Agenda

• What is the data layout of Lustre?
• Requirement of layout lock
• Implementation details
• Use cases
What’s data layout

- Data layout determines how to place data to OSTs
- Stored as xattr trusted.lov on MDT
- Interpreted at the LOV layer
- Used to be immutable
Why do we need layout lock?

- Requirements of changing layout on the fly
  - HSM
  - Replication
  - Data on MDS
  - Restriping, a.k.a. migration

- Fundamental of all above data placement features
  - They all need to update layout
Layout lock attributes

- An IBITS lock: MDS_INODEBITS_LAYOUT
- Required to cache layout on the client
  - When layout lock is cancelled, layout will be invalidated
  - Losing layout lock usually won’t cause reconfiguration of IO stack
- IO depends on Layout lock
  - Make sure Layout is correct before IO starts
- Capability connect flags: OBD_CONNECT_LAYOUTLOCK
  - Supported in Lustre 2.4
Layout lock requesting

- Usually piggybacked by IT_GETATTR or IT_OPEN
  - Does not add overhead for extra lock RPCs
  - To make sure statahead runs quickly
  - Only granted if there is no conflicting layout lock queued
    - mdt_object_lock_try() is invented for this purpose

- Otherwise layout lock is explicitly requested by IT_LAYOUT
  - Normal DLM request handling on the MDT
  - Layout will be returned in LVB of DLM reply, or completion AST
A tricky case about layout lock

- Layout lock is an IBITS lock but mainly used in IO path
  - We can’t hold layout lock to do IO to avoid cascading problem, ...
    - Layout lock can be lost/revoked any time during an IO
    - Cascading problem: A client holds layout lock, then tries to access OSTs; If OSTs are unreachable, and layout lock is being cancelled, the client will be evicted by MDT and OST on the same time
  - But have to make sure IO is done on correct layout
    - The points are contradictory by themselves
  - Think about it carefully to make sure you’re doing the right thing
IO Handling with Layout

IO init

Refresh Layout

Do IO

Verify Layout

Send IO to OSTs

MDT

Layout Changed

MDT

No

End IO

Yes

Req. LL

Req. LL

OSTs

Send IO to OSTs

Req. LL

MDT

MDT

Resp. LL

MDT
Case study

- Layout swap – to swap layouts of two files
  - MDT operation: MDS_SWAP_LAYOUTS
  - Revoke layout lock on both files and exchange layout

- Restriping (a.k.a Migration) – to change stripe number of a file
  - Get data version of original file
  - Create temporary file with desired stripe count and OSTs
  - Copy file content to temporary file
  - Swap layouts and delete temporary file (which now has old objects)
  - From now on, all IO to original file will use new layout, even for file handles which are opening during restriping
Case study – cont.

- Data on MDS - small files on MDT and move to OSTs if too big
  - Allocate OST object(s) to form a layout
  - Migrate (small) data from MDT inode to OST(s)
  - Revoke layout lock for the update of layout

- Replication – to store the same data on multiple OSTs
  - If one replica failed to write, MDT is notified to take replica out of layout
  - Clients are notified of new layout, ignoring stale replica
  - New replica is created and layout is updated again

- HSM
  - Layout lock is needed to restore and release files
We are committed to **Moving Lustre* Forward** by enabling open, powerful, multi-vendor storage solutions for tomorrow’s data centers.

Our investment brings additional innovation to Lustre* that will accelerate its leadership in HPC and extend to Cloud.