Lustre* 2.11 and Beyond

Andreas Dilger, Intel High Performance Data Division

SC 2017

Legal Notices and Disclaimers

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life-saving, life-sustaining, critical control or safety systems, or in nuclear facility applications.

Intel products may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

This document may contain information on products in the design phase of development. The information herein is subject to change without notice. Do not finalize a design with this information. Intel may make changes to dates, specifications, product descriptions, and plans referenced in this document at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure.

Intel Corporation or its subsidiaries in the United States and other countries may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

Performance estimates or simulated results based on internal Intel analysis or architecture simulation or modeling are provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice Revision #20110804.

Intel® Advanced Vector Extensions (Intel® AVX)* provides higher throughput to certain processor operations. Due to varying processor power characteristics, utilizing AVX instructions may cause a) some parts to operate at less than the rated frequency and b) some parts with Intel® Turbo Boost Technology 2.0 to not achieve any or maximum turbo frequencies. Performance varies depending on hardware, software, and system configuration and you can learn more at http://www.intel.com/go/turbo.

Intel processors of the same SKU may vary in frequency or power as a result of natural variability in the production process.

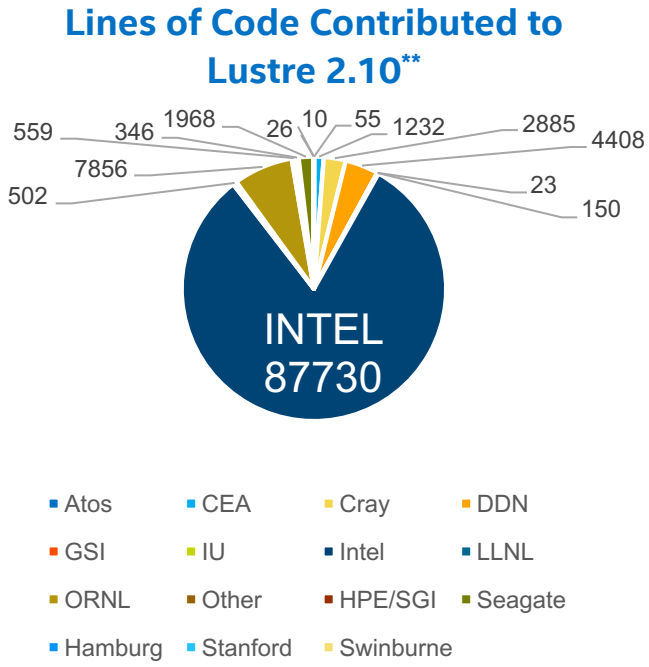
Intel, the Intel logo, 3D-Xpoint, Optane, Xeon Phi, and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

* Other names and brands may be claimed as the property of others.

Intel & Lustre* Software

Intel continues to invest in Lustre with:

- The latest **Major Releases** of Lustre
- **Next generation Feature Development** and enhancements
- Public **LTS Maintenance Releases** providing ongoing updates
- Expansion of the **Worldwide Lustre Support** ecosystem
- **Ongoing Investment** in test, validation, and release processes
- **Community Participation** in OpenSFS and EOFS



**Data courtesy of Dustin Leverman (ORNL)

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 begins (FLR Phase 1 Delayed Resync)

2.12/2.13 plans continued functional and performance improvements

- File Level Redundancy continues (FLR Phase 2 Immediate Resync)
- DNE2 directory auto-stripe to improve usability and performance
- FLR Phase 3 Erasure Coded Striped Files
- Persistent Client Cache

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

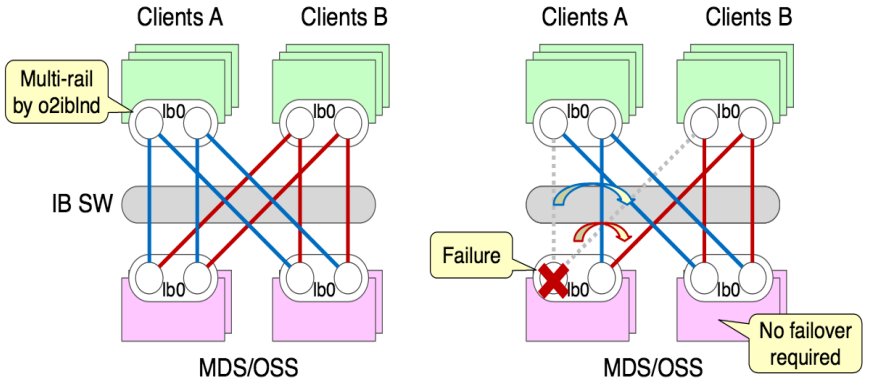
Builds on LNet Multi-Rail in Lustre 2.10 (Intel, HPE/SGI*)

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

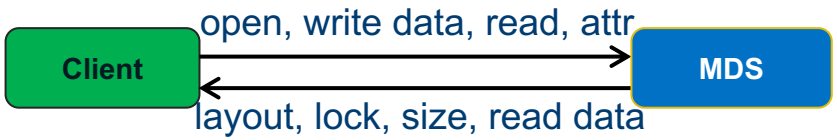
Size on MDT for small files

Integrates with PFL to simplify usage

- Start file on MDT, grow onto OSTs if larger

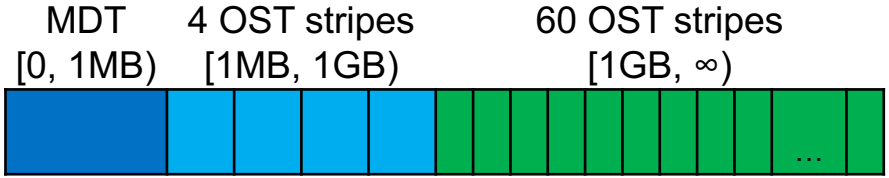
Complementary with DNE 2 striped directories

- 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)

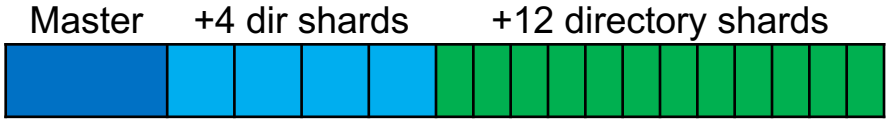
Directory migration from single to striped/sharded directories

- 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)
- 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

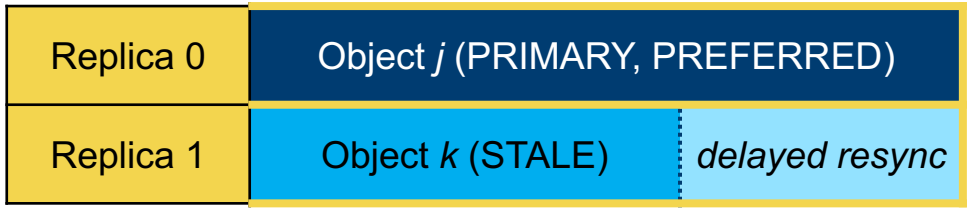
- On-disk encryption (Datto*)
- Project quota accounting (Intel)
- Declustered RAID (dRAID) (Intel)
- Metadata Allocation Class (Intel)
- Likely lots more...

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 ...



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

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)

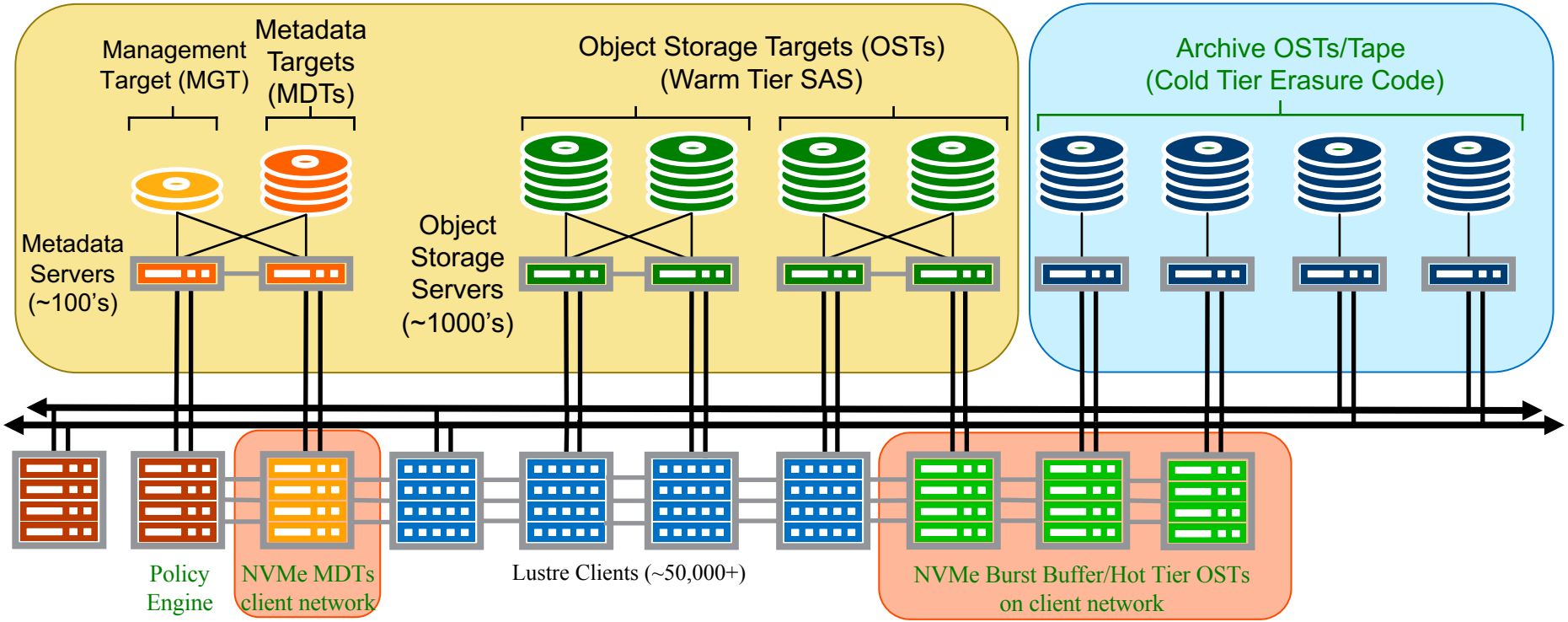
- Reduce memory usage on client and server for large systems
- Reduce network pings and recovery times
- Aggregate `statfs()` RPCs on the MDS (LU-10018)

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

Tiered Storage and File Level Redundancy

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



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

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

Piecewise compression

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

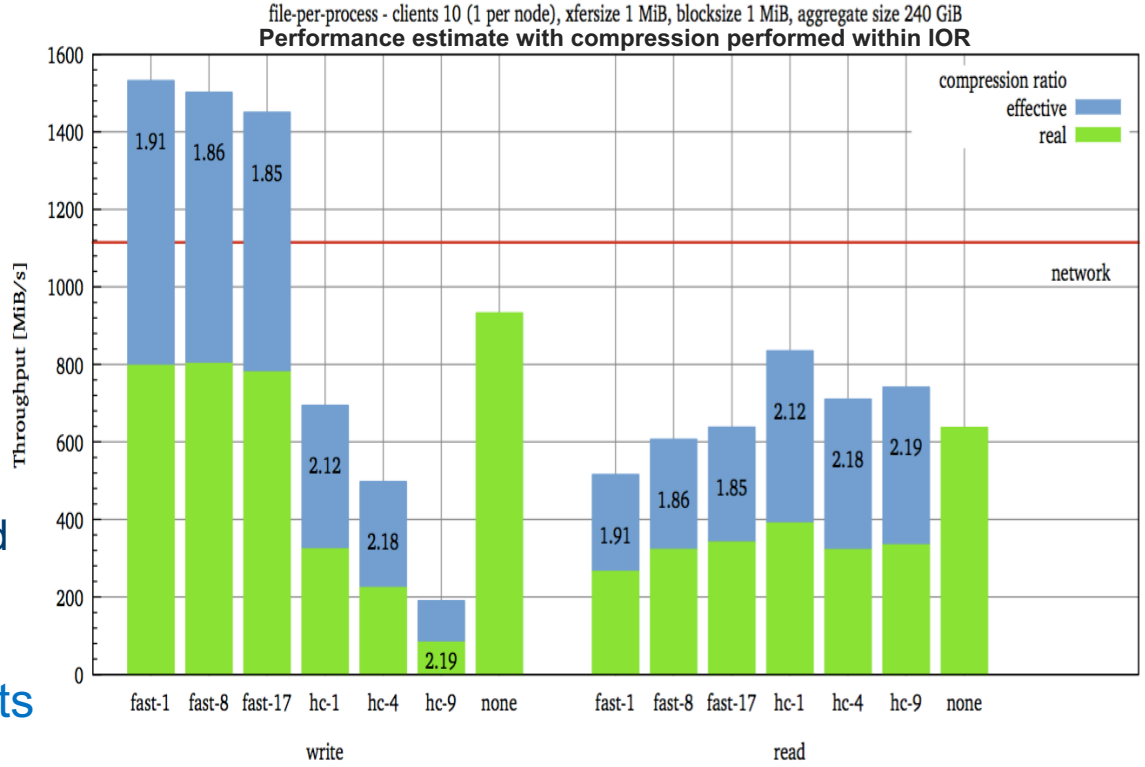
Integrated with ZFS data blocks

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

Avoid de-/re-compressing data

Good performance/space benefits

- Graph courtesy Uni Hamburg



Persistent Client-side Cache (LU-10092 DDN* 2.12)

Leverage fast client-local NVMe or NVRAM device

Mount filesystem image file directly on clients

- Used ad-hoc today - single client RW/shared RO mount
- Integrate handling into Lustre transparently

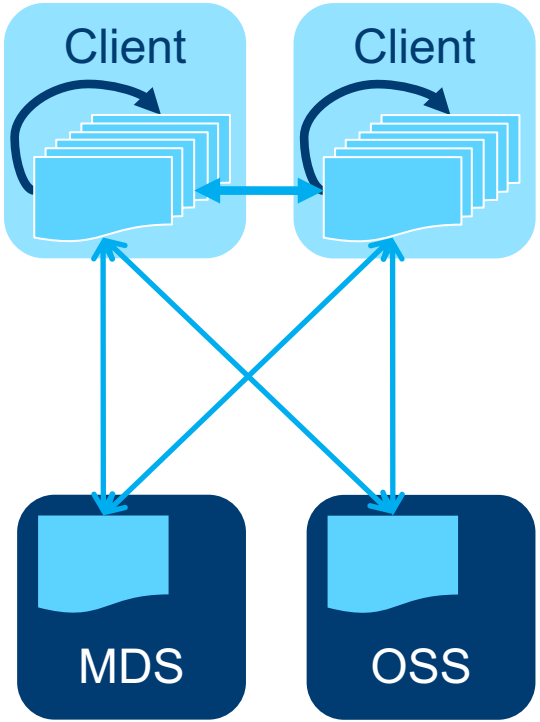
Automatic client-local mount of filesystem image file

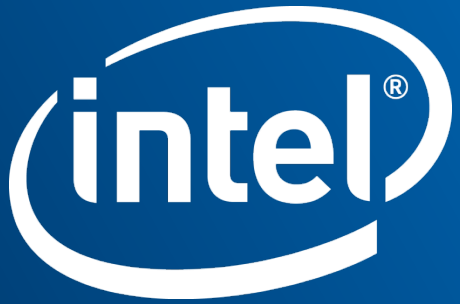
- Image is one object on OST, whole directory tree on client
- **Low overhead**, few Lustre locks, **100k+ IOPS/client**
- Access, migrate, replicate with **large reads/writes to OST**

OSS/MDS/client can export directly for shared use

- Use Data-on-MDT to re-export image to other clients
- HSM migrates *from* client, FLR to mirror to OST

Archive whole filesystem tree to tape when unused




intel®