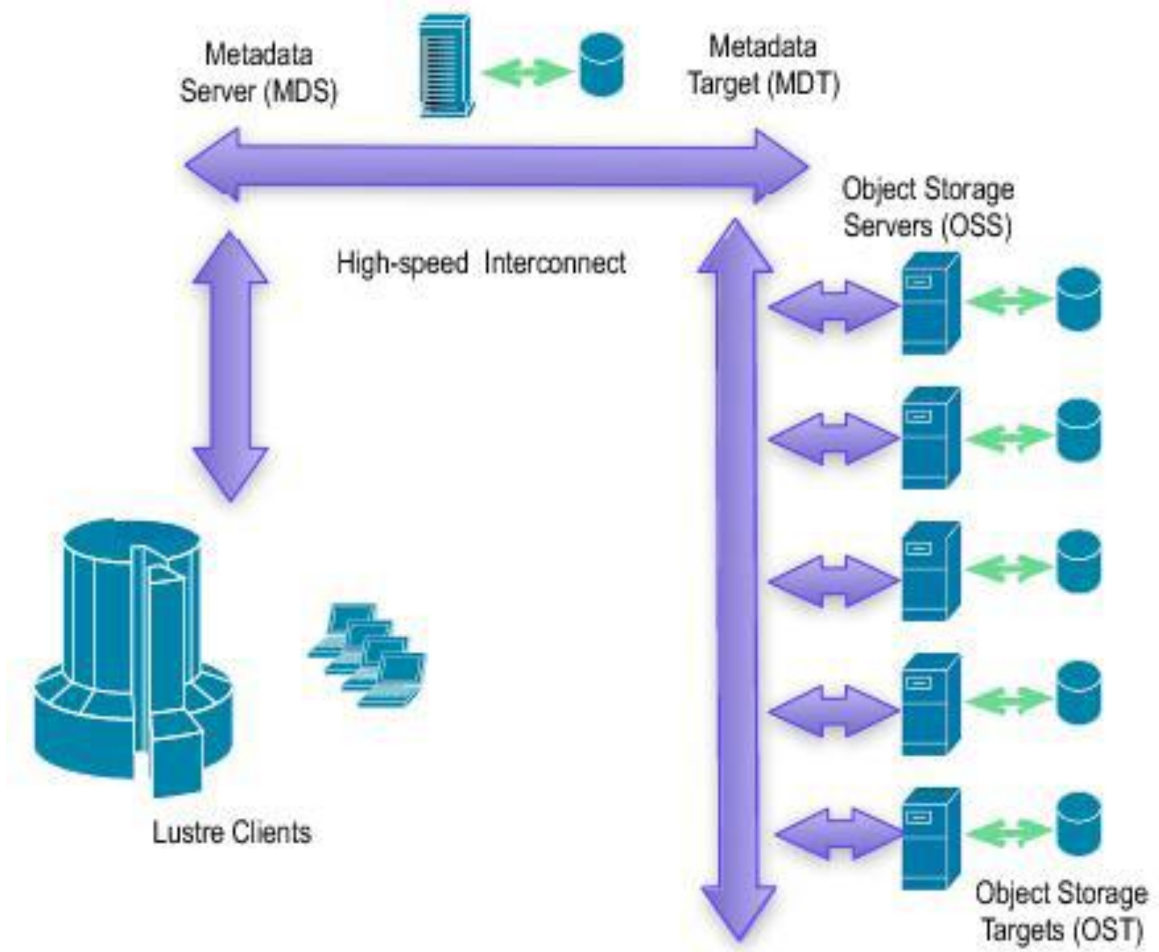
pNFS Lustre layout discussion

IETF 83, nfsv4 WG – Paris, March 28, 2012

Sorin Faibish - EMC
Peng Tao - EMC

Lustre FS architecture and layouts



Lustre layout example

- Extended Attribute EA for a file A with stripe count of three , look like:

```
EA ---> <obj id x, ost p>
        <obj id y, ost q>
        <obj id z, ost r>
        stripe_size and stripe_width
```

- For a stripe of 1MB, then this would means that

  obj x = [0,1M]), [4M,5M), on OST p;

  obj y = [1M, 2M), [5M, 6M), on OST q;

  obj z = [2M,3M), [6M,7M), on OST r.

Motivation for a pNFS Lustre layout

- Has similar behaviors as file layout from stripe structure perspective
- Has as similar behavior as object layout but not identical
- Makes sense to introduce a new layout that makes the best of both file and object
- Intention is to leave the Lustre server unchanged for data servers OSS/OST
- Extend the MDS of Lustre to support pNFS MDS operations
- Or use a new MDS for pNFS cluster (Address MD scalability of Lustre)

Lustre vs pNFS file layout

Similarities:

- Both maintain file layout information on MDS and use layout information to map file data to DS (OST for Lustre)
- Both use similar data striping patterns per file with similar granularity:
  - files on DS for pNFS
  - files as objects on OST for Lustre

Lustre vs pNFS file layout

Differences:

- Lustre layout can support OST level data redundancy like RAID.
- pNFS file layout can't by RFC5661; one unit can be mapped to only one DS list.
- Both data path protocol and control protocol between MDS and OSS/OST are different.
- Implementation wise, Lustre layout supports POSIX while pNFS file only supports close-to-open semantics

Lustre vs pNFS object layout

Similarities:

- Both use layout information to map large files onto object files on DS's:
  - OST (Lustre)
  - OSD (pNFS object)
- Both support several RAID algorithms for data redundancy

Lustre vs pNFS object layout

Differences:

- Use different data path protocols:
  - Lustre uses ptlRPC and Lustre protocol to send/receive data to DS
  - object layout is tight with OSD/OSD-2 commands
- Use different layout management
  - Lustre file extent locks are decoupled and managed by OSTs,
  - pNFS object use layouts to manage read/write permissions managed solely by MDS.
- Implementation wise:
  - Lustre layout supports POSIX while
  - pNFS object only supports close-to-open semantics

Summary and Conclusions

- Similar layout architectures are used
- Lustre decouples extents IO permission to DS, and pNFS controls it in MDS 3.
- When only close-to-open semantics are possible (POSIX may break), pNFS file and object layouts allow shared IO not caring about data lost; Lustre doesn't.
- pNFS file layout supports MDS/DS multi-pathing via NFSv41 trunking. Similarly, Lustre supports failover-pairs of MDS/OSS.
- pNFS file layout cannot support any DS level data redundancy (such as RAID1, RAID5 etc.)
- Both Lustre and object layout can support different RAID algorithms on DS level but the client is involved in the RAID in the case of objects.

Discussion

- Next steps:
  - Discussion in the nfsv4 list (started)
  - Proposal to LSF (Peng Tao)
  - Discussion with Lustre community (LUG)
  - Draft will be posted before next IETF
- Q&A