Whamcloud

Lustre status and path forward

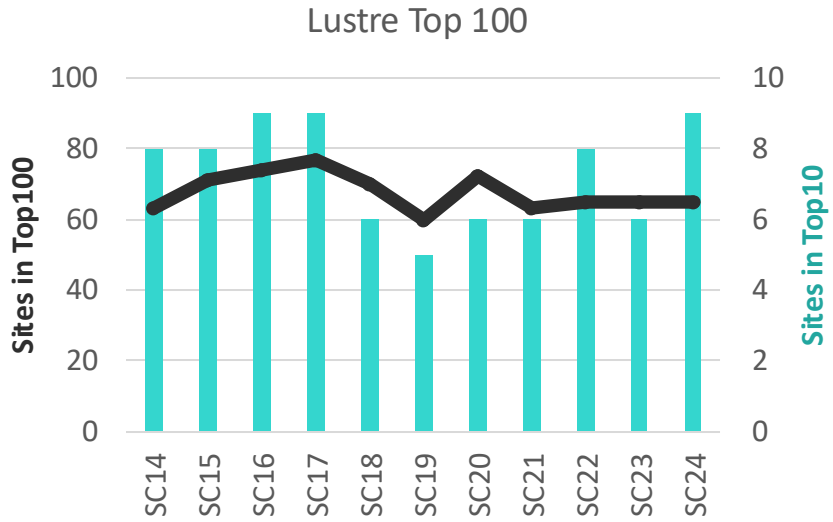
Sébastien Buisson

Senior Lustre Engineer

Big is Beautiful

Whamcloud

- Lustre is the preferred choice for Exascale systems
  - World's largest AI/ML systems (Eos, Selene, X-AI, Scaleway)…
  - …also 2RU servers available with 60W/80R GB/s and 3M read IOPS



- Proven scalability of servers/clients to meet system requirements
  - 100M+ IOPS, 10 M+ metadata op/sec, 100B+ files today, ongoing improvements
  - Capacity can grow almost without limits - 100's PB today, 1 EB+ in the near future
  - Bandwidth scales almost linearly, aggregate 10's of TB/s today
  - Fully support client capabilities - 100's cores, TB's RAM, multi-100Gbps NICs, GPU RDMA

- Continued improvements for large system deployments
  - Steady feature development to meet evolving system and application needs
  - Virtualization of filesystem for multi-tenancy and data privacy

… but Size is not Everything

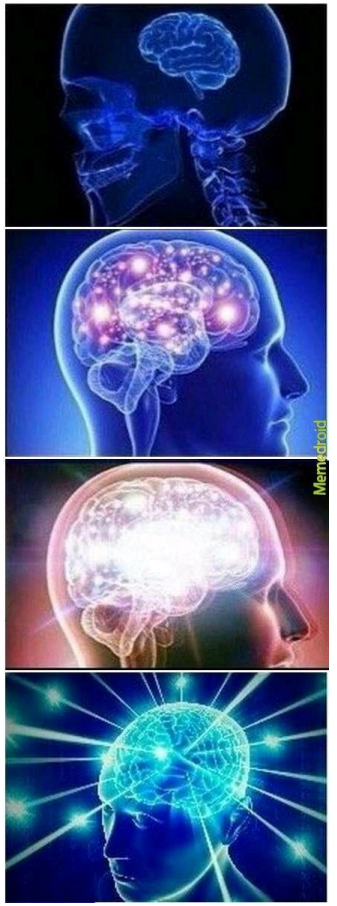
Whamcloud

- Demand for fast storage is everywhere
- Ongoing improvements for smaller systems too
  - Ongoing improvements in ease of use, flexibility, monitoring, reliability
  - Configurable security, encryption, multi-tenant isolation, quotas, etc.

Evolution of IO Interfaces

Whamcloud

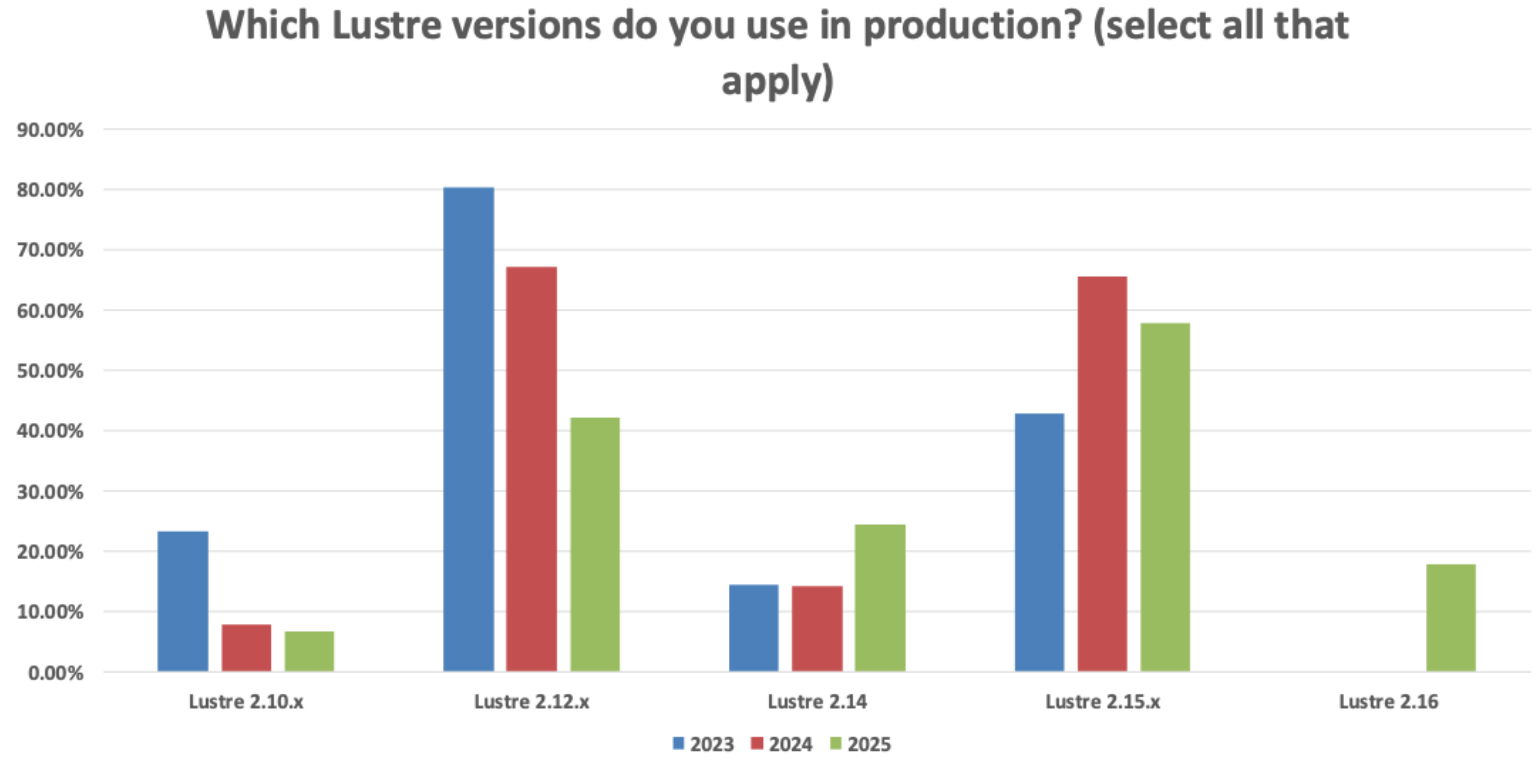
- POSIX has been the standard IO interface for decades
  - Protects substantial investment in developed applications and tools
  - Consistent behavior avoids chasing interface-of-the-month
  - Data portability via protocol export (Lustre, NFS, SMB, S3, …)
- Keeping up with hardware speedups demands continual optimization
  - Unaligned IO, cross-dir/file prefetch, WBC for creates
  - Asynchronous meta/data ops via Linux io_uring, batched file create
- Opt-in API *extensions* for apps with special performance needs
  - Relaxed semantics/interfaces when/where applications need/understand it
  - Data stored and continues to be accessible via standard APIs afterward
- POSIX will continue to be the common interface going forward



Lustre Community Release News

Lustre LTS Releases

- Lustre 2.15.0 GA June 2022
  - Replaced 2.12.x as LTS branch
- Lustre 2.15.7 coming soon
  - RHEL 9.6 clients
  - https://wiki.lustre.org/Lustre_2.15.7_Changelog



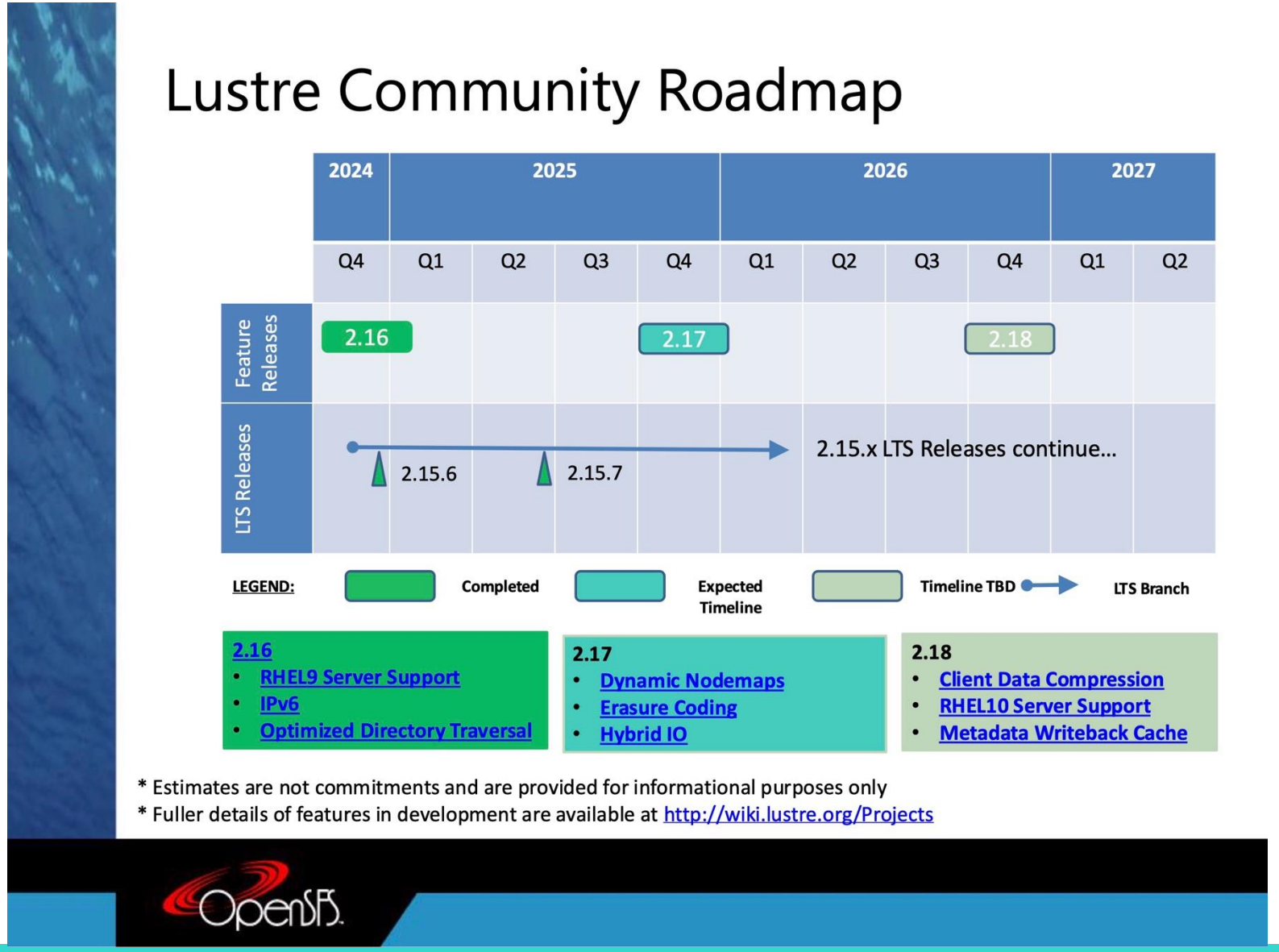
https://wiki.opensfs.org/Lustre_Community_Survey

Lustre 2.16

Whamcloud

- 2.16.0 GA Nov 8th http://wiki.lustre.org/Release_2.16.0
  - 2.16.1 fixes important bug (LU-18435) but otherwise identical
  - 2.16.x is NOT an LTS branch 😀
- RHEL 9.4 servers/clients
- SLES15 SP6/Ubuntu 24.04 clients
- Interop/upgrades from latest Lustre 2.15.x
- Features in the release
  - RHEL 9.x server support
  - IPv6 Support (LU-10391)
  - Optimized Directory Traversal (LU-15975)
    - Aka “Improved Statahead”

Lustre Community Roadmap



Planned Feature Release Highlights

Whamcloud

- **2.17** now open for major feature landings
  - **Hybrid IO Optimizations** – Hybrid BIO/DIO and server writeback cache (WC, Oracle)
  - **Dynamic Nodemaps** – ephemeral/hierarchical configuration for subdirectory trees (WC)
  - **File Level Redundancy - Erasure Coding** (FLR-EC) – M:N data redundancy (ORNL)
- **2.18** already has some features under development and detailed design
  - **Client-side Data Compression** – reduce network and storage usage/cost (WC, UHamburg)
  - **Fault Tolerant MGS** (LMR-FTM) – mirror MGS service and logs across MDS nodes
  - **Trash Can/Undelete** (TCU) – allow file recovery after accidental/malicious deletion
- **2.19** features under discussion for development
  - **Metadata Writeback Cache** (WBC2) – single-client metadata speedup (WC)
  - **Lustre Metadata Redundancy** (LMR1b) – MDT0000 services can run on other MDTs
  - **Lustre Metadata Redundancy** (LMR2a) – ROOT directory mirroring to other MDTs

LNet Improvements

Whamcloud

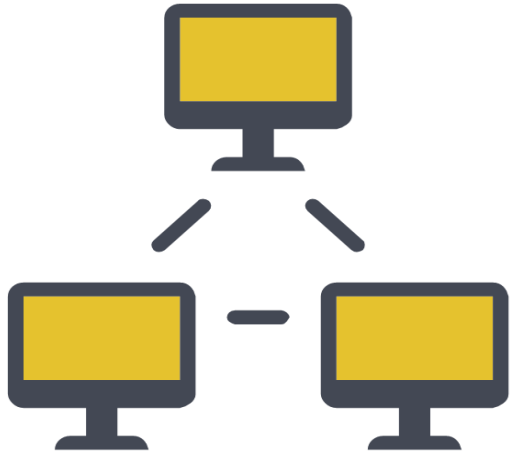
- IPv6 large NID support (LU-10391 SuSE, ORNL, HPE)
  - Demand for IPv6 in cloud deployments as IPv4 addresses are exhausted.

2.16

  - Also enables other large addresses (e.g. direct IB GUID instead of IPoIB)

2.17

- Mount without server NIDs in configuration logs (LU-10360, WC)
  - Servers register with MGS at mount, MGS IR Table sends server NIDs to clients
  - Clients can now (optionally) get server NIDs only from MGS IR Table, not config logs
  - Simplifies network setup, server migration/failover config, etc.
- Improve handling of MGS with many NIDs (LU-16738, WC)
  - Round-robin DNS records for multiple MGS NIDs at mount command-line
  - Display MGS hostname instead of NIDs for "`mount`" and "`df`" output
- Improve network transfer for sparse reads (LU-16897, WC)
  - Don't send holes/zero pages over network when no blocks allocated
- Remote LND configuration discovery for EFA LND (LU-18808, AWS)



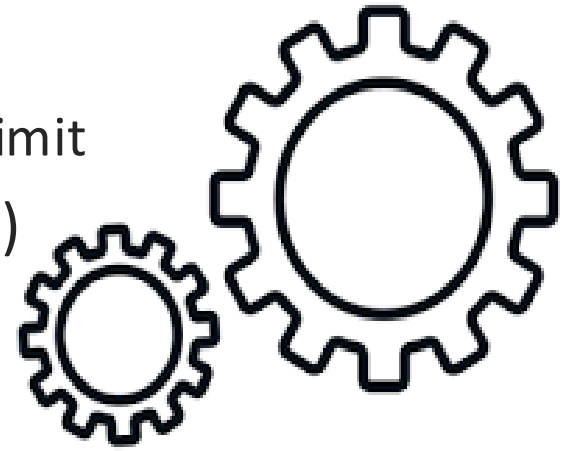
Client-Side Functionality Improvements

Whamcloud

Ongoing ease-of-use and performance improvements for users and admins

- Ongoing code updates/cleanup for upstream 6.x kernels (ORNL, HPE, WC, SuSE)
- Allow specifying CPU cores to exclude from CPT list (LU-17501 WC)
  - `options libcfs cpu_pattern="C[0,1]"` skips cores on each NUMA node
- Obfuscate file/dir names in debug logs to limit PII release (LU-18810 WC)
- 2.17 Reduce RPC latency for client IO via CPU power states (LU-18446 NVIDIA)

- 2.18 Project quota aggregation/nesting (LU-18222 WC)
  - Like OST Pool Quotas, allows PROJIDs to be “nested” into larger PROJID limit
- Client-side performance stats via statfs for each target (LU-7880 WC)
- FLR Erasure Coded files with delayed resync (LU-10911 ORNL)
- Discussed upstreaming client into Linux kernel (ORNL, AWS)



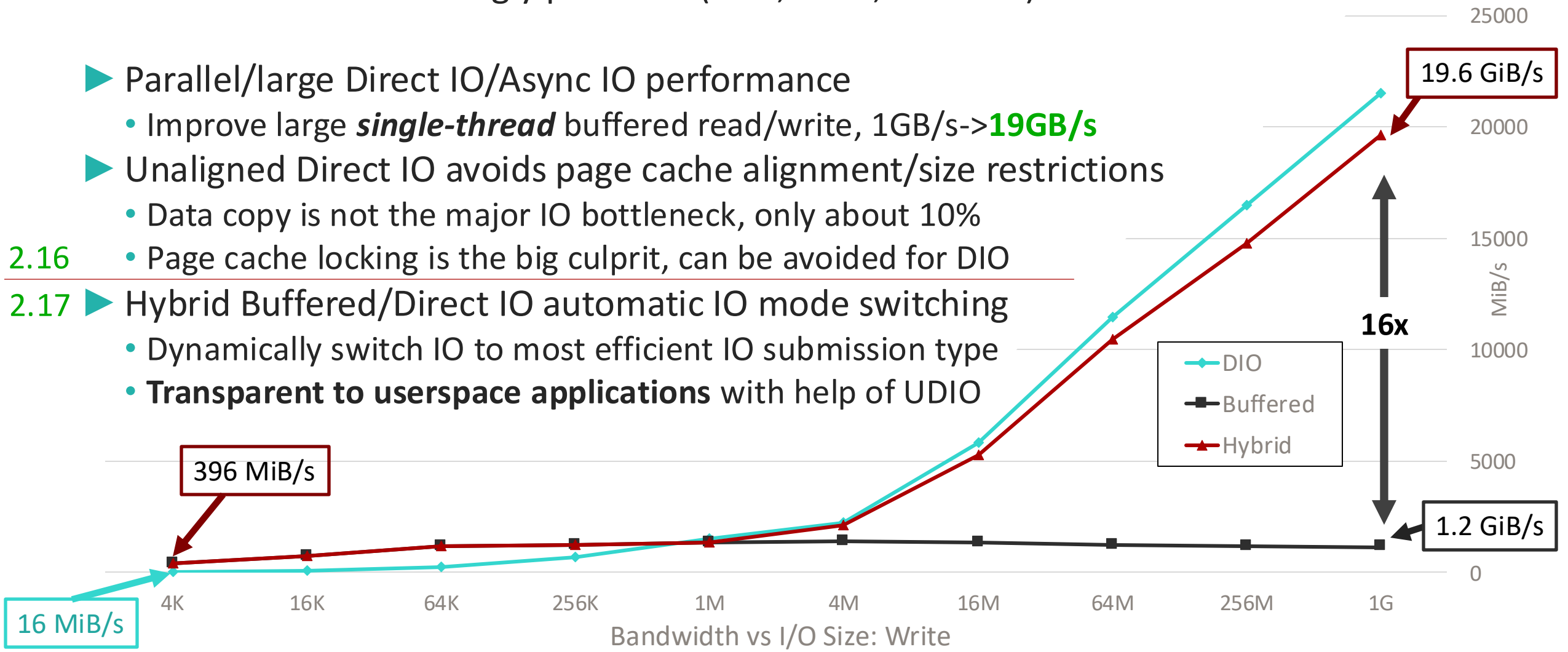
Hybrid IO: Improved Application IO Performance (2.17)

Client nodes are increasingly powerful (CPU, RAM, network) and data intensive

- Parallel/large Direct IO/Async IO performance
  - Improve large ***single-thread*** buffered read/write, 1GB/s->**19GB/s**
- Unaligned Direct IO avoids page cache alignment/size restrictions
  - Data copy is not the major IO bottleneck, only about 10%
  - Page cache locking is the big culprit, can be avoided for DIO

2.16

2.17
- Hybrid Buffered/Direct IO automatic IO mode switching
  - Dynamically switch IO to most efficient IO submission type
  - **Transparent to userspace applications** with help of UDIO



Improved Data Security and Multi-tenancy

Whamcloud

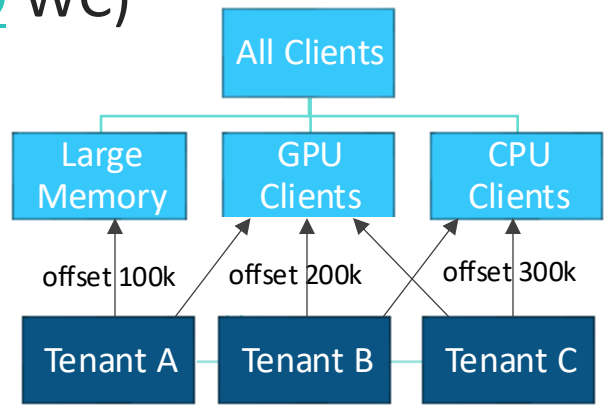
Increasing demand to isolate users and their data for legal/operational reasons

- Nodemap ID offset range for UID/GID/PROJID mapping (LU-18109 WC)
- Kerberos fixes and improvements (multiple NVIDIA, WC)
- Client-local `root` capability in nodemap (LU-18694 WC)
  - Allow root operations (chown, quota) within ID offset range
- Dynamic/hierarchical nodemap configuration (LU-17431 WC)
  - In-memory nodemap configuration for short-lived group (batch job)
- Multiple and read-only filesets per nodemap (LU-18357 WC)
- Configurable capabilities mask per nodemap (LU-17410 WC)
  - Defaults to all client capabilities disabled for security

2.17

2.18

- Encrypted fscrypt backup/restore without key (LU-16374 WC)



Client FLR Erasure Coded Files (2.17+, ORNL)

- Erasure coding adds data redundancy without 2-3x mirror overhead
  - Improve data availability above hardware and network reliability
  - Initial target for large read-mostly files, adds redundancy/availability with low cost
- Add erasure coding to new/old striped files ***after*** write completed
  - Delayed redundancy avoids overhead during initial application write
- For large striped files - add N parity per M data *stripes* (e.g. 8d+2p or 16d+3p)
  - Fixed **RAID-4** parity layout *per file*, declustered by file, CPU-optimized EC code (Intel ISA-L)
  - EC RAID pattern independent of number of stripes in file, works with PFL layouts
  - Can survive full OST loss with minimal recovery delay, transparent to applications

dat0	dat1	...	dat15	par0	par1	par2	dat16	dat17	...	dat31	par3	par4	par5	...
0MB	1MB	...	15M	p0.0	q0.0	r0.0	16M	17M	...	31M	p1.0	q1.0	r1.0	...
128	129	...	143	p0.1	q0.1	r0.1	144	145	...	159	p1.1	q1.1	r1.1	...
256	257	...	271	p0.2	q0.2	r0.2	272	273	...	287	p1.2	q1.2	r1.2	...

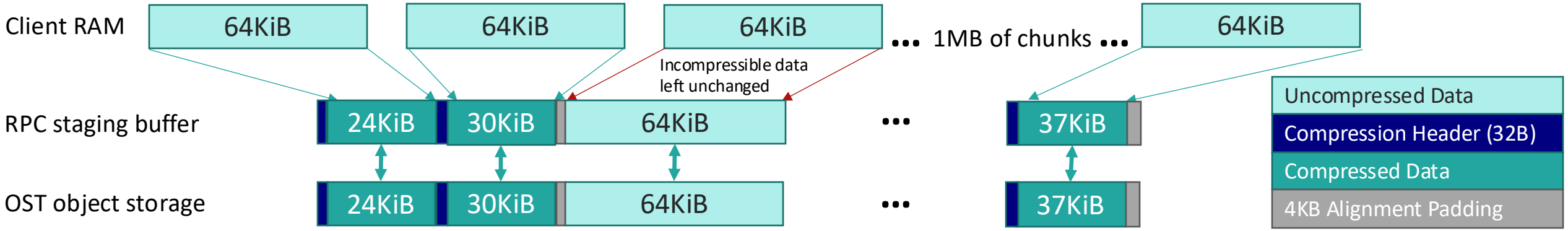
Existing stripes… Parity stripes added… Existing stripes… Parity stripes added…

Client-Side Data Compression (2.17+ WC)

Whamcloud

Increased capacity and lower cost per GB for all-flash OSTs

- Parallel (de-)compression of RPCs on client cores for GB/s speeds, **reduces server CPU load**
- (De-)Compress (`lzo`, `lz4`, `zstd`,…) RPC on client in chunks (64KiB-4MiB+)
  - **Per directory or file component** selection of algorithm, level, chunk size (PFL, FLR)
  - Keep "uncompressed" chunks as-is for incompressible data/file (`.gz`, `.jpg`, `.mpg`, …)

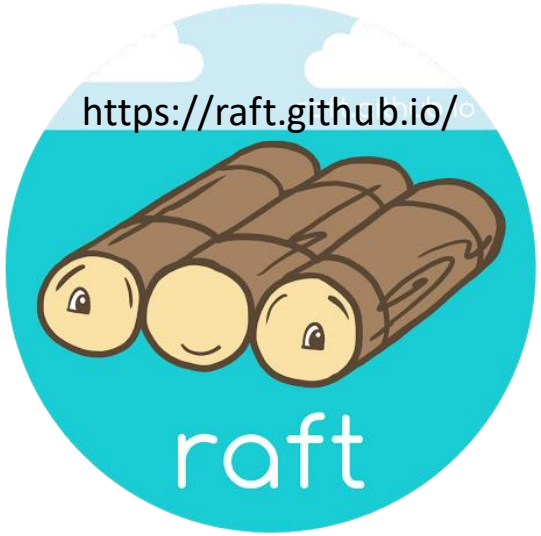


- Client writes/reads whole chunk(s), (de-)compresses to/from RPC staging buffer
  - Larger chunks improve compression, but higher decompress/read-modify-write overhead
- Could write uncompressed to one FLR mirror for random IO pattern
  - Data (re-)compression during mirror/migrate to second mirror on slow tier (via data mover)

Fault Tolerant MGS (LMR-FTM) (2.18 WC)

Whamcloud

- Run MGS service on multiple MDS nodes for availability (LU-17819)
  - Allow clients to get config llogs from **any MGS node**
  - Reduces mount time/timeouts, distributes load in large clusters
  - MGS Imperative Recovery even if “primary” MGS node restarts
- Mirror MGS config logs to remote MDTs for redundancy
  - Use RAFT Consensus algorithm to coordinate MGS cluster
  - MGS Leader election, heartbeat, consistent log updates
  - Append-only logs, matches existing MGS config llog format



Trash Can/Undelete for Files and Directories (2.18 WC)

Whamcloud

- Allow files/directories to be undeleted after `rm`/`rmdir`
  - Rescue users from fat-finger mistakes or malicious scripts
  - Handle "`rm -r`" properly to allow whole-tree recovery
- Deleted files in trash are flagged and treated specially
  - Removed from user/group/project quota and `df` usage
  - Files cannot be read to avoid abuse, and apps know files are deleted
- Virtual `.Trash` directory visible in every directory
  - Can use normal tools to list and recover files or directories
  - `.Trash` is hidden from normal directory listing
- Users can view and recover their own files
  - Configurable expiry time before cleanup (e.g. max age = 7d)
  - Configurable filesystem fullness threshold (e.g. 80% full)
  - More sophisticated cleanup policy in userspace (e.g. by user, project, nodemap)

Whamcloud

Thank You!
Questions?

Client-Side User Tools Improvements

Whamcloud

Sometimes it's the small things that make a big difference

- **`lfs df --mdt/--ost`** shows only MDT or OST devices (LU-17516 WC)
- **`lfs find -xattr/attr`** finds files with specific attributes (LU-15743 LU-16760 LANL)
- **`lfs find -printf/ls`** to better display found files (LU-7495 LU-15504 LANL, WC)
- **`lfs setstripe –C -N`** creates N (over)stripes per OST, up to 32x (LU-16938 HPE)

2.16

- **`lctl list_param --path`** prints full pathname for data scraping (LU-17343 WC)

2.17

- Numerous man-page improvements for `lfs` and `lctl` sub-commands (LU-4315 WC)
- Allow split **`lctl/lfs`** sub-commands (e.g. **`mirror_foo`** -> **`mirror foo`**) (LU-18114 WC)
- **`lctl get/list_param --merge`** aggregates similar output lines (LU-14590 WC)
- **`mount.lustre`** sets client-local parameters from `/etc/lustre` (LU-11077 WC)
- **`lfs find/project/quota`** allows named projects from `/etc/projects` (LU-13335 WC)
- **`lfs migrate/mirror`** allow filenames/FIDs from file (LU-18454 WC)