

**(CT106-3-2-SNA)**

**SYSTEM AND NETWORK ADMINISTRATION**

**GROUP ASSIGNMENT**

**INTAKE CODE : APU2F2502CS(CYB)**

**LECTURER : Dr. Manimegalai Rajenderan**

**HAND OUT DATE : WEEK 2 (9 March 2025)**

**SUBMISSION DATE : WEEK 12 (27 May 2025)**

**GROUP : Lab 11 - Group 15**

|  |  |
| --- | --- |
| **Student ID** | **Student Name** |
| TP080121 | Yaw Kai Yuan |
| TP080661 | Gan Kai Hong |
| TP080254 | Er Chen Liang |
| TP080719 | Ting Jac Sheinn |
| TP080388 | Chong Wai Kit |

Table of Contents

[1.0 Introduction 4](#_Toc199158645)

[2.0 Network Configuration 5](#_Toc199158646)

[2.1 Connection between Server and Client 5](#_Toc199158647)

[2.2 Set up Hostname on server and client using FQDN 9](#_Toc199158648)

[2.3 Static IP Configuration on Rocky 13](#_Toc199158649)

[3.0 Network File System (NFS) Configuration 22](#_Toc199158650)

[3.1 NFS Installation of Rocky 22](#_Toc199158651)

[3.2 NFS Installation of Ubuntu 25](#_Toc199158652)

[3.3 NFS Configuration 27](#_Toc199158653)

[3.4 Testing NFS 34](#_Toc199158654)

[3.5 Permanent Mount Points Configuration 40](#_Toc199158655)

[4.0 Apache Web Server Configuration 42](#_Toc199158656)

[4.1 Apache Web Server Installation 42](#_Toc199158657)

[4.2 Apache Web Server Configuration 44](#_Toc199158658)

[4.3 Access the Website in Ubuntu Client 50](#_Toc199158659)

[5.0 Configure a Certificate Authority (CA) 53](#_Toc199158660)

[5.1 OpenSSL Installation 53](#_Toc199158661)

[5.2 Preparing the Public Key Infrastructure Directory 54](#_Toc199158662)

[5.3 Create a Certificate Authority 55](#_Toc199158663)

[5.4 Create a Virtual Host 61](#_Toc199158664)

[5.5 Distribute the Public Certificate of the CA 63](#_Toc199158665)

[5.6 Testing HTTPS between Client and Server 67](#_Toc199158666)

[5.7 Postfix and Dovecot Configuration 74](#_Toc199158667)

[5.8 Testing secure email between Client and Server 79](#_Toc199158668)

[6.0 Additional Service 86](#_Toc199158669)

[6.1 Samba Installation 86](#_Toc199158670)

[6.2 Samba Configuration 88](#_Toc199158671)

[6.3 Test on Ubuntu Client 97](#_Toc199158672)

[6.4 Permanent Mount Point Configuration 101](#_Toc199158673)

[7.0 Conclusion 105](#_Toc199158674)

[8.0 Workload Matrix 106](#_Toc199158675)

[9.0 References 107](#_Toc199158676)

# 1.0 Introduction

In today’s era, client systems play an important role in maintaining the network security and scalability. Our assignment this time mainly focused on the installation, configuration and management of basic services on Rocky server and Ubuntu client. The content of these two devices includes network configuration, file sharing using NFS and Samba, and secure communication between the Rocky server and Ubuntu Client devices using Apache and SSL/TLS.

In this assignment, we took a systematic and practical approach to create a functional environment which can simulate a real system and network management tasks. As a result, we tested each configuration between the two devices to ensure the connectivity, performance, and security of the devices, which can show the understanding of us to the basic system management tasks.

# 2.0 Network Configuration

## 2.1 Connection between Server and Client

First to Start the configuration of the network between the Rocky server and Ubuntu Client, open the Tools menu and search for the Network Manager.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.1.1 Virtual Box Setup

Next, click on the Create button and go to the NAT Networks and type in the Networks name and IPv4 Prefix. Here we use the IPv4 prefix 192.168.200.0/24. After that, click on the enable DHCP and make sure it is checked.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.1.2 Virtual Box Networks Set Up

After completed the creation of the NAT Networks, go to the setting of the Rocky Server and check the Network is using the NAT Networks which just created.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.1.3 Rocky Server Network Setup

Then, make sure that the Ubuntu Client is also using the same NAT Networks.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.1.4 Ubuntu Client Network Setup

Afterward, start the Rocky Server and type this command in the terminal:

* Ifconfig

This command can show the system networks interface which included IP address. And it shows that the IP address is automatically assigned to 192.168.200.4.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.1. 5 Check Rocky IP Address

At the Ubuntu Client also type the same command to see the IP address which is also automatically assigned to 192.168.200.5.

A screenshot of a computer screen

AI-generated content may be incorrect.

Figure 2.1. 6 Check Ubuntu IP Address

Next, ping the Ubuntu Client from Rocky Server by typing ping command:

* ping 192.168.200.5

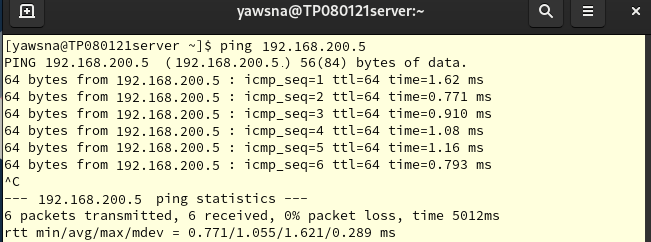


Figure 2.1.7 Ping Ubuntu Client

Ping the Rocky Server from Ubuntu Client by typing:

* ping 192.168.200.4

Which means that Rocky Server and Ubuntu Client can successfully communicate with each other by using our NAT Network.

A screenshot of a computer screen

AI-generated content may be incorrect.

Figure 2.1.8 Ping Rocky Server

## 2.2 Set up Hostname on server and client using FQDN

FQDN is the full and accurate name of a system on the Internet or on a local network. It ensures unique identification of the system throughout the network.

**Step 1**:

Before configuring the Rocky Server hostname, check the system's default hostname using the following command:

* hostname

It shows that the default hostname of this device is TP080121server.kaiyuan.org.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.2.1 Check Rocky Hostname

**Step 2**:

Next, change the hostname by typing this command:

* sudo nano /etc/hostname

A black and white text

AI-generated content may be incorrect.

Figure 2.2.2 Edit the hostname

**Step 3**:

After exit the nano editor, check the hostname is being changed or not:

* cat /etc/hostname

And it shows that the hostname is being changed successfully.



Figure 2.2 3 Check the hostname after modified

**Step 4**:

Next, we add our Rocky Server IP and its FQDN, edit the /etc/host by typing the following command:

* sudo nano /etc/hosts

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.2.4 Maps hostnames to Rocky Server IP Address

**Step 5**:

After exit the nano editor, use the cat /etc/hosts to make sure that the edit is successfully saved.

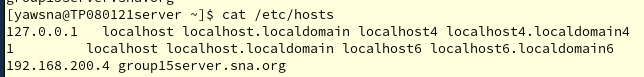


Figure 2.2. 5 Checking Hosts File

Last, reboot Rocky Server after the hostname is saved.



Figure 2.2 6 Reboot

**Step 6**:

If the changes that made are correct and saved successfully, ping the Rocky Server by typing the server's name instead of using the IP address.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.2.7 Testing the FQDN

**Step 7**:

Next, switch to the Ubuntu Client and check the hostname.

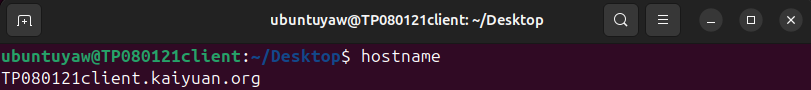


Figure 2.2.8 Check the hostname

**Step 8**:

After checking the hostname, we remove the previous hostname and type this command to add our FQDN:

* sudo nano /etc/hostname

A screen shot of a computer

AI-generated content may be incorrect.

Figure 2.2.9 Configure Hostname

**Step 9**:

Then check the FQDN with this command:

* sudo cat /etc/hostname



Figure 2.2. 10 Checking hostname file

When the hostname is already be changed, reboot the system.



Figure 2.2. 11 Reboot

And finally, the hostname of both devices is set up successfully.

## 2.3 Static IP Configuration on Rocky

A static IP address is an IP address that does not change. This is different from a dynamic IP address that is assigned by DHCP and can change over time.

**Step 1**:

First, verified that the NetworkManager package was installed on the Rocky Server by using the following command:

* rpm –q NetworkManager



Figure 2.3.1 Checking NetworkManager exist

**Step 2**:

After that, use these following commands to start the Network Manager Service:

* sudo systemctl enable NetworkManager
* sudo systemctl start NetworkManager

A screen shot of a computer

AI-generated content may be incorrect.

Figure 2.3.2 Enable NetworkManager Service

**Step 3**:

Then check the Network Manager status by typing the following command:

* sudo systemctl status NetworkManager

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.3.3 NetworkManager Status

**Step 4**:

After confirming the service status, check the network interface name and current IP configuration using by typing ifconfig.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.3.4 Check IP Address Configuration

**Step 5**:

Then, check the current network connection status and identify the correct interface in use by using this command:

* sudo nmcli general status
* sudo nmcli dev status

A screen shot of a computer

AI-generated content may be incorrect.

Figure 2.3.5 Check System Network Configuration

**Step 6**:

After checking the network connection status, go to the directory that contains the connection configuration files.

* sudo ls –l /etc/NetworkManager/system-connections



Figure 2.3.6 Identify Netowrk Connection Name

**Step 7**:

Next, open the connection profile using the nano editor to configure the settings:

* sudo nano /etc/NetworkManager/system-connections/enp0s3.nmconnection



Figure 2.3.7 Open nano editor to edit network configuration file

**Step 8**:

Then, make sure that the settings are same with the given picture below.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.3.8 Config IPv4

**Step 9**:

After exit the nano editor, restart the NetworkManager and check the status of it:

* sudo systemctl restart NetworkManager
* sudo systemctl status NetworkManager

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.3. 9 Restart the NetworkManager Service

**Step 10**:

After restart the Network Manager, check the network connection status again by using this command:

* nmcli

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.3. 10 Check the changes

**Step 11**:

To confirm the active connection, type the following command:

* nmcli connection show

Which can show the connections to us to ensure the correction of the connection.

A close up of numbers

AI-generated content may be incorrect.

Figure 2.3. 11 Check Network Settings

**Step 12**:

Also, check the enp0s3 connection by just using this command:

* nmcli connection show enp0s3

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.3. 12 Check Network Settings

**Ubuntu**

**Step 1:**

First, go to the Ubuntu terminal and check if the NetworkManager installed or not:

* dpgk –l |grep Network Manager

A computer screen shot of a program

AI-generated content may be incorrect.

Figure 2.3. 13 Check the NetworkManager is it install or not

**Step 2**:

As the Network Manager was installed by default, start the service by just using the following commands:

* sudo systemctl enable NetworkManager
* sudo systemctl start NetworkManager

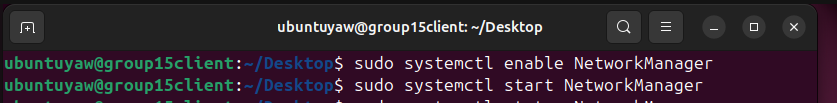


Figure 2.3.14 Enable NetworkManager Service

**Step 3**:

After starting the service up, check the status of the Network Manager:

* sudo systemctl status NetworkManager

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.3.15 NetworkManager Service

**Step 4**:

Next, use **ifconfig** commands to make sure the Ubuntu is connected to the enp0s3 connections.

A computer screen shot of a program

AI-generated content may be incorrect.

Figure 2.3.16 Check IP Address Configuration

**Step 5**:

Then, check the connections by using this command:

* sudo nmcli

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.3.17 Check Network Configuration

If the connection shows that it is connected to the enp0s3, it means that the static IP configuration is successful.

# 3.0 Network File System (NFS) Configuration

NFS is a protocol that allows a computer to access files over a network as if they were on its local hard drive.

This sub-section covers the installation of necessary NFS packages on both the Rocky Linux server and the Ubuntu client.

## 3.1 NFS Installation of Rocky

**Step 1**:   
To enable NFS functionality on the Rocky Linux server, the nfs-utils package, which provides the necessary utilities and daemons, is installed.

o sudo dnf install nfs-utils

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.1. 1 **:** Installation of the nfs-utils package on a Rocky Linux server

Figure 3.1.1 shows the terminal output for the installation of the nfs-utils package using the dnf package manager. This command installs nfs-utils along with its dependencies such as gssproxy, libev, libnfsidmap, libverto-libev, rpcbind, and sssd-nfs-idmap.

**Step 2**:

After installation, the NFS server service needs to be enabled to start automatically at boot and started for the current session.

o sudo systemctl enable nfs-server --now



Figure 3.1. 2 **:** Enabling and starting nfs-server service on Rocky Linux

Figure 3.1.2 illustrates this command's execution. The creation of a symbolic link confirms that the service is enabled and started.

**Step 3**:

To verify that the NFS server service is active and running correctly, its status is checked.

o sudo systemctl start nfs-server

o sudo systemctl status nfs-server

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.1. 3 **:** Checking nfs-server service status on Rocky Linux

Figure 3.1.3 displays the output after starting the service and then checking its status. The 'active (exited)' status indicates the NFS server processes have initialized successfully, and the main daemon is operational. The output shows processes for exportfs -r and rpc.nfsd.

**Step 4**:

To determine which versions of the NFS protocol the server supports, the /proc/fs/nfsd/versions file is examined.

o sudo cat /proc/fs/nfsd/versions



Figure 3.1. 4 **:** Showing available NFS versions on the Rocky Linux server

Figure 3.1.4 shows the available NFS versions on the Rocky Linux server. The output +3 +4 +4.1 +4.2 indicates the server supports NFS versions 3, 4, 4.1, and 4.2.

## 3.2 NFS Installation of Ubuntu

**Step 1**:

On the Ubuntu client, the local package index is updated to ensure access to the latest software information from configured repositories.

o sudo apt update -y

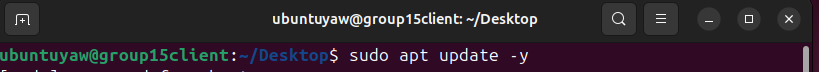


Figure 3.2.1 : Refreshing the package list on the Ubuntu client

**Step 2**:

The nfs-kernel-server package, which provides core NFS server components, is typically for servers but its installation status is checked on the client.

o sudo apt install nfs-kernel-server -y

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 3.2.2 **:** Trying to install the nfs-kernel-server package on Ubuntu client

Figure 3.2.2 shows the attempt to install nfs-kernel-server. The output indicates that nfs-kernel-server is already the newest version (1:2.6.4-3ubuntu5.1).

**Step 3**:

The nfs-common package, essential for NFS client functionality (mounting and interacting with NFS shares), is installed or its status verified.

o sudo apt install nfs-common -y

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.2.3 **:** nfs-common installation on Ubuntu client

Figure 3.2.3 showcases the command execution. The output nfs-common is already the newest version (1:2.6.4-3ubuntu5.1) and nfs-common set to manually installed confirms the necessary client-side tools are present.

## 3.3 NFS Configuration

This sub-section details the configuration steps performed on the Rocky Linux server to set up and export an NFS share.

**Step 1**:

A directory to be shared via NFS is created on the server.

o sudo mkdir /var/nfs/group15sna -p

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.3. 1 **:** Creating NFS shared directory (/var/nfs/group15sna) on the server

Figure 3.3.1 shows both the command prompt output and a file explorer view confirming the successful creation of the /var/nfs/group15sna directory.

**Step 2**:

The initial ownership and permissions of the newly created shared directory are displayed.

o sudo ls -dl /var/nfs/group15sna/

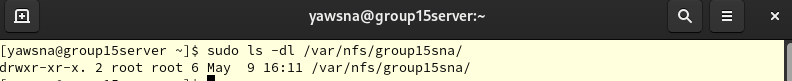


Figure 3.3. 2 **:** Showing initial shared directory permissions on the server

Figure 3.3.2 displays this command's output, showing the directory owned by root with default permissions.

**Step 3**:

The ownership of the shared directory is changed to nobody:nobody, a common practice for managing access control for NFS shares.

o sudo chown nobody:nobody /var/nfs/group15sna/



Figure 3.3. 3 **:** Changing shared directory ownership to nobody:nobody

**Step 4**:

The main NFS server configuration file, /etc/exports, is opened for editing to define shared directories.

o sudo nano /etc/exports



Figure 3.3. 4 **:** Opening the /etc/exports file on the server for editing

Figure 3.3.4 shows this command opening the file in the nano text editor.

**Step 5**:

An entry is added to the /etc/exports file to specify the shared directory, the client IP address authorized to access it, and associated permissions.

o /var/nfs/group15sna 192.168.200.81(rw,sync,no\_subtree\_check)

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.3. 5 **:** Configuration entry in /etc/exports defining the NFS share and its options

Figure 3.3.5 presents this configuration entry within the /etc/exports file. This line shares /var/nfs/group15sna with the client at 192.168.200.81 (Note: the primary client IP in this document is 192.168.200.5) granting read/write (rw), synchronous writes (sync), and disabling subtree checking (no\_subtree\_check).

**Step 6**:

To apply the changes made to /etc/exports, the NFS server's export table is updated. This command re-exports all directories listed in /etc/exports or synchronizes the current export table with /etc/exports.

o sudo exportfs -arv



Figure 3.3. 6 **:** Exporting NFS shared file systems using exportfs -arv

Figure 3.3.6 shows the output exporting 192.168.200.81:/var/nfs/group15sna, confirming the share is exported.

**Step 7**: The current list of exported file systems and their options is displayed to confirm the export status after configuration changes.

o sudo exportfs -s



Figure 3.3. 7 **:** Displaying current NFS exports and options with exportfs -s

Figure 3.3.7 exhibits the result, showing /var/nfs/group15sna exported to 192.168.200.81 with options like sync,wdelay,hide,no\_subtree\_check,sec=sys,rw,secure,root\_squash,no\_all\_squash.

**Step 8**:

The rpcbind and nfs-server services are enabled for auto-start and started immediately, then the nfs-server service is restarted to ensure all configurations are active.

o sudo systemctl enable --now rpcbind nfs-server

o sudo systemctl restart nfs-server

o sudo systemctl status nfs-server

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 3.3. 8 **:** Enabling, restarting, and checking status of rpcbind and nfs-server services

Figure 3.3.8 presents these system control commands. The status check confirms the nfs-server is active (exited).

**Step 9**:

To allow client traffic to reach the NFS services, rules are added to the server's permanent firewall configuration.

o sudo firewall-cmd --add-service={nfs,nfs3,rpc-bind,mountd} --permanent



Figure 3.3. 9 **:** Adding NFS-related services to the server's permanent firewall rules

Figure 3.3.9 shows the command to add nfs, nfs3, rpc-bind, and mountd services. The output success indicates the rules were added.

**Step 10**:

The firewall configuration is reloaded to apply the new rules without dropping existing connections.

o sudo firewall-cmd --reload



Figure 3.3. 10 **:** Reloading the firewall configuration on the server

Figure 3.3.10 shows this command and its success output.

**Step 11**:

The active configurations for the firewall's public zone are listed to verify that the necessary NFS-related services have been successfully added and are permitted.

o sudo firewall-cmd --zone=public --list-all

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 3.3. 11 **:** Listing active configurations for the public firewall zone

Figure 3.3.11 displays the public zone's active interface (enp0s3) and lists nfs, nfs3, mountd, and rpc-bind among the permitted services.

**Step 12**:

After firewall modifications, the systemd daemon is reloaded, and the nfs-server service is restarted and its status checked to ensure all configurations are correctly updated and active.

o sudo systemctl daemon-reload

o sudo systemctl restart nfs-server

o sudo systemctl status nfs-server

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 3.3. 12 Reloading systemd daemon and restarting/checking nfs-server status after firewall updates

Figure 3.3.12 captures this sequence, with the final status showing nfs-server.service as active (exited).

## 3.4 Testing NFS

This sub-section describes the procedures to test the NFS setup by accessing the shared directory from the client and verifying read/write operations.

**Step 1**:

On the NFS server, test files are created within the shared directory /var/nfs/group15sna/.

o cd /var/nfs/group15sna/

o sudo touch testingnfs{1..4}.txt

o ls -l

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 3.4. 1 **:** Creating test files in the server's shared NFS directory

Figure 3.4.1 demonstrates these command steps, with ls -l confirming the creation of testingnfs1.txt through testingnfs4.txt. Figure 3.4.2 provides a server file explorer view showing these test files.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.4. 2 **:** Server file explorer view showing test files in its shared directory

Figure 3.4.2 provides a file explorer depiction from the server, showing the test files testingnfs1.txt to testingnfs4.txt within /var/nfs/group15sna/, thus confirming their existence.

**Step 2**:

On the Ubuntu client, a local directory is created to serve as a mount point for the NFS share.

o sudo mkdir /nfs/Shared\_From\_group15server -p

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.4. 3 **:** Creating the local mount point directory on the Ubuntu client.

Figure 3.4.3 shows the mkdir command and an accompanying file explorer view as evidence of the directory's creation.

**Step 3**:

The NFS share from the server is mounted to the local mount point on the Ubuntu client.

o sudo mount 192.168.200.4:/var/nfs/group15sna /nfs/Shared\_From\_group15server



Figure 3.4. 4 **:** Mounting the NFS share from the server onto the Ubuntu client

Figure 3.4.4 shows the execution of this mount command. This command maps the server's shared directory to the client's local /nfs/Shared\_From\_group15server directory.

**Step 4**:

The df -h command is used on the client to list mounted file systems and verify that the NFS share is successfully mounted.

o df -h

A screen shot of a computer

AI-generated content may be incorrect.

Figure 3.4. 5 **:** Verifying the mounted NFS share on the client using df -h

Figure 3.4.5 displays the output, which includes the NFS share 192.168.200.4:/var/nfs/group15sna mounted on /nfs/Shared\_From\_group15server, confirming the successful mount. The listed size for the share is 22G with 16G available.

**Step 5**:

The contents of the mounted NFS share are listed from the client to verify read access to files created on the server.

o ls -l /nfs/Shared\_From\_group15server/

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 3.4. 6 **:** Client view of server's files in the mounted NFS share

Figure 3.4.6 illustrates this, displaying testingnfs1.txt through testingnfs4.txt.

**Step 6**:

To test write access from the client to the NFS share, a new file is created in the mounted directory.

o sudo touch /nfs/Shared\_From\_group15server/client\_g15.txt



Figure 3.4. 7 **:** Client creating a test file (client\_g15.txt) in the NFS share

**Step 7**:

On the server, the contents of the shared directory /var/nfs/group15sna/ are listed to confirm that the file created by the client (client\_g15.txt) is present.

o ls -l /var/nfs/group15sna/

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.4. 8 **:** Server's shared directory listing showing the client-created file

Figure 3.4.8 displays the output, showing client\_g15.txt (owned by nobody:nobody) alongside the server-created files. Figure 3.4.9 provides a server file explorer view also confirming the presence of client\_g15.txt.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.4. 9 **:** Server file explorer view confirming the presence of the client-created file

**Step 8**: As a further test, a directory named testsna is created within the shared NFS directory on the server.

o sudo mkdir testsna

o ls -l

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.4. 10 **:** Server creating a test directory (testsna) in the NFS share

Figure 3.4.10 records this test, with the ls -l output confirming the creation of the testsna directory on the server.

**Step 9**:

On the client, the contents of the mounted NFS share are listed again to verify that the directory change made on the server is visible.

o ls -l /nfs/Shared\_From\_group15server/

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.4. 11 **:** Client view of the NFS share showing the server-created testsna directory

Figure 3.4.11 shows the output on the client, where the testsna directory appears, confirming synchronization.

## 3.5 Permanent Mount Points Configuration

This sub-section outlines how to configure the Ubuntu client to automatically mount the NFS share at boot time using /etc/fstab.

**Step 1**:

The /etc/fstab file (file system table) on the Ubuntu client is opened for editing using the nano text editor.

o sudo nano /etc/fstab



Figure 3.5. 1 **:** Opening the /etc/fstab file on the Ubuntu client for editing

Figure 3.5.1 captures the execution of this command.

**Step 2**:

A new line is added to /etc/fstab to define the NFS share, its local mount point, file system type, and mount options.

o 192.168.200.4:/var/nfs/group15sna /nfs/Shared\_From\_group15server nfs auto,nofail,noatime,nolock,intr,tcp,actimeo=1800 0 0

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 3.5. 2 **:** NFS mount entry added to the client’s /etc/fstab file

Figure 3.5.2 illustrates the relevant excerpt of the /etc/fstab file with this new entry. The auto option enables automatic mounting at boot.

**Step 3**: To apply the /etc/fstab changes and test the permanency of the mount, the sudo mount -a command is executed, followed by a system reboot.

o sudo mount -a

o sudo reboot



Figure 3.5. 3 **:** Using sudo mount -a and reboot commands on the client

Figure 3.5.3 captures the execution of these commands.

**Step 4**:

After the client machine has rebooted, the contents of the mount point are listed to confirm that the NFS share was automatically mounted.

o ls -l /nfs/Shared\_From\_group15server

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 3.5. 4 **:** Verifying automatic mounting of the NFS share on the client after reboot

Figure 3.5.4 presents the output, showing the shared files and directories (client\_g15.txt, testingnfs1.txt to testingnfs4.txt, and testsna), confirming the automatic mount was successful.

# 4.0 Apache Web Server Configuration

The Apache Web Server is a widely used open-source web server software that allows you to host and serve websites over the internet or a local network.

## 4.1 Apache Web Server Installation

**Step 1:**

To install Apache Web Server, use the following command to access it.

o **sudo dnf install httpd -y**

A white background with black text

AI-generated content may be incorrect.

Figure 4.1.1 Install Apache

**Step 2**:

After installation, we need to enable and launch the Apache Web Server by using the following command:

o **sudo systemctl enable httpd**

o **sudo systemctl start httpd**



Figure 4.1.2 Enable Apache Service

**Step 3**:

Check the status of httpd to make sure the Apache web server is enabling and active:

o **sudo systemctl status httpd**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4.1.3 Apache Status

## 4.2 Apache Web Server Configuration

**Step 1**:

let http and https service allow by the firewall, then reload it.

o **sudo firewall-cmd –permanent –add-service=http**

o **sudo firewall-cmd –permanent –add-service=https**

o **sudo firewall-cmd --reload**

A close-up of a computer code

AI-generated content may be incorrect.

Figure 4.2.1 Allow by firewall

**Step 2**:

We need to create separate directories to saving the document roots for two websites, to complete it, use the two following commands:

o **sudo mkdir -p /var/www/group15server.sna.org/public\_html**

o **sudo mkdir -p /var/www/group15serverweb2.sna.org/public\_html**



Figure 4.2.2 Create directory

**Step 3**:

After creating the documents, we need to change the ownership of the documents:

o **sudo chown -R apache:apache /var/www/group15server.sna.org**

o **sudo chown -R apache:apache /var/www/group15serverweb2.sna.org**

And we need to grant read and execute permissions to others but only write permission for the owner:

o **sudo chown -R 755 /var/www**



Figure 4.2.3 Change Permission

**Step 4**:

To test web server responses, we need to generate simple HTML content by using these commands:

o **echo "<h1>Welcome group15 apache web test1</h1>" | sudo tee /var/www/group15server.sna.org/public\_html/index.html**

o **echo "<h1>Welcome group15 apache web test2</h1>" | sudo tee /var/www/group15serverweb2.sna.org/public\_html/index.html**

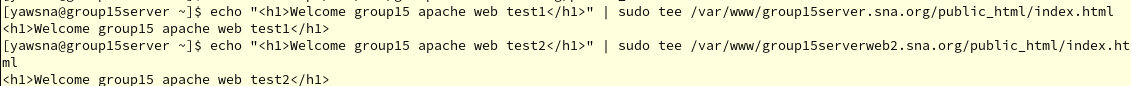


Figure 4.2.4 HTML Code

**Step 5**:

Edit the virtual configuration file to write up the content of the website:

o **sudo nano /etc/httpd/conf.d/group15server.sna.org.conf**



Figure 4.2.5 Create Virtual Host

**Explanation**:

The purpose of enabling virtual hosts is to host multiple websites on one server. Each website can have its own configuration, making the Apache server flexible, organized and scalable.

**Step 6**:

By including these entries, Apache delivers requested content from certain directories when clients visit the mentioned domain names.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4.2.6 Virtual Host Configuration

**Explanation of the above commands**:

* **<VirtualHost \*:80>**: This defines a virtual host listening on port 80 and \* means it will respone to any IP address on that port.
* **ServerName group15server.sna.org**: This specifies the domain name for this virtual host.
* **DocumentRoot /var/www/group15server.sna.org/public\_html**: This sets the root directory for the website content.
* **<Directory /var/www/group15server.sna.org/public\_html>** : This section controls access to the specified directory
* **Options Indexes FollowSymLinks**: Allows listing of the directory if no index file is present. Allows symbolic links in this directory.
* **AllowOverride None**: .htaccess files will not override these settings.
* **Require all granted**: Allows access to all clients.
* **</Directory>:** This close the directory block.
* **ErrorLog /var/log/httpd/group15server\_error.log**: The path to the error log file for this virtual host.
* **CustomLog /var/log/httpd/group15server\_access.log combined**: The path to the access log file for this virtual host.
* **</VirtualHost>**: This close the virtual host block.

**Step 7**:

Repeat the same command in *figure 4.2.5* and *4.2.6* (change to second website name)



Figure 4.2.7 Second Website

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4.2.8 Virtual Host Configuration

**Step 8**:

Check the syntax error in Apache's configuration files before restarting it:

o **sudo apachectl configtest**



Figure 4.2.9 Check syntax error

**Step 9**:

Restart Apache after applies the new virtual host, then check the status to ensure Apache is running and listening on the expected ports:

o **sudo systemctl restart httpd**

o **sudo systemctl status httpd**

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 4.2.10 Restart Apache

## 4.3 Access the Website in Ubuntu Client

**Step 1**:

We need to maps the domain names to the Rocky server’s IP address on the Ubuntu client. This command can help us to access it:

o **sudo nano /etc/hosts**



Figure 4.3.1 Check Ubuntu hosts

**Step 2**:

Add this entry then save and exit:

o **192.168.200.4 group15server.sna.org group15serverweb2.sna.org**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4.3.2 hosts configuration

**Step 3**:

Open Firefox and enter to ensure virtual host and document root were configured correctly:

o **http://group15server.sna.org**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4.3.3 Testing Apache Website 1

**Step 4**:

Open a new tab to check the second website:

o **http://group15serverweb2.sna.org**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4.3.4 Testing Apache Website 2

# 5.0 Configure a Certificate Authority (CA)

A certificate authority (CA) is a trusted organization or company that issues digital certificates used to verify the identity of a website, user, or device and enable secure communications on the Internet.

## 5.1 OpenSSL Installation

**Step 1**:

To configure SSL in Apache, you need to install mod\_ssl because Apache by default only serves content over HTTP.

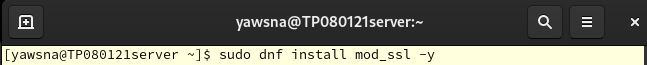
o sudo dnf install mod\_ssl -y

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.1.1 Install openssl*

**Step 2**:



*figure 5.1.2 Install modssl*

The command **sudo dnf install mod\_ssl -y** is run to install the SSL module for the Apache Web Server on the server. This module allows Apache to utilize HTTPS connections or securely transferring web content through encryption.

## 5.2 Preparing the Public Key Infrastructure Directory

**Step 1**:

Create the Public Key Infrastructure Directory using the following commands:

o sudo mkdir -p /etc/pki/CA/{certs,crl,newcerts,private}:

o sudo chmod 700 /etc/pki/CA/private

o sudo touch /etc/pki/CA/index.txt

o sudo echo 1000 > /etc/pki/CA/serial



*figure 5.2.1 Create Public Key Infrastructure Directory*

The displayed screenshot shows commands that were run to set up the Certificate Authority (CA) working directory structure:

* **sudo mkdir -p /etc/pki/CA/{certs,crl,newcerts,private}:**

Creates the main CA directory and subdirectories for certificates, CRLs, new certificates, and the private key.

* **sudo chmod 700 /etc/pki/CA/private:**

Sets permissions (700) on the private directory to restrict access to the owner only, crucially protecting the CA's private key.

* **sudo touch /etc/pki/CA/index.txt:**

Creates the index.txt file, which will serve as the CA's certificate database log.

* **sudo echo 1000 > /etc/pki/CA/serial:**

Initializes the serial file with 1000. This file tracks the next serial number for issued certificates.

## 5.3 Create a Certificate Authority

**Step 1**:

To Configure OpenSSL execute the command:

o sudo nano /etc/ssl/openssl.cnf.



*figure 5.3.1 Open Configuration File*

Use sudo to call the nano text editor to open the /etc/ssl/openssl.cnf file. This file is the main configuration file of OpenSSL.

**Step 2**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.3.2 Make Sure req\_extensions exists*

This section defines the default settings used when generating CSRs and self-signed certificates. Make sure req\_extensions = v3\_req exists.

* **default\_bits = 2048:** Sets the default key size to 2048 bits.
* **default\_md = sha256:** Specifies the default digest algorithm as SHA-256.
* **default\_keyfile = privkey.pem:** Sets the default filename for the private key.
* Other lines define templates for required information and extensions in the certificate request and the final certificate.

**Step 3**:

Make sure add the following line

o subjectAltName = @alt\_names

o DNS.1 = group15server.sna.org

o DNS.2 = [www.group15server.sna.org](https://www.group15server.sna.org)

A screenshot of a computer

AI-generated content may be incorrect.

figure 5.3.3 SAN configuration

[ v3\_req ] Section: Defines standard Version 3 extensions for certificate requests. It also links to the [ alt\_names ] section for Subject Alternative Names.

[ alt\_names ] Section: Specifies the Subject Alternative Names (SANs) for the certificate.

* DNS.1 = group15server.sna.org
* DNS.2 = [www.group15server.sna.org](https://www.group15server.sna.org)
* These entries display the domain names the certificate will be valid for in addition to the common name. This is significant for web servers where clients verify the certificate by comparing it to the domain name they are accessing.

**Step 4**:

Create the Private Key and Root Certificate

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.3.4 Create Private Key and Root Certificate*

**sudo openssl genrsa -out .../group15\_CA\_key.pem 2048:**

* Generates the CA's private key using the RSA algorithm with a 2048-bit length.
* Saves the private key to the specified file (/etc/pki/CA/private/group15\_CA\_key.pem).

**sudo openssl req -x509 -new -nodes -key .../group15\_CA\_key.pem -out .../group15\_CA\_cert.pem -days 3650:**

Purpose: Creates a self-signed X.509 certificate (the CA's root certificate).

**-x509**: Output a self-signed certificate instead of a CSR.

**new**: Generate a new certificate request.

**nodes**: Do not encrypt the private key.

**key ...:** Specifies the CA private key generated in the previous step as the signing key.

**out ...:** Specifies the output file for the CA's root certificate (/etc/pki/CA/group15\_CA\_cert.pem).

**days 3650**: Sets the certificate's validity period to 3650 days.

**Step 5**:

Create a CSR (Certificate Signing Request) for the server

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.3.5 Create CSR*

**sudo openssl req -new -key /path/to/server.key -out /path/to/server.csr:**

Purpose: Generates a new certificate signing request for the server.

-new: Indicates generating a new request.

-key ...: Specifies the private key file that corresponds to this CSR.

-out ...: Defines the output file path for the generated CSR (/etc/pki/CA/G15\_ser.csr).

**Step 6**:

Sign the CSR with the CA to generate the server certificate

osudo openssl ca -batch -keyfile /etc/pki/CA/private/group15\_CA\_key.pem -cert /etc/pki/CA/group15\_CA\_cert.pem -in /etc/pki/CA/G15\_ser.csr -out /etc/pki/tls/certs/ggggg15.crt -days 365 -notext -md sha256:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.3.6 Sign CSR with the CA*

**Explanation of the above commands:**

This command is used by the CA to sign a CSR and issue a new certificate.

* **-batch**: Runs the command in non-interactive mode.
* **-keyfile** ...: Specifies the CA's private key file.
* **-cert** ...: Specifies the CA's root certificate file.
* **-in ...**: Specifies the input file (the server's CSR) that was generated earlier.
* **-out** ...: Specifies the output file path for the newly issued server certificate (/etc/pki/tls/certs/ggggg15.crt).
* **-days 365**: Sets the validity period of the issued server certificate to 365 days.
* **-notext**: Prevents outputting the text version of the certificate.
* **-md sha256**: Specifies the SHA-256 algorithm for signing.

This displays the output of the openssl ca command from the previous step.

It verifies the signature on the certificate request is valid.

A list of information about the newly created server certificate will be shown including the Serial Number, the Validity dates, information about the Subject and Subject Alternative Names.

It confirms the certificate is certified for 365 days.

Lastly, it shows the CA's database (index.txt) has been modified with the new certificate entry.

## 5.4 Create a Virtual Host

**Step 1**:



*figure 5.4.1 Create Virtual Host*

**sudo nano /etc/httpd/conf.d/ssl\_group15server.conf:**

This command accessing the Apache configuration file where the server will be set up to use the issued SSL certificate to enable secure HTTPS connections for the website.

**Step 2**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.4.2 Virtual Host Configuration*

This shows the configuration details within the Apache SSL Virtual Host file.

**<VirtualHost \*:443>**: Defines a virtual host listening on port 443, the standard port for HTTPS.

**ServerName group15server.sna.org**: Specifies the domain name this virtual host responds to.

**DocumentRoot** ...: Points to the directory containing the website files.

**SSLEngine on**: Enables SSL/TLS for this virtual host.

**SSLCertificateFile ...ggggg15.crt**: Specifies the path to the server's signed certificate.

**SSLCertificateKeyFile ...group15\_CA.key.pem**: Specifies the path to the server's private key.

**SSLCACertificateFile ...group15\_CA.cert.pem**: Specifies the path to the CA's root certificate. This allows clients to verify the server's certificate chain.

**Step 3**:



*figure 5.4.3 Check Syntax*

**sudo apachectl configtest:**

This command verifying the Apache configuration for syntax errors after setting up the SSL virtual host, confirming that the changes are valid before applying them.

**Step 4:**



*figure 5.4.4 Restart httpd*

**sudo systemctl restart httpd:**

This screenshot documents restarting the Apache service to apply the new SSL configuration, making the website accessible over HTTPS.

## 5.5 Distribute the Public Certificate of the CA

**Step 1**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.5.1 Use SSH Connect ubuntu and rocky*

**sudo apt install openssh-server –y:**

This command is installing the SSH server on the Ubuntu client, enabling secure remote login capabilities to the client machine.

**Step 2**:

A screen shot of a computer

AI-generated content may be incorrect.

*figure 5.5.3 Enable SSH*

**sudo systemctl enable ssh:**

This enabling the SSH service on the Ubuntu client, ensuring that secure remote login via SSH is available after the system starts.



*figure 5.5.4 Start SSH*

**sudo systemctl start ssh**:

Immediately starts the SSH service on the Ubuntu client for the current session.

**Step 3**:

Send CA to ubuntu using the following commands

o scp /etc/pki/CA/group15\_CA.cert.pem ubuntuyaw@192.168.200.81:~:



*figure 5.5.5 Send CA to Ubuntu*

Securely copies the CA's public root certificate file from the server to the home directory of the user on the Ubuntu client (IP 192.168.200.81) using SCP.

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.5.6 Ubuntu Received the CA*

This screenshot provides evidence that the CA's root certificate was successfully transferred and received on the Ubuntu client, located in the user's home directory.

**Step 4**:

A computer screen shot of a program

AI-generated content may be incorrect.

*figure 5.5.2 Update Certificate*

**sudo cp /path/to/group15server.crt /usr/local/share/ca-certificates/group15CA.crt:**

Copies the public root certificate of custom CA to a standard system directory on Ubuntu where additional CA certificates are stored.

**sudo update-ca-certificates:**

Updates the system's list of trusted CA certificates.

Processes the certificates in the designated directories and incorporates the custom CA into the system's trust store. The output confirms "1 added" and the database is updated.

## 5.6 Testing HTTPS between Client and Server

**Step 1**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.6.1 Firefox privacy setting*

This shows testing the HTTPS connection to the server from the client, showing how the browser reacts to the server's certificate and allows viewing the certificate details to verify it was issued by our custom CA.

**Step 2**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.6.2 Firefox Certificate Manager*

This proves that we have successfully added our custom CA to the list of trusted authorities in the client browser. This is important for the browser to trust the server certificate issued by our CA so it can successfully establish a secure HTTPS connection with no warnings.

**Step 3**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.6.3 Certificate Detail*

This screenshot shows the "Edit CA certificate trust settings" dialog box in Firefox.

This dialog allows the user to define what services the browser should trust a specific certificate for when it's treated as a Certificate Authority.

**Step 4**:

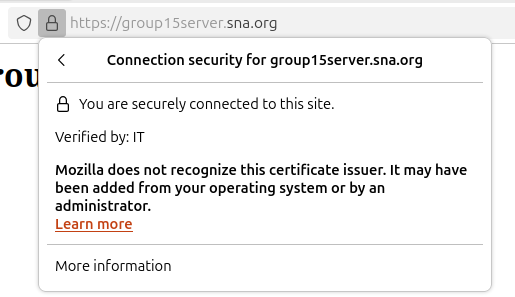
A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.6.4 The Lock Logo Exists*

This proof of a successful and secure HTTPS connection between the client and server. The message confirms that the connection was verified using our custom CA, demonstrating that the CA setup and client trust configuration were successful for web security.

**Step 5**:



*figure 5.6.5 Website is Verified by IT*

This shows that the HTTPS connection is secure and trusted. It illustrates that our custom CA (known as "IT") verified the server's certificate, which resides in the operating system's trust store, proving that the secure web server and CA trust were implemented properly.

**Step 6**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.6.6 Website Is Encrypted*

This displays the "Page Info" window in Firefox, specifically the "Security" tab, for <https://group15server.sna.org/>.

It confirms the website identity is "Verified by: IT".

Shows the connection is "Connection Encrypted" using strong protocols and cipher suites (like TLS 1.3 with AES\_256\_GCM\_SHA384).

**Step 7**:

After pressing the view certificates button, we can see that these are the certificates we created, one is the root certificate and the other is the CA certificate.

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.6.7 Root Certificate and CA Certificate*

## 5.7 Postfix and Dovecot Configuration

**Step 1**:



*figure 5.7.1 Open Postfix Configuration File*

**sudo nano /etc/postfix/main.cf:**

The screenshot shows accessing the main configuration file for the Postfix mail server, which is essential to setting up and configuring the email services, including enabling secure email transport as required by the assignment.

**Step 2**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.7.2 Configure Postfix*

This display configuration lines within the Postfix main configuration file (/etc/postfix/main.cf) on the server.

Purpose: These lines configure Postfix to support secure email transfer using TLS for both sending and receiving mail.

* **smtpd\_use\_tls = yes:** Enables TLS for incoming SMTP connections.
* **smtpd\_tls\_cert\_file = ...ggggg15.crt:** Specifies the server's certificate file to be used for TLS.
* **smtpd\_tls\_key\_file = ...group15\_CA.key.pem:** Specifies the server's private key file for TLS.
* **smtp\_tls\_CAfile = ...group15\_CA.cert.pem:** Specifies the CA's root certificate file for verifying other mail servers' certificates.
* Other settings define TLS protocols, security levels, and integrate with SASL for authentication.

**Step 3**:



*figure 5.7.3 Restart Postfix*

**sudo systemctl restart postfix:**

Restarting the service is necessary for any recent configuration changes to become active.

**Step 4**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.7.4 Configure dovecot*

This display configuration lines within the Dovecot SSL configuration file (/etc/dovecot/conf.d/10-ssl.conf) on the server.

These lines configure Dovecot, the IMAP/POP3 server, to support secure connections using SSL/TLS for email retrieval.

* **ssl = yes:** Enables SSL/TLS support for Dovecot.
* **ssl\_cert = <...ggggg15.crt:** Specifies the server's certificate file path.
* **ssl\_key = <...group15\_CA.key.pem:** Specifies the server's private key file path.
* **ssl\_ca = <...group15\_CA.cert.pem:** Specifies the CA's root certificate file path.

**Step 5**:



*figure 5.7.5 Restart dovecot*

Restarting the Dovecot service to apply the secure email retrieval configuration, preparing it to handle encrypted connections from email clients.

## 5.8 Testing secure email between Client and Server

**Step 1**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.7.6 Find Manage Certificate in Thunderbird*

This screenshot shows the steps in configuring the email client certificate settings, which is required to ensure that the email client can securely connect to the mail server and trust the server certificate issued by custom CA to send and pay email secure and receive secure mail.

**Step 2**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.7.7 Add Certificate*

Purpose: The client asks the user to explicitly trust a new certificate called "group15server.sna.org" as a CA.

Key Action: The checks in the "Trust this CA to identify websites" and "Trust this CA to identify email users" checkboxes are made, allowing the client to use this CA to verify server certificates for both web and, most importantly, email.

**Step 3**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.7.8 Set up Account*

The above screenshot shows the account settings for an email client on Ubuntu Client and configuration to connect to a mail server.

Key Secure settings:

INCOMING SERVER (IMAP): Configured to connect to group15server.sna.org using Port 993 with connection security SSL/TLS. This is the normal configuration for IMAPS.

OUTGOING SERVER (SMTP): Configured to connect to group15server.sna.org using Port 465 with connection security SSL/TLS. This is the normal configuration for SMTPS.

A screenshot of a security exception

AI-generated content may be incorrect.

*figure 5.7.9 Validate Certificate*

The screenshot above confirms that the email client was able to verify the server's certificate for the specified address. The message "Valid Certificate" implies that the server's identity is trusted therefore, secure email connections are possible.

**Step 4**:

After pressing the view certificates button, we can see that these are the certificates we created, one is the root certificate and the other is the CA certificate.

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.7.10 Root Certificate and CA certificate*

**Step 5**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.7.11 Testing Email*

This is a record for SECTION 5.8 "Testing secure email between Client and Server". It depicts the user composing an email using an email client that has been configured. They are sending it from an account on the server to the same account. The fact that they are composing and sending and will check if it is received is the main process of testing the secure email setup.

**Step 6**:

A screenshot of a computer

AI-generated content may be incorrect.

*figure 5.7.12 Send Email Securely*

This shows the inbox of the email client of the Ubuntu client, display the mail folders and received mail messages for the yawsna@group15server.sna.org account.

It is clearly showing the test email with the subject "testing secure CA" in the inbox list, while the right pane displays the content within.

The fact there is an email within the inbox clearly shows that it has been successfully delivered and then received mail.

# 6.0 Additional Service

Samba is an open-source software suite that enables Linux/Unix systems to share files and printers with Windows systems in a network environment. It is highly configurable and can achieve cross-platform communication. The protocol it uses is the Server Message Block Protocol (SMB), which complies with the standard for file sharing on Windows networks.

## 6.1 Samba Installation

**Step 1**:

Use the following command to install Samba

o sudo dnf install samba samba-client samba-common -y

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6.1.1 Samba Installation

**Step 2**:

Enable and start the samba service

o sudo systemctl enable smb

o sudo systemctl start smb



Figure 6.1.2 Start Samba Service

**Step 3**:

Check the status of samba service

o sudo systemctl status smb

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6.1.3 Samba Service Status

## 6.2 Samba Configuration

**Step 1**:

Open the samba configuration file

o sudo nano /etc/samba/smb.conf



Figure 6.2.1 Configuration of Samba

**Step 2**:

We add the following command to define the private shared folder and public folder

o [ShareFolder]

o path = /srv/samba/shared

o valid users = @sambashare

o read only = no

o browsable = yes

o force user = sambauser

o writable = yes

o [PublicFolder]

o path = /srv/samba/public\_shared

o browsable = yes

o guest ok = yes

o read only = no

o writable = yes

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6.2.2 Configuration of Samba

**Explanation of the above commands**:

* **[ShareFolder]:** This is the name of the Samba share, for example [\\server\_name\ShareFolder](file:///\\server_name\ShareFolder). This share is more secure and typically used for restricted access.
* **path = /srv/samba/shared**: This specifies the directory path on the server that will be shared.
* **valid users = @sambashare:** This restricts access to the share to users who are members of the smbshareusers group.
* **read only = no**: This setting allows **read/write** access to the shared folder.
* **browsable = yes:** This setting means that the share will be visible when browsing the network in a Linux file manager.
* **force user = sambauser**: This forces all operations on this share to be executed by the **sambauser** account,
* **writable = yes**: This allows the users can create, modify, and delete files in this shared folder.
* **[PublicFolder]**: This is the name of the Samba share, for example [\\server\_name\PublicFolder](file:///\\server_name\PublicFolder). This share is intended for public access.
* **guest ok = yes:** This setting allows guest access to the share.

**Step 3**:

Check the configuration file for syntax errors. If correct, there will be an OK message.

o testparm

A close-up of a text

AI-generated content may be incorrect.

Figure 6.2.3 Check Configuration File Syntax

**Step 4**:

Restart the samba service

o sudo systemctl restart smb



Figure 6.2.4 Restart Samba Service

**Step 5**:

Create a sambabuser who can only access ShareFolder

o sudo useradd -M -s /sbin/nologin sambauser

o sudo groupadd sambashare

o sudo usermod -aG sambashare sambauser

o sudo smbpasswd -a sambauser

o sudo smbpasswd -e sambauser

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6.2.5 Create sambauser Account

**Explanation of the above commands:**

* sudo useradd -M -s /sbin/nologin sambauser

useradd -M: This command creates a new user without creating a home directory. The -s option sets the login shell for the user. /sbin/nologin is specified, meaning the user cannot log into the system interactively.

* sudo groupadd sambashare

This command creates a new group called sambashare. This group will be used to manage Samba users who need access to shared folders.

* sudo usermod -aG sambashare sambauser

This command adds the sambauser to the sambashare group. -aG: ensures that the group the user is currently in is not deleted and adds the user to the group.

* sudo smbpasswd -a sambauser

This command adds sambauser to Samba's password database. We need to set a new password for sambauser during this command.

* sudo smbpasswd -e sambauser

This command enables the sambauser in the Samba database, allowing the user to authenticate and access the shared folders.

**Step 6**:

Change the ownership of the file using the following command

o sudo chown -R sambauser:sambashare /srv/samba/shared

o sudo chown -R 0775 /srv/samba/shared

o sudo chown -R nobody:nobody /srv/samba/public\_shared

o sudo chown -R 0777 /srv/samba/public\_shared



Figure 6.2.6 Change Permission



Figure 6.2.7 Change Permission

**Explanation of the above commands**:

* **sudo chmod**: This is used to modify the permissions of a file or directory
* **-R**: This ensures the permissions are applied to the directory and all its contents
* **sambauser:sambashare**: The ownership is being set to the user sambaser and the group
* **nobody:nobody**: This is the user and group typically used for guest access in Linux. It's often used for anonymous access.
* **/srv/samba/shared**: The directory whose ownership is being modified. This is a shared directory intended for sambauser access.
* **/srv/samba/public\_shared**: The directory whose ownership is being modified. This is a shared directory intended for public access.
* **0775**: Full read, write, and execute permissions for the owner, group. Full read and execute permissions for other.
* **0777**: Full read, write, and execute permissions for the owner, group and other.

**Step 7**:

Add the following services to allow by the firewall

o sudo firewall-cmd –zone=public –add-service=samba –permanent

o sudo firewall-cmd --reload

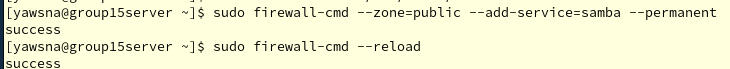


Figure 6.2.8 Firewall Configuration

**Step 8**:

Create some file in the shared path and public shared path

o sudo mkdir -p /srv/samba/shared

o echo “This is test file for samba share group15.” | sudo tee /srv/samba/shared/testsmb.txt

o ls /srv/samba/shared

o sudo mkdir -p /srv/samba/public\_shared

o sudo touch /srv/samba/public\_shared/group15test.txt

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6.2.9 Create some Testing File

**Explanation of the above commands**:

* **sudo mkdir -p /srv/samba/shared**: This command creates the /srv/samba/shared directory. -p: This option tells mkdir to create any parent directories that do not already exist.
* **sudo tee /srv/samba/shared/testsmb.txt**: This command use tee to redirect the output to a file. The output will be written to the /srv/samba/shared/testsmb.txt file.
* **sudo touch /srv/samba/public\_shared/group15test.txt**: Create an empty test file in the public\_shared folder

**Step 9**:

SELinux is a mandatory access control system. For samba to work successfully, we need to associate the correct SELinux context with the shared directory and its files.

o sudo setenforce 0

o sudo semanage fcontext -a -t samba\_share\_t “/srv/samba/shared(/.\*)?”

o sudo restorecon -Rv /srv/samba/shared

o sudo semanage fcontext -a -t samba\_share\_t “/srv/samba/public\_shared(/.\*)?”

o sudo restorecon -R /srv/samba/

o sudo setenforce 1

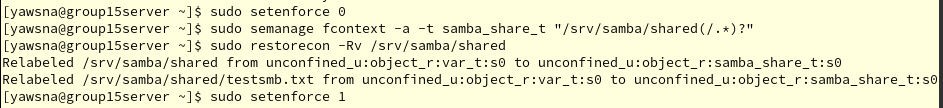


Figure 6.2.10 Configure SELinux

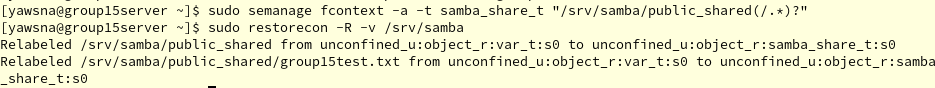


Figure 6.2.11 Configure SELinux

**Explanation of the above commands**:

* **setenforce 0**: This command temporarily disables SELinux enforcement on the system.
* **semanage fcontext**: This command manages file contexts in SELinux.
* **-a -t samba\_share\_t**: The **-a** option is for adding a new file context, and the **-t samba\_share\_t** specifies that the files in **/srv/samba/shared** should be labeled with the **samba\_share\_t** context which is a predefined SELinux type for Samba shared files.
* **“/srv/samba/shared(/.\*)?”**: This specifies the directory path **/srv/samba/shared** and uses **(/.\*)?** to recursively match it, including any files or subdirectories.
* **sudo restorecon -Rv /srv/samba/shared**: This command is used to restore the default SELinux security context on files and directories. **-R**: This option applies the command recursively to all files and subdirectories under **/srv/samba/shared**. **-v**: This option provides verbose output, so you can see what changes are being made. **/srv/samba/shared**: This is the directory on which the contexts are being restored.
* **setenforce 1**: This command re-enables SELinux enforcement,

## 6.3 Test on Ubuntu Client

**Step 1**:

Install the samba client

o sudo apt install samba-client cifs-utils -y



Figure 6.3.1 Install Samba on Ubuntu

**Step 2**:

Mount a samba shared directory from the Rocky server to the Ubuntu client. We need to enter a password because this is not a public folder

o sudo mount -t cifs //192.168.200.4/SharedFolder /mnt -o user=sambauser,uid=$MY\_UID,gid=$MY\_GID,vers=3.0

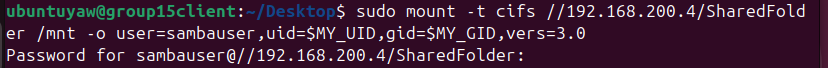


Figure 6.3.2 Mount Share Directory

**Explanation of the above commands**:

* **sudo mount**: This is the mount command with superuser privileges, used to mount file systems.
* **-t cifs**: Specifies the type of file system to mount. This is the protocol used by Samba for file sharing.
* **//192.168.200.4/SharedFolder**: This is the IP address of the rocky server and the name of the shared directory
* **/mnt**: This is the local mount point on the Ubuntu client where the Samba share will be mounted
* **-o user=sambauser,uid=$MY\_UID,gid=$MY\_GID,vers=3.0**: This specifies the mount options including the username, user id, group id, and the CIFS protocol version to use.

**Step 3**:

Check if our client has the file just created on the rocky server

o ls -l /mnt

o cat /mnt/testsmb.txt

A screenshot of a computer screen

AI-generated content may be incorrect.

Figure 6.3.3 Check the Shared File

**Step 4**:

We create a file on the client to see if the rocky server receives it.

o sudo touch /mnt/test\_from\_ubuntu.txt

o ls -l /srv/samba/shared



Figure 6.3.4 Create a file



Figure 6.3.5 Check file from Ubuntu

According to the results in the figure, the server and client successfully shared the directory.

**Step 5**:

Now we test to mount the public folder

o sudo mount -t cifs //192.168.200.4/PublicFolder /mnt/public -o guest,uid=$MY\_UID,gid=$MY\_GID,vers=3.0



Figure 6.3.6 Mount PublicFolder

**Step 6**:

Check if our client has the file just created on the rocky server

o ls -l /mnt/public

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6.3.7 Check File

**Step 7**:

We create a file on the client to see if the rocky server receives it.

o sudo touch /mnt/public/test\_from\_ubuntupublic.txt

o ls -l /srv/samba/public\_shared

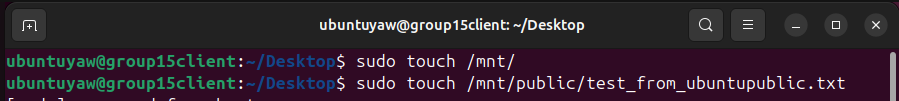


Figure 6.3.8 Create a File in PublicFolder

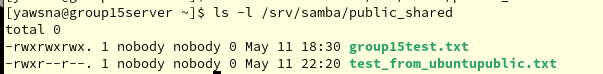


Figure 6.3.9 Check File from ubuntu

According to the results in the figure, the server and client successfully shared the directory.

## 6.4 Permanent Mount Point Configuration

Step 1:

To increase security, we do not directly enter the username and password in /etc/fstab but store them in a file.

o sudo mkdir -p /etc/samba/credentials

o sudo nano /etc/samba/credentials/g15server



Figure 6.4.1 Create a Directory for credential



Figure 6.4.2 Open /etc/samba/credentials/g15server file

**Step 2**:

We add the following command in nano:

o username=sambauser

o password=1234

o domain=MYGROUP

A screenshot of a computer

AI-generated content may be incorrect.

Figure 6.4.3 Credential Details

**Explanation of the above commands**:

**username=sambauser**: This specifies the username that will be used to authenticate when connecting to the Samba share.

**password=1234**: This specifies the **password** for the sambauser account.

**domain=MYGROUP**: This specifies the domain or workgroup to which the sambauser belongs.

**Step 3**:

To mount the file permanently we need to configure following file

o sudo nano /etc/fstab



Figure 6.4.4 Open /etc/fstab file

**Step 4**:

We add the following command to automatically mount the SharedFolder and PublicFolder on startup

o //192.168.200.4/PublicFolder /mnt/public cifs guest 0 0

o //192.168.200.4/SharedFolder /mnt cifs credentials=/etc/samba/credentials/g15server 0 0

A screenshot of a computer screen

AI-generated content may be incorrect.

Figure 6.4.5 Configure Permanent Mount Point

**Explanation of the above commands**:

* **//192.168.200.4/PublicFolder**: This is the Samba share we are mounting.
* **/mnt/public**: This is the mount point where the Samba share will be mounted on the Ubuntu client.
* **cifs**: CIFS protocol for Samba file sharing.
* **guest**: This indicates that the mount is done with guest access, meaning no credentials are required.
* **0 0**: These are dump and fsck options, which are typically set to 0 for network file systems like CIFS.
* **credentials=/etc/samba/credentials/g15server**: This tells the system to use the credentials file located at /etc/samba/credentials/g15server for authentication when connecting to the Samba share.

**Step 5**:

We reboot the ubuntu client



Figure 6.4.6 Reboot Ubuntu

**Step 6**:

Once finish the reboot, check if we can still access the file

o ls -l /mnt

o ls -l /mnt/public

A computer screen shot of a program

AI-generated content may be incorrect.

Figure 6.4.7 Check the file still exist after reboot

# 7.0 Conclusion

In summary, we were able to find the correct configuration method in the actual configuration of HTTPS and SMTPS and increase our experience in the field of system and network management in this assignment. In addition, we can also improve our practical capabilities in information security, service deployment, cross-system communication, fault diagnosis, and other aspects by completing this assignment.

After we resolve the issues encountered during the configuration of these clients and servers, we will consider the operations and configurations that may lead to problems as valuable experience gained from this assignment. This process has improved our ability to quickly locate the errors and troubleshoot in future actual works.

# 8.0 Workload Matrix

|  |  |  |
| --- | --- | --- |
| **Student Name** | **Task** | **Percentage** |
| Yaw Kai Yuan | - Additional Feature  - Documentation | 20% |
| Gan Kai Hong | - Network File System Configuration | 20% |
| Er Chen Liang | - Network Configuration  - Conclusion | 20% |
| Ting Jac Sheinn | - Certificate Authority Configuration | 20% |
| Chong Wai Kit | - Apache Web Server Configuration | 20% |

# 9.0 References

Codex, A. C. (2024, January 26). Configuring static IP address on Rocky Linux 9. Reintech. <https://reintech.io/blog/configuring-static-ip-address-rocky-linux-9>

Garn, D. (2022, April 1). Install Apache web server. Red Hat. <https://www.redhat.com/en/blog/install-apache-web-server>

Gridscale. (n.d.). Hostname and FQDN on Ubuntu. Gridscale. <https://gridscale.io/en/community/tutorials/hostname-fqdn-ubuntu/>

Hostman. (2023, November 24). *How to install and configure SSH on Ubuntu 22.04.* <https://hostman.com/tutorials/how-to-install-and-configure-ssh-on-ubuntu-22-04/>

Kenlon, S. (2022, June 16). Configure NFS on Linux. Red Hat. <https://www.redhat.com/en/blog/configure-nfs-linux>

Red Hat, Inc. (n.d.). Setting up Samba for high availability. In *High Availability Add-On Administration* (RHEL 7 documentation). Red Hat. <https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/7/html/high_availability_add-on_administration/s1-sambasetup-haaa>

SSLS.com. (2019, July 9). How to install an SSL certificate on Apache. SSLS.com. <https://www.ssls.com/knowledgebase/how-to-install-an-ssl-certificate-on-apache/>

Yang, K., & Tagliaferri, L. (2022, April 19). How to set up Apache virtual hosts on Ubuntu 20.04. DigitalOcean. <https://www.digitalocean.com/community/tutorials/how-to-set-up-apache-virtual-hosts-on-ubuntu-20-04>