Exploring Lustre Overstriping For Shared File Performance on Disk and Flash

Michael Moore, Cray
Patrick Farrell, Whamcloud

✉ mmoore@cray.com
✉ pfarrell@whamcloud.com

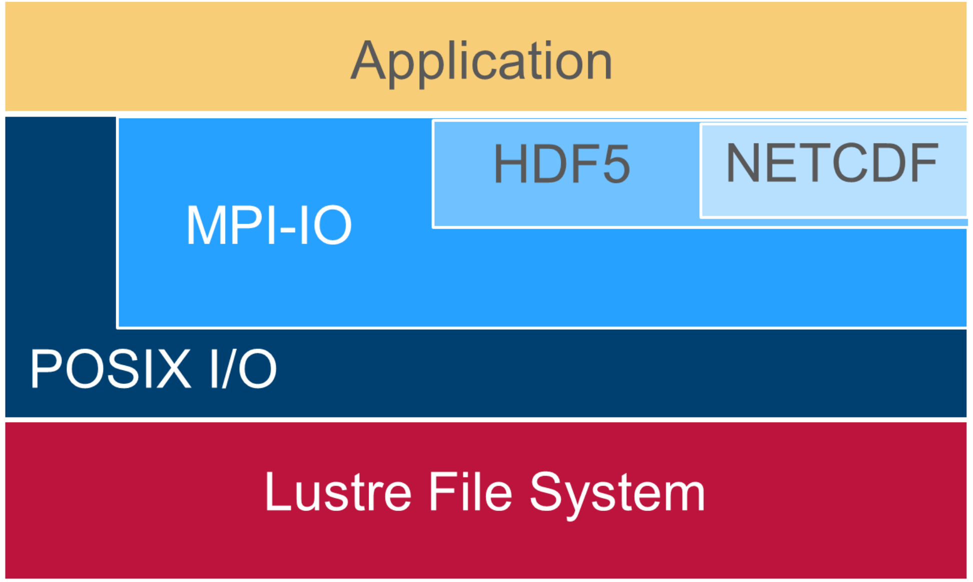
CRAY™

AGENDA

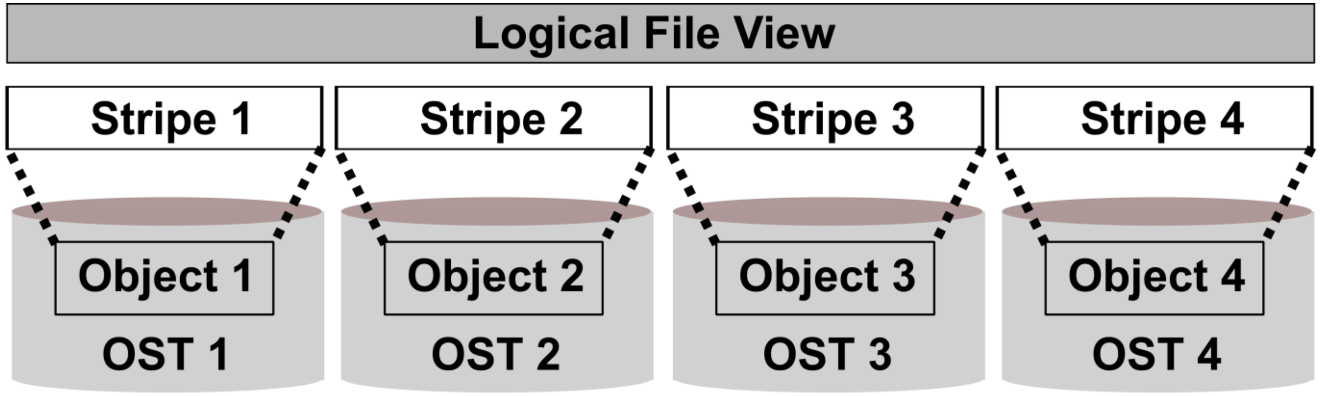
- Purpose
  - Present experimental results from a new Lustre feature called “overstriping”
- Improving shared file workloads on Lustre file systems
  - Shared file performance is challenging on Lustre
  - Longer I/O time means longer job times
- Limitations addressed by Lustre overstriping
- Results
  - ClusterStor L300N
  - Flash based OST
- Summary
- Q&A

ACRONYMS

- APIs
  - POSIX – Portable Operating System Interface
  - MPI-IO – Message Passing Interface I/O
- Lustre
  - OSS – Object Storage Server
  - OST – Object Storage Target
  - LDLM – Lustre Distributed Lock Manager
- Other
  - FPP – File Per Process



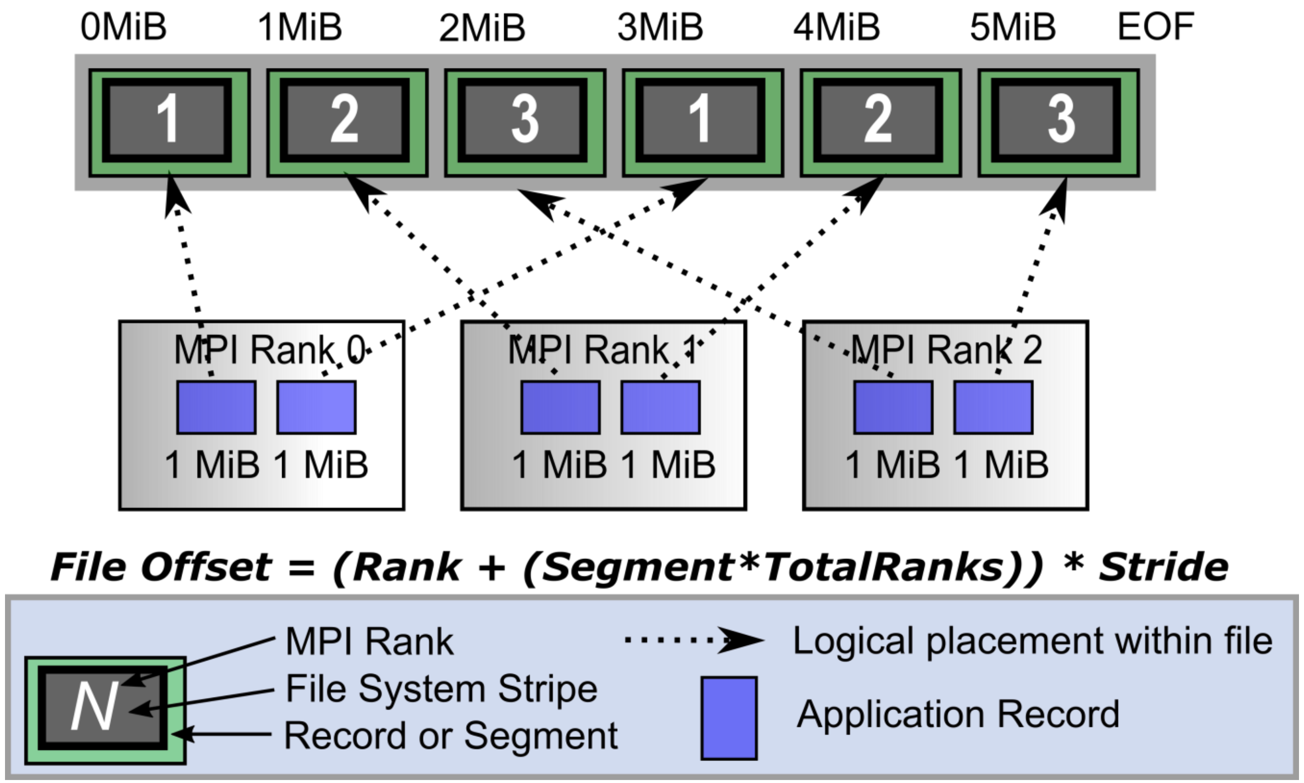
CURRENT LUSTRE STRIPING



```
[user@lustre testdir]$ lfs getstripe shared.4stripes.4osts
shared.4stripes.4osts
lmm_stripe_count:  4
lmm_stripe_size:   1048576
lmm_pattern:       raid0
lmm_layout_gen:    0
lmm_stripe_offset: 4
  obdidx      objid         objid           group
  0           92959130      0x58a719a       0
  1           92893867      0x58972ab       0
  2           92988569      0x58ae499       0
  3           92922653      0x589e31d       0
```

SHARED FILES

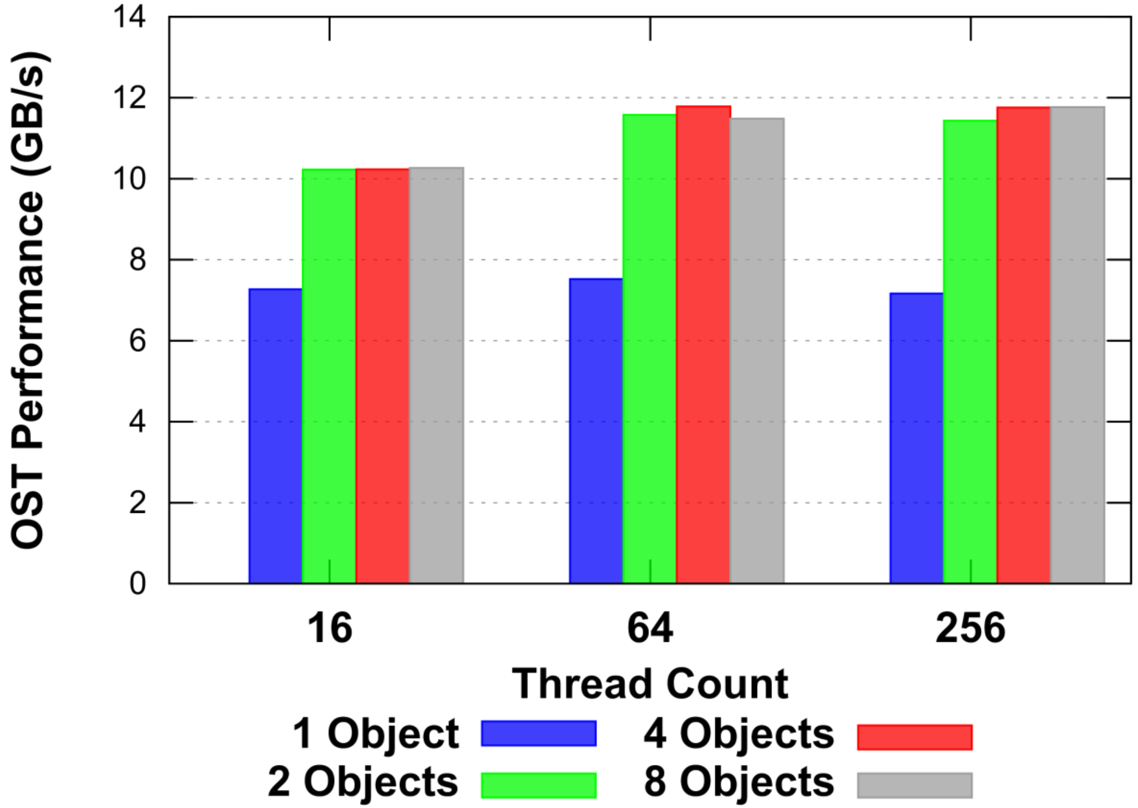
- A single file accessed by many ranks
- Shared file access
  - API (POSIX, MPI-IO), Libraries
  - Access pattern
- Investigation focus
  - Shared files with a strided access pattern
  - Writes
- Currently striping behavior allows
  - Striping widely
  - One stripe per OST per file



LOCAL FILE SYSTEM LIMITATION

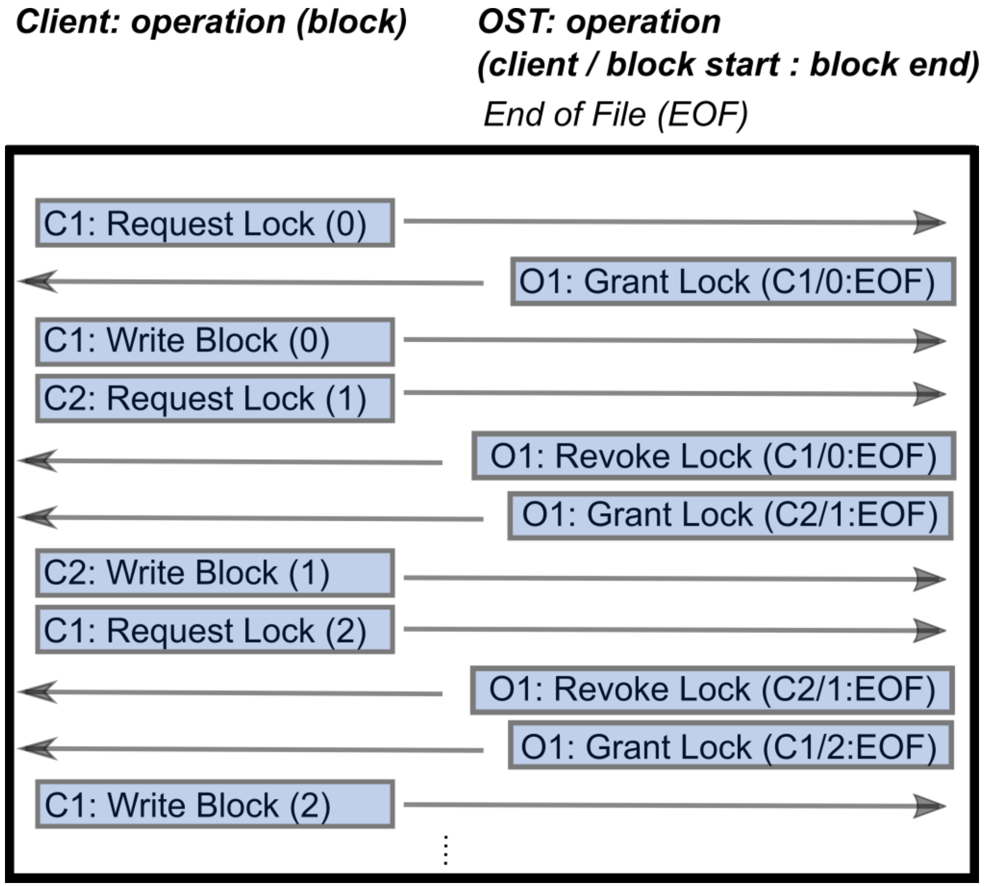
- Page cache limitations
  - High bandwidth rates constantly add and free pages from cache for a single object
  - Incremental performance improvements but already highly optimized
- Flash OST single object limit
  - 7.1 GB/s for write
  - 7.5 GB/s for read
- Additional objects required to achieve expected performance
- Increasing OST speeds make this issue more acute

Obdfilter-survey Write Performance on Single Flash OST



LDLM CONTENTION

- Lustre maintains consistency through locks of a byte range
  - Non-overlapping byte range locks are allowed
  - Lustre optimizes by expanding lock requests causing artificial conflicts
- Multiple Lustre clients needed to achieve expected OST performance
- Increasing OST speeds make this issue more acute



Client: operation (block)

OST: operation (client / block start : block end)

End of File (EOF)

C1: Request Lock (0)

O1: Grant Lock (C1/0:EOF)

C1: Write Block (0)

C2: Request Lock (1)

O1: Revoke Lock (C1/0:EOF)

O1: Grant Lock (C2/1:EOF)

C2: Write Block (1)

C1: Request Lock (2)

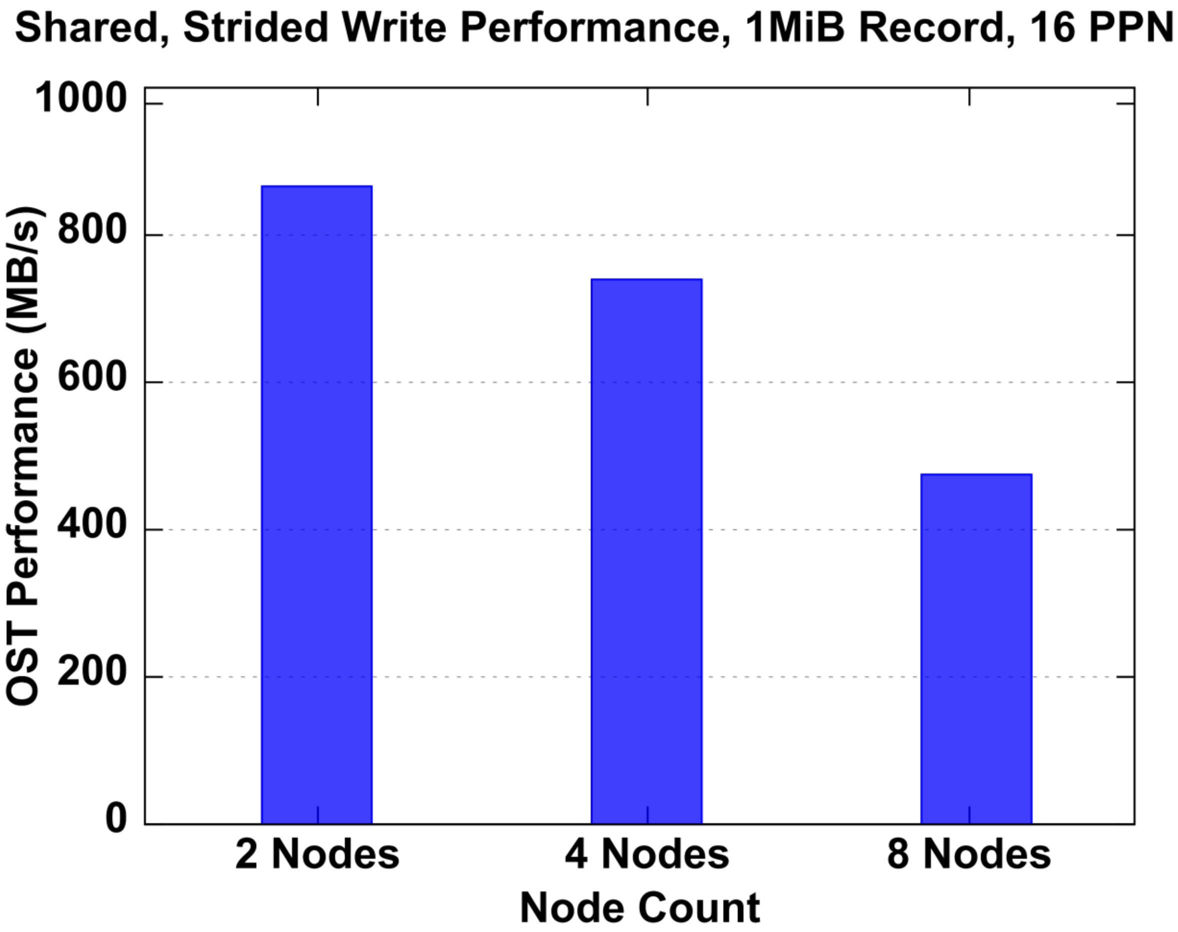
O1: Revoke Lock (C2/1:EOF)

O1: Grant Lock (C1/2:EOF)

C1: Write Block (2)

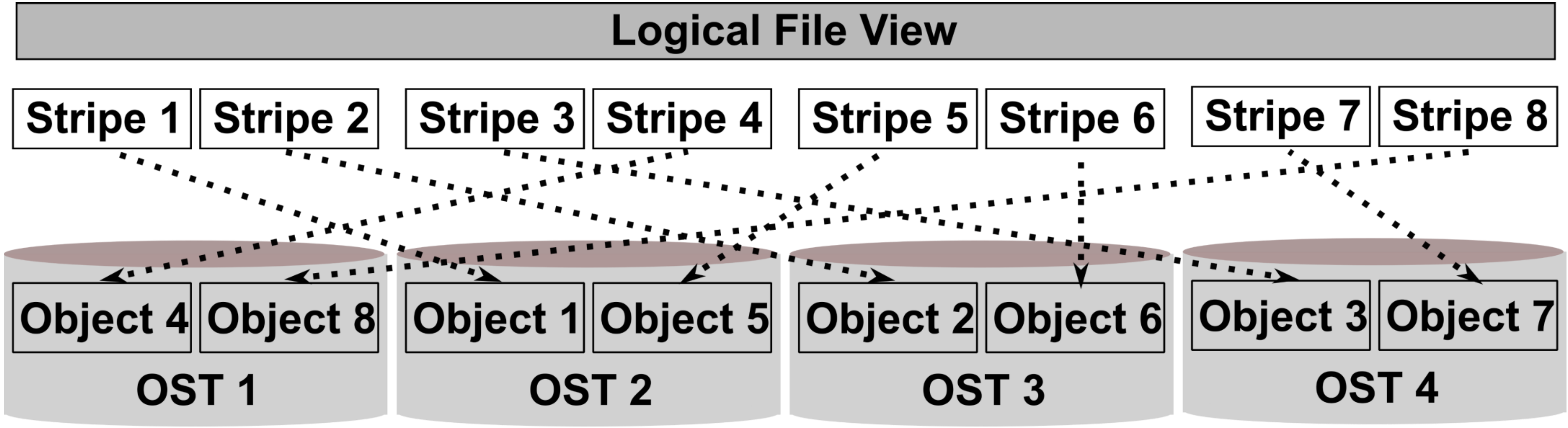
⋮

LDLM CONTENTION PERFORMANCE



OVERSTRIPING DEFINED

- Multiple stripes per OST
- Implementation
  - Remove sanity checks for a single stripe per OST
  - Modify ***lfs*** to describe and show layouts



OVERSTRIPING COMMANDS

- The following examples assume a file system with 4 OSTs
- Lustre pools can be used to restrict OSTs stripes are placed on
- Currently planned options

Striping Description	Command	Result
Striping	`lfs setstripe  --stripe-count 4 filename`	4 Stripes on 4 OSTs
Overstriping	`lfs setstripe --overstripe-count 8 filename`	8 Stripes on 4 OSTs
Striping, manual	`lfs setstripe --ost 0,3,1,2 filename`	4 stripes on 4 OSTs, in order
Overstriping, manual	`lfs setstripe --ost 0,1,0,2,1,2,3,3 filename`	8 stripes on 4 OSTs, in order

OVERSTRIPING LFS GETSTRIPE

```
[user@lustre testdir]$ lfs getstripe shared.8stripes.4osts
shared.8stripes.4osts
lmm_stripe_count:  8
lmm_stripe_size:   1048576
lmm_pattern:       raid0 - overstriping
lmm_layout_gen:    0
lmm_stripe_offset: 8
lmm_pool:          disk
  obdidx    objid       objid          group
  1         39748073    0x25e81e9      0
  2         39840878    0x25fec6e      0
  3         39789909    0x25f2555      0
  0         39826705    0x25fb511      0
  1         39748074    0x25e81ea      0
  2         39840879    0x25fec6f      0
  3         39789910    0x25f2556      0
  0         39826706    0x25fb512      0
```

Overstriping in use

2 stripes per OST

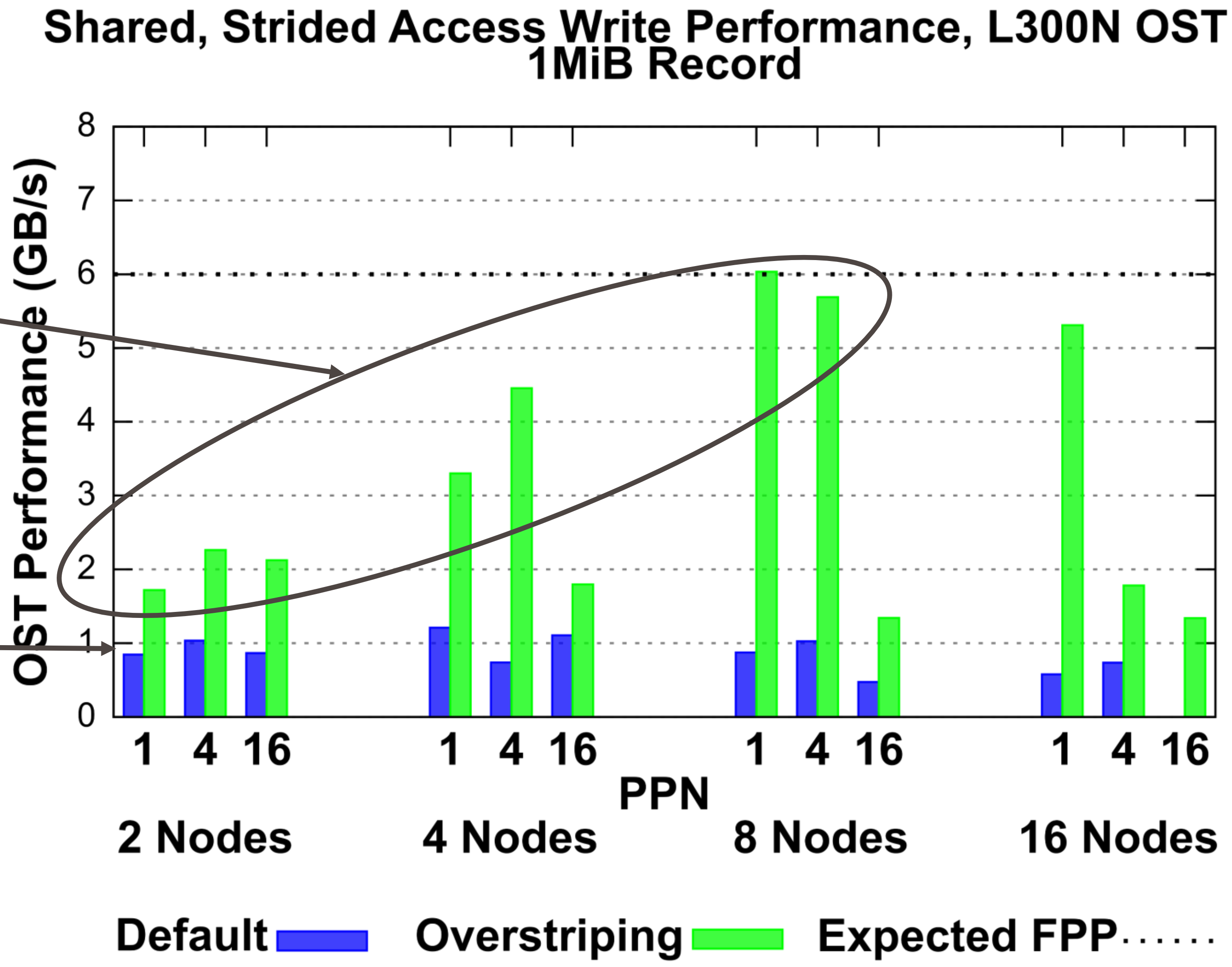
SHARED FILE PERFORMANCE

- Test Environment
  - 2 L300N and 1 L300F ClusterStor SSUs
    - Flash OST based on L300F hardware but no RAID protection
  - Infiniband based cluster
    - 48 clients, dual socket Ivy Bridge
- IOR used for client performance testing
  - A shared, strided access pattern
  - Each node writes 64GB of data, equal to the amount of memory on the node

OVERSTRIPING WRITES ON DISK

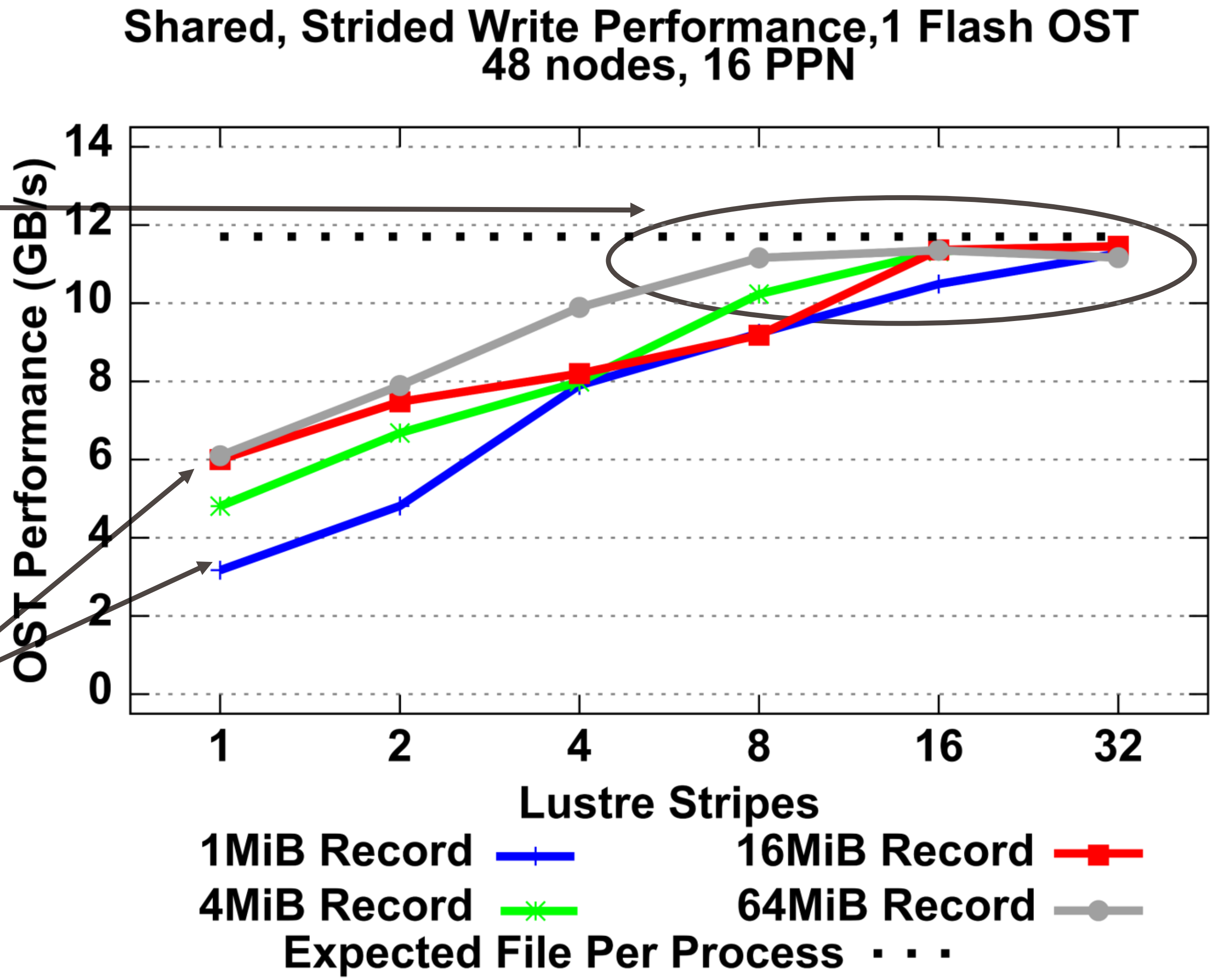
Increasing performance, up to 8 nodes, with overstriping

Consistently low performance with a single stripe



OVERSTRIPING WRITES ON FLASH

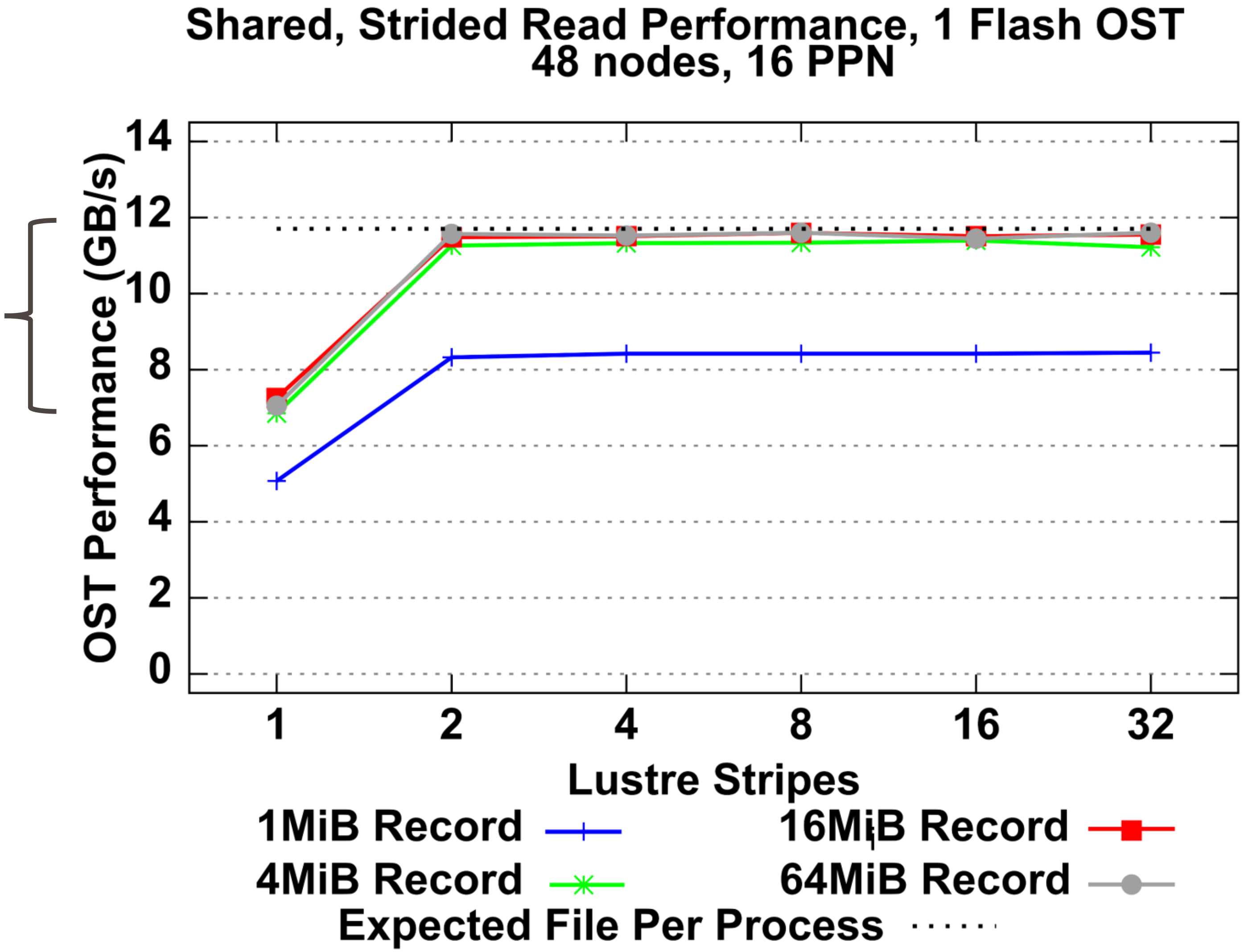
All record sizes achieve near peak performance



Increased LDLM contention with smaller record and Lustre stripe sizes

OVERSTRIPING READS ON FLASH

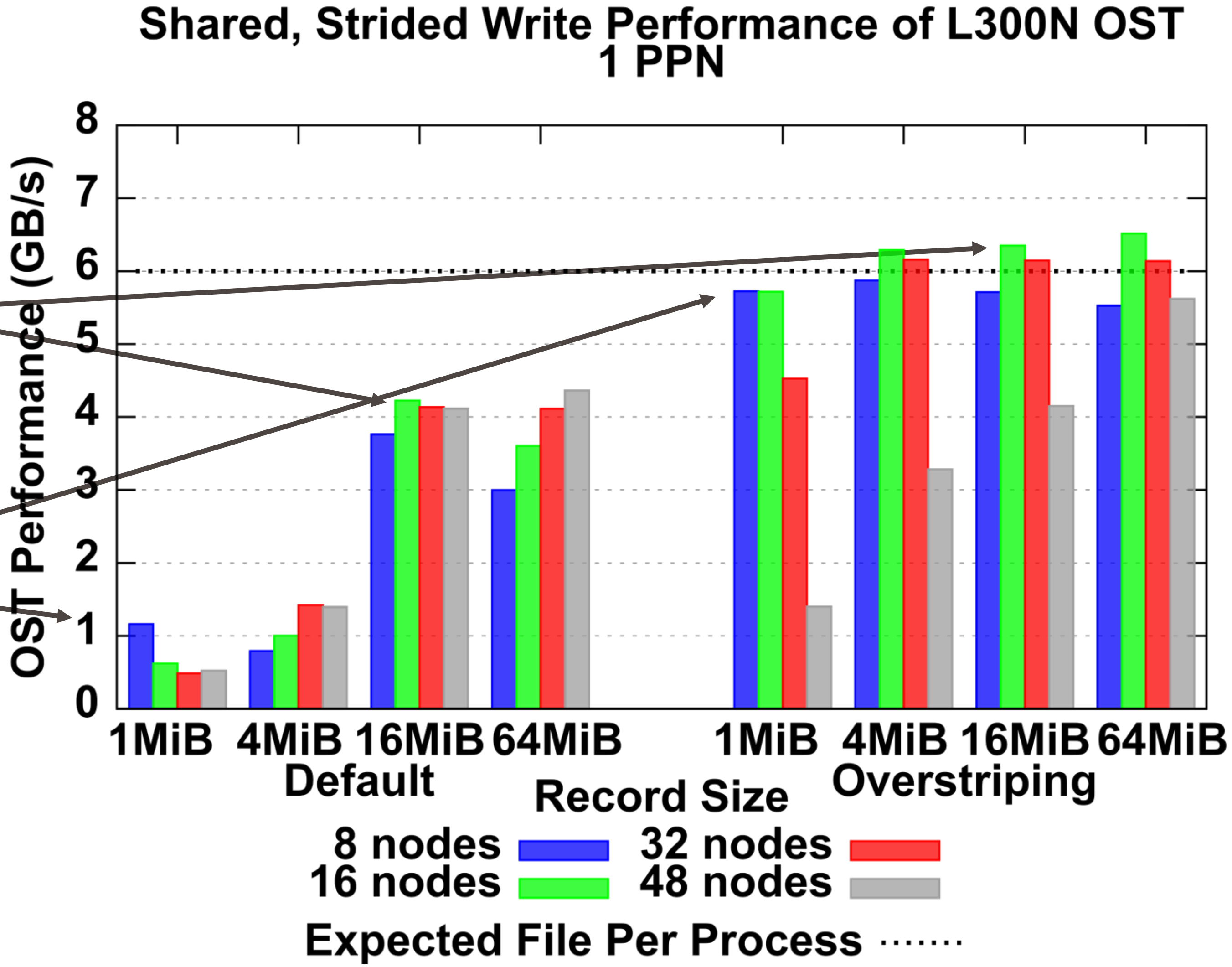
Second stripe overcomes local file system performance limitation



AGGREGATOR PERFORMANCE ON FLASH

Larger record and stripe sizes show less improvement due to less LDLM contention

5x - 6x improvement



AVAILABILITY

- Overstriping will land in upstream Lustre 2.13
- Likely included in NEO and CLE releases later this year
- Support in Cray MPICH is not set
  - Overstriping can still be used for MPI-IO just not set through MPI-IO hints

SUMMARY

- Shared file performance limitations cause longer job times
- Lustre overstriping addresses two limitations
  1. Local file system performance
  2. LDLM Contention
- Addressing this limitations will be more important as OST speeds increase
- Overstriping set using the same utility as current striping
- Overstriping improves shared file write and read performance
  - Large improvements, up to 6x, between single stripe and overstriping
  - Multiple objects needed for full read performance
  - Multiple objects and reduced LDLM contention for full write performance

SAFE HARBOR STATEMENT

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts.

These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.

THANK YOU

QUESTIONS?

✉ mmoore@cray.com

✉ pfarrell@whamcloud.com

cray.com

@cray_inc

linkedin.com/company/cray-inc-/