NFS Version 4 Feature Set

Mike Eisler

mre@Eng.Sun.Com

OUTLINE

- Wish list items
- Summary of e-mail (nfs4-wg@sunroof.eng.sun.com) discussion
- Security Requirements
- Other wish list items

SUMMARY OF WISH LIST

- internationalisation
- facilitate operation on the Internet
- improved security semantics
- locking
- embrace non-UNIX systems
- caching
- extensibility

[Mailing list discussion/proposals are identified in *italics*]

INTERNATIONALISATION

- *Whether this is Unicode, or UTF-8, or something else, NFS should explicitly cope with non-US-ASCII character sets.*

FACILITATE OPERATION ON THE INTERNET

- Goals: reduce connections, latency
- No in-band method to query a server for exported file system list
  - *New SHAREINFO procedure that is equivalent to MOUNT's EXPORT procedure, OR*
  - *Use a magic file handle to READDIR+*
- No in-band way to negotiate security flavors
  - *Add a new error code, SEC_MISMATCH*
  - *Add a new procedure, SECINFO or create an extended attribute*

- No in-band locking
  - *Combine filing and locking in one protocol*
- WebNFS client must interpret relative URLs and re-compose with base URL to issue another base URL
  - *Add flags to LOOKUP to let clients do:*
    - ***do multi-component lookup from any directory file handle, from absolute root, or from the "floating" or virtual root.***
    - ***mount point crossing***
    - ***symbolic link following***
- WebNFS has more round trips than http
  - *A new COMPOUND procedure to combine multiple operations in one round trip, e.g.:*
    - ***COMPOUND: PUTFH 0x0123456; LOOKUP "foo"; GETFH; READ 0 32768 -->***
    - ***<-- fh of "foo", contents of file "foo"***

IMPROVED SECURITY SEMANTICS

- **Mount point crossing can result in clients using the wrong security options**
  - *a SEC_MISMATCH code is needed*
- **Existing MOUNT protocol doesn't return sufficient security parameters, such as read-only vs. read-write access, GSS-API quality of protection, security service, etc.**
  - *Use SHAREINFO or READDIR+ on magic file handle to return security parameters*

- **Existing MOUNT protocol server implementations often allow clients to circumvent file access permissions with weak authentication**
  - *Use LOOKUP as the* ***only*** *method for resolving path names into file handles*
- **No ACL support**
- **No scalable, global uid/gid name space:**
  - *Leverage principal name space of security mechanism. Return file ownership in terms of the principal name appropriate for the security flavor used in RPC request, OR,*
  - *Create a generic Identity*

LOCKING

- **Existing NLM protocol is broken in that there are situations where a client can be unaware that it has lost its lock even after a server recovers.**
  - *Apply Mogul's Spritely NFS tokens to locking, OR*
    - ***Efficient use of bandwidth during normal operation***
    - ***Scalability issue: Server recovery may require call backs to every client that had one or locks.***
  - *Apply Cheriton/Gray Leases to locking*
    - ***Simple recovery from client and server crashes***
    - ***Long leases slow server recovery***
    - ***Short leases require frequent lease updates of long held locks***

NON-UNIX SYSTEMS

[Note this section has a PC-centric slant]

- **The lack of export crossing causes problems for some clients**
  - *Add mount point crossing option to LOOKUP*
- ***COPY procedure***
  - *No known clients need this, so the consensus seems to be to remove it from the wish list*
- **Mandatory locking**
  - *Should cope with mandatory locking schemes that differ from the UNIX form*
- ***A "hidden" and "archive" bit would be useful to PCs***

- **Case Sensitivity**
  - *Allow clients to specify case-insensitive semantics on operations that manipulate path names, if the server supports it.*
- ***Also allow clients to specify wild-card semantics on path name manipulation operations, if the server supports it.***
- **fattr3 is UNIX-centric**
  - *Perhaps servers should be allowed to not return values for some fattr3 attributes, such as file ids.*

CACHING

- ***Exclude strong consistency from wish list***
- **Locking and I/O is usually inefficient in NFS because clients with locks usually don't read from and write to cache.**
  - *Give the clients the option of providing "call back hints" in lock requests:*
    - ***A client could lock the entire file and give server a hint that it is really only locking a fraction of the file. OR***
    - ***The client indicates to the server that it would like to maintain a consistent cache of the region it is locking.***

EXTENSIBILITY

- **On several occasions, it would have been convenient to just add procedures to the a version of the NFS protocol. (e.g. adding procedures to do safe asynchronous writes)**
  - *Add a GETMINORVERS procedure that returns the minor version number supported by a server, and an vector of the supported procedures.*
  - *This has been a contentious issue on the alias:*
    - ***Is it preferable to deliver small wads more frequently or large wads less frequently?***

- **Extended Attributes**
  - *“named” attributes that use string names.*
  - *“numbered” attributes that use integers to represent the attributes.*
  - *If we have extended attributes, does it make sense to always return attributes on NFS operations as NFS V3 does?*

SECURITY MECHANISMS

- **ONCRPC-WG is closing in on RPCSEC_GSS, an RPC flavor that uses GSS-API**
- **ONCRPC-WG is considering stating that Application protocol specs that reference RPCSEC_GSS must identify specific GSS-API mechanisms that are required for the Application.**
- **Keith Moore has proposed a set of requirements that is largely met by RPCSEC_GSS, GSS-API, Kerberos V5 and SPKM.**
  - Keith suggests that there be a low-infrastructure, but strong security mechanism to let “lonesome cowboys scattered at random” on the Internet securely exchange in NFS traffic.

ADDITIONAL WISH LIST ITEMS

- **Proxy servers and proxy server caches**
- **Dynamic file handles**
  - Persistent file handle model is limiting for NFS servers
- **Extensible file types**