RootUsers Guides, tutorials, reviews and news for System Administrators.

How To Provide NFS Network Shares to Specific Clients

Posted by Jarrod on October 20, 2015 Go to comments Leave a comment (0)

With NFS we can export specific directories within a file system over the network to other clients allowing us to share various files over the network. It is important to configure this properly and secure it as much as possible so that only the required clients have access to the NFS share, otherwise it may be possible for anyone to mount it and access the data.

To do this we are going to use the /etc/exports file on the NFS server and lock down shares to only be accessible by specific IP addresses.

If you're interested to see how different NFS versions perform be sure to check out our NFS benchmarks.

Example Environment

Here is a list of our servers that we will be testing with.

- NFS Client: 192.168.0.100 – This Linux client will mount a directory from the NFS server.
- NFS Server: 192.168.0.200 – This Linux server will serve a directory over NFS.

NFS Configuration

The server that has the data to share will act as the NFS server and needs the nfs-utils package installed.

```
yum install nfs-utils -y
```

Once installed we can enable our NFS server to automatically start the required NFS service on boot, we'll also start the service up now as it's not running by default after installation.

```
systemctl enable nfs
systemctl start nfs
```

For further information on basic service management with systemctl, see our guide here.

Next the firewall must be configured in order to correctly allow NFS traffic through, this can be done as shown with firewalld. This change will allow TCP port 2049 NFS traffic into the server from any source. The firewall configuration must also be reloaded as we have put a permanent rule in place which will not apply to the running configuration.

```
firewall-cmd --permanent --add-service=nfs
firewall-cmd --reload
```

We'll also create the directory on the NFS server that we are going to share over NFS, in this example it's going to be /root/nfs however this can be elsewhere.

```
mkdir /root/nfs
```

The NFS server mount points are configured with the /etc/exports file, this file lists the directories that are available to be accessed over NFS. Alternatively configuration files can also be created within the /etc/exports.d/ directory as long as they have the .exports extension.

Below is an example NFS configuration within the /etc/exports file.

```
[root@server ~]# cat /etc/exports
/root/nfs       192.168.0.100(rw,async)
```

As shown we are saying that the /root/nfs directory is available only to the IP address of 192.168.0.100, so only the system at this IP address will be able to successfully access and mount the directory. Hostnames can also be used instead of IP addresses.

It is important to note that there is no space between the IP address and the options (rw,async), if there was a space here, then the IP address would have default options and the (rw,async) would instead apply to any other client that attempts to access the NFS share, which would essentially give read/write access to anyone.

After any changes to the /etc/exports file we need to use the exportfs command to update the table of exported NFS file systems.

```
exportfs -arv
```

The -a flag will export all directories, the -r flag will reexport all directories and remove any old entries, while the -v flag provides verbosity and will output all of the NFS exports.

Mounting the NFS Share

Now that we have prepared the NFS share, only 192.168.0.100 should be able to mount it, let’s test. The client systems will also need the nfs-utils package installed to be able to mount NFS.

First on the client we can use the showmount command to view a list of mounts exported on the NFS server.

```
[root@client ~]# showmount -e 192.168.0.200
Export list for 192.168.0.200:
/root/nfs 192.168.0.100
```

We can attempt to mount this with NFS with the mount command, the free space in the mount should then show up with the df command.

```
[root@client ~]# mount -t nfs 192.168.0.200:/root/nfs /mnt
[root@client ~]# df -h
Filesystem              Size  Used Avail Use% Mounted on
/dev/sda3                18G  4.3G   14G  25% /
devtmpfs                905M     0  905M   0% /dev
tmpfs                   914M   80K  914M   1% /dev/shm
tmpfs                   914M  8.9M  905M   1% /run
tmpfs                   914M     0  914M   0% /sys/fs/cgroup
/dev/sda1               297M  148M  150M  50% /boot
192.168.0.200:/root/nfs   18G  4.3G   14G  25% /mnt
```

This mount is not persistent, if we perform a reboot of the client system this will not be mounted automatically, to do this we can add the following into the /etc/fstab file on the client server.

```
192.168.0.200:/root/nfs     /mnt    nfs rw,async   0       0
```

After saving your changes to this file, you can first unmount /mnt with ‘umount /mnt’ and then run ‘mount -a’ which will attempt to mount everything not mounted already in /etc/fstab, this should mount 192.168.0.200:/root/nfs back to /mnt and this will also automatically happen on reboot.

If any other system other than 192.168.0.100 tries to mount this they will receive the below error after the mount fails.

```
mount.nfs: access denied by server while mounting 192.168.0.200:/root/nfs
```

Note that this is only limiting the clients by IP address based on the configuration within the /etc/exports file on the NFS server. There is not actually any authentication performed, for that we can use Kerberos enabled NFS exports. Kerberos can also provide encryption, as by default any content transferred over NFS will be sent in clear text which is not secure.

Summary

As shown by specifying the allowed client IP addresses within the /etc/exports NFS configuration file we can limit access to only those clients and deny all other access, essentially locking down access to the file shares. To defend against incorrectly exports configuration the firewall could also be modified to only allow NFS traffic in from specific client systems that require access to the NFS server. To provide authentication and encryption we would need to look at implementing Kerberos.

This post is part of our Red Hat Certified Engineer (RHCE) exam study guide series. For more RHCE related posts and information check out our full RHCE study guide.

Share this:

1

How To, Linux, Security Linux, NFS, RHCE, Security

← Configure Postfix to Forward Mail to a Central Relay Server

How To Create Static Routes to Route IP Traffic →

Leave a comment ? 0 Comments.

Leave a Comment

NOTE - You can use these HTML tags and attributes:
`<a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote cite=""> <cite> <code> <del datetime=""> <em> <i> <q cite=""> <s> <strike> <strong>`

NAME

EMAIL

Website URL

□ Notify me of follow-up comments by email.

□ Notify me of new posts by email.

SUBMIT