Introduction to a "Network File System" (NFS)

☞ What was life like before NFS?

Introduction to the Network File System (NFS)

☞ NFS is built on top of:

- ➳ UDP - User Datagram Protocol (unreliable delivery)
- ➳ XDR - eXternal Data Representation (machine independent data format)
- ➳ RPC - Remote Procedure Call

NFS overview

☞ Two protocols, mount and nfs: "mount" protocol establishes initial link between client and server machines

NFS overview

- NFS protocols provide a set of RPCs for remote file operations
  - Searching a directory
  - Reading a set of directory entries
  - Manipulating links and directories

NFS overview

- Accessing file attributes
- Read and writing files
- Notably missing are open() and close()

NFS overview

- There was no equivalent to traditional UNIX file table on the server side – NFS was designed to be stateless on the server side, and each request for data included a full set of arguments rather than assuming persistent state. However, this stateless wasn't all that practical and we soon ended up with caching and file handles.

NFS problems

- Performance
  - Modified data may be cached locally on the client
  - Once the cache flushes to the server, the data must be written to disk before results are returned to the client and the cache is flushed

NFS problems

- File write operation semantics
  - UNIX semantics (without NFS)
    - Writes to an open file are visible immediately to other users who have the file open at the same time
    - The file is viewed as a single resource

NFS problems

- Session semantics
  - Writes to an open file are not visible to others having it open at the same time
  - Once a file is closed the changes are visible only in the sessions opened later

NFS problems

- NFS is close to UNIX semantics, but...
  - there are two client caches: file blocks and file attributes
  - cached attributes are validated with server on an open()
  - the old biod/nfsiod process implemented read-ahead and delayed-write techniques on the client-side, but is not seen much (if at all) these days

NFS problems

- newly created files may not be visible to other sites fo
  up to 30 seconds
- it is indeterminant whether writes to a file will be
  immediately seen by other clients who have the file oper
  for reading
- If a single NFS stat() request hangs, it can hang up
  UNIX commands, like “df”!

NFS problems

- "magic cookies" (random numbers) used to short-cut future validations. Given to client from server, client can use it to re-connect whenever a server comes back up after a crash.

NFS problems

```
Network File System, FSINFO Call DH:0x75867c04
    Program Version: 3
    V3 Procedure: FSINFO (19)
    object
        length: 12
        hash: 0x75867c04
        type: Linux knfsd (new)
        version: 1
        encoding: 0 0 0
            auth_type: no authentication (0)
            fsid_type: major/minor/inode (0)
            fileid_type: root (0)
        authentication: none
        file system ID: 3,2 (inode 4112441)
            major: 3
            minor: 2
```

```
    inode: 4112441
file ID: root inode
```

NFS problems

- The original NFS protocol can be spoofed (no encryption nor authentication). The first attempts to add authentication were not all that good (see USAH p 492).
- Note that "stale cookies" can make a client hang (solution: remount the filesystem on the client to make it get a new, fresh cookie).
- RPCSEC is supposed to cure all manner of security problems, but depends on kerberos infrastructure.

What are the differences in v2 and v3?

See RFC1813 http://www.ietf.org/rfc/rfc1813.txt for a full description of v3. There is a good summary at nfs.sourceforge.net of the differences in v2 and v3:

- In v2, clients can access only 2 gigabytes of a file. In v3, much larger (64 bit)
- v3 supports larger reads and writes

What are the differences in v2 and v3?

- ☞ Idea of "Weak Cache Consistency" introduced in v3 to help detect if modifications are happening to an object (file or directory).
- ☞ Server-based access checks

What are the differences in v2 and v3?

☞ v3 supports "safe asynchronous writes", where a server is permitted to reply before it has synced data to the drive.

Starting NFS on Linux

```
[root@sophie root]# more /etc/exports
#
/home/exports   monet.cs.fsu.edu(ro,no_root_squash,insecure)

[root@sophie root]# /etc/init.d/nfs start
Starting NFS services:                                     [  OK  ]
Starting NFS quotas:                                       [  OK  ]
Starting NFS daemon:                                       [  OK  ]
Starting NFS mountd:                                       [  OK  ]

[root@sophie root]# /etc/init.d/iptables stop
Flushing firewall rules:                                   [  OK  ]
Setting chains to policy ACCEPT: filter                    [  OK  ]
Unloading iptables modules:                                [  OK  ]
```

Starting NFS on Linux

On the client side:

```
mount sophie:/etc/exports /mnt-tmp
```

Starting NFS on Linux

What is actually done when on a Linux machine when you run `/etc/init.d/nfs`

```
exportfs   # /etc/exports
rpc.rquotad
rpc.nfsd
rpc.mountd
```

Starting NFS on Solaris

```
shareall   # /etc/dfs/dfstab, not /etc/dfs/sharetab
mountd
nfsd
```

NFS Security

- ☞ Don't export to hosts for which non-trusted users have root access.
- ☞ If you don't control root on the machine then don't export the file system.
- ☞ Block NFS traffic at your router/firewall, if possible.

Tuning NFS

- You can adjust the number of nfsd
- Use nfsstat -c to see client-side NFS traffic
- Use nfsstat -s to see server-side NFS traffic

Tuning NFS

```
/usr/sbin/nfsstat -s

Server rpc stats:
calls      badcalls   badauth    badclnt    xdrcall
28         0          0          0          0
Server nfs v3:
null         getattr    setattr    lookup     access     readlink
2        7%  10      35% 0        0% 2        7% 3       10% 0        0%
read         write      create     mkdir      symlink    mknod
8       28%  0        0% 0        0% 0        0% 0        0% 0        0%
remove       rmdir      rename     link       readdir    readdirplus
0        0%  0        0% 0        0% 0        0% 0        0% 1        3%
fsstat       fsinfo     pathconf   commit
0        0%  2        7% 0        0% 0        0%
```

Tuning NFS

☞ Tuning with mount command:

- rsize=n → Set the read buffer size to n bytes.
- wsize=n → Set the write buffer size to n bytes.
- timeo=n → Set the NFS timeout to n tenths of a second.
- retrans=n → The number of NFS retransmissions.

Tuning NFS

- Tuning with sysctl command:
  - Do `sysctl -a | egrep '(r|w)mem'`
  - Increasing both net.core and net.ipv4 memory settings seems to help

Automounting

- Original implementations were buggy, and some (Ultrix) required reboots to straighten out problems.
- For most production environments, the reasons for automounting are less of an issue from server-to-server since this is not done a great deal in practice and almost never to random hosts as auto-mounting assumes; for server-to-client, this would be only a benefit where a number of distinct NFS servers needed to be accessed

on an irregular basis by a given client – not all common these days. A better solution for this problem is likely **sshfs**

Beyond NFS

☞ NFS v4 (RFC3530 – http://www.ietf.org/rfc/rfc3530.txt)

1. adds state (NFS was originally stateless)

Beyond NFS

2. file delegation – the client can work on a local copy of a file until another client requests the same file
3. multiple RPCs in a single request
4. better security with RPCSEC/GSS

Beyond NFS

5. improved ACL support
6. consolidates disparate parts into a single NFS mechanism (no longer lock, mount, stat, nfs)

Beyond NFS

☞ AFS – Andrew File System

1. was in development since the late 1980s
2. better security than NFS, but never saw the success that NFS did and seems to be on the retreat

Beyond NFS

3. AFS has been used in global configurations; Morgan Stanley, for instance, has a global AFS network (25,000+ hosts over 6 continents (good slide presentation at http://www-conf.slac.stanford.edu/AFSBestPractices/Sli
4. OpenAFS – IBM released a branch for open source development, but has dropped all commercial support

More references

A very good reference for NFS operations can be found http://nfs.sourceforge.net/nfs-howto/