Restart/Continue pNFS Metastripe '87

Metastripe: pNFS for Metadata

- Adds cooperating MDS servers, scale-out metadata, and parallel metadata operations to NFSv4, using new pNFS metadata layout type(s)
- Adapts and extends the proposal by Eisler, in 2010
- First presented at NFSv4 WG IETF 86, Orlando

Recap: changes from prior drafts

- Reduced layout state overhead
  - Avoid requirement to take metastripe layouts on regular files
    - Propose *stripe hints* (attribute)
      - Coarse-grained layout (file-system)
      - Deviceid hint (attribute)

Inode Striping

- Filesystem “singleton” layout
  - Holds device list and stripe pattern
  - Permits clients to request stripe hint
    - Recommended attribute
  - Used for:
    - Open, GETATTR, LOCK

CohortFS

Dentry Striping

- All other metastripe (layouts on directories)
  - Essentially just like original metastripe
  - Used for:
    - Name-based operations (CREATE)
    - Directory enumeration (in parallel)

Simplify metadata layout slightly

- Remove layout filehandles (try to)
- Simplified device model
  - Define one device structure and layout device presentation
    - always use it
  - Keep “offsets” opaque

Changes since 00

- Pranoop Ersani (NetApp) has joined the draft as a co-author
- LAYOUTCOMMIT data defined (for dentry striping)
- Improved language around stateid validity
- Protocol examples (more needed)
- Minor edits

New Items for discussion

- Layout subtyping
- Opaque data in LAYOUTGET
- Device lifecycle and LAYOUTRECALL

Layout subtyping

- Curently, there is one new layout type, LAYOUT4_METADATA, with inode striping and dentry striping sub-types
- LAYOUTGET iomode argument overload to specify a desired layout subtype

Problems

· Inode stripe hints have filesystem granularity, and are “just” devices

· Dentry striping granularity has per-file (directory) granularity and stronger logical device semantics, like ordinary pNFS

· Server implementations which support only one of the two subtypes impose a discovery penalty on clients

· Server implementations supporting both sub types still might not prefer overlapping device namespaces for both

Hypothetical

- Split inode and dentry striping layout types?
- WG feedback requested

LAYOUTGET opaque data

- The need for layout-type specific data in LAYOUTGET in support of future layout types
- But possibly not metastripe, as discussed in previous slide

Device Lifecycle and LAYOUTRECALL

- RFC 5661 (13.5) specifies that if a device used in pNFS layouts is deleted, the server SHOULD first recall all layouts using the device id
- We believe this requirement holds (at least) as written for dentry striping layouts
- We believe this requirement is not appropriate for inode striping layouts

CB_DEVICEID_NOTIFY and dentry striping layouts

- For inode striping layouts, we propose that a server implementation MAY use CB_DEVICEID_RECALL with a value of NOTIFY_DEVICEID4_DELETE when the corresponding layout is an inode striping Layout
- (Presuming that inode striping is changed to be a distinct layout type. Otherwise, if the device is ONLY used in inode striping layout[s].)

Progress on Implementation

- CohortFS has a draft implementation in progress

Current draft

http://tools.ietf.org/html/draft-mbenjamin-nfsv4-pnfs-metastripe-01

Q/A