NETWORK FILE SHARING

NFS

- Network File System (NFS) protocol allow Linux client to mount remote file systems and interact with those file systems as they are mounted locally.
- NFS stand for Network File System
- NFS is used to share files and printer between Linux / Unix systems
- Red Hat Enterprise Linux 6 supports NFSv2, NFSv3, and NFSv4 clients.
- By default RHEL6 use NFSv4 if the server supports it.

NFSv1

- NFSv1 was the development stage of NFS protocol.
- It was used only for in house experimental purpose.

NFSv2

- NFSv2 supports only 32 bit.
- NFSv2 only allowed the first 2 GB of a file to be read
- NFSv2 operated only over UDP

NFSv3

- NFSv3 supports 64 bit file system.
- NFSv3 can handle files larger than 2 GB.
- NFSv3 supports asynchronous writes on the server. asynchronous writes improve write performance.
- NFSv3 supports additional file attributes in many replies, to avoid the need to re-fetch them.
- NFSv3 supports READDIRPLUS operation. READDIRPLUS operation get file handles and attributes along with file names when scanning a directory.
- NFSv3 supports TCP. Using TCP as a transport made NFS over a WAN more feasible.

NFSv4

- NFSv4 retains all NFSv3 advantages.
- NFSv4 supports ACLs.
- NFSv4 uses the virtual file system to present the server's export.
- NFSv4 supports Pseudo file system. Pseudo File System provide maximum flexibility. Exports Pathname on servers can be changed transparently to clients.
- NFSv4 have locking operations as the part of protocol which keep track of open files and delegations.
- NFSv4 works through firewalls and on the Internet.

- NFS port no is 2049
- NFS server config file /etc/exports
- Nfs client config file /etc/fstab
- Nfs shares are mounted at boot time by /etc/rc.d/init.d/netfs

NFS CONFIGURATION LAB

- Create a NFS server machine with
  hostname =server
  ip address =192.168.0.254

- Create a linux client machine with
  hostname =client
  ip address = 192.168.0.253

- make the ip address and hostnames permanent on both server and client side

Configuring NFS SERVER

If we want to configure nfs service we have to install the few Packages like

- **nfs-utils, portmap** , **nfs4-acl-tools xinetd(** rhel 5)
- **nfs-utils, rpcbind** , **nfs4-acl-tools xinetd(** rhel 6)
- check whether the packages installed or not if there are no packages install by using rpm or yum

-
  ```
  rpm  -qa | grep nfs
  rpm  -ivh nfs* --nodeps --force
  ```

-
  ```
  rpm  -qa | grep portmap
  rpm  -ivh portmap* --nodeps --force
  ```

-
  ```
  rpm  -qa | grep xinetd*
  rpm  -ivh xinetd* --nodeps --force
  ```

Configuring NFS SERVER

- restart the xinetd portmap nfs services

```
service nfs restart
service portmap restart
service xinetd restart
```

- To enable the service at boot time use chkconfig comm

```
chkconfig   nfs   on
chkconfig   portmap   on
chkconfig   xinetd   on
```

Configuring NFS SERVER

- After rebooting verify the status it must be in running state

```
service    nfs    status
service   portmap status
service    xinetd    status
```

- make a directory /nsfshare and give read write or full permissions to the directory
- Mkdir /nfsshare
- chmod   777 /nfsshare
- now open vi /etc/exports file then add entry in the file
- /nfsshare   192.168.0.253 (rw , sync)
-                  (NFS-Client IP)
- Save and quit
- Restart the nfs service and restart nfs daemons
- exportfs - avr

exportsfs

- **exportfs** command is used to maintain the current table of exported file systems for NFS.
- This list is kept in a separate file named **/var/lib/nfs/xtab** which is read by **mountd** when a remote host requests access to mount.
- This **xtab** file is initialized with the list of all file systems named in **/etc/exports** by invoking **exportfs –a** .
- administrators can choose to add and delete individual file systems without modifying**/etc/exports** using **exportfs.**
- **exportfs** and it's partner program **mountd** work in one of two modes, a legacy mode which applies to 2.4 and earlier versions of the Linux kernel, and a new mode which applies to 2.6 and later versions providing the **nfsd** virtual filesystem has been mounted at **/proc/fs/nfsd** or**/proc/fs/nfs**. If this filesystem is not mounted in 2.6, the legacy mode is used.

EXPORTFS

- In the new mode, **exportfs** does not give any information to the kernel but only provides it to**mountd** through the **/var/lib/nfs/xtab** file. **mountd** will listen to requests from the kernel and will provide information as needed.
- In the legacy mode, any export requests which identify a specific host (rather than a subnet or netgroup etc) are entered directly into the kernel's export table as well as being written to**/var/lib/nfs/xtab**. Further, any mount points listed in **/var/lib/nfs/rmtab** which match a non host-specific export request will cause an appropriate export entry for the host given in**rmtab** to be entered into the kernel's export table.

Client side configuration

- verify showmount - e 192.168.0.12 (server ip)
- make a directory mkdir /nfs
- mount on /nfs

```
#mount –t  nfs 192.168.0.254 :/nfsshare  /nfs
                    ↑
              (NFS-Server IP)
```

verify df –h

- mount it permanently in /etc/fstab

```
192.168.0.254:/nfsshare  /nfs  nfs  defaults   0  0
```

```
#df  -h
```

- cd /nfs and create a touch file
- touch client

And verify whether it is shared on server side on not

differences b/w nfsv3 & nfsv4

1.Transport protocols

- For NFSv3, the MOUNT service is normally supported over the TCP and UDP protocols.
- For NFSv4, only the TCP protocol is supported.
  NFS v4 is designed for internet use. One unique network port is used on NFSv4. This predetermined port is fixed.
- The default is port 2049. Using NFS v4 through firewalls is easier than with earlier NFS versions.

2. Locking operation

- NFS v3 protocol is stateless, so an additional Network Lock Manager (NLM) protocol, an auxiliary protocol for file locking, is required to support locking of NFS-mounted files READ/WRITE. Also NLM is stateful in that the server LOCKED keeps track of locks.
- NFSv4 is stateful. Locking operations(open/read/write/lock/locku/close) are part of the protocol proper. NLM is not used by NFSv4.

3. Required Services

- NFSv3 relies on Remote Procedure Calls (RPC) to encode and decode requests between clients and servers. NFSv3 depends on portmapper, rpc.mountd, rpc.lockd, rpc.statd.
- NFSv4 has no interaction with portmapper, rpc.mountd, rpc.lockd, and rpc.statd, since protocol support has been incorporated into the v4 protocol. NFSv4 listens on "well-known" TCP port (2049) which eliminates the need for the portmapper interaction. The mounting and locking protocols have been incorpated into the V4 protocol which eliminates the need for interaction with rpc.mountd and rpc.lockd.

4. Security

- NFS v3 supports export/mount. Thus the host makes the mount request, not a user of the file system.
- With NFSv4, the mandatory security mechanisms are oriented towards authenticating individual users, e.g. by configuring the Kerberos version 5 GSS-API or other security mechanism.

Daemons of nfs-Server

- Nfsd
- Lockd
- Rpciod
- Rpc.{mountd,rquotad,statd}

Troubleshooting

- Issue ----- **df -h is hung**
- check in client df -h , service nfs status, check any nfs mount points
- login to the nfs server
- service nfs status
- service nfs start/restart
- check the mountpoint exists in nfs server
- if it exists ..no issue from server side ..just restart nfs service
- move to client machine mount it
- Mount –t nfs 192.168.0.12:/nfsshare /nfs

NFS Error codes