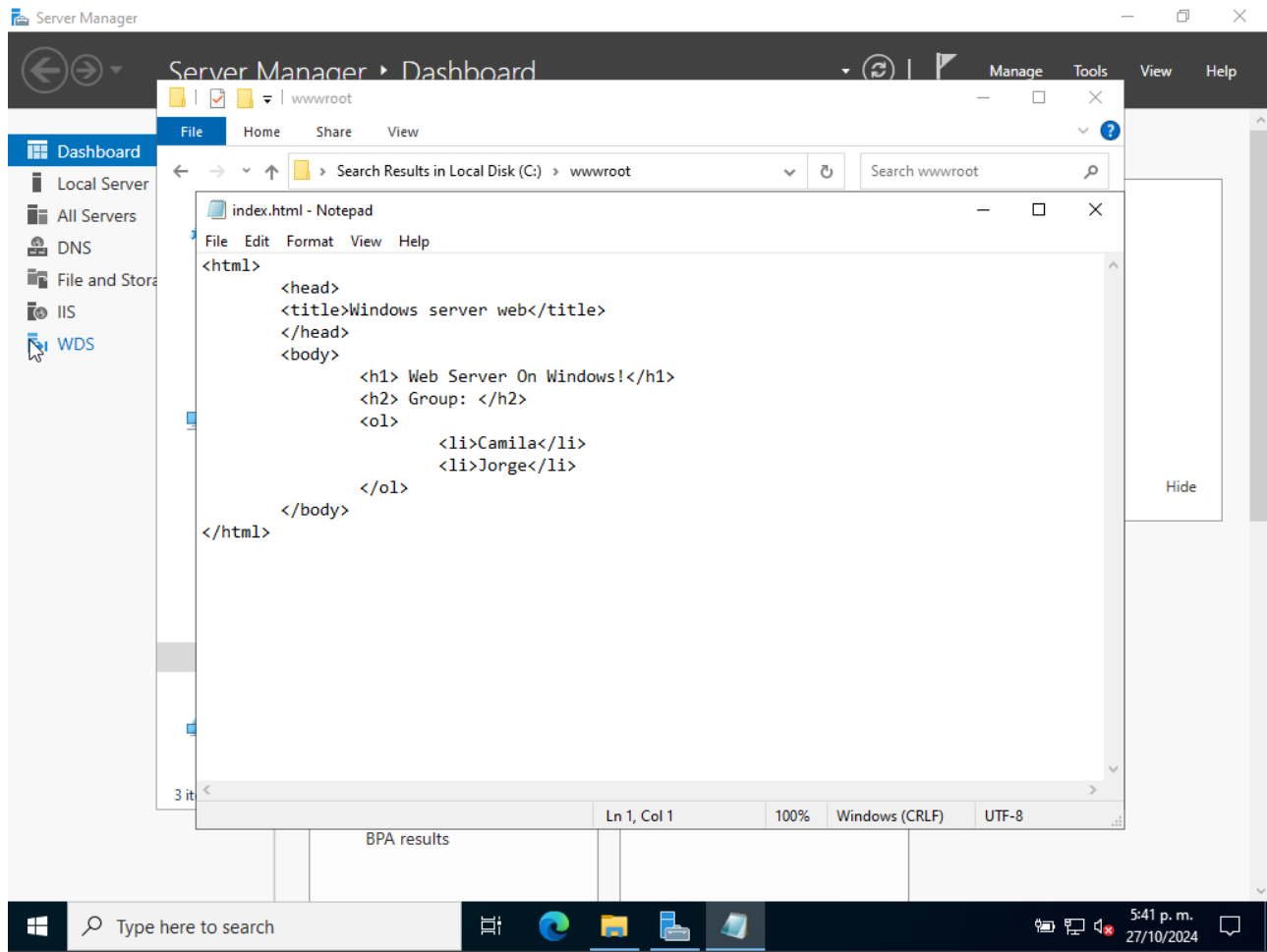


Figure 101. Customizing index.html file in windows web server

- We save the file and ensure that the extension of the file is correct (.html)

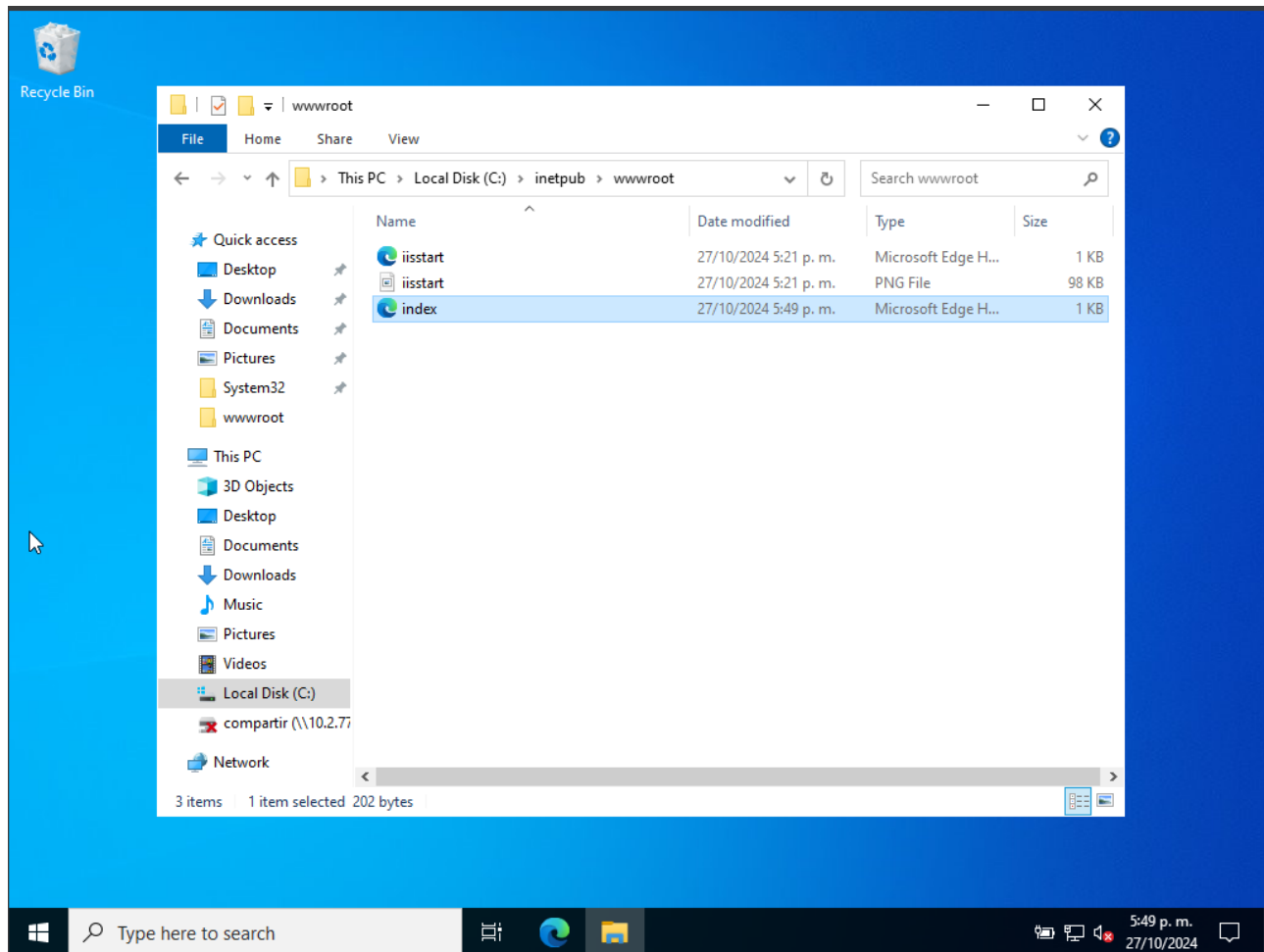


Figure 102. Saving index.html file into Windows Web Service

- To ensure that the server starts when the system boots, we press Windows + R and type 'services.msc'

UNIVERSIDAD

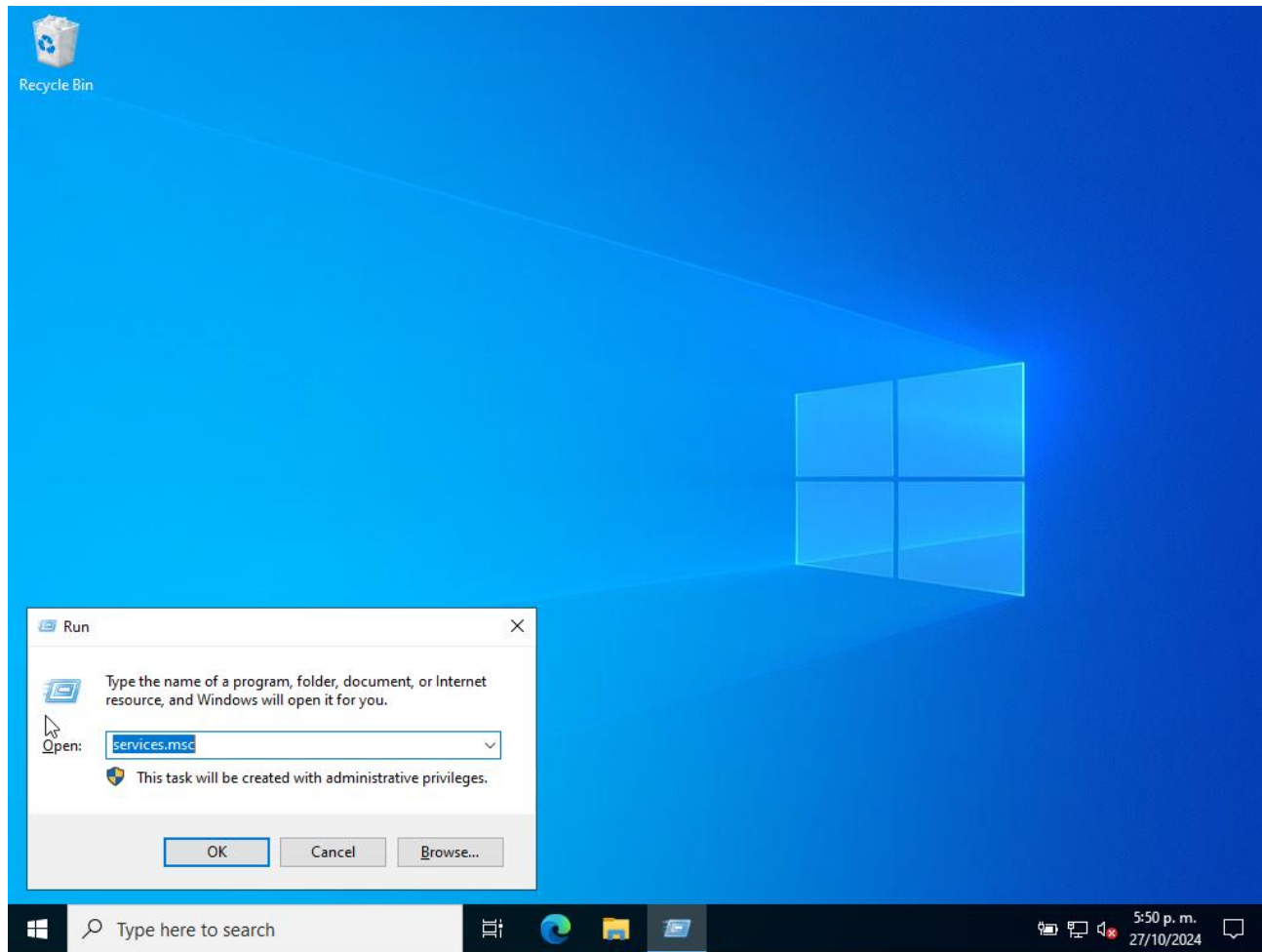


Figure 103. Opening Windows services configuration

- We search 'World Wide Web Publishing Services' and right click on it and select 'Properties'

UNIVERSIDAD

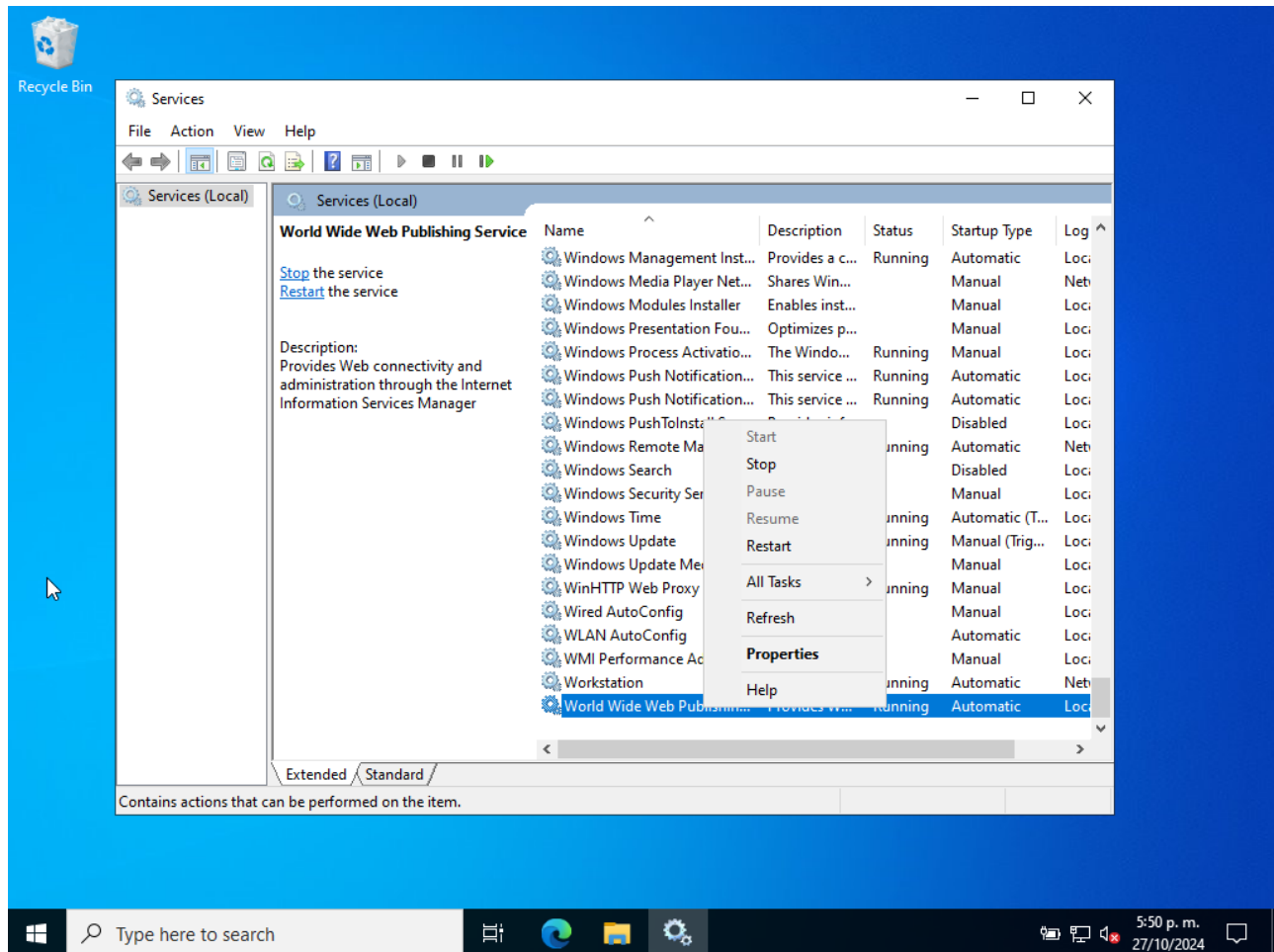


Figure 104. Searching web service in Windows services configuration

- In 'Startup Type' section we select 'Automatic'

UNIVERSIDAD

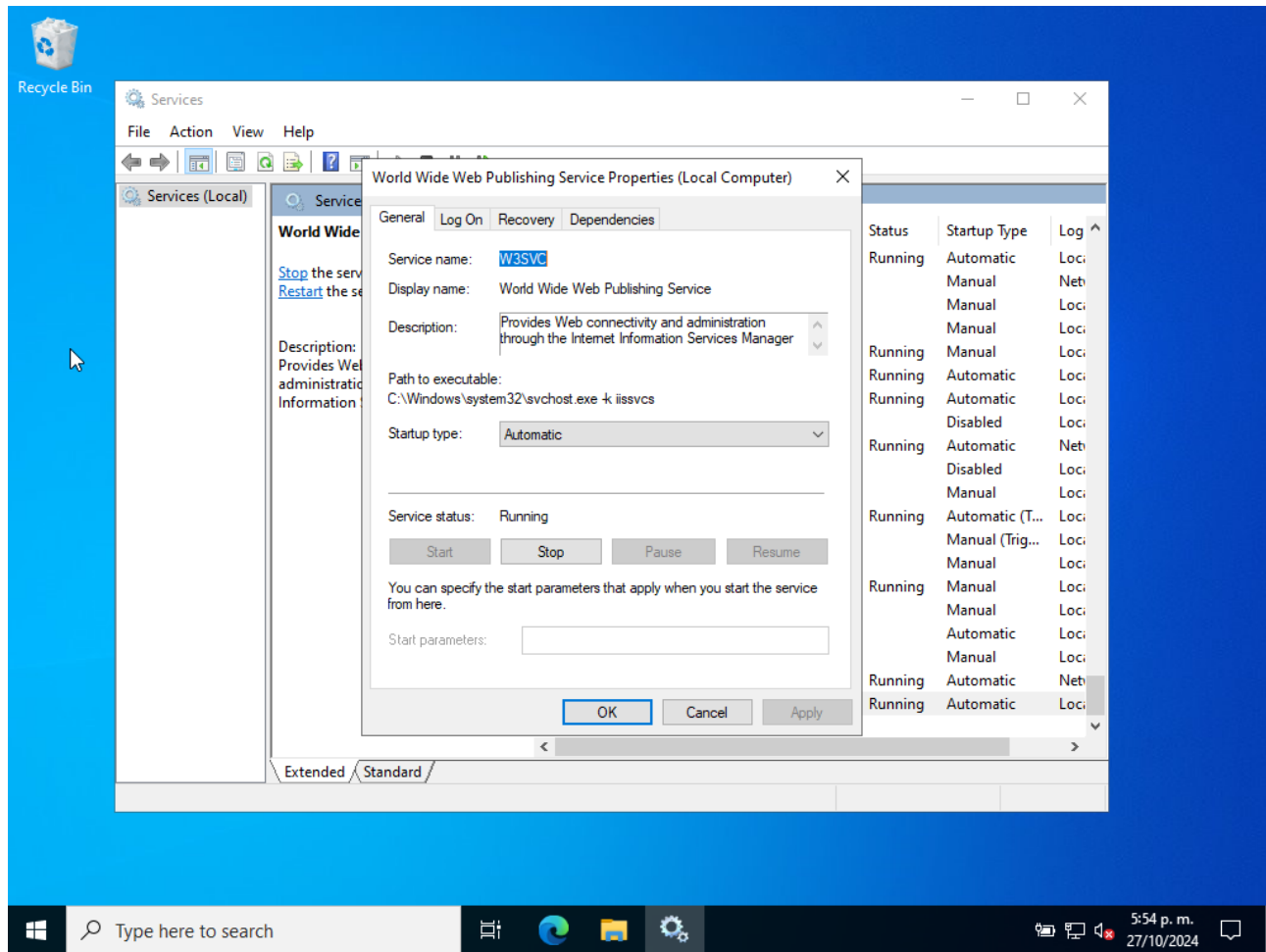
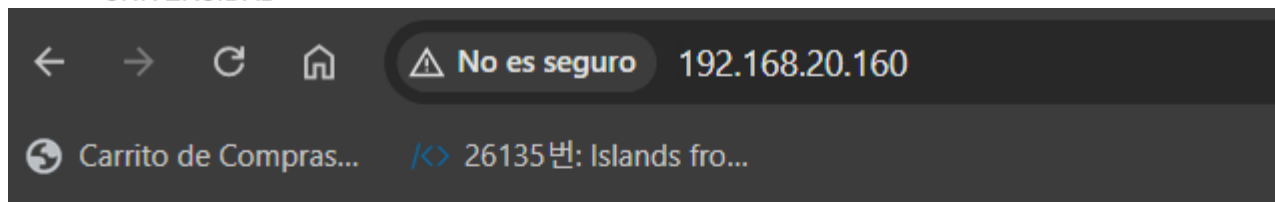


Figure 105. Configuring web server to start when Windows boots

- We open the browser and look for the IP address of the Windows Server machine



Web Server On Windows!

Group:

1. Camila
2. Jorge

Figure 106, Verifying that Windows web server is working

- Remember that Windows is a secondary DNS server for Slackware and Solaris, so to search for the web server by domain name, we configure the DNS zone file on Solaris and Slackware and synchronize it with the secondary DNS on Windows. This way, querying **windows.[domain]** will allow us to access the web server

Modificado

```

;
; /etc/DNS/named.soa file
;
; name server SOA file
;
@ IN SOA ns1.gamboa.com.it. root.gamboa.com.it. (
2020050101 ; serial
43200 ; refresh
3600 ; retry
432000 ; expire
86400 ; minimum time-to-live
)

gamboa.com.it. IN NS ns1.gamboa.com.it.
ns1             IN      A      10.2.77.194
ns2             IN      A      10.2.77.193

www             IN      A      10.2.77.194
mail            IN      A      10.2.77.200
servicios       IN      A      10.2.77.201
windows IN      A      10.2.77.196
juegos          IN      AAAA    ::ffff:0a02:4dcb

```

Figure 107. Configuring gamboa.com.it to access the Windows web server

UNIVERSIDAD

```

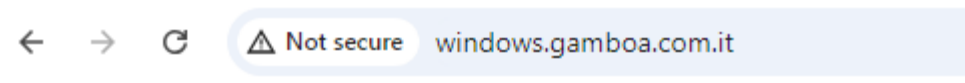
GNU nano 6.0 /etc/DNS/torres.org.uk.hosts Modified
;
; zone torres.org.uk configuration
;
$TTL 2020050101 ; serial
43200 ; refresh
3600 ; retry
432000 ; expire
86400 ; minimum time-to-live
)

torres.org.uk. IN NS ns1.torres.org.uk.
ns1 IN A 10.2.77.193
ns2 IN A 10.2.77.194

; Ipv4
www IN A 10.2.77.193
mail IN A 10.2.77.200
servicios IN A 10.2.77.201
windows IN A 10.2.77.196
; Ipv6
juegos IN AAAA ::ffff:0a02:4dcb
  
```

Figure 108. Configuring torres.org.uk to access the Windows web server

- We open the browser and enter the domain name of Windows (windows.gamboa.com.it Or windows.torres.org.uk)



Web Server On Windows!

Group:

1. Camila
2. Jorge

Figure 109. Accessing the Windows web server by domain name

2. Hosting Service Configuration

- We open the Apache httpd configuration file /etc/apache2/2.4/httpd.conf

```
root@solaris:~# nano /etc/apache2/2.4/httpd.conf
```

Figure 110. Opening httpd.conf file

- At the end of the file, we place the <VirtualHost></VirtualHost> tags for each virtual host service. Inside these tags, we specify the port (which is 80 for a web server), the directory where the

UNIVERSIDAD

index.html file will be stored, and the domain that will be used to access it in the browser (network.camila.com.co and security.jorge.org.jp)

```

        DocumentRoot "/var/apache2/2.4/htdocs"
</VirtualHost>
#Virtual Host for network.camila.com.co
<VirtualHost *:80>
    ServerName network.camila.com.co
    DocumentRoot "/var/apache2/2.4/htdocs/network"
    <Directory "/var/apache2/2.4/htdocs/network">
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
#Virtual Host for security.jorge.org.jp
<VirtualHost *:80>
    ServerName security.jorge.org.jp
    DocumentRoot "/var/apache2/2.4/htdocs/security"
    <Directory "/var/apache2/2.4/htdocs/security">
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>

```

 Ver ayuda
  Guardar
  Reemplazar
  Pegar txt
  Ortografía
  Ir a línea
  Salir
  Leer fich.

Figure 111. Configuring Virtual Host service

- We create the directories specified in the Virtual Host tags (Figure 43)

```

root@solaris:~# mkdir -p /var/apache2/2.4/htdocs/network
root@solaris:~# mkdir -p /var/apache2/2.4/htdocs/security

```

Figure 112. Creating Virtual Host directories

- Inside these directories, we create an index.html file, open it using 'nano /var/apache2/2.4/htdocs/[domain]', and customize it for each host

UNIVERSIDAD

```

Modificado

<html>
<body>
<h1> Welcome to Camila Page!!</h1>
<h2> Virtual Host service </h2>
</body>
</html>

```

[Nuevo fichero]

^G Ver ayuda ^O Guardar ^W Bu
 ^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^I OrtografÃ-a ^_ Ir a lÃ-nea

Figure 113. Customizing network.camila.com.co page

```

Modificado

<html>
<body>
<h1>Welcome to Jorge Page!!!</h1>
<h2> Virtual Host Service </h2>
</body>
</html>

```

[Nuevo fichero]

^G Ver ayuda ^O Guardar ^W Bu
 ^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^I OrtografÃ-a ^_ Ir a lÃ-nea

Figure 114. Customizing security.jorge.org.jp page

UNIVERSIDAD

- To access each host by its domain name, we configure the **/etc/named.conf** file on the DNS server and add the corresponding domains (camila.com.co and jorge.org.jp), along with their configuration files for that zone

Modificado

```

};
    allow-transfer { 192.168.20.100; 192.168.20.152; };
};
zone "torres.org.uk" {
    type slave;
    file "/etc/bind/torres.org.uk.hosts";
    masters { 192.168.20.100; };
};
zone "camila.com.co" {
    type master;
    file "/etc/bind/camila.com.co.hosts";
}
zone "jorge.org.jp" {
    type master;
    file "/etc/bind/jorge.org.jp.hosts";
}

```

^G Ver ayuda ^O Guardar ^W B
^X Salir ^R Leer fich. ^N Reemplazar ^U Pegar txt ^T Ortografía ^_ Ir a línea

Figure 115. Adding new zones in **/etc/named.conf** zones file

- Then, we open each zone file located in **/etc/bind/[domain].hosts** and specify the IP of the web server (Solaris Machine) along with its domain name: security for jorge.org.jp and network for camila.com.co

UNIVERSIDAD

```
Modificado

@ IN SOA ns1.jorge.org.jp. root.jorge.org.jp. {
2020050101 ; serial
43200 ; refresh
3600 ; retry
432000 ; expire
86400 ; minimum time-to-live
)

jorge.org.jp IN NS ns1.jorge.org.jp.
ns1 IN A 192.168.20.101

security IN A 192.168.20.101
```

Figure 116. Configuring zone file *Jorge.org.jp*

```
@ IN SOA ns1.camila.com.co. root.camila.com.co. (
2020050101 ; serial
43200 ; refresh
3600 ; retry
432000 ; expire
86400 ; minimum-time-to-live
)
camila.com.co. IN NS ns1.camila.com.co.
ns1 IN A 192.168.20.101
network IN A 192.168.20.101
```

Figure 117. Configuring zone file *camila.com.co*

UNIVERSIDAD

- We restart the DNS service using the command **svcadm enable network/dns/server**. Then, we verify that the DNS service is responding to the configured domain names using the **nslookup** command

```
root@solaris:/etc/bind# nslookup security.jorge.org.jp
Server:          192.168.20.101
Address:         192.168.20.101#53

Name:   security.jorge.org.jp
Address: 192.168.20.101
```

Figure 118. Verifying the domain jorge.org.jp is responding

```
root@solaris:/etc/bind# nslookup network.camila.com.co
Server:          192.168.20.101
Address:         192.168.20.101#53

Name:   network.camila.com.co
Address: 192.168.20.101

root@solaris:/etc/bind# nano camila.com.co.hosts
```

Figure 119. Verifying the domain camila.com.co is responding

- To test the service on our client computer, we open the network settings and set the DNS server of Solaris

UNIVERSIDAD

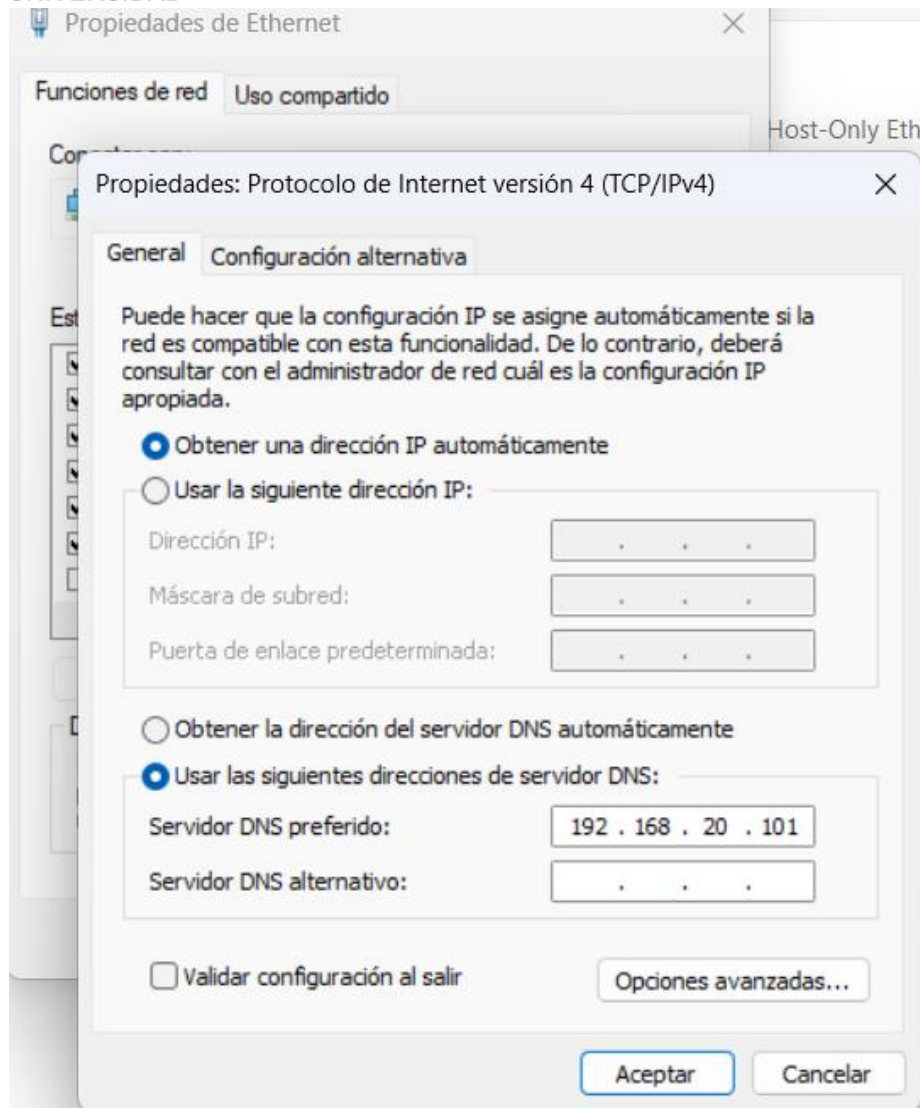
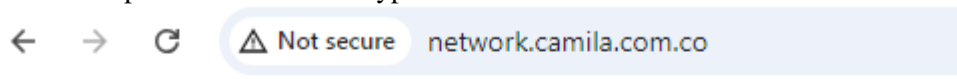


Figure 120. Configuring DNS server on the client machine

- We open the browser and type the domain names of the hosts



Welcome to Camila Page!!

Virtual Host service

Figure 121. Verifying that the Virtual Host of network.camila.com.co is working

Welcome to Jorge Page!!!

Virtual Host Service

Figure 122. Verifying that the Virtual Host of security.jorge.org.jp is working

Conclusions

Router Configuration: We configured Cisco 1941 routers by assigning names, setting privileged mode passwords, and configuring interfaces. Using console connections and setting up logging provided us with practical skills in managing and securing network devices, crucial for network administration.

Static Routing Implementation: Configuring static routes on the routers ensured that all devices across different subnets could communicate. This task highlighted the importance of routing in network design and taught us how to manually set up and troubleshoot network paths, which is essential for maintaining network connectivity.

Web Server Installation Across Different Platforms: Installing Apache on Solaris and Nginx on Slackware allowed us to compare different server environments. Setting up basic webpages and ensuring servers started automatically on these platforms enhanced our ability to deploy and manage web services across diverse operating systems.

Virtual Hosting Configuration: On the Solaris machine, we configured virtual hosts to manage multiple domains on a single server. We set up DNS entries for network.camila.com.co and security.jorge.org.jp. This exercise taught us how to efficiently use server resources to host multiple websites, a common requirement in real-world IT environments.


DNS Service Implementation: By configuring DNS services to resolve domain names to the appropriate IP addresses, we ensured that each virtual host could be accessed by name. This task was crucial in understanding how DNS integrates with web hosting to provide seamless access to websites.

Using Cisco Packet Tracer for Network Simulation: We created a network in Cisco Packet Tracer, set up PCs and routers, and verified connectivity using ping and traceroute. This simulation experience was invaluable for visualizing and understanding network behavior, helping us predict and solve networking issues effectively.

References

- *Cisco 1941 series integrated services routers data sheet.* (2017, August 22). Cisco. https://www.cisco.com/c/en/us/products/collateral/routers/1900-series-integrated-services-routers-isr/data_sheet_c78_556319.html
- *Cisco packet tracer multi user connection.* (n.d.). Rssing.com. Retrieved November 4, 2024, from <https://examtut1.rssing.com/chan-63784374/article21.html>

UNIVERSIDAD

- Documentation Group. (n.d.). *Welcome! - the Apache HTTP server project*. Apache.org. Retrieved November 4, 2024, from <https://httpd.apache.org/>
- *Infraestructura de red con Red Hat y sus partners*. (n.d.). Redhat.com. Retrieved November 4, 2024, from <https://www.redhat.com/es/partners/network-infrastructure>
- *nginx*. (n.d.). Nginx.org. Retrieved November 4, 2024, from <https://nginx.org/en/>
- *Physical components*. (2015, March 20). Cisco. https://www.cisco.com/c/en/us/td/docs/net_mgmt/prime/network/3-8/reference/guide/phcmp.html
- Slack. (n.d.). *Virtual hosts: qué son y cómo funcionan*. Slack. Retrieved November 4, 2024, from <https://slack.com/intl/es-es/blog/transformation/virtual-hosts-que-son-y-como-funcionan>
- *Windows Server 2022*. (n.d.). Microsoft.com. Retrieved November 4, 2024, from <https://www.microsoft.com/es-co/windows-server>
- Worldamb [@worldamb_]. (n.d.).  *tracert ¿COMO UTILIZARLO! ¿QUE ES? ¿PARA QUE SIRVE? Traceroute*. Youtube. Retrieved November 4, 2024, from https://www.youtube.com/watch?v=Y5MBbo_UPdM