Layout engineering

- Experimenting with pNFS layouts for new storage types and domains
  - Nit-picking, like opaque data in LAYOUTGET
  - New ways to do things we already (or wanna) do, like Ceph or Lustre layouts
  - New things that we don't do at all, like fine-grained replication control
- RFC 5661 specifies private layout type ranges for experimentation
- Requires RFC Draft for registration
  - Revisit this?

Ceph placement functions

- Ceph "CRUSH" family of placement functions mapping blocks to OSDs
  - Not periodic
  - In fact, it is psuedo-random
- Cycling over a fixed device list is inadequate to describe CRUSH placement functions
  - Explosion of devices and device info
  - In Ceph, every file (or even every file extent) might correspond to a unique device
  - Results in server creating tons of virtual devices
- This goes way beyond the need for segmented layouts
  - (And not even those are supported by the Linux client at the moment)

Stripe mapping

- In pNFS file, the striping pattern is device-specific, while it is layout-specific for block and object
- It might be interesting to experiment with file-like layouts that carry a striping pattern
  - We could support many more striping patterns without an explosion in the number of devices
  - This would help with Ceph
  - And might even obviate a Ceph layout (or might not)
  - And might relate to Lustre layouts
  - Boaz does not agree

Ceph devices and layouts

- We have been trying to avoid a native Ceph layout
  - But what we have is kind of a hack
  - And requires one or more devices for every file
  - And segmented layouts
- A native Ceph layout would allow the pNFS client to speak RADOS directly to Ceph OSDs
  - Lustre has similar issues
- In Ceph, the natural approach is to associate a device with a CRUSH map ruleset
  - Ceph layout might also want to carry additional parameters, such as snapshot info

Consistency mechanism diversity

- Today we have close-to-open
- What if we want fine-grained update consistency?
- What if we want fine-grained mutable replication?
  - Can pNFS layouts describe replication control strategies?
    - Multipathing support in devices provides a starting point (maybe)
      - Restricts replication to a single layout type (for a given file)
      - Some sort of "stacked" layout?
      - Eisler's MD striping ideas offers another way to think about this
    - Placement functions?
      - Layout must identify replication servers
      - Layout might identify consistency and integrity mechanisms
      - Or is that per FS?
    - Do range delegations play a role?
  - Looking for a way to negotiate consistency mechanisms