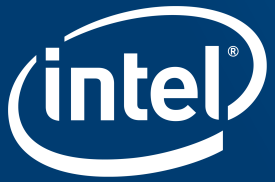


Lustre* 2.11 and Beyond

Andreas Dilger, Intel High Performance Data Division

LAD 2017

Upcoming Feature Highlights

2.11 landings in progress with several features landed or underway

- File DLM lockahead
- Data-on-MDT for improved small file performance/latency
- File Level Redundancy (FLR Phase 1 delayed resync)
- DNE directory restriping for ease of space balancing and DNE2 adoption

2.12/2.13 plans continued functional and performance improvements

- File Level Redundancy continues (FLR Phase 2 immediate resync)
- DNE directory auto-split to improve usability and performance of DNE2

LNet Dynamic Discovery (LU-9480 2.11)
LNet Network Health (LU-9120 2.12)

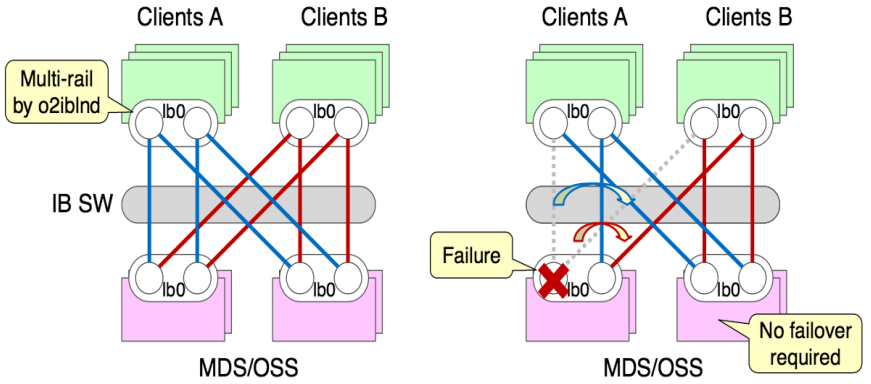
Builds on LNet Multi-Rail in Lustre 2.10 (Intel, HPE/SGI*)

LNet Dynamic Discovery

- *Automatically* configure peers that share multiple LNet networks
- Avoids need for admin to specify Multi-Rail configuration for nodes

LNet Network Health

- Detect network interface, router faults
- Handle LNet fault w/o Lustre recovery
- Restore connection when available



Data-on-MDT Small File Perf (LU-3285 Intel 2.11)

Avoid OST overhead (data, lock RPCs)

High-IOPS MDTs (mirrored SSD vs. RAID-6 HDD)

Avoid contention with streaming IO to OSTs

Prefetch file data with metadata

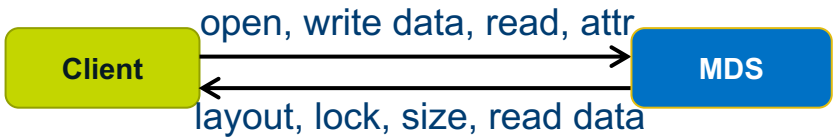
Size on MDT for small files

Integrates with PFL to simplify usage

- Start file on MDT, grow onto OSTs if larger

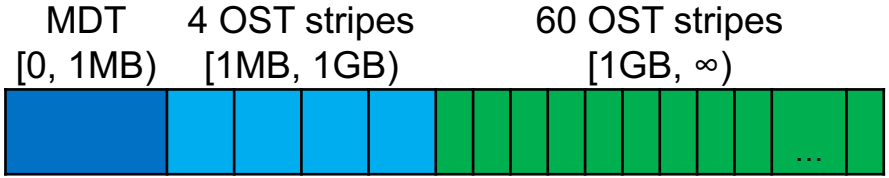
Complementary with DNE 2 striped directories

- Scale small file IOPS with multiple MDTs



Small file IO directly to MDS

Example DoM/PFL File Layout



https://jira.hpdd.intel.com/browse/LU-3285

DNE Improvements (LU-4684 Intel 2.11/2.12)

Directory migration from single to striped/sharded directories

- Rebalance space usage, improve large directory performance
- Inodes are also migrated along with directory entries

Automatic directory restriping to reduce/avoid need for explicit striping at create

- Start with single-stripe directory for low overhead in common use cases
- Add extra shards when master directory grows large enough (e.g. 32k entries)
- Existing *dir entries* stay in master, or are migrated to shards asynchronously?
- New entries+inodes created in new directory shards on MDTs to distribute load
- Performance scales as directory grows

MDT Pools for space/class management

ZFS Enhancements Related to Lustre (2.11+)

Lustre 2.10.1/2.11 `osd-zfs` updated to use ZFS 0.7.1

- File create performance (parallel lock/alloc, new APIs) (Intel)
- LFSCK ZFS OI Scrub, ldiskfs->ZFS backup/restore (LU-7585 Intel)

Features in ZFS 0.7.x

- Dynamic dnode size for better xattr performance/space (LLNL)
- Optimized parallel dnode allocation (Delphix, LLNL, Intel)
- Improved kernel IO buffers allocation (ABD) (others, Intel)
- Multi-mount protection (MMP) for improved HA safety (LLNL)
- Optimized CPU and QAT h/w checksums, parity (others, Intel)
- Better JBOD/drive handling (LEDs, auto drive resilver) (LLNL)

OpenZFS

Features for ZFS 0.8.x

- On-disk encryption (Datto)
- Project quota accounting (Intel)
- Declustered RAID (dRAID) (Intel)
- Metadata Allocation Class (Intel)
- Likely lots more...

Miscellaneous Improvements (2.11)

Client Asynchronous `ladvise` Lock Ahead (LU-6179 Cray*)

- Client (MPI-IO) to request read/write DLM locks before IO to avoid latency/contention
- MPI-I/O integration so applications can benefit from this without modification

Token Bucket Filter (NRS-TBF) Improvements (LU-9658, LU-9228 DDN*)

- Support UID/GID for TBF policies, compound policy specification
- Real-time priorities for TBF rules if server is overloaded

File access and modification auditing with ChangeLogs (LU-9727 DDN)

- Record UID/GID/NID in ChangeLog for `open()` (success or fail), other operations

Nodemap and virtualization improvements (LU-8955 DDN)

- Configure audit, SELinux, default file layout on per-nodemap basis

Upstream Kernel Client (LU-9679 ORNL)

Kernel 4.14 updated to approximately Lustre 2.8, with some fixes from Lustre 2.9

Lustre 2.10 updated to work with kernel ~4.12 (LU-9558)

Improve kernel internal time handling (LU-9019)

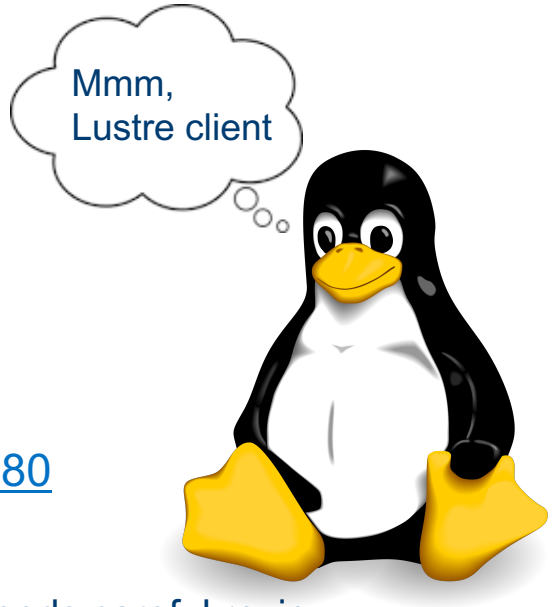
- 64-bit clean to avoid Y2038 issues
- remove `jiffies` and `cfs_time_*()` wrapper functions

Continued user header changes (LU-6401)

- Allow building user tools against upstream kernel

Kernel tracepoints for logging/debugging/perf analysis (LU-8980

- Replace `CDEBUG()` macros and Lustre kernel debug logs
- Has potential to improve (or not?) debugging of Lustre problems, needs careful review

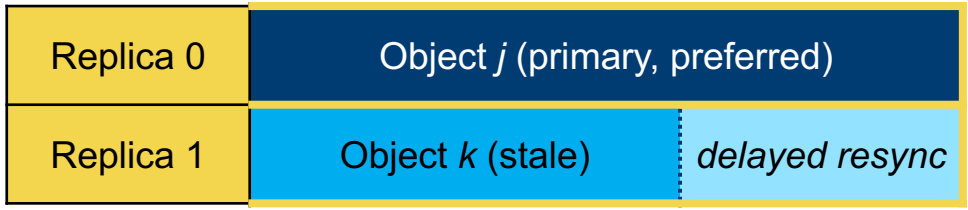


File Level Redundancy (LU-9771 Intel 2.11/2.12)

Based on Progressive File Layout (PFL) feature in Lustre 2.10 (Intel, ORNL)

Significant value and functionality added for HPC and other environments

- Optionally set on a per-file/dir basis (e.g. mirror input files and one daily checkpoint)
- Higher availability for server/network failure – finally better than HA failover
- Robustness against data loss/corruption – mirror (and later M+N erasure coding)
- Increased read speed for widely shared input files – N-way mirror over many OSTs
- Mirror/migrate files over multiple storage classes – NVRAM->SSD->HDD (e.g. Burst Buffer)
- Local vs. remote replicas (WAN)
- Partial HSM file restore
- File versioning (no resync replica)
- Many more possibilities ...



Client-Side Data Compression (LU-10026 2.12)
University Hamburg

Piecewise compression

- Compressed in 32KB chunks
- Allows sub-block read/write

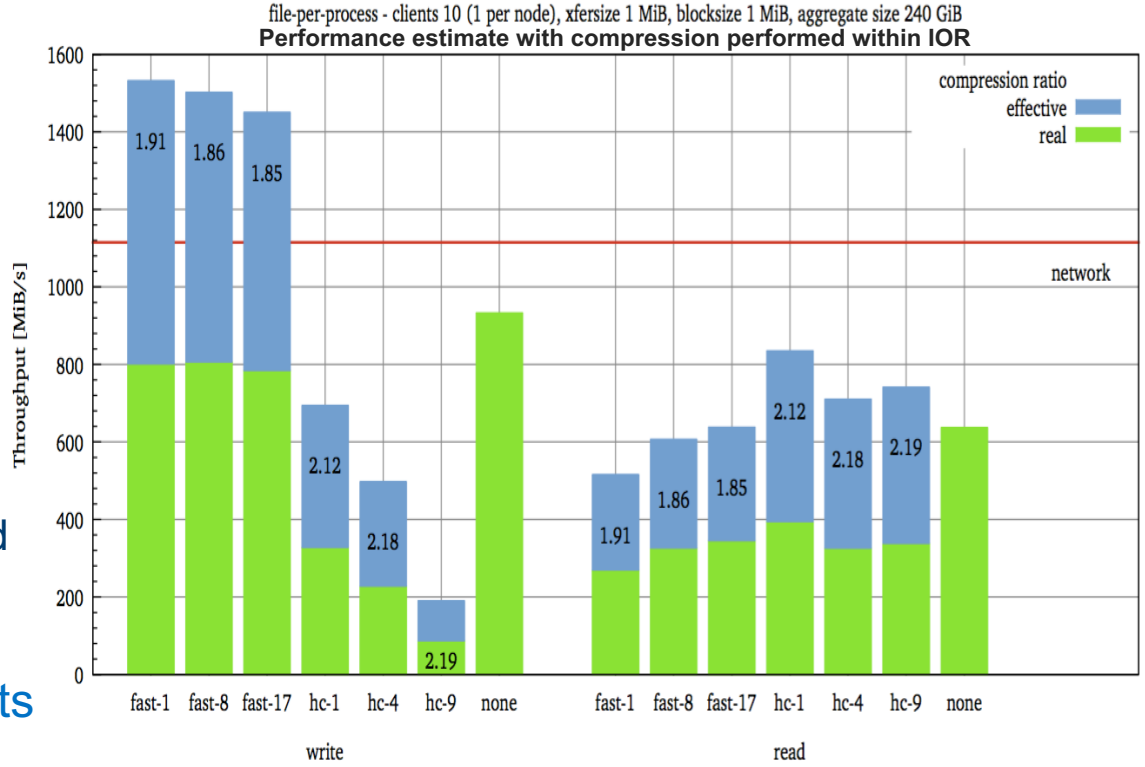
Integrated with ZFS data blocks

- Leverage per-block type/size
- Code/disk format changes needed

Avoid de-/re-compressing data

Good performance/space benefits

- Graph courtesy Uni Hamburg



Tiered storage with Composite Layouts (2.12/2.13)

Policy engine to manage migration over tiers, rebuild replicas, ChangeLogs

- Policies for pathname, user, extension, age, OST pool, mirror copies, ...
- FLR provides mechanisms for safe migration of (potentially in-use) data
- Integration with job scheduler and workflow for prestage/drain/archive

Multiple policy and scanning engines presented at LUG'17

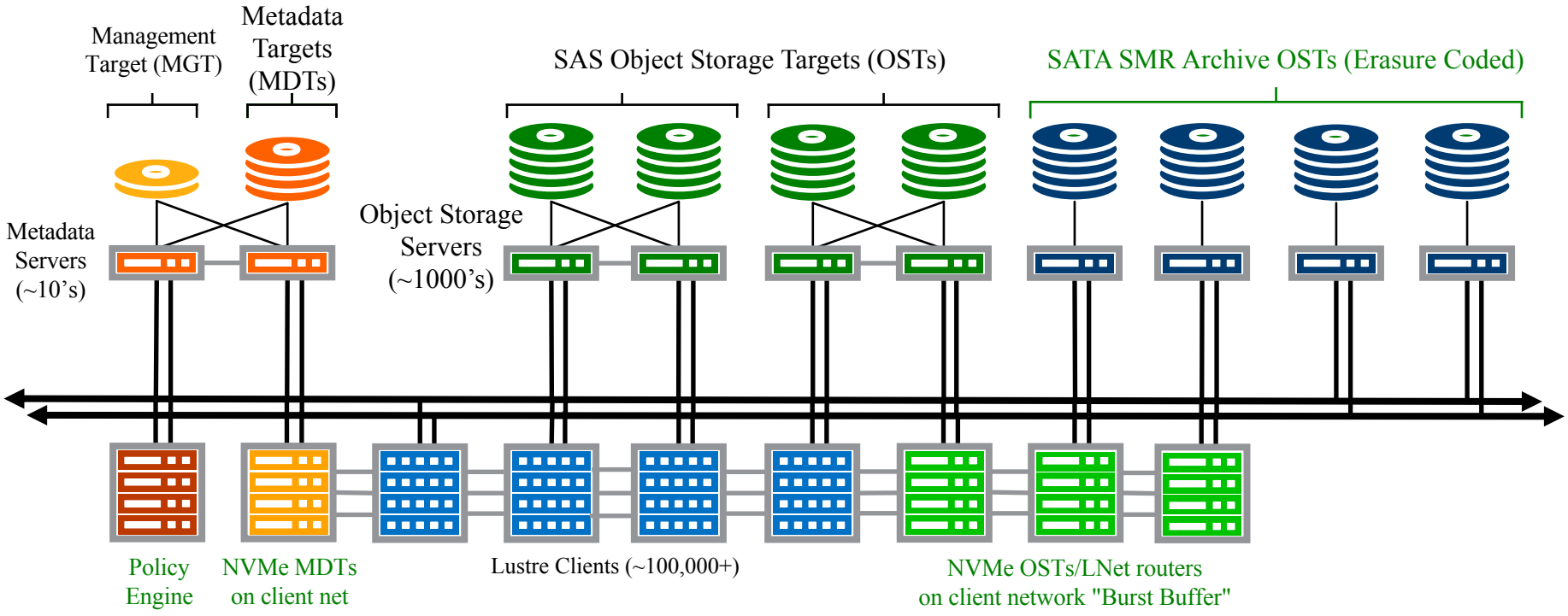
Multiple presentations on tiered storage at LAD'17

- Integrated burst buffers are a natural starting point

This is largely a userspace integration task, with some hooks into Lustre

Tiered Storage and File Level Redundancy

Data locality, with *direct access* from clients to all storage tiers as needed



Improved client efficiency (2.11/2.12+)

Small file write optimizations (LU-1575, LU-9409 Cray, Intel)

- Reduce client and RPC/server overhead for small (<= 4KB) reads/writes

Disconnect idle clients from servers (LU-7236 Intel)

- Reduce memory usage on client and server for large systems
- Reduce network pings and recovery times
- Aggregate `statfs()` RPCs on the MDS (LU-10018)

Reduce wakeups and background tasks on idle clients (LU-9660 Intel)

- Synchronize wakeups between threads/clients (per jobid?) to minimize jitter
- Still need to avoid DOS of server if all clients ping/reconnect at same time

DNE Metadata Redundancy (2.13+)

New directory layout hash for mirrored directories, mirrored MDT inodes

- Each dirent copy holds multiple MDT FIDs for inodes
- Store dirent copy on each mirror directory shard
- Name lookup on any MDT can access via any FID
- Copies of mirrored inodes stored on different MDTs

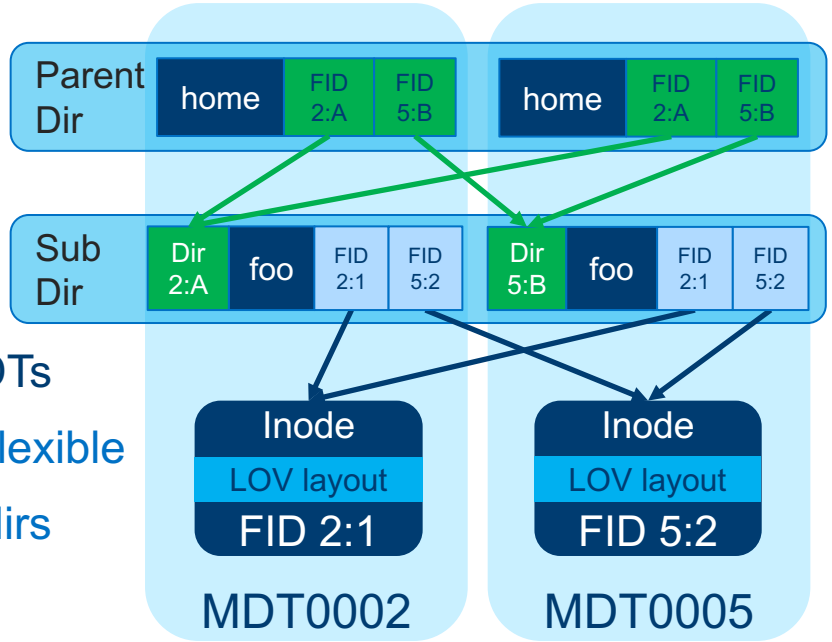
DNE2 distributed transaction for update recovery

- Ensure that copies stay in sync on multiple MDTs

Redundancy policy per-filesystem or subtree, is flexible

Flexible MDT space/load balancing with striped dirs

Early design work started, discussions ongoing



Notices and Disclaimers

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

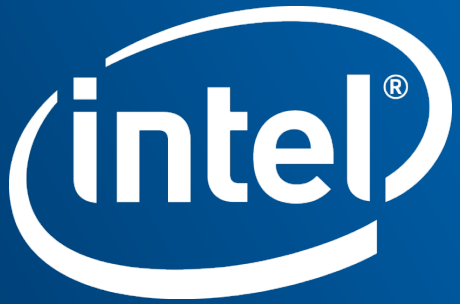
Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

No animals were harmed during the production of these slides. BPA and gluten free.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit **www.intel.com/benchmarks**


intel®