Whamcloud

Lustre status and path forward

Sébastien Buisson

Senior Lustre Engineer

Trends in High Performance Storage

AI/ChatGPT/LLM driving surge in new parallel flash storage users.
Compute models (weather, finance, …) grow resolution, inputs, historical data.
New computing paradigms continually drive new usage storage patterns.

Many needs met by all-flash storage, but not everyone has budget to scale.
Need more capacity, reduced costs, transparently on multiple storage types.

HDD, QLC plus compression trend to lowest $/TB, more accessible than tape.

Meta/data redundancy improves availability, reduces hardware complexity.

More security, multi-tenancy, data isolation (medical, privacy, IP, legal, …).

Evolution of IO Interfaces

Whamcloud

▶ POSIX continues to be the standard IO interface

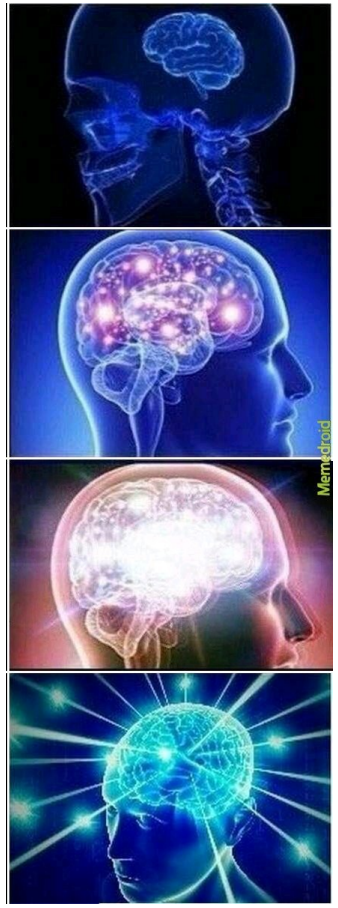
- Protects significant investment in application development
- Consistent behavior avoids chasing interface-of-the-month
- Multi-protocol access (NFS, SMB, S3) with unified namespace

▶ Performance demands continual optimization

- Unaligned IO optimizes application performance internally
- Linux `io_uring` de-facto API for async meta/data ops
- Specialized interfaces where/when applications need it

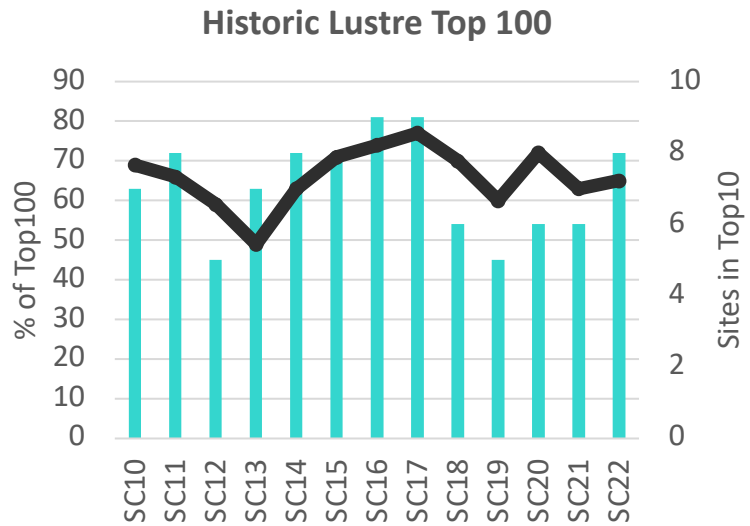
▶ Opt-in API extensions for apps with special needs

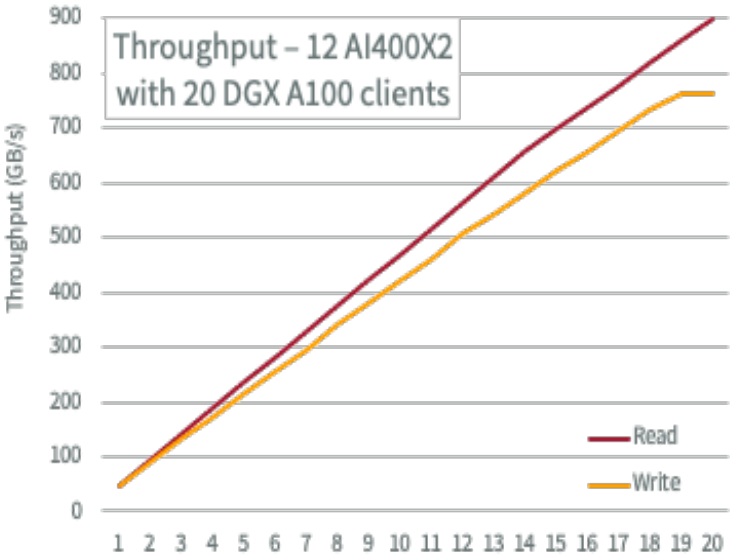
- Data also remains accessible via standard APIs afterward



Big is Beautiful

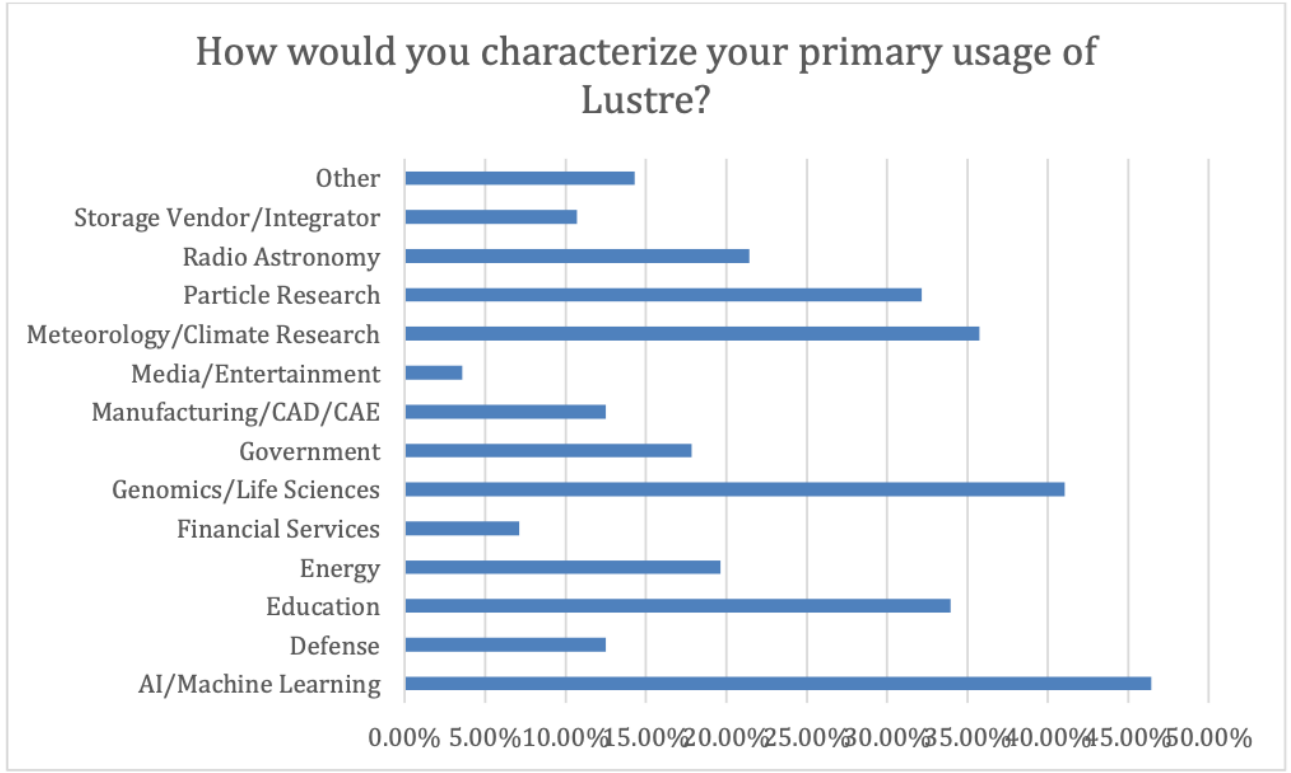
- Lustre remains the preferred choice for largest systems (8 of Top10)
  - Also 2RU servers available with 60W/80R GB/s and 3M read IOPS
- Lustre is the solution of choice for scaling AI/ML (Eos, Selene, Cam1, …)
- Proven scalability of servers and clients to meet system requirements
  - Metadata operations millions/s, 100B+ files today, ongoing improvements
  - Capacity can grow almost without limits - 100's PB today, 1 EB+ already possible ($$$)
  - 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
- Ongoing improvements for Top10 and other large system deployments
  - Steady feature development to meet evolving system and application needs
  - Working closely with multiple sites for next-generation systems in years ahead





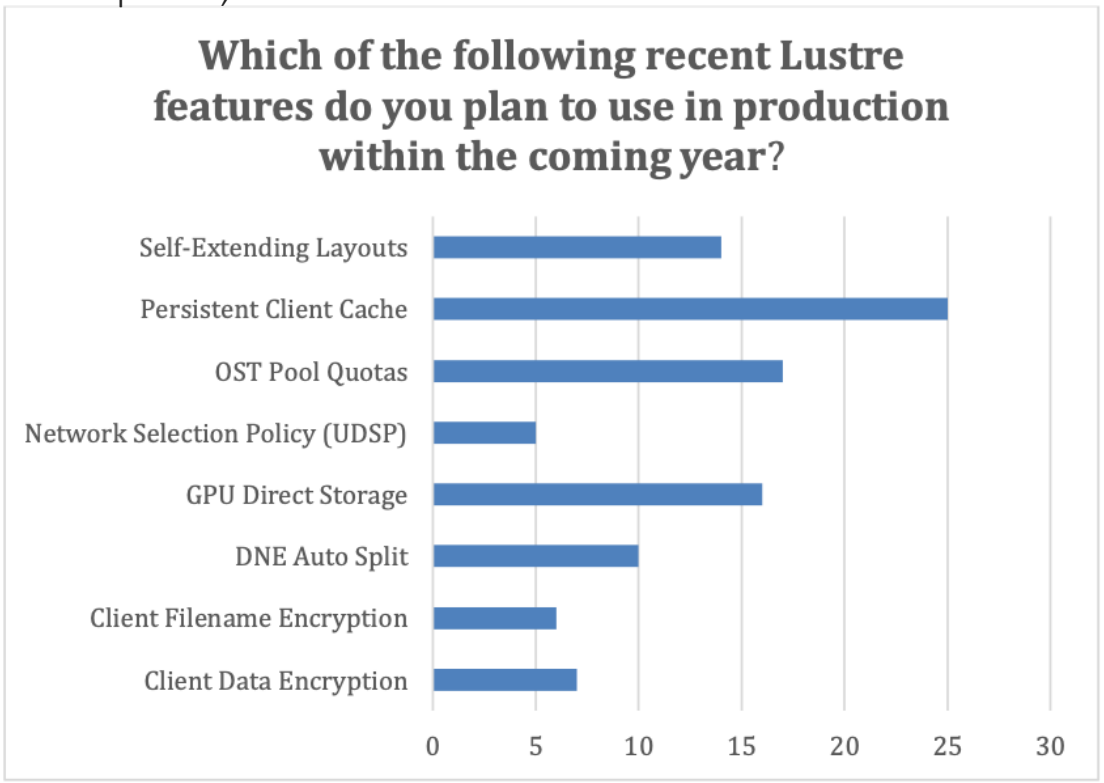
… but Size is not Everything

- Lustre at ease in a variety of market segments



https://wiki.opensfs.org/Lustre_Community_Survey

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



Lustre Community Release News

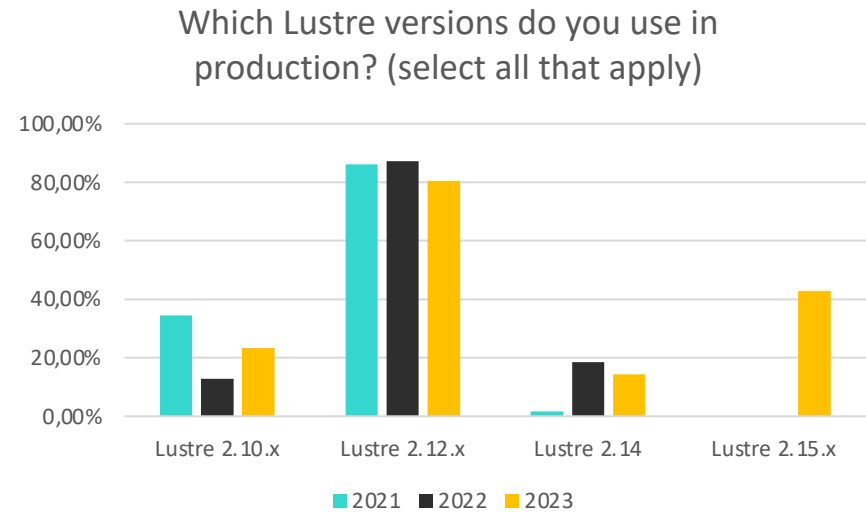
Whamcloud

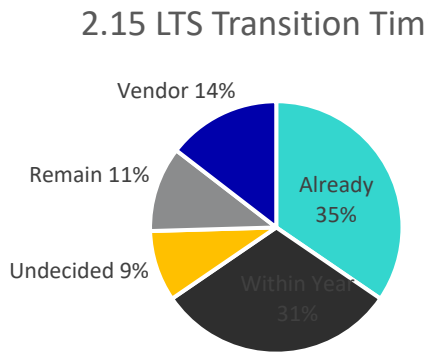
Lustre LTS Releases

- Lustre 2.15.0 GA June 2022
  - Replaced 2.12.x as LTS branch
- Lustre 2.15.3 expected in Q2
  - RHEL 8.8 servers/clients(LU-16755); RHEL 9.2 clients (LU-16756)
  - https://wiki.lustre.org/Lustre_2.15.3_Changelog

Lustre Major Releases

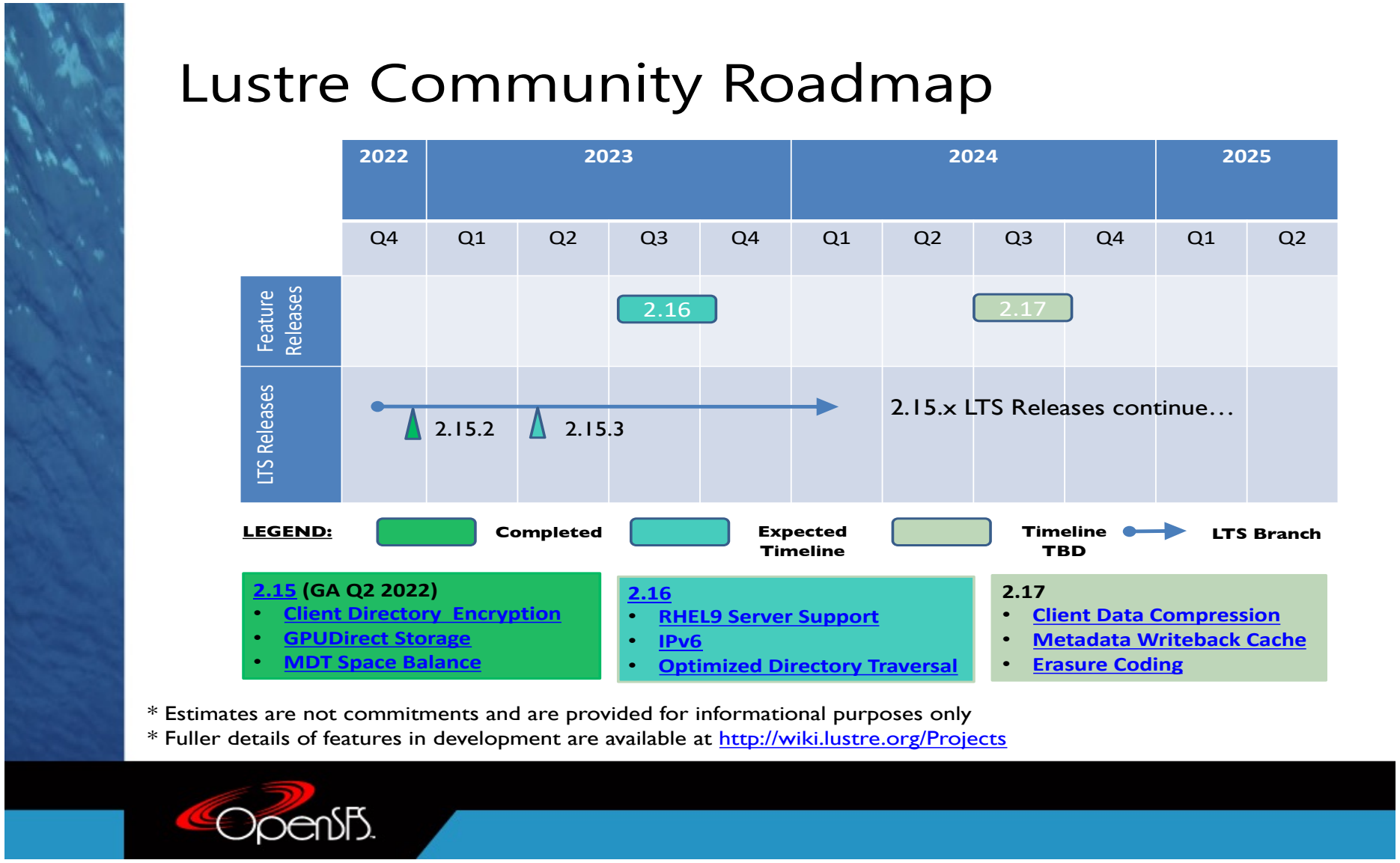
- 2.16 approaching feature completion
  - LNet IPv6 addressing – must-have functionality for future deployments
  - Optimized Directory Traversal (WBC1) – improve efficiency for accessing many files
- 2.17 development on major features well underway
  - Client-side data compression – reduce network and storage usage, costs
  - Metadata Writeback Cache (WBC2) – order of magnitude better metadata speed





https://wiki.opensfs.org/Lustre_Community_Survey

Lustre Community Roadmap



Planned Feature Release Highlights

Whamcloud

- **2.16** approaching feature completion
  - **LNet IPv6 addressing** – must-have functionality for future deployments (SuSE, ORNL)
  - **Optimized Directory Traversal** (WBC1) – improve efficiency for accessing many files (WC)
- **2.17** has major features already well underway
  - **Client-side data compression** – reduce network and storage usage, costs (WC, UHamburg)
  - **Metadata Writeback Cache** (WBC2) – order of magnitude better metadata speed (WC)
- **2.18** feature proposals in early stages
  - **File Level Redundancy - Erasure Coding** (FLR-EC) – reduce cost, improve availability (ORNL)
  - **Lustre Metadata Redundancy** (LMR1) – improve availability for large DNE systems
  - **Client Container Image** (CCI) – improved handling of aggregations of many small files

LNet Improvements

Whamcloud

Demand for IPv6 in new deployments as IPv4 is exhausted

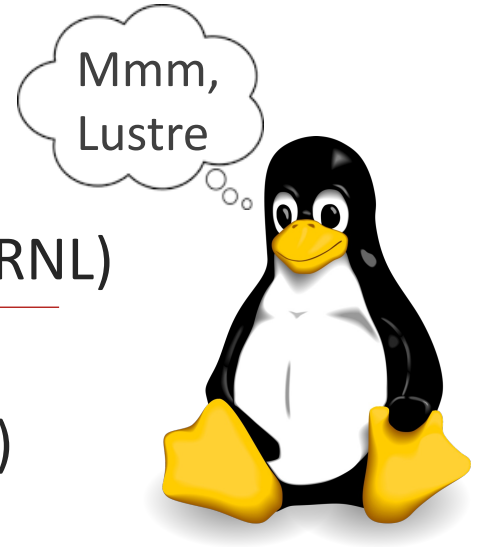
- Relatively few external-facing Lustre systems means 10.x.y.z is still viable for now

- IPv6 large NID support (LU-10391 SuSE, ORNL)
  - Variable-sized NIDs (8-bit LND type, 8-bit address size, 16-bit network number, 16-byte+ address)
  - Interoperable with existing current LNDs whenever possible
  - Enhancements to LNet/`socklnd` for large NIDs mostly finished
  - Work ongoing to handle large NIDs in Lustre code
    - Mount, config logs, Imperative Recovery, Nodemaps, root squash, etc.
- Improved network discovery/peer health (HPE, WC)
- Simplified/dynamic server node addressing (LU-14668 WC)
  - Detect added/changed server interfaces automatically (LU-10360)
  - Reduce (eventually eliminate) static NIDs in Lustre config logs
  - Simplified handling for IPv6 NIDs by clients

Client-Side Usability and Performance Improvements

Whamcloud

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



- Parallel file/directory rename within a directory (LU-12125 WC)
- `llstat`, `llobdstat` easier to use (LU-13705 WC)

2.15
- `lfs find -printf` formatted output of specific fields (LU-10378 ORNL)

2.16
- `lfs migrate` performance improvements (LU-16587 HPE, WC)
- `lfs migrate` bandwidth limit, progress updates (LU-13482 Amazon)
- Ongoing code updates/cleanup for newer kernels (ORNL, HPE, SuSE)

2.17
- Client-side Data Compression (LU-10026 WC, UHamburg, Intel)
- Buffered/DIO performance/efficiency improvements (LU-13805, LU-14950 WC)
- Erasure Coded FLR files (LU-10911 ORNL)

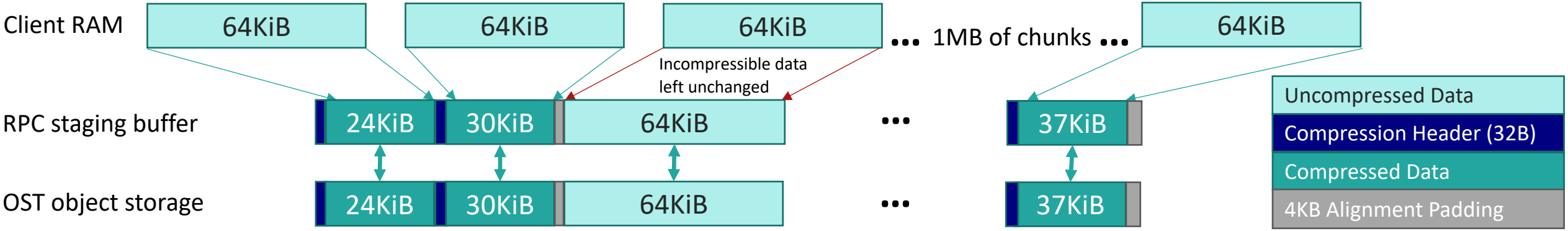
Client-Side Data Compression (WC, UHamburg 2.17+)

Whamcloud

Increased capacity and lower cost for all-flash OSTs

- Parallel compression of RPCs on client cores for GB/s speeds, **no server CPU overhead!**

- (De-)Compress (`lzo`, `lz4`, `gzip`,…) RPC on client in chunks (64KiB-1MiB+) (LU-10026)
  - **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
- Optional write uncompressed to one FLR mirror for random IO pattern
- Optional data (re-)compression during mirror/migrate to slow tier (via data mover)

Server-side Capacity and Efficiency Improvements

Whamcloud

Ongoing performance and capacity scaling for next-gen hardware and systems

- OST object directory scalability for multi-PB OSTs (LU-11912 WC)
  - Regularly create new object subdirectories (every 32M creates vs. 4B creates)
  - Better handling for billions of objects, grouping by age optimizes RAM and IOPS
- Read-only mounting of OST and MDT devices (LU-15873 WC)
- Improved `e2fsck` for large dir and shared block errors (LU-14710, LU-16171 WC)
- `lljobstat` utility for easily monitoring "top" jobs (LU-16228 WC)

2.16

  - Add IO size histograms to job_stats output, handle bad job names better

2.17

- Reduced transaction size for many-striped files/dirs (LU-14918 WC)
- Improved ldiskfs mballoc efficiency for large filesystems (LU-14438 Google, WC, HPE)
- Parallel e2fsck for pass2/3 (directory entries, name linkage) (LU-14679 WC)

Improved Data Security and Containerization

Growing dataset sizes and varied uses increases need to isolate users and their data

- Filenames encrypted on client in directory entries (LU-13717 WC)
- Migrate/mirror of encrypted files without key (LU-14667 WC)
- Encrypted file backup/restore/HSM without key (LU-16374 WC)

2.15
- Nodemap project quota mapping, squash all files to project (LU-14797 WC)

2.16
- Read-only mount enforced for nodemap clients (LU-15451 WC)
- Kerberos authentication improvements (LU-16630, LU-16646 WC, NVIDIA)
- Nodemap Role-Based Admin Controls (fscrypt, changelog, chown, quota) (LU-16524 WC)
- Cgroup/memcg memory usage limits for containers/jobs on clients (LU-16671 WC, HPE)

Metadata Scaling Improvements (WC 2.15+)

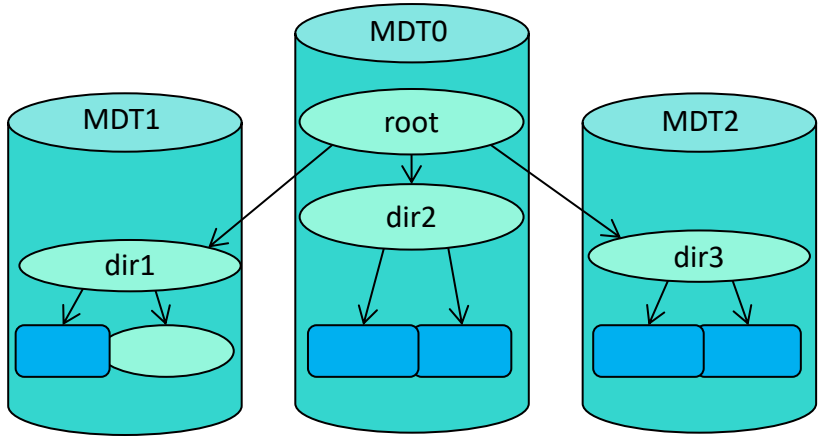
Improve usability and ease of DNE metadata horizontal performance/capacity scaling

- **DNE MDT Space Balance** - load balancing with normal `mkdir` (LU-13417, LU-13440)
  - Round-robin/balanced subdirs, limited layout inheritance depth, less need for striped dirs
- Single-dir migration without recursion - "`lfs migrate -m -d <dir>`" (LU-14975)

2.15
- Balanced migration prefers less full MDTs - "`lfs migrate -m -1 <dir>`" (LU-13076)

2.16
- DNE inode migration improvements (LU-14719, LU-15720)
  - Pre-check target space, stop on error, improved CRUSH2 hash
- More robust DNE MDT llog recovery (LU-16203, LU-16159)
  - Handle errors and inconsistencies in recovery logs better
- DNE locking, remote RPC optimization (LU-15528)
  - Distributed transaction performance, reduce lock contention

2.17
- Lustre Metadata Robustness/Redundancy (LU-12310)
  - Phase 1 to distribute/mirror MDT0000 services to other MDTs



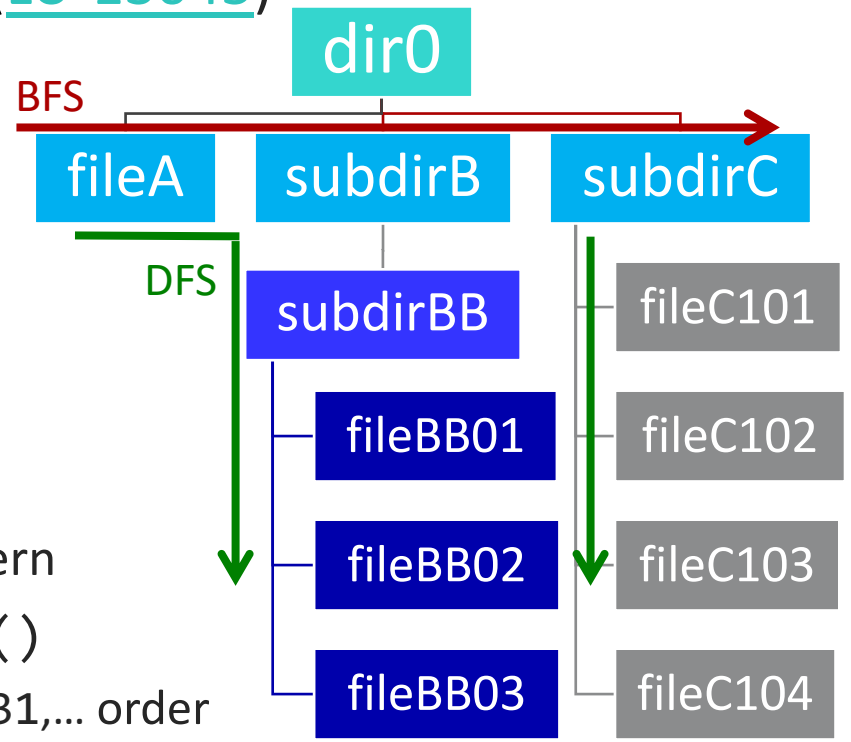
Batched Cross-Directory Statahead (WBC1) (WC 2.16)

Whamcloud

Improved access speed and efficiency for large directories/trees

- IO500 `mdtest-{easy/hard}-stat` performance improved 77%/95%

- **Batched RPC** infrastructure for multi-update operations (LU-13045)
  - Allow multiple getattrs/updates packed into a single MDS RPC
  - More efficient network and server-side request handling
- **Batched statahead** for `ls -l`, `find`, etc. (LU-14139)
  - Aggregate getattr RPCs for existing statahead mechanism
- **Cross-Directory statahead** pattern matching (LU-14380)
  - Detect breadth-first (**BFS**) depth-first (**DFS**) directory tree walk
  - Direct statahead to next file/subdirectory based on tree walk pattern
  - Detect strided pattern for alphanumeric ordered traversal + `stat()`
    - e.g. `file00001`,`file001001`,`file002001`... or `file1`,`file17`,`file31`,... order



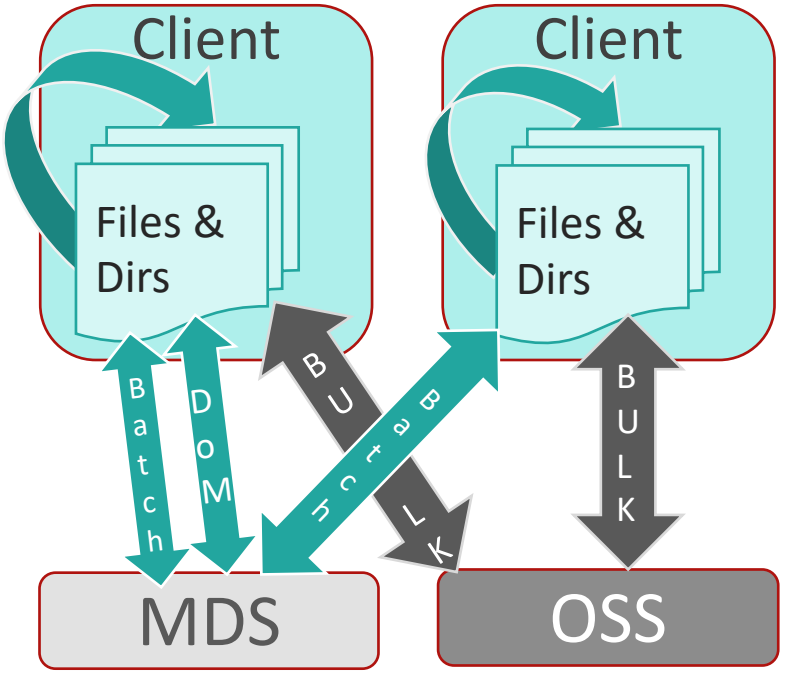
Metadata Writeback Cache (WBC2) (WC 2.17+)

Whamcloud

10-100x speedup for single-client file/dir create-intensive workloads

- Genome extraction/processing, untar/build, data ingest, producer/consumer

- Create new dirs/files **in client RAM without RPCs** (LU-10983)
  - 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
  - Repeat as needed for subdirectories being accessed remotely
  - 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
  - Able to benchmark under intensive multi-client workloads
- WBC for pre-existing directories, PCC integration in later release
  - Read all directory entries before create to avoid duplicate filenames



Thank You!
Questions?