Lustre* Features in Development

High Performance Data Division

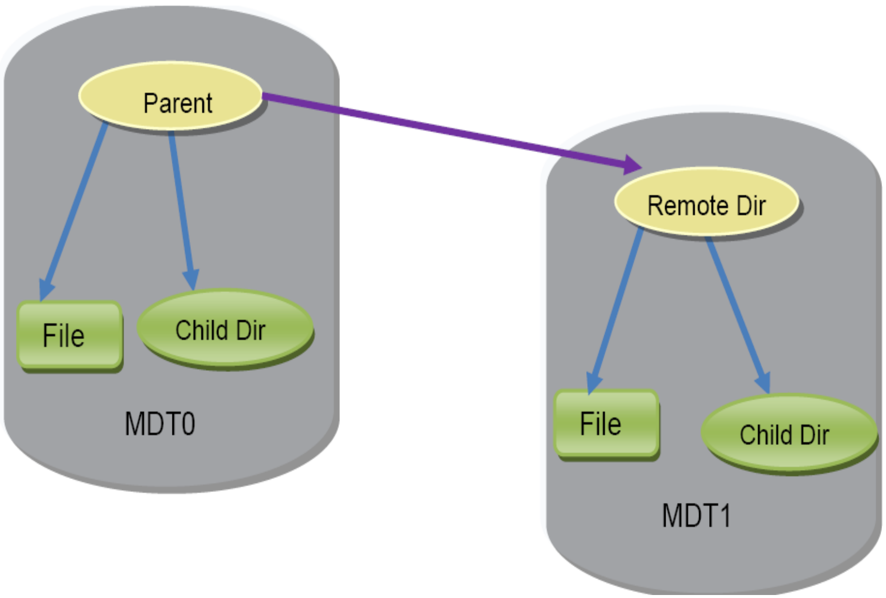
Oleg Drokin

Japan LUG

October 16th 2013

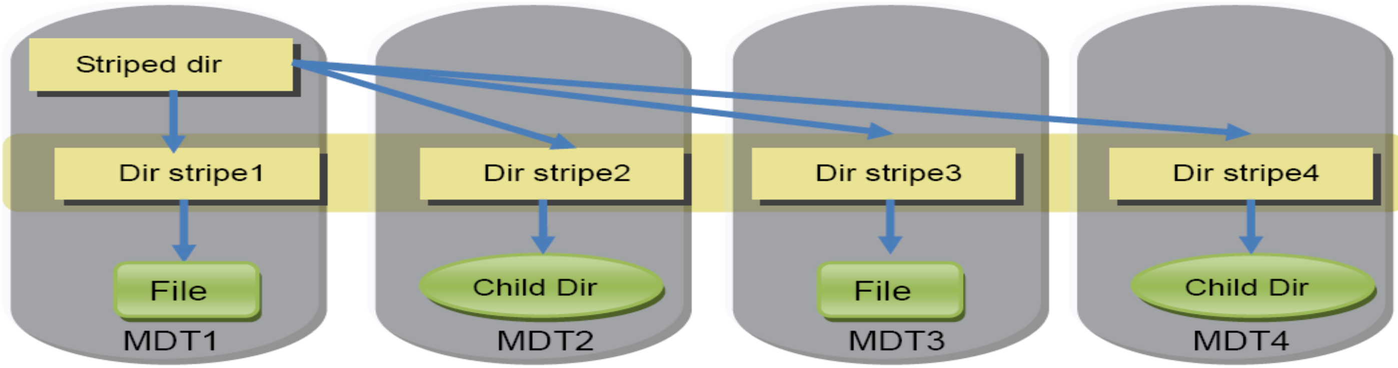
Distributed Namespace (DNE) – Phase I

- Already included in Lustre 2.4.x
- Allows for remote directories created statically
  - lfs mkdir -i <MDT_index>
- Normal files created in the same directory as the parent MDT
  - Synchronous metadata operations for cross metadata server modifications
    - Slow
  - Does not support rename from one MDT to another



Distributed Namespace (DNE) – Phase II

- Ongoing OpenSFS funded project; Intel HPDD developers
- MDTs
  - Improved performance for very large directories



- Interface and tool for migrating inodes between MDTs
  - Move inodes, directories to new MDTs for load balancing
  - Does not require copying the file data
- Asynchronous remote updates
  - Distributed operations across MDTs without 2-phase sync commits
  - Commit-on-Share on distributed updates allows recovery if client fails
  - Allows safe rename, link across MDTs

Lustre File System Check (LFSCK) – Phase II/III

- Ongoing OpenSFS funded project; Intel HPDD developers
- OST Internal Lustre consistency check/repair
  - Verify object directories (O/{seq}/d*, LAST_ID)
  - Verify parent FID xattr for each OST object
- MDT/OST consistency check/repair
  - Verify MDT layout xattr references valid objects
  - Fix duplicate/missing objects
  - Clean up orphan objects on OSTs
- MDT/MDT consistency check/repair
  - Verify DNE remote directories/files
  - Verify DNE striped directories

CLIO Simplification

- OpenSFS funded design; Intel HPDD developers
- Remove clio extent locking
  - Reduces memory usage
  - Reduces duplication of functionality of LDLM
  - Significant simplification of one of the obscure pieces of Lustre code
- Streamline IO callpath
  - Allows for faster reads and writes, even single threaded
  - Submit pages in batches from kernel to OSC

Fujitsu developments from FEFS

- Make /proc statistic optional per subsystem
  - Reduces client memory usage on large filesystems
- Opcode time execution histogram
  - Extra statistic about server operations
- OSC request pool aggregation
  - Another client memory reduction measure
- Open/replay handling rewrite
  - No longer store list of verbatim requests to be resent, reconstruct as needed
    - Memory reduction
    - Would help interoperability too
- http://www.opensfs.org/wp-content/uploads/2013/04/LUG2013-intel-Fujitsu-20130413.pdf

Shared Key Crypto

- Primary developer Indiana University; funded by OpenSFS
- Simplified node authentication and RPC encryption
  - WAN or other separate administrator domains
  - Uses existing Lustre GSSAPI/sptlrpc infrastructure from Kerberos
- Shared secret key is known by clients and servers
  - Key distribution external to Lustre (USB key, phone, (e)mail, pigeon)
  - Different keys for different client clusters
  - Servers can understand multiple keys per cluster
  - Rotate keys as needed, lifetimes can overlap
- Authenticate remote *nodes* instead of *users* like Kerberos
- Uses AES-128 encryption
  - Flexible to allow other encryption in the future

UID Mapping

- Primary developer Indiana University; funded by OpenSFS
- Multiple clusters with different UID/GID maps
  - WAN or other separate administrator domains
  - Maps are maintained only on a cluster granularity
- Remote cluster nodes defined by client NID range
  - Optionally authenticated by shared-key authentication
  - *Cluster* can be one node or a whole campus
- Map remote UID/GID to MDS-local values on MDS
  - Does not need any changes to remote clients
  - Store remote UID/GID in MDS-local rage
- Map remote UID/GID on OSS for quota

T10-PI Data Integrity

- Xyratex development
- SCSI Standard, implemented in some HBAs and disks
- Data integrity from syscall interface to disk for each sector
  - 16-bit Guard Tag (CRC or IP checksum)
  - 32-bit Reference Tag (low bits of sector address)
  - 16-bit App Tag (from application, if there was an API)
- Computed on client for each page (N sectors) on write
  - Kept with page in cache until RPC is generated
- Sent with each sector in RPC for read/write
  - Optionally in RPC request (increases request size), or with bulk data transfer
- Returned from server for each page (N sectors) on read
- Validated by peer, resend RPC on error

File Replication

- OpenSFS funded design; Intel HPDD developers
- Mirror files across multiple OSTs (RAID-1+0)
  - Redundancy in case of OST failure (disk, network, software)
  - Improved read performance of hot files
- Phase I: Read-Only Replicas
  - Copy file in userspace then merge copy onto original inode
  - Implement reads, layout, failure handling, no overhead for write
  - Different layouts for each copy (tiered storage, RAID-1+6?)
- Phase II: Synchronous File Replication
  - Send each write to multiple OSTs, wait for commit
  - Implement writes, immediate redundancy, write overhead
- Phase III: Asynchronous File Replication
  - Send each write to multiple OSTs, no waiting
  - Complex recovery model (needs DAOS features)

Small Files on MDT

- OpenSFS funded design; Intel HPDD developers
- Combine MDS and OSS into Unified Target
  - Remove duplication of common code
  - Allow client to read/write file data from MDT, index on OST
- Store small files on high-IOPS MDT storage
  - Reduce RPCs for small files (size, locks, RAID-5/6 r-m-w)
  - Migrate file data to OST if it grows too large
  - Small-file workloads may use only MDTs?

intel®