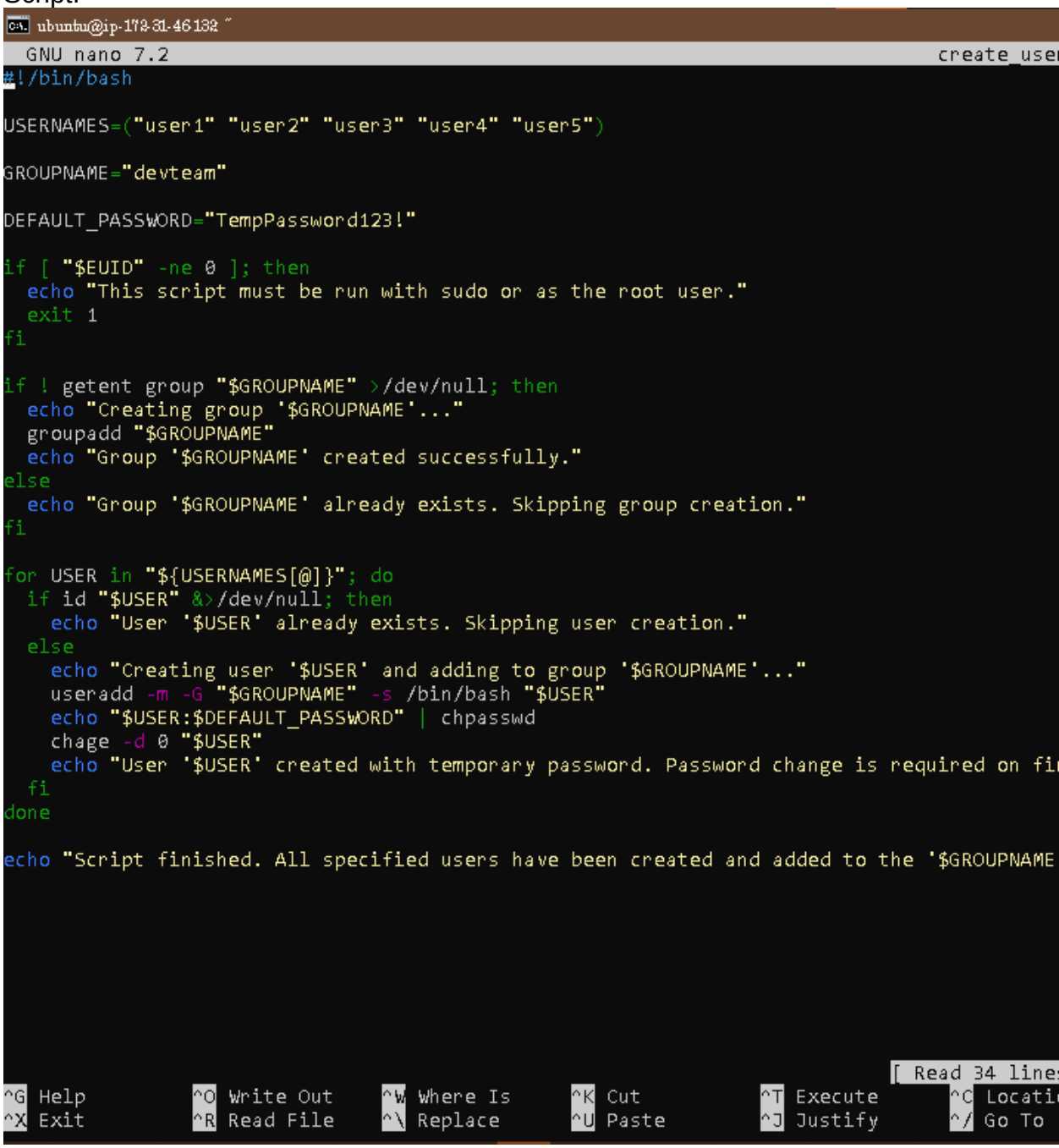
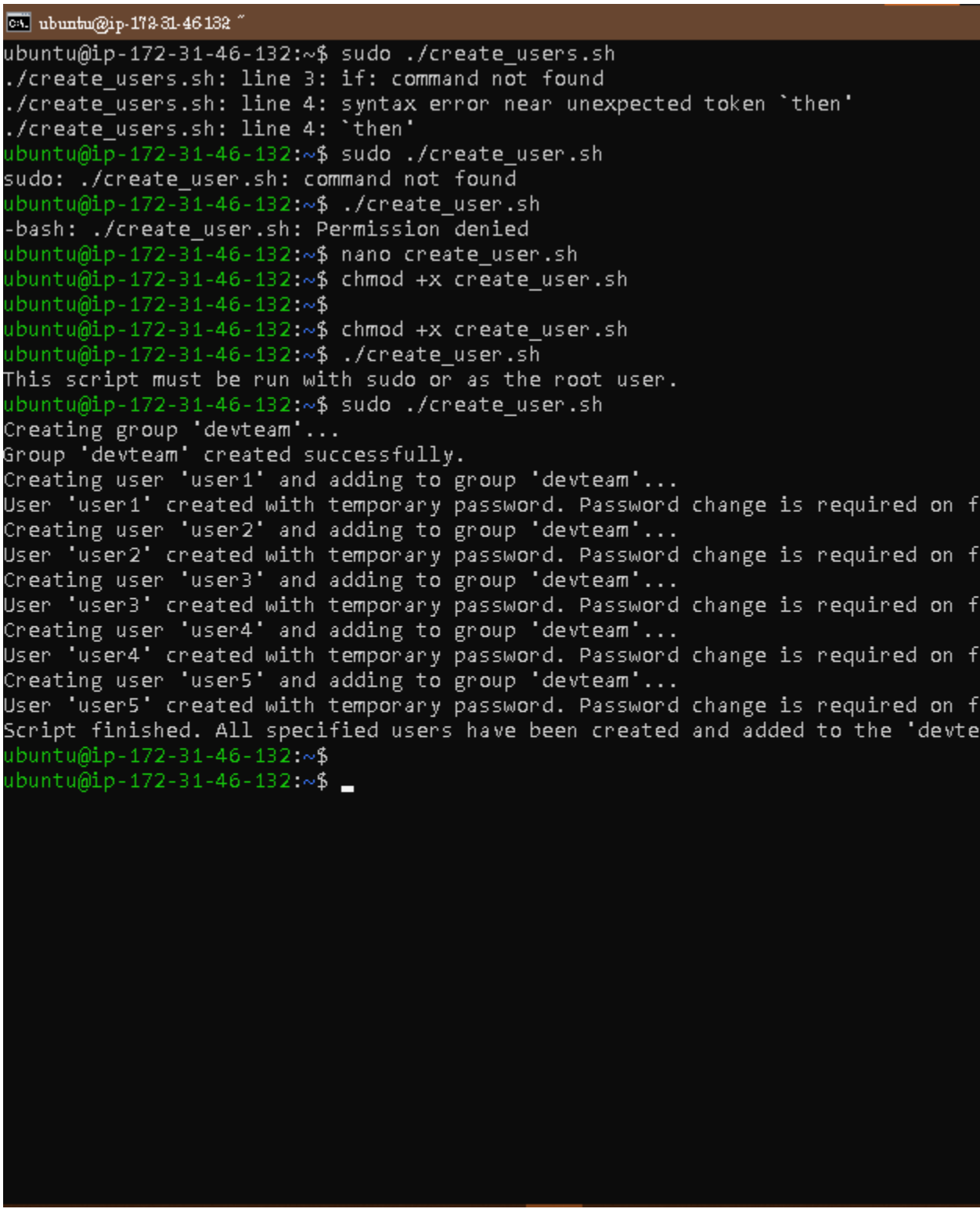
Linux Admininstration Works Scripts and Solution

1. _User &GroupManagementAutomation
    a. Script:



b. Screenshot of Solution



:

2. . File Permissions&ACLs Project

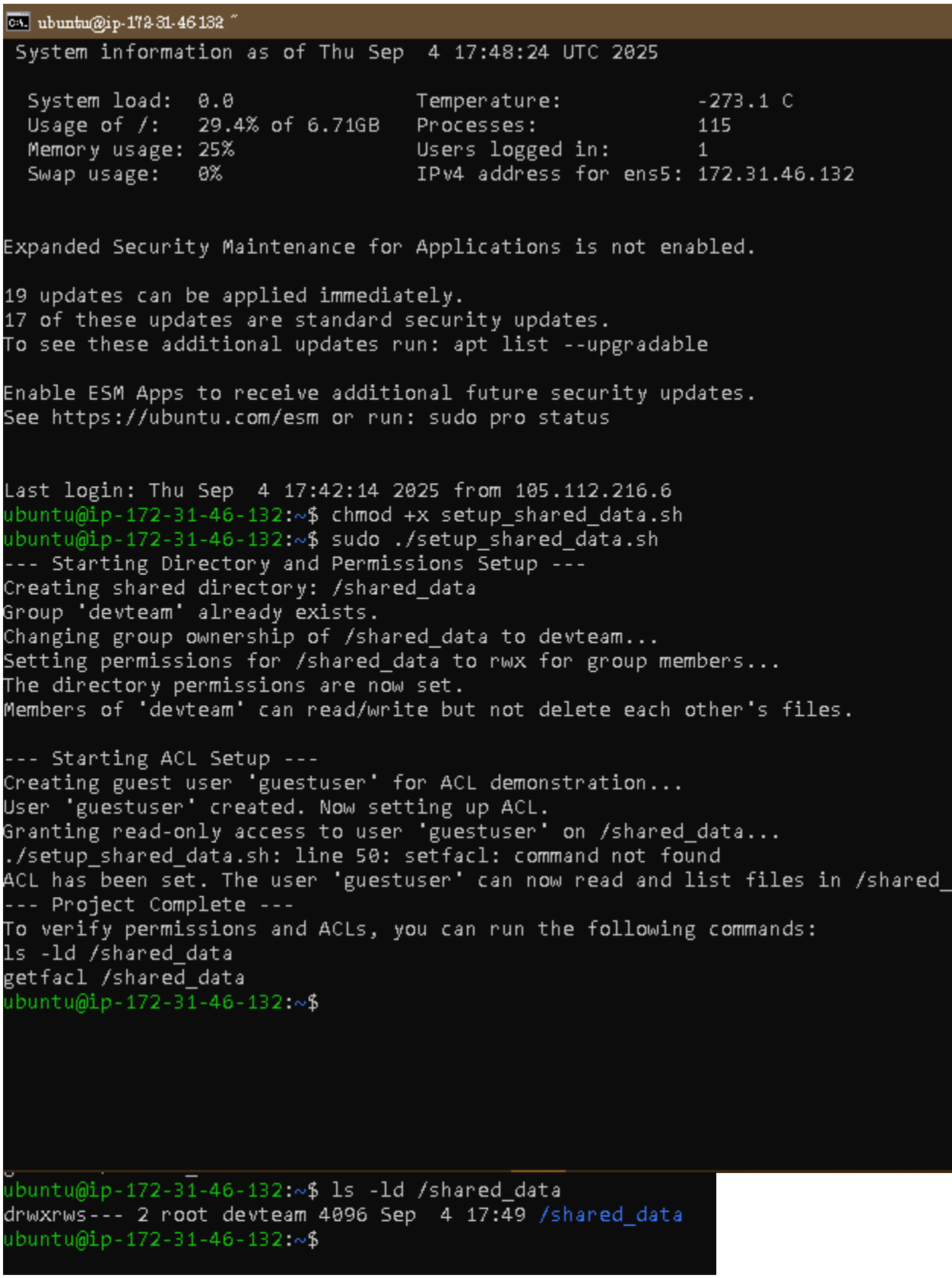
a. Script

: #!/bin/bash

```
SHARED_DIR="/shared_data"
GROUP_NAME="devteam"
GUEST_USER="guestuser"

if [ "$EUID" -ne 0 ]; then
  echo "This script must be run with sudo or as the root user."
  exit 1
fi

echo "--- Starting Directory and Permissions Setup ---"

if [ ! -d "$SHARED_DIR" ]; then
  echo "Creating shared directory: $SHARED_DIR"
  mkdir -p "$SHARED_DIR"
else
  echo "Shared directory $SHARED_DIR already exists."
fi

if ! getent group "$GROUP_NAME" >/dev/null; then
  echo "Creating group '$GROUP_NAME'..."
  groupadd "$GROUP_NAME"
  echo "Group '$GROUP_NAME' created successfully."
else
  echo "Group '$GROUP_NAME' already exists."
fi

echo "Changing group ownership of $SHARED_DIR to $GROUP_NAME..."
chgrp "$GROUP_NAME" "$SHARED_DIR"

echo "Setting permissions for $SHARED_DIR to rwx for group members..."
chmod 2770 "$SHARED_DIR"

echo "The directory permissions are now set. "
echo "Members of '$GROUP_NAME' can read/write but not delete each other's
files."

echo ""
echo "--- Starting ACL Setup ---"

if ! id "$GUEST_USER" &>/dev/null; then
  echo "Creating guest user '$GUEST_USER' for ACL demonstration..."
  useradd -m "$GUEST_USER"
  echo "User '$GUEST_USER' created. Now setting up ACL."
else
  echo "User '$GUEST_USER' already exists. Skipping user creation."
```

```
fi

echo "Granting read-only access to user '$GUEST_USER' on $SHARED_DIR..."
setfacl -m u:"$GUEST_USER":r-x "$SHARED_DIR"

echo "ACL has been set. The user '$GUEST_USER' can now read and list files in $SHARED_DIR."
echo ""
echo "--- Project Complete ---"
echo "To verify permissions and ACLs, you can run the following commands:"
echo "ls -ld $SHARED_DIR"
echo "getfacl $SHARED_DIR"
```

Solution:

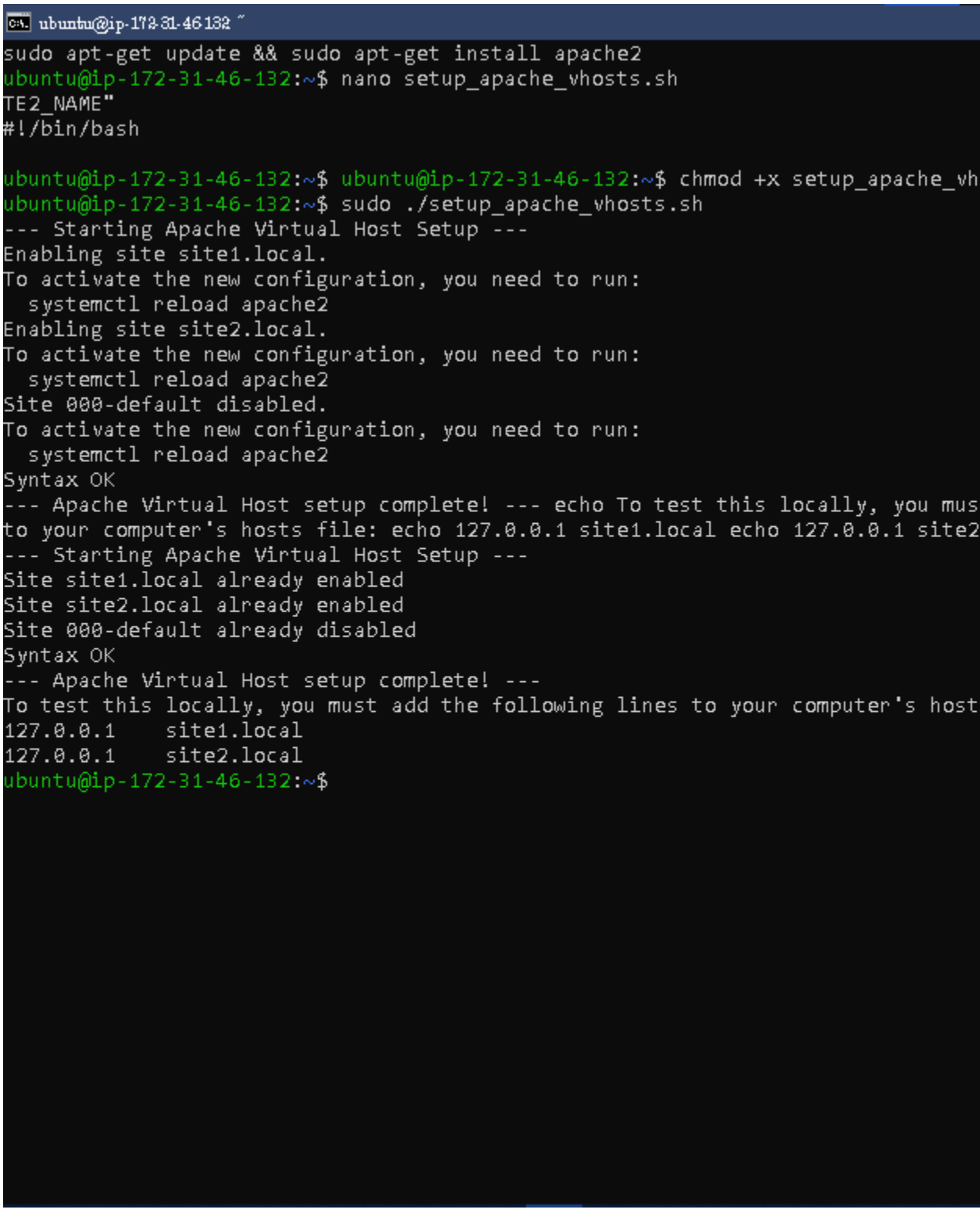

ubuntu@ip-172-31-46-132
System information as of Thu Sep 4 17:48:24 UTC 2025
System load: 0.0
Usage of /: 29.4% of 6.71GB
Memory usage: 25%
Swap usage: 0%
Temperature: -273.1 C
Processes: 115
Users logged in: 1
IPv4 address for ens5: 172.31.46.132
Expanded Security Maintenance for Applications is not enabled.
19 updates can be applied immediately.
17 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
Last login: Thu Sep 4 17:42:14 2025 from 105.112.216.6
ubuntu@ip-172-31-46-132:~$ chmod +x setup_shared_data.sh
ubuntu@ip-172-31-46-132:~$ sudo ./setup_shared_data.sh
--- Starting Directory and Permissions Setup ---
Creating shared directory: /shared_data
Group 'devteam' already exists.
Changing group ownership of /shared_data to devteam...
Setting permissions for /shared_data to rwx for group members...
The directory permissions are now set.
Members of 'devteam' can read/write but not delete each other's files.
--- Starting ACL Setup ---
Creating guest user 'guestuser' for ACL demonstration...
User 'guestuser' created. Now setting up ACL.
Granting read-only access to user 'guestuser' on /shared_data...
./setup_shared_data.sh: line 50: setfacl: command not found
ACL has been set. The user 'guestuser' can now read and list files in /shared_
--- Project Complete ---
To verify permissions and ACLs, you can run the following commands:
ls -ld /shared_data
getfacl /shared_data
ubuntu@ip-172-31-46-132:~$
ubuntu@ip-172-31-46-132:~$ ls -ld /shared_data
drwxrws--- 2 root devteam 4096 Sep 4 17:49 /shared_data
ubuntu@ip-172-31-46-132:~$

3. ApacheVirtualHosts Setup
Script: #!/bin/bash

```
SITE1_NAME="site1.local"
SITE2_NAME="site2.local"
SITE_ROOT="/var/www"
APACHE_CONF_DIR="/etc/apache2/sites-available"

if [[ "$EUID" -ne 0 ]]; then
  echo "This script must be run with sudo or as the root user."
  exit 1
fi

if ! command -v apache2 &> /dev/null; then
  echo "Apache is not installed. Please run the following command to
install it:"
  echo "sudo apt-get update && sudo apt-get install apache2"
  exit 1
fi

echo "--- Starting Apache Virtual Host Setup ---"

mkdir -p "$SITE_ROOT/$SITE1_NAME/public_html"
mkdir -p "$SITE_ROOT/$SITE1_NAME/logs"

mkdir -p "$SITE_ROOT/$SITE2_NAME/public_html"
mkdir -p "$SITE_ROOT/$SITE2_NAME/logs"

echo "<html><body><h1>Welcome to
$SITE1_NAME</h1></body></html>" >
"$SITE_ROOT/$SITE1_NAME/public_html/index.html"

echo "<html><body><h1>Welcome to
$SITE2_NAME</h1></body></html>" >
"$SITE_ROOT/$SITE2_NAME/public_html/index.html"

cat > "$APACHE_CONF_DIR/$SITE1_NAME.conf" << EOF
<VirtualHost *:80>
    ServerName $SITE1_NAME
    ServerAlias www.$SITE1_NAME
    DocumentRoot $SITE_ROOT/$SITE1_NAME/public_html
    ErrorLog $SITE_ROOT/$SITE1_NAME/logs/error.log
    CustomLog $SITE_ROOT/$SITE1_NAME/logs/access.log combined
    <Directory "$SITE_ROOT/$SITE1_NAME/public_html">
        Options Indexes FollowSymLinks
```

```
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
EOF

cat > "$APACHE_CONF_DIR/$SITE2_NAME.conf" << EOF
<VirtualHost *:80>
    ServerName $SITE2_NAME
    ServerAlias www.$SITE2_NAME
    DocumentRoot $SITE_ROOT/$SITE2_NAME/public_html
    ErrorLog $SITE_ROOT/$SITE2_NAME/logs/error.log
    CustomLog $SITE_ROOT/$SITE2_NAME/logs/access.log combined
    <Directory "$SITE_ROOT/$SITE2_NAME/public_html">
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
EOF

a2ensite "$SITE1_NAME.conf"
a2ensite "$SITE2_NAME.conf"

a2dissite 000-default.conf

apache2ctl configtest
systemctl reload apache2

echo "--- Apache Virtual Host setup complete! ---"
echo "To test this locally, you must add the following lines to your
computer's hosts file:"
echo "127.0.0.1    $SITE1_NAME"
echo "127.0.0.1    $SITE2_NAME"
```

Solution:


ubuntu@ip-172-31-46-132
sudo apt-get update && sudo apt-get install apache2
ubuntu@ip-172-31-46-132:~$ nano setup_apache_vhosts.sh
TE2_NAME"
#!/bin/bash
ubuntu@ip-172-31-46-132:~$ ubuntu@ip-172-31-46-132:~$ chmod +x setup_apache_vh
ubuntu@ip-172-31-46-132:~$ sudo ./setup_apache_vhosts.sh
--- Starting Apache Virtual Host Setup ---
Enabling site site1.local.
To activate the new configuration, you need to run:
systemctl reload apache2
Enabling site site2.local.
To activate the new configuration, you need to run:
systemctl reload apache2
Site 000-default disabled.
To activate the new configuration, you need to run:
systemctl reload apache2
Syntax OK
--- Apache Virtual Host setup complete! --- echo To test this locally, you mus
to your computer's hosts file: echo 127.0.0.1 site1.local echo 127.0.0.1 site2
--- Starting Apache Virtual Host Setup ---
Site site1.local already enabled
Site site2.local already enabled
Site 000-default already disabled
Syntax OK
--- Apache Virtual Host setup complete! ---
To test this locally, you must add the following lines to your computer's host
127.0.0.1 site1.local
127.0.0.1 site2.local
ubuntu@ip-172-31-46-132:~$

4. SSL/TLS Implementation:

Script : #!/bin/bash

```
SITE_NAME="site1.local"
SSL_CERT_DIR="/etc/ssl/certs"
SSL_KEY_DIR="/etc/ssl/private"
APACHE_CONF_DIR="/etc/apache2/sites-available"

if [[ "$EUID" -ne 0 ]]; then
  echo "This script must be run with sudo or as the root user."
  exit 1
fi

if ! command -v apache2 &> /dev/null; then
  echo "Apache is not installed. Please run the following command to install it:"
  echo "sudo apt-get update && sudo apt-get install apache2"
  exit 1
fi

if ! command -v openssl &> /dev/null; then
  echo "OpenSSL is not installed. Please run the following command to install it:"
  echo "sudo apt-get update && sudo apt-get install openssl"
  exit 1
fi

echo "--- Starting SSL Virtual Host Setup ---"

mkdir -p "$SSL_CERT_DIR"
mkdir -p "$SSL_KEY_DIR"
```

```
echo "Generating self-signed SSL certificate..."

echo -e "NG\nLagos\nLagos\n\n\n$SITE_NAME\n\n\n" | openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout "$SSL_KEY_DIR/server.key" -out "$SSL_CERT_DIR/server.crt" &> /dev/null


echo "Creating HTTPS Virtual Host configuration for $SITE_NAME..."

cat > "$APACHE_CONF_DIR/$SITE_NAME-ssl.conf" << EOF

<VirtualHost *:443>

   ServerName $SITE_NAME

   ServerAlias www.$SITE_NAME

   DocumentRoot /var/www/$SITE_NAME/public_html

   ErrorLog /var/www/$SITE_NAME/logs/error.log

   CustomLog /var/www/$SITE_NAME/logs/access.log combined

   SSLEngine On

   SSLCertificateFile $SSL_CERT_DIR/server.crt

   SSLCertificateKeyFile $SSL_KEY_DIR/server.key

   <Directory "/var/www/$SITE_NAME/public_html">

      Options Indexes FollowSymLinks

      AllowOverride All

      Require all granted

   </Directory>

</VirtualHost>

EOF


echo "Enabling the SSL module..."

a2enmod ssl &> /dev/null


echo "Enabling the new HTTPS virtual host..."

a2ensite "$SITE_NAME-ssl.conf" &> /dev/null
```

echo "Testing Apache configuration for syntax errors..."

apache2ctl configtest

echo "Reloading Apache to apply changes..."

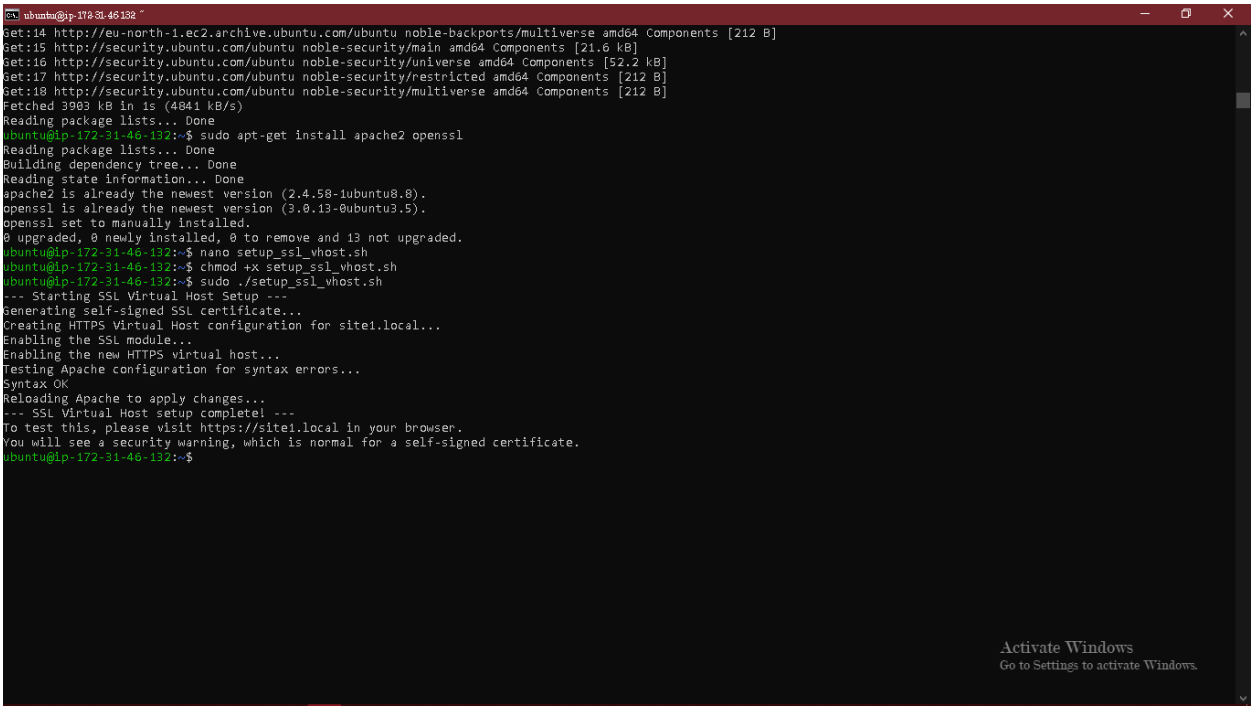
systemctl reload apache2

echo "--- SSL Virtual Host setup complete! ---"

echo "To test this, please visit https://$SITE_NAME in your browser."

echo "You will see a security warning, which is normal for a self-signed certificate."

Solution:



5. MySQLRemoteAccess&Security:

Script: #!/bin/bash

```
DB_USER="remote_user"
DB_PASS="StrongPassword123!"
DB_NAME="remote_db"

if [[ "$EUID" -ne 0 ]]; then
  echo "This script must be run with sudo or as the root user."
  exit 1
fi

if ! command -v mysql &> /dev/null; then
  echo "MySQL is not installed. Please run the following command to install it:"
  echo "sudo apt-get update && sudo apt-get install mysql-server"
  exit 1
fi

echo "--- Starting MySQL Remote Access Configuration ---"

sudo sed -i 's/bind-address = 127.0.0.1/bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf

sudo systemctl restart mysql

mysql -e "CREATE DATABASE IF NOT EXISTS $DB_NAME;"
mysql -e "CREATE USER IF NOT EXISTS '$DB_USER'@'%' IDENTIFIED BY '$DB_PASS';"
mysql -e "GRANT SELECT, INSERT, UPDATE, DELETE ON $DB_NAME.* TO '$DB_USER'@'%';"
mysql -e "FLUSH PRIVILEGES;"

echo "--- MySQL remote access setup complete! ---"
echo "Database '$DB_NAME' and user '$DB_USER' have been created."
```

echo "Remote connections are now enabled for this user from any host."

Solution :

```
ubuntu@ip-172-31-46-132:~$ nano setup_mysql_remote.sh
ubuntu@ip-172-31-46-132:~$ chmod +x setup_mysql_remote.sh
ubuntu@ip-172-31-46-132:~$ sudo ./setup_mysql_remote.sh
--- Starting MySQL Remote Access Configuration ---
--- MySQL remote access setup complete! ---
Database 'remote_db' and user 'remote_user' have been created.
Remote connections are now enabled for this user from any host.
ubuntu@ip-172-31-46-132:~$
```

6. Firewall Configuration:

Script: #!/bin/bash

ALLOWED_IP_RANGE="192.168.1.0/24"

if [["$EUID" -ne 0]]; then

  echo "This script must be run with sudo or as the root user."

  exit 1

fi

if ! command -v ufw &> /dev/null; then

  echo "ufw is not installed. Please run the following command to install it:"

  echo "sudo apt-get update && sudo apt-get install ufw"

  exit 1

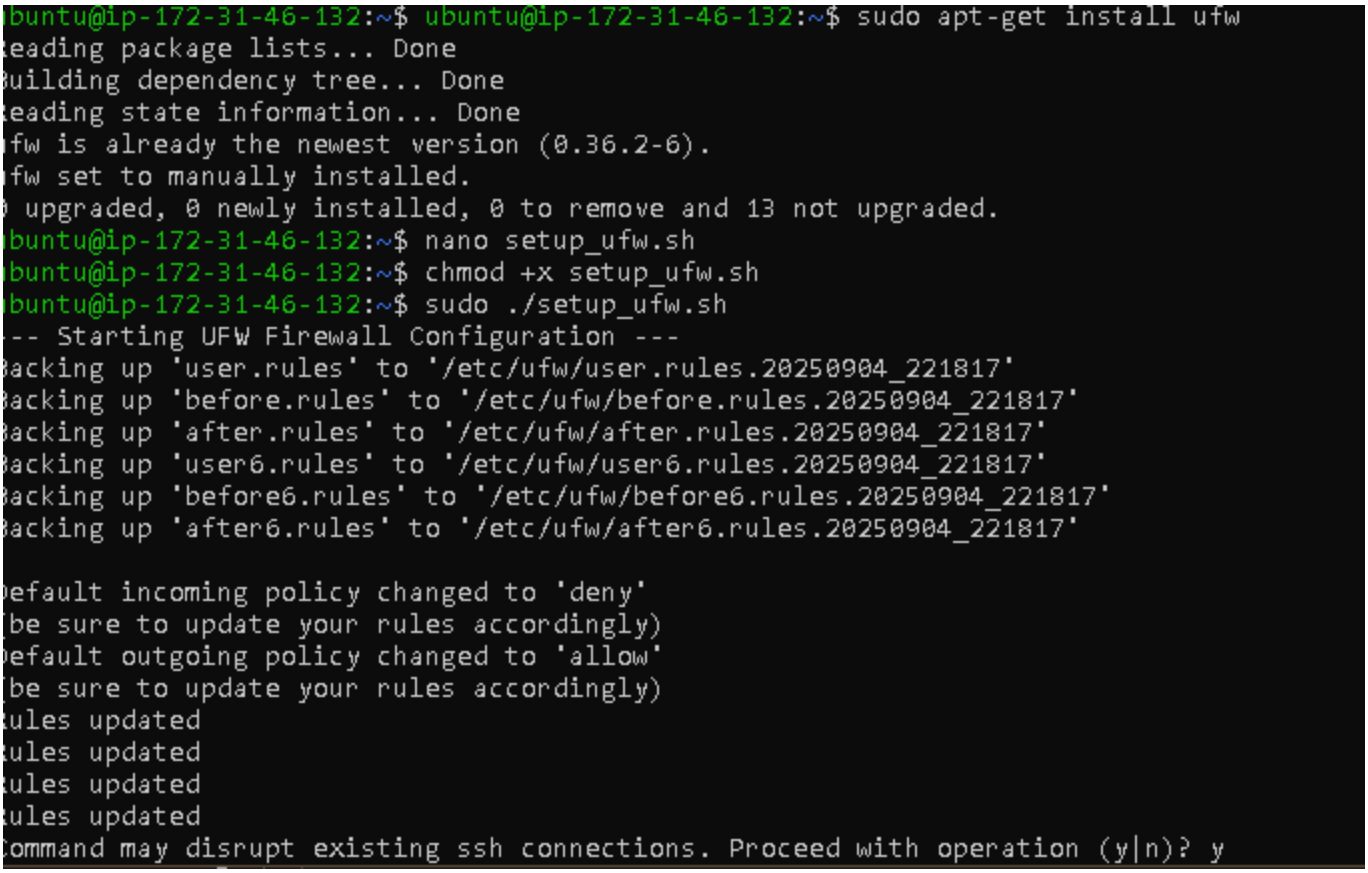
fi

echo "--- Starting UFW Firewall Configuration ---"

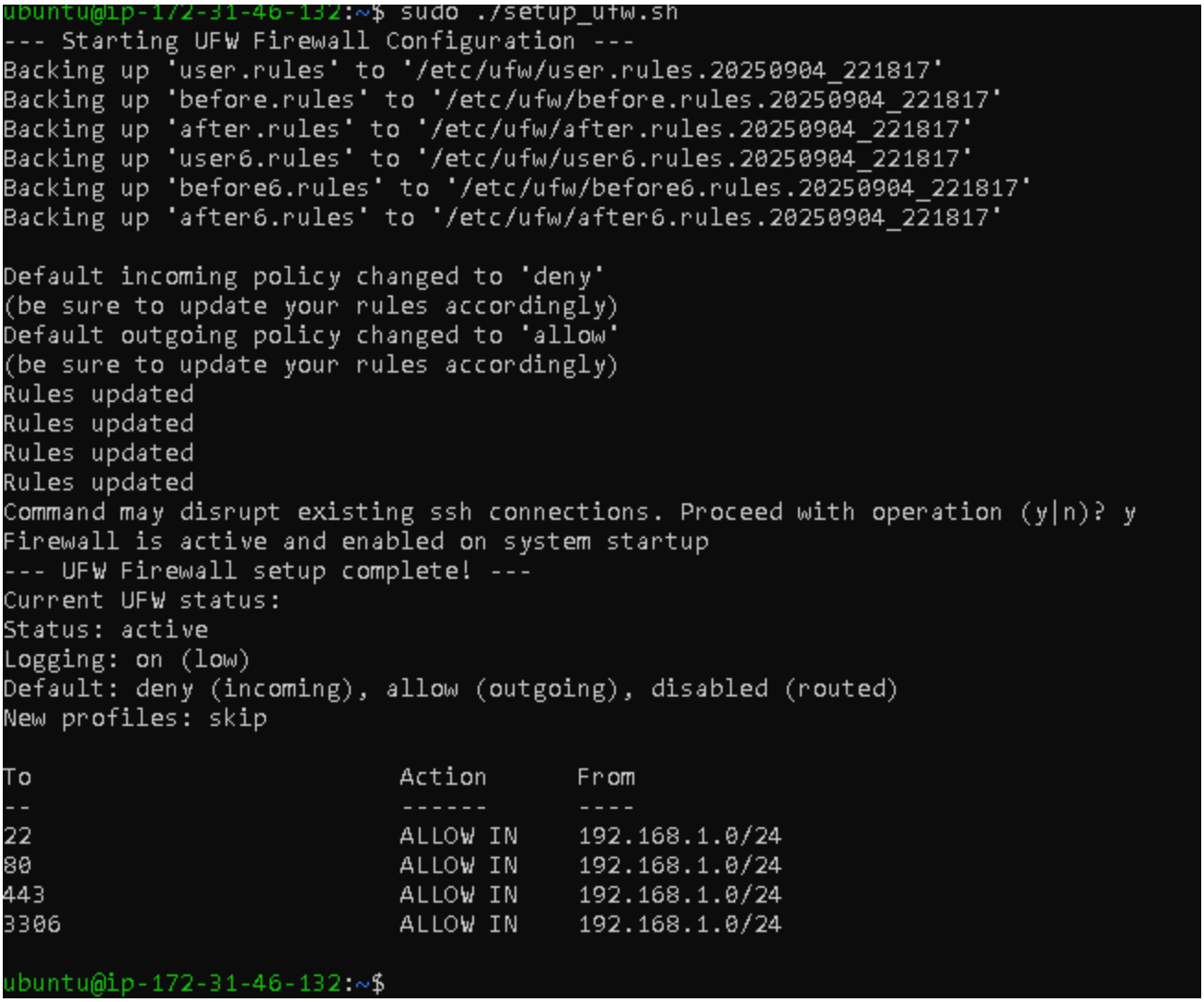
ufw --force reset

```
ufw default deny incoming
ufw default allow outgoing

ufw allow from $ALLOWED_IP_RANGE to any port 22
ufw allow from $ALLOWED_IP_RANGE to any port 80
ufw allow from $ALLOWED_IP_RANGE to any port 443
ufw allow from $ALLOWED_IP_RANGE to any port 3306

ufw enable

echo "--- UFW Firewall setup complete! ---"
echo "Current UFW status:"
ufw status verbose
```

Solution:


buntu@ip-172-31-46-132:~$ ubuntu@ip-172-31-46-132:~$ sudo apt-get install ufw
eading package lists... Done
uilding dependency tree... Done
eading state information... Done
fw is already the newest version (0.36.2-6).
fw set to manually installed.
upgraded, 0 newly installed, 0 to remove and 13 not upgraded.
buntu@ip-172-31-46-132:~$ nano setup_ufw.sh
buntu@ip-172-31-46-132:~$ chmod +x setup_ufw.sh
buntu@ip-172-31-46-132:~$ sudo ./setup_ufw.sh
-- Starting UFW Firewall Configuration ---
acking up 'user.rules' to '/etc/ufw/user.rules.20250904_221817'
acking up 'before.rules' to '/etc/ufw/before.rules.20250904_221817'
acking up 'after.rules' to '/etc/ufw/after.rules.20250904_221817'
acking up 'user6.rules' to '/etc/ufw/user6.rules.20250904_221817'
acking up 'before6.rules' to '/etc/ufw/before6.rules.20250904_221817'
acking up 'after6.rules' to '/etc/ufw/after6.rules.20250904_221817'
efault incoming policy changed to 'deny'
be sure to update your rules accordingly)
efault outgoing policy changed to 'allow'
be sure to update your rules accordingly)
ules updated
ules updated
ules updated
ules updated
ommand may disrupt existing ssh connections. Proceed with operation (y|n)? y

```bash
#!/bin/bash

ALLOWED_IP_RANGE="192.168.1.0/24"

if [[ "$EUID" -ne 0 ]]; then
  echo "This script must be run with sudo or as the root user."
  exit 1
fi

if ! command -v ufw &> /dev/null; then
  echo "ufw is not installed. Please run the following command to install it:"
  echo "sudo apt-get update && sudo apt-get install ufw"
  exit 1
fi

echo "--- Starting UFW Firewall Configuration ---"

ufw --force reset
ufw default deny incoming
ufw default allow outgoing

ufw allow from $ALLOWED_IP_RANGE to any port 22
ufw allow from $ALLOWED_IP_RANGE to any port 80
ufw allow from $ALLOWED_IP_RANGE to any port 443
ufw allow from $ALLOWED_IP_RANGE to any port 3306

ufw enable

echo "--- UFW Firewall setup complete! ---"
echo "Current UFW status:"
ufw status verbose
```



7. System MonitoringScript :

Script: #!/bin/bash

LOG_FILE="/var/log/sys_health.log"

echo "--- System Health Report ---" >> $LOG_FILE

echo "Timestamp: $(date)" >> $LOG_FILE

echo "--- CPU Usage ---" >> $LOG_FILE

iostat >> $LOG_FILE

echo "--- Memory Usage ---" >> $LOG_FILE

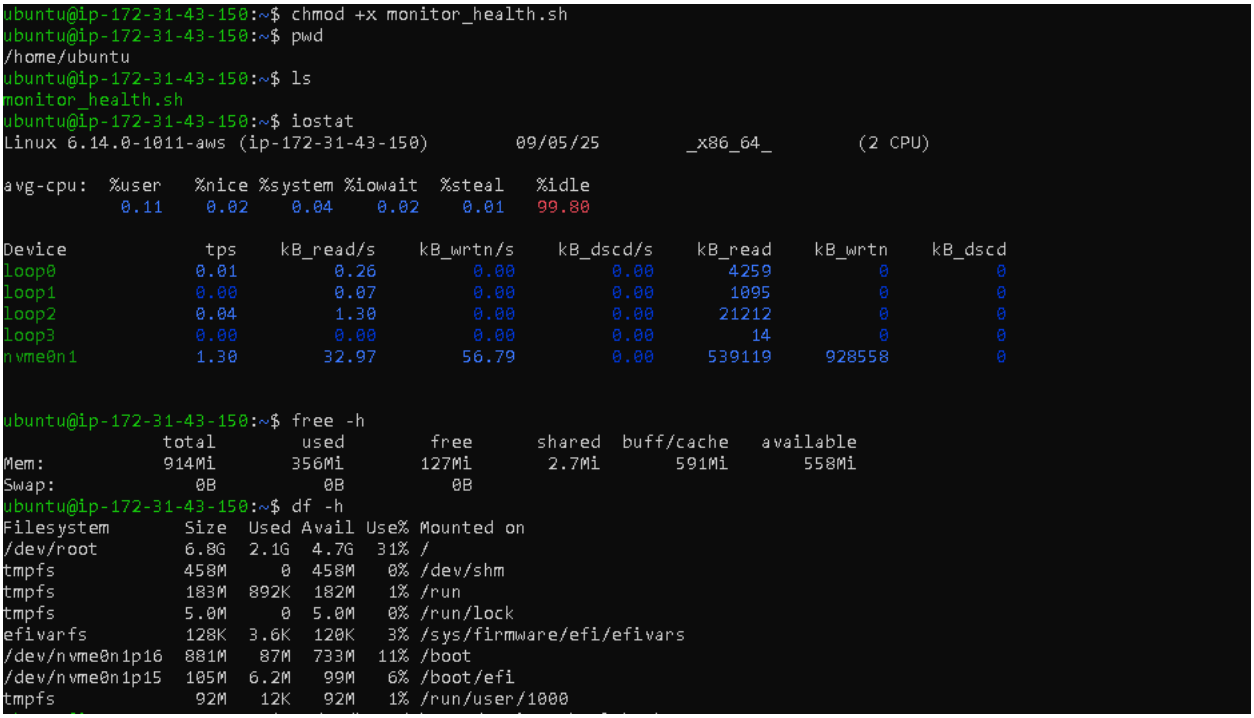
free -h >> $LOG_FILE

echo "--- Disk Usage ---" >> $LOG_FILE

df -h >> $LOG_FILE

echo "" >> $LOG_FILE

Solution:



8. . Log Rotation Setup

Script: #!/bin/bash

LOG_FILE="/var/log/my_app.log"

CONF_FILE="/etc/logrotate.d/my_app"

APP_NAME="my_app"

```
if [[ "$EUID" -ne 0 ]]; then
  echo "This script must be run with sudo or as the root user."
  exit 1
fi

if ! command -v logrotate &> /dev/null; then
    echo "logrotate is not installed. Please run the following command to install it:"
    echo "sudo apt-get update && sudo apt-get install logrotate"
    exit 1
fi

echo "--- Starting Log Rotation Setup for $APP_NAME ---"

echo "Creating a dummy log file for testing..."
echo "This is a custom log entry." > $LOG_FILE

echo "Creating logrotate configuration file at $CONF_FILE..."
cat > $CONF_FILE << EOF
$LOG_FILE {
    daily
    rotate 7
    compress
    missingok
    notifempty
    su root root
}
EOF
```

echo "Logrotate configuration created. To test it, you can run the following command:"

echo "sudo logrotate -f $CONF_FILE"

echo "Running logrotate now to show you the result."

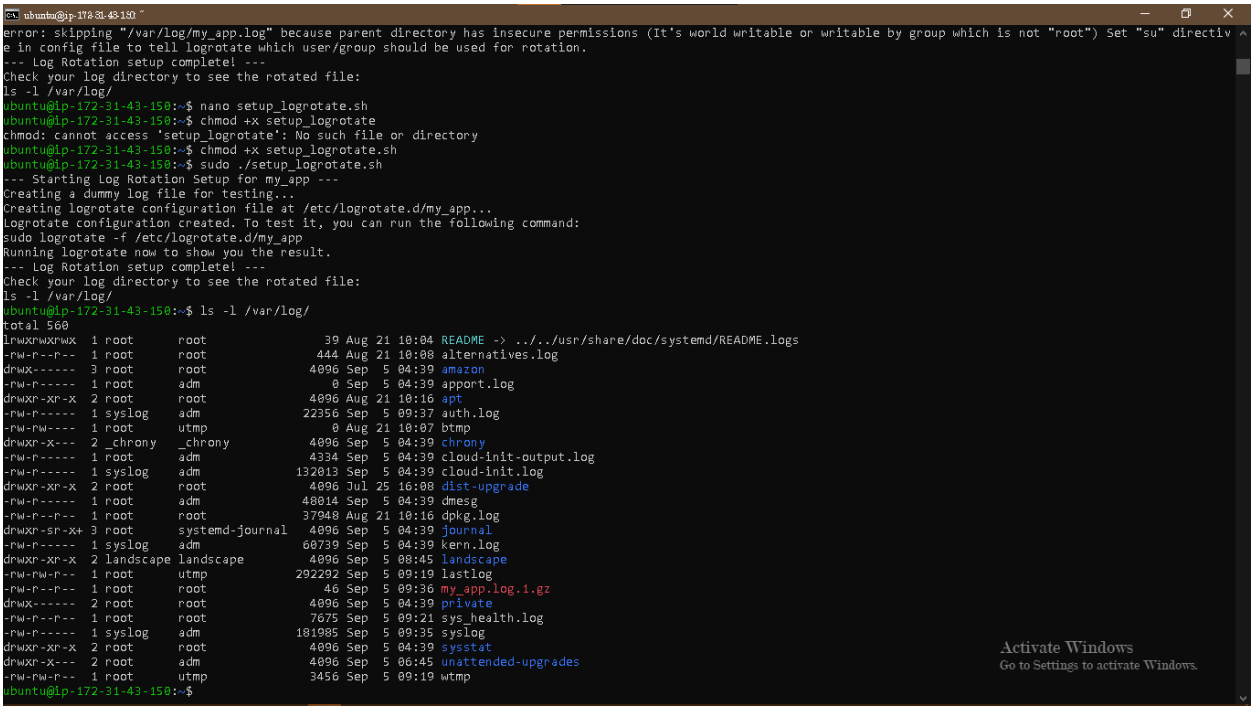
logrotate -f $CONF_FILE

echo "--- Log Rotation setup complete! ---"

echo "Check your log directory to see the rotated file:"

echo "ls -l /var/log/"

Solution ;



9. . DNSServerSetup

Script:

#!/bin/bash

```
BIND_CONF_DIR="/etc/bind"
ZONE_NAME="myuniversity.local"
ZONE_FILE="$BIND_CONF_DIR/$ZONE_NAME.db"

if [[ "$EUID" -ne 0 ]]; then
  echo "This script must be run with sudo or as the root user."
  exit 1
fi

if ! command -v named-checkconf &> /dev/null; then
    echo "bind9 is not installed. Please run the following command to install it:"
    echo "sudo apt-get update && sudo apt-get install bind9 bind9utils"
    exit 1
fi

echo "--- Starting BIND9 DNS Server Setup ---"

echo "Configuring named.conf.options for caching and forwarding..."
sed -i 's/dnssec-validation auto;/dnssec-validation no;/' "$BIND_CONF_DIR/named.conf.options"
sed -i '/listen-on-v6 { any; };/a \
\    forwarders { \
\        8.8.8.8; \
\        8.8.4.4; \
\    }; \
\    allow-query { any; };' "$BIND_CONF_DIR/named.conf.options"

echo "Adding custom zone to named.conf.local..."
cat >> "$BIND_CONF_DIR/named.conf.local" << EOF
zone "$ZONE_NAME" {
```

```
    type master;
    file "$ZONE_FILE";
};
EOF

echo "Creating zone file for $ZONE_NAME..."
cat > "$ZONE_FILE" << EOF
\$TTL 86400
@ IN SOA ns1.myuniversity.local. admin.myuniversity.local. (
    2024040901  ; Serial
    3600        ; Refresh
    1800        ; Retry
    604800      ; Expire
    86400       ; Negative Cache TTL
)

@   IN  NS      ns1.myuniversity.local.
ns1 IN  A       127.0.0.1

@   IN  A       127.0.0.1
www IN  A       127.0.0.1
mail IN MX 10   mail.myuniversity.local.
mail IN A       127.0.0.1
EOF

echo "Setting correct ownership and permissions for zone file..."
chown bind:bind "$ZONE_FILE"
chmod 644 "$ZONE_FILE"
```

echo "Testing BIND9 configuration for syntax errors..."

named-checkconf

named-checkzone "$ZONE_NAME" "$ZONE_FILE"

echo "Restarting BIND9 service to apply changes..."

systemctl restart bind9

echo "--- BIND9 DNS Server setup complete! ---"

echo "To test, temporarily set your nameserver to 127.0.0.1 and run: dig www.myuniversity.local"

Solution:

```
ubuntu@ip-172-31-43-150:~$ sudo ./setup_bind9.sh
--- Starting BIND9 DNS Server Setup ---
Configuring named.conf.options for caching and forwarding...
Adding custom zone to named.conf.local...
Creating zone file for myuniversity.local...
Setting correct ownership and permissions for zone file...
Testing BIND9 configuration for syntax errors...
zone myuniversity.local/IN: loaded serial 2024040901
OK
Restarting BIND9 service to apply changes...
--- BIND9 DNS Server setup complete! ---
To test, temporarily set your nameserver to 127.0.0.1 and run: dig www.myuniversity.local
ubuntu@ip-172-31-43-150:~$
```

10. SSHKeyAuthentication+Hardening:

Script: #!/bin/bash

SSH_DIR="/etc/ssh"

SSHD_CONFIG="$SSH_DIR/sshd_config"

SSH_USER=""

if [["$EUID" -ne 0]]; then

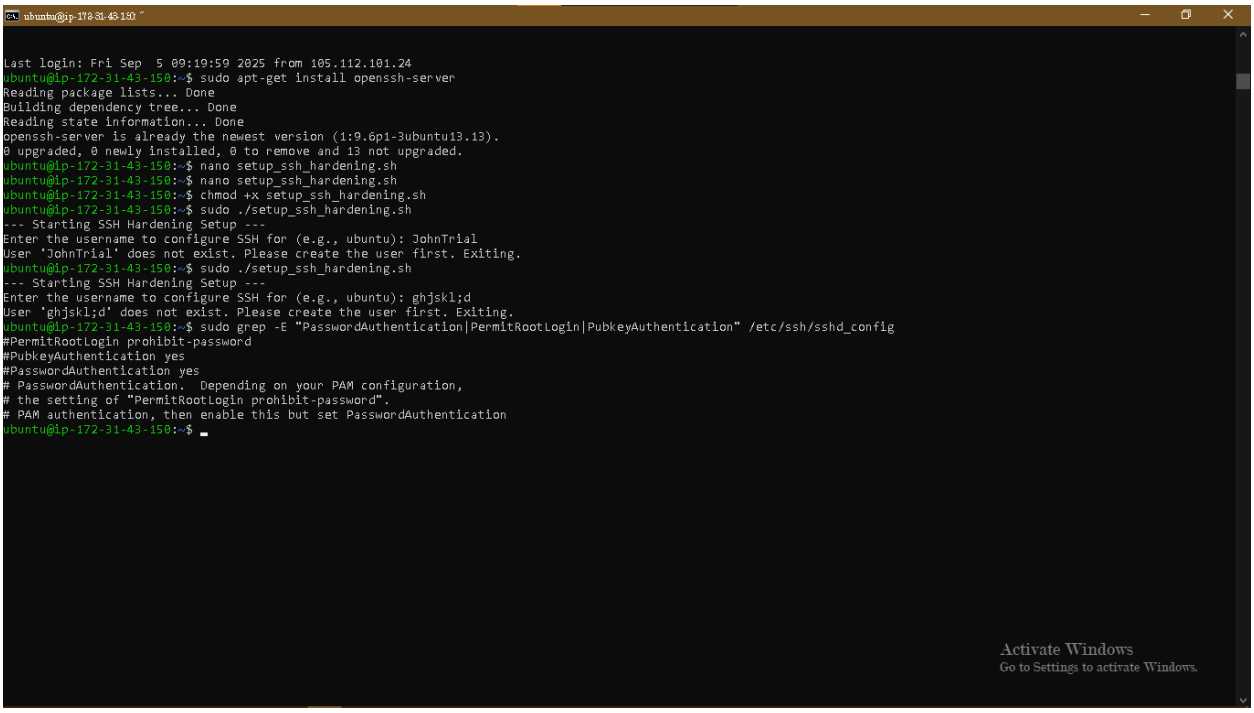
  echo "This script must be run with sudo or as the root user."

  exit 1

```
fi

if ! command -v sshd &> /dev/null; then
    echo "OpenSSH server is not installed. Please run the following command to install it:"
    echo "sudo apt-get update && sudo apt-get install openssh-server"
    exit 1
fi

echo "--- Starting SSH Hardening Setup ---"
read -p "Enter the username to configure SSH for (e.g., ubuntu): " SSH_USER

if [[ -z "$SSH_USER" ]]; then
    echo "Username cannot be empty. Exiting."
    exit 1
fi

if [[ "$SSH_USER" == "root" ]]; then
    echo "This script is designed to disable root login. Please enter a standard user. Exiting."
    exit 1
fi

if ! id "$SSH_USER" &> /dev/null; then
    echo "User '$SSH_USER' does not exist. Please create the user first. Exiting."
    exit 1
fi

echo "Creating .ssh directory and authorized_keys for user '$SSH_USER'..."
mkdir -p /home/$SSH_USER/.ssh
touch /home/$SSH_USER/.ssh/authorized_keys
```

```
chown -R $SSH_USER:$SSH_USER /home/$SSH_USER/.ssh
chmod 700 /home/$SSH_USER/.ssh
chmod 600 /home/$SSH_USER/.ssh/authorized_keys

echo "Generating an SSH key pair for user '$SSH_USER'..."
ssh-keygen -t rsa -b 4096 -f /home/$SSH_USER/.ssh/id_rsa -N ""

echo "Copying the public key to authorized_keys..."
cat /home/$SSH_USER/.ssh/id_rsa.pub >> /home/$SSH_USER/.ssh/authorized_keys

echo "Configuring sshd_config to disable password authentication and root login..."
sed -i 's/^#\?PubkeyAuthentication.*/PubkeyAuthentication yes/' "$SSHD_CONFIG"
sed -i 's/^#\?PasswordAuthentication.*/PasswordAuthentication no/' "$SSHD_CONFIG"
sed -i 's/^#\?PermitRootLogin.*/PermitRootLogin no/' "$SSHD_CONFIG"

echo "Restarting the SSH service to apply changes..."
systemctl restart sshd

echo "--- SSH Hardening complete! ---"
echo "The SSH private key has been saved to: /home/$SSH_USER/.ssh/id_rsa"
echo "Copy this private key to your local machine to connect."
echo "Use the command: ssh $SSH_USER@your_server_ip"
```

solution:



11. Script:

```
#!/bin/bash

set -e


echo "=== Step 1: Creating application script ==="

cat << 'EOF' | sudo tee /usr/local/bin/my_app.sh > /dev/null

#!/bin/bash

echo "Hello World! The service ran at $(date)" >> /var/log/my_app.log

EOF


sudo chmod +x /usr/local/bin/my_app.sh


echo "=== Step 2: Creating systemd service file ==="

cat << 'EOF' | sudo tee /etc/systemd/system/my_app.service > /dev/null

[Unit]
```

Description=My Hello World App

After=network.target

[Service]

Type=simple

ExecStart=/usr/local/bin/my_app.sh

Restart=always

[Install]

WantedBy=multi-user.target

EOF

echo "=== Step 3: Reloading systemd and enabling service ==="

sudo systemctl daemon-reload

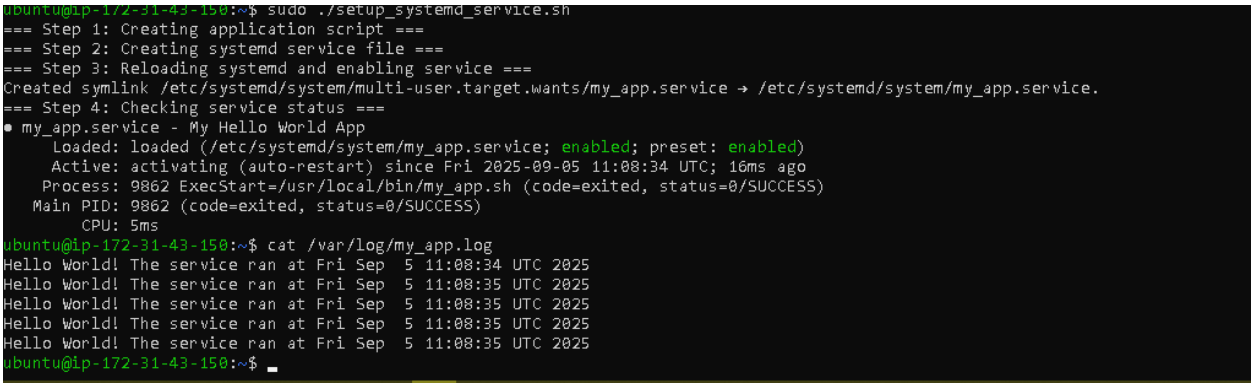
sudo systemctl enable my_app.service

sudo systemctl start my_app.service

echo "=== Step 4: Checking service status ==="

sudo systemctl status my_app.service --no-pager

solution:



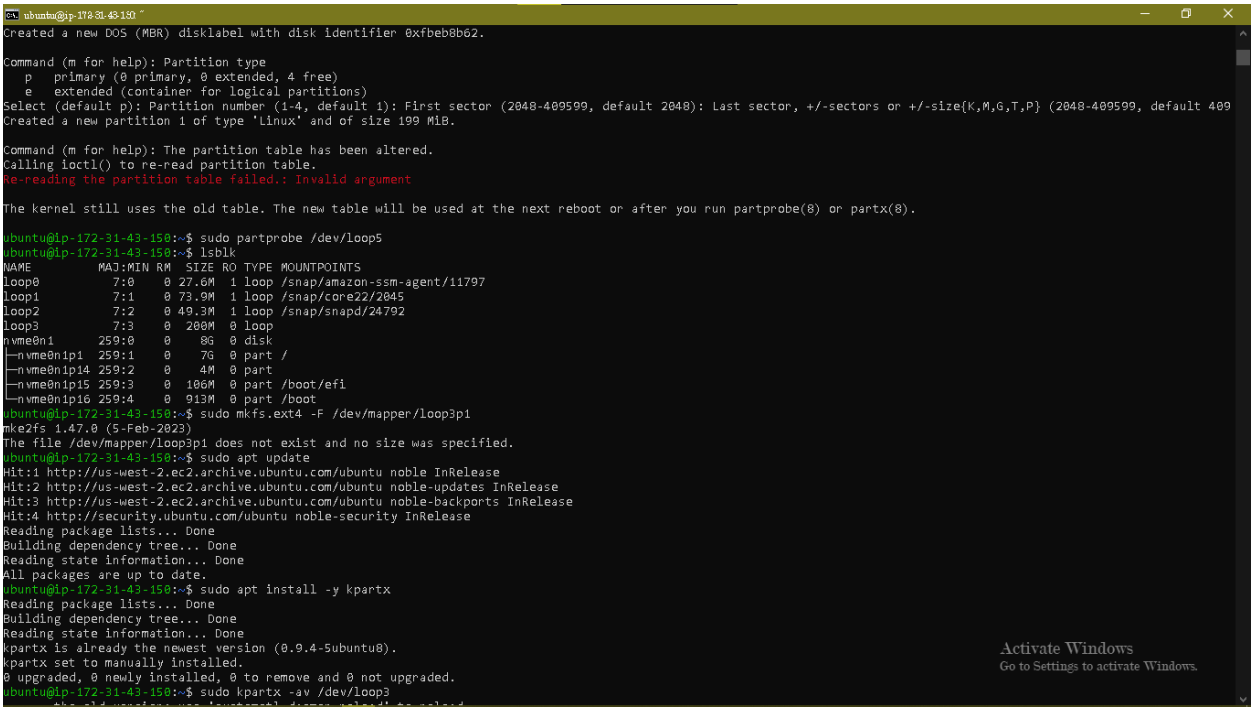
12. DiskPartitioning & Mounting

Script:

```
#!/bin/bash
set -e

echo "=== Step 1: Create a 200MB virtual disk file ==="
DISK_FILE=/mnt/virtualdisk.img
sudo dd if=/dev/zero of=$DISK_FILE bs=1M count=200

echo "=== Step 2: Attach the file as a loop device ==="
LOOP_DEVICE=$(sudo losetup -f --show $DISK_FILE)
echo "Loop device created: $LOOP_DEVICE"

echo "=== Step 3: Partition the loop device (single primary partition) ==="
echo -e "n\np\n1\n\n\nw" | sudo fdisk $LOOP_DEVICE

echo "=== Step 4: Refresh loop devices and map partitions ==="
sudo losetup -d $LOOP_DEVICE
LOOP_DEVICE=$(sudo losetup -f --show $DISK_FILE)
sudo partprobe $LOOP_DEVICE

PARTITION=${LOOP_DEVICE}p1
echo "Partition created: $PARTITION"

echo "=== Step 5: Format partition as ext4 ==="
sudo mkfs.ext4 -F $PARTITION

echo "=== Step 6: Create mount point and mount temporarily ==="
MOUNT_POINT=/mnt/mydata
```

```
sudo mkdir -p $MOUNT_POINT
sudo mount $PARTITION $MOUNT_POINT

echo "=== Step 7: Verify temporary mount ==="
df -h | grep $MOUNT_POINT

echo "=== Step 8: Add to /etc/fstab for persistence ==="
UUID=$(sudo blkid -s UUID -o value $PARTITION)
echo "UUID=$UUID $MOUNT_POINT ext4 defaults 0 2" | sudo tee -a /etc/fstab

echo "=== Step 9: Test fstab by unmounting and remounting ==="
sudo umount $MOUNT_POINT
sudo mount -a
df -h | grep $MOUNT_POINT

echo "=== Step 10: Reboot test (simulated) ==="
echo "Normally you'd run: sudo reboot"
echo "After reboot, check: df -h | grep $MOUNT_POINT"
```

Solution:



13. . Postfix Mail Server (Local Only)

Script: #!/bin/bash

set -e

echo "=== Step 1: Updating packages ==="

sudo apt update -y

echo "=== Step 2: Installing Postfix (local only) and mailutils ==="

Preseed Postfix configuration (local only)

echo "postfix postfix/mailname string localhost" | sudo debconf-set-selections

echo "postfix postfix/main_mailer_type string Local only" | sudo debconf-set-selections

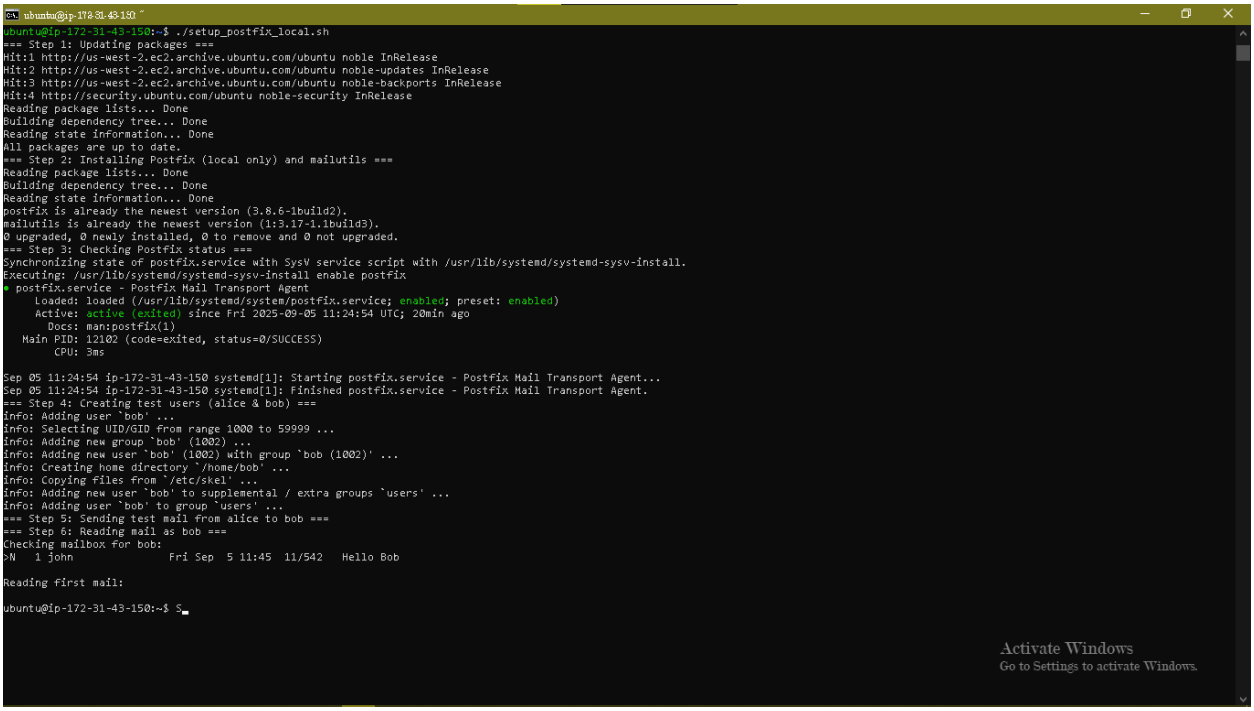
sudo DEBIAN_FRONTEND=noninteractive apt install -y postfix mailutils

echo "=== Step 3: Checking Postfix status ==="

sudo systemctl enable postfix

```
sudo systemctl start postfix

sudo systemctl status postfix --no-pager


echo "=== Step 4: Creating test users (alice & bob) ==="

if ! id alice &>/dev/null; then

  sudo adduser --disabled-password --gecos "" alice

fi

if ! id bob &>/dev/null; then

  sudo adduser --disabled-password --gecos "" bob

fi


echo "=== Step 5: Sending test mail from alice to bob ==="

echo -e "Subject: Hello Bob\nHi Bob, this is a local mail test from Alice." | sudo -u alice sendmail bob


sleep 2  # Give Postfix time to deliver


echo "=== Step 6: Reading mail as bob ==="

sudo -u bob bash -c "echo 'Checking mailbox for bob:' && mail -H && echo && echo 'Reading first mail:' && echo 1 | mail -f"
```

Solution:



14. Backup&RestoreProject

Script:

#!/bin/bash

set -e

Directories

SOURCE_DIR="/var/www/html"

BACKUP_DIR="/backup"

Ensure backup directory exists

sudo mkdir -p $BACKUP_DIR

Create a timestamp

TIMESTAMP=$(date +"%Y%m%d_%H%M%S")

Backup filename

```
BACKUP_FILE="$BACKUP_DIR/html_backup_$TIMESTAMP.tar.gz"

echo ">>> Creating backup of $SOURCE_DIR to $BACKUP_FILE"

# Create the backup
sudo tar -czf $BACKUP_FILE -C /var/www html

echo ">>> Backup created successfully!"

# List backups
ls -lh $BACKUP_DIR/html_backup_*.tar.gz

# Test restore: Extract latest backup into /var/www/html_restored
LATEST_BACKUP=$(ls -t $BACKUP_DIR/html_backup_*.tar.gz | head -n 1)
RESTORE_DIR="/var/www/html_restored"

echo ">>> Restoring latest backup $LATEST_BACKUP to $RESTORE_DIR"

# Ensure restore directory exists
sudo rm -rf $RESTORE_DIR
sudo mkdir -p $RESTORE_DIR

# Extract
sudo tar -xzf $LATEST_BACKUP -C /var/www
sudo mv /var/www/html $RESTORE_DIR

echo ">>> Restore completed! Files are in $RESTORE_DIR"
```

Solution:

```
ubuntu@ip-172-31-43-150:~$ df -h | grep mydata
ubuntu@ip-172-31-43-150:~$ sudo apt update
Hit:1 http://us-west-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-west-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-west-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
ubuntu@ip-172-31-43-150:~$ nano setup_backup_restore.sh
ubuntu@ip-172-31-43-150:~$ chmod +x setup_backup_restore.sh
ubuntu@ip-172-31-43-150:~$ sudo ./setup_backup_restore.sh
>>> Creating backup of /var/www/html to /backup/html_backup_20250905_125258.tar.gz
tar: /var/www: Cannot open: No such file or directory
tar: Error is not recoverable: exiting now
ubuntu@ip-172-31-43-150:~$ sudo mkdir -p /var/www/html
echechoubuntu@ip-172-31-43-150:~$  "Hello Backup Test!" | sudo tee /var/www/html/index.html
Hello Backup Test!: command not found
ubuntu@ip-172-31-43-150:~$ sudo mkdir -p /var/www/html
ubuntu@ip-172-31-43-150:~$ echo "Hello Backup Test!" | sudo tee /var/www/html/index.html
Hello Backup Test!
ubuntu@ip-172-31-43-150:~$ sudo ./setup_backup_restore.sh
>>> Creating backup of /var/www/html to /backup/html_backup_20250905_130020.tar.gz
>>> Backup created successfully!
>>> Available backups:
-rw-r--r-- 1 root root  20 Sep  5 12:52 /backup/html_backup_20250905_125258.tar.gz
-rw-r--r-- 1 root root 179 Sep  5 13:00 /backup/html_backup_20250905_130020.tar.gz
>>> Restoring latest backup: /backup/html_backup_20250905_130020.tar.gz
>>> Restore completed! Files are now in /var/www/html_restored
ubuntu@ip-172-31-43-150:~$
```

15. Containerization Challenge

Script: #!/bin/bash

set -e

echo ">>> Updating packages..."

sudo apt update -y

echo ">>> Installing prerequisites..."

sudo apt install -y apt-transport-https ca-certificates curl software-properties-common

echo ">>> Installing Docker..."

Add Docker’s official GPG key

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg

# Add stable repo
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] \
  https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt update -y
sudo apt install -y docker-ce docker-ce-cli containerd.io

echo ">>> Enabling and starting Docker..."
sudo systemctl enable docker
sudo systemctl start docker

echo ">>> Pulling Nginx container..."
sudo docker pull nginx:latest

echo ">>> Running Nginx container on port 8080..."
# Stop any existing container named mynginx
if [ "$(sudo docker ps -aq -f name=mynginx)" ]; then
    sudo docker rm -f mynginx
fi
sudo docker run -d --name mynginx -p 8080:80 nginx:latest

echo ">>> Checking container status..."
sudo docker ps | grep mynginx
```

echo ">>> Testing Nginx locally..."

curl -I http://localhost:8080 || true

echo ">>> Done! Visit http://<your-server-public-ip>:8080 in your browser."

Solution: