

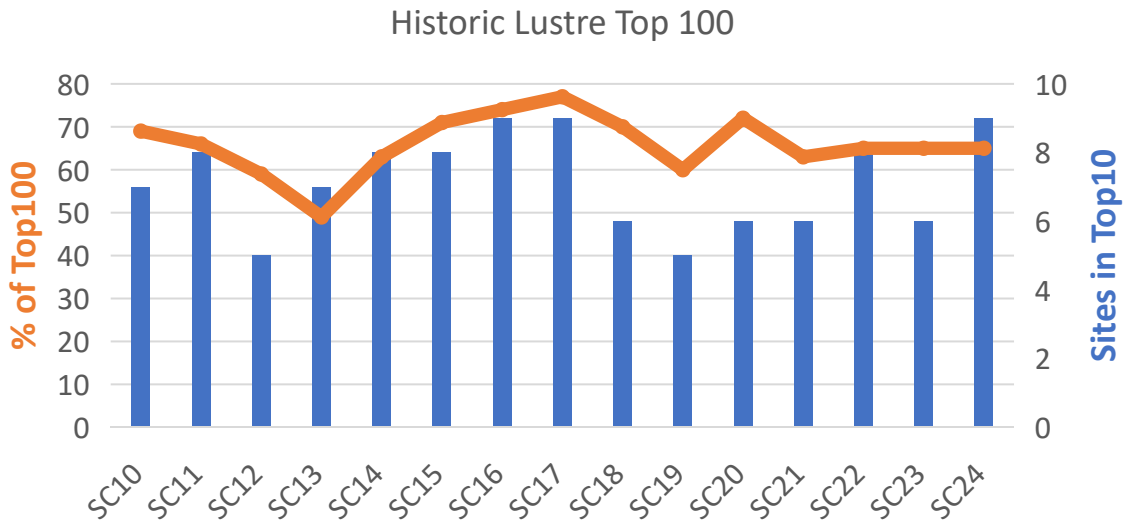
Whamcloud

Lustre update and beyond

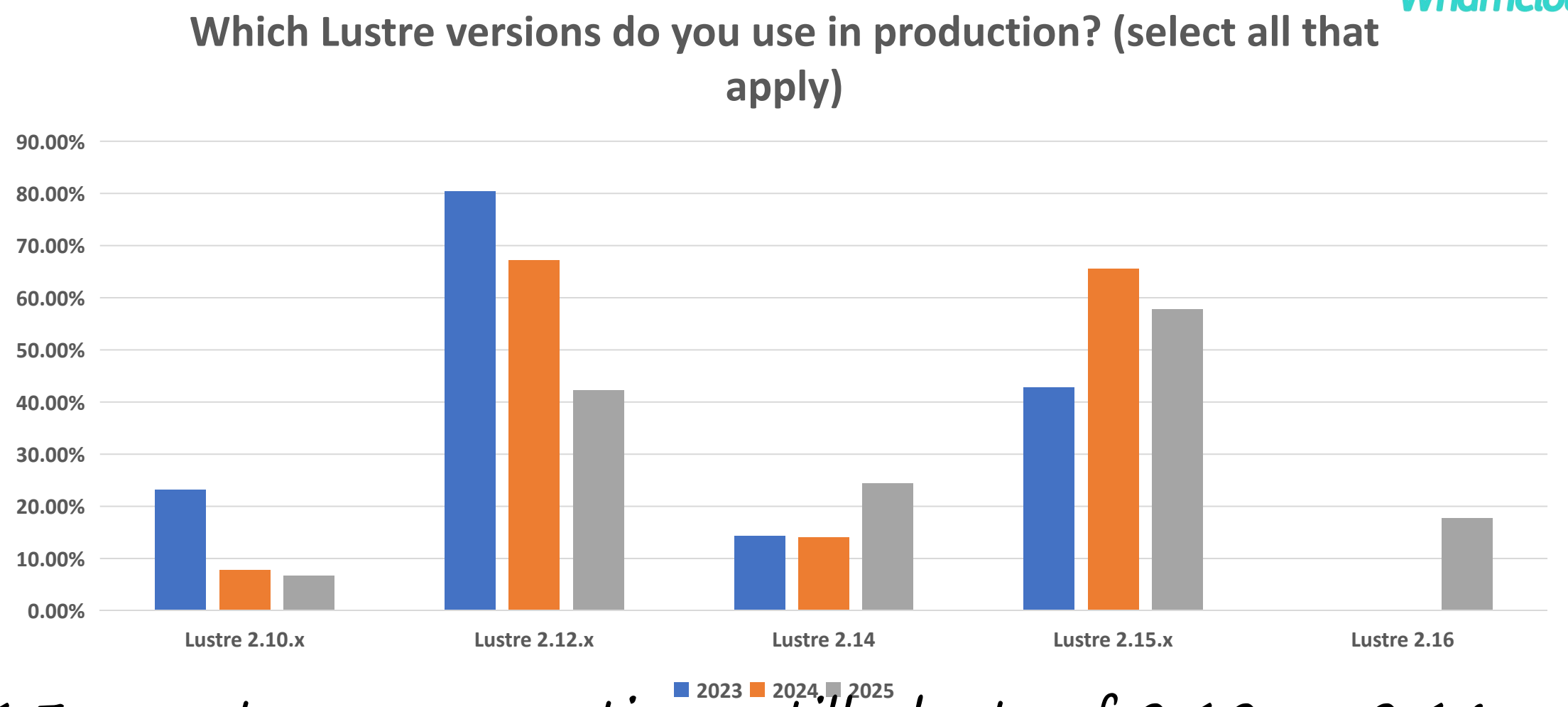
Xu Zhenyu (bobijam@whamcloud.com)

OpenSFS Lustre Working Group

- Has been operating as the forum for collaboration on the community releases for over a decade
- Led by Peter Jones (Whamcloud) and Rick Mohr (ORNL)
- All welcome to attend and/or join the mailing list
- For more information visit the wiki
- http://wiki.opensfs.org/Lustre_Working_Group



Community Survey – Lustre Versions

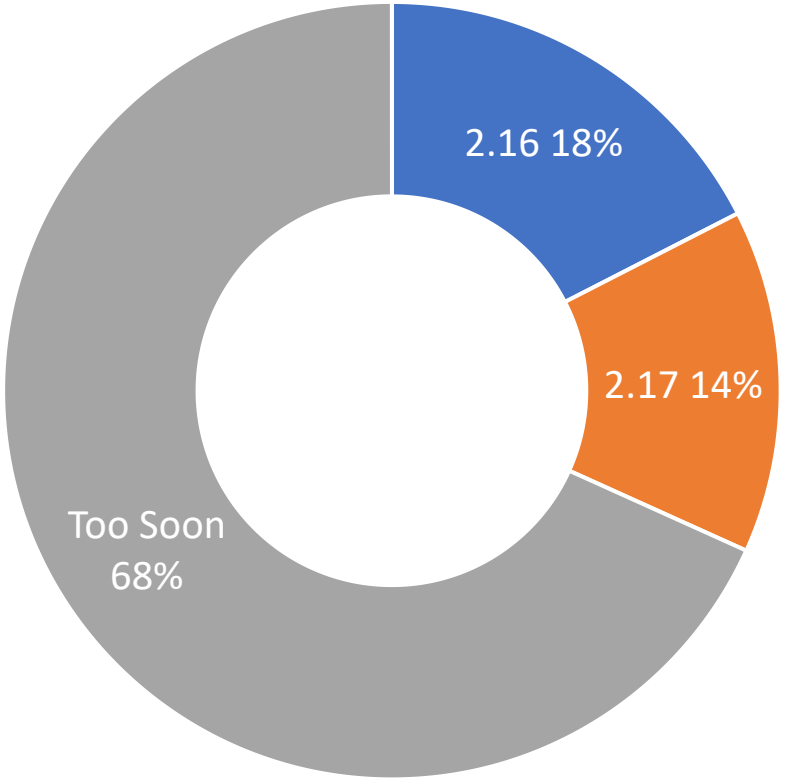


2.15.x most common option; still plenty of 2.12.x; 2.16 significant

Community Survey - Transition

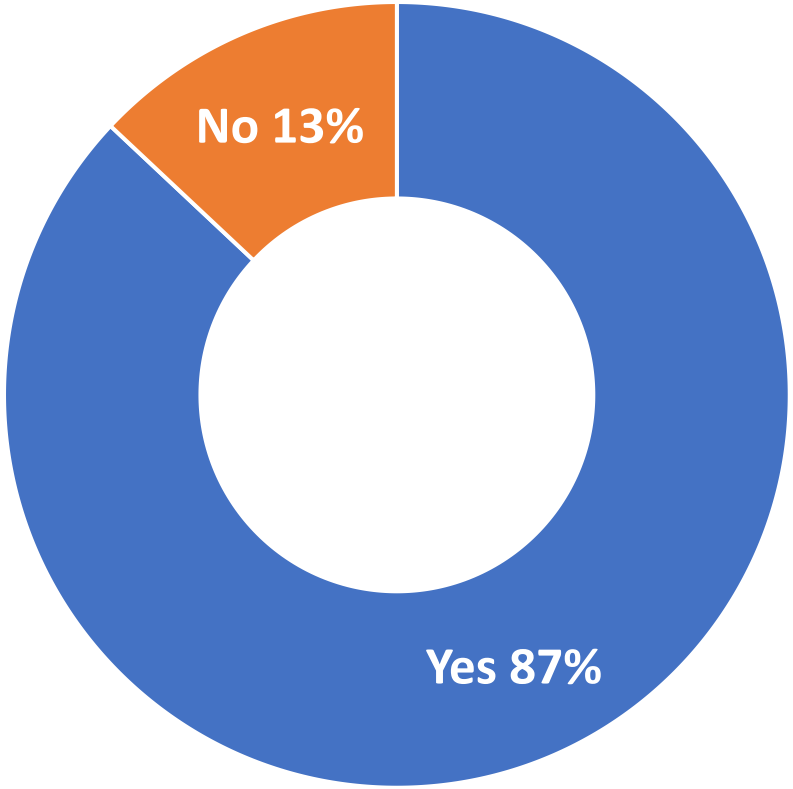
Whamcloud

Next LTS Transition



Most sites not ready to consider moving again right now

Beneficial for Lustre Upstream Client in Distro?



Strong interest in upstreaming efforts

Lustre 2.15.x LTS

- Lustre 2.15.8 expected late Q4
  - Timeline will be driven by RHEL 9.7 release

LTS	GA	Changelog
2.15.0	Jun-22	https://wiki.lustre.org/Lustre_2.15.0_Changelog
2.15.1	Aug-22	https://wiki.lustre.org/Lustre_2.15.1_Changelog
2.15.2	Jan-23	https://wiki.lustre.org/Lustre_2.15.2_Changelog
2.15.3	Jun-23	https://wiki.lustre.org/Lustre_2.15.3_Changelog
2.15.4	Dec-23	https://wiki.lustre.org/Lustre_2.15.4_Changelog
2.15.5	Jun-24	https://wiki.lustre.org/Lustre_2.15.5_Changelog
2.15.6	Dec-24	https://wiki.lustre.org/Lustre_2.15.6_Changelog
2.15.7	Jun-25	https://wiki.lustre.org/Lustre_2.15.7_Changelog

Lustre 2.16

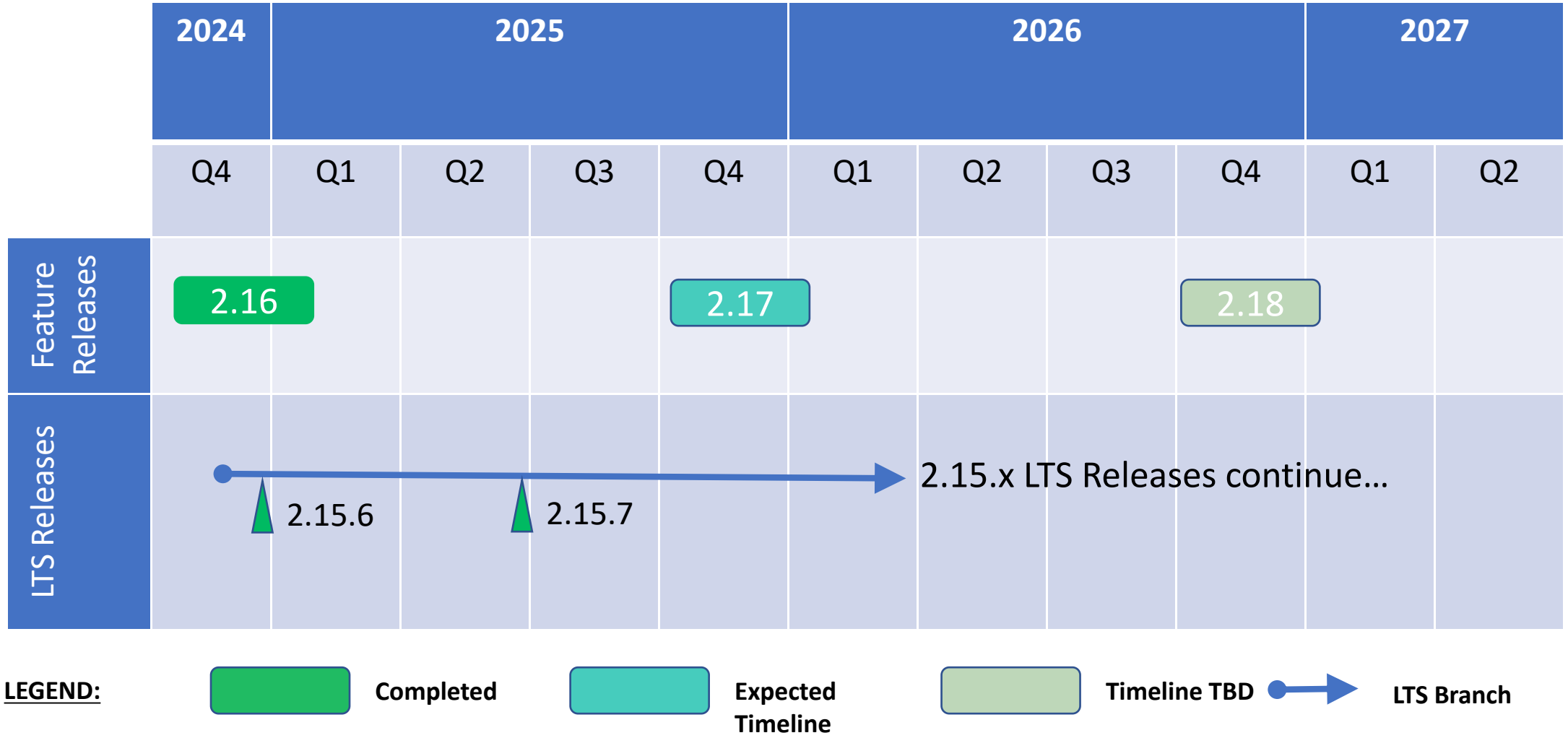
- 2.16.0 GA Nov 8th ,2024
  - http://wiki.lustre.org/Release_2.16.0
  - 2.16.1 fixes important bug (LU-18435) but otherwise identical
  - 2.16.x is NOT an LTS branch ☺
- RHEL 9.4 servers/clients
- SLES15 SP6/Ubuntu 24.04 clients
- Interop/upgrades from latest Lustre 2.15.x
- Features in the release
  - RHEL 9.x server support MERGED
  - IPv6 Support (LU-10391) MERGED
    - Final test framework change needed for routine automated testing
  - Optimized Directory Traversal (LU-15975) MERGED
    - Aka “Improved Statahead”

Lustre 2.17

- Targeted for Q4,2025
  - http://wiki.lustre.org/Release_2.17.0
- RHEL 9.6 servers/clients
- RHEL 10.0/SLES15 SP7/Ubuntu 24.04 clients
- Interop/upgrades from latest Lustre 2.15.x
- Features targeted for release:
  - Dynamic Nodemaps (LU-17431) COMPLETE
  - Erasure Coding (LU-10911) DEFERRED
  - Hybrid IO (LU-18033) COMPLETE

Lustre Community Roadmap

Whamcloud



Planned Feature Release Highlights

Whamcloud

- **2.17** major feature landings complete, release stabilization underway
  - **Hybrid IO Optimizations** – Hybrid BIO/DIO and server writeback cache (WC, Oracle)
  - **Nodemap Improvements** – manage, virtualize, isolate filesystem tenants (WC)
  - **Dynamic LNet NID Configuration** – simplify and improve dynamic network config (WC)
- **2.18** already has several features well under development
  - **File Level Redundancy - Erasure Coding** (FLR-EC) – M:N data redundancy (WC, ORNL)
  - **Client-side Data Compression** – reduce network and storage usage/cost (WC)
  - **Fault Tolerant MGS** (LMR-FTM) – replicate MGS config management across MDS (WC)
  - **Trash Can/Undelete** (TCU) – allow file recovery after accidental/malicious deletion (WC)
- **2.19** features under discussion for development
  - **Lustre Metadata Redundancy** (LMR1b) – replicate quota, flock, … services to other MDS
  - **Metadata Writeback Cache** (WBC2) – single-client metadata speedup (WC)
  - **Lustre Metadata Redundancy** (LMR2a) – ROOT/ directory mirroring to other MDTs

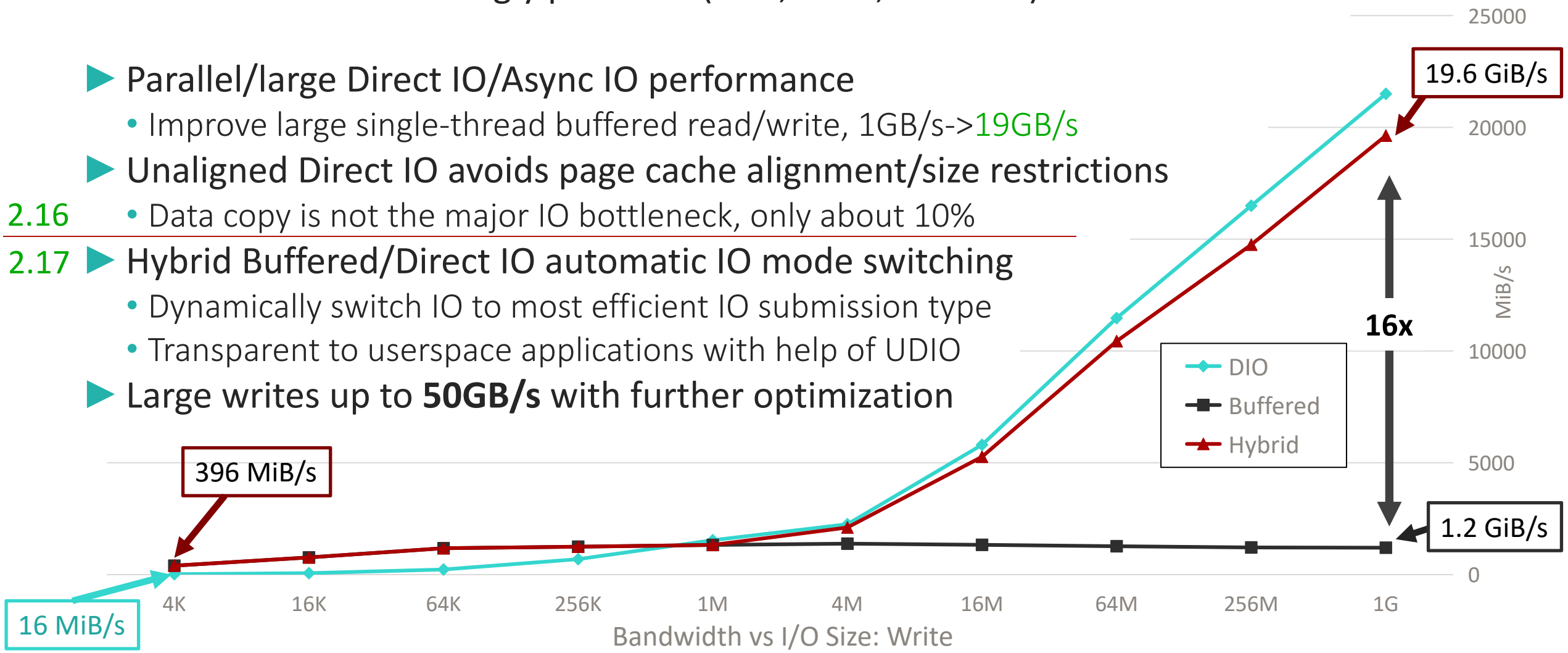
Hybrid IO: Improved Application IO Performance

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

2.16
  - Data copy is not the major IO bottleneck, only about 10%

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
- Large writes up to **50GB/s** with further optimization



Improved Data Security and User Isolation (2.17 WC)

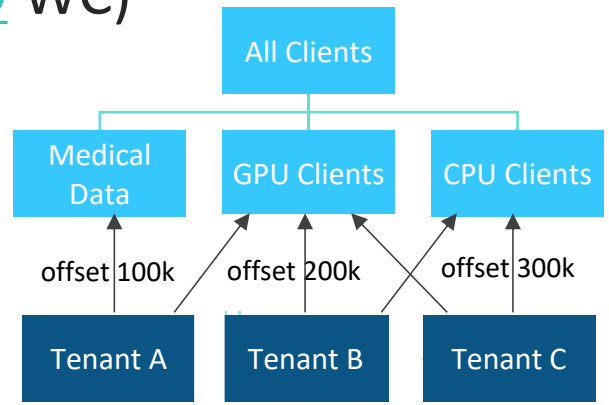
Whamcloud

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` chown/quota within nodemap (LU-18694 WC)
- Dynamic/hierarchical nodemap configuration (LU-17431 WC)
- 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
- Ban clients from filesystem via nodemap (LU-19157 WC)

2.18
- Nodemap client identification by SSK/Kerberos Key (LU-18357 WC)
- Encrypted fscrypt backup/restore without key (LU-16374 WC)





Dynamic NID Configuration (2.17 WC)

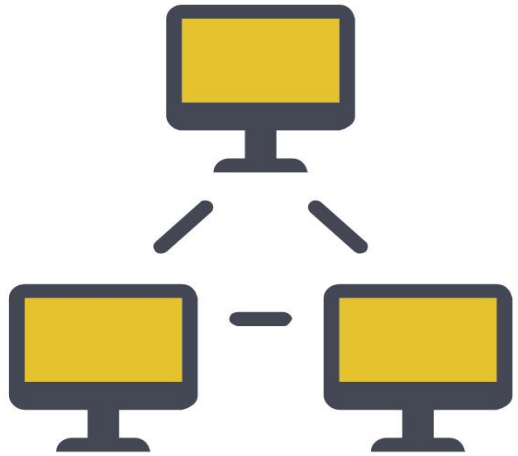
Whamcloud



- Servers can configure thousands of NIDs for tenant VLANs (LU-xxxx)
- Mount client without server NIDs in configuration logs (LU-10360, WC)
  - Servers register with MGS at mount time, 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, dynamic configuration
- Notify clients when server NIDs dynamically added/removed (LU-xxxx)
  - Filtered for clients that cannot access the new LNet due to VLAN
- Improve handling of MGS with many NIDs (LU-16738)
  - Display MGS hostname instead of NIDs in `/proc/mounts`

2.17
  - Hide ugly list of MGS NIDs for “mount” and “df” output

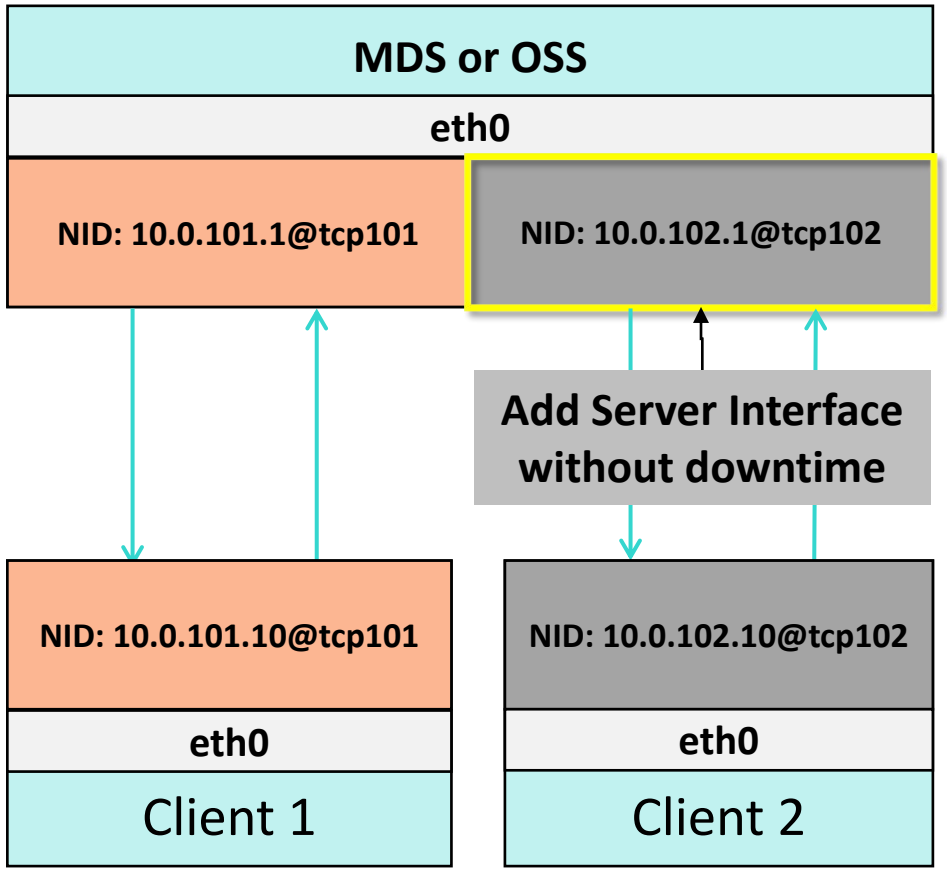
2.18
  - Round-robin DNS records for multiple MGS NIDs at mount command-line



Dynamic Server NIDs (LU-10360 2.17 WC)

Whamcloud

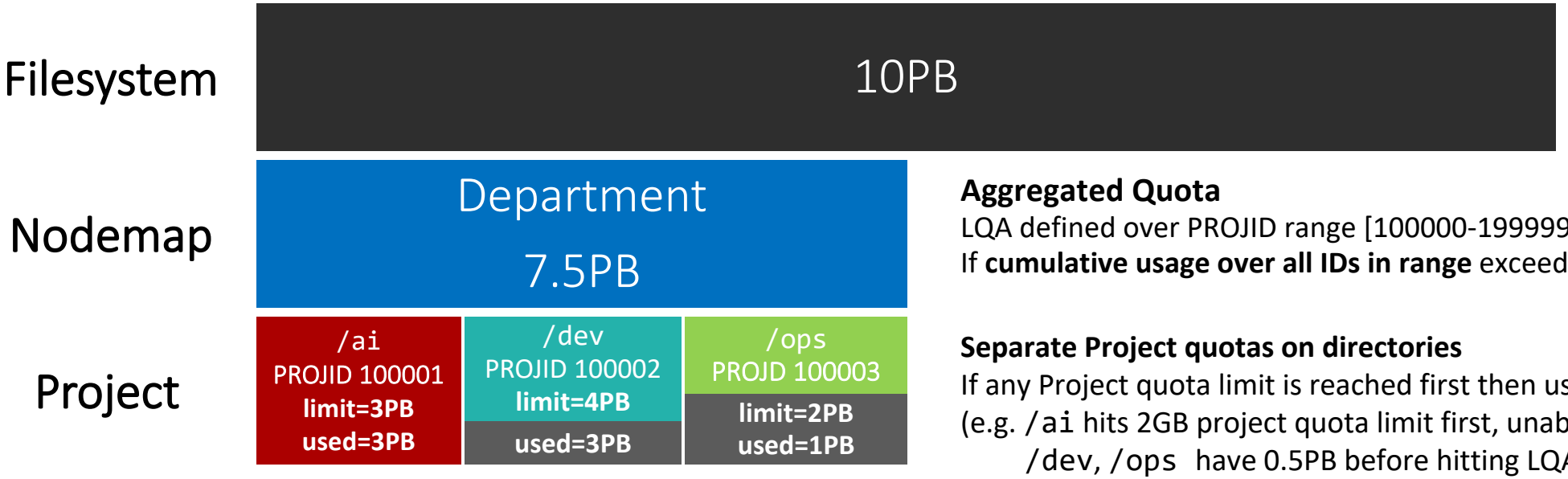
- New NID(s) can be dynamically added to servers online
  - Allow thousands of dynamic server NIDs to be configured
- No need to store dynamic NIDs in config log
  - Avoid writeconf outage during re-addressing/migration
  - More flexible for dynamic network environments
- Potentially useful for Cloud configurations
  - Move OSTs to new VM without target reconfiguration
- Dynamic NID configuration process
  - OSTs/MDTs advertise newly configured NIDs to MGS
  - MGS stores NIDs for each target in Imperative Recovery Log
  - Client uses MGS IR log to learn target NIDs if not in config log
    - Tunable to enable: `mdc.*.enable_dynamic_nids=1`



Lustre Quota Aggregation (LQA) (2.18 LU-18222 WC)

Whamcloud

- Aggregates quotas across a ranges of ProjIDs, UIDs and GIDs
  - Range quota is independent from quota on individual IDs
  - Lowest limit reached first is enforcing
- Facilitates multi-level quota for nodemaps
  - Delegate usage of Project, User, Group quota to Tenants



Aggregated Quota
LQA defined over PROJID range [100000-199999]
If **cumulative usage over all IDs in range** exceeds LQA limit then EDQUOT

Separate Project quotas on directories
If any Project quota limit is reached first then user receives EDQUOT
(e.g. `/ai` hits 2GB project quota limit first, unable to write more
`/dev`, `/ops` have 0.5PB before hitting LQA limit)

Client FLR Erasure Coded Files (LU-10911 2.18+ WC, ORNL)

Whamcloud

- 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 Nd+Mp RAID Sets (e.g. 8d+2p or 16d+3p)
  - Fixed **RAID-4** RAID Set 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	…

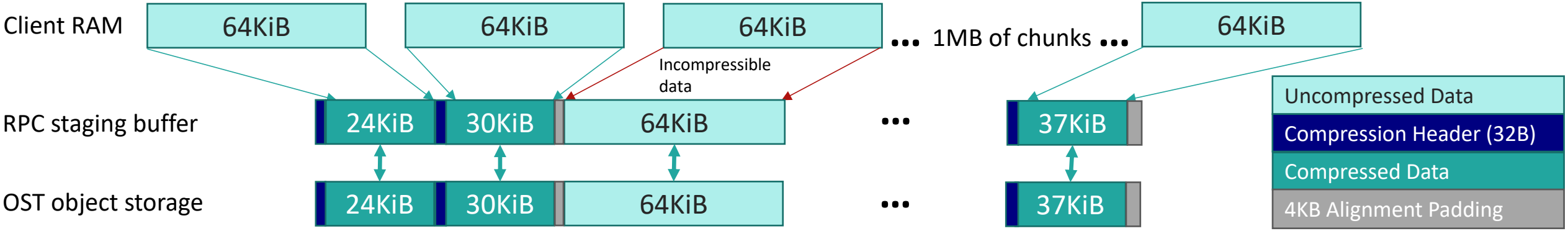
Existing stripes… Parity stripes added… Existing stripes… Parity stripes added…

Client-Side Data Compression (LU-10026 2.18 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)

Trash Can/Undelete (LU-18465 2.18 WC)

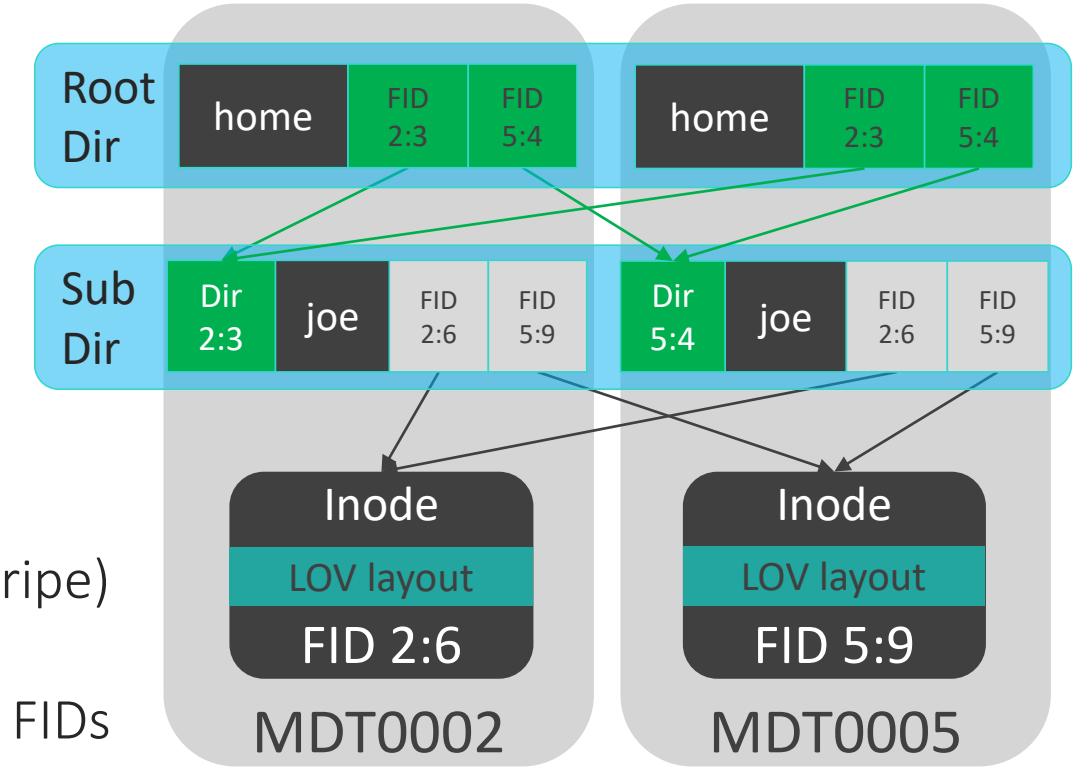
Whamcloud

- 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)



Lustre Metadata Redundancy (LMR) (2.18+)

- LMR1a: Fault Tolerant MGS (LU-17819)
- LMR1b: Replicate services to other MDTs
  - Mirror FLDB, Quota, `flock()` scaling over MDTs
- LMR1c: DNE performance (LU-7426)
  - Improve llog transaction ordering/sync
  - Improves **all** DNE operation performance
- LMR2: Replicate top-level dirs for availability
  - 2a: `ROOT/` (rarely changed) mirrored to other MDTs
  - 2b: Subdirectories mirrored to 2+ MDTs (via setdirstripe)
  - 2c: Per-file metadata replication in a later stage
  - 2d: `e2fsck` to allow directory entries with multiple FIDs
- LMR3: Complex recovery handling
  - Write to directory while mirror offline, full MDT rebuild
- LMR4: LFSCK to repair/rebuild inconsistent mirrors



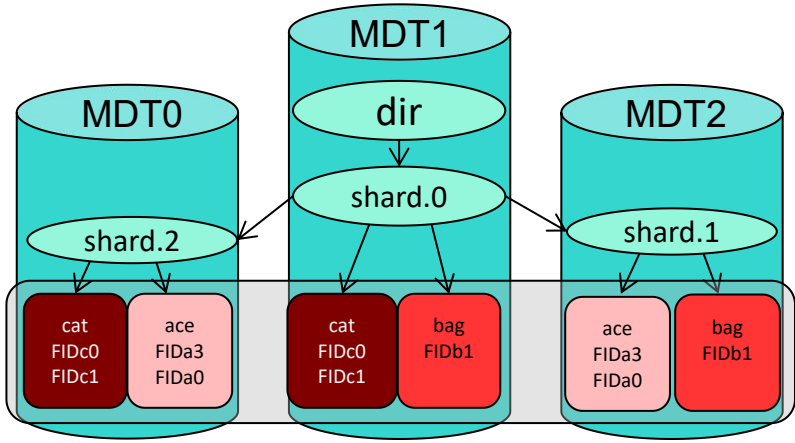
Fault Tolerant MGS (LMR-FTM) (2.18 WC)

Whamcloud

- Run MGS service on multiple MDS nodes for availability (LU-17819)
  - Allow clients to get config llogs from **any MDS 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
- Provide redundant storage for other configuration logs
  - FLDB, Quota, Nodemap, …

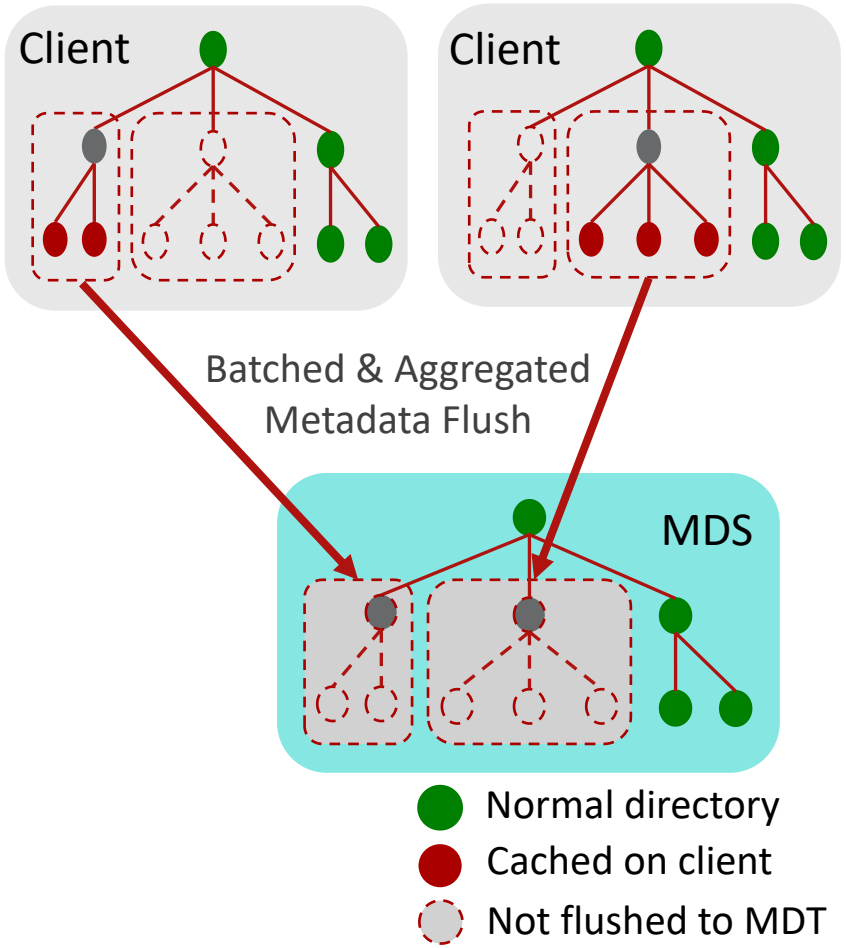


MDT0000 removal/replacement (LMR1b/2a) (2.18+)



- Remove critical service dependency on MDT0000
  - FLDB (maps FID->MDT/OST) redundancy via RAFT
  - Nodemap config redundancy via RAFT
  - Quota limit config redundancy via RAFT
- Quota Manager distributed to multiple servers
  - Hash distribution by UID/GID/PROJID, easily parallel operations
- LDLM `flock()` distribution and scaling over multiple MDS nodes
  - `flock()` locking is not really related to MDT inodes, so can run anywhere
  - Need to handle deadlock detection, but is disjoint for process groups, nodemaps
- Mirror ROOT/ directory to multiple MDTs
  - Rarely changes for most uses, some update overhead is acceptable
- Enough to allow removing/replacing MDT0000 on live system

Metadata Writeback Cache (WBC2) (WC 2.18+)

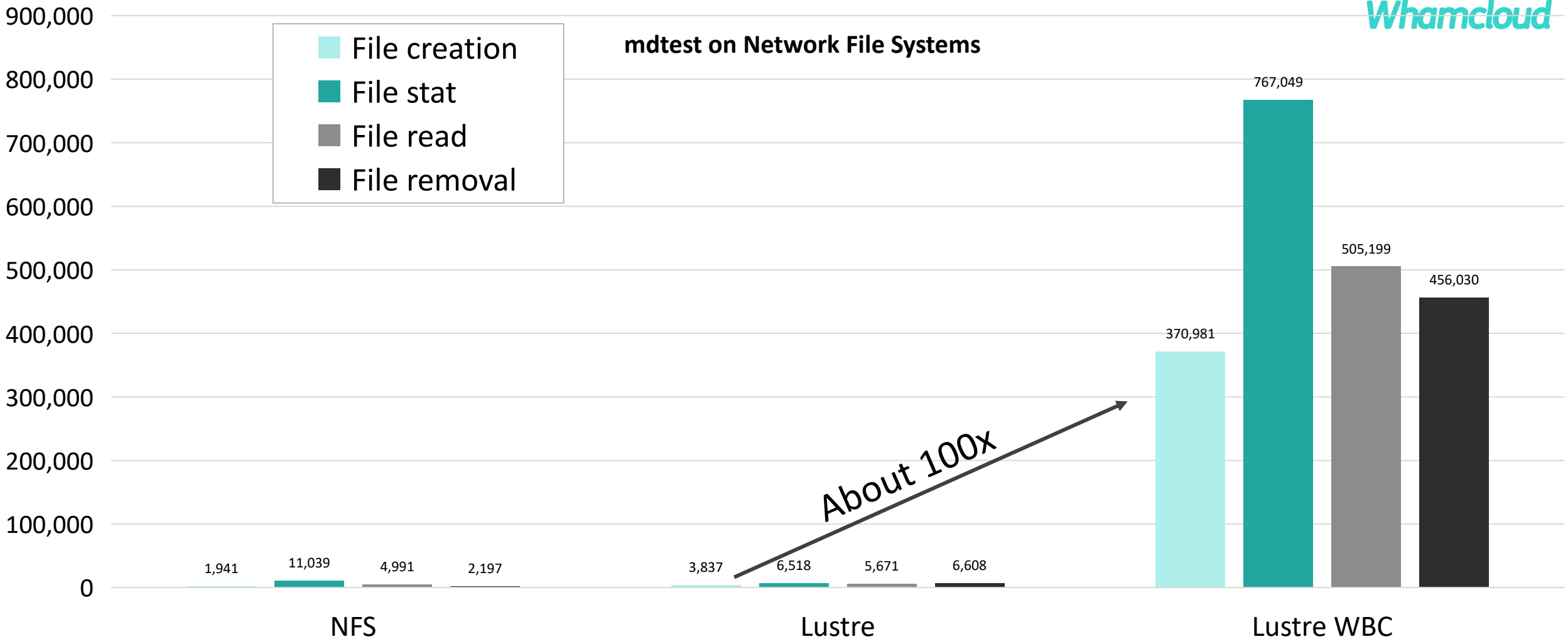


10-100x speedup for single-client create-intensive workloads

- Gene extraction/scanning, untar/build, data ingest, producer/consumer

- Create new dirs/files **in client RAM without RPCs**
  - Lock new directory exclusively at `mkdir` time
  - Cache new files/dirs/data in RAM until cache flush or remote access
- **No RPC round-trips** for file modifications in new directory
- Batch RPC for efficient directory fetch and cache flush
- **Files globally visible on remote client access**
  - Flush top-level entries, exclusively lock new subdirs, unlock parent
  - Flush rest of tree in background to MDS/OSS by age or size limits
- Productization of WBC code well underway
  - Some complexity handling partially-cached directories
  - Need to integrate space usage with quota/grant

Metadata WBC Performance Improvements



Lustre: DDN AI400X Appliance (20 X SAMSUNG 3.84TB NVMe, 4X IB-HDR100)
Lustre clients: Intel Gold 5218 processor, 96 GB DDR4 RAM, CentOS 8.1
Local Filesystem on SSD: Intel SSDSC2KB240G8



Whamcloud

Thank You!



ddn