NFS Server configuration

Configure the server

Step 1: Start and enable the newly-installed nfs-utils service.

[tcarrigan@rhel ~]$ sudo systemctl start nfs-server.service

[tcarrigan@rhel ~]$ sudo systemctl enable nfs-server.service

Step 2: Confirm the nfs-server service is up and running.

[tcarrigan@rhel ~]$ sudo systemctl status nfs-server.service

● nfs-server.service - NFS server and services

   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service*; enable>*

   Active: active (exited) since Wed 2020-06-24 12:50:23 EDT*; 18min >*

 Main PID: 61026 (code=exited, status=0/SUCCESS)

    Tasks: 0 (limit: 50657)

   Memory: 0B

   CGroup: /system.slice/nfs-server.service

Jun 24 12:50:23 server.example.com systemd[1]: Starting NFS server a>

Jun 24 12:50:23 server.example.com systemd[1]: Started NFS server an>

lines 1-10/10 (END)

Step 3: Verify the NFS version (you can see this information in column two).

[tcarrigan@rhel ~]$ rpcinfo -p | grep nfs

    100003    3   tcp   2049  nfs

    100003    4   tcp   2049  nfs

    100227    3   tcp   2049  nfs\_acl

\*Note that you can find the NFS daemon configuration files at /etc/nfs.conf. You can also find the config file for the mount at /etc/nfsmount.conf.

The NFS service is now up and running on your server. Next, let's create an NFS share.

Create and export the share

First, we need to designate a folder for sharing. Since one doesn't already exist on my system, I will create a directory to share.

[tcarrigan@rhel ~]$ sudo mkdir -p /test/nfs\_share/docs

Now, I learned from some trial and error and then well-written resources that you can avoid a lot of headache by changing the permissions and ownership to match the following:

[tcarrigan@rhel ~]$ sudo chown -R nobody: /test/nfs\_share/docs/

[tcarrigan@rhel ~]$ sudo chmod -R 777 /test/nfs\_share/docs/

\*Note: you may not be able to do this in a production environment due to security considerations. Be sure that you know what you are doing before you remove all restrictions from a file or directory.

Next, we need to create an /etc/exports file.

[root@server docs]# vi /etc/exports

Make the following entry in the new file:

/test/nfs\_share   172.25.1.0/24**(rw,sync,no\_all\_squash,root\_squash)**

To better understand the parameters used here, let's break them down one by one.

* rw - Allows us to read and write to the NFS share.
* sync - Requires writing of changes to the disk before any other operations are completed.
* no\_all\_squash - Maps all UIDs and GIDs from the client request to the identical UIDs and GIDs on the NFS server.
* root\_squash - Maps requests from the client-side root user to an anonymous UID/GID.

Now that we have created the share, let's export it to the client(s).

[root@server docs]# exportfs -rav

exporting 172.25.1.0/24:/test/nfs\_share

Notice that I mapped the entire subnet here. You can include only a single IP or hostname here if you prefer.

Modify the firewall

We installed the server, and then created and exported the share. Next, we configure a tunnel through the firewall. We will be adding rules for nfs, rpc-bind, and mountd. Don't forget to reload the firewall config when completed.

Seen here:

**[root@server]#** firewall-cmd --permanent --add-service=nfs

success

**[root@server]#** firewall-cmd --permanent --add-service=rpc-bind

success

**[root@server]#** firewall-cmd --permanent --add-service=mountd

success

**[root@server]#** firewall-cmd --reload

success

With the server-side completed, we can now focus our attention on the client machine.

Configure the client

\*Note: all future steps are carried out on the client machine.

Since we already updated our system and installed the nfs-utils package, this should be pretty straightforward.

We start by creating an entry in /etc/hosts for the NFS server. It should look similar to this:

**[root@client]#** cat /etc/hosts

127.0.0.1  localhost

::1        localhost

172.25.1.5 localhost

Now, let's see if anything is shared from the NFS server. If you followed along in the earlier sections, you should see /test/nfs\_share/docs as a shared directory.

[root@client ~]# showmount --exports nfs-server

Export list for nfs-server:

/test/nfs\_share/docs 172.25.1.0/24

Next, create a directory on the client machine to mount the remote share.

[tcarrigan@client ~]$ sudo  mkdir p /test/client\_share

Now that we have created a mount directory, let's mount the share.

[tcarrigan@client ~]$ sudo mount -t nfs 172.25.1.5:/test/nfs\_share/docs /home/tcarrigan/test/client\_share

Run the following command to verify the share:

[tcarrigan@client ~]$ sudo mount | grep -i nfs

Finally, to ensure that the mount is persistent across reboots, add the following line to the /etc/fstab file:

172.25.0.5:/test/nfs\_share/docs /home/tcarrigan/test/client\_share  nfs defaults 0 0

Easy day.

Proof of concept

As a culmination of our efforts, let's test the configured share. Create a file on the server in /test/nfs\_share/docs named test\_doc.

[tcarrigan@server docs]$ ls -lrt

total 4

-rw-r--r--. 1 root root 39 Jun 25 16:21 test\_doc

Let's see if our test\_doc is exported to our client machine via NFS.

On the client machine:

[tcarrigan@client ~]$ cd test/client\_share/

[tcarrigan@client client\_share]$ ls

docs

[tcarrigan@client client\_share]$ ls docs/

test\_doc

[tcarrigan@client client\_share]$

Here we see the test\_doc exists on the NFS Server.

To test in the other direction, I create a file on the client named client-test-doc.

[tcarrigan@client docs]$ vi client-test-doc

Let's jump over to the server and see if we can view the newly created file.

NFS server:

[tcarrigan@server docs]$ ls -lrt

   total 8

   -rw-r--r--. 1 root      root      39 Jun 25 16:21 test\_doc

   -rw-rw-r--. 1 tcarrigan tcarrigan  5 Jul  6 13:25 client-test-doc

We can see both the original file test\_doc as well as the newly-created file client-test-doc.