Small File I/O Performance in Lustre

Mikhail Pershin, Joe Gmitter

Intel HPDD

April 2018

Overview

- Small File I/O Concerns
- Data on MDT (DoM) Feature Overview
- DoM Use Cases
- DoM Performance Results
- Small File I/O Improvements Beyond DoM

Small File I/O Concerns

- Small file data uses a single OST
  - No parallel access to data
- Random I/O patterns
  - More seeking to access data
  - More latency sensitive
  - Slows down concurrent streaming I/O
- Data is small
  - Can't do data read-ahead
  - More RPCs for the same amount of data

DoM can help!

- Aims to improve small file I/O performance
- Stores small file data directly on the MDT
- DoM files grow on OSTs after the MDT size limit is reached
- Feature was introduced in Lustre 2.11.0

DoM Feature Foundation

- Avoids extra RPCs – less RPC pressure on OSTs (more on MDT)
  - DLM locks, attributes, read RPC, glimpse
- Separates large and small I/O data streams
  - Less I/O overhead on OSTs
  - Avoid blocking small I/Os behind large streaming I/O workloads
- New benefits from data residing on MDT(s) are possible
  - Data is accessible during metadata operations
  - File size is immediately available
- MDT servers are often faster, a factor leading to increased performance
  - Difference in storage types can affect performance significantly

DoM Implementation Overview

- The Progressive File Layout (PFL) feature is the foundation
  - The PFL file starts with the first component on the MDT
  - The file will grow to OSTs when the MDT size limit is reached
- A new layout for DoM files is introduced
  - Example: `lfs setstripe -E 1M -L mdt … <path>`
- Client was modified to send I/O requests to an MDT
- MDT was modified to serve incoming I/O requests
  - New I/O service threads
  - I/O methods on the MDT
  - MDT and OST become nearly the same for I/O request handling

Where can DoM help?

- Accessing small file data residing on an MDT
  - read and stat operations
  - Further improvements in DoM Phase 2 planned, e.g. read-on-open
- Reducing the amount of small random I/O on OSTs
  - Better streaming I/O results on OSTs
  - Improved latency of OSTs handling streaming I/Os
  - Especially for OSTs with HDDs and RAID-5/6 redundancy
- MDT has better storage, e.g. NVMe vs HDDs on OSTs
  - Re-use this potential for small files without need for OSTs upgrade

Where is DoM not ideal?

- Write patterns in general
  - DoM not much better for writes if OST and MDT hardware is identical
  - DNE may help with this, pending targeted phase 2 optimization improvements
  - Helps to improve OSTs streaming writes indirectly
    - Combined write efficiency between small and large I/O workloads
- File create and other metadata operations
  - Metadata operations will have the same result for small files in DoM
  - A DoM file create needs extra work for a special layout

DoM Limits

There are two parameters to control DoM limits:

- Per-file Value
  - Amount of a file’s data to store on the MDT
  - Generated on file creation as size of the first component
  - It can be set directly or inherited from the parent directory
- Per-MDT Parameter
  - Maximum data size allowed by the server
  - Can be changed and saved in config

Setting DoM Stripe Value

The ‘`lfs setstripe`’ command is the tool to perform this action.

- Create a new file with 1MB DoM component in PFL file:

```
lfs setstripe –E 1M –L mdt –E EOF -c 4 <file>
```

- Set default 64KB DoM component for a directory in PFL file:

```
lfs setstripe –E 64K –L mdt –E 64M –c 1 –E -1 –c -1 –S 4M <dir>
```

- Get information about DoM files:

```
lfs getstripe [–L mdt] <file>
```

```
lfs find –L mdt <dir>
```

Display DoM Stripe Information

`lfs getstripe` example output

```
client$ lfs getstripe /mnt/testfs/dom/64M
  lcm_entry_count:   3
    lcme_id:              1
    lcme_extent.e_start: 0
    lcme_extent.e_end:   65536
      lmm_stripe_count:  0
      lmm_stripe_size:   65536
      lmm_pattern:       mdt
    lcme_id:              2
    lcme_extent.e_start: 65536
    lcme_extent.e_end:   33554432
      lmm_stripe_count:  1
      lmm_stripe_size:   65536
      lmm_pattern:       raid0
      - 0: { l_ost_idx: 3, l_fid: [0x100030000:0x138a:
0x0] }
```

```
lcme_id:              3
    lcme_extent.e_start: 33554432
    lcme_extent.e_end:   EOF
      lmm_stripe_count:  4
      lmm_stripe_size:   4194304
      lmm_pattern:       raid0
      - 0: { l_ost_idx: 1, l_fid: [0x100010000:0x138b:0x0] }
      - 1: { l_ost_idx: 2, l_fid: [0x100020000:0x138b:0x0] }
      - 2: { l_ost_idx: 0, l_fid: [0x100000000:0x138a:0x0] }
      - 3: { l_ost_idx: 3, l_fid: [0x100030000:0x138b:0x0] }
```

Limiting DoM size on the MDT

Main parameter is **`lod.<mdt_name>.dom_stripesize`**

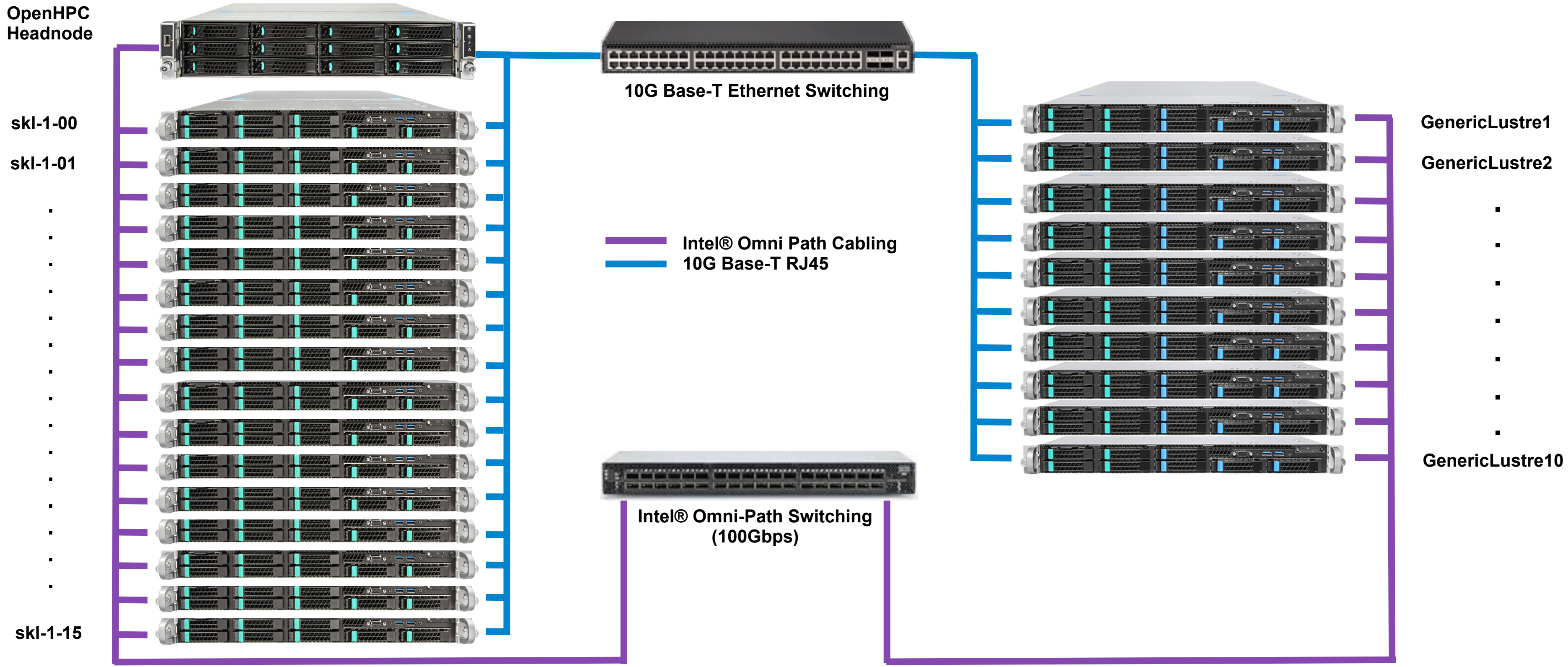
- Controls the maximum DoM component size on the server
- Disables DoM files on server when set to 0
- Runtime parameter setting and getting:

```
# lctl set_param lod.*MDT0000*.dom_stripesize=65536
# lctl set_param -P lod.*.dom_stripesize=2M
# lctl get_param lod.*.dom_stripesize
```

- Save parameter in config:

```
# lctl conf_param fsname-MDT0000.lod.dom_stripesize=0
```

Performance Testbed Architecture



Performance Testbed Architecture (cont.)

1 MDS/1 MDT, 8 OSS/32 OST, 16 clients

- Servers: 10x Generic Lustre servers with two slightly different configurations
  - Each system comprises of:
    - 2x Intel® Xeon E5-2697v3 (Haswell) CPUs, 1x Intel® Omni-Path x16 HFI, 128GB DDR4 2133MHz RAM
    - 8 with 4x Intel® P3600 2.0TB 2.5" (U.2) NVMe devices
    - 2 with 4x Intel® P3700 800GB 2.5" (U.2) NVMe devices
    - 1 with 2x Intel® S3700 400GB's for MGT
- Clients: 16x 2S Intel® Xeon Scalable Compute nodes
  - 1x OpenHPC Headnode
  - Hardware components:
    - 2x Intel® Xeon Scalable 6148 CPUs, 1x Intel® Omni-Path x16 HFI, 192GB DDR4 2466MHz, local boot SSD
- Fabric: 100Gbps Intel® Omni-Path
  - None-blocking fabric with single switch design.
  - Server optimisations: "`options hfi1 sge_copy_mode=2 krcvqs=4 wss_threshold=70`"
    - Improve generic RDMA performance, only recommended on servers that do not do any MPI

Data-on-MDT Benchmarks

File Striping Cases

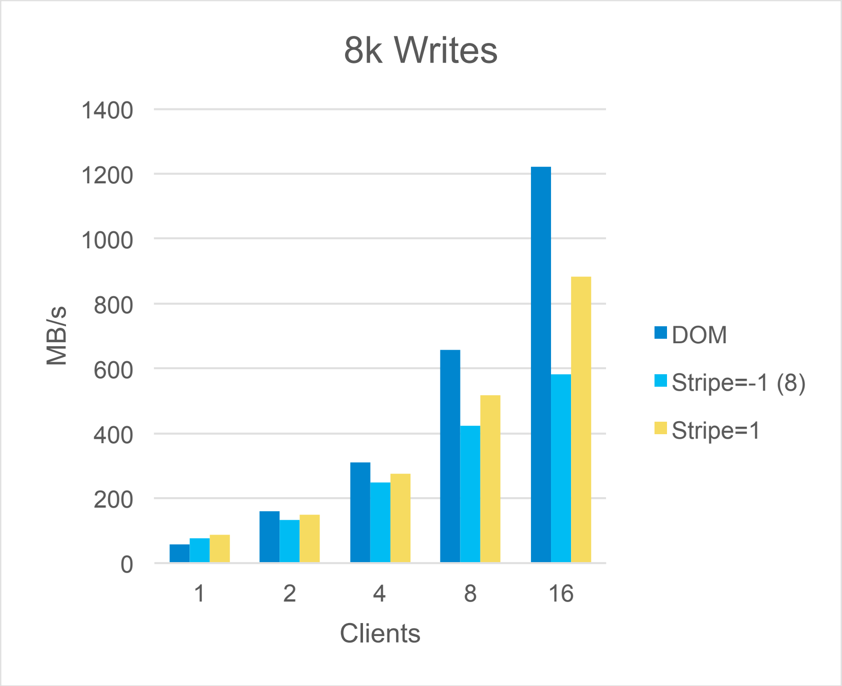
- OST file, 1 stripe
- OST file, -1 stripe
  - 8 stripes in this test scenario
- DOM file

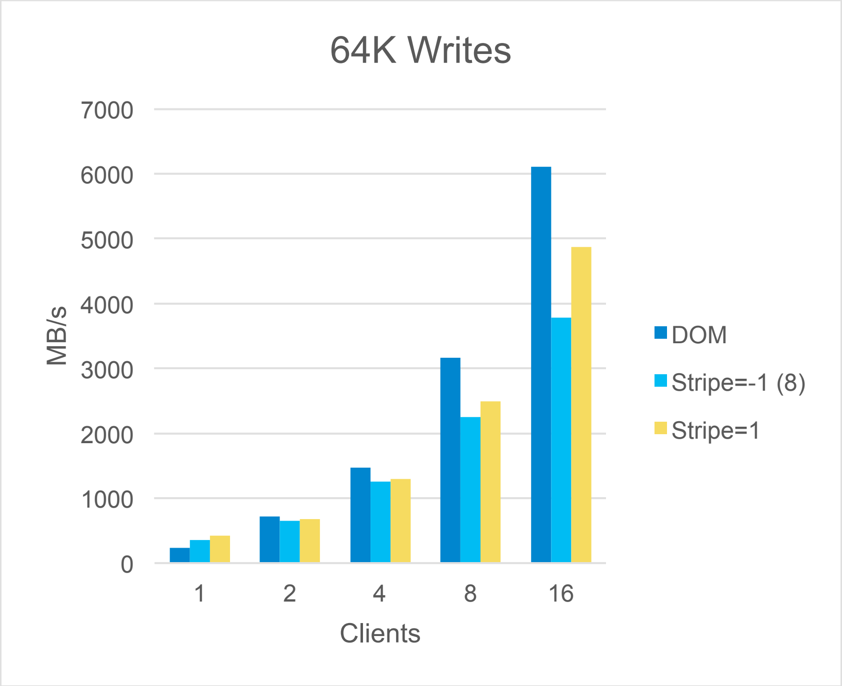
Benchmarks & Options

- IOR, FIO for detailed read/write
  - File per process
  - Random read/write
- compilebench, dbench
  - Simulates user activity with target to metadata and small file operations

IOR, WRITE, sub-DoM size

IOR … -t 8k/64k -s 16 -w -F -k -E -z -m -Z -i 4096

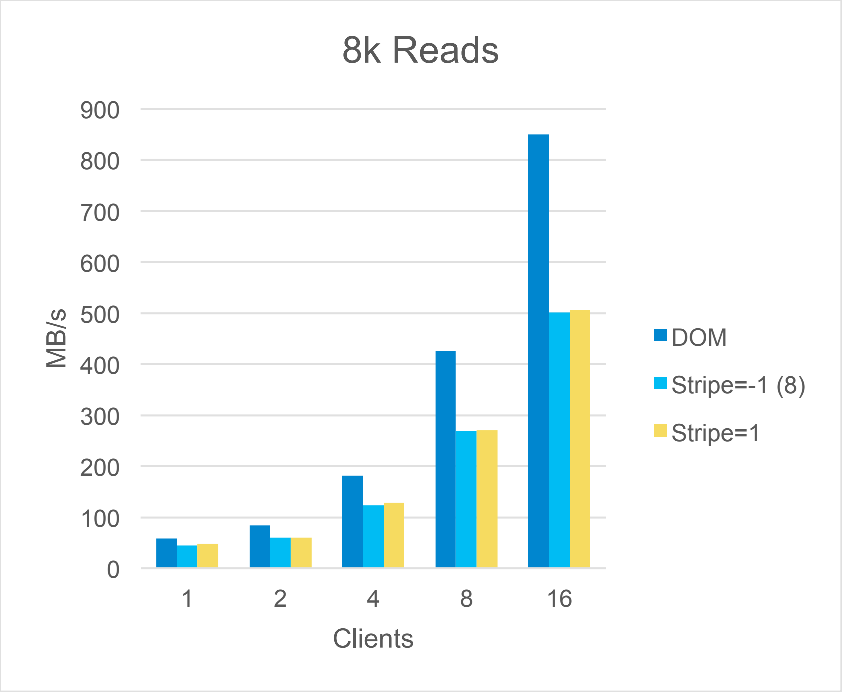


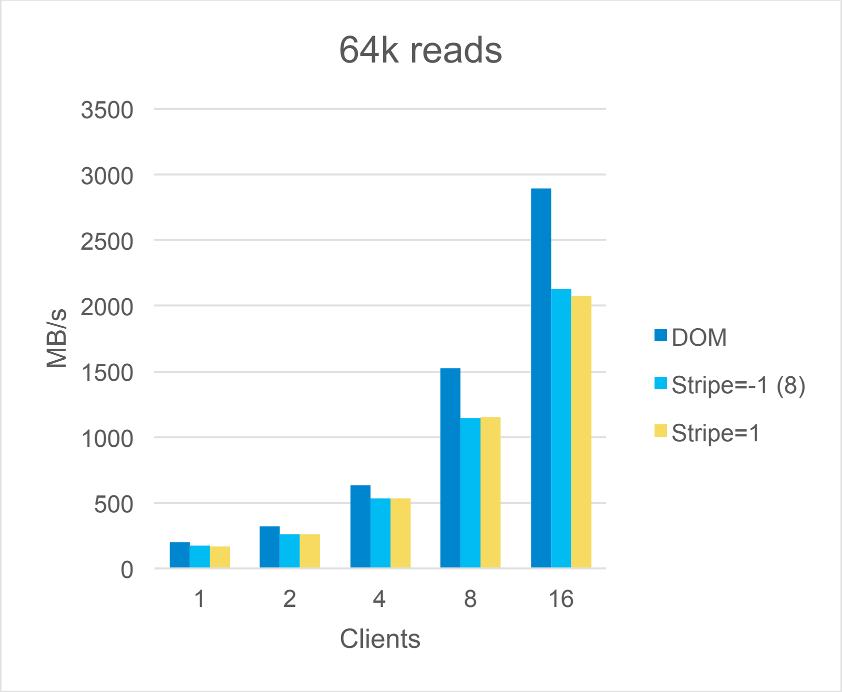


DoM showing improved performance as client count increases

IOR, READ, sub-DoM size

IOR … -t 8k/64k -s 16 -r -F -k -E -z -m -Z -i 4096

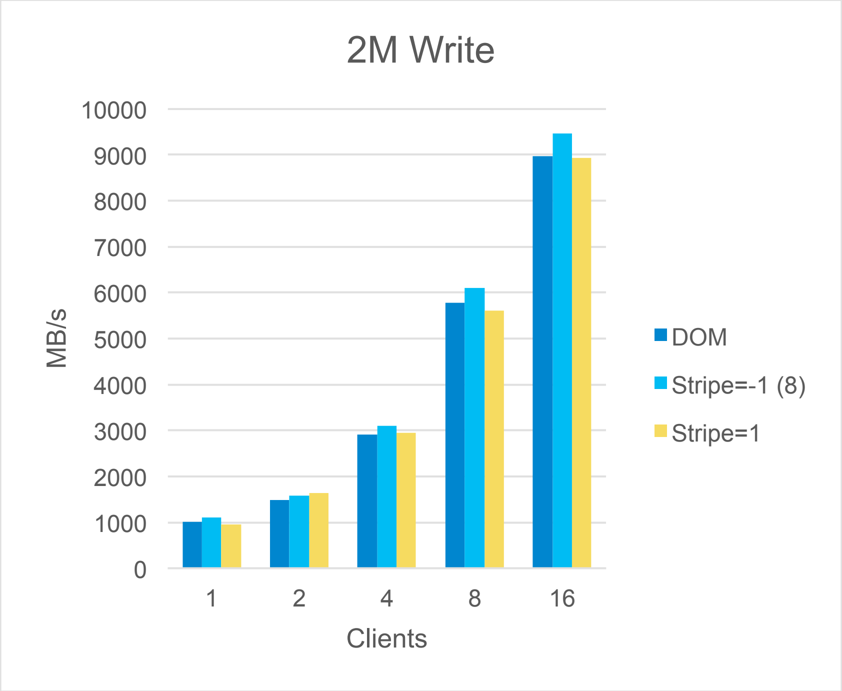


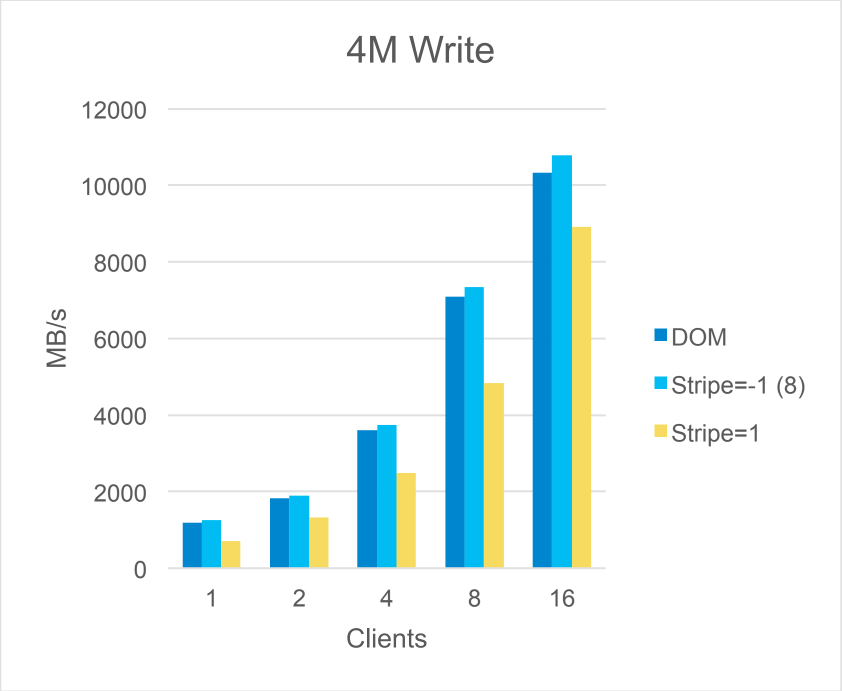


DoM performance significantly better, especially for smaller reads

IOR, WRITE, over-DoM size

IOR … -t 512k -s 16 -w -F -k -E -z -m -Z -i 4096

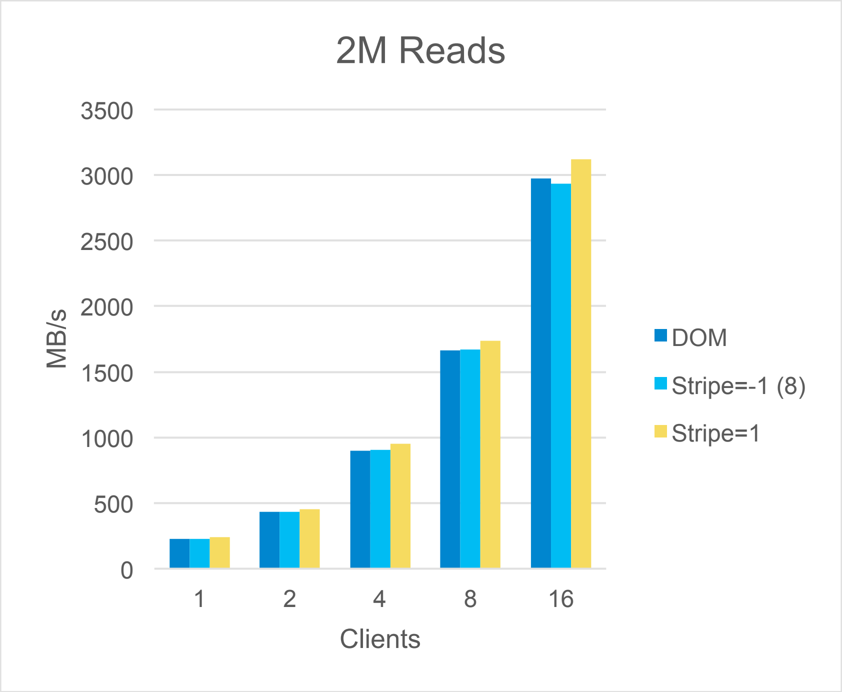


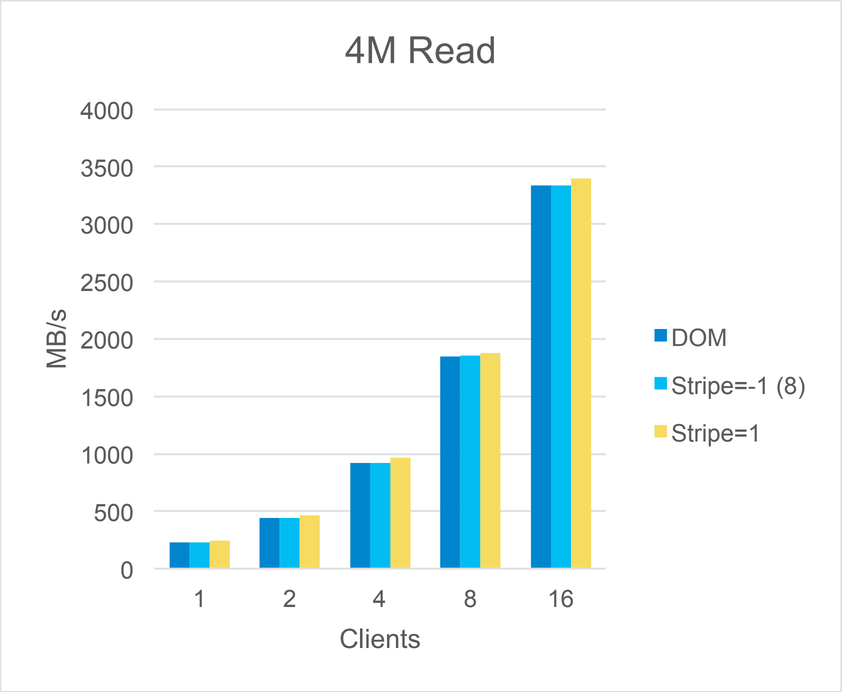


No write performance degradation for files over the DoM size limit

IOR, READ, over-DoM size

IOR … -t 512k -s 16 -r -F -k -E -z -m -Z -i 4096

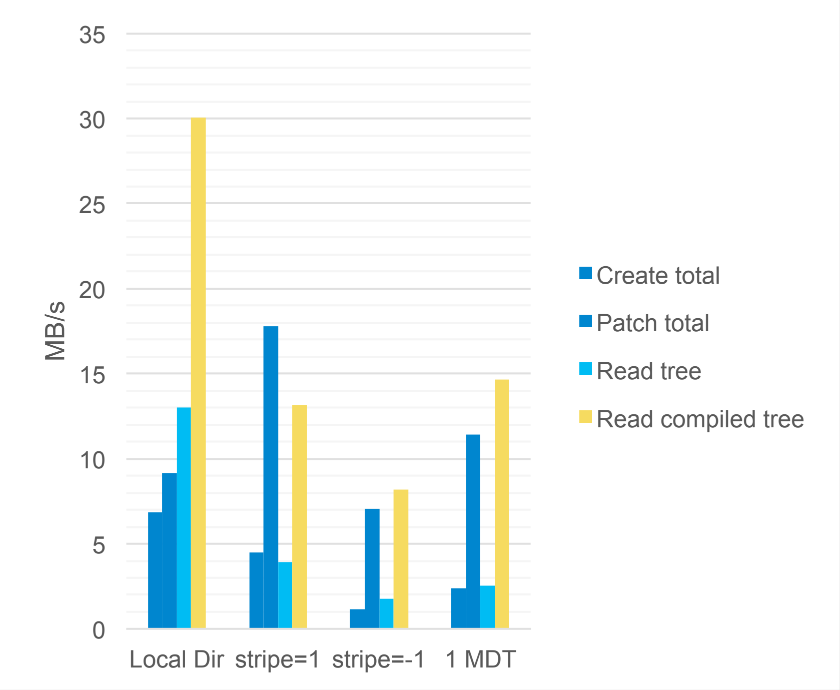


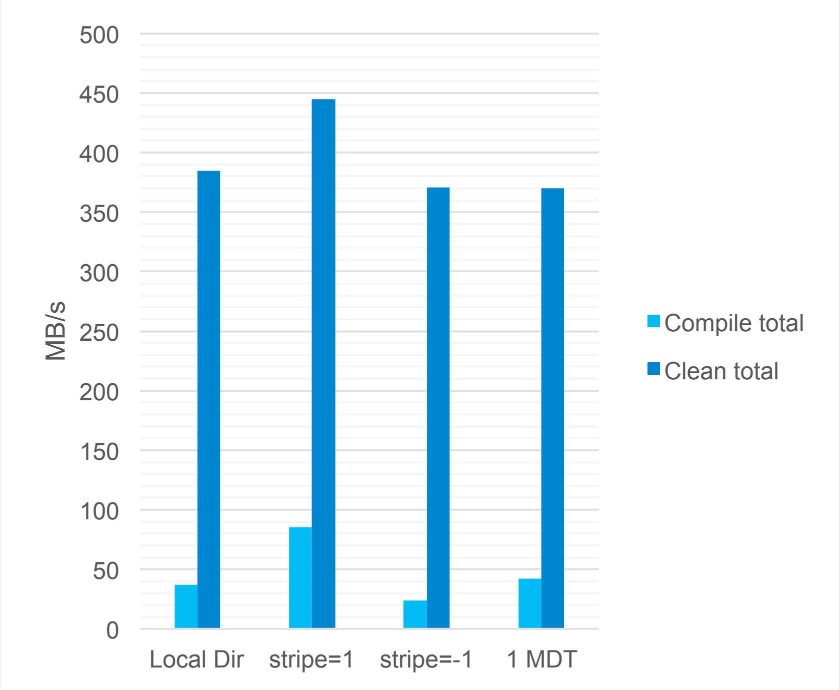


No read performance degradation for files over the DoM size limit

Compilebench

Compilebench –D <dir> -i 10 -r 20





Not all cases show an improvement in DoM phase 1 and are a focus area for phase 2

DoM Phase 2

There is still a significant amount of work to do with DoM

- DOM+DNE optimization
- DOM file migration to/from OST and MDT-to-MDT
- Various optimizations for read and stat operations
- Performance fixes and improvements for known issues

DoM Phase 2: DNE Optimization

- Not a primary target for DoM Phase 1
- DNE has good potential in improving DoM write operations
- Preliminary testing shows that it is scalable under high load
- DNE allows control for how MDTs are used in DoM
- LU-10895

DoM Phase 2: Migration

Add DoM migration support to `lfs migrate` (LU-10177):

- To MDT from OSTs
  - For small files currently residing on OSTs
- To OSTs from MDT
  - For files which have grown beyond the DoM component
  - If space needs to be reclaimed on the MDT
- Between MDTs for load/space balancing

DoM Phase 2: Performance Improvements

Further performance improvements are a primary goal for DOM Phase 2

- DoM Phase 1 was targeted for functionality and stability
- Optimizations for READ/STAT:
  - Read prefetch on file opening (read-on-open LU-10181)
    - Current patch shows faster `read()` operations
  - Glimpse-ahead: MDT returns size from client on `stat()` (LU-10181)
  - Read data on `stat()` and `readdir()` (LU-10919)

DoM Recommendations

- Review the MDT role carefully with DoM
  - The MDT needs more space
  - The MDT will experience higher RPC pressure
- SSD-based MDS is perfect for DoM
  - Especially when OSTs uses HDD or RAID-5/6
- Consider a DoM+DNE setup in phase 2
  - Scale metadata and I/O performance
  - Can use larger capacity MDTs for DoM specifically

Beyond DoM: Small File I/O Improvement Work

Other potential areas beyond DoM for advancing small file I/O performance:

- Close in background (one fewer sync RPC)
- Save on sync `ENQUEUE` RPC if open/create RPC returns an exclusive lock
- Improved block grants handling with ZFS
- Multi-file read/write in a single RPC/RDMA
- Client initial create/open metadata handling
- Client metadata write-back cache

Legal Notices and Disclaimers

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life-saving, life-sustaining, critical control or safety systems, or in nuclear facility applications.

Intel products may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

This document may contain information on products in the design phase of development. The information herein is subject to change without notice. Do not finalize a design with this information. Intel may make changes to dates, specifications, product descriptions, and plans referenced in this document at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure.

Intel Corporation or its subsidiaries in the United States and other countries may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

Performance estimates or simulated results based on internal Intel analysis or architecture simulation or modeling are provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice Revision #20110804.

Intel® Advanced Vector Extensions (Intel® AVX)* provides higher throughput to certain processor operations. Due to varying processor power characteristics, utilizing AVX instructions may cause a) some parts to operate at less than the rated frequency and b) some parts with Intel® Turbo Boost Technology 2.0 to not achieve any or maximum turbo frequencies. Performance varies depending on hardware, software, and system configuration and you can learn more at http://www.intel.com/go/turbo.

Intel processors of the same SKU may vary in frequency or power as a result of natural variability in the production process.

Intel, the Intel logo, 3D-Xpoint, Optane, Xeon Phi, and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

* Other names and brands may be claimed as the property of others.

intel®