09/22/2014

DataDirect™ NETWORKS
INFORMATION IN MOTION™

# Lustre Metadata Fundamental Benchmark and Performance

**DataDirect Networks Japan, Inc.**

Shuichi Ihara

# Lustre Metadata Performance

DataDirect NETWORKS
INFORMATION IN MOTION

- Lustre metadata is a crucial performance metric for many Lustre user
  - LU-56 SMP Scaling (Lustre-2.3)
  - DNE (Lustre-2.4)
- Metadata performance is related to small file performance on Lustre
- But, metadata performance is still a little mysterious ☺
  - Performance differentiation by metadata type and access patterns?
  - What is the impact of hardware resources for metadata performance?
- This presentation: use standard metadata benchmark tools to analyze metadata performance on Lustre today

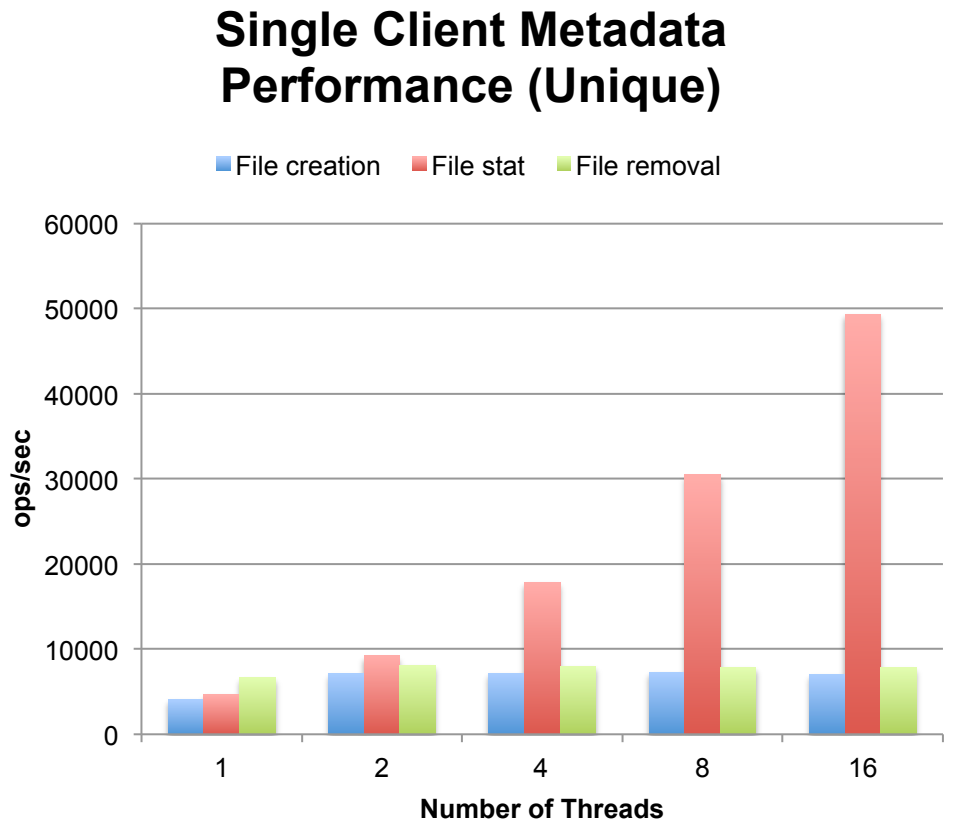
# Lustre Metadata Benchmark Tools

DataDirect NETWORKS
INFORMATION IN MOTION

- mds-survey
  - Build into Lustre code
  - Similar to obdfilter-suvey
  - Generates loads on MDS to simulate Lustre metadata performance
- mdtest
  - Major metadata benchmark tool used by many large HPC sites
  - Runs on clients using MPI
  - Several metadata operation and access patterns are supported

# Single Client Metadata Performance Limitation

- Single client Metadata performance does not scale with threads.




lustre/include/lustre_mdc.h

```
/**
 * Serializes in-flight MDT-modifying RPC requests to preserve idempotency.
 *
 * This mutex is used to implement execute-once semantics on the MDT.
 * The MDT stores the last transaction ID and result for every client in
 * its last_rcvd file. If the client doesn't get a reply, it can safely
 * resend the request and the MDT will reconstruct the reply being aware
 * that the request has already been executed. Without this lock,
 * execution status of concurrent in-flight requests would be
```

Client can send many metadata requests to MDS simultaneously, but MDS needs to store each client's last transaction ID and it's serialized.

```
    /** Intent associated with currently executing request. */
    struct lookup_intent    *rpcl_it;
    /** Used for MDS/RPC load testing purposes. */
    int             rpcl_fakes;
};
```

- LU-5319 supports multiple slots per client in last_rcvd file (Under development by Intel and Bull).

# Modified mdtest for Lustre

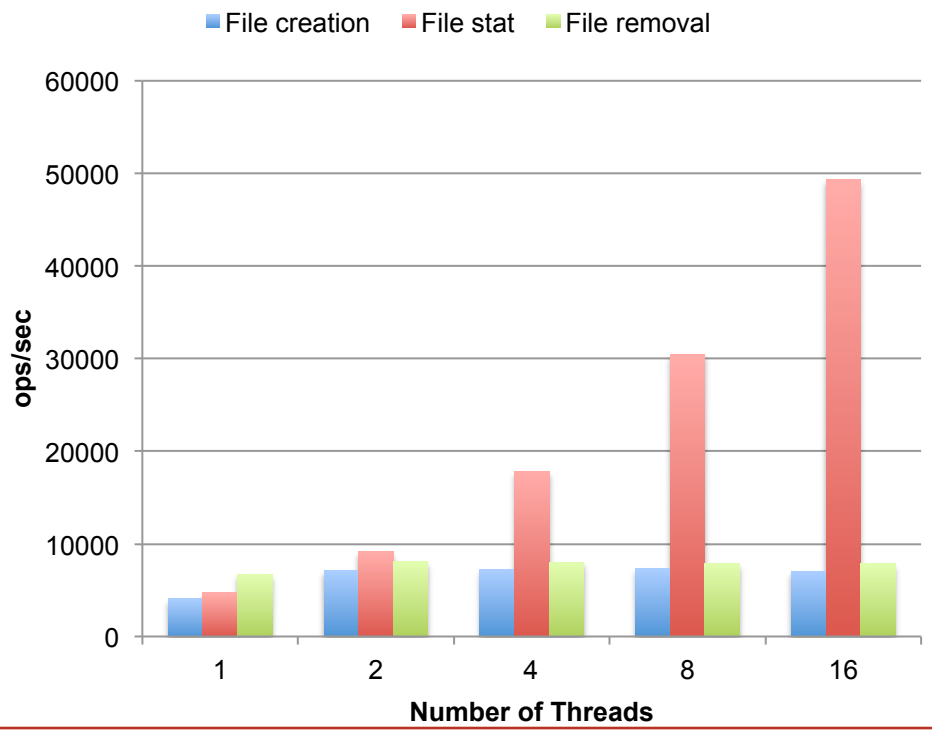
- Basic Function
  - Supports multiple mount points on a single client
  - Helps generating heavy metadata load from single client
- Background
  - Originally developed by Liang Zhen for LU-56 work
  - We rebased and cleaned up codes and made few enhancements
- Enables metadata benchmarks on a small number of clients
  - Regression testing
  - MDS server sizing
  - Performance optimization

# Performance Comparison

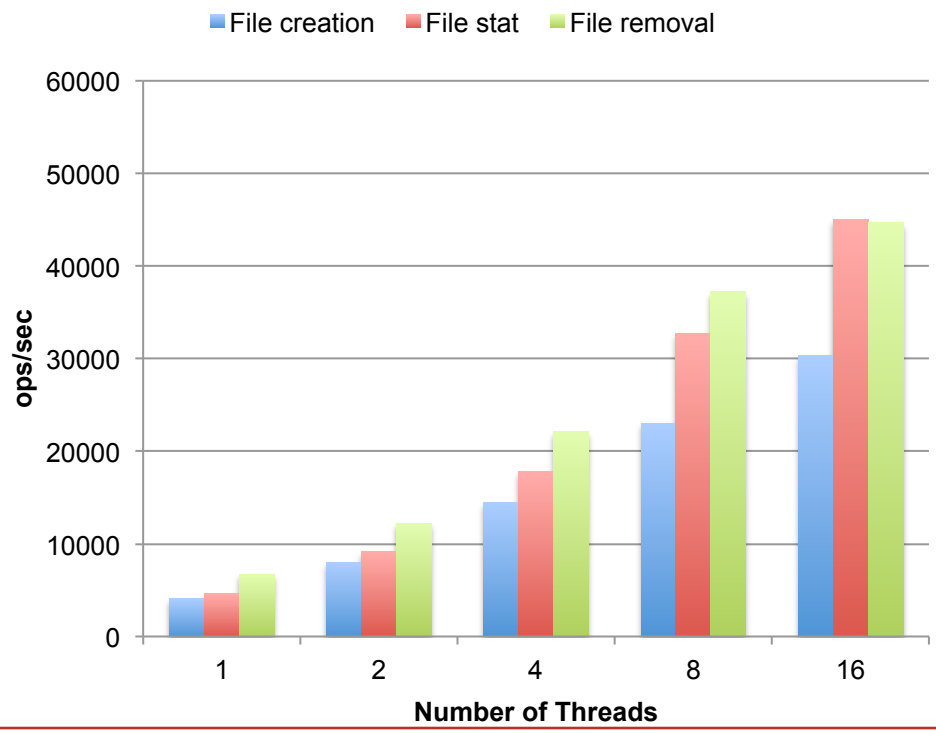
Single Lustre client mounts /lustre_0, /lustre_1, .... /lustre_31 for single filesystem

```
# mdtest –n 10000 –u –d /lustre_{0-15}
```

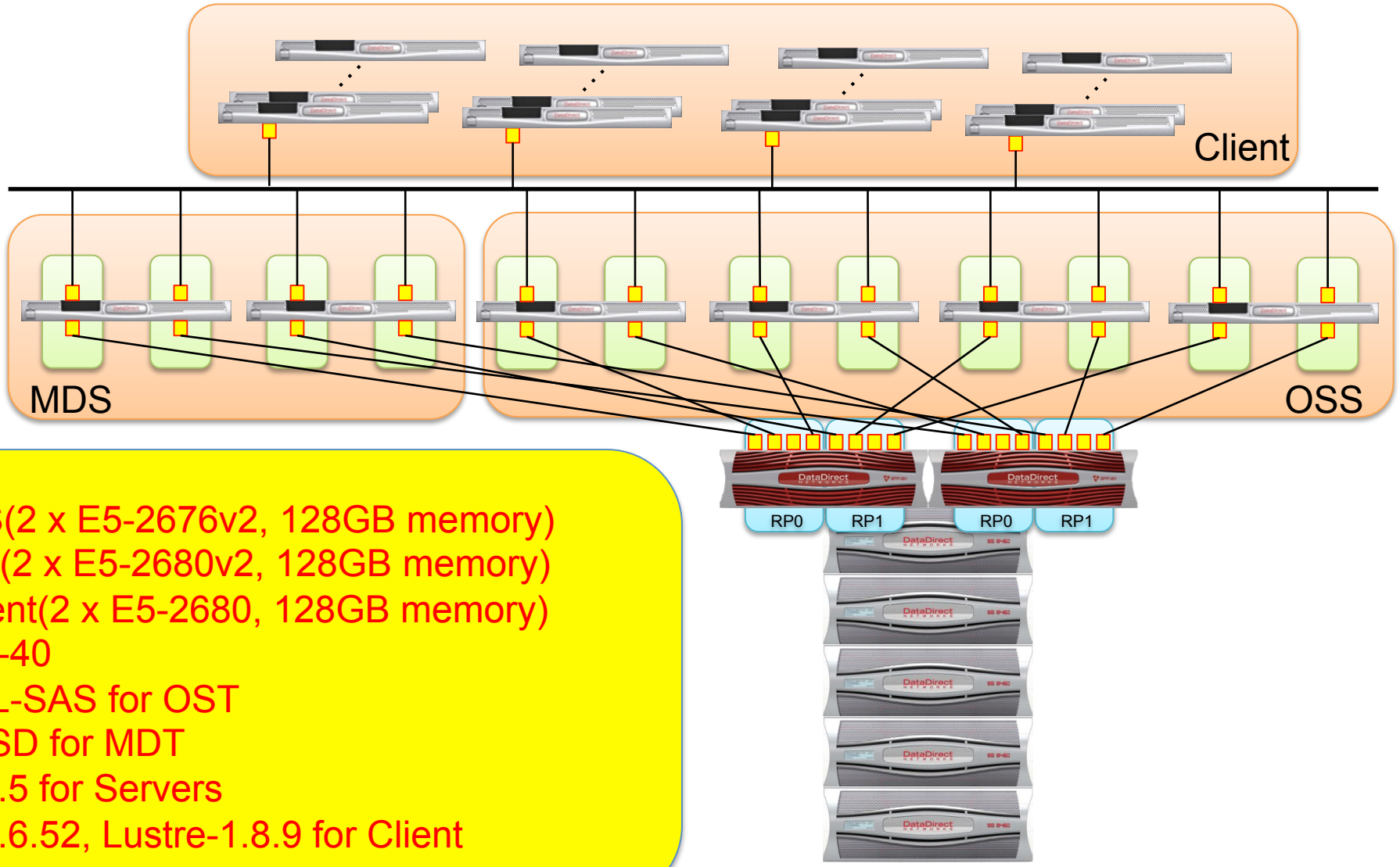
**Single Client Metadata Performance (Unique, single mountpoint)**




**Single Client Metadata Performance (Unique, multi-mountpoints)**



# Benchmark Configuration




2 x MDS(2 x E5-2676v2, 128GB memory)
4 x OSS(2 x E5-2680v2, 128GB memory)
32 x Client(2 x E5-2680, 128GB memory)
SFA12K-40
400 x NL-SAS for OST
8 x SSD for MDT
Lustre-2.5 for Servers
Lustre-2.6.52, Lustre-1.8.9 for Client

# Metadata Benchmark Method

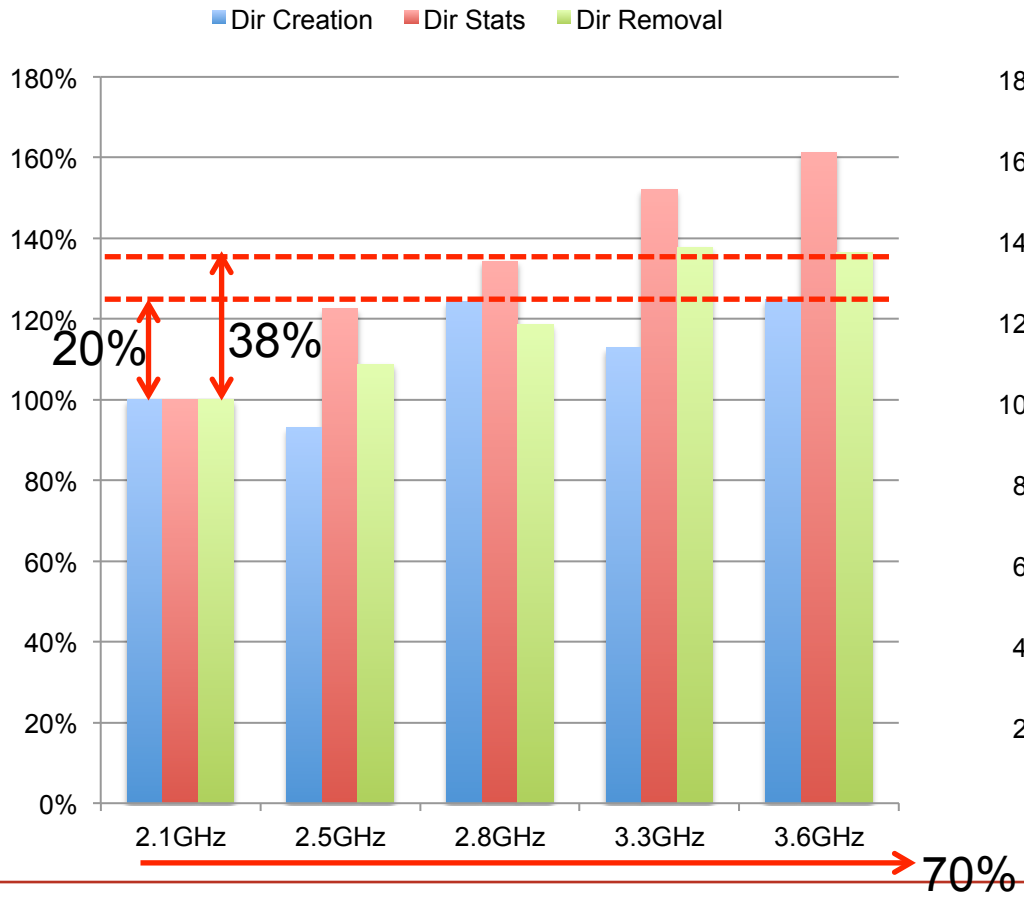
DataDirect NETWORKS
INFORMATION IN MOTION

- Tested Metadata Operations
  - Directory/File Creation
  - Directory/File Stat
  - Directory/File Removal
- Access patterns
  - To Unique Directory and shared directory
    - P0 -> /lustre/Dir0/file.0.0, P1 -> /lustre/Dir1/file.0.1 (Unique)
    - P0 -> /lustre/Dir/file.0.0, P1 -> /lustre/Dir/file.1.0 (Shared)
  - Stride pattern
    - P0 creates files on /lustre/Dir0/file.0.0, P1 calls stat() to P0 created files and finally, P2 calls unlink() to them

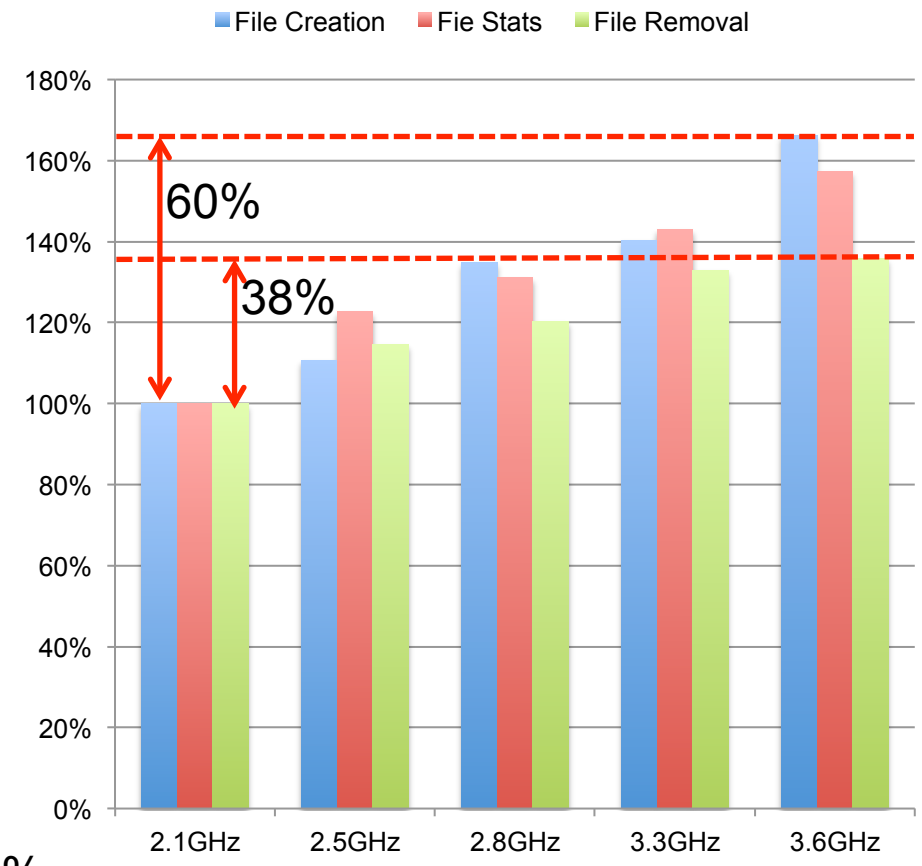
# Lustre Metadata Performance Impact MDS's CPU speed

- Metadata Performance comparison (Unique Directory)
  - 32 clients(1024 mount points), 1024 processes, 1.28M Files
  - Tested on 16 CPU cores with 2.1, 2.5, 2.8, 3.3 and 3.6GHz CPU Speed (MDS)

**Directory Operation(Unique)**




**File Operation(Unique)**

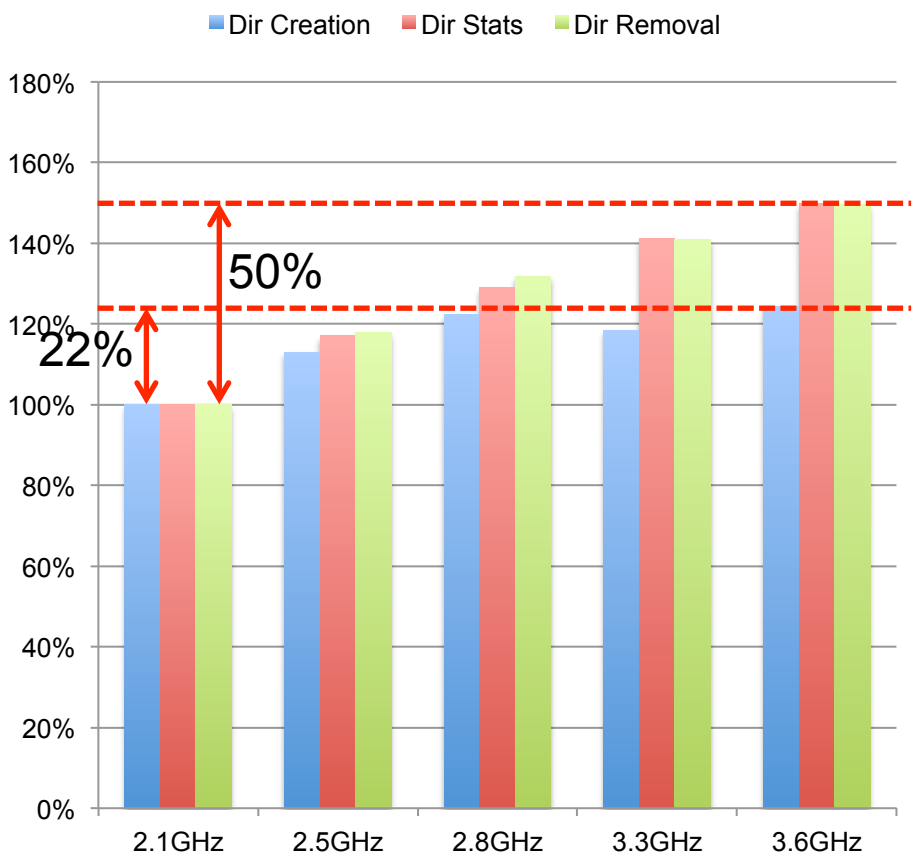


# Lustre Metadata Performance Impact MDS's CPU speed

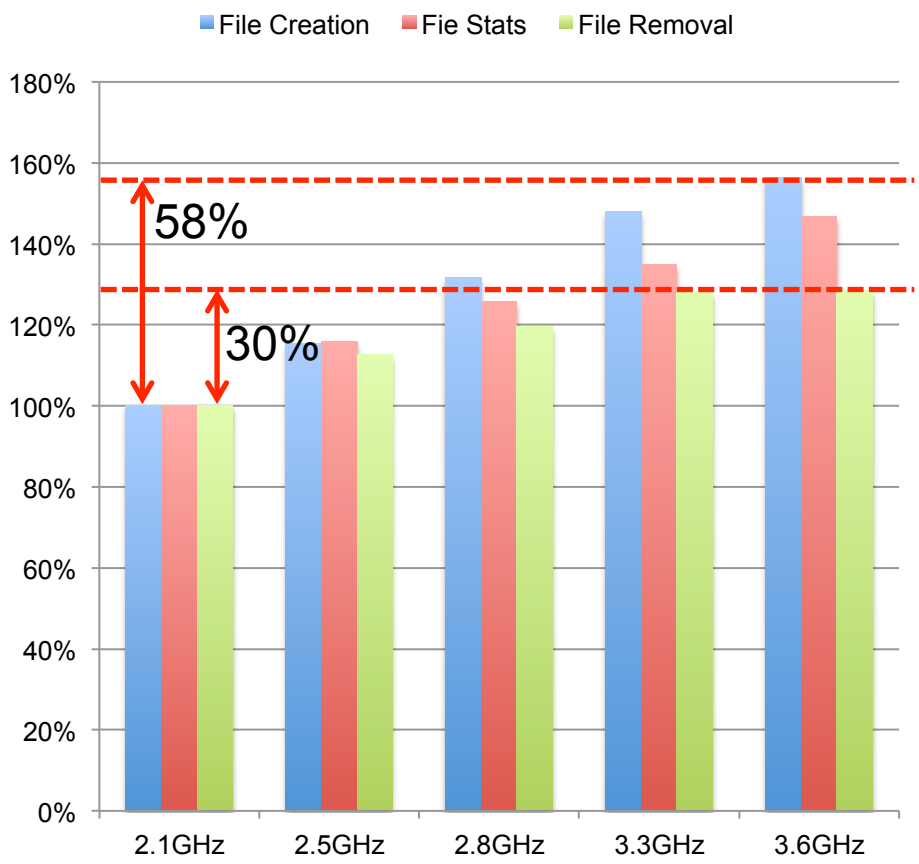
DataDirect NETWORKS
INFORMATION IN MOTION

- Metadata Performance comparison (Shared Directory)
  - 32 clients(1024 mount points), 1024 processes, 1.28M Files
  - Tested on 16 CPU cores with 2.1, 2.5, 2.8, 3.3 and 3.6GHz CPU Speed (MDS)

**Directory Operation(Shared)**

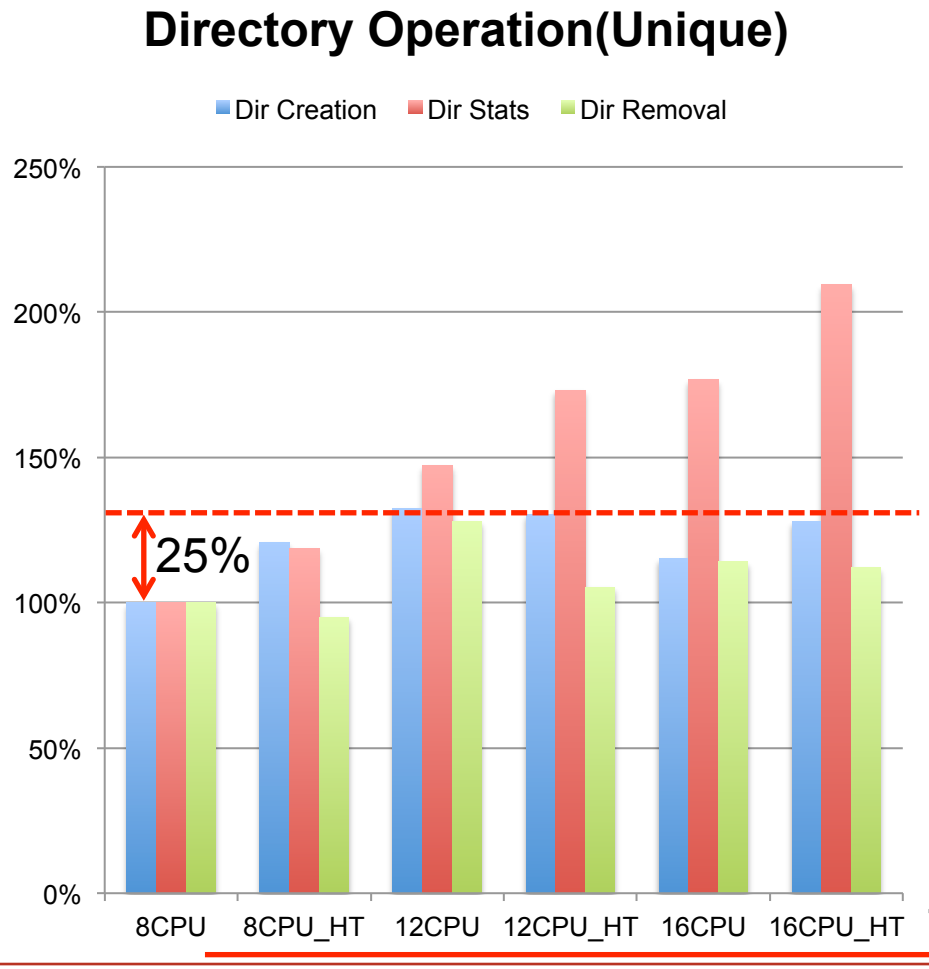


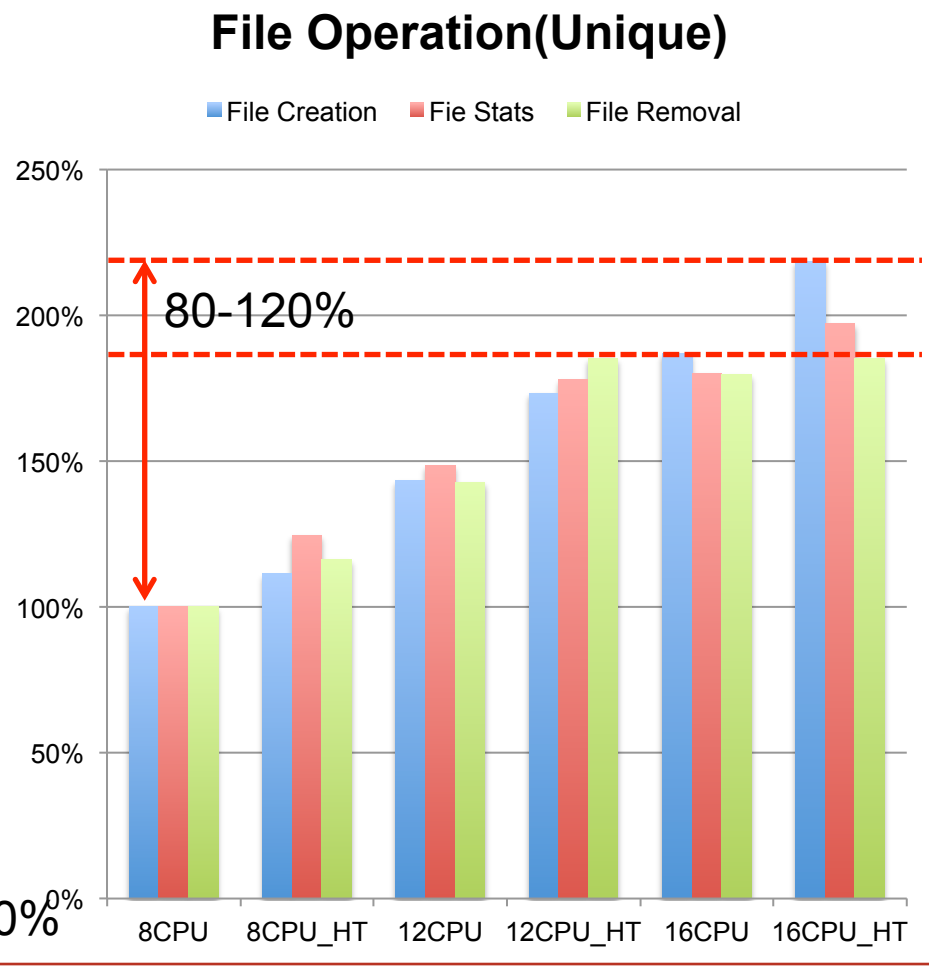

**File Operation(Shared)**



# Lustre Metadata Performance Impact MDS's CPU Cores

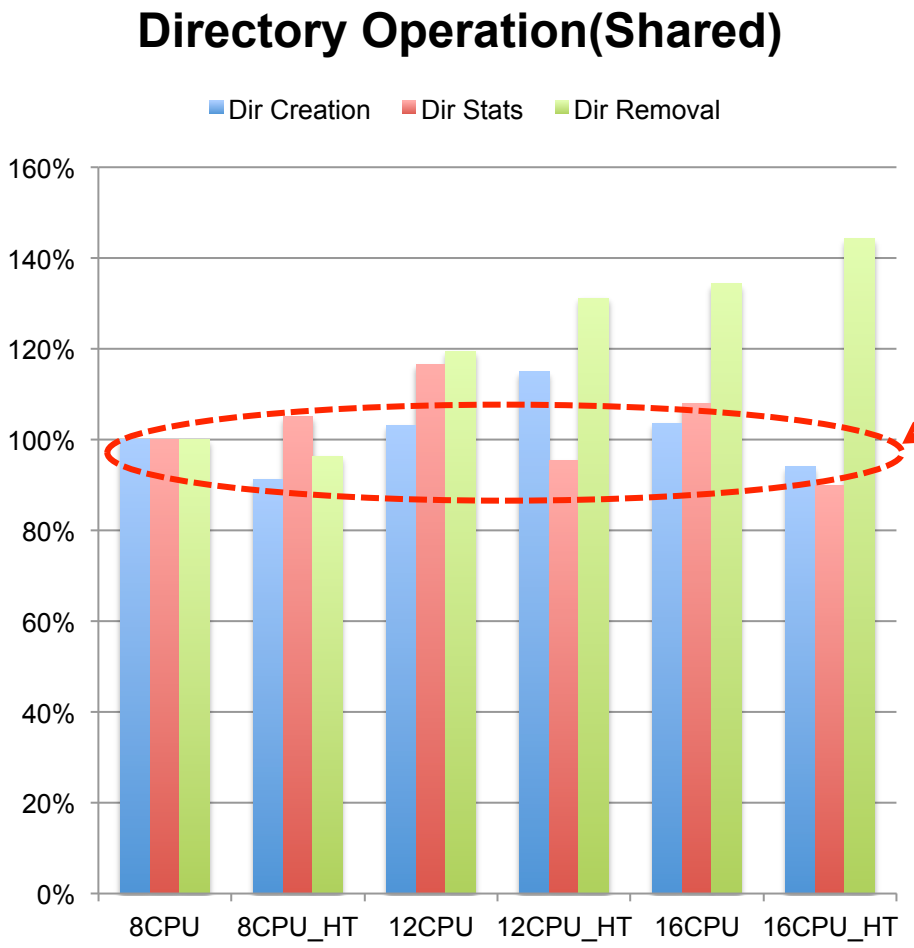
- Metadata Performance comparison (Unique Directory)
  - 32 clients(1024 mount points), 1024 processes, 1.28M Files
  - Tested on 3.3GHz CPU speed with 8, 12 and 16 CPU cores w/wo logical processors

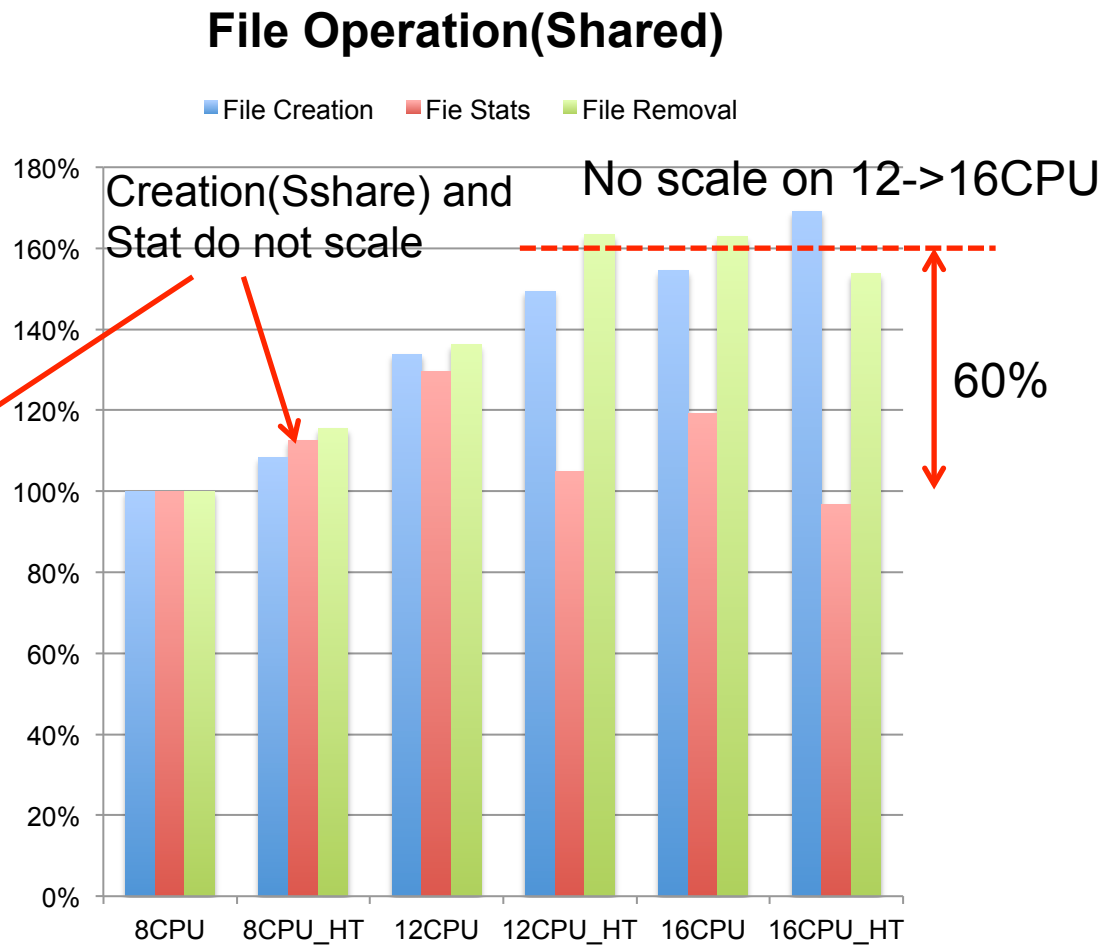


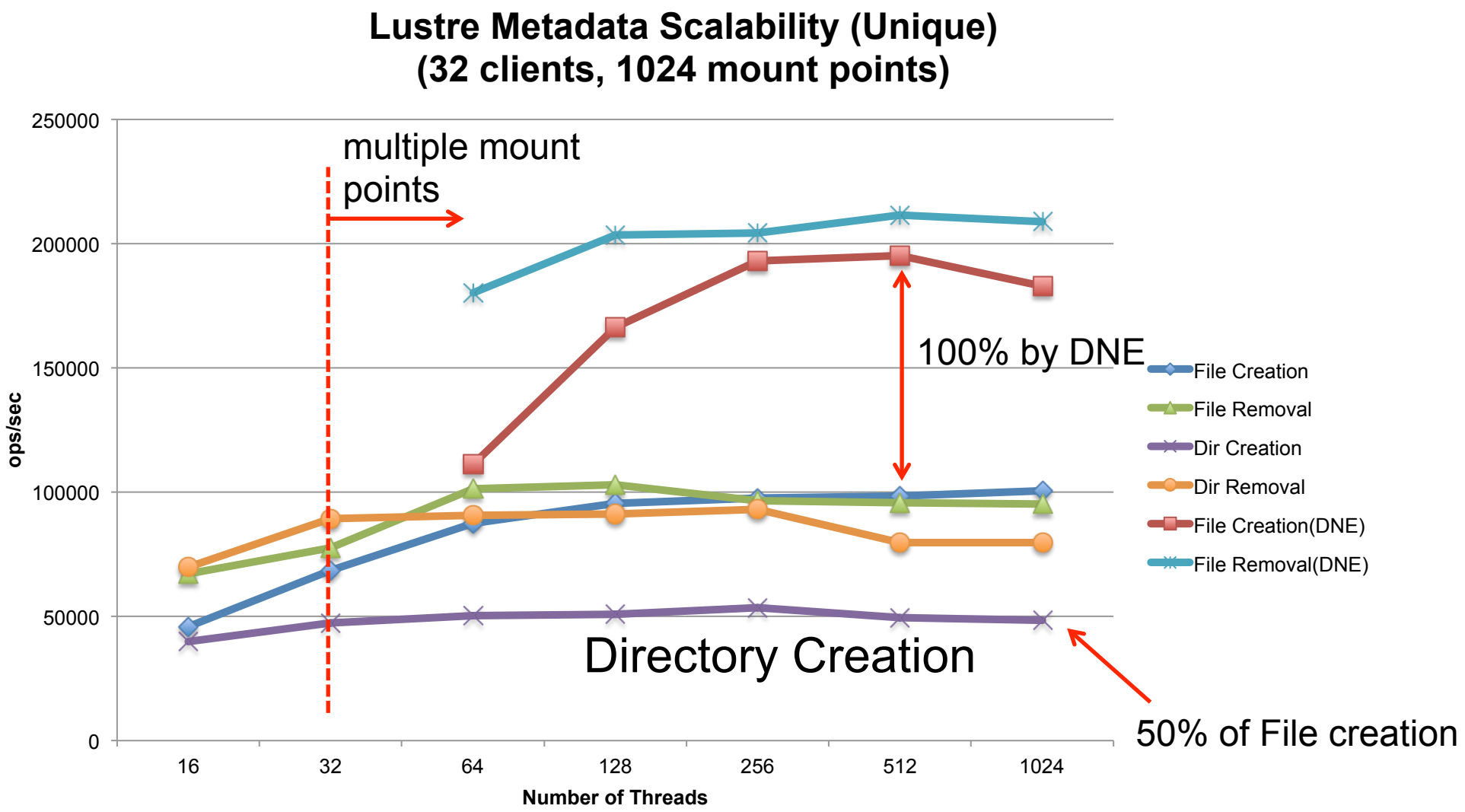
# Lustre Metadata Performance Impact MDS's CPU Cores

- Metadata Performance comparison (Shared Directory)
  - 32 clients(1024 mount points), 1024 processes, 1.28M Files
  - Tested on 3.3GHz CPU speed with 8, 12 and 16 CPU cores w/wo logical processors

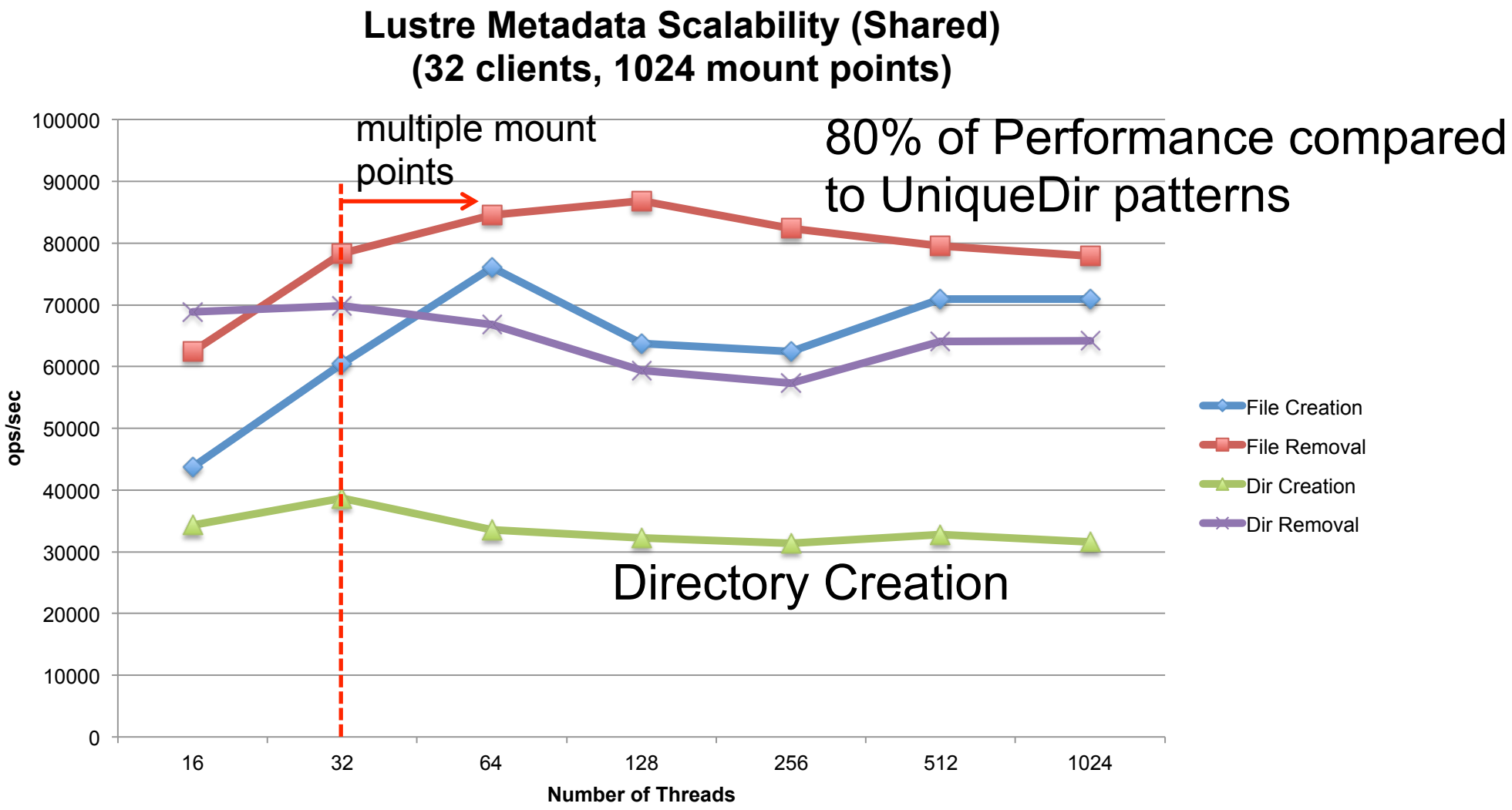


# Lustre Metadata Performance
# MDSs Scalability (Unique Directory)



# Lustre Metadata Performance
# MDSs Scalability (Shared Directory)

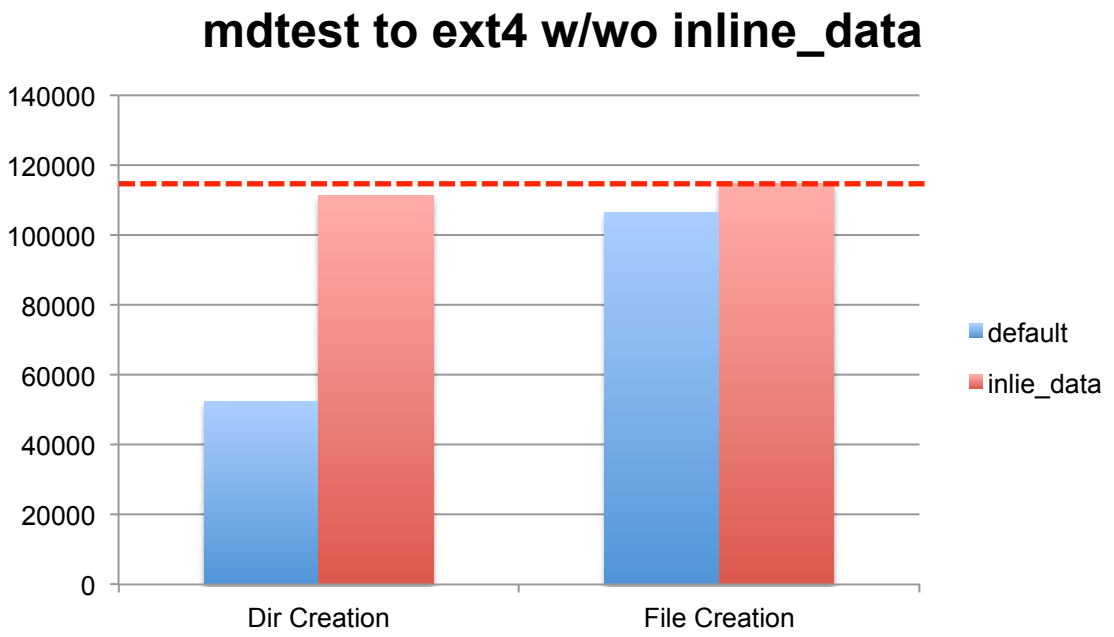


# Why Lustre directory creation is slower than File creation?

DataDirect NETWORKS
INFORMATION IN MOTION

```
ext4_mkdir()
-> ext_init_new_dir()
  ->ext4_try_create_inline_dir()
    -> ext4_append()
      -> ext4_getblk()
        -> ext4_map_blocks()
```

Average costs (inline_data vs default)
ext4_getblk() 1.4us vs 9.8us
ext4_map_block() 0.5us vs 4.8us



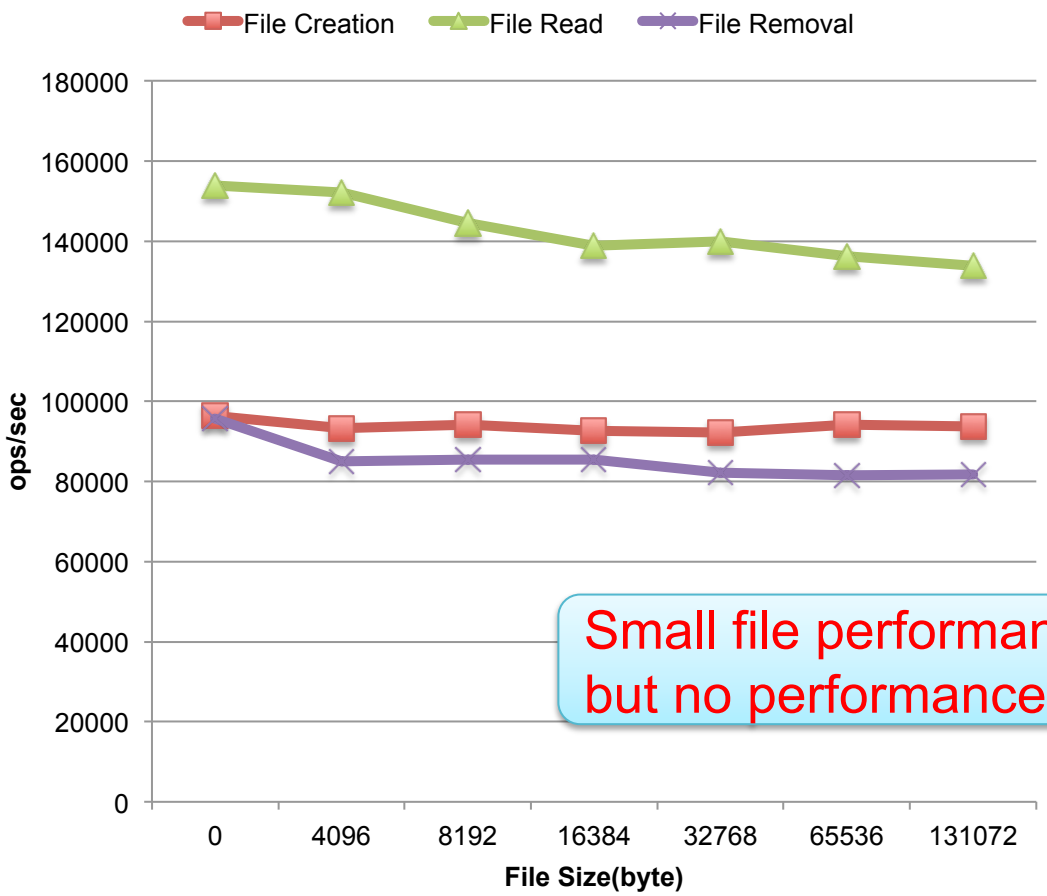

- "Inline_data" feature is available on newer kernel. RHEL7 also supports it.
- "Inline_data" is not available in ldiskfs since "dir_data" in ldiskfs and "inline_data" are incompatible, today.
- Similar idea might be good? Investigating on LU-5603

# Lustre Metadata Performance
# File creation and removal for small files

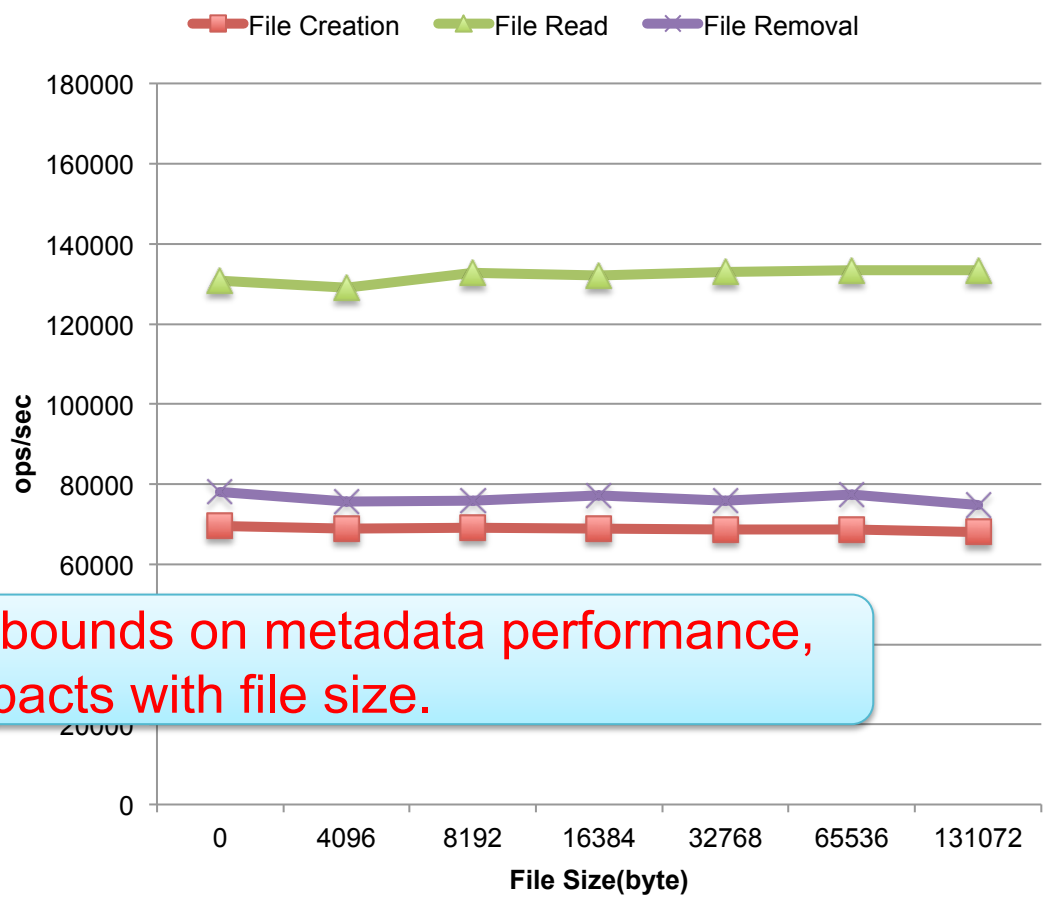
DataDirect NETWORKS
INFORMATION IN MOTION

Creating files with actual file size (4K, 8K, 16K, 32K, 64K and 128K)
(Stripe Count=1)

**Small File Performance (Unique Directory)**
**(32 clients, 1024 mount points)**



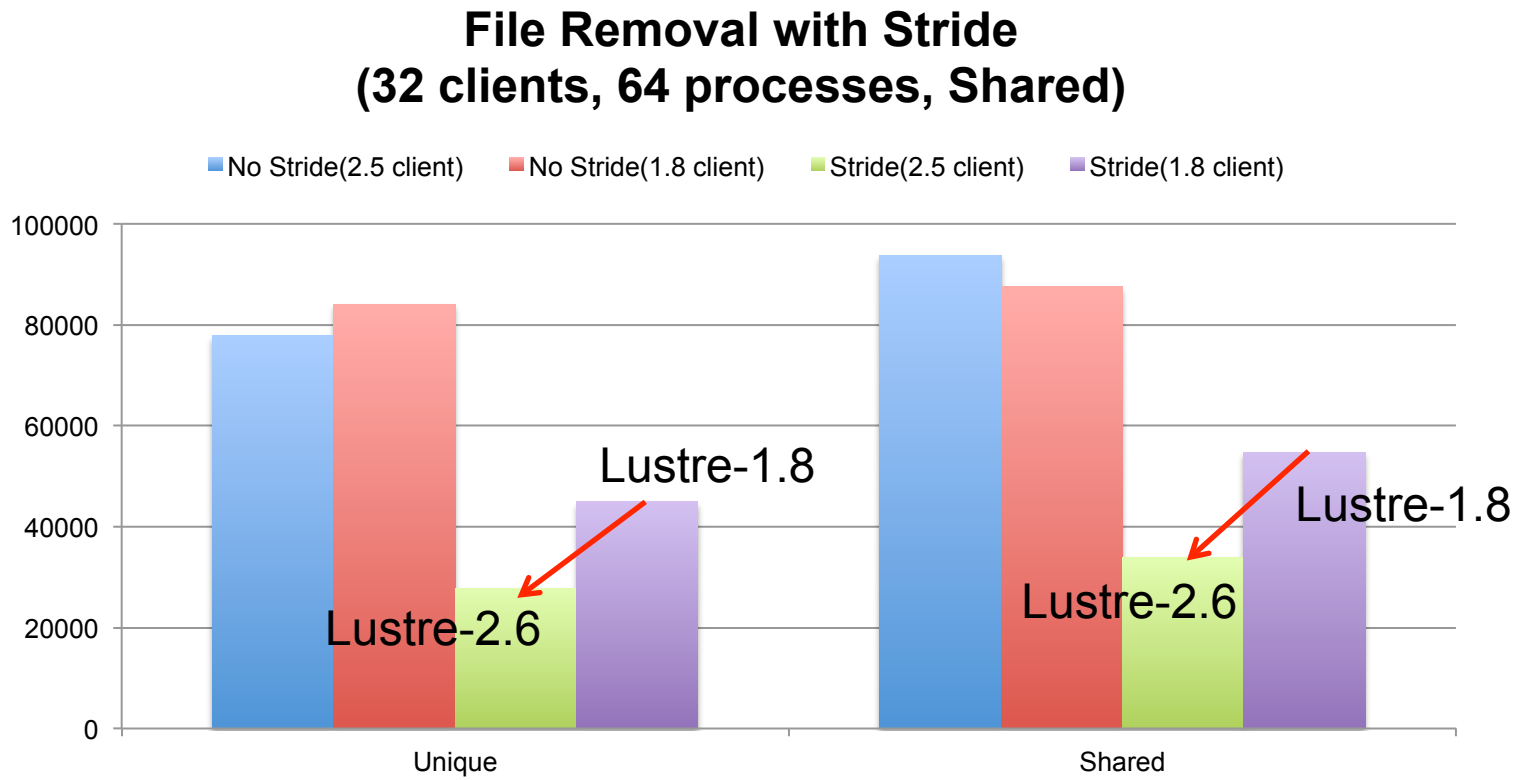

**Small File Performance (Shared Directory)**
**(32 clients, 1024 mount points)**




Small file performance bounds on metadata performance, but no performance impacts with file size.

# Lustre Metadata Performance
# Stride access pattern

- Stride ('-N' in mdtest) helps avoiding local locks in cache for stat() and unlink() operation after file creation.




- LU-5608 for regressions in 2.6 client for stride metadata access pattern.

# Summary Observations

- MDS Server resources significantly affect Lustre Metadata performance
  - Performance scales well by number of CPU core and CPU Speed in unique directory access, but not CPU bound for shared directory access pattern
  - Collected baseline results with 16 CPU cores, but need more tests on CPU cores
- Performance is highly dependent on metadata access pattern
  - Example: Directory Creation vs. File Creation
  - "Stride" option helps avoiding local locks in cache
  - With actual file size (instead of zero byte), less impact in the case of a small number of OST(e.g. up to 40 OST), but testing on large number of OSTs is needed

# Metadata Performance: Future Work

DataDirect NETWORKS
INFORMATION IN MOTION

- Known Issues and Optimizations
  - Client-side metadata optimization and especially single-client metadata performance
  - Various performance regressions in Lustre 2.5/2.6 that need to be addressed (e.g. LU5608)
- Areas of Future Investigation
  - Real-world metadata use scenarios and metadata problems
  - Real-world small-file performance (e.g. life sciences)
  - Impact of OST data structures on real world metadata performance
  - DNE scalability on very large systems with many MDSs/MDTs and many OSSs/OSTs