How to set up NFS server and client on CentOS 7

Author : Dan Nanni

Categories : CentOS, Filesystem, Networking

Tagged as : mountnfs

Question: I would like to share a directory via NFS between CentOS 7 hosts. What is the procedure for configuring NFS server and client, and enabling NFS mounts between them in CentOS 7 environment?

NFS is a distributed file system protocol built on top of Sun's RPC (remote procedure call). NFS is popularly used to share directories and files in LAN network environments, and is also adopted in many Network Attached Storage (NAS) systems and cloud storage backends. In a typical NFS deployment, an NFS server exports part of its local file system as NFS shares, and one or more remote NFS clients mount and access exported shares within their own file system.

Here is **how to set up an NFS share between an NFS server and a client, both running on CentOS 7**.

Preparation

In this tutorial, two CentOS 7 hosts will be used, one for an NFS server and the other for an NFS client. Many read/write operations in NFS involve accessing and updating timestamps, but NFS itself does not have any mechanism to synchronize time between a server and a client. For reliable NFS operations, it is thus highly recommended to set up NTP on every NFS server and client host to avoid any clock skew among them.

Set up an NFS Server on CentOS 7

Install necessary NFS daemons and utilities with `yum`:

```
 $ sudo yum install nfs-utils
```

Next prepare a folder to export via NFS. In this example, I am going to export /var/nfs of local filesystem.

```
 $ sudo mkdir /var/nfs
$ sudo chmod 777 /var/nfs
```

Now open /etc/exports with a text editor, and specify this folder with additional export options as follows.

```
  /var/nfs    192.168.1.0/24(rw,sync,no_subtree_check)
```

The export options used indicate the following. Obviously you can customize the options to meet your requirements.

- **192.168.1.0/24:** Only NFS clients with IP addresses from 192.168.1.0/24 can access the NFS share. Other forms of ACLs (e.g., 192.168.1.175, *.xmodulo.com) are also possible.
- **rw:** This option allows NFS clients to perform both read and write on the share.
- **sync:** This option makes the NFS server reply to a write request only after the write is successfully committed to the storage, which improves reliability at the cost of performance.
- **no_subtree_check:** This option disables subtree checking which verifies that an accessed file has correct permissions and belongs to the exported tree. Disabling subtree checking can improve performance at the cost of security.

Next, start necessary services and set them to auto-start upon boot.

```
 $ sudo systemctl enable rpcbind
$ sudo systemctl start rpcbind
$ sudo systemctl enable nfs-server
$ sudo systemctl start nfs-server
```

If you modify /etc/exports, you need to activate the change by running:

```
 $ sudo exportfs -a
```

Finally, you should open up the port used by NFS service in the default firewall as follows.

```
 $ sudo firewall-cmd --zone=public --add-service=nfs --permanent
$ sudo firewall-cmd --zone=public --add-service=rpc-bind --permanent
$ sudo firewall-cmd --zone=public --add-service=mountd --permanent
$ sudo firewall-cmd --reload
```

To check if the folder is successfully exported, run:

```
 $ sudo exportfs
```

```
  /var/nfs       192.168.1.0/24
```

The output should indicate what folders (e.g., /var/nfs) are exported to whom (e.g., 192.168.1.0/24).

Set up an NFS Client on CentOS 7

Now let's see how we can set up a remote NFS client to mount and access the exported NFS share.

On a separate CentOS 7 host, install necessary NFS daemons and utilities.

```
 $ sudo yum install nfs-utils
```

Prepare a local mount point for an NFS share.

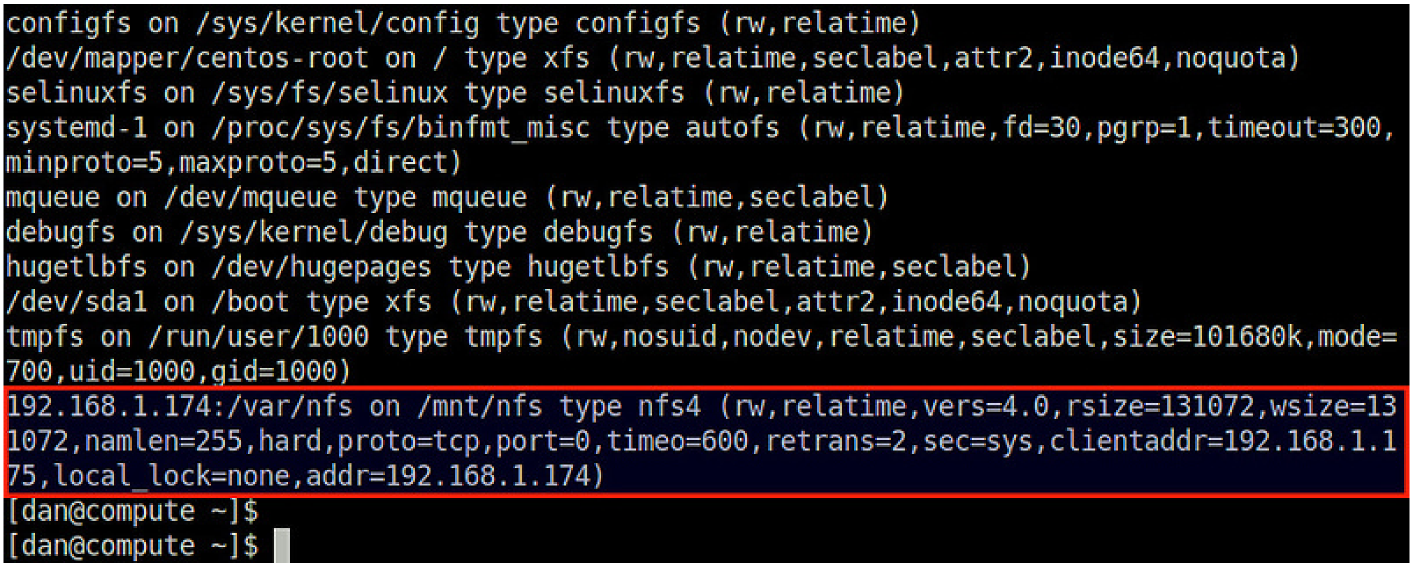
```
 $ sudo mkdir /mnt/nfs
```

Go ahead and mount a remote NFS share with `mount` command as follows. In this example, the NFS server's IP address is 192.168.1.174.

```
 $ sudo mount -t nfs 192.168.1.174:/var/nfs /mnt/nfs
```

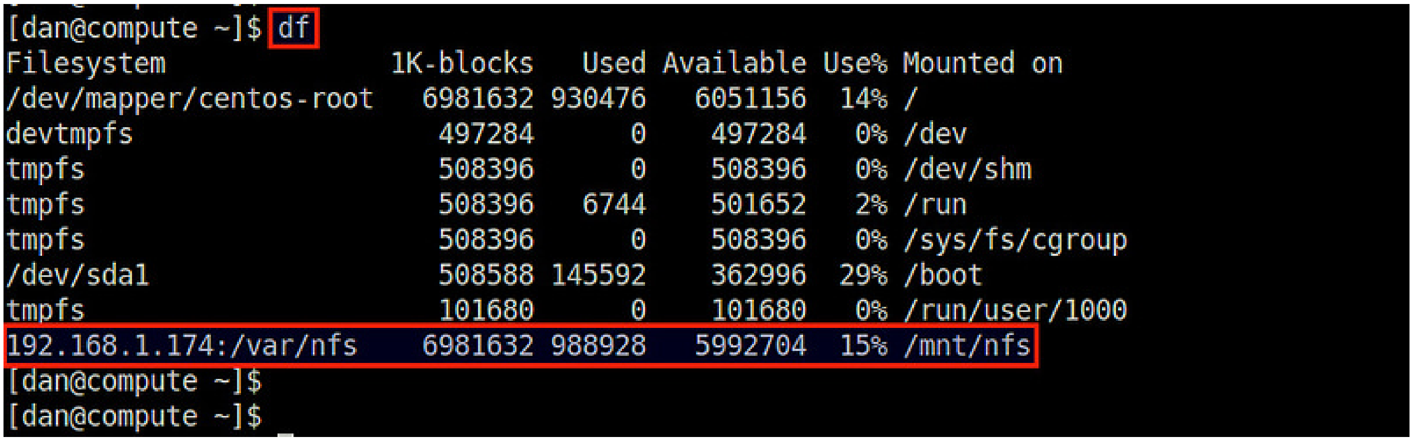
To verify that a remote NFS share is successfully mounted, run `mount` command without any argument, and check if the output contains the mounted NFS share.

```
 $ mount
```



```
configfs on /sys/kernel/config type configfs (rw,relatime)
/dev/mapper/centos-root on / type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
selinuxfs on /sys/fs/selinux type selinuxfs (rw,relatime)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=30,pgrp=1,timeout=300,
minproto=5,maxproto=5,direct)
mqueue on /dev/mqueue type mqueue (rw,relatime,seclabel)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,seclabel)
/dev/sda1 on /boot type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,seclabel,size=101680k,mode=
700,uid=1000,gid=1000)
192.168.1.174:/var/nfs on /mnt/nfs type nfs4 (rw,relatime,vers=4.0,rsize=131072,wsize=13
1072,namlen=255,hard,proto=tcp,port=0,timeo=600,retrans=2,sec=sys,clientaddr=192.168.1.1
75,local_lock=none,addr=192.168.1.174)
[dan@compute ~]$
[dan@compute ~]$
```

The `df` command should show extra storage from the NFS share.



```
[dan@compute ~]$ df
Filesystem              1K-blocks   Used Available Use% Mounted on
/dev/mapper/centos-root   6981632 930476   6051156  14% /
devtmpfs                   497284      0    497284   0% /dev
tmpfs                      508396      0    508396   0% /dev/shm
tmpfs                      508396   6744    501652   2% /run
tmpfs                      508396      0    508396   0% /sys/fs/cgroup
/dev/sda1                  508588 145592    362996  29% /boot
tmpfs                      101680      0    101680   0% /run/user/1000
192.168.1.174:/var/nfs    6981632 988928   5992704  15% /mnt/nfs
[dan@compute ~]$
[dan@compute ~]$ _
```

To unmount the NFS share, use `umount` command.

```
$ sudo umount /mnt/nfs
```

If you want to have a remote NFS share automatically mounted upon boot, you can add the following line in /etc/fstab.

```
$ sudo vi /etc/fstab
```

```
192.168.1.174:/var/nfs    /mnt/nfs  nfs defaults 0 0
```

Note that the mounted NFS share changes the ownership of any file created by the `root` to the `nfsnobody` user. This is an NFS feature called "root squash", which essentially reduces the access right of the remote superuser for security reasons. You can disable root squash by using **no_root_squash** export option, but is not recommended as any (malicious) remote root user would then be authorized to change any file in shared folders with root privilege.