Whamcloud

Lustre 2.15 and Beyond

Andreas Dilger, Lustre Principal Architect

UF UNIVERSITY of FLORIDA

ddn

LUG 2021

Planned Feature Release Highlights

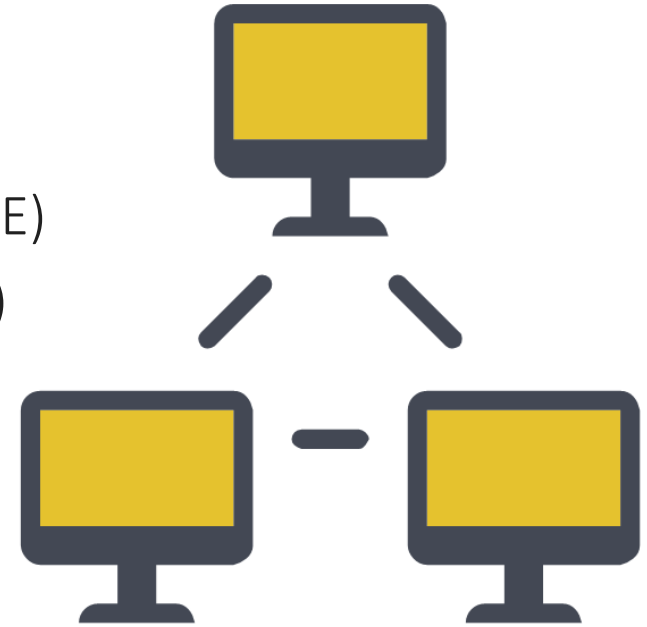
Whamcloud

- **2.15** feature development and landing already in progress
  - **LNet User Defined Selection Policy** – rules for selecting interface (WC)
  - **LNet IPv6 addressing** – allow 160-bit NIDs, more flexible server configuration (SuSE)
  - **Client-side *filename* encryption** – persistent encryption filenames from client to disk (WC)
- **2.16** plans continued functional and performance improvements
  - **File Level Redundancy - Erasure Coding (EC)** – efficiently store file redundancy (WC, ORNL)
  - **Metadata Writeback Cache (WBC)** – low latency file operations in client RAM (WC)
- **2.17** feature proposals in early discussion stages
  - **Client-side data compression** – leverage client CPUs to reduce network/storage usage
  - **File Level Redundancy - Immediate Write** – write to mirrors directly from client

LNet Improvements (2.15)

Whamcloud

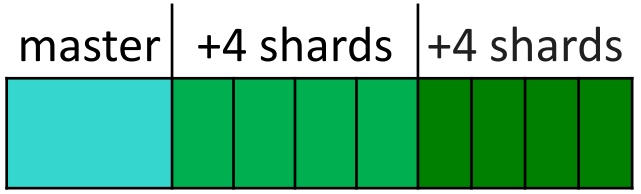
- LNet User Defined Selection Policy (UDSP) (LU-9121, WC)
  - Allow policies for local/remote interface prioritization by NID
  - e.g. primary IB with TCP backup, select "best" router NID for client/server
- Multiple TCP sockets for 100GigE+ performance (LU-12815, WC)
  - Allow `conns_per_peer=N` for `socklnd` (4.1GB/s->9.5GB/s on 100GbE)
  - Auto-configure based on interface speed (e.g. 40Gbps=2, 100Gbps=5)
- Simplified/dynamic server node addressing (LU-14668, WC)
  - Detect added/changed server interfaces automatically (LU-10360)
  - Reduce (eventually eliminate?) static NIDs in config logs
- IPv6 NID support (LU-10391, SuSE)
  - NIDs up to 160 bits (16-bit type, 16-bit network number, 128-bit address)
  - Interoperable with existing current LNDs when possible



MDT DNE Usability Improvements (WC 2.14+)

Whamcloud

- **DNE automatic directory split** above threshold (LU-11025)
- **DoM component shrink** if MDT free space low (LU-12785)
  - Reduce or eliminate DoM component to avoid ENOSPC on MDT

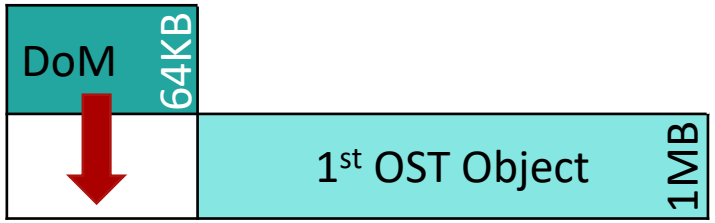


- **Optimized rename operations** for common cases (e.g. rsync, mpiFileUtils, vim, emacs, etc.)
  - Avoid needless remote entry (`.filenameXXXXXX`, `filename.XXXXXXXX`, `*.bak`, `*.sav`, `*.orig`, `*~`)

2.14
  - Avoid remote symlink when renamed across MDTs (LU-11631)

2.15
  - Parallel directory locking for file/directory rename within same directory (LU-12125)
- **DNE space balance `mkdir()`** for filesystems (LU-13439, LU-13440)
  - `lfs setdirstripe -D -c 1 -i -1 [--max-depth[-rr] <levels>] <directory>`
  - Round-robin and/or space balanced remote subdirectories, limited depth
  - Stay on same parent MDT if space is already balanced

2.16
- DoM->OST migration optimization (LU-13612)
  - Avoid full-file copy when removing DoM component



Encryption – Data at Rest (2.14/2.15, DDN)

Whamcloud

- Protect from storage theft/loss, network/user snooping
- Encryption done at Lustre client level (LU-12275)
  - Clients keep unencrypted data in memory
  - Data encrypted when sent to/stored on/read from servers
  - Only users/client keyring have access to encryption keys
- Use `fscrypt` library (ext4/f2fs/…) as basis (don't invent it!)

2.14
  - **Per directory tree** encryption with (optionally) multiple user key(s)

2.15
- Filenames encrypted in directory entries (LU-13717)
  - Filenames base64 encoded, can still list directories, unlink files/dirs if needed
- Migrate/mirror of encrypted files without key (LU-14667)
  - Avoid pinning files to storage tier

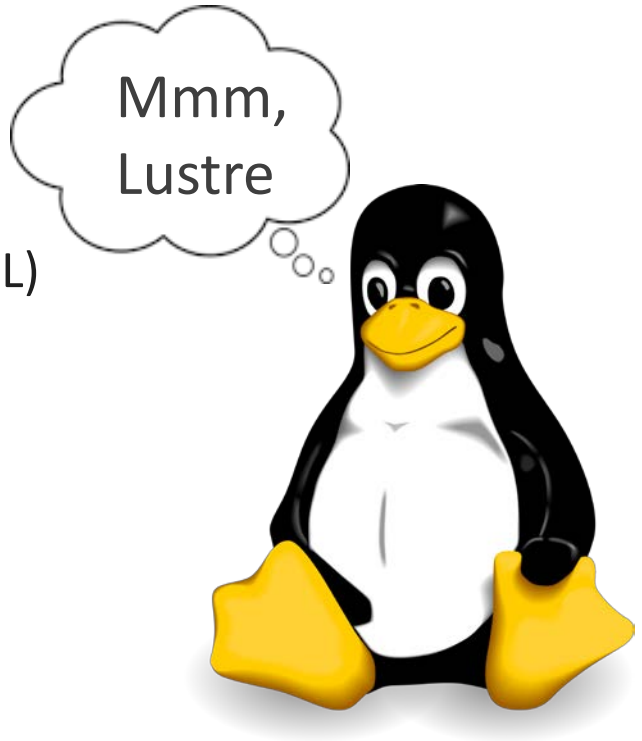
Client Kernel Code Improvements (ORNL, SuSE, WC 2.14+)

Whamcloud

- `fallocate()` for preallocation (ldiskfs) and hole punch (LU-3606, LU-14160, WC, AEON)
- `SEEK_HOLE/SEEK_DATA` to efficiently handle sparse files (LU-10810, WC)

2.14
- `statfs()` on directory with projid returns project quota limits (LU-9555, WC)

2.15
- Automatic open lock caching on client (LU-10948, WC, ORNL)
- Handle large ACLs up to 8k entries (LU-14430, WC)



- Fully in sync with (old) upstream kernel client changes (LU-12511, SuSE, ORNL)
- Updates to build with 5.10 kernel (LU-14195, SuSE, ORNL)
- Updates to build with 5.12 kernel (LU-14651, SuSE, ORNL)
- Ongoing cleanups and code simplification (LU-many, SuSE, ORNL)
- IPv6 NID support (LU-10391, SuSE)

2.16
- `o2iblnd` cleanups for in-kernel OFED (LU-8874)
- Upstream ldiskfs patches (LU-6220)

Backend OSD Improvements

(2.14+)

Whamcloud

- **Fix mount time for 1PB ldiskfs OSTs**, 10min->**20s** (LU-12988, LU-13241, WC, HPE)
- **Parallel e2fsck** speedup for pass1 (inodes) and pass5 (bitmaps) (LU-8465, WC)
- ZFS 2.0 support (LU-13946, LLNL, many others)

2.14
  - Persistent L2ARC, Special VDEVs (Metadata Allocation Class), …

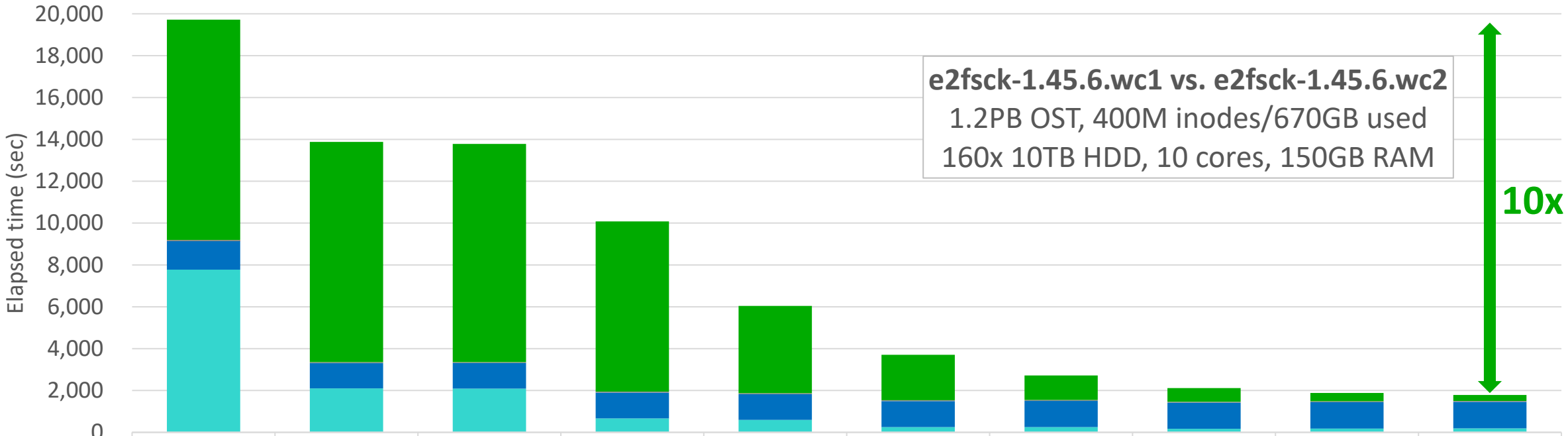
2.15
- ZFS 2.1 dRAID VDEVs (LLNL, HPE, Intel)
- Improved ldiskfs mballoc efficiency for large/full filesystems (LU-14438, Google, WC)
  - Global free space tracking to avoid scanning millions of block groups
  - O(1) lookup of power-of-two free space, O(logN) lookup of other sizes

2.16
- OST object directory scalability (LU-11912, WC)
  - Group objects by age to limit directory size and improve insertion/removal efficiency
- Parallel e2fsck speedup for pass2 (directory scanning) (LU-14679)
  - Now the slowest part of e2fsck (was 7% of runtime, now 70% of total runtime)
- Enable ldiskfs metadata checksum feature (LU-13650)

Parallel e2fsck Performance at Scale (LU-8465) (2.14ish)

Whamcloud

- Multi-threaded pass1 inode/block scan, and parallel pass5 bitmap loading



	original	thr=1	thr=2	thr=4	thr=8	thr=16	thr=32	thr=64	thr=128	thr=256	
pass5	10,523	10,524	10,427	8,133	4,166	2,168	1,169	654	399	286	36x
pass4	55	50	49	52	52	47	44	43	43	47	
pass3	0	0	0	0	0	0	0	0	0	0	
pass2	1,363	1,206	1,217	1,226	1,226	1,239	1,255	1,248	1,261	1,265	70%
pass1	7,777	2,101	2,092	669	597	254	250	171	181	191	40x

7%

Improved Single Client Performance (2.14+)

Whamcloud

- **Improve parallel client readahead** (LU-12043, LU-13386, LU-13412, WC)
  - Parallel/unaligned readahead for single user thread (e.g. "**dd**") from 1.9GB/s->**4.0GB/s**
- **Async Direct IO** and **`io_uring`** on Linux 5.1+ (LU-4198, LU-13801, WC, Uber)
  - Improved 4KB random IO via **`libaio`**, write 100k->**266k IOPS**; read 80k->**610k IOPS**
  - Non-POSIX async interface for IO and (evolving) metadata operations

2.14

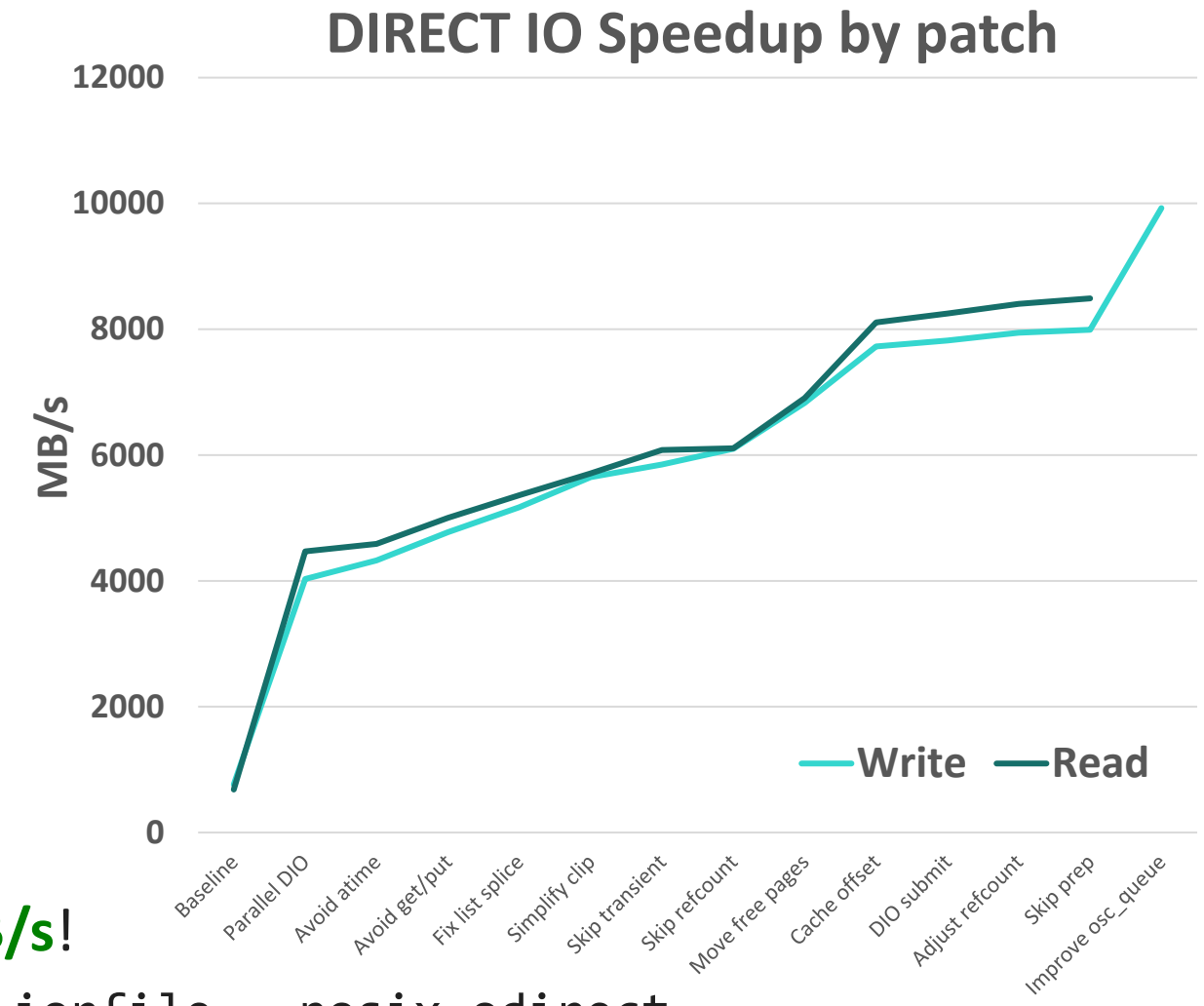
2.15

- **Improved mmap readahead** chunk detection (LU-13669, WC)
  - Reduced **pagefault latency**, avg 512usec->**52usec**, max 37msec->**6msec**
- **Parallel/large DIO/AIO performance** (LU-13798, LU-13799, HPE, WC)
  - Improve large single-thread **`read()`**/**`write()`**, reduced overhead (700MB/s->**10GB/s**!!!)
- **Improved NID->CPT binding** (LU-14676, WC)
  - Better distribution of RPCs from a single client across server CPU cores

Faster O_DIRECT (LU-13798, LU-13799) (HPE, WC 2.15)

Whamcloud

- Prototype patch series by Patrick Farrell
  https://review.whamcloud.com/39436...
- Parallel large single thread O_DIRECT IO
  - 64MB read/write ~700MB/s->**4GB/s**
- Reduce O_DIRECT submission overhead
  - No per-page inode timestamp updates
  - Skip unnecessary page/request refcount
  - Improve page list handling
  - Cache page offset calculation
  - Improve DIO space accounting
- Further DIO speedup from 4GB/s->**10GB/s**!




```
mpirun -np 1 IOR -wr -t64M -b64G -o iorfile --posix.odirect
```

FLR Erasure Coded Files (LU-10911) (WC, ORNL 2.16)

Whamcloud

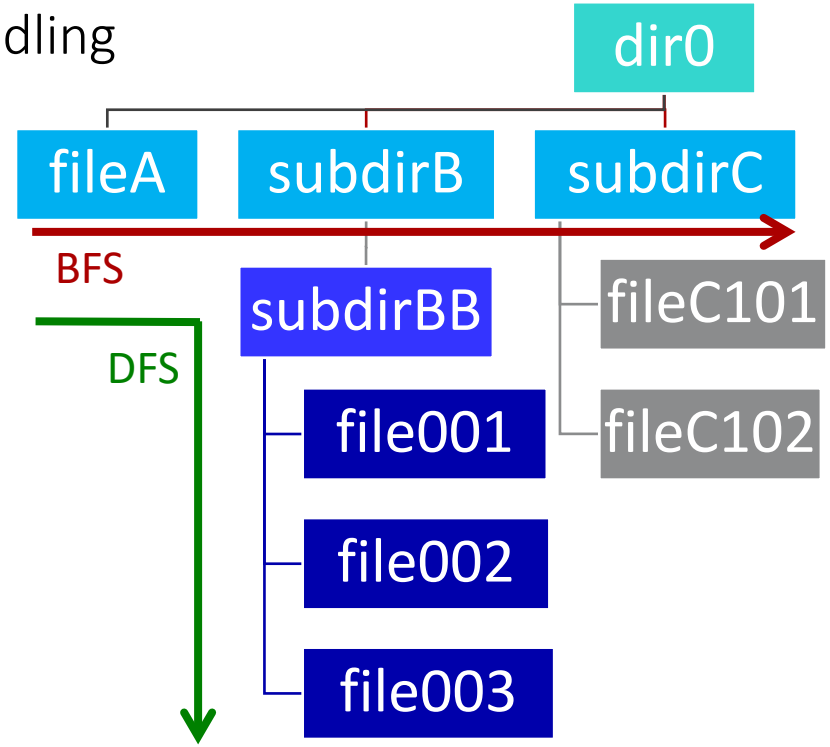
- Erasure coding adds redundancy without 2x/3x mirror overhead
- Add erasure coding to new/old striped files *after* write done
  - Use delayed/immediate mirroring for files being actively modified
  - Leverage CPU-optimized EC code (Intel ISA-L) for best performance
- For striped files - add N parity per M data *stripes* (e.g. 16d+3p)
  - Fixed RAID-4 parity layout *per file*, but declustered *across files*
  - Parity declustering avoids IO bottlenecks, CPU overhead of too many parities
    - e.g. split 128-stripe file into 8x (16 data + 3 parity) with 24 parity stripes

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	...

Batched Cross-Directory Statahead (WC 2.15)

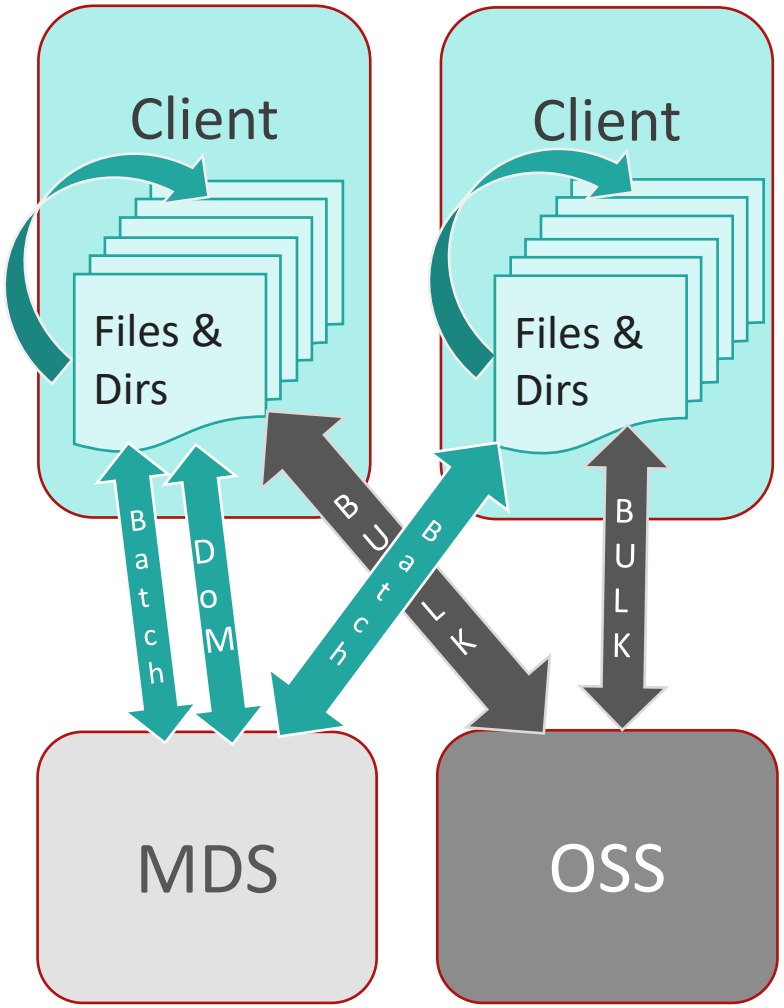
Whamcloud

- **Batched RPC operations** for efficient multi-object RPC (LU-13045)
  - Allow multiple getattr and update requests packed into a single MDS RPC
  - More efficient on the network and for 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)
  - Existing statahead only detects `readdir()`-ordered `stat()`
  - Detect pattern for alphanumeric ordered traversal + `stat()`
  - Detect breadth-first (BFS) depth-first (DFS) directory traversal
  - Direct statahead to next file/subdirectory based on pattern



Metadata Writeback Cache (WBC) (LU-10983) (WC 2.16+)

- 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
- **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
- WBC prototype developed to test concept
  - 10-20x *single-client* speedup in early testing (`untar`, `make`, …)
- Productization of WBC code well underway
  - Some complexity handling partially-cached directories
  - Need to integrate space usage with quota/grant



Whamcloud

Thank You!
Questions?