7. Troubleshooting

This is intended as a step-by-step guide to what to do when things go wrong using NFS. Usually trouble first rears its head on the client end, so this diagnostic will begin there.

7.1. Unable to See Files on a Mounted File System

First, check to see if the file system is actually mounted. There are several ways of doing this. The most reliable way is to look at the file `/proc/mounts`, which will list all mounted filesystems and give details about them. If this doesn't work (for example if you don't have the `/proc` filesystem compiled into your kernel), you can type **mount -f** although you get less information.

If the file system appears to be mounted, then you may have mounted another file system on top of it (in which case you should unmount and remount both volumes), or you may have exported the file system on the server before you mounted it there, in which case NFS is exporting the underlying mount point (if so then you need to restart NFS on the server).

If the file system is not mounted, then attempt to mount it. If this does not work, see Symptom 3.

7.2. File requests hang or timeout waiting for access to the file

This usually means that the client is unable to communicate with the server. See Symptom 3 letter b.

7.3. Unable to mount a file system

There are two common errors that mount produces when it is unable to mount a volume. These are:

1. failed, reason given by server: *Permission denied*

   This means that the server does not recognize that you have access to the volume.

   - Check your `/etc/exports` file and make sure that the volume is exported and that your client has the right kind of access to it. For example, if a client only has read access then you have to mount the volume with the **`ro`** option rather than the **`rw`** option.
   - Make sure that you have told NFS to register any changes you made to `/etc/exports` since starting **nfsd** by running the **exportfs** command. Be sure to type **exportfs -ra** to be extra certain that the exports are being re-read.
   - Check the file `/proc/fs/nfs/exports` and make sure the volume and client are listed correctly. (You can also look at the file `/var/lib/nfs/xtab` for an unabridged list of how all the active export options are set.) If they are not, then you have not re-exported properly. If they are listed, make sure the server recognizes your client as being the machine you think it is. For example, you may have an old listing for the client in /etc/hosts that is throwing off the server, or you may not have listed the client's complete address and it may be resolving to a machine in a different domain. One trick is login to the server from the client via **ssh** or **telnet**; if you then type **who**, one of the listings should be your login session and the name of your client machine as the server sees it. Try using this machine name in your `/etc/exports` entry. Finally, try to **ping** the client from the server, and try to **ping** the server from the client. If this doesn't work, or if there is packet loss, you may have lower-level network problems.
   - It is not possible to export both a directory and its child (for example both `/usr` and `/usr/local`). You should export the parent directory with the necessary permissions, and all of its subdirectories can then be mounted with those same permissions.

2. *RPC: Program Not Registered* (or another "RPC" error)

   This means that the client does not detect NFS running on the server. This could be for several reasons.

- First, check that NFS actually is running on the server by typing **rpcinfo -p** on the server. You should see something like this:

```
program vers proto   port
          100000    2   tcp    111  portmapper
          100000    2   udp    111  portmapper
          100011    1   udp    749  rquotad
          100011    2   udp    749  rquotad
          100005    1   udp    759  mountd
          100005    1   tcp    761  mountd
          100005    2   udp    764  mountd
          100005    2   tcp    766  mountd
          100005    3   udp    769  mountd
          100005    3   tcp    771  mountd
          100003    2   udp   2049  nfs
          100003    3   udp   2049  nfs
          300019    1   tcp    830  amd
          300019    1   udp    831  amd
          100024    1   udp    944  status
          100024    1   tcp    946  status
          100021    1   udp   1042  nlockmgr
          100021    3   udp   1042  nlockmgr
          100021    4   udp   1042  nlockmgr
          100021    1   tcp   1629  nlockmgr
          100021    3   tcp   1629  nlockmgr
          100021    4   tcp   1629  nlockmgr
```

  This says that we have NFS versions 2 and 3, rpc.statd version 1, network lock manager (the service name for **rpc.lockd**) versions 1, 3, and 4. There are also different service listings depending on whether NFS is travelling over TCP or UDP. UDP is usually (but not always) the default unless TCP is explicitly requested.

  If you do not see at least *portmapper*, *nfs*, and *mountd*, then you need to restart NFS. If you are not able to restart successfully, proceed to Symptom 9.

- Now check to make sure you can see it from the client. On the client, type **rpcinfo -p`server`** where server is the DNS name or IP address of your server.

  If you get a listing, then make sure that the type of mount you are trying to perform is supported. For example, if you are trying to mount using Version 3 NFS, make sure Version 3 is listed; if you are trying to mount using NFS over TCP, make sure that is registered. (Some non-Linux clients default to TCP). Type **`man rpcinfo`** for more details on how to read the output. If the type of mount you are trying to perform is not listed, try a different type of mount.

  If you get the error *No Remote Programs Registered*, then you need to check your `/etc/hosts.allow` and `/etc/hosts.deny` files on the server and make sure your client actually is allowed access. Again, if the entries appear correct, check `/etc/hosts` (or your DNS server) and make sure that the machine is listed correctly, and make sure you can ping the server from the client. Also check the error logs on the system for helpful messages: Authentication errors from bad `/etc/hosts.allow` entries will usually appear in `/var/log/messages`, but may appear somewhere else depending on how your system logs are set up. The man pages for *syslog* can help you figure out how your logs are set up. Finally, some older operating systems may behave badly when routes between the two machines are asymmetric. Try typing **tracepath [server]** from the client and see if the word "asymmetric" shows up anywhere in the output. If it does then this may be causing packet loss. However asymmetric routes are not usually a problem on recent linux distributions.

  If you get the error *Remote system error - No route to host*, but you can ping the server correctly, then you are the victim of an overzealous firewall. Check any firewalls that may be set up, either on the server or on any routers in between the client and the server. Look at the man pages for **ipchains**, **netfilter**, and **ipfwadm**, as well as the IPChains-HOWTO and the Firewall-HOWTO for help.

7.4. I do not have permission to access files on the mounted volume

This could be one of two problems. If it is a write permission problem, check the export options on the server by looking at `/proc/fs/nfs/exports` and make sure the filesystem is not exported read-only. If it is you will need to re-export it read/write (don't forget to run **exportfs -ra** after editing `/etc/exports`). Also, check `/proc/mounts` and make sure the volume is mounted read/write (although if it is mounted read-only you ought to get a more specific error message). If not then you need to re-mount with the **`rw`** option.

The second problem has to do with username mappings, and is different depending on whether you are trying to do this as root or as a non-root user.

If you are not root, then usernames may not be in sync on the client and the server. Type **id [user]** on both the client and the server and make sure they give the same *UID* number. If they don't then you are having problems with NIS, NIS+, rsync, or whatever system you use to sync usernames. Check group names to make sure that they match as well. Also, make sure you are not exporting with the **`all_squash`** option. If the user names match then the user has a more general permissions problem unrelated to NFS.

If you are root, then you are probably not exporting with the **`no_root_squash`** option; check `/proc/fs/nfs/exports` or `/var/lib/nfs/xtab` on the server and make sure the option is listed. In general, being able to write to the NFS server as root is a bad idea unless you have an urgent need -- which is why Linux NFS prevents it by default. See Section 6, “Security and NFS” for details.

If you have root squashing, you want to keep it, and you're only trying to get root to have the same permissions on the file that the user *nobody* should have, then remember that it is the server that determines which uid root gets mapped to. By default, the server uses the *UID* and *GID* of nobody in the `/etc/passwd` file, but this can also be overridden with the **`anonuid`** and **`anongid`** options in the `/etc/exports` file. Make sure that the client and the server agree about which *UID nobody* gets mapped to.

7.5. When I transfer really big files, NFS takes over all the CPU cycles on the server and it screeches to a halt

This is a problem with the *fsync()* function in 2.2 kernels that causes all sync-to-disk requests to be cumulative, resulting in a write time that is quadratic in the file size. If you can, upgrading to a 2.4 kernel should solve the problem. Also, exporting with the **`no_wdelay`** option forces the program to use *o_sync()* instead, which may prove faster.

7.6. Strange error or log messages

1. Messages of the following format:

   ```
   Jan 7 09:15:29 server kernel: fh_verify: mail/guest permission failure, acc=4, error=13
   Jan 7 09:23:51 server kernel: fh_verify: ekonomi/test permission failure, acc=4, error=13
   ```

   These happen when a NFS *setattr* operation is attempted on a file you don't have write access to. The messages are harmless.

2. The following messages frequently appear in the logs:

   ```
   kernel: nfs: server server.domain.name not responding, still trying
   kernel: nfs: task 10754 can't get a request slot
   kernel: nfs: server server.domain.name OK
   ```

   The "can't get a request slot" message means that the client-side RPC code has detected a lot of timeouts (perhaps due to network congestion, perhaps due to an overloaded server), and is throttling back the number of concurrent outstanding requests in an attempt to lighten the load. The cause of these messages is basically sluggish performance. See Section 5, “Optimizing NFS Performance” for details.

3. After mounting, the following message appears on the client:

   ```
   nfs warning: mount version older than kernel
   ```

   It means what it says: You should upgrade your mount package and/or am-utils. (If for some reason

upgrading is a problem, you may be able to get away with just recompiling them so that the newer kernel features are recognized at compile time).

4. Errors in startup/shutdown log for **lockd**

```
nfslock: rpc.lockd startup failed
```

They are harmless. Older versions of **rpc.lockd** needed to be started up manually, but newer versions are started automatically by **nfsd**. Many of the default startup scripts still try to start up **lockd** by hand, in case it is necessary. You can alter your startup scripts if you want the messages to go away.

5. The following message appears in the logs:

```
kmem_create: forcing size word alignment - nfs_fh
```

This results from the file handle being 16 bits instead of a multiple of 32 bits, which makes the kernel grimace. It is harmless.

7.7. Real permissions don't match what's in /etc/exports

`/etc/exports` is *very* sensitive to whitespace - so the following statements are not the same:

```
/export/dir hostname(rw,no_root_squash)
/export/dir hostname (rw,no_root_squash)
```

The first will grant *hostname* **`rw`** access to `/export/dir` without squashing root privileges. The second will grant *hostname* **`rw`** privileges with root squash and it will grant everyone else read/write access, without squashing root privileges. Nice huh?

7.8. Flaky and unreliable behavior

Simple commands such as **ls** work, but anything that transfers a large amount of information causes the mount point to lock.

This could be one of two problems:

1. It will happen if you have **ipchains** on at the server and/or the client and you are not allowing fragmented packets through the chains. Allow fragments from the remote host and you'll be able to function again. See Section 5, “Optimizing NFS Performance” for details on how to do this.
2. You may be using a larger **`rsize`** and **`wsize`** in your mount options than the server supports. Try reducing **`rsize`** and **`wsize`** to 1024 and seeing if the problem goes away. If it does, then increase them slowly to a more reasonable value.

7.9. nfsd won't start

Check the file `/etc/exports` and make sure root has read permission. Check the binaries and make sure they are executable. Make sure your kernel was compiled with NFS server support. You may need to reinstall your binaries if none of these ideas helps.

7.10. File Corruption When Using Multiple Clients

If a file has been modified within one second of its previous modification and left the same size, it will continue to generate the same inode number. Because of this, constant reads and writes to a file by multiple clients may cause file corruption. Fixing this bug requires changes deep within the filesystem layer, and therefore it is a 2.5 item.