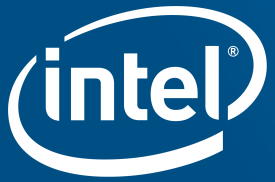


Lustre* Features In Development

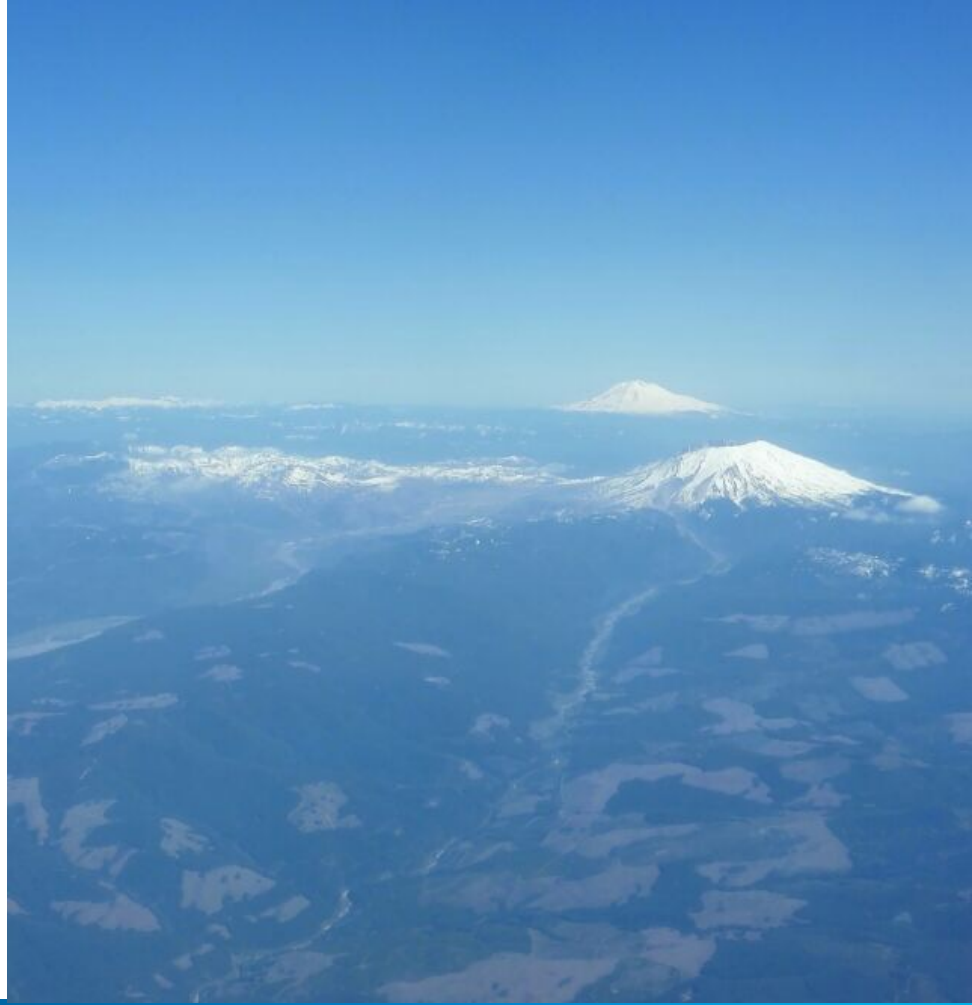
Fan Yong

High Performance Data Division, Intel

CLUG 2017 @Beijing

Outline

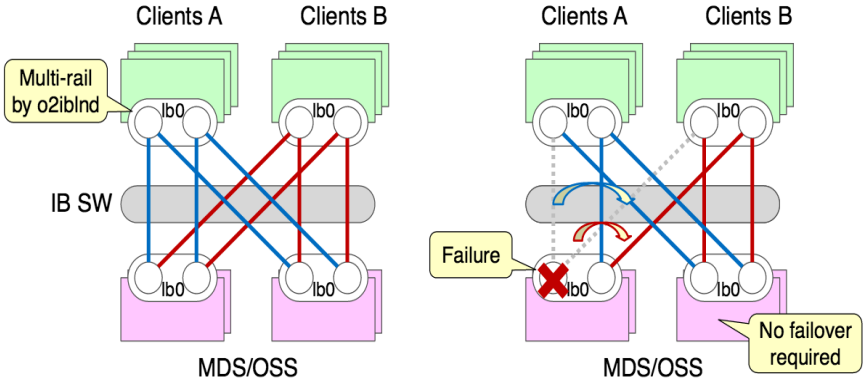
- LNet reliability
- DNE improvements
- Small file performance
- File Level Redundancy
- Miscellaneous improvements
- ZFS enhancements
- Client side data compression



LNet Network Health (LU-9120, Intel)

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

- Detect network failures
  - Local interface, remote interface, router
- Handle related faults in LNet
  - Resend across different type(s) of interface(s)
  - Mitigate & simplify upper layer error handling
- Restore connection when available
  - Avoid upper layer complex recovery



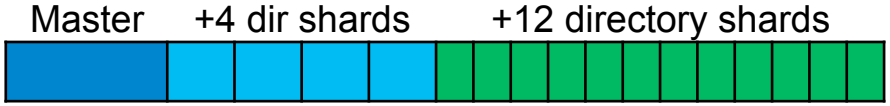
DNE Improvements (LU-4684, Intel)

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)
- Some policies for the restriping
  - Existing *dir entries* stay in master, or are migrated to shards asynchronously
  - New created entries in new shards to distribute load, or rebalance globally
- Performance scales as directory grows



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

Avoid OST overhead (data, lock RPCs)

Pre-fetch file data with metadata

Size on MDT for small files

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

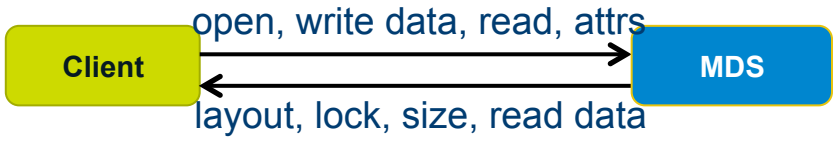
Avoid contention with streaming IO to OSTs

Integrates with PFL to simplify usage

- Start file on MDT, grow onto OSTs if larger

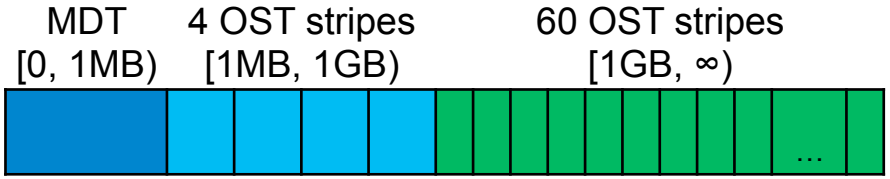
Complementary with DNE 2 striped directories

- Scale small file IOPS with multiple MDTs



Small file IO directly to MDS

Example DoM/PFL File Layout

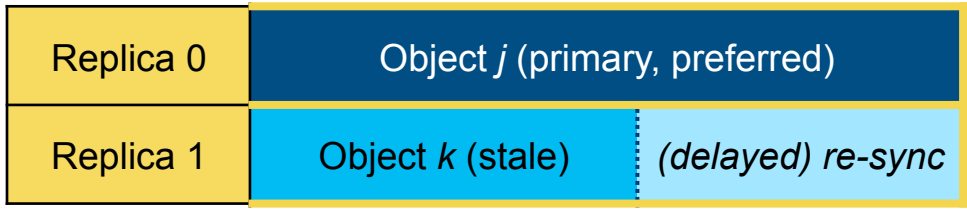


File Level Redundancy (LU-9771, Intel)

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 re-sync replica)
- Many more possibilities …



Miscellaneous Improvements

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

- Real-time priorities for TBF rules if server is overloaded (LU-9228, DDN*)
- Support UID/GID for TBF policies, compound policy specification (LU-9658)

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

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

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)

ZFS Enhancements Related to Lustre*

Lustre `osd-zfs` updates to use zfs-0.7.x

- Use ZFS native object accounting (Intel)
  - More accurate object quota
- OI Scrub for ZFS (LU-7585, Intel)
  - Server side file-level backup/restore
  - Migration from ldiskfs to ZFS
- File create performance (Intel)
  - Parallel lock/alloc
  - New APIs

ZFS backend new features

- Dynamic dnode size (0.7.x, LLNL)
  - Better xattr performance/space
- Optimized parallel dnode allocation (0.7.x, Delphix, LLNL, Intel)
- Multi-mount protection (#6279, 0.7.x, LLNL)
  - MMP, improve HA safety
- Native encryption for ZoL (#5769, 0.8.x, Datto)
- Project quota on ZFS (#6290, 0.8.x, Intel)
- Declustered RAID (dRAID) (0.8.x, Intel)

Client Side Data Compression (LU-10026, Uni Hamburg)

Piecewise compression

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

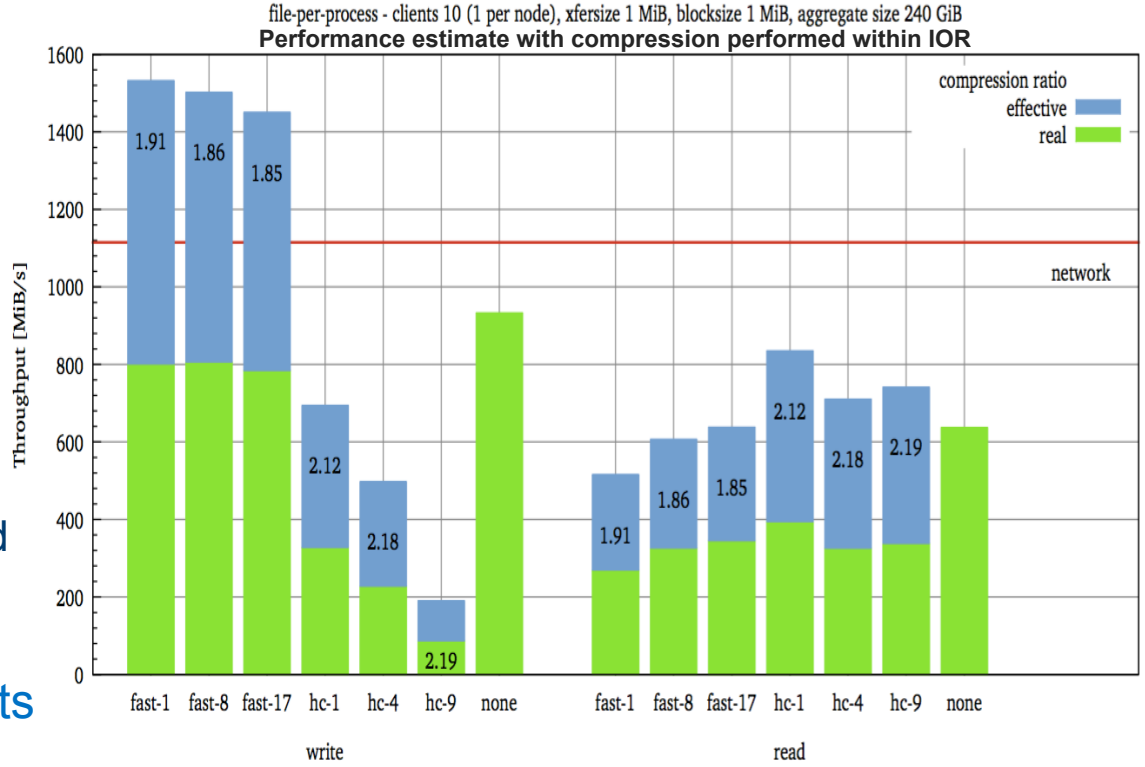
Integrated with ZFS data blocks

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

Avoid de-/re-compressing data

Good performance/space benefits

- Graph courtesy Uni Hamburg



Notices and Disclaimers

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

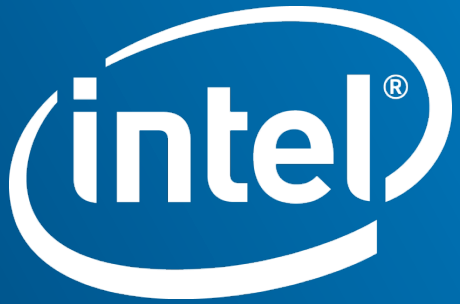
Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

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®