NFS Server Task: Professional Project Document

NAME : Ziad Mahmoud Ahmed Abdelgwad

Objective:

Simulate an NFS server and client setup on a single Linux machine using localhost. Configure a shared directory, mount it locally, and test file sharing.

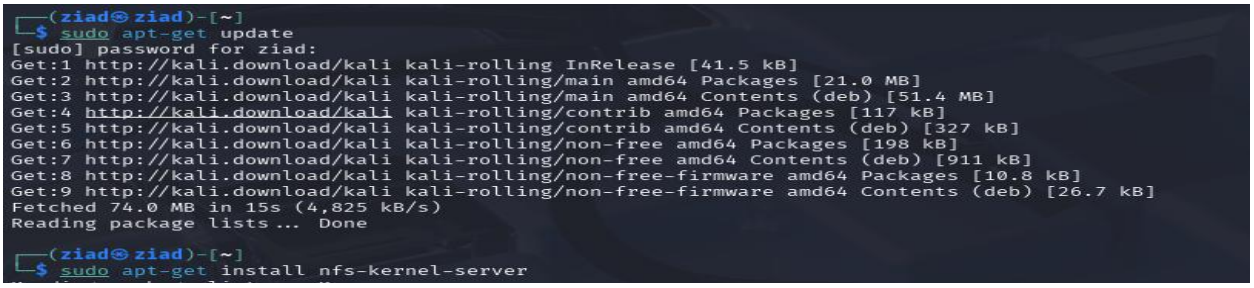
Step 1: Install and Configure the NFS Server

The first step is to install the necessary NFS server packages and verify that the service is running correctly.

1.1. Install the NFS server package:
The following commands were used to update the package list and install the NFS kernel server on a Debian-based system (Kali Linux).

Bash

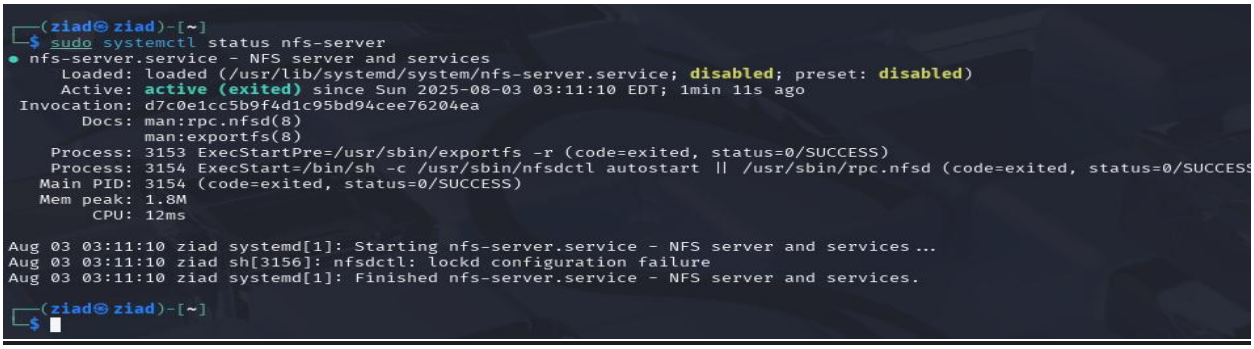
```
sudo apt-get update

sudo apt-get install nfs-kernel-server
```



1.2. Check the service status:
After installation, the systemctl command was used to verify that the nfs-server service is active and running.

Bash

```
sudo systemctl status nfs-server
```



Step 2: Create and Export the Shared Directory

This step involves creating the shared directory, setting its permissions, and configuring the NFS server to export it to the local machine.

2.1. Create the shared directory:
A new directory /srv/nfs_server was created, and its ownership and permissions were adjusted to ensure the NFS service can access it.

Bash

```
sudo mkdir -p /srv/nfs_server

sudo chown nobody:nogroup /srv/nfs_server

sudo chmod 777 /srv/nfs_server
```

2.2. Edit the /etc/exports file:
The nano text editor was used to open the /etc/exports file.

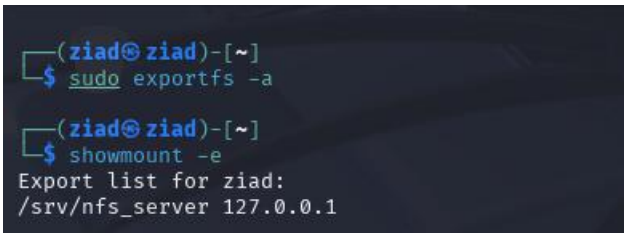
Bash

```
sudo nano /etc/exports
```

The following line was added to the file to export the directory to the loopback IP address (127.0.0.1) with read/write (rw) access and other options for stable sharing.

2.3. Export and verify the configuration:
The exportfs -a command was executed to apply the changes without a reboot, and showmount -e was used to confirm that the directory is correctly exported.
Bash

```
sudo exportfs -a

showmount -e
```



Step 3: Simulate a Client and Mount the Directory

A separate directory was created to act as the client's mount point, and the NFS share was mounted to it.

3.1. Create the client mount point:

```
sudo mkdir -p /mnt/nfs_client
```

3.2. Mount the shared directory:
The NFS share was mounted using the server's loopback IP. The mount | grep nfs command was used to verify the successful mount.

```
sudo mount -t nfs 127.0.0.1:/srv/nfs_server /mnt/nfs_client

mount | grep nfs
```

Step 4: Test File Sharing

To prove that the NFS setup is working, files were created on both the server and client paths.

4.1. Create a file on the server side:
The tee command was used with sudo to create a file with content in the server's shared directory.

```
echo "This is a file created on the server." | sudo tee /srv/nfs_server/server_file.txt
```

4.2. Verify the file on the client side:
The ls -l command on the client's mount point showed the new file. Its content was verified using cat.

```
ls -l /mnt/nfs_client
```

```
cat /mnt/nfs_client/server_file.txt
```

Step 5: Configure Persistent Mount

This section demonstrates how to make the NFS mount persistent across system reboots using /etc/fstab.

5.1. Edit the /etc/fstab file:
The file was opened with nano.

```
sudo nano /etc/fstab
```

The following line was added to ensure the NFS share is automatically mounted at startup.

5.2. Reload systemd and test the mount:
After adding the entry, systemctl daemon-reload was used to refresh the system's configuration cache, followed by mount -a to test the new fstab entry.

```
sudo umount /mnt/nfs_client
```

```
sudo systemctl daemon-reload
```

```
sudo mount -a
```

Step 6: Add a Firewall Rule

The final optional step involves adding a basic firewall rule to allow NFS traffic from the loopback IP address.

6.1. Install ufw:
As ufw was not found on the system, it was installed using apt-get.

sudo apt-get update

sudo apt-get install ufw

```
┌──(ziad㉿ziad)-[~]
└─$ sudo apt-get update
sudo apt-get install ufw
Hit:1 http://http.kali.org/kali kali-rolling InRelease
Reading package lists ... Done
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
Suggested packages:
```

6.2. Configure and enable the firewall:
The rule to allow NFS from localhost was added, and then the firewall was enabled.
Bash

sudo ufw allow from 127.0.0.1 to any port nfs

sudo ufw enable

6.3. Check the firewall status:
The ufw status verbose command confirmed that the firewall is active and the rule for NFS traffic from 127.0.0.1 has been applied.

```
┌──(ziad㉿ziad)-[~]
└─$ sudo ufw allow from 127.0.0.1 to any port nfs
Rules updated

┌──(ziad㉿ziad)-[~]
└─$ sudo ufw allow from 127.0.0.1 to any port nfs
Skipping adding existing rule

┌──(ziad㉿ziad)-[~]
└─$ sudo ufw enable
Firewall is active and enabled on system startup

┌──(ziad㉿ziad)-[~]
└─$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
2049                       ALLOW IN    127.0.0.1
```

]