**Server**

Task 1: Setup

1. Server Administrator Tasks (VM1):

- Create local user accounts for each client on the server.

**Client1:**

sudo useradd -m client1

sudo passwd client1

New password: client1

**Client2:**

sudo useradd -m client2

sudo passwd client2

New password: client2

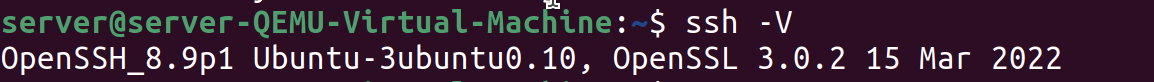
- Install and enable SSHD.

sudo apt install openssh-server

sudo systemctl enable ssh

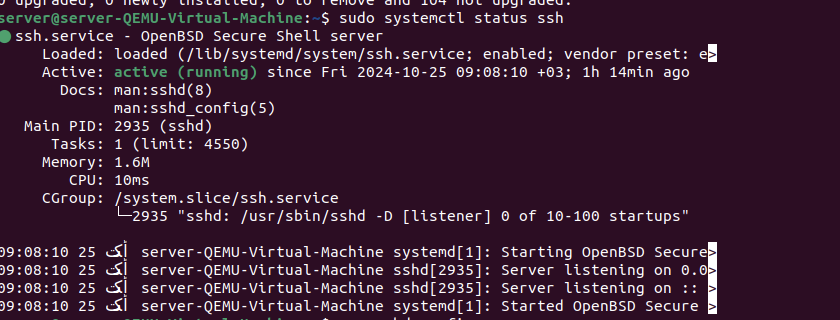
**- To check SSHD**

ssh -V



**Verify:**

sudo systemctl status ssh



- Configure SFTP for file transfer operations.

sudo nano /etc/ssh/sshd\_config

**ip address of current server:**

ifconfig

192.168.64.3

ssh [server@192.168.64.3](mailto:server@192.168.64.3)

**-Secure SFTP to client1 and client2 only**

sudo nano /etc/ssh/sshd\_config

*[at the end we modify and write the folowwing to ensure SFTP is restricted to client1 and client2 only]*

Match User clien1,client2

ForceCommand internal-sftp

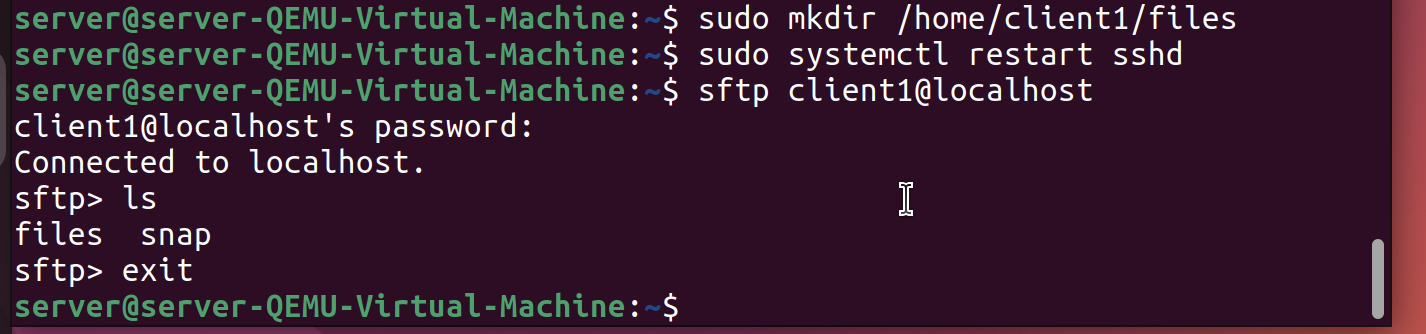
ChrootDirectory /home/%u

AllowTCPForwarding no

X11Forwarding no

**-Try to connect client1 and client2 via SFTP**

sftp client1@localhost



**if they fail to get connected, we ensure that clients have 755 permision and should be owned by root. This will also ensure they can download and upload files**

A screenshot of a computer program

Description automatically generated

Task 2: Configuration

1. Server Configuration (VM1):

**• SSH and SFTP Configuration:**

* Enable SSH access for clients using local account credentials, that is, configure SSHD server such that clients can access any file via their local accounts, using username and password.

sudo nano /etc/ssh/sshd\_config

PasswordAuthentication and set it to yes

A blue text on a black background

Description automatically generated

Since, at the beginning, clients were made to access only sftp connection, while trying to connect localhost via ssh we will be denied.

A computer screen shot of white text

Description automatically generated

to fix this error we modify the permissions in the sshd\_config by removing ForceCommand internal-sftp from the following:

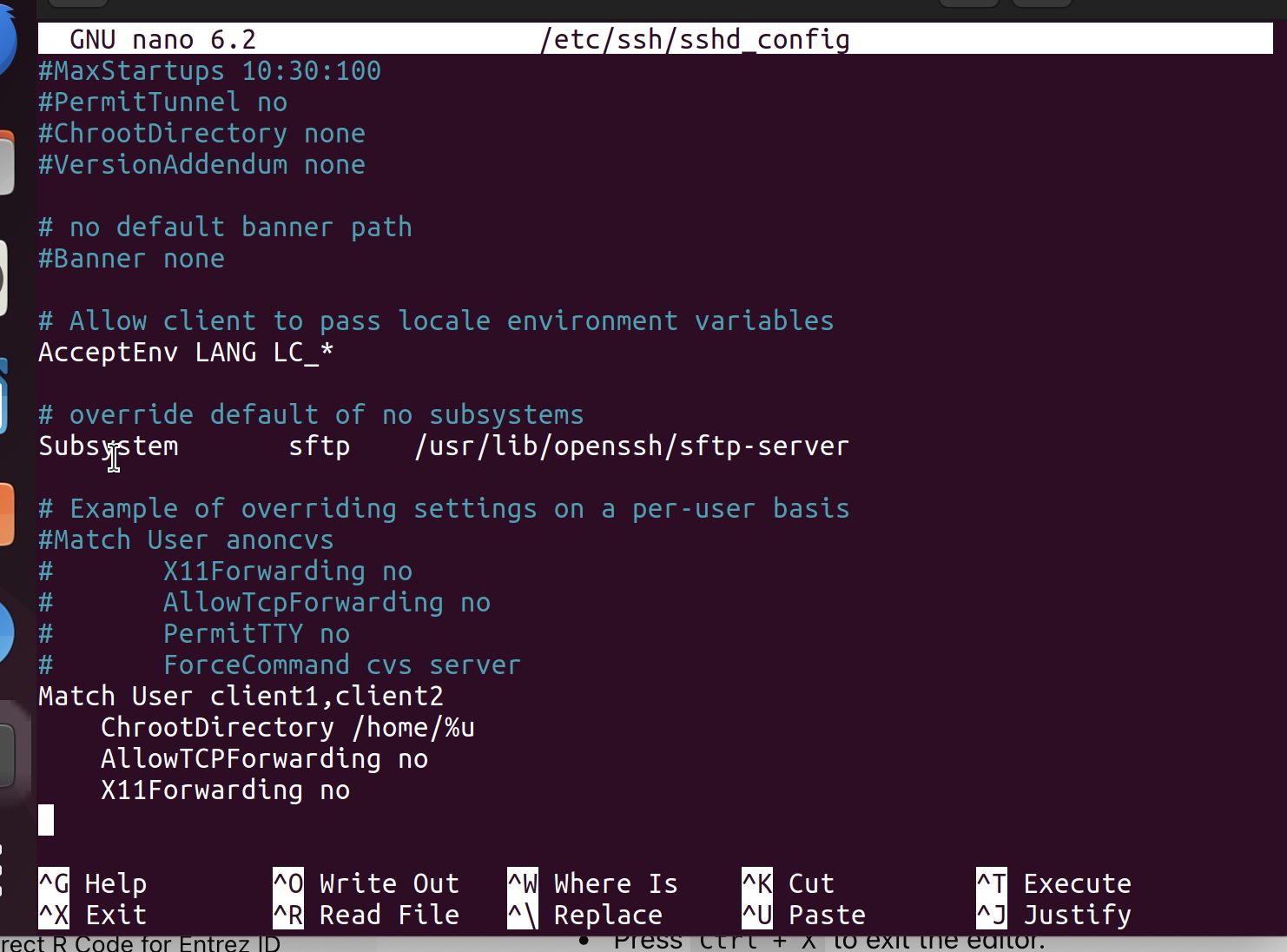
Match User clien1,client2

ForceCommand internal-sftp

ChrootDirectory /home/%u

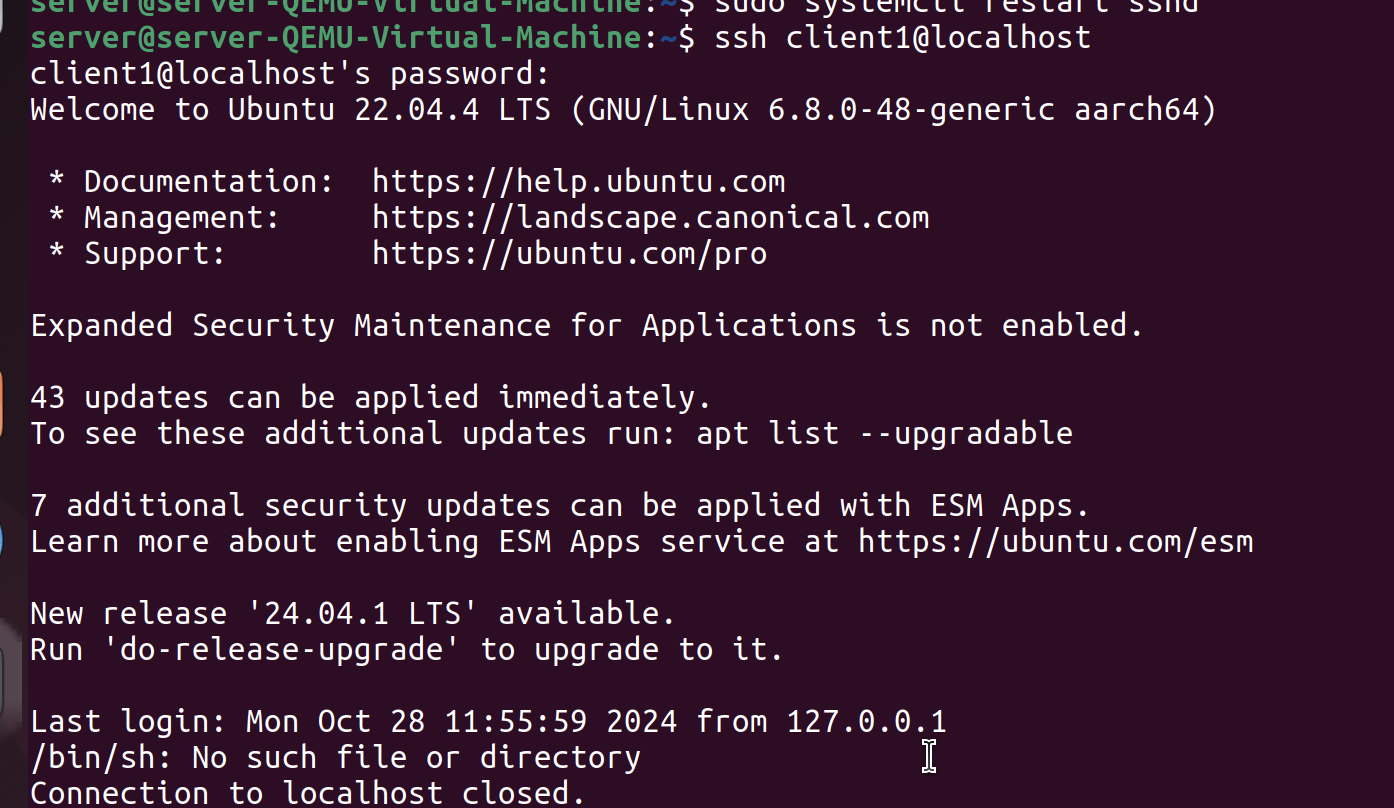
AllowTCPForwarding no

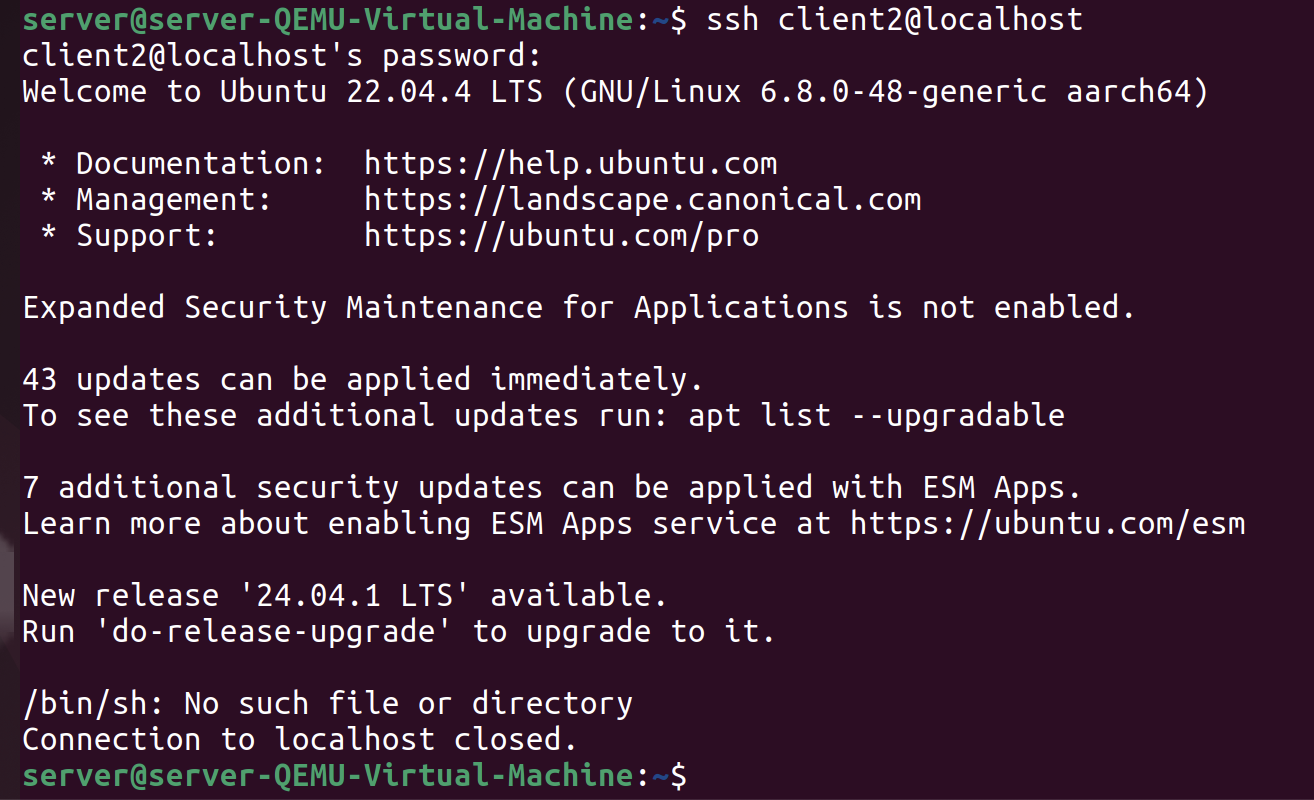
X11Forwarding no



*Removing the whole match user command will ensure clients have acces to both SSH and SFTP*

*Testing:*





*If the client is in different VM, remove firewall from the clients machine using:*

sudo ufw allow ssh

*and then to enable ssh in server from clients use*

ssh username@ip\_address

* Set up SFTP for file uploads/downloads.

**client1:**

sudo chown root:root /home/client1

sudo chmod 755 /home/client1

sudo mkdir /home/client1/filesClient1.txt

**client2:**

sudo chown root:root /home/client2

sudo chmod 755 /home/client2

sudo mkdir /home/client2/filesClient2.txt

* Secure copy will be used to transfer files to and from server.

*Secure copy protocol (SCP) is a command that allows for efficient and secure file transfers between two systems across a network. It employs Secure Shell (SSH) to authenticate systems and encrypt data.sf*

**Syntax:** scp [options] [source username@IP]:/[directory and file name] [destination username@IP]:/[destination directory]

*since I am testing all these things in same local machine, ip address of my clients and server is same*

**to copy file: “fileServer.txt” from server to client1:**

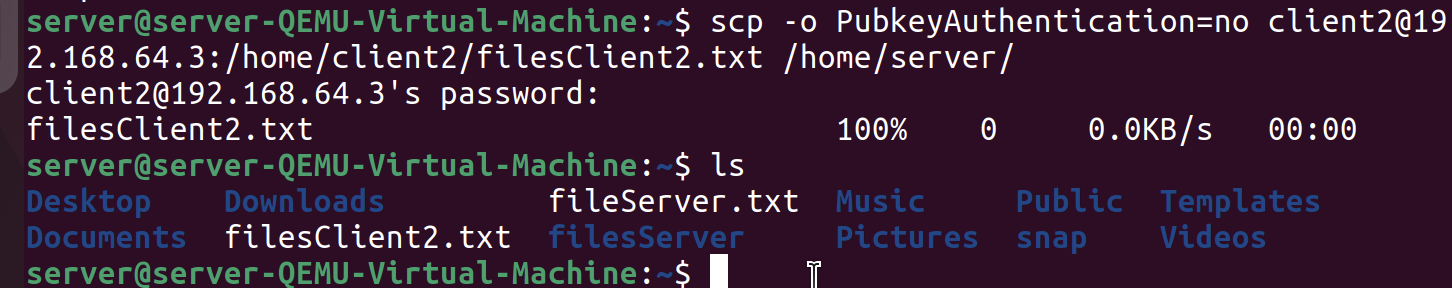
scp fileServer.txt client1@192.168.64.3:/home/client1/

A screen shot of a computer

Description automatically generated

**to copy file: “fileClient2.txt” from client2exit to server:**

scp -o PubkeyAuthentication=no [client2@192.168.64.3:/home/client2/filesClient2.txt /home/server/](mailto:client2@192.168.64.3:/home/client2/filesClient2.txt%20/home/server/)



we used -o so that while pasting in server, it doesn’t show error about denied permission due to public key and password

Task 3: Shell Scripting

**Server Side Shell script 1: (Network.sh)**

**Install network tools:** sudo apt install net-tools

**Enable permission:** chmod u+x Network.sh

**Enable permission:** chmod u+x traceroute.sh

*to install traceroute:* sudo apt install traceroute

**Network.sh**

#!/bin/bash

clear

echo

output\_network="network.log"

> "$output\_network"

server\_ip="192.168.64.3"

client1\_ip="192.168.64.2"

# client2\_ip="192.168.64.2"

echo "Server IP: $server\_ip" >> "$output\_network"

echo "Client1 IP: $client1\_ip" >> "$output\_network"

echo >> "$output\_network"

echo "Pinging from server to clients: " >> "$output\_network"

echo "----------------------------------" >> "$output\_network"

echo >> "$output\_network"

for i in {1..3}

do

echo "Attempt $i:" >> "$output\_network"

echo "====================================" >> "$output\_network"

echo "Pinging from server (VM1) to client1 (VM2)..." >> "$output\_network"

ping -c 1 -W 2 "$client1\_ip" >> "$output\_network" 2>&1

if [ $? -ne 0 ]; then

echo "Ping to client1[$client1\_ip] failed. Running traceroute and diagnostic checks..." >> "$output\_network"

./traceroute.sh >> "$output\_network" 2>&1

date "+%Y-%m-%d %H:%M:%S - Connectivity with client1 [$client1\_ip] failed, running diagnostics." >> "$output\_network"

else

echo >> "$output\_network"

date "+%Y-%m-%d %H:%M:%S - Connectivity with client1 [$client1\_ip] is ok." >> "$output\_network"

fi

echo >> "$output\_network"

# echo "Pinging from server (VM1) to client2 (VM3)..." >> "$output\_network"

# ping -c 1 -W 2 "$client2\_ip" >> "$output\_network" 2>&1

# if [ $? -ne 0 ]; then

# echo "Ping to client2[$client2\_ip] failed. Running traceroute and diagnostic checks..." >> "$output\_network"

# ./traceroute.sh >> "$output\_network" 2>&1ssh username@192.168.64.2

# date "+%Y-%m-%d %H:%M:%S - Connectivity with client2 [$client2\_ip] failed, running diagnostics." >> "$output\_network"

# else

# echo >> "$output\_network"

# date "+%Y-%m-%d %H:%M:%S - Connectivity with client2 [$client2\_ip] is ok." >> "$output\_network"

#fi

# # echo >> "$output\_network"

echo "-------------------------------------------------" >> "$output\_network"

echo >> "$output\_network"

done

echo >> "$output\_network"

echo "Pinging from clients to server: " >> "$output\_network"

echo "----------------------------------" >> "$output\_network"

echo >> "$output\_network"

for i in {1..3}

do

echo "Attempt $i:" >> "$output\_network"

echo "====================================" >> "$output\_network"

echo "Pinging from client1 (VM2) to server (VM1)..." >> "$output\_network"

ssh -i ~/.ssh/id\_rsa -o StrictHostKeyChecking=no samia@$client1\_ip "ping -c 1 -W 2 $server\_ip" >> "$output\_network" 2>&1

ssh\_exit\_status=$?

if [ $ssh\_exit\_status -ne 0 ]; then

echo "Ping to server [$server\_ip] failed. Running traceroute and diagnostic checks..." >> "$output\_network"

ssh -i ~/.ssh/id\_rsa -o StrictHostKeyChecking=no samia@$client1\_ip "./traceroute.sh" >> "$output\_network" 2>&1

date "+%Y-%m-%d %H:%M:%S - Connectivity with server [$server\_ip] failed, running diagnostics." >> "$output\_network"

else

echo >> "$output\_network"

date "+%Y-%m-%d %H:%M:%S - Connectivity with server [$server\_ip] is ok." >> "$output\_network"

fi

echo >> "$output\_network"

# echo "Pinging from client2 (VM3) to server (VM1)..." >> "$output\_network"

# ssh -i ~/.ssh/id\_rsa -o StrictHostKeyChecking=no samia@$client1\_ip "ping -c 1 -W 2 $server\_ip" >> "$output\_network" 2>&1

# ssh\_exit\_status=$?

# if [ $ssh\_exit\_status -ne 0 ]; then

# echo "Ping to server [$server\_ip] failed. Running traceroute and diagnostic checks..." >> "$output\_network"

# ssh -i ~/.ssh/id\_rsa -o StrictHostKeyChecking=no samia@$client1\_ip "./traceroute.sh" >> "$output\_network" 2>&1

# date "+%Y-%m-%d %H:%M:%S - Connectivity with server [$server\_ip] failed, running diagnostics." >> "$output\_network"

# else

# echo >> "$output\_network"

# date "+%Y-%m-%d %H:%M:%S - Connectivity with server [$server\_ip] is ok." >> "$output\_network"

# fi

echo >> "$output\_network"

echo "-------------------------------------------------" >> "$output\_network"

echo >> "$output\_network"

done

echo >> "$output\_network"

echo "Showing results from network.log file......."

cat network.log

**traceroute.sh**

#!/bin/bash

echo "Running traceroute..."

echo

echo "Routing Table:"

echo "==============="

netstat -rn

echo "-----------------------------------------------------------------------------"

echo

echo "Hostname:"

echo "=========="

hostname

echo "-------------------------------------"

echo

echo "Testing local DNS server for google.com:"

echo "========================================="

nslookup google.com

echo "-------------------------------------"

echo

echo "Tracing route to google.com:"

echo "=============================="

traceroute google.com

echo "---------------------------------------------------"

echo

echo "Pinging google.com to check connectivity:"

echo "========================================="

ping -c 4 google.com

echo "--------------------------------------------------------------------"

echo

echo "Traceroute completed."

*traceroute.sh should be in client1 and client2 as well so that while testing pings from clients to server, this file can run.*

scp traceroute.sh [client1@192.168.64.3:/home/client1/](mailto:client1@192.168.64.3:/home/client1/)

scp traceroute.sh [client2@192.168.64.3:/home/client2/](mailto:client2@192.168.64.3:/home/client2/)

**Server Side Shell script 2: (System.sh)**

**Enable permission:** chmod u+x System.sh

System.sh file:

#!/bin/bash

output\_disk="disk\_info.log"

output\_cpu\_mem="mem\_cpu\_info.log"

> "$output\_disk"

> "$output\_cpu\_mem"

#Disk

echo "Disk informations" >> "$output\_disk"

echo "----------------------------------" >> "$output\_disk"

echo >> $output\_disk

echo "Disk space:" >> "$output\_disk"

df -H >> "$output\_disk"

echo "==================================" >> "$output\_disk"

echo >> $output\_disk

echo "Disk usage for HOME:" >> "$output\_disk"

du -sh "$HOME" >> "$output\_disk"

echo "==================================" >> "$output\_disk"

echo >> $output\_disk

echo "Disk usage for Subdirectories:" >> "$output\_disk"

du -hs "$HOME"/\* >> "$output\_disk"

echo >> $output\_disk

#Memory CPU

echo "Memory and CPU informations" >> "$output\_cpu\_mem"

echo "----------------------------------" >> "$output\_cpu\_mem"

echo >> "$output\_cpu\_mem"

echo "Memory Usage (Percentage):" >> "$output\_cpu\_mem"

free | grep Mem | awk '{print $3/$2 \* 100.0, "%"}' >> "$output\_cpu\_mem"

echo "==================================" >> "$output\_cpu\_mem"

echo >> "$output\_cpu\_mem"

echo "Memory Free (Percentage):" >> "$output\_cpu\_mem"

free | grep Mem | awk '{print $4/$2 \* 100.0, "%"}' >> "$output\_cpu\_mem"

echo "==================================" >> "$output\_cpu\_mem"

echo >> "$output\_cpu\_mem"

echo "CPU Model:" >> "$output\_cpu\_mem"

lscpu | grep -i 'architecture\|vendor' >> "$output\_cpu\_mem"

echo "==================================" >> "$output\_cpu\_mem"

echo >> "$output\_cpu\_mem"

echo "CPU Cores:" >> "$output\_cpu\_mem"

nproc >> "$output\_cpu\_mem"

echo "==================================" >> "$output\_cpu\_mem"

echo >> "$output\_cpu\_mem"

#Displaying output from files

echo "Showing output of disk\_info.log..."

echo

cat "$output\_disk"

echo

echo "Showing output of mem\_cpu\_info.log...."

echo

cat "$output\_cpu\_mem"

echo