|  |
| --- |
| **OS Project 1**  **Scripting**  Abdulrahman Selmi 202104994  Aly Soliman 202105591  Ahmed Yousef 202003774  Osama Hardan 202107304 |

Contents

[**Project Contribution Table** 3](#_Toc162138529)

[**Scripts:** 4](#_Toc162138530)

[Config-dns.sh 4](#_Toc162138531)

[Config-mosh.sh 4](#_Toc162138532)

[Config-nginx.sh 5](#_Toc162138533)

[Config-nginx-auth.sh 5](#_Toc162138534)

[Config-site.sh 6](#_Toc162138535)

[Config-sshd.sh 7](#_Toc162138536)

[Create-client.sh 8](#_Toc162138537)

[Server.sh 9](#_Toc162138538)

[Unsuccessful-attempts.sh 11](#_Toc162138539)

[Clients.csv 12](#_Toc162138540)

[Main.sh: 12](#_Toc162138541)

[Verify.sh 13](#_Toc162138542)

[Meshping.sh 14](#_Toc162138543)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Project Contribution Table** | | | | |
|  |  | *Selmi* | *Aly* | *Osama* | *Yousef* |
|  | **Task 1** |  | | | |
| 1 | Setup |  |  |  | x |
|  | **Task 2** |  | | | |
| 2 | Web Server |  |  |  | x |
| 3 | SSH and SFTP |  |  | x |  |
| 4 | DNS |  |  | x |  |
| 5 | Logs |  |  | x |  |
| 6 | Clients Group | x |  |  |  |
| 7 | Log Invalid | x |  |  |  |
| 8 | Handle Excessive | x |  |  |  |
| 9 | Verification |  | x |  |  |
|  | **Task 3** |  | | | |
| 10 | Automation |  |  |  | x |
|  | **Task 4** |  | | | |
| 11 | Ping |  | x |  |  |
| 12 | Traceroute |  | x |  |  |
|  | **Contribution** | 25% | 25% | 25% | 25% |

# **Scripts:**

## Config-dns.sh

|  |
| --- |
| #!/bin/bash  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root."  exit 1  fi  DNS\_SERVER\_1="9.9.9.9"  DNS\_SERVER\_2="149.112.112.112"  sed -i '/nameserver/d' /etc/resolv.conf  echo "nameserver $DNS\_SERVER\_1" | tee -a /etc/resolv.conf > /dev/null  echo "nameserver $DNS\_SERVER\_2" | tee -a /etc/resolv.conf > /dev/null  echo "Quad9 DNS servers configured successfully." |

## Config-mosh.sh

|  |
| --- |
| #!/bin/bash  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root."  exit 1  fi  dnf install mosh -y  tee /etc/firewalld/services/mosh.xml > /dev/null <<EOF  <?xml version="1.0" encoding="utf-8"?>  <service>  <short>mosh</short>  <description>Mosh service</description>  <port protocol="udp" port="60000"/>  <port protocol="udp" port="60001"/>  <port protocol="udp" port="60002"/>  </service>  EOF  firewall-cmd --permanent --add-service=mosh  firewall-cmd --reload  echo "Mosh configured successfully." |

## Config-nginx.sh

|  |
| --- |
| #!/bin/bash  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root" >&2  exit 1  fi  dnf install -y nginx > /dev/null  systemctl enable nginx  systemctl start nginx  firewall-cmd --permanent --zone=public --add-service=http  firewall-cmd --reload  echo "NGINX service status:"  systemctl status nginx  curl -I http://localhost |

## Config-nginx-auth.sh

|  |
| --- |
| #!/bin/bash  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root" >&2  exit 1  fi  echo "Installing HTTPD Tools..."  dnf install -y httpd-tools > /dev/null  echo "Creating .htpasswd file..."  htpasswd -cb /etc/nginx/.htpasswd admin admin  while IFS=, read -r username fullname  do  htpasswd -b /etc/nginx/.htpasswd "$username" "$username"  done < ../clients.csv  cat > /etc/nginx/conf.d/default.conf <<EOF  server {  listen 80;  server\_name localhost;  location / {  root /usr/share/nginx/html;  index index.html index.htm;  auth\_basic "Needs Authentication";  auth\_basic\_user\_file /etc/nginx/.htpasswd;  }  }  EOF  cat > /usr/share/nginx/html/index.html <<EOF  <!DOCTYPE html>  <html>  <head>  <title>CMPS405 Lab</title>  </head>  <body style="display:grid;place-items:center;">  <h1 style="font-size:4rem;">Welcome to Operating Systems Lab</h1>  </body>  </html>  EOF  systemctl restart nginx  echo "NGINX service status:"  systemctl status nginx  curl -I http://localhost |

## Config-site.sh

|  |
| --- |
| #!/bin/bash  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root."  exit 1  fi  if [ $# -ne 1 ]; then  echo "Usage: $0 <username>"  exit 1  fi  username="$1"  user\_home="/home/$username"  if [ ! -d "$user\_home" ]; then  echo "Error: Home directory for user $username does not exist."  exit 1  fi  site\_directory="$user\_home/site"  nginx\_html="/usr/share/nginx/html/$username"  if [ ! -d "$site\_directory" ]; then  mkdir -p "$site\_directory"  fi  ln -sfn "$site\_directory" "$nginx\_html"  chown -R nginx:nginx "$site\_directory"  chmod -R 755 "$site\_directory"  chmod o+rx "$user\_home"  echo "Website directory for user $username configured successfully." |

## Config-sshd.sh

|  |
| --- |
| #!/bin/bash  configure\_ssh\_server() {  systemctl enable sshd  systemctl start sshd    # check if the line exists in the file  grep -q "AllowGroups clients" /etc/ssh/sshd\_config || echo "AllowGroups clients" >> /etc/ssh/sshd\_config    systemctl restart sshd  }  configure\_firewall() {  firewall-cmd --permanent --zone=public --add-service=ssh  firewall-cmd --reload  }  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root" >&2  exit 1  fi  echo "Installing OpenSSH server..."  dnf install -y openssh-server > /dev/null  echo "Configuring OpenSSH server..."  configure\_ssh\_server  echo "Adding firewall rules..."  configure\_firewall  echo "SSH service status:"  systemctl status sshd |

## Create-client.sh

|  |
| --- |
| #!/bin/bash  create\_user() {  local username=$1  local full\_name=$2  local home\_dir="/home/$username"    if id "$username" &>/dev/null; then  echo "User $username already exists."  exit 1  fi    useradd -m -c "$full\_name" -s /bin/bash -d "$home\_dir" "$username"    # local password=$(pwgen -s 12)  local password=$username    echo -e "$password\n$password" | passwd "$username"    if ! getent group clients > /dev/null 2>&1; then  groupadd clients  fi  usermod -aG clients "$username"    if ! grep -q "^$username" /etc/sudoers; then  echo "$username ALL=(ALL:ALL) ALL" >> /etc/sudoers  fi    echo "User $username created with password: $password"  }  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root" >&2  exit 1  fi  if [ $# -ne 2 ]; then  echo "Usage: $0 <username> <full name>"  exit 1  fi  # dnf install -y pwgen > /dev/null 2>&1  create\_user "$1" "$2" |

## Server.sh

|  |
| --- |
| #!/bin/bash  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root" >&2  exit 1  fi  echo "Setting up needed things for the server..."  echo  echo "Updating the system..."  dnf update -y  echo;echo  echo -n "Do you want to install MATE Desktop? (y/n) "  read answer  if [ "$answer" == "y" ] || [ "$answer" == "Y" ] ;then  echo "Installing MATE Desktop..."  dnf groupinstall -y "MATE Desktop"  systemctl set-default graphical.target  else  echo "MATE Desktop will not be installed."  fi  echo;echo  echo "Setting up server..."  chmod +x ./\*.sh  echo  echo "Setting up SSH..."  ./config-sshd.sh  echo;echo  echo "Setting up NGINX Webserver..."  ./config-nginx.sh  echo;echo  echo "Configuring Quad9 DNS..."  ./config-dns.sh  echo;echo  echo "Setting up Mobile Shell..."  ./config-mosh.sh  echo;echo  echo "Creating clients and configuring client's website from clients.csv..."  while IFS=, read -r username fullname  do  ./create-client.sh "$username" "$fullname"  ./config-site.sh "$username"  done < ../clients.csv  echo;echo  echo "Setting up NGINX authentication..."  ./config-nginx-auth.sh  echo;echo  echo "Setting up consolidating logs..."  ./unsuccessful-attempts.sh &  echo;echo  echo "Please reboot if major updates happened (e.g. installed MATE)." |

## Unsuccessful-attempts.sh

|  |
| --- |
| #!/bin/bash  if [ "$(id -u)" -ne 0 ]; then  echo "This script must be run as root" >&2  exit 1  fi  dnf install -y cronie > /dev/null  systemctl enable crond  systemctl start crond  mkdir -p "/home/server/log"  LOG="/home/server/log/unsuccessful\_attempts.log"  touch "$LOG"  consolidate\_logs() {  for user in $(ls /home); do  if [ -f "/home/$user/invalid\_attempts.log" ]; then  cat "/home/$user/invalid\_attempts.log" >> "$LOG"  fi  done  }  cleanup\_logs() {  if [ $(find "$LOG" -type f -mtime +7) ];then  rm -f "$LOG"  else  echo "No logs to clean up."  fi  }  schedule\_cleanup() {  (crontab -l ; echo "0 0 \* \* \* /home/server/scripting/server/unsuccessful-attempts.sh") | crontab -  }  consolidate\_logs  cleanup\_logs  schedule\_cleanup |

## Clients.csv

|  |
| --- |
| client1,Client 1  client2,Client 2 |

## Main.sh:

|  |
| --- |
| #!/bin/bash  clients\_group() {  if getent group clients > /dev/null; then  echo clients group exists  if groups $USER | grep -q "clients"; then  echo $USER is part of clients group  echo Proceeding with script execution...  else  echo $USER is not part of clients group  sudo usermod -aG clients $USER  fi  else  sudo addgroup clients  sudo usermod -aG clients $USER  fi  }  connect\_to\_server() {  # read -p "Enter Username: " username  ssh $USER@$SERVER\_IP 2> ./err  code=$?  if cat ./err | grep -q "No route to host"; then  echo "Connection Failed. Server may be down." >&2  exit 1  fi  }  log\_invalid\_attempt() {  # Print p timestamp t and user u  echo $(date +"%Y-%m-%d %H:%M:%S") - $USER: $1 >> ~/invalid\_attempts.log  }  handle\_excessive\_invalid\_attempts() {  echo "Unauthorized user!"  rsync ~/invalid\_attempts.log $USER@$SERVER\_IP:/home/$USER  # gnome-session-quit --no-prompt  }  main() {  if [ "root" != $USER ]; then  clients\_group  connect\_to\_server  if [ $code -eq 255 ]; then  log\_invalid\_attempt "SSH Login failed. Invalid password"  handle\_excessive\_invalid\_attempts  fi  else  echo Please run script using normal user privileges  exit 1  fi  }  if [ $# -ne 1 ]; then  echo "Please provide server IP" >&2  exit 1  else  SERVER\_IP=$1  main  fi |

## Verify.sh

|  |
| --- |
| #!/bin/bash  username="techuser"  group="admins"  path="/home/client1/main.sh"  verify() {  # If techuser exits delete it  if id $username &> /dev/null; then  sudo deluser $username  sudo rm -r /home/$username  fi  # Create techuser  sudo useradd -m $username  echo "$username:techuser" | sudo chpasswd &> /dev/null  $ Run main.sh under techuser  sudo -u $username $path  # If admins group exists delete it  if getent group $group &> /dev/null; then  sudo delgroup $group  fi  # If admins group in sudoers delete it  if sudo grep -qE "^%$group" "/etc/sudoers"; then  sudo sed -i "/^%$group/d" "/etc/sudoers"  fi  # Create admins group  sudo groupadd admins  # Add admins group to sudoers  echo "%admins ALL=(ALL) ALL" | sudo tee -a /etc/sudoers  # Add techuser to admins group  sudo usermod -aG $group $username  # Run main.sh udner techuser  sudo -u $username $path  }  verify |

## Meshping.sh

|  |
| --- |
| #!/bin/bash  log() {  msg="$(date +"%Y-%m-%d %H:%M:%S") $1"  echo $msg  echo $msg >> ~/network.log  }  echo  for ip in $@; do  ping -c 2 -W 4 $ip  if [ $? -ne 0 ]; then  log "Unable to ping $ip. Trying traceroute..."  traceroute $ip  if [ $? -ne 0 ]; then  log "Unable to traceroute $ip. Rebooting target machine..."  ssh reboot@$ip "echo reboot | sudo -S reboot"  fi  fi  log "Connectivity with $ip is ok"  echo  done |