Lustre Community BOF
Lustre in HPC, AI and the Cloud

November 19, 2019

SC19 Denver, CO hpc is now.

EOFS European Open File System

OpenSFS

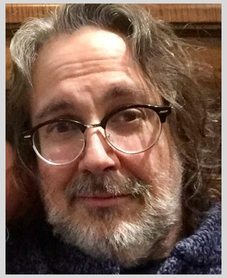
Agenda

- Welcome - OpenSFS / EOFS
- Lustre Trademark
- Community Events
- Lustre Community Release Update
- Lustre Features and Roadmap
- HPC
- AI / Cloud

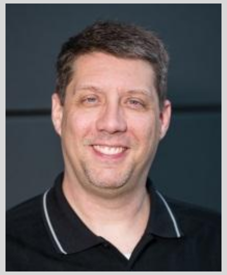
OpenSFS and the Lustre Community

- OpenSFS facilitates a community around Lustre
  - Organization for both Vendors (Participants) and Users (Members) to discuss features and directions
- Promote Lustre and the Lustre community
  - Ensure Lustre remains vendor-neutral, open, and freely downloadable
- Organize the Lustre Users Group conference
- Does Not
  - Fund Lustre development
  - Provide Lustre support (but we can help!)

2019 OpenSFS Board



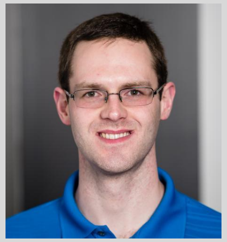
President: **Steve Simms**
IU



Vice President: **Kevin Harms**
ANL

Secretary: **Ken Rawlings**
IU

Treasurer: **Kirill Lozinskiy**
LBNL / NERSC



Director at Large: **Shawn Hall**
BP

Director-at-Large: **Sarp Oral**
ORNL

OpenSFS Supporters

AEON COMPUTING

CRAY

DataDirect™ NETWORKS

FUJITSU

Hewlett Packard Enterprise

intel

NetApp

RAID INCORPORATED

Argonne NATIONAL LABORATORY

arm

BAE SYSTEMS INSPIRED WORK

bp

cea

ConocoPhillips

CSCS

Fermilab

fs fuse SOLUTIONS

Hewlett Packard Enterprise Data Science Institute University of Houston

INDIANA UNIVERSITY

iRODS

BERKELEY LAB

Lawrence Livermore National Laboratory

Los Alamos NATIONAL LABORATORY EST. 1943

NASA

NCSA

NOAA NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION U.S. DEPARTMENT OF COMMERCE

OAK RIDGE National Laboratory

PGS

SDSC SAN DIEGO SUPERCOMPUTER CENTER

SIG SUSQUEHANNA

Sandia National Laboratories

SRCC

TACC

UF UNIVERSITY of FLORIDA

Advanced Technologies Group UBER

OpenSFS®

OpenSFS Membership

- Thank you to all our current and past Supporters
- Member
  - $1,000 annually
  - For Lustre users
- Participant
  - $5,000 annually
  - For Lustre providers
- If you are at this meeting and not a Supporter, why?

Lustre Trademark

- Seagate has transferred ownership of the Lustre trademark to the combined ownership OpenSFS and EOFS
  - Thank you to Seagate!

- Seagate retains rights to use trademark as part of the agreement

Lustre Community Release Update

- Peter Jones - Director Of Engineering at Whamcloud, Inc
  - Lustre Working Group Co-Chair

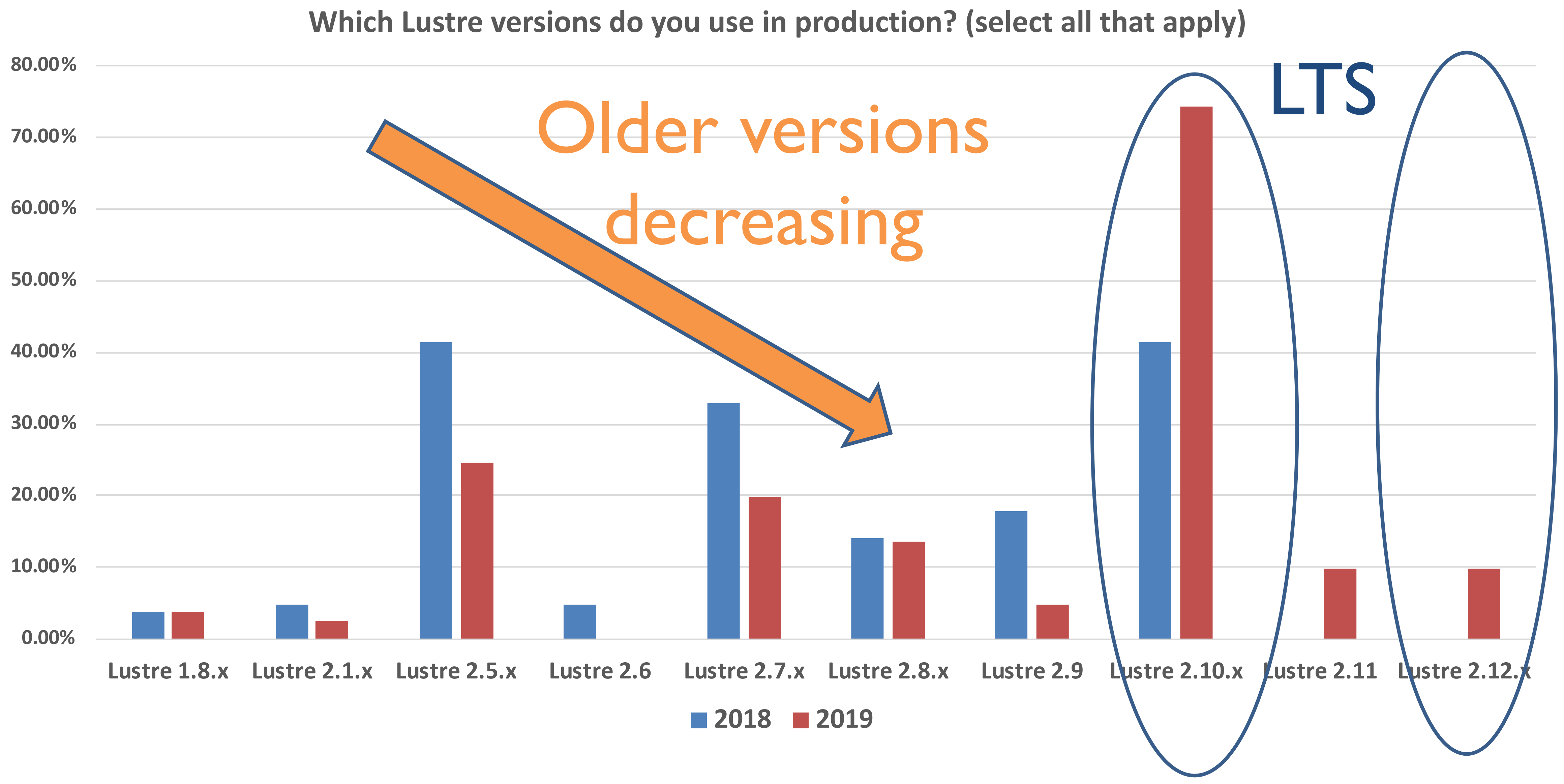
- Lustre Working Group (LWG)
  - Peter Jones (Whamcloud) & Dustin Leverman (Oak Ridge)
  - Every other Thursday - 11AM PT / 12PM MT / 1PM CT / 2PM ET
  - Dialin: +1 (916) 469-2710 - Conference ID: 71884717
  - http://wiki.opensfs.org/Lustre_Working_Group

Lustre Community Release Update

SC19

Peter Jones, Whamcloud
OpenSFS Lustre Working Group

Lustre Community Survey (Mar 2019)



Lustre 2.10.x LTS is the most widely-used production release

Lustre LTS Transition

- LTS change from 2.10.x to 2.12.x announced at SC18
  - Driver was amount of change to support newer kernels
- Some concerns have been raised that two years is not LT ☺
- Large numbers of sites deployed 2.10.x LTS releases
  - Some sites upgraded fully to 2.12.x
  - Some sites using 2.12.x clients
  - Some sites work with vendors who offer longer support versions
  - Some sites using changes ported back to 2.10.x if required
- Will again seek community input before any future changes

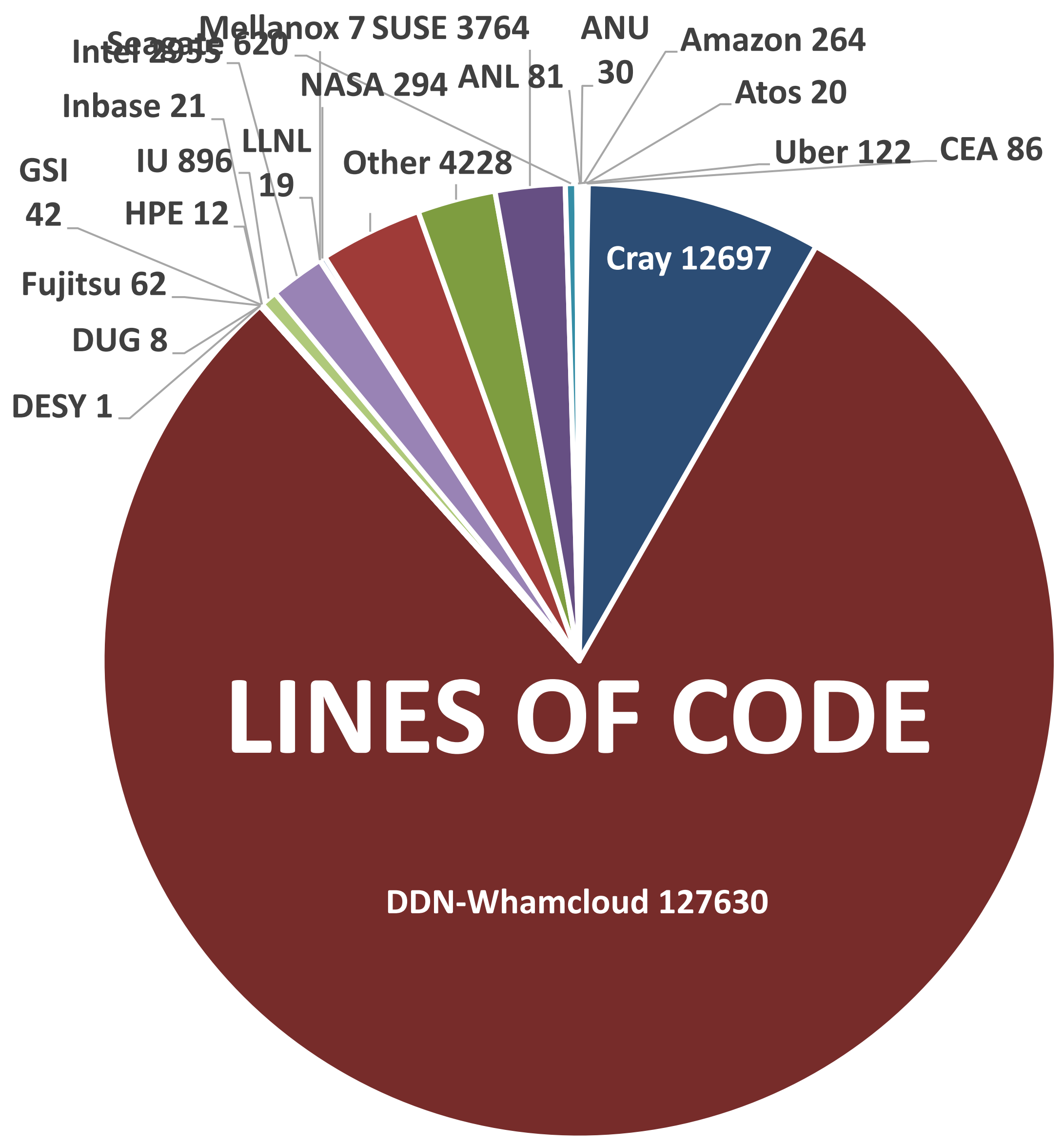
Lustre 2.12.x LTS

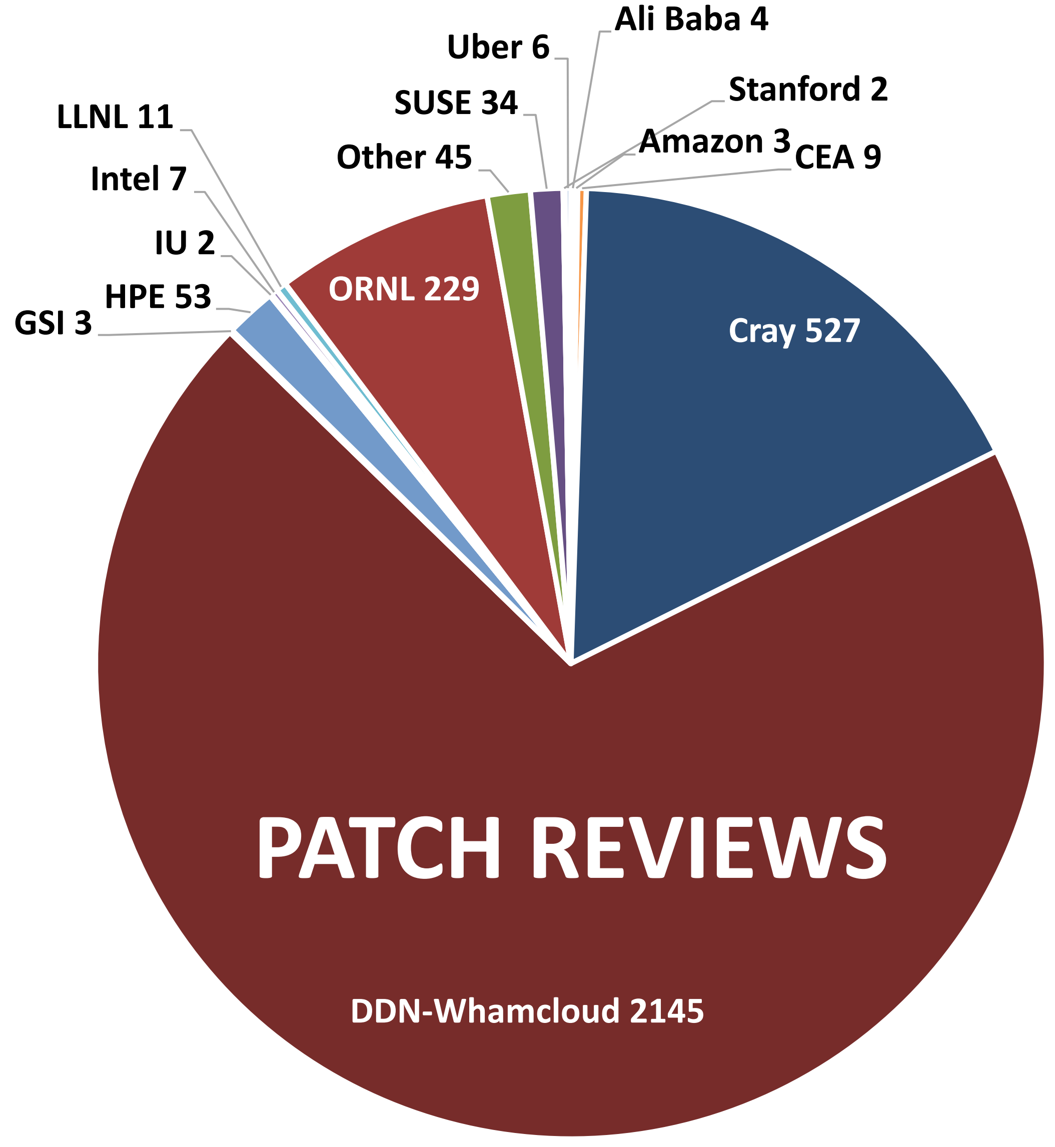
- Lustre 2.12.3 went GA Oct 21st
  - RHEL 7.7 server and client support
  - RHEL 8.0 client support
  - MOFED 4.7
  - ZFS 0.7.13
  - Bug fixes from early 2.12.x deployments
  - http://wiki.lustre.org/Lustre_2.12.3_Changelog
- Lustre 2.12.4 targeted for late Q4
  - RHEL 8.1 client support

Lustre 2.13

- RC1 in release testing ATM; hopefully GA by end of month
- OS support
  - RHEL 7.7 servers/clients
  - RHEL8.0/SLES12 SP4/Ubuntu 18.04 clients
- Interop/upgrades from latest Lustre 2.12.x
- ZFS version ships with ZFS 0.7.13 by default
- Number of useful features
  - Persistent Client Cache (LU-10092)
  - Overstriping (LU-9846)
  - Self Extending Layouts (LU-10070)
- http://wiki.lustre.org/Release_2.13.0

Lustre 2.13 Contributions





Aggregated data by organization between 2.12.50 and 2.12.90 tags

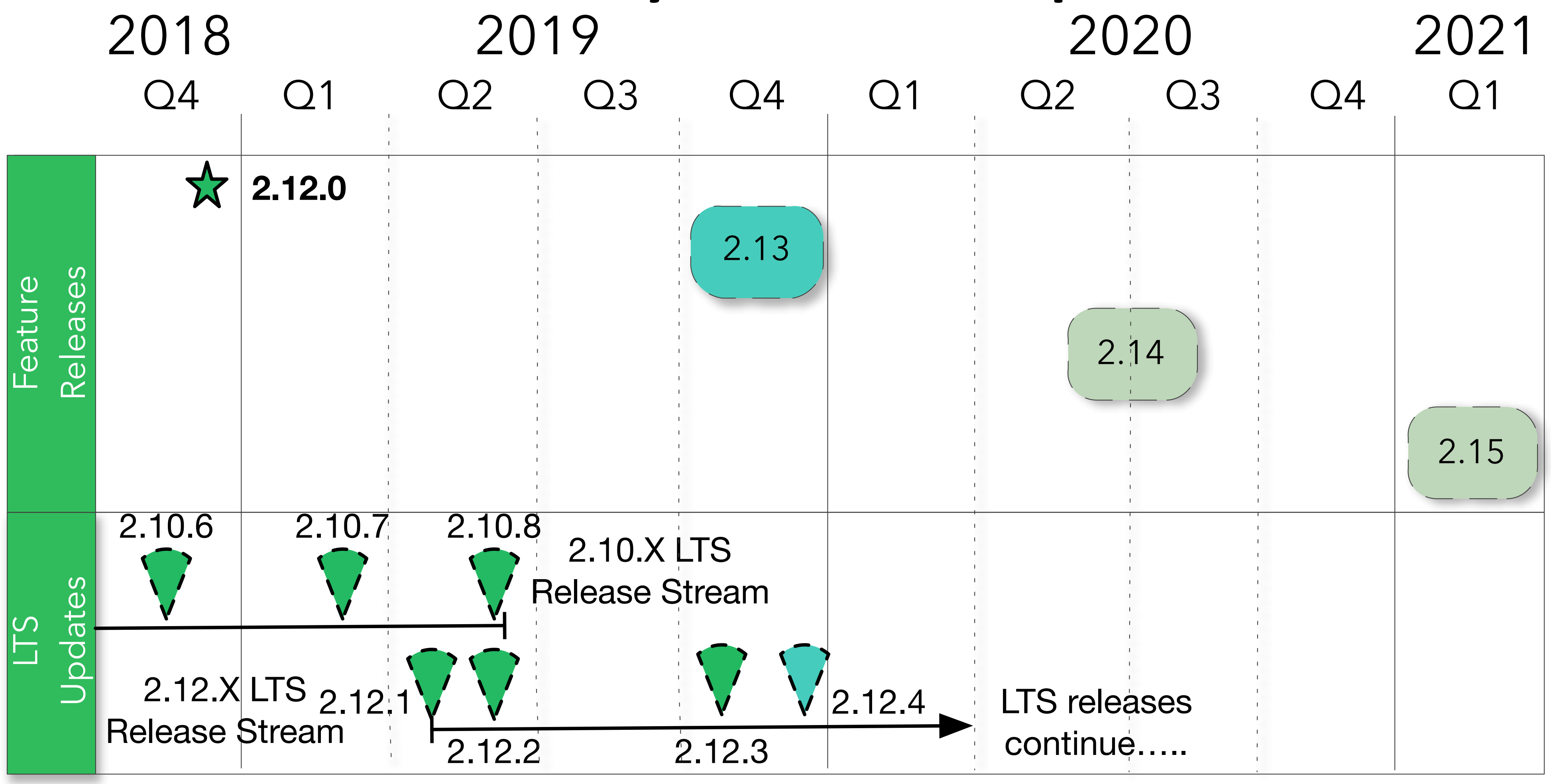
Source: http://git.whamcloud.com/fs/lustre-release.git/shortlog/refs/heads/master

Data courtesy of Dustin Leverman (ORNL)

Lustre 2.14

- Targeting late Q2/early Q3 release
- OS support
  - RHEL 8.1 servers/clients
  - RHEL 8.1/SLES15 SP1/Ubuntu 20.04 clients
- Interop/upgrades from 2.13 and latest Lustre 2.12.x
- ZFS version ships with ZFS 0.8.2 by default
- Number of useful features
  - FLR Erasure Coding (LU-10911)
  - Pool Quota (11023)
  - DNE Auto Restriping (LU-11025)
- http://wiki.lustre.org/Release_2.14.0

Lustre Community Roadmap



LEGEND:

🟦	Expected Timeline
▨	Timeline TBD
🟩	Completed
↓	LTS Branch

2.12
- Lazy Size on MDT
- LNet Health
- DNE Dir Restriping

2.13
- Persistent Client Cache
- Multi-Rail Routing
- Overstriping

2.14
- FLR Erasure Coding
- Pool Quota
- DNE Auto Restriping

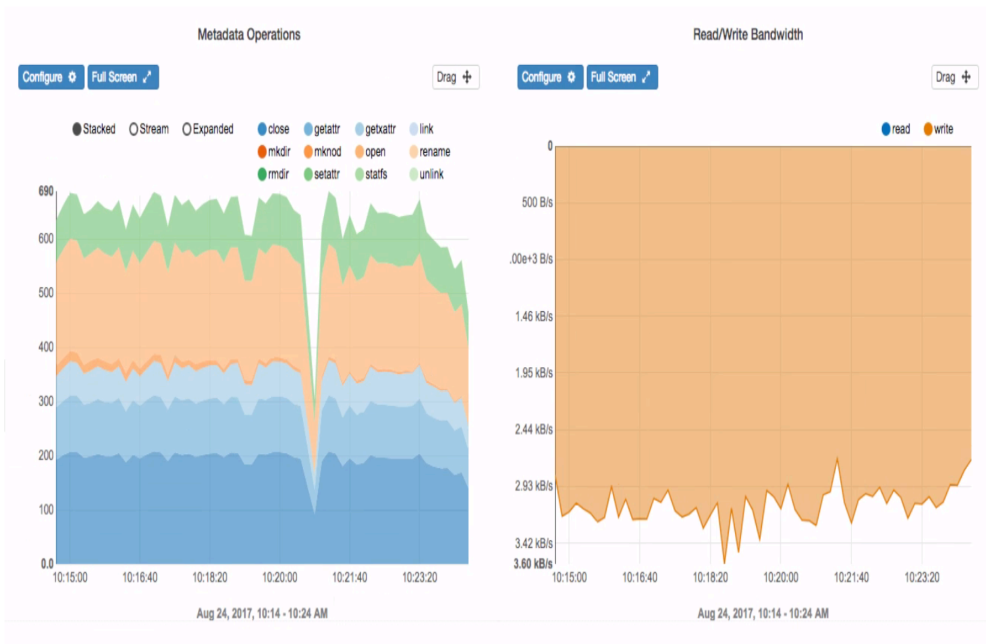
2.15
- Client Encryption
- Writeback Cache

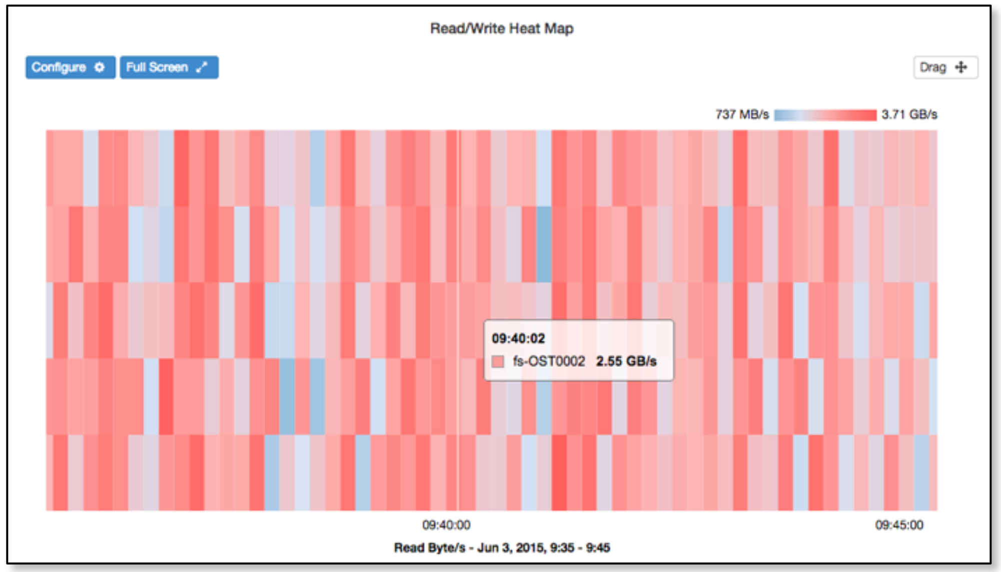
* Estimates are not commitments and are provided for informational purposes only
* Fuller details of features in development are available at http://wiki.lustre.org/Projects

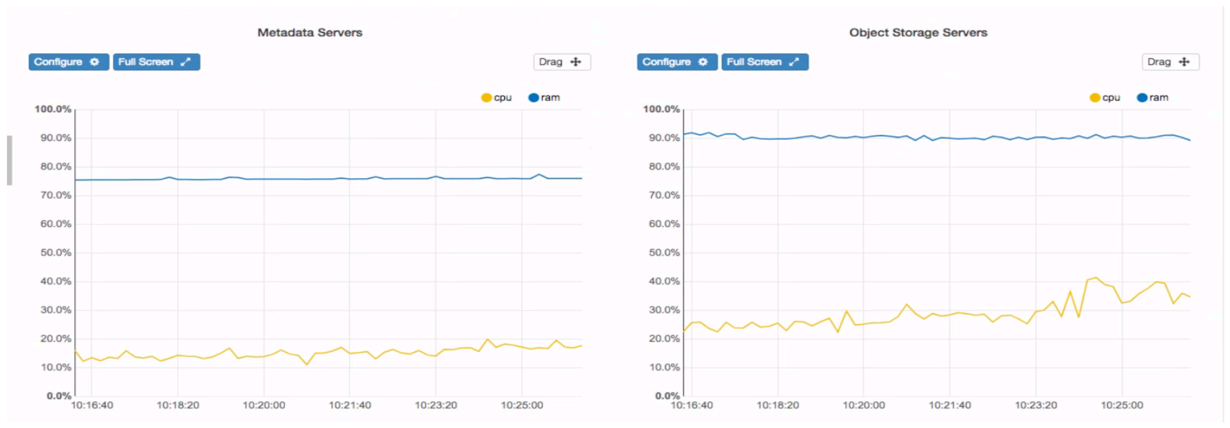
Integrated Manager for Lustre

https://github.com/whamcloud/integrated-manager-for-lustre/releases

- IML 5.0 GA in May
  - Lower manager/agent resource usage
- IML 5.1 now GA
  - https://github.com/whamcloud/integrated-manager-for-lustre/releases/tag/v5.1.0
  - Lustre 2.12.3 support
  - Parallel deploy servers with CLI







Lustre's Longevity

- Project active for over 20 years
- Maintaining stability and performance in the most demanding environments is not an easy problem to solve
- Constant evolution to deal with changing requirements of hardware and usage
- Permissive open source license has meant many organizations can use and contribute effectively
- For some further Lustre heritage…

https://hps.vi4io.org/_media/events/2019/hpc-iodc-lustre_next_20_years-dilger.pdf

Lustre at SC19

- **Sat Nov 16th**
  - HPCAST (Grand Hyatt) 8 am
- **Tue Nov 19th**
  - DDN booth (#617) 1:15pm
  - IO-500 BOF (205-207) 12:15pm
  - Arm BOF (301-303) 5:15pm
  - OpenSFS Lustre BOF (205-207) 5:15pm
- **Weds Nov 20th**
  - SUSE booth (#1917) 3pm
  - Stanford booth (#1255) 3:30pm
- **Thur Nov 21st**
  - Stanford booth (#1255) 1:30pm

Summary

- LTS model has been well adopted
- Lustre 2.12.3 and IML 5.1 GA
- Lustre 2.13 and 2.12.4 coming soon
- Lustre 2.14 targeted for late summer 2020
- LWG http://wiki.opensfs.org/Lustre_Working_Group

Thank you

Open Scalable File Systems, Inc.
3855 SW 153rd Drive
Beaverton, OR 97006
Ph: 503-619-0561
Fax: 503-644-6708
admin@opensfs.org

www.opensfs.org

Lustre Features and Roadmap

- Andreas Dilger - Chief Technical Officer at Whamcloud, Inc.

Lustre 2.14 and Beyond

Andreas Dilger, Whamcloud

Upcoming Release Feature Highlights

- **2.13** feature complete, November, 2019
  - Persistent Client Cache (PCC) – store file data in client-local NVMe/NVRAM
  - LNet Multi-Rail Routing – extend MR to/through routers, handle mixed interfaces
  - DNE space balanced remote directory – improve load/space balance across MDTs
  - Layout OST Overstriping – allow multiple objects from one OST in a striped file
  - Self-Extending Layouts (SEL) – better handle OST out-of-space in the middle of a file
- **2.14** has several important features under active development
  - DNE directory auto-split – improve usability and performance with multiple MDTs
  - File Level Redundancy - Erasure Coding (EC) – efficiently store striped file redundancy
  - OST Pool Quotas – manage space on tiered storage targets using OST pools
- **2.15** plans continued functional and performance improvements
  - Metadata Writeback Cache (WBC) – low latency file operations in client RAM
  - Client-side data encryption – persistent encryption from client to disk

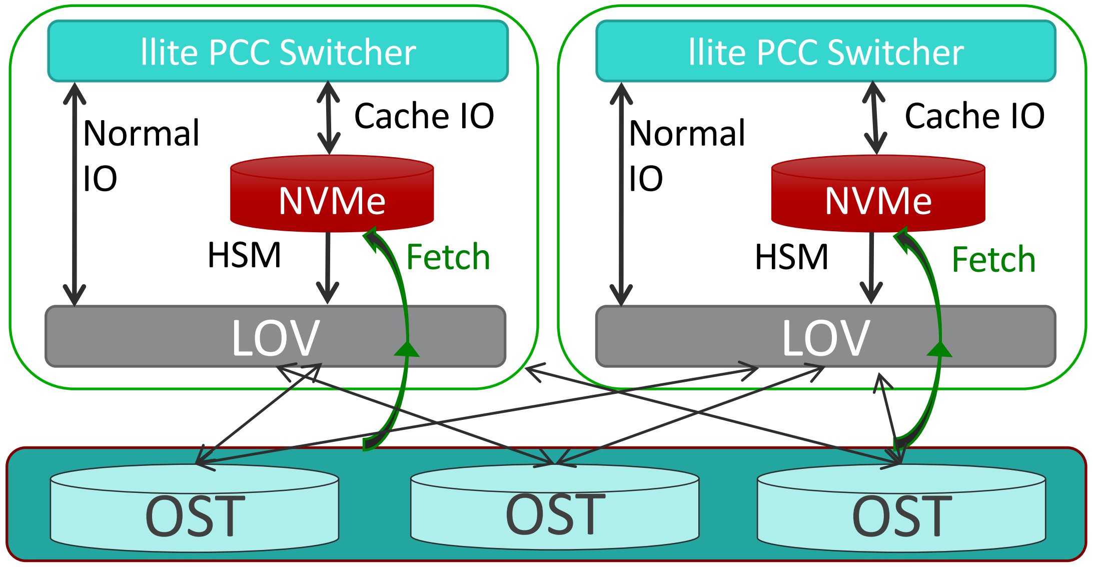
Persistent Client Cache (PCC) (LU-10092, DDN,WC) (2.13)

Whamcloud

- **Reduce latency**, improve small/unaligned IOPS, reduce network traffic
- PCC integrates Lustre with a persistent per-client **local cache storage**
  - **A local filesystem** (e.g. `ext4` or `ldiskfs`) is created on client device (SSD/NVMe/NVRAM)
  - **Data is local to client**, no global/visible namespace is provided by PCC
  - **HSM POSIX copytool** fetches whole files into PCC by user command, job script, or policy
  - New files created in PCC are *also* created on Lustre MDS
- Lustre uses local PCC data, or normal OST RPCs
  - Further file read/write access "directly" to cache file
  - No data/IOPS/attributes off client while file in PCC
  - 2.13 File migrated out of PCC via HSM upon remote access

2.14

- Separate **shared read** vs. **exclusive write** cache
- Integrate with DAX for NVRAM cache device
  - Use dedicated NVRAM filesystem (e.g. NOVA) for speed

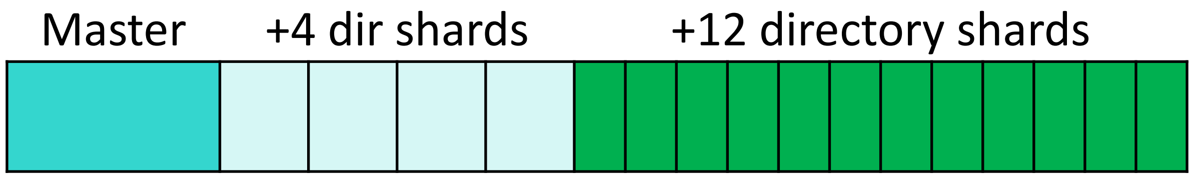


DNE Usability and Performance Improvements (2.13+)

- ▶ **Space balance new directories** on "best" MDT based on available inodes/space
  - Simplifies multiple MDTs without overhead of striping all directories, similar to OST balance
  - 2.12 Explicitly when creating a new directory with "`lfs mkdir -i -1`" (LU-10277, WC)

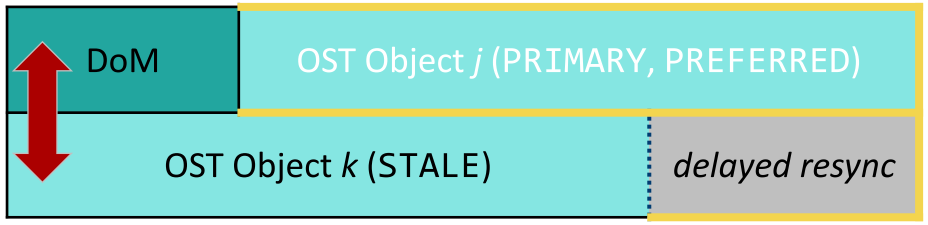
  - Transparently select "best" MDT for normal `mkdir()` based on parent policy (LU-10784, WC)
    - Set default policy on parent via "`lfs setdirstripe -i -1 dir`" (LU-11213, WC)
    - Most useful for root directory and top-level user directories
- 2.13 ▶ **Improved DNE file create performance** for clients (LU-11999, Uber)

- 2.14 ▶ **Automatic directory restriping** as directory size grows (LU-11025, WC)
  - Create one-stripe directory for low overhead, scale shards/capacity/performance with size
  - Add extra directory shards when master directory grows large enough (e.g. 10k entries)
  - Move existing dirents (not inodes) to new directory shards
  - New dirents and inodes created on new MDTs



Data-on-MDT (DoM) Improvements (WC) (2.13+)

- Convert write locks to read locks w/o cache flush (LU-10175)
- General usability and stability improvements
- FLR mirror/migrate DoM file (LU-11421)
  - Mirror DoM data to OST object
  - Migrate DoM data to/from OST object
  - No MDT-MDT mirroring yet



- Performance and functional improvements
  - 2.13 Target IO-500 `mdtest-hard-{write,read}` (3901-byte parallel file create in shared dir)

2.14
- Dynamic DoM component size by MDT free space (LU-12785)
- Merge data write with MDS_CLOSE RPC (LU-11428)
- Cross-file data prefetch via statahead (LU-10280)
- Allow MDT-only filesystem (LU-10995)

Improved Client Efficiency for AI/ML (2.13+)

- **Single thread DNE create** performance (LU-11999, Uber)
  - Reduce locking overhead/latency *for single-threaded* workloads (780/sec -> 2044/sec)
- **Parallel client readahead** performance (LU-8709, LU-12043, WC)
  - Improved *single-threaded readahead* (e.g. "dd") from 1.9GB/s -> 4.0GB/s
- **Overstriping OST objects** better large/fast OSTs on fewer clients (LU-9846, Cray, WC)
  - "`lfs setstripe -C|--overstripe-count` *`stripe_count`*" for multiple objects per OST
- **Improved small file handling** (IO-500 `mdtest-hard-{write,read}` performance)
  - Cache small files after create (LU-11623, LU-12325, LU-10948, ..., WC)
- **Improved strided readahead** (IO-500 `ior-hard-read` performance)
  - 2.13 Detect and handle page-unaligned strided reads (LU-12644, WC)
  - 2.14 Allow readahead to continue for slightly "imprecise" strides
- **Local client mount on OST/MDT** for data mover/resync (LU-10191, WC)
  - Beginning of optimization for local IO path to avoid RPC + data copy

Server Performance Improvements for Flash (2.12+)

- **Reduce server CPU** overhead to improve small flash IOPS (LU-11164, WC, DDN)
  - Reduced CPU usage translates directly to improved IOPS
- **Avoid page cache** on flash OSS (LU-11347, WC)
  - Avoids CPU/lock overhead/lock for page eviction
- **TRIM flash storage** on ldiskfs (LU-11355, DDN)
  - Release unused blocks of filesystem via `fstrim`

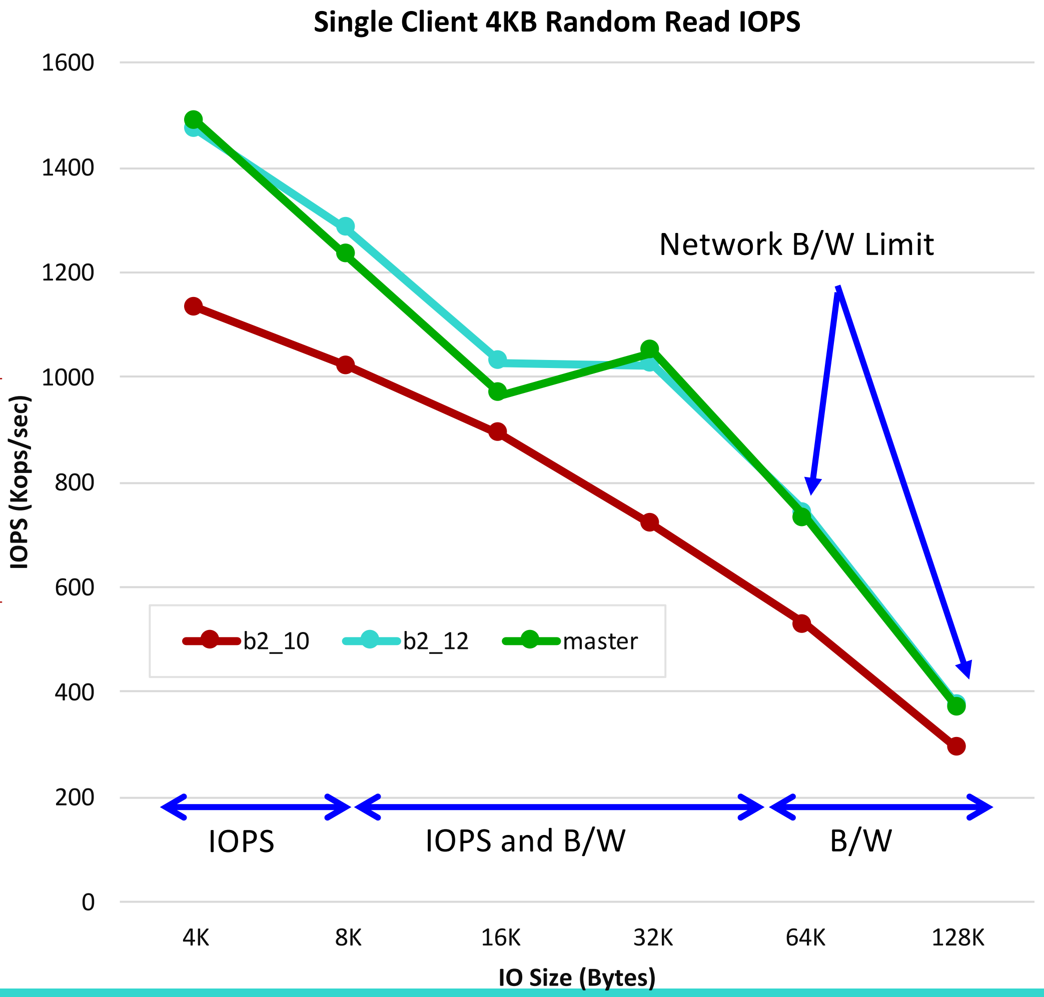
2.12

- **Self Extending Layouts** (LU-10070, Cray)
  - Avoids out-of-space in the middle of files
  - Good for PFL with smaller flash OSTs than disk OSTs

2.13

2.14

- Continued reductions of overhead and latency
  - Improve small, unaligned and interleaved writes
  - Lockless unaligned IO submission from clients
  - Writeback cache with IO merging on servers
  - Improve IO-500 `ior-hard-write`



File Level Redundancy (FLR) Enhancements (WC) (2.13+)

Whamcloud

- **Lustre-level mirroring for files**, configured arbitrarily per file/directory
- **Mirror NOSYNC flag** + timestamp to allow *file version/snapshot* (LU-11400)
- Mount client directly on OSS without impacting recovery (LU-12722)

2.13
- "`lfs mirror resync/delete --pool`" to simplify tiering (LU-11022)

2.14
- **Erasure coding** adds redundancy without 2x/3x mirror overhead (LU-10911)
  - Add erasure coding to new/old striped files *after* write done
  - Leverage CPU-optimized EC code (Intel ISA-L) for best client performance
  - For striped files - add N parity per M data *stripes* (e.g. 16d+3p)
  - **Fixed RAID-4 parity** layout *per file,* **declustered Parity** across files to avoid IO bottlenecks

2.15
- HSM in composite layout (LU-10606)
  - Allow multiple archives per file (S3, tape, ...)
  - Allow partial file restore from archive

TBD
- File version/reflink within namespace?
  - Access like VAX/VMS using "`filename,1`"?

Replica 0	Flash Object *j* (PRIMARY, PREFERRED)	
Replica 1	HDD Object *k* (STALE)	*delayed resync*
Replica 2	HSM S3 Archive	

Miscellaneous Improvements (2.13/2.14)

Whamcloud

- **Overstriping** allows multiple file stripes per OST (LU-9846, Cray, WC)
  - Useful for shared-file workloads or very large OSTs
- **`lfs find`** integration with Lazy Size-on-MDT (LU-11367, WC)

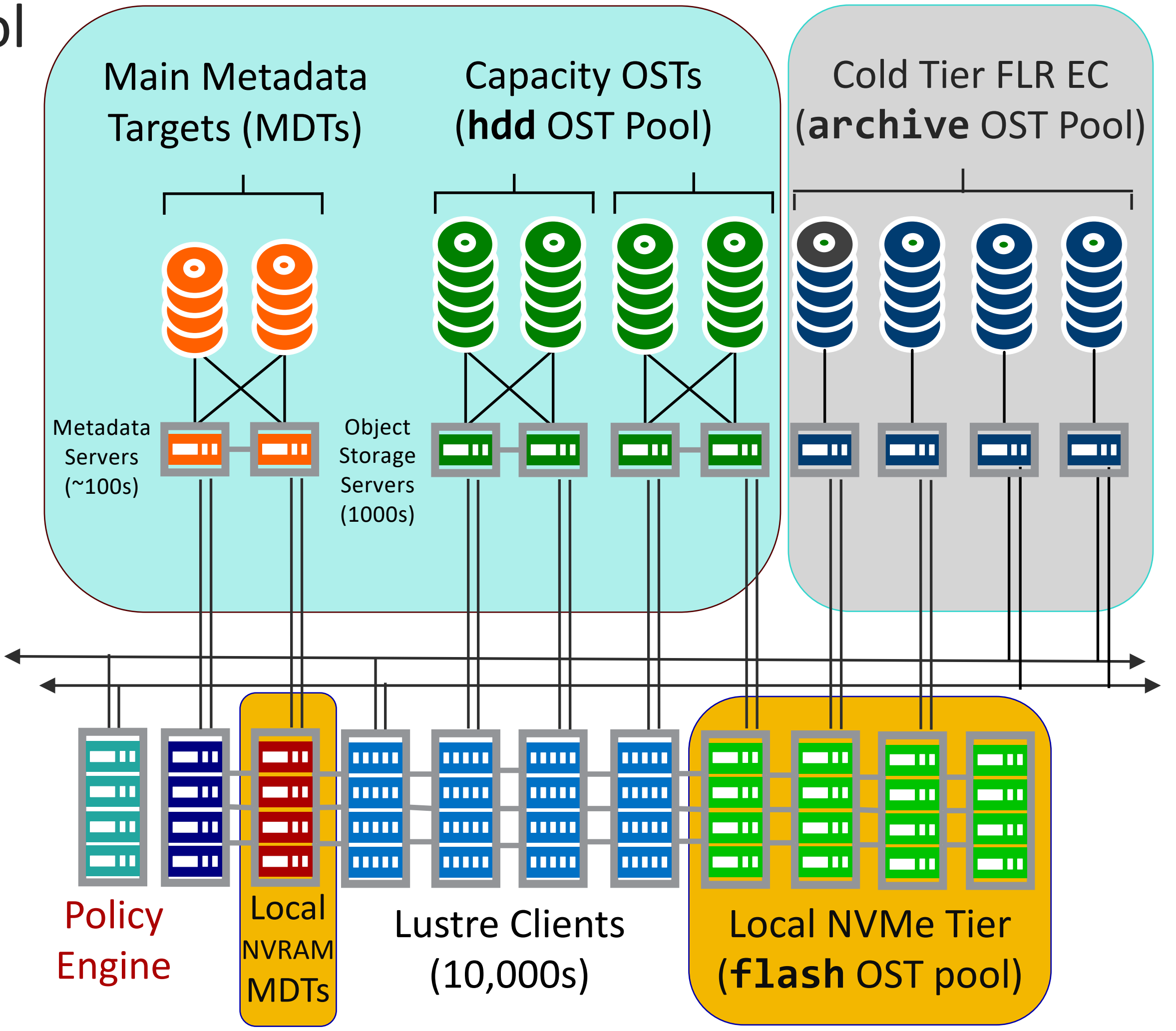
2.13
- **Upstream kernel client cleanups** still in active development/merge (ORNL, SuSE)

2.14
- **Pool Selection Policy** by filename extension, NID, UID/GID (LU-11234, WC)
- **Dynamic OSS page cache** based on RPC IO size (LU-12071, WC)
- **`fallocate()`** for file preallocation (ldiskfs only), hole punch (LU-3606, WC, User)
- **`statx()`** for lightweight attribute fetching (LU-10934, WC)
- **`O_TMPFILE`** for creating temporary files outside namespace (LU-9512)

Pool Quotas for OSTs (LU-11023, Cray) (2.14+)

- ▶ Account/limit space for OSTs in a specific pool
  - Control usage of small flash OSTs in tiered config
- ▶ Use existing Lustre quota infrastructure
  - OST already tracks space per UID/GID/ProjID
  - Pool usage based on sum of current OSTs in pool
- ▶ Add pool quota limits per UID/GID/ProjID
  - No extra accounting on the OSTs
  - 2.14 Only new aggregation/reporting by MDS

- TBD ▶ Integrate OST selection with quota usage
  - Avoid OSTs with no/little quota available
- ▶ Add MDT pools after OST pools complete
  - Manage/balance DoM MDT space usage
  - Handle MDT storage classes (e.g. NVRAM vs. NAND)


Main Metadata Targets (MDTs)
Capacity OSTs (hdd OST Pool)
Cold Tier FLR EC (archive OST Pool)
Metadata Servers (~100s)
Object Storage Servers (1000s)
Policy Engine
Local NVRAM MDTs
Lustre Clients (10,000s)
Local NVMe Tier (flash OST pool)

Client-Side Data Encryption at Rest (LU-12755, WC) (2.15)

Whamcloud

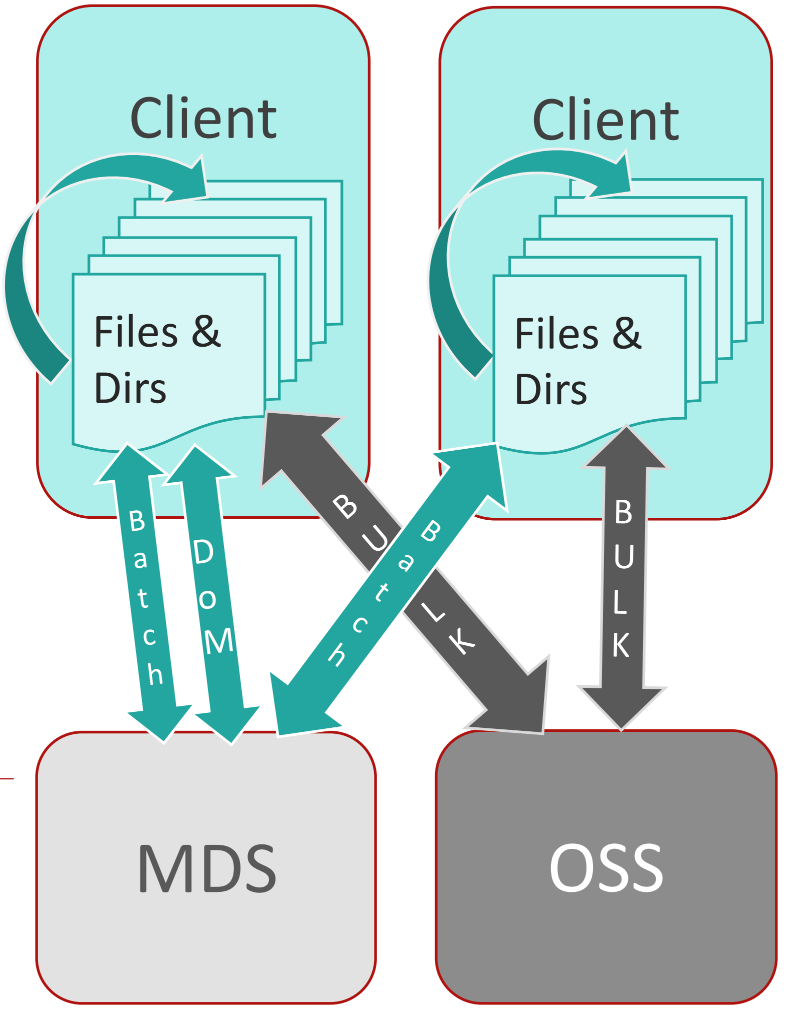
- Protect from storage theft/mistakes network/admin snooping
- **Encryption on Lustre client** down to storage
  - Data is **encrypted before sending** to servers
  - Data is **decrypted after receiving** from servers
  - Servers/storage only see encrypted data/filenames
  - Only client nodes need access to user encryption keys
  - **Transparent to backend** filesystem/storage (ldiskfs/ZFS)
  - Utilize larger client CPU/accelerator capacity
- **Ext4/f2fs `fscrypt` library/tools base** (don't invent it!)
  - Tunable encryption setting/key(s) per directory tree
  - **Per-file encryption key(s)**, itself encrypted by user key
    - **Fast and secure deletion** of file once per-file key is erased
  - Filenames encrypted in MDT directory entries

Metadata Writeback Cache (WBC) (LU-10983, WC) (2.15+)

- ▶ Create new dirs/files **without RPCs in client RAM** (or local NVMe)
  - Lock new directory exclusively at mkdir time
  - Cache new files/dirs/data only in RAM/local NVMe until cache flush
- ▶ **No RPC round-trips** for file modifications in new directory
- ▶ **Files globally visible on flush** to MDS, *normal afterward*
  - Flush top directory to MDS upon other client access, lock conflict
    - o Create top-level entries, exclusively lock new subdirs, release parent
    - o Repeat as needed for portion of namespace being accessed remotely
  - Flush rest of tree in background to MDS/OSS by age or size limits
- ▶ Basic WBC prototype developed to test concept
  - No cache/quota/space limits, no background flushing, no batching, ...

2.15 • 10-20x *single-client* speedup in early testing (`untar`, `make`, ...)

2.16 ▶ **Aggregate operations to server** to improve performance
  - Batch operations in one RCP to reduce network traffic/handling
  - Batch operations to **filesystem to reduce disk IOPS**



Whamcloud

Thank You!
Questions?

Ongoing ldiskfs Improvements (2.13+)

- Major `ldiskfs` features merged into upstream `ext4/e2fsprogs` (WC, Cray)
  - Large xattrs (up to 64KB/xattr) stored in separate inode (**`ea_inode`**)
  - Large directories over 10M entries/2GB (**`large_dir`**)
  - 2.13 Project quota accounting/enforcement (**`project`**)

- 2.14 One more Lustre-specific feature remains to be merged to `ext4/e2fsprogs`
  - Extended data in directory (**`dirdata`**) - needs unit test interface before merge
- Existing `ext4` features available that could be used by Lustre on `ldiskfs`
  - Efficient block allocation for large OSTs (**`bigalloc`**)
  - Tiny files (1-600/3800ish bytes) stored directly in the MDT 1KB/4KB inode (**`inline_data`**)

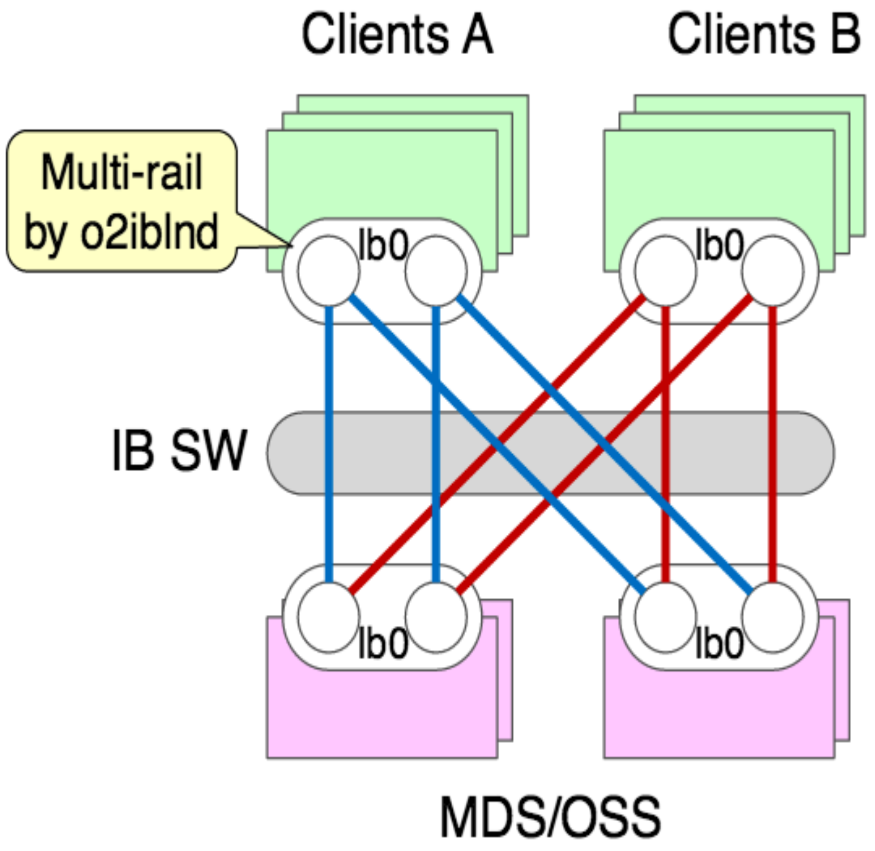
  - 2.15 Metadata integrity checksums (**`metadata_csum`**)
- New `ext4` features currently under development
  - **Data Verity** – Merkle tree of data checksums stored persistently on *read-only* files
  - **Directory shrink** – reduce directory block allocation as files deleted

LNet Multi-Rail Selection Policy (2.13+)

- Multi-Rail routing (LU-11299, WC, Cray)
  - Extend LNet Multi-Rail to router nodes
  - 2.13 Improve handling of mixed MR/single networks

2.14

- User Defined Selection Policy (LU-9121, WC)
  - Fine grained control of interface selection
    - TCP vs. IB networks, primary vs. backup
  - Optimize RAM/CPU/PCI data transfers
  - Useful for large NUMA machines



Community Events

Gaël Delbary - CEA

Kristy Kallback-Rose - NERSC

FROM RESEARCH TO INDUSTRY

LAD 2020

November 19th, 2019

Gaël DELBARY

LAD 2020

- **Thanks to attendees, speakers who came to the last LAD**

- **Thanks to fill up the survey (> 50% of attendees)**

- **Next LAD:**
  - As requested in the survey, we will try to open sooner the registration, like the call for papers. Dates TBD
  - Period: From 09/07/2020 to 09/30/2020. 3 days conference.
  - Venue: Maybe another city that Paris, perhaps Bordeaux, TBD

Questions?

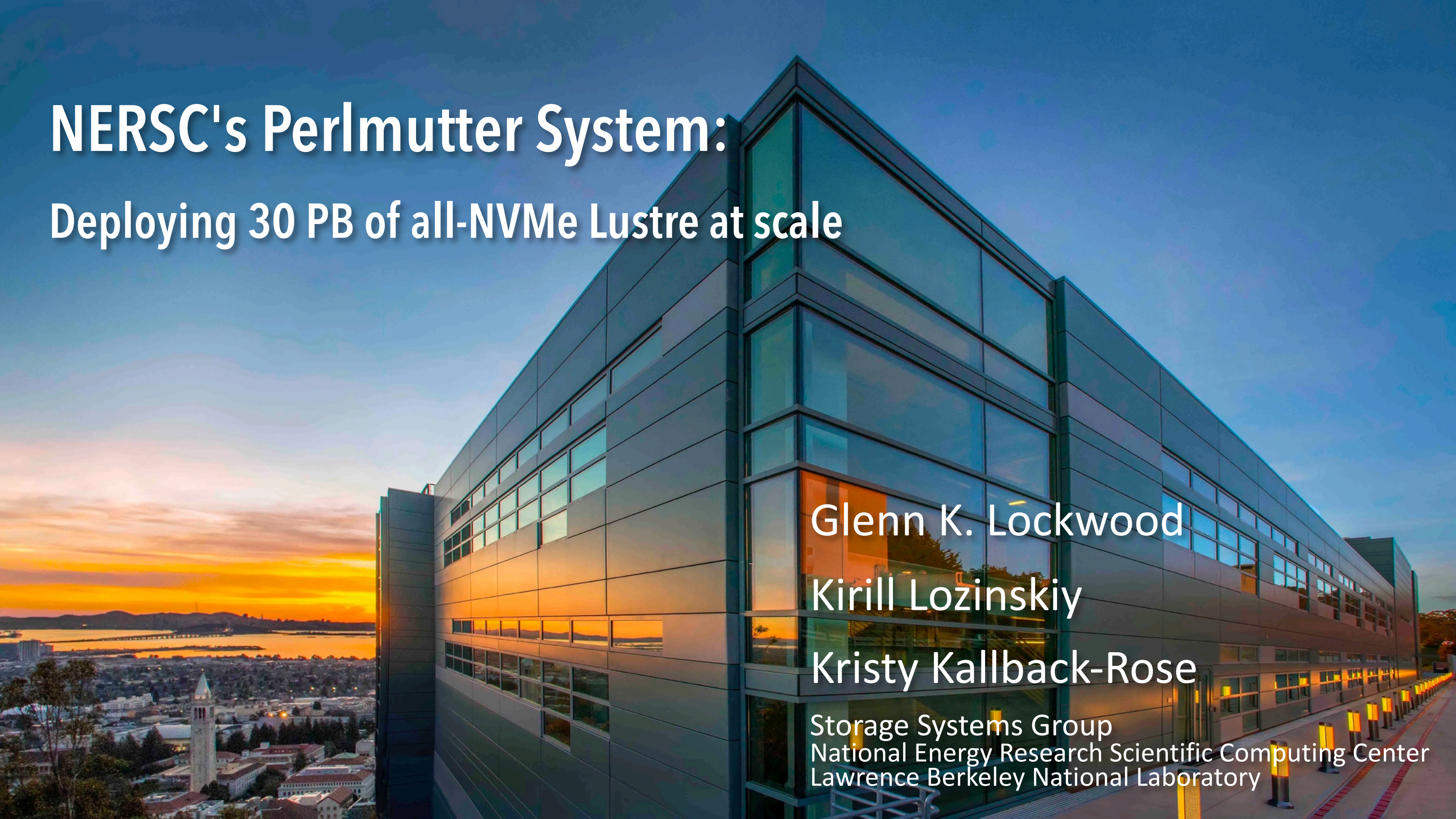
SAVE THE DATE

LUSTRE USER GROUP (LUG) 2020

JUNE 9 - 12, 2020

BERKELEY, CALIFORNIA

Hosted by LBNL/NERSC, UC Berkeley Research IT, and OpenSFS



NERSC's Perlmutter System:

Deploying 30 PB of all-NVMe Lustre at scale

Glenn K. Lockwood

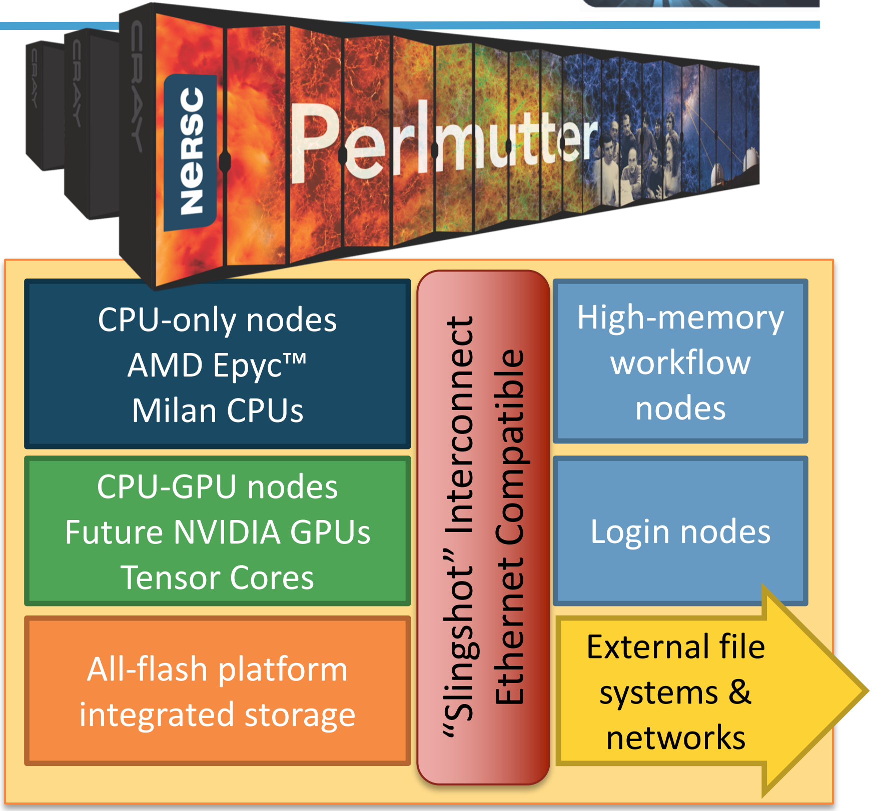
Kirill Lozinskiy

Kristy Kallback-Rose

Storage Systems Group
National Energy Research Scientific Computing Center
Lawrence Berkeley National Laboratory

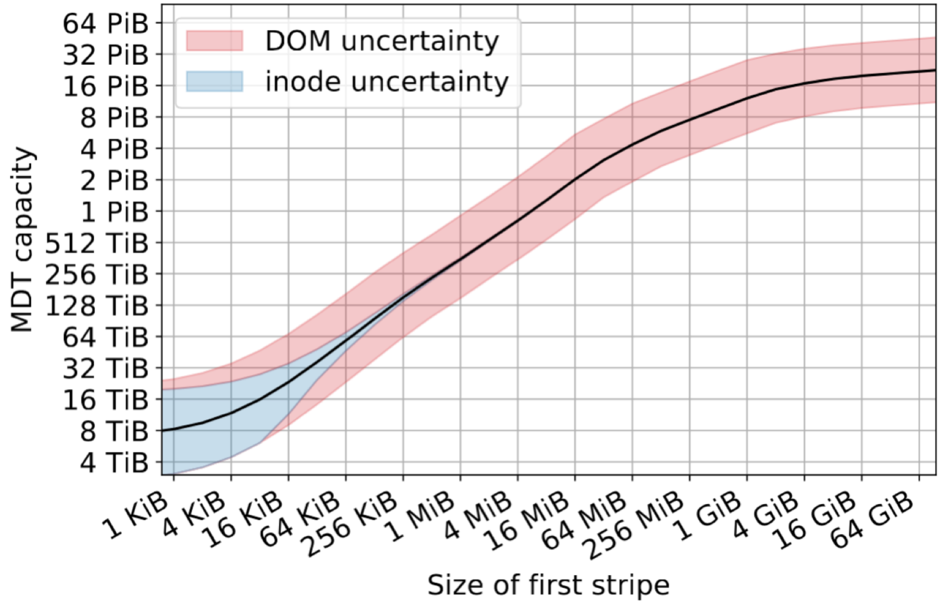
NERSC's 2020 System: Perlmutter

- **Cray Shasta**
  - 3x to 4x capability of Cori
  - GPU-accelerated and CPU-only nodes
  - Cray Slingshot interconnect
- **Single-tier, all-flash Lustre**
  - Cray E1000F-based system
  - 30 PB usable capacity
  - ≥ 4.0 TB/s sustained bandwidth
  - ≥ 7,000,000 IOPS (read *and* write)
  - ≥ 3,200,000 file creates/sec



NERSC-Cray Collaboration

- How much capacity? How much endurance?
- DNE, DOM configuration?
- ldiskfs or ZFS? Which meets performance spec?



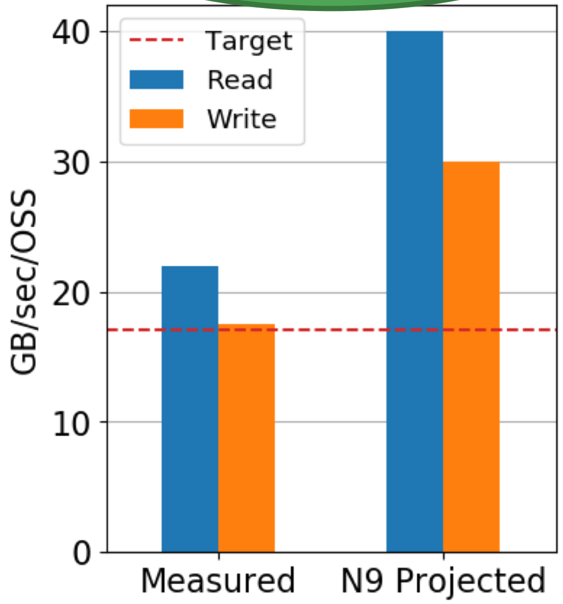
Above: optimal MDT capacity for different DOM component sizes. See Lockwood et al.,"A Quantitative Approach to Architecting All-Flash Lustre File Systems" in High Performance Computing, Chapter 16. https://doi.org/10.1007/978-3-030-34356-9_16

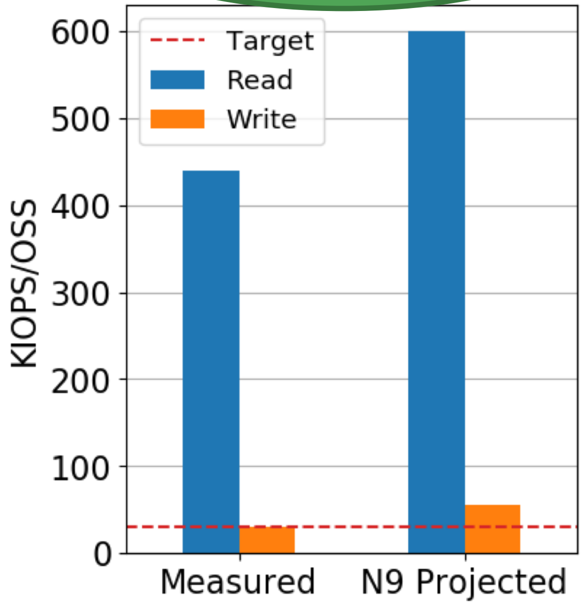
Cray's work on all-NVMe Performance - IOR

Cray's Test OSS	Perlmutter (Cray E1000F)
24c AMD Rome	≥24c AMD Rome
2x EDR (100G)	≥2x Slingshot (200G)
PCIe Gen3 SSDs	PCIe Gen4 SSDs
Cray Lustre (community + forward-ported patches)	
Declustered RAID6 8+2+1 (GridRAID)	

22 GB/s/OSS read
18 GB/s/OSS write

440 KIOPS/OSS read
30 KIOPS/OSS write





NERSC

Thank you—and we're hiring!

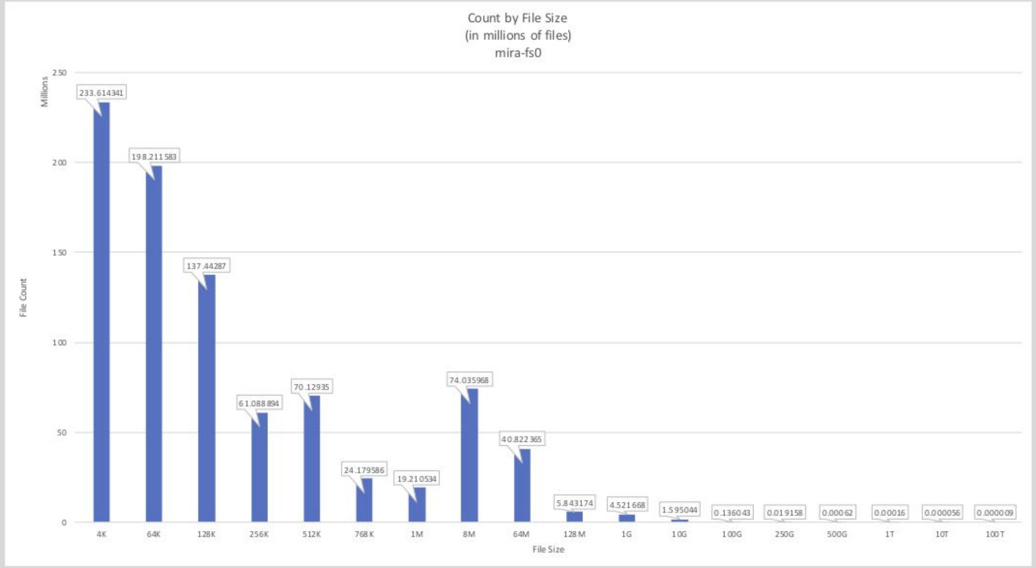
Ask any NERSC staff for more info

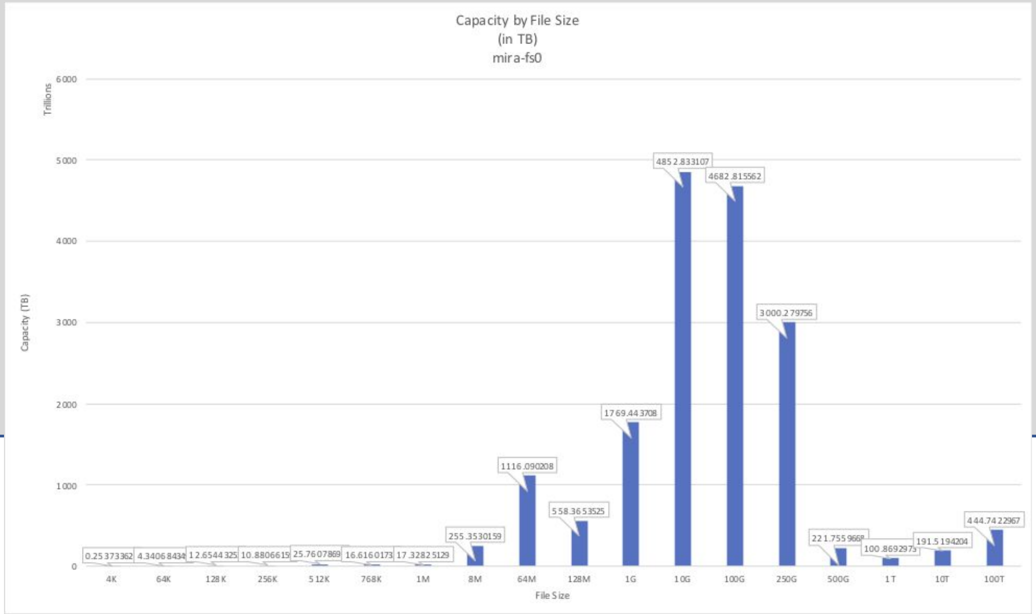
HPC

Argonne Leadership Computing Facility
Data-on-Metadata Projections

- ALCF acquiring new Global (or Centerwide) File System
  - Cray E-1000 - Lustre
  - 150 PB @ 1000 GB/s

- ALCF has existing large scale file system in production since 2012
  - DDN 12k - GPFS
  - 20 PB @ 200 GB/s





European Open File System

ALCF
DoM Projections

scaling	7.5x
total files	870,420,000
files >64M	52,820,845
files >8M	126,820,845
md cap	12.26
target NVM	320.00
DoM available	**307.74**

DoM Size	Naïve Est.	File Count Est.
64K	389.11 365.50 332.41	**291.21** **267.59** **234.51**
128K	778.22 730.99 664.83	487.29 440.06 373.90
512K	3112.86 2923.96 2659.32	1186.14 997.24 732.60
1024K	6225.73 5847.92 5318.64	1751.45 1373.64 844.35

DoM Size	Savings files ≥64M	Savings files ≥8M
64K	(8%) 23.61	(19%) 56.69
128K	(10%) 47.23	(23%) 113.39
512K	(16%) 188.90	(38%) 453.55
1024K	(22%) 377.80	(52%) 907.09

BP migration to PFL

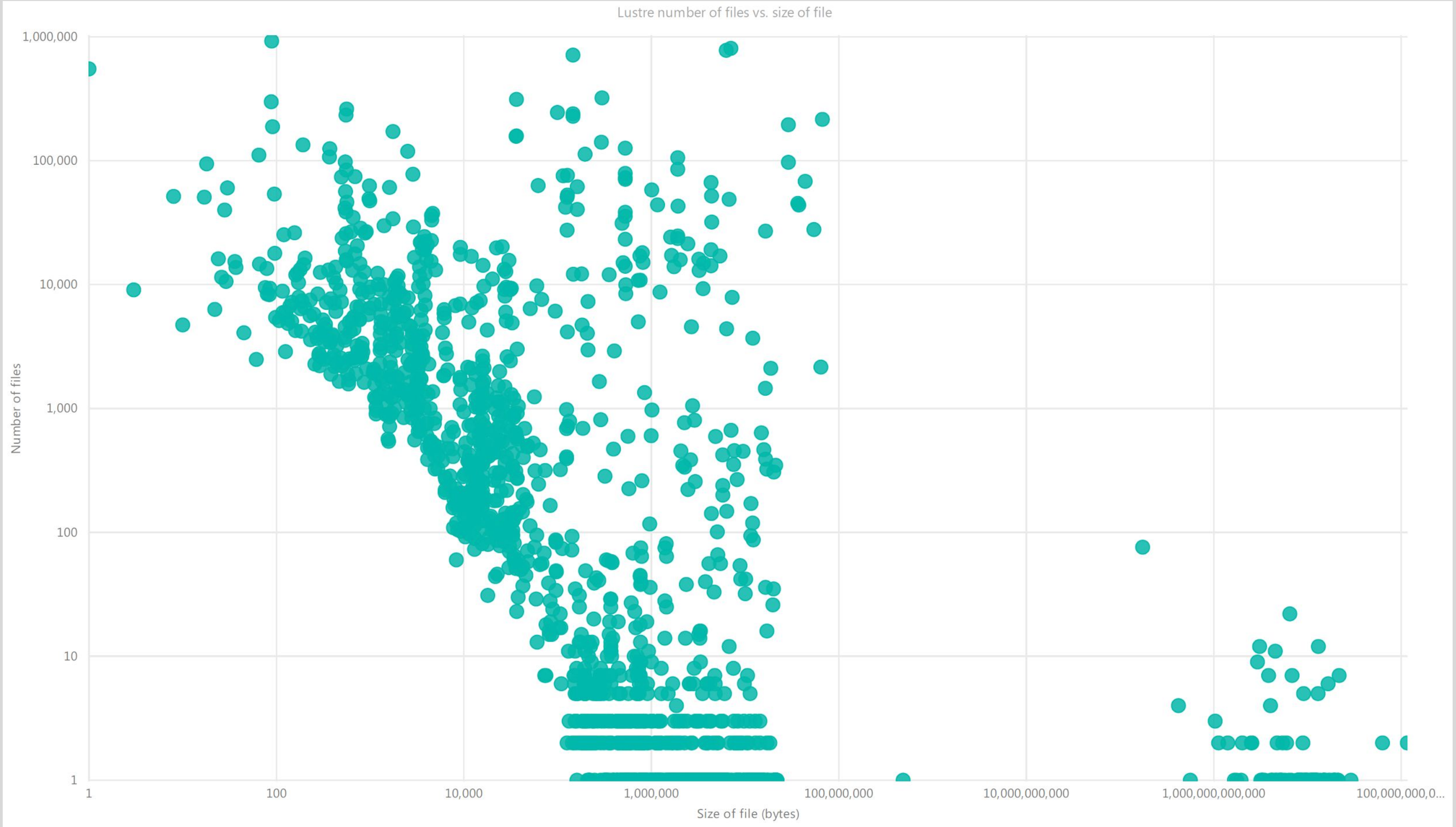
- Upgraded all 4 file systems in place to Lustre 2.10.5 in Oct. 2018
- Recursively changed default directory striping to PFL in Dec. 2018
- All new files have been PFL ever since

BP realized benefits of PFL

- No more OST imbalance and full OSTs
- No more jobs hammering a single OST
- Every job better utilizing the file system

BP interest in DoM

- 79.7% of our files are <= 1 MB


Lustre number of files vs. size of file
Number of files
1,000,000
100,000
10,000
1,000
100
10
1
1
100
10,000
1,000,000
100,000,000
10,000,000,000
1,000,000,000,000
100,000,000,0...
Size of file (bytes)

BP interest in DoM

- Recently acquired new file system with 30 TB usable NVMe for MDT
- Plan on using DoM + PFL
- Expecting good improvements for small files

Stanford: DoM/PFL on Sherlock

DoM/PFL in production since February 2019 with ~1,500 clients

- DoM
    - **Performance problems with shared files** with multiple writers, because locking is made on the whole DoM region (no extent locking; see **LU-10664**)
    - Double sentence: **impact on filesystem metadata performance**
- PFL
    - **Multiple benefits in dynamic file striping with PFL:** automatically improve performance for large files and avoid OST usage unbalance
    - Appending to a PFL file will cause **all** layout components to be instantiated; **FIXED in Lustre 2.13** but not backported to 2.12 yet; see **LU-9341**

AI & Cloud

AI & Cloud Discussion

- End Users
  - Who is using Lustre in the cloud?
  - For what workloads?
  - Why cloud vs on-prem?
- Service Providers
  - Are you using Lustre for your AI customers?
  - What configuration are you using?

Additional Discussion Topics

- What conferences should we target to generate interest in Lustre?
- How to improve user experience?
- How to simplify layout specifications?

- What are we missing?

Mark Your Calendars: Community Events

- LUG'20
  - June 9 - 12 - Berkeley, CA
- ISC'20 Lustre BOF
  - June 21 - 25 - Frankfurt, Germany
- LAD'20
  - September - Paris, France

SAVE THE DATE

LUSTRE USER GROUP (LUG) 2020

JUNE 9 - 12, 2020

BERKELEY, CALIFORNIA