# Analysis of DNE I & II in the Latest Lustre* Releases

Adam Roe – HPC Solutions Architect

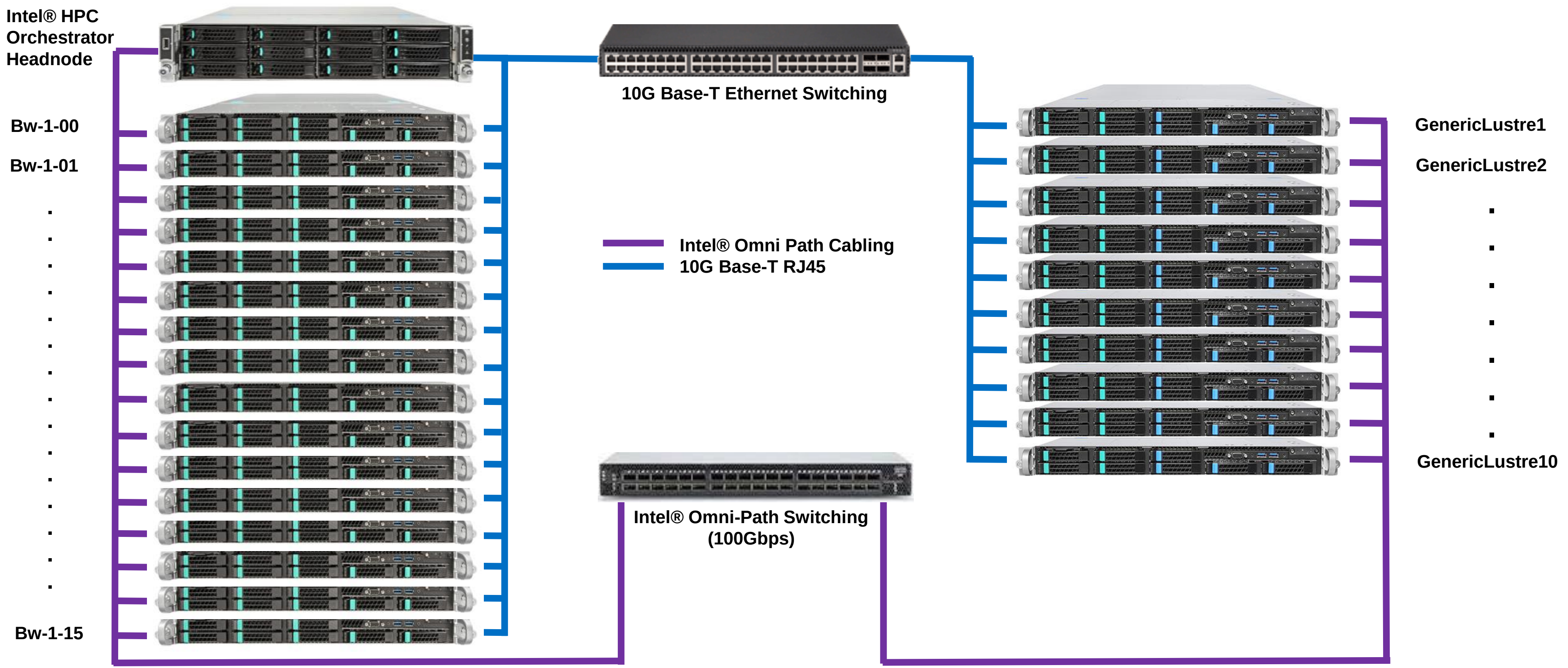
HPDD Technical Consulting Engineering

adam.j.roe@intel.com

# Objectives

- Preliminary Scaling tests of DNE I and DNE II on a ZFS* backend
  - How does DNE II behave with a ZFS backend; how many MDT's per server, once the peak has been reached does the performance scale linearly with server count?
- Small File Performance: Can I increase small file performance by with DNE Phase II?
  - How does leveraging multiple MDT's across multiple servers effect overall file system small file operations?
- Lustre: A new design model
  - Removing dedicated metadata server(s) and distributing metadata across OSS, is this a good idea and does it produce any tangible performance (or other) benefits?
- Does leveraging DNE Phase II with ZFS as a backend eliminate the bottlenecks we currently see with ZFS metadata performance?

# Testbed Architecture



# Testbed Architecture (Cont.)

## Server

- 10x Generic Lustre servers with two slightly different configurations
  - Each System comprises of:
    - 2x Intel® Xeon E5-2697v3 (Haswell) CPU's
    - 1x Intel® Omni-Path x16 HFI
    - 128GB DDR4 2133MHz Memory
    - Eight of the nodes contain - 4x Intel P3600 2.0TB 2.5" (U.2) NVMe devices, while the other two have 4x Intel® P3700 800GB 2.5" (U.2) NVMe devices
    - One node equipped with 2x Intel® S3700 400GB's for MGT
- 16x 2S Intel® Xeon E5v4 (Broadwell) Compute nodes
  - 1x Intel® HPC Orchestrator (Beta 2) Headnode
  - Hardware Components:
    - 2x Intel® Xeon E5-2697v4 (Broadwell) CPU's
    - 1x Intel® Omni-Path x16 HFI
    - 128GB DDR4 2400MHz Memory
    - Local boot SSD
- 100Gbps Intel® Omni-Path Fabric
  - None-blocking fabric with single switch design.
  - Server side optimisations: "options hfi1 sge_copy_mode=2 krcvqs=4 wss_threshold=70"
    - Improve generic RDMA performance on Lustre server side, generally you can be more aggressive with krcvqs on the server side

# Intel® SSD DC P3600 Series

nvm EXPRESS™ PCI EXPRESS® 3.0, x4

Low latency performance optimized for mixed workloads. Up to 6x the performance of a SATA SSD at half the latency and CPU utilization!

400GB 800GB 1.2TB 1.6TB 2TB

2.5inx15mm

x4 HHHL Add in card

## Performance

| | |
|---|---|
| Random 4k Read | Up to 450k IOPS |
| Random 4k Write | Up to 56k IOPS |
| Random 4k 70/30 R/W | Up to 160k IOPS |
| Sequential Read | Up to 2600 MB/s |
| Sequential Write | Up to 1700 MB/s |
| Avg Active R/W Power | 7-10/8-25 W |
| Idle Power | 4 W |

## Uses

- Virtualization
- Private Cloud
- Database
- HPC
- Caching and tiering

For additional information visit the PCIe Family Page on www.intel.com/ssd

## Features

Power loss protection

End-to-end data protection UBER $10^{-17}$, 2M hours MTBF

20nm HET NAND for mixed workload performance at 3 DWPD

Low latency performance of NVMe

# The Software Stack in more Detail

## Lustre Server Side: Master Build #3419

| | |
|---|---|
| Kernel: | 3.10.0-327.22.2.el7_lustre.x86_64 |
| Lustre Version: | lustre-2.8.56-1 |
| ZFS Version: | zfs-0.6.5.7-1 |
| e2fsprogs: | 1.42.13.wc5-7 |
| OS: | CentOS 7.2 – (yum updated to latest build as of 15th August 2016) |
| Network Stack: | CentOS In-kernel + Intel Fabric suite 10.1.1.0.9 |

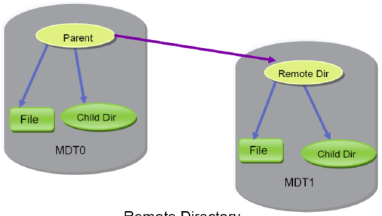
## Lustre Client Side:

| | |
|---|---|
| Kernel: | 3.10.0-327.22.2.el7.x86_64 |
| Lustre Version: | lustre-client-2.8.0 |
| Cluster Stack: | Intel® HPC Orchestrator Advanced (Beta2) |
| OS: | CentOS 7.2 – (yum updated to latest build) |
| Network Stack: | CentOS In-kernel + Intel Fabric suite 10.1.1.0.9 |

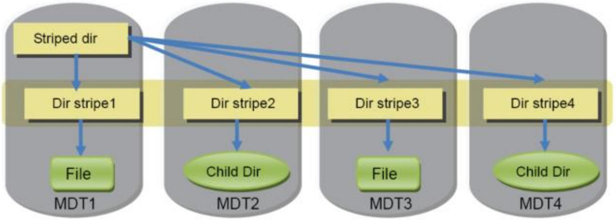
# Distributed NamespacE Phase I & II

I will make the assumption that we are all relatively familiar with the concept of DNE. DNE I, remote directories and DNE II, striped directories.

- The testing will mainly focus on DNE Phase II, since Phase I was been around for a number of years
- Phase I testing will be used to create a baseline only and use for comparative performance



Remote Directory



# Persistent Parameters across all testing

- All tests will use 256 processes (cores, HT disabled on clients) with 16 processes per client
- 10,000 items per tree node ( -I 10000) & a unique working directory for each task (-u), the point is to test “file system” scaling, single directories will be investigated later
- Despite the “-u” flag we still leverage DNE2
- No Lustre* specific tunings were made, all out of the box stock

# Scaling Testing

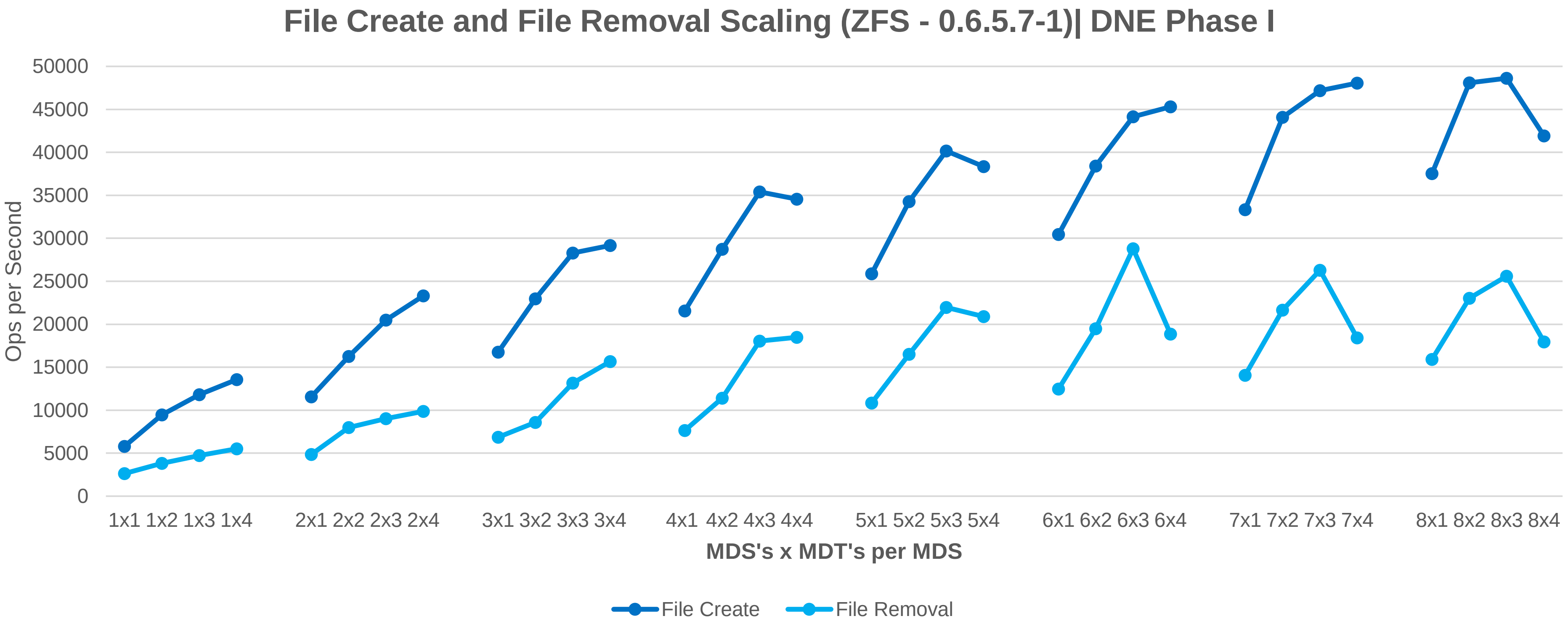
How does DNE I and DNE II Scale across up to 8 servers with 4 MDT’s per server with ZFS as a backend?

# Distributed NamespacE Phase I, scaling testing using MDTEST

## Using DNE Phase I to get a baseline of performance and comparative data to DNE Phase II.

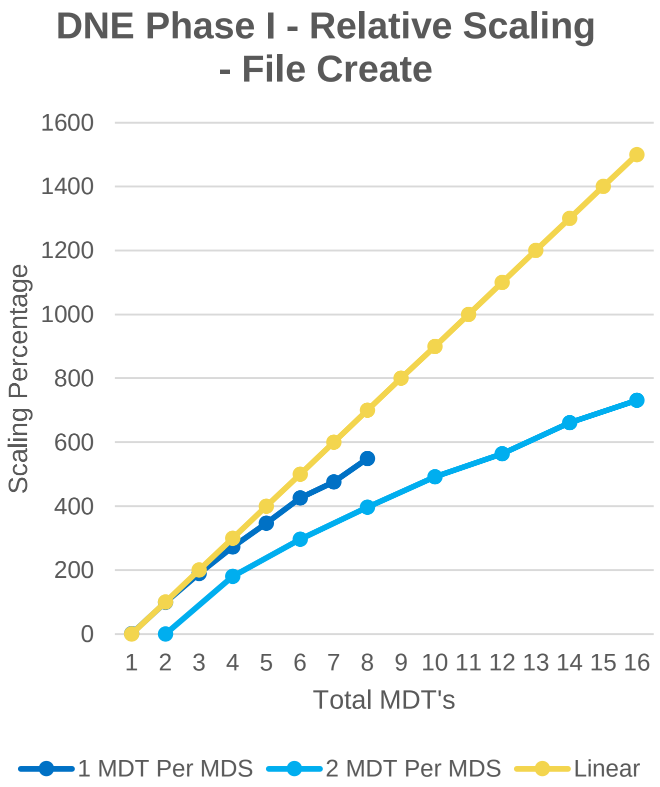
- Released and functional since Lustre 2.4
- Well integrated and plenty of data available out there to describe scaling and performance
- Use as proof point for the platform and for later comparisons

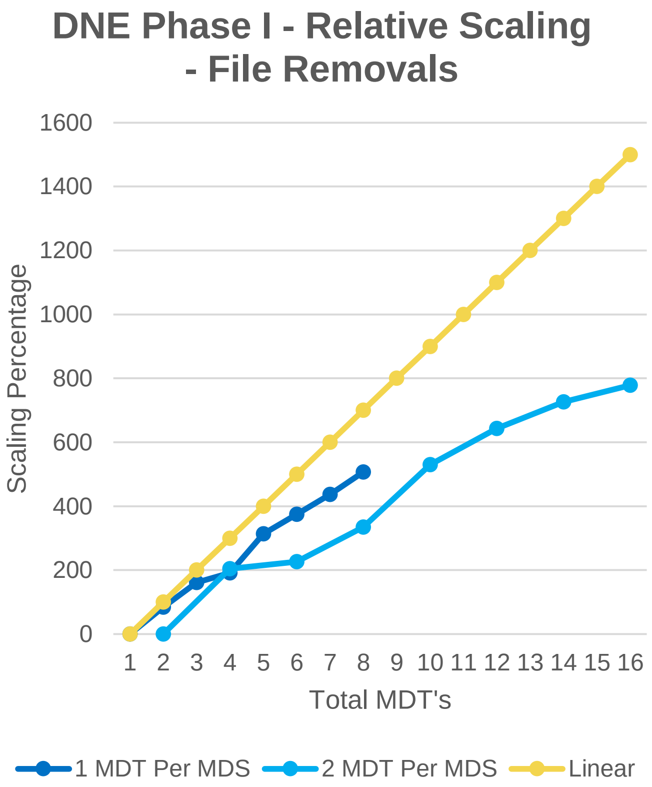
# DNE Phase I, ZFS



```
$MPICMD ./mdtest -i 3 -I 10000 -F -C -T -r -u -d /mnt_point/@/mnt/point2/@etc.
```

# DNE Phase I, ZFS – Relative Scaling





# Thoughts

Across the board, the scaling was generally pretty good, however most applications (like MDTEST does), do not allow for you to run in multiple directories.

- DNE I has to be managed, whereas DNE II is a sort of a deploy and forget configuration
- Scaling single user / single application performance isn’t possible in most cases
- Excluding the 1x and 2x server configurations, after 2x MDT’s per server performance was flat or worse
- Surprised to see such regression with 4x MDT per MDS as I increased the MDS count

# Distributed NamespacE Phase II, scaling testing using MDTEST

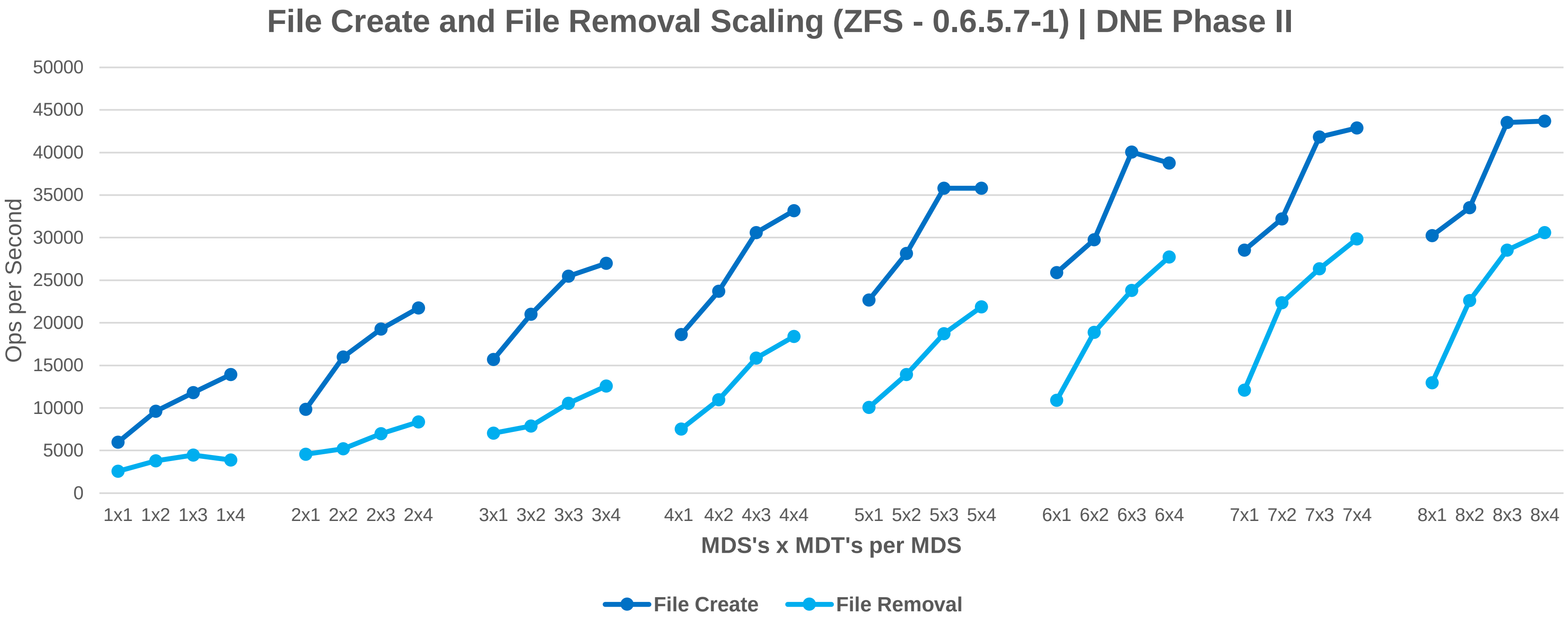
Scaling with DNE Phase II including some information on single directory scaling and how it benefits Lustre with small files.

- Launched as of Lustre 2.8
- Documentation is pretty limited and despite being launched stability is at times questionable
- The scope of “lfs setdirstripe” could be expanded:
  - Stripe multiple “**specific**” MDT’s, e.g. … “-c 0,4,8,12” instead of just striping a series of MDT’s, “-c 4”
  - https://jira.hpdd.intel.com/browse/LU-8616

# DNE Phase II Usage

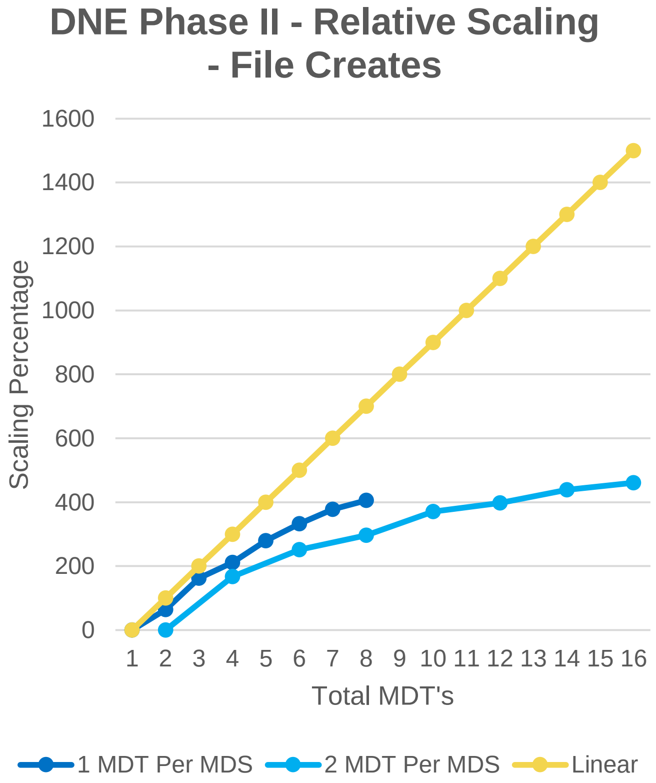
- lfs setdirstripe –D -i X -c X /mnt/zlfs2/DNE2_X
  - Where “-i” MDT start index
  - Where “-c” MDT stripe count
  - “-D” Set default stripe for all file and sub-directories
    - Critical in most instances else you lose the MDT round robin on the sub-directories
  - Even though MDTEST is using “-u”, by passing “-D” file allocation in that directory is round robin between the MDT’s

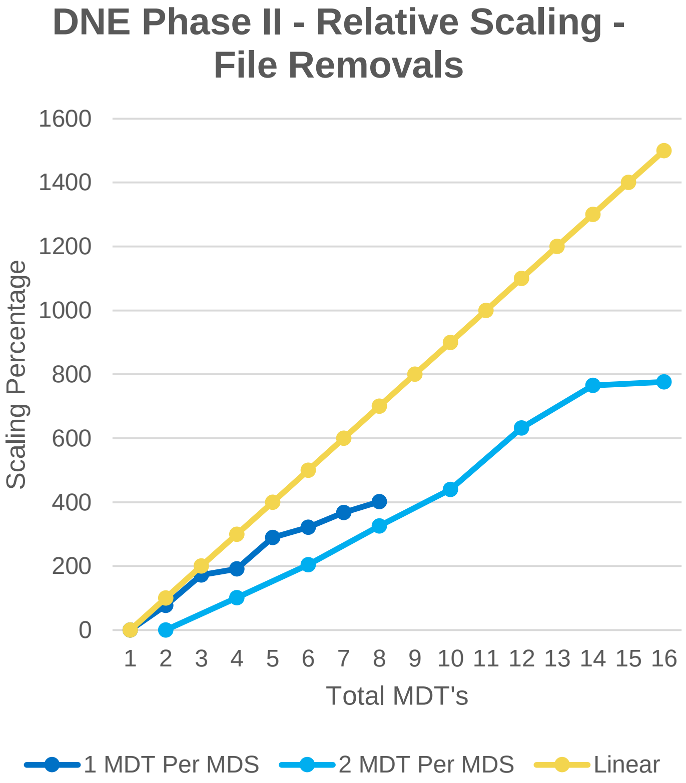
# DNE Phase II, ZFS



```
$MPICMD ./mdtest -i 3 -I 10000 -F -C -T -r -u -d /mnt_point/
```

# DNE Phase II, ZFS – Relative Scaling





# Thoughts

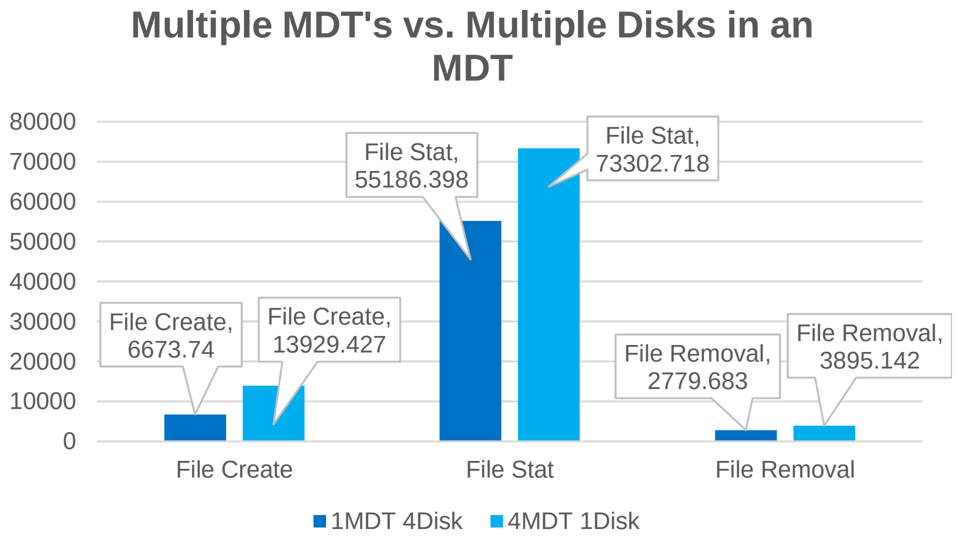
Overall I was very surprised how well DNEII + ZFS scaled at the single MDS level. Some observations:

- File Stat was all over the place – Data was not shareable. After 2x MDS's File Stat was pretty static and didn't scale much more
- Impressed with the scalability by multiple MDT's per MDS
- Except from one of the test cases (1x4) File Removal scaled well, in all configurations - I was very pleased with the results here

# ZFS Stripe vs. DNE II Stripe

I used ZFS to stripe 4x devices, I then compared this to 4x zpools each with one device using DNEII.

- With the current ZFS Implementation DNEII stripe is more effective (as shown in chart)
- This was consistent across servers, I.e. 8x Server 32MDT was better than 8x Server 8MDT (4x device per MDT)

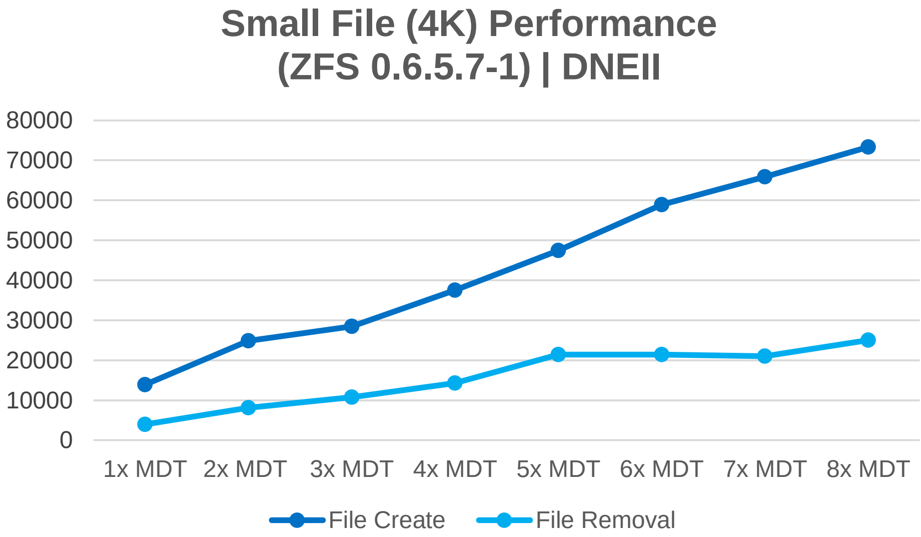


```
$MPICMD ./mdtest -i 3 -I 10000 -F -C -T -r -u -d /mnt_point/
```

# Small File Performance Scaling with DNE II

MDTEST: 4k file operations, scaling between 1 to 8 MDS’s with a single MDT per MDS.

- File create scales pretty well, approximately, scaling at nearly 96%
- File removal goes flat after 5x MDT’s, with gains after being marginal
- Not included here, but file stat scaled quite well – however the standard deviation was too high +/- 30%



```
$MPICMD ./mdtest -i 3 -I 10000 -z 1 -b 1 -L -u -F –w 4096 -d /out/DIR
```

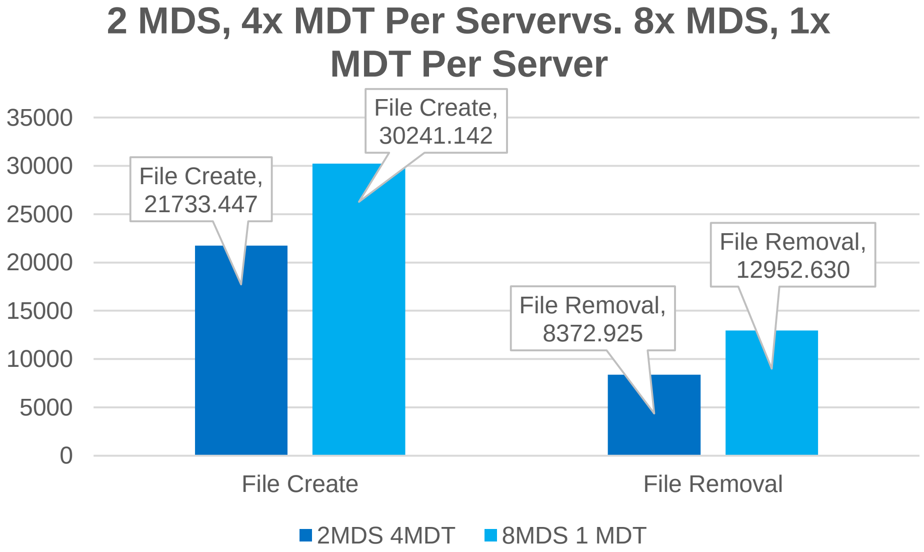
# Lustre a New Design Model

Now with DNE II, why do we need dedicated metadata servers? What performance impact do we see collocating the MDT’s with OST’s, i.e. 1x MDT per OSS (In a none HA configuration).

# Distributing MDT's across more servers?

Comparison between what you would see in a traditional setup (2x MDS's) to say adding those disks to all the OSS's.

- It is clear that there is more performance spreading the MDT's across more servers as appose to having more MDT's in a single server
- 32% better on Creates and 42% better on Removals



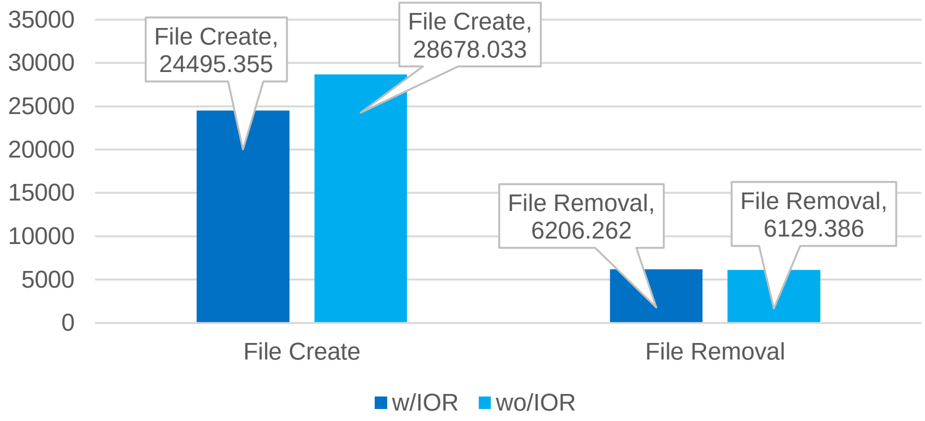
* Test bed details on Slides 3 & 4

```
$MPICMD ./mdtest -i 3 -I 10000 -F -C -T -r -u -d /mnt_point/
```

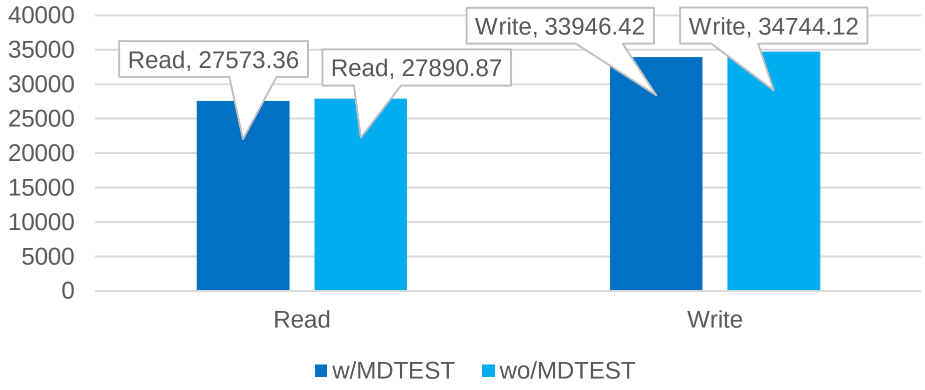
# Comparison of both

- MDTEST Run using DNE2 across all MDT's while running an IOR test across the same filesystem at the same time
- A separate run with one benchmark at a time, to see the performance degradation running on the same server
- Questions:
  - How negligible is the performance impact (throughout) to the IOR run while running MDTEST at the same time?
  - Will the MDTEST performance when collocated on the OST's be good enough to exceed the 2x4 configuration?

## MDTEST - Collocation | With / Without IOR



## IOR - Collocation | With / Without MDTEST



```
$MPICMD ./IOR -wr -C -F -i 3 -t 1m -b 1m -s 32768 -a MPIIO -o /mnt_point/
$MPICMD ./mdtest -i 3 -I 10000 -F -C -T -r -u -d /mnt_point/
```

# Thoughts

Pleased with the overall performance, the only concern is the engineering scope required for production use.

- So long as the performance when collocating the MDT’s and OST’s does not drop below the tests with a standard configuration, I think this is a feasible architecture
- MDTEST results were a little lower than expected, but still within a reasonable margin
- From an engineering perspective is this even a feasible configuration in production?
  - What effort is required to overcome issues such as recovery?
- With this architecture that would mean at least one servers must have: an MGT, MDT and OST… is this a good idea?

# Thoughts (Cont.)

## A use case for such a solution:

- Smaller 2 to 6 OSS systems where adding dedicated metadata servers makes Lustre less competitive against other file systems
- Density, network resources (switch ports), cost, power, cooling – are all factors
- The capability to make a Small Form Factor solution even more competitive against alternatives such as HA NAS
  - http://www.intel.com/content/dam/www/public/us/en/documents/white-papers/optimized-entry-level-lustre-solution-white-paper.pdf
- With most ZFS deployments the recommended 9+2 RAIDz2 configuration usually leaves space in the JBOD

# ZFS 0.7.0-rc1 + Lustre Master

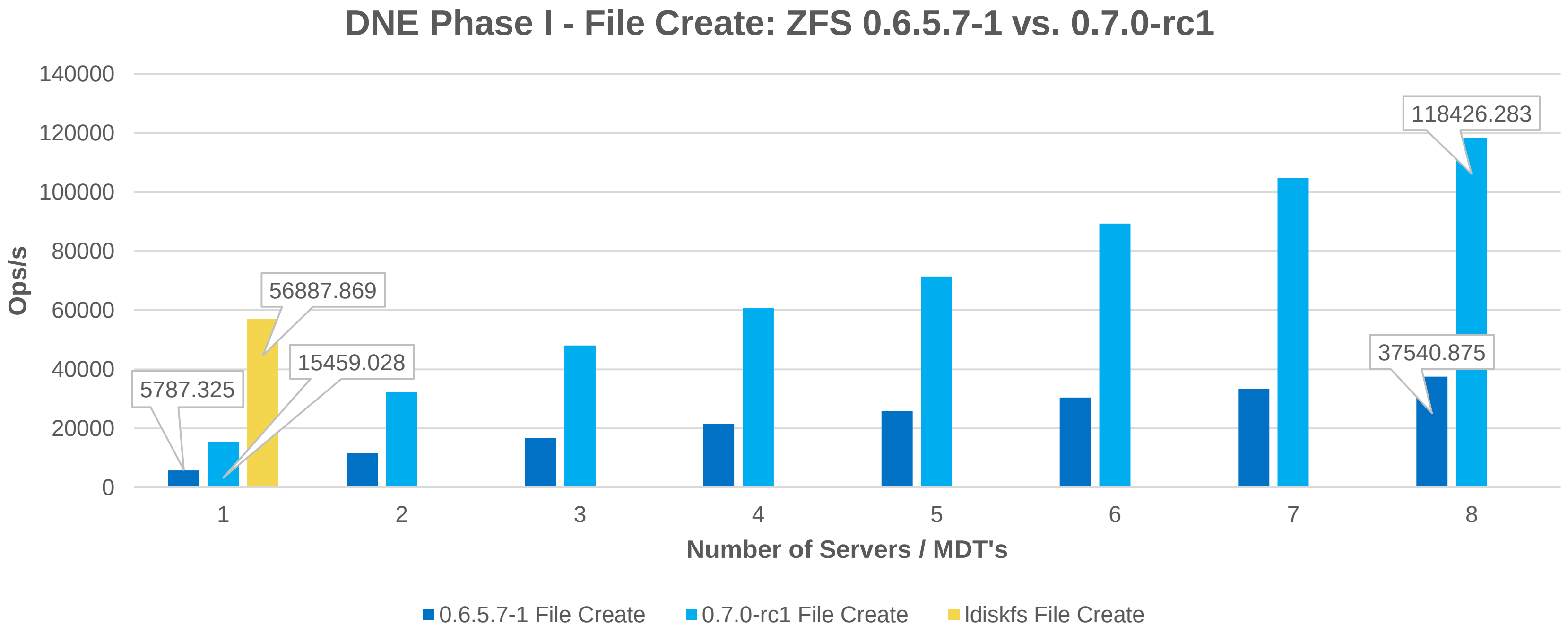
Lets test the landed ZFS metadata patches and the latest Lustre master release to see how much of a difference it has made.

# Set up

Some of the metadata performance improvement patches have landed in ZFS master, how well do they perform with Lustre?

- ZFS 0.7.0-rc1
- Lustre Master (as of 16/09/2016) b3442 + LU-8619
- Hardware Configuration has not changed, as of slide 4 & 5
- zpool configuration and command line remain unchanged (same scripts as before)
- Testing DNE Phase I only, to get indicative scaling numbers
- No stabilisation or integration testing done with ZFS 0.7.0-rc1 and Lustre as of yet

# DNE Phase I ZFS 0.7.0-rc1: Indicative Scaling Tests



* Test bed details on Slides 3 & 4

$MPICMD ./mdtest -i 3 -I 10000 -F -C -T -r -u -d /mnt_point/@/mnt/point2/@etc.

# Thoughts

- 2.8x to 3x Performance scaling going from ZFS 0.6.5-1 to ZFS 0.7.0-rc1
- The LDISKFS numbers are higher than you would achieve with a traditional MDT
- The performance cap on ZFS is a limitation of the code not the hardware
  - Expect to achieve similar results on lesser hardware
- ZFS and LDISKFS performance will be closer with traditional MDT’s (SAS 15k / SAS SSDs)
  - ZFS Stripe vs. DNE II Stripe Slide; virtually no performance uplift adding more devices to a pool

# Summary

Why was there no LDISKFS? Is ZFS end to end in production viable? Is the scaling enough?

# Why isn’t there any LDISKFS Numbers?

In short, I wasn’t able to get enough stability using DNE Phase II and LDISKFS as a backend.

- In my testing ZFS was near enough totally stable with DNE Phase II
- The high performing NVMe devices + the higher performing backend (LDISKFS) seem to exaggerate the occurrence of issues when using DNE Phase II – Phase I was stable
  - https://jira.hpdd.intel.com/browse/LU-8581 - “In osd xattr cache implementation several list operations didn't consider RCU concurrency.”
  - ~~https://jira.hpdd.intel.com/browse/LU-8508 - Other issue, not DNE Related but is a blocker~~

# Scaling and Reliability

## Overall how did DNE Phase II scale and did it work?

- Generally scaling was not too bad
  - Scaling was no where near what you see with OST’s, but this is early days of DNE Phase II
  - To get the best ROI on your hardware 2x MDT’s per MDS is enough
  - MDS scaling was relatively predictable, 3x MDS’s scaled near linearly for the single MDT tests, with the relative increase reducing steadily
- Would I use it in production right now?
  - Yes I do… twice. The instability I have been seeing is primarily related to high MDT ops so for me it isn’t such an issue as both instances are using ZFS 0.6.5.7-1 as a backend
  - This may have to be revisited once the ZFS metadata patches land

# ZFS in Production with DNE Phase II

Does the current ZFS implementation leveraging DNE Phase II make end to end ZFS deployments in production viable?

- With the current implementation (0.6.5.7-1), yes it does, but if you are willing to pay for the extra hardware for future performance
- With the upcoming ZFS Patches to improve metadata performance, why wouldn't you want to? The benefits of a COW backend file system far out weight any issues I can think of
  - ARC/L2ARC support and in the future ZIL
  - Snapshotting and Compression capabilities
  - RAIDz2 is faster than RAID6
  - Lower hardware BOM cost.

# Questions and Future Work

- Further testing with new ZFS metadata patches to see how scalability is effected
- Testing with LDISKFS, unfortunately unforeseen circumstances did not allow for me to test this on the current platform

**With Thanks**

Cyndi Peach, Peter Jones, Andreas Dilger, James Vaughn intel®