Lustre* 2.11 and Beyond

Andreas Dilger, Intel High Performance Data Division

Lustre User Group 2017

Statements regarding future functionality are estimates only and are subject to change without notice
* Other names and brands may be claimed as the property of others.

Performance and Feature Improvement Highlights

2.10 is nearly here and has several long-awaited features

- Multi-Rail LNet for improved performance and reliability, ***continuing in 2.11***
- Progressive File Layout (PFL) for improved performance and usability
- ZFS related improvements (snapshots, create performance, ...), ***continues in 2.11***

2.11 looking to be a very interesting release

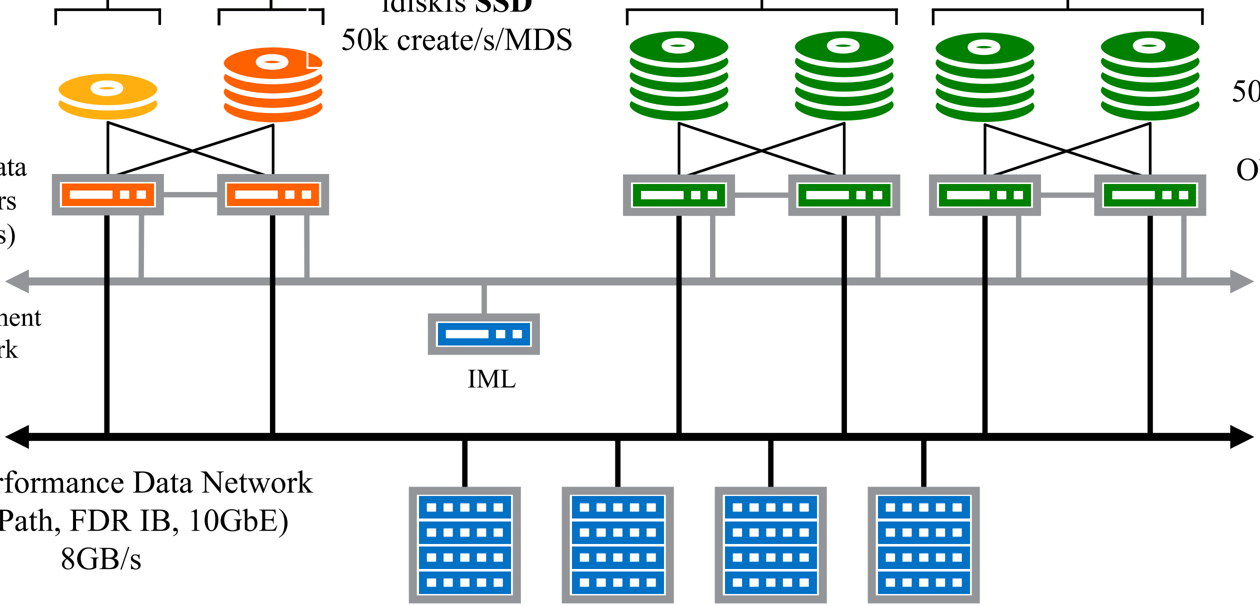
- Data-on-MDT for improved small file performance/latency
- File Level Redundancy (FLR Phase 1 delayed resync) for reliability/performance
- DNE directory restriping for ease of space balancing and DNE2 adoption

2.12/2.13 plans for 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

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Lustre Performance – *Roughly* How it Looks Today



Management Target (MGT)

Metadata Targets (MDTs)

ZFS **HDD**
10k create/s/MDS

ldiskfs **SSD**
50k create/s/MDS

Object Storage Targets (OSTs)

HDD/SSD
8GB/s/OSS

SSD OSS
50k 4-32KB IOPS

Metadata Servers (~10’s)

Object Storage Servers (~1000’s)

Management Network

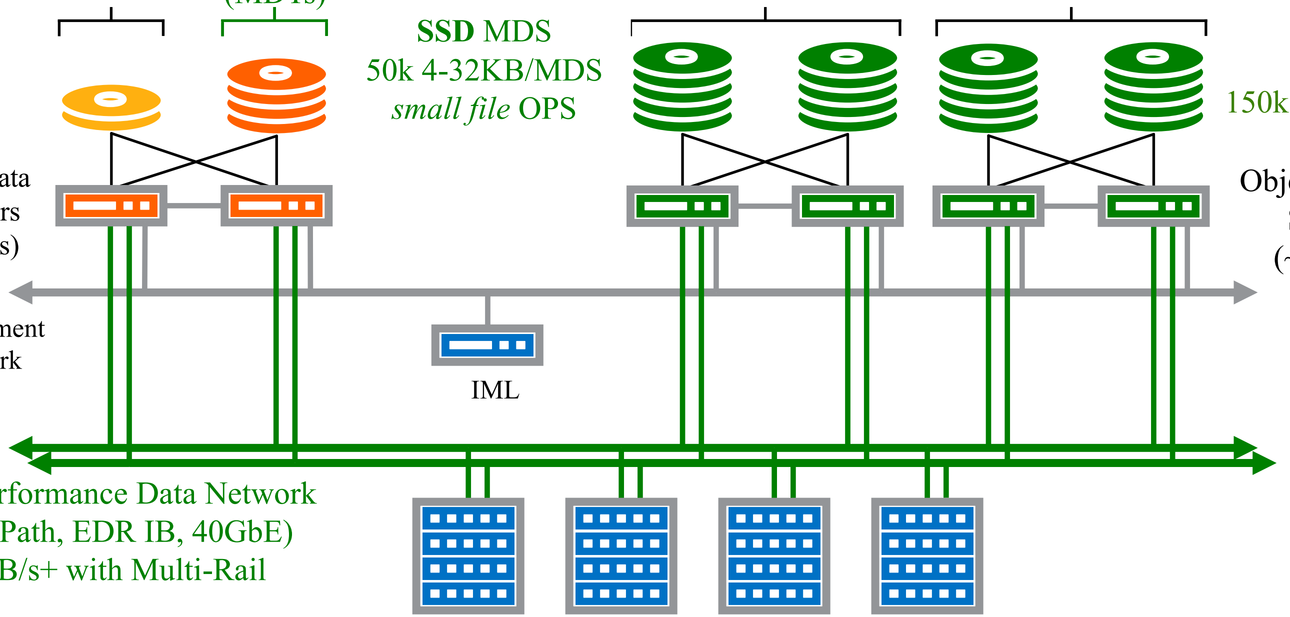
IML

High Performance Data Network
(OmniPath, FDR IB, 10GbE)
8GB/s

Lustre Clients (~50,000)

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Performance *Targets* for Lustre (2.11+)



Management Target (MGT)

Metadata Targets (MDTs)

ZFS **SSD**
100k create/s/MDS

SSD MDS
50k 4-32KB/MDS
small file OPS

Object Storage Targets (OSTs)

HDD/SSD
10-12GB/s/OSS

SSD OSS
150k 4-32KB IOPS

Metadata Servers (~10's)

Object Storage Servers (~1000's)

Management Network

IML

High Performance Data Network
(OmniPath, EDR IB, 40GbE)
20GB/s+ with Multi-Rail

Lustre Clients (~50,000)

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

ZFS Enhancements affecting Lustre (2.11)

Lustre `osd-zfs` improvements to use ZFS more efficiently

- Improved file create performance (Intel)
- Update to ZFS 0.7.0 final release, if not in 2.10

Changes to core ZFS code (0.7.0+)

- QAT hardware-assisted checksums and compression (Intel)
- Improved kernel memory allocation for IO buffers (ABD) (others, Intel)
- Improved fault handling and notification (FMA/ZED/degraded mode) (Intel)
- Multi-mount protection (MMP) for improved HA safety (LLNL)
- Project quota accounting for ZFS (Intel)

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

New Features for ZFS (Intel, ANL ZFS 0.8+)

Declustered parity RAID (dRAID) & distributed hot spaces

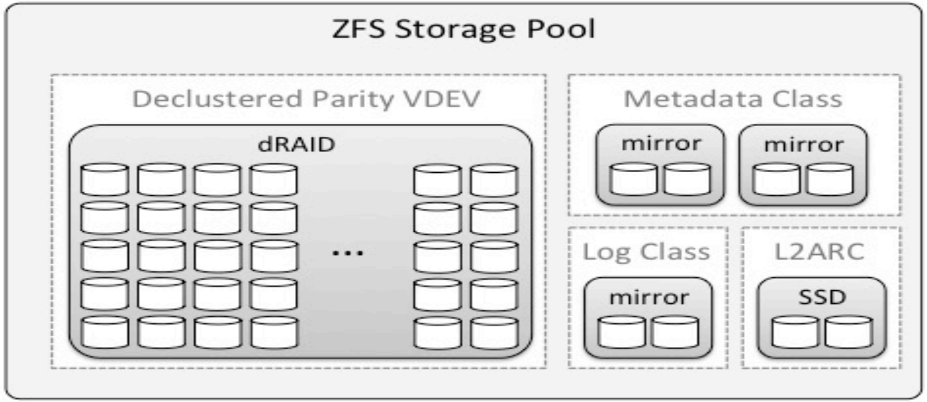
- Resilver across all drives in zpool, potentially improve performance by `O(num_vdevs)`
- Use bandwidth/IOPS of "hot spare" drives in normal operation rather than leave them idle
- *or* Trade off extra capacity of large drives for shorter risk window when drive has failed

https://github.com/zfsonlinux/zfs/wiki/dRAID-HOWTO

Metadata allocation class

- Isolate small and large block sizes/lifetimes
- Store all metadata on dedicated SSD/NVRAM
- *or* Store all metadata in separate Metaslabs

https://github.com/zfsonlinux/zfs/issues/3779



Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

LNet Dynamic Discovery, Network Health (2.11/2.12)

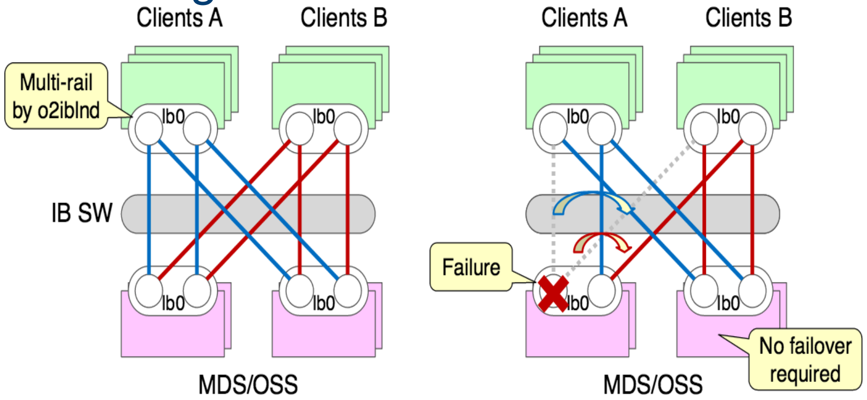
Builds on LNet Multi-Rail in Lustre 2.10 (Intel, SGI/HPE*)

LNet Dynamic Discovery (LU-9480)

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

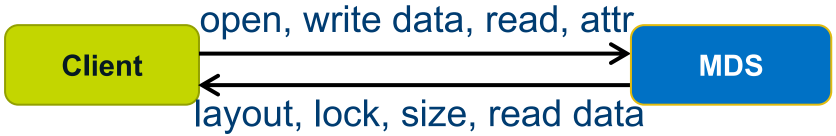
LNet Network Health (LU-9120)

- Detect network interface, router faults
- Automatically handle fault conditions
- Restore connection when available



Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Data-on-MDT Small File Performance (Intel 2.11)


Client

open, write data, read, attr

MDS

layout, lock, size, read data

Small file IO directly to MDS

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

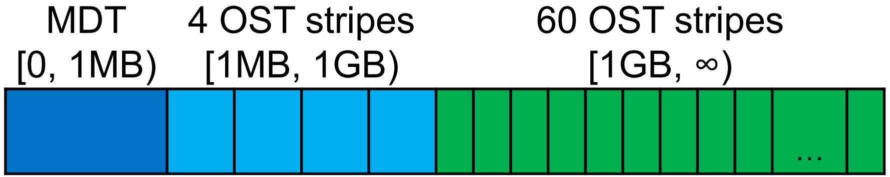
Integrates with PFL to simplify usage

- Start file on MDT, grow onto OSTs if larger

Complementary with DNE 2 striped directories

- Scale small file IOPS with multiple MDTs

Example DoM/PFL File Layout


MDT [0, 1MB)

4 OST stripes [1MB, 1GB)

60 OST stripes [1GB, ∞)

…

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

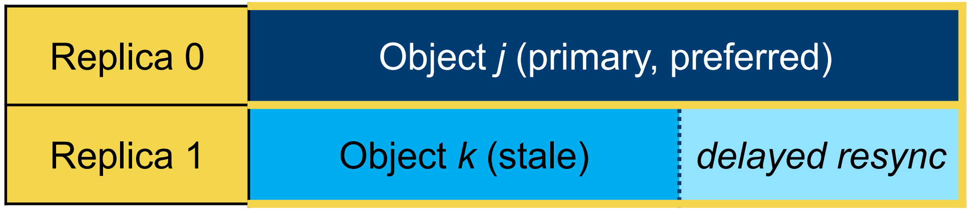
Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

File Level Redundancy (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 ...



Replica 0 | Object *j* (primary, preferred)

Replica 1 | Object *k* (stale) | *delayed resync*

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

FLR Phased Implementation Approach

Can implement Phases 2/3/4 in any order

Phase 0: Composite Layouts from PFL project (Intel, ORNL 2.10)

- Plus OST pool inheritance, Project/Pool Quotas

Phase 1: Delayed read-only mirroring – depends on Phase 0 (Intel 2.11)

- Mirror and migrate data after file is written using external resync tool

Phase 2: Immediate write mirroring – depends on Phase 1 (Intel 2.12)

- Clients write to multiple mirrors directly

Phase 3: Integration with policy engine/copytool – depends on Phase 1

- Automated migration between tiers using policy/space, parallel scan/resync

Phase 4: Erasure coding for striped files – depends on Phase 1 (2.13)

- Avoid 2x or 3x overhead of mirroring large files

Statements regarding future functionality are estimates only and are subject to change without notice.

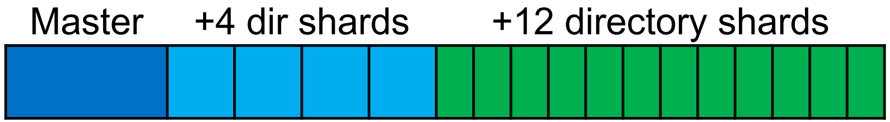
Ongoing DNE Improvements (Intel 2.11/2.12)

Directory migration from single to striped/sharded directories (LU-4684)

- 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 of common case
- Add extra shards when master directory grows large enough (e.g. 64k entries)
- Existing *entries* stay in master, or are migrated to shards asynchronously?
- New entries+inodes created in new shards/MDTs
- Performance scales as directory grows



MDT Pools for space/class management?

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Miscellaneous Improvements (2.11)

Client Lock Ahead (Cray*)

- Client (MPI-IO) to request read/write DLM locks before IO to avoid latency/contention

Network Request Scheduler – Token Bucket Filter (NRS-TBF) (DDN*, Uni Mainz)

- Improve flexibility of specifying TBF rules on the MDS and OSS; global scheduling (2.12?)

Kernel tracepoints for logging/debugging/performance analysis (ORNL)

Small file write investigation and optimizations (Intel)

- Reduce client (and RPC/server?) overhead for small (~8KB) writes

Ldiskfs scalability improvements (Seagate*)

- Support for > 256TB OSTs, > 10M directory entries; more than 4B files per FS?

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Advanced Lustre* Research
Intel Parallel Computing Centers

Uni Hamburg + German Client Research Centre (DKRZ)

- Adaptive optimized ZFS data compression
- Client-side data compression integrated with ZFS

GSI Helmholtz Centre for Heavy Ion Research

- TSM* HSM copytool for Lustre

University of California Santa Cruz

- Automated client-side load balancing

Johannes Gutenberg University Mainz

- Global adaptive Network Request Scheduler
- Allocate bandwidth to jobs at startup for Quality of Service

Lawrence Berkeley National Laboratory

- Spark and Hadoop on Lustre

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Lustre Client-side Data Compression (2.12?)

IPCC project: Uni Hamburg

Client-side data compression

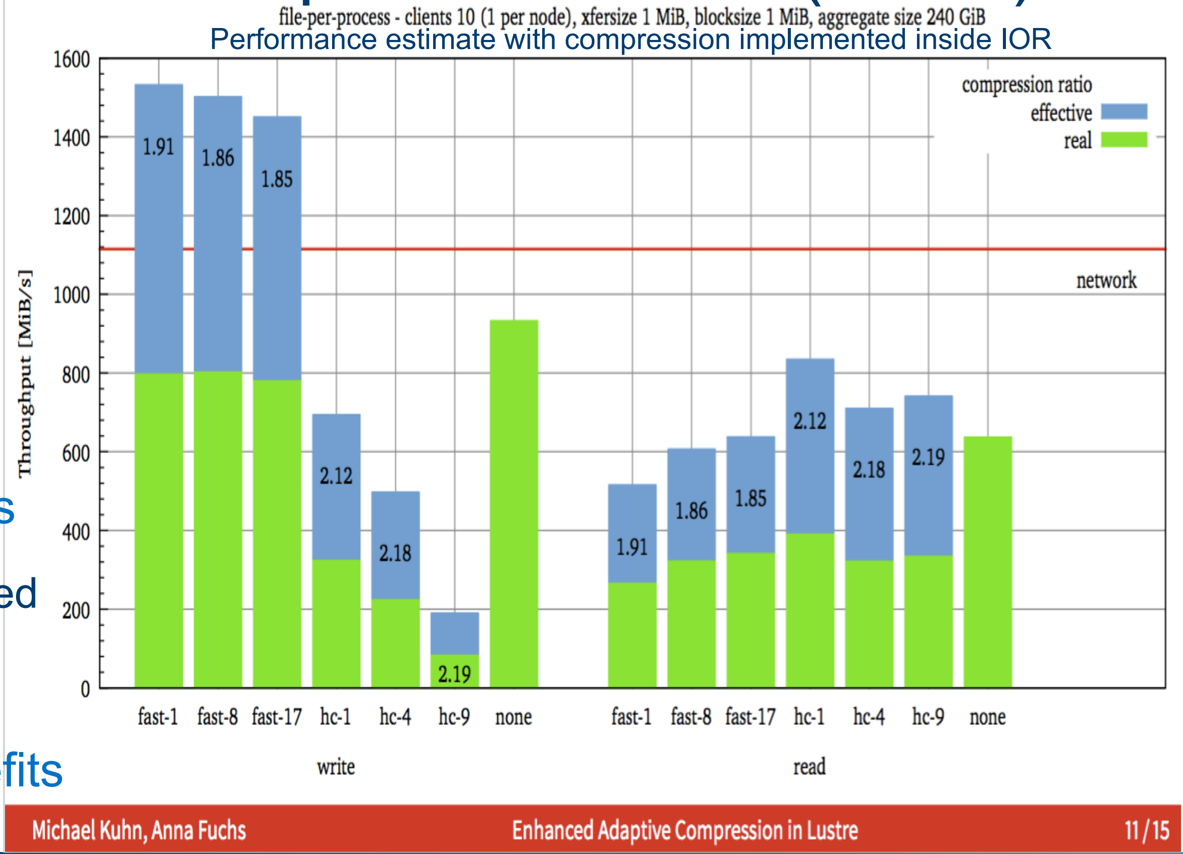
- Compressed in 32KB chunks
- Allows sub-block read/write

Integrated with ZFS data blocks

- Code/disk format changes needed

Avoid de-/re-compressing data

Good performance/space benefits



file-per-process - clients 10 (1 per node), xfersize 1 MiB, blocksize 1 MiB, aggregate size 240 GiB

Performance estimate with compression implemented inside IOR

Michael Kuhn, Anna Fuchs   Enhanced Adaptive Compression in Lustre   11/15

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Lustre Client-side Compression + Hardware Assist

Intel Quick Assist Technology (QAT)

- PCI card/chipset accelerator

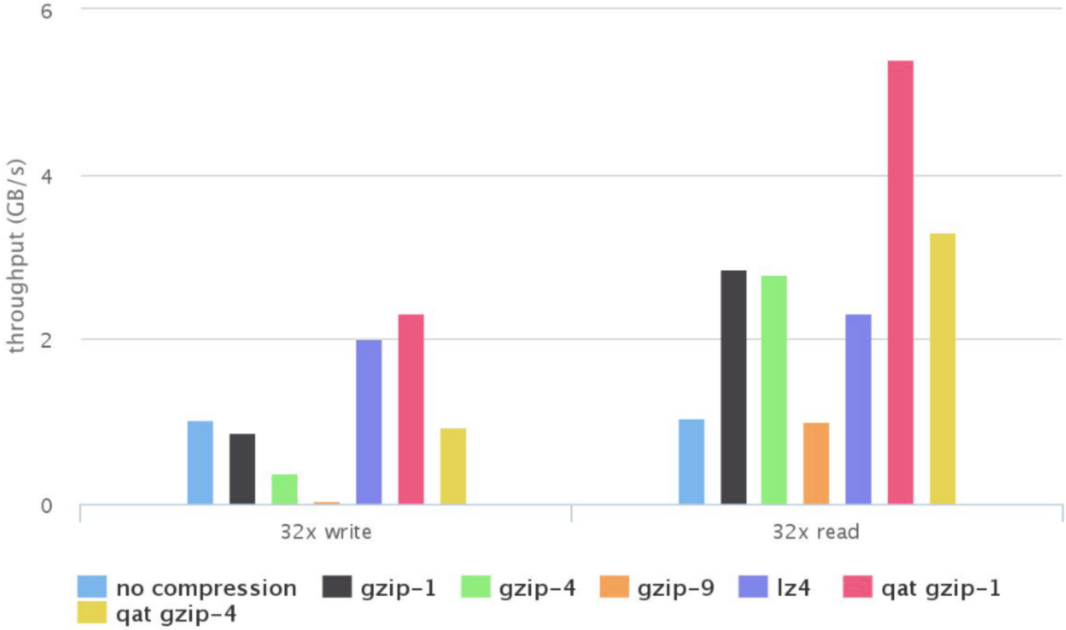
Compression *improves* performance

QAT integrated with ZFS 0.7.0

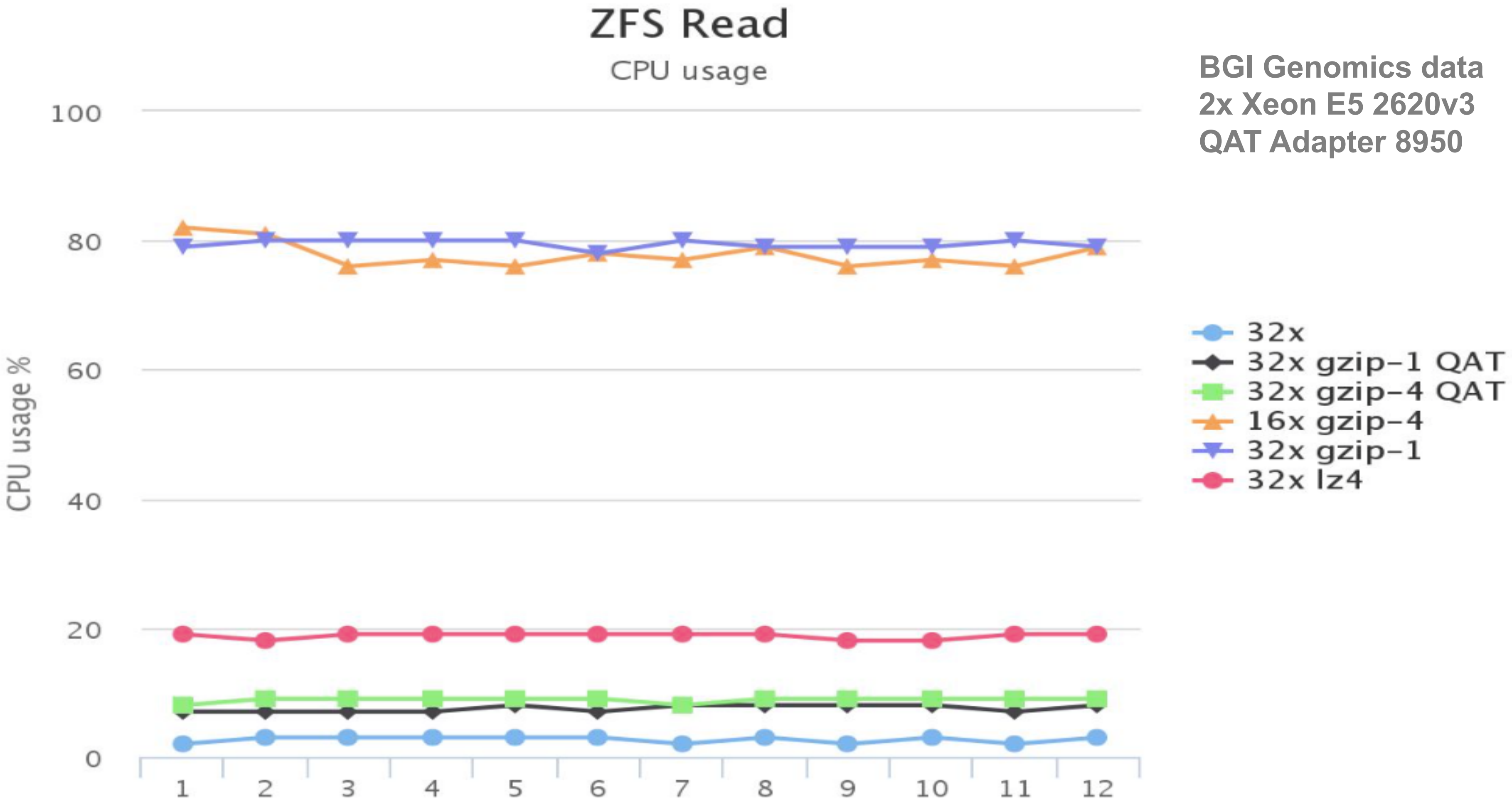
Benchmark shows ZFS local perf

- Data from Beijing Genomics Institute
- 2 Intel® Xeon E5 2620v3
- QAT Adapter 8950

ZFS throughput (BGI genomics data)



Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.



BGI Genomics data
2x Xeon E5 2620v3
QAT Adapter 8950

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Development Roadmap Continues - 2.13 and Beyond

Tiered storage with Composite Layouts and File Level Redundancy

- Policy engine to manage migration over tiers, rebuild replicas, ChangeLogs
  - Policies for pathname, user, extension, age, OST pool, mirror copies, ... ?
- Multiple policy and scanning engines are being presented this week
- Lustre optimizations such as fast inode scanning could be implemented
  - Internal *OST maps* on MDT to allow fast rebuild of failed OST without scanning?
- This is largely a userspace integration task, with some hooks into Lustre

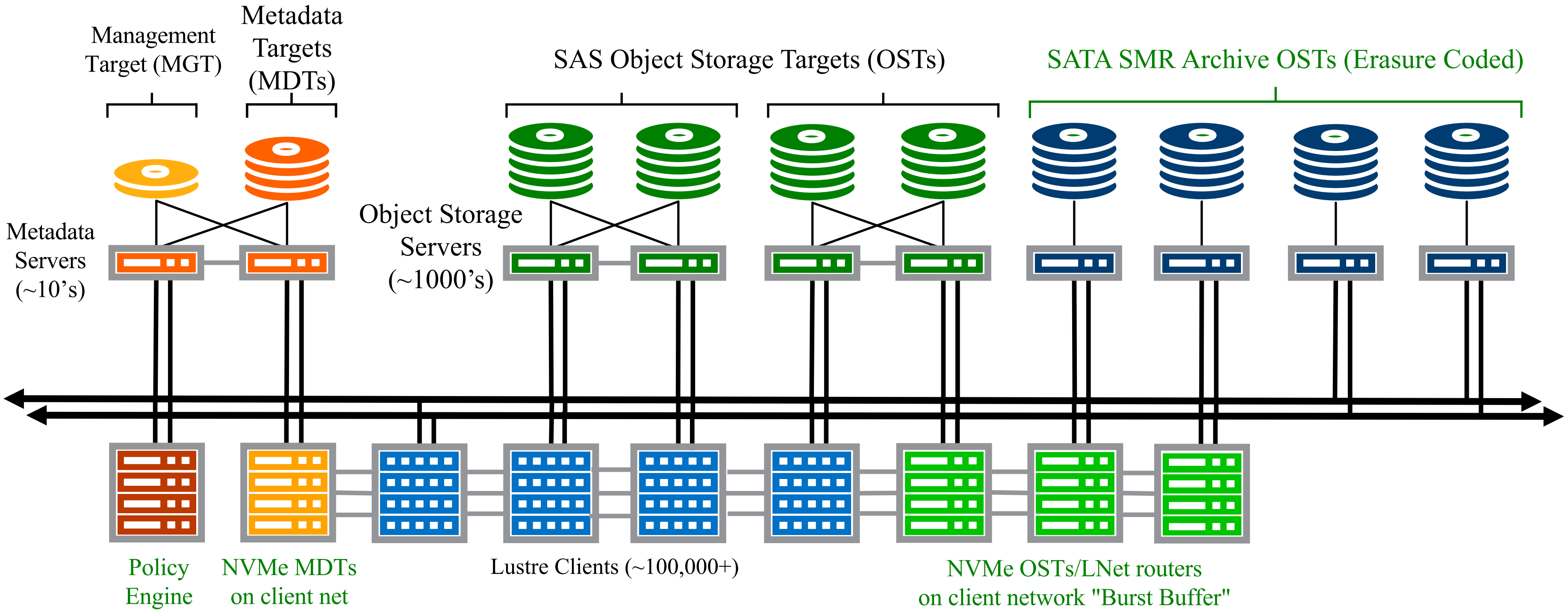
Metadata Redundancy via DNE2 distributed transactions

- New striped directory layout hash for mirrored directories, mirrored MDT inodes
- Use same mechanism as DNE2 for consistency and recovery
- Early design work started, discussions ongoing

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

Multi-Tiered Storage and File Level Redundancy

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


Management Target (MGT)
Metadata Targets (MDTs)
SAS Object Storage Targets (OSTs)
SATA SMR Archive OSTs (Erasure Coded)
Metadata Servers (~10's)
Object Storage Servers (~1000's)
Policy Engine
NVMe MDTs on client net
Lustre Clients (~100,000+)
NVMe OSTs/LNet routers on client network "Burst Buffer"

Statements regarding future functionality are estimates only and are subject to change without notice.
* Other names and brands may be claimed as the property of others.

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.

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®