| | |
|---|---|
| | **Experiment No : 9** **Date :** |
| **Title** | **Configuration of NFS Server** |
| **Aim** | To Configuration of NFS File server and transfer files to a windows client |
| **Hardware Requirement** | Personal Computer |
| **Software Requirement** | Linux Operating System(Ubuntu 16.04) , Shell-Interpreter |

**Theory**

Network File System (NFS) is a networking protocol for distributed file sharing. A file system defines the way data in the form of files is stored and retrieved from storage devices, such as hard disk drives, solid-state drives and tape drives. NFS is a network file sharing protocol that defines the way files are stored and retrieved from storage devices across networks.

The NFS protocol defines a network file system, originally developed for local file sharing among Unix systems and released by Sun Microsystems in 1984. The NFS protocol specification was first published by the Internet Engineering Task Force (IETF) as an internet protocol in RFC 1094 in 1989. The current version of the NFS protocol is documented in RFC 7530, which documents the NFS version 4 (NFSv4) Protocol.

NFS enables system administrators to share all or a portion of a file system on a networked server to make it accessible to remote computer users. Clients with authorization to access the shared file system can mount NFS shares, also known as shared file systems. NFS uses Remote Procedure Calls (RPCs) to route requests between clients and servers.

NFS is one of the most widely used protocols for file servers. NFS implementations are available for most modern operating systems (OSes), including the following:

- Hewlett Packard Enterprise HP-UX

- IBM AIX
- Microsoft Windows
- Linux
- Oracle Solaris

Cloud vendors also implement the NFS protocol for cloud storage, including Amazon Elastic File System, NFS file shares in Microsoft Azure and Google Cloud Filestore.

Any device that can be attached to an NFS host file system can be shared through NFS. This includes hard disks, solid state drives, tape drives, printers and other peripherals. Users with appropriate permissions can access resources from their client machines as if those resources are mounted locally.

NFS is an application layer protocol, meaning that it can operate over any transport or network protocol stack. However, in most cases NFS is implemented on systems running the TCP/IP protocol suite. The original intention for NFS was to create a simple and stateless protocol for distributed file system sharing.

Early versions of NFS used the User Datagram Protocol (UDP) for its transport layer. This eliminated the need to define a stateful storage protocol; however, NFS now supports both the Transmission Control Protocol (TCP) and UDP. Support for TCP as a transport layer protocol was added to NFS version 3 (NFSv3) in 1995.

**Benefits of Using NFS**

Over the years, NFS has evolved to support more security, better file sharing (locking), and better (caching) performance. Moreover, it's a relatively affordable and easy-to-use solution for network file sharing that uses existing internet protocol infrastructure.

At present, here are the benefits of the NFS service:

- Multiple clients can use the same files, which allows everyone on the network to use the same data, accessing it on remote hosts as if it were acceding local files.

- Computers share applications, which eliminates the needs for local disk space and reduces storage costs.
- All users can read the same files, so data can remain up-to-date, and it's consistent and reliable.
- Mounting the file system is transparent to all users.
- Support for heterogeneous environments allows you to run mixed technology from multiple vendors and use interoperable components.
- System admin overhead is reduced due to centralization of data.
- Fewer removable disks and drives laying around provides a reduction of security concerns—which is always good!

**Working of File System work**

NFS is a client-server protocol. An NFS server is a host that meets the following requirements:

- has NFS server software installed;
- has at least one network connection for sharing NFS resources; and
- is configured to accept and respond to NFS requests over the network connection.

An NFS client is a host that meets the following requirements:

- has NFS client software installed;
- has network connectivity to an NFS server;
- is authorized to access resources on the NFS server; and
- is configured to send and receive NFS requests over the network connection.

NFS was initially conceived as a method for sharing file systems across workgroups using Unix. It is still often used for ad hoc sharing of resources.

## Installation Step By Step

## Server Side Configuration

**Step-1** **Install NFS Package**

Before we start NFS configuration, first we need to install **NFS package**. let's install required packages using the command below:

```
root@Server16:~# apt install nfs-kernel-server
```

**To Check Status of NFS Server**

**/etc/init.d/nfs-kernel-server status      (status should be active)**

**Step-2** **Create a Shared Directory and change owner**

we need to create a directory first that we will share with the client. Use command below to create a directory-

```
root@Server16:~# mkdir /home/nfsshare
```

Change owner of directory use below command.

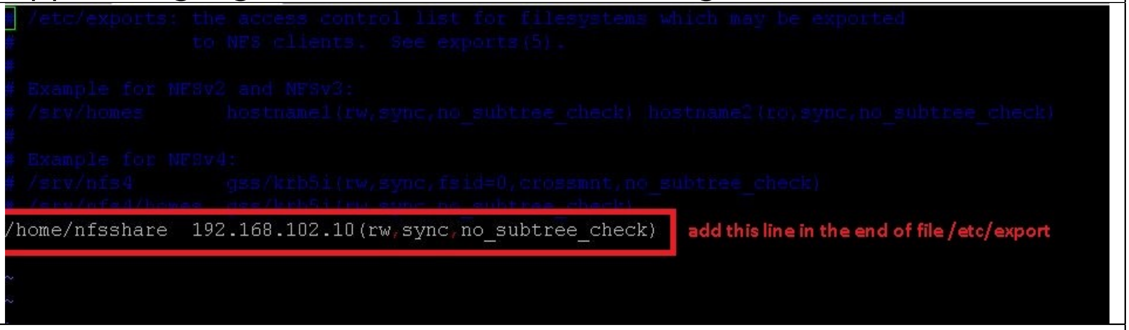
```
root@Server16:~# chown nobody:nogroup /home/nfsshare/
```

**Step-3** **Configure export directory**

For sharing a directory, we need to make an entry
into /etc/export configuration file.
Let's modify export configuration file using command below

```
root@Server16:~# vi /etc/exports
```

Append highlighted line in the end of configuration file as below



```
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
/home/nfsshare  192.168.102.10(rw,sync,no_subtree_check)
~
~
```

add this line in the end of file /etc/export

Save and Exit from the file.

| | |
|---|---|
| | **Where switches as follow:**<br>**rw =** This option allows the client computer to read and write.<br>**sync =** This option allows NFS to write changes back to disk<br>**no_subtree_check =** By default, NFS translates requests from a root user remotely into a non-privileged user on the server. This was intended as a security feature to prevent a root account on the client from using the file system of the host as root. no_root_squash disables this behaviour for certain shares |
| **Step-4** | **Restart NFS Server Service to apply changes** |
| | root@Server16:~# **systemctl restart nfs-server.service** |
| **Step-5** | **Adjust Firewall in case it's not disabled.** |
| | Check Firewall Status using the command below |
| | **root@Server16:~# ufw status**<br>**Status: inactive** |
| | In my case, it is inactive, if it is active then run the command below to allow NFS access. |
| | **root@Server16:~# ufw allow from 192.168.102.10 to any port nfs**<br>**Rules updated** |
| | Reload Firewall service to apply changes |
| | **root@Server16:~# ufw reload**<br>**Firewall reloaded** |
| **Client Side Configuration:** | |
| **Step-6** | **Install NFS client package** |
| | Before we start mounting, First we need to install nfs-common package on the client system let do it by hitting command below. |
| | **root@US16:~# apt install nfs-common** |
| **Step-7** | **Create the mount point Client end.** |

we need to create mount point first, let do it by hitting command below.

```
root@US16:~# mkdir /mnt/nfserver
```

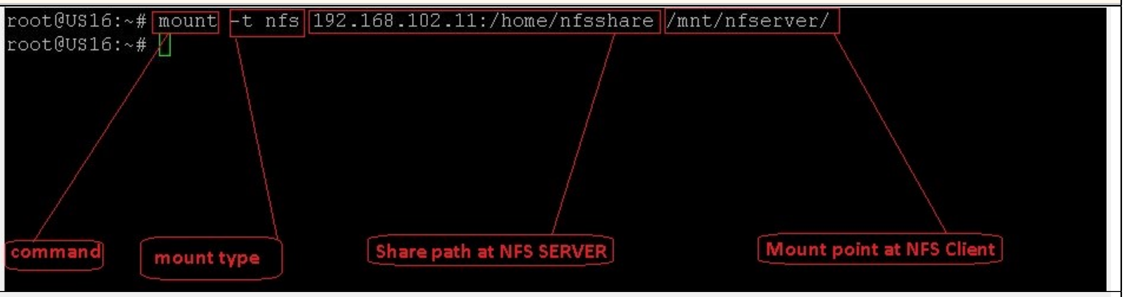
**Step-8** **Mount NFS share on the Client end.**

We need to reboot system after making changes into fstab file, but let's use command mount -a to apply changes.

```
root@US16:~# mount -a
```

Let's check mount point using below command.

```
root@US16:~# df -h
Filesystem                    Size  Used Avail Use% Mounted on
udev                         478M     0  478M   0% /dev
tmpfs                         100M  3.2M   97M   4% /run
/dev/mapper/US16--vg-root      6.3G  2.8G  3.3G  46% /
192.168.102.11:/home/nfsshare  6.6G  5.8G  471M  93%
/mnt/nfserver
```



```
root@US16:~# mount -t nfs 192.168.102.11:/home/nfsshare /mnt/nfserver/
root@US16:~#
```

**NFS Client Side**

Use command df -h to list mounted point:

```
root@US16:~# df -h
Filesystem                    Size  Used Avail Use% Mounted on
udev                         478M     0  478M   0% /dev
tmpfs                         100M  3.2M   97M   4% /run
/dev/mapper/US16--vg-root      6.3G  2.8G  3.3G  46% /
192.168.102.11:/home/nfsshare  6.6G  5.8G  471M  93%
/mnt/nfserver
```

Create test directory to the mounted folder

```
root@US16:/mnt/nfserver# mkdir testnfs/
```

**NFS Server Side**

Let's check created directory at NFS SERVER

```
root@Server16:~# ls -la /home/nfsshare/
total 12
drwxr-xr-x  3 nobody nogroup 4096 Jul  7 16:22 .
drwxr-xr-x 10 root   root    4096 Jul  7 15:07 ..
drwxr-xr-x  2 nobody nogroup 4096 Jul  7 16:22 testnfs
```

**NFS Share working as expected**

**Step-10** **Create fstab entry to mount NFS share permanently**

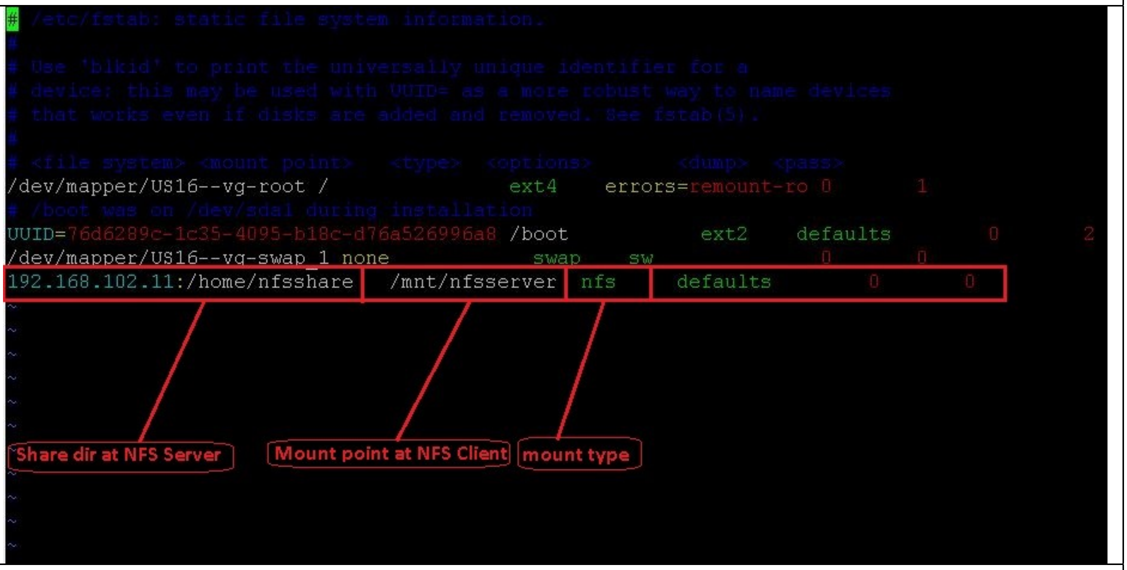
We can mount any NFS share by mount command but it's temprory mounting and will unmount once system reboot, To solve this problem let's make a fstab file entry to mount permanently.

```
root@US16:~# vi /etc/fstab
```

Append the line below into the file fstab

```
192.168.102.11:/home/nfsshare   /mnt/nfsserver  nfs    defaults
      0      0
```

**save and exit form the file.**



```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
/dev/mapper/US16--vg-root /               ext4    errors=remount-ro 0       1
# /boot was on /dev/sda1 during installation
UUID=76d6289c-1c35-4095-b18c-d76a526996a8 /boot           ext2    defaults        0       2
/dev/mapper/US16--vg-swap_1 none            swap    sw              0       0
192.168.102.11:/home/nfsshare   /mnt/nfsserver  nfs     defaults        0       0
~
~
~
~
~
~
~
~
~
~
~
```

**Step-11** **Apply make changes of fstab file using below command.**

We need to reboot system after making changes into fstab file, but let's use command mount -a to apply changes.

```
root@US16:~# mount -a
```

Let's check mount point using below command.

```
root@US16:~# df -h
Filesystem                    Size  Used Avail Use% Mounted on
udev                         478M     0  478M   0% /dev
tmpfs                         100M  3.2M   97M   4% /run
/dev/mapper/US16--vg-root      6.3G  2.8G  3.3G  46% /
192.168.102.11:/home/nfsshare  6.6G  5.8G  471M  93%
/mnt/nfserver
```

**Conclusion**

This NFS server experiment has successfully demonstrated the process of setting up and configuring a Network File System (NFS) server in a Linux environment.

Through the successful completion of this experiment, the participants have gained practical experience in configuring and managing an NFS server-client environment, which is a widely used technique for distributed file sharing in both enterprise and small-scale computing scenarios.

The ability to share files, applications, and data across multiple systems, while maintaining data consistency and centralized management, highlights the benefits of the NFS protocol and its widespread adoption in modern computing infrastructures.

Overall, this NFS server experiment has provided valuable hands-on experience in the setup and configuration of a network file sharing system, equipping the participants with the necessary skills to implement and troubleshoot similar deployments in real-world scenarios.

| | |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| **Signature** | |
| | |
| | |
| | |
| **Date** | |
| | |
| **Grade** | |
| | |