outthink limits

News from Research

Sven Oehme - oehmes@us.ibm.com

SC16 Salt Lake City, Utah | hpc matters.

Agenda

- **Performance engineering matters**
  - **Disk drive engineering – how fast is a drive really ?**
  - **How to get data from storage to consumer – Network overhaul**
  - **More than 32 Sub blocks, why and what can we expect from them**
- Spectrum Scale with NVMe
- Spectrum Scale with NVMeoF
- ESS GF-2 first Performance results
- IOPS - does it actually mean anything ??
- Docker – Dean Hildebrand

Performance engineering matters

Imagine you need to deliver the following goals :

- 2.5 TB/sec single stream IOR as requested from ORNL
- 1 TB/sec 1MB sequential read/write as stated in CORAL RFP
- Single Node 16 GB/sec sequential read/write as requested from ORNL
- 50k creates/sec per shared directory as stated in CORAL RFP
- 2.6 Million 32k file creates/sec as requested from ORNL

What innovations in Storage would that require ?

Lets start with how fast is a disk drive ?

If you ask the Disk Vendor – 150 MB/sec per drive

If you ask you Block Storage Seller – 100 MB/sec per drive

If you ask an application Person – always to slow

If you ask a HPC admin – it depends

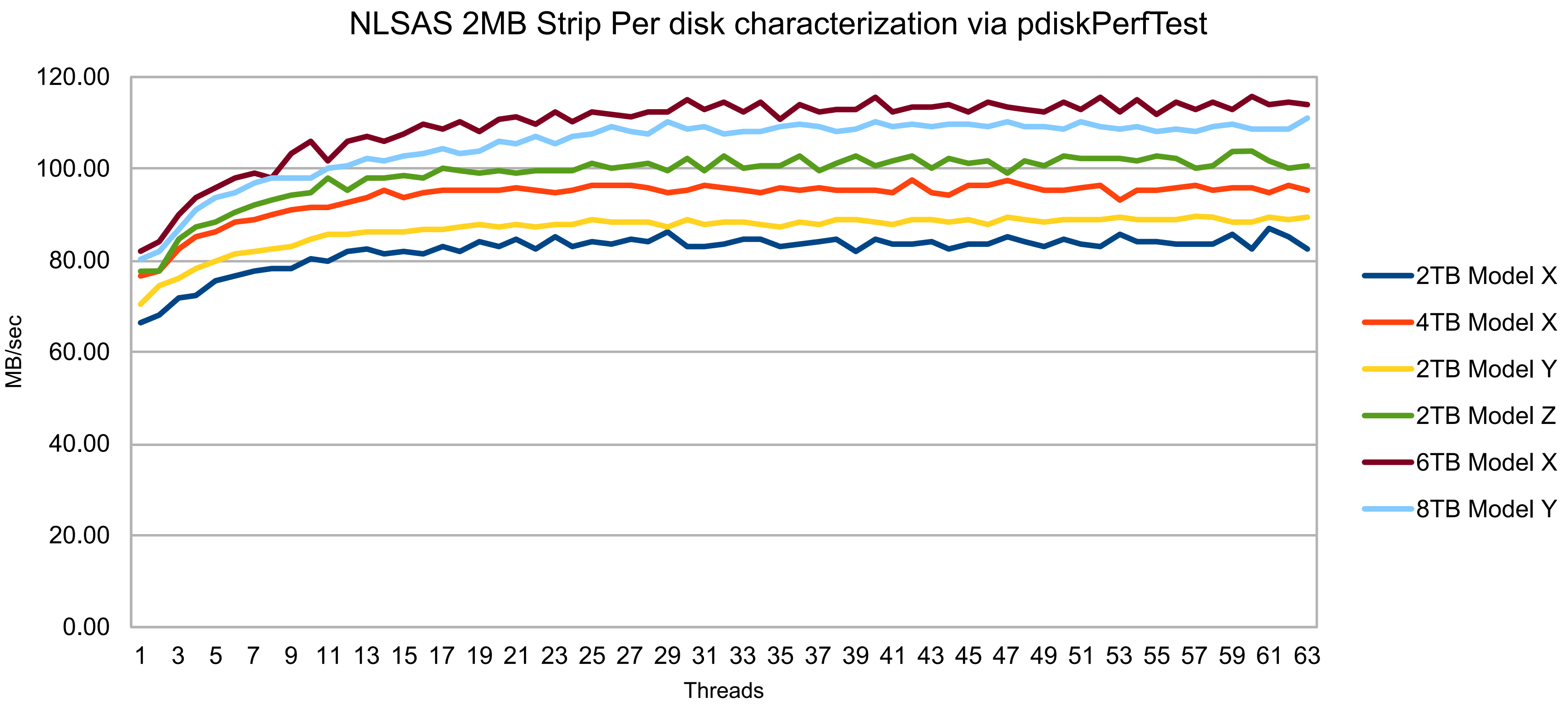
So who is right and how fast are they really ?

So what Influences total system speed

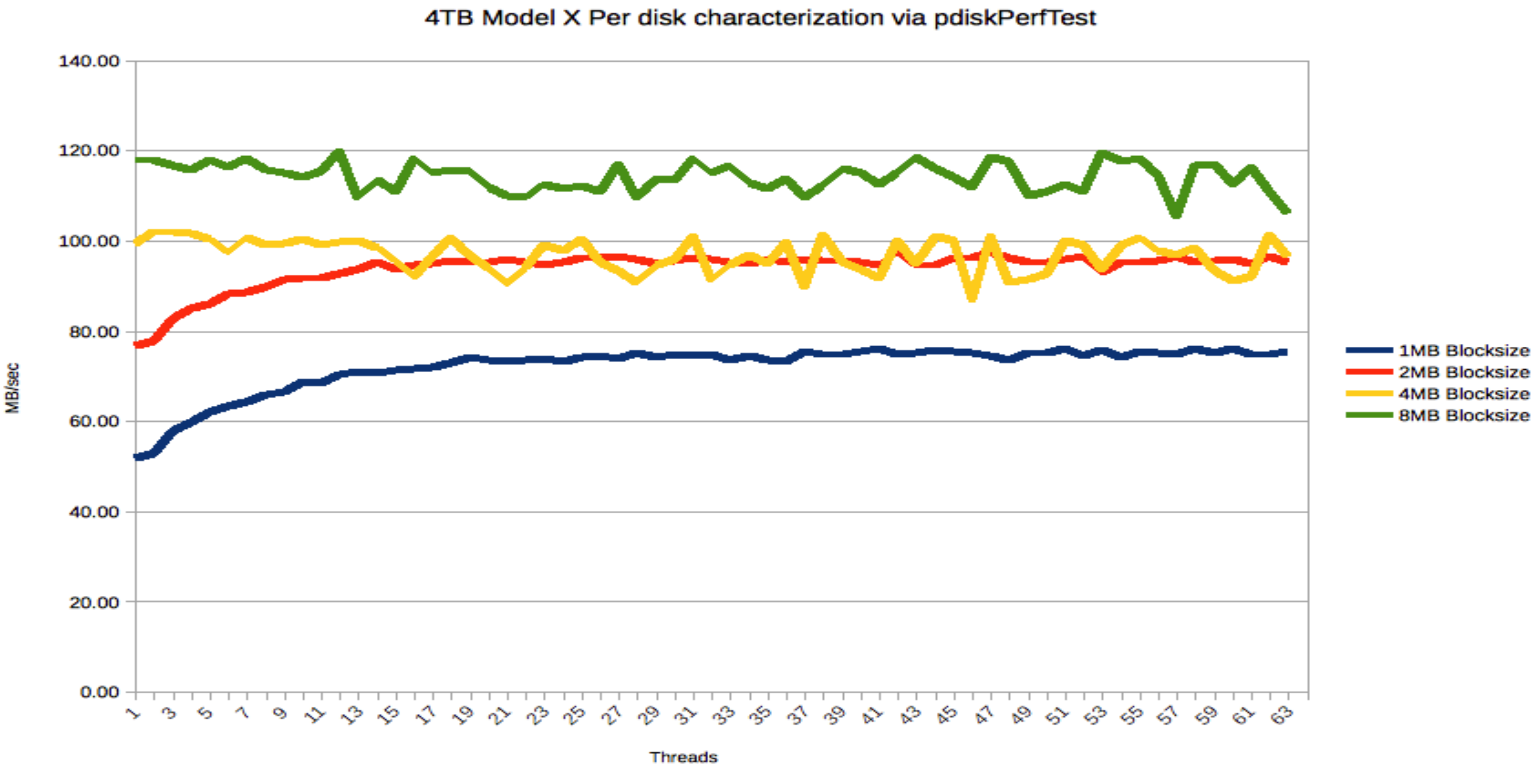
- Filesystem Overhead - who talks to block storage these days ?
- Controller Overhead - SW vs HW and how good is your raid implementation ?
- Raid mode Overhead - that's a simple math problem 1P vs 2P vs 3P ..
- Cache efficiency - complex , main issue is what context is that i/o performed
- Application access Patterns - random vs sequential
- Access Pattern the disks sees - you think its sequential, you are most likely wrong

So how fast perform disks under a large scale filesystem and what can one expect ?

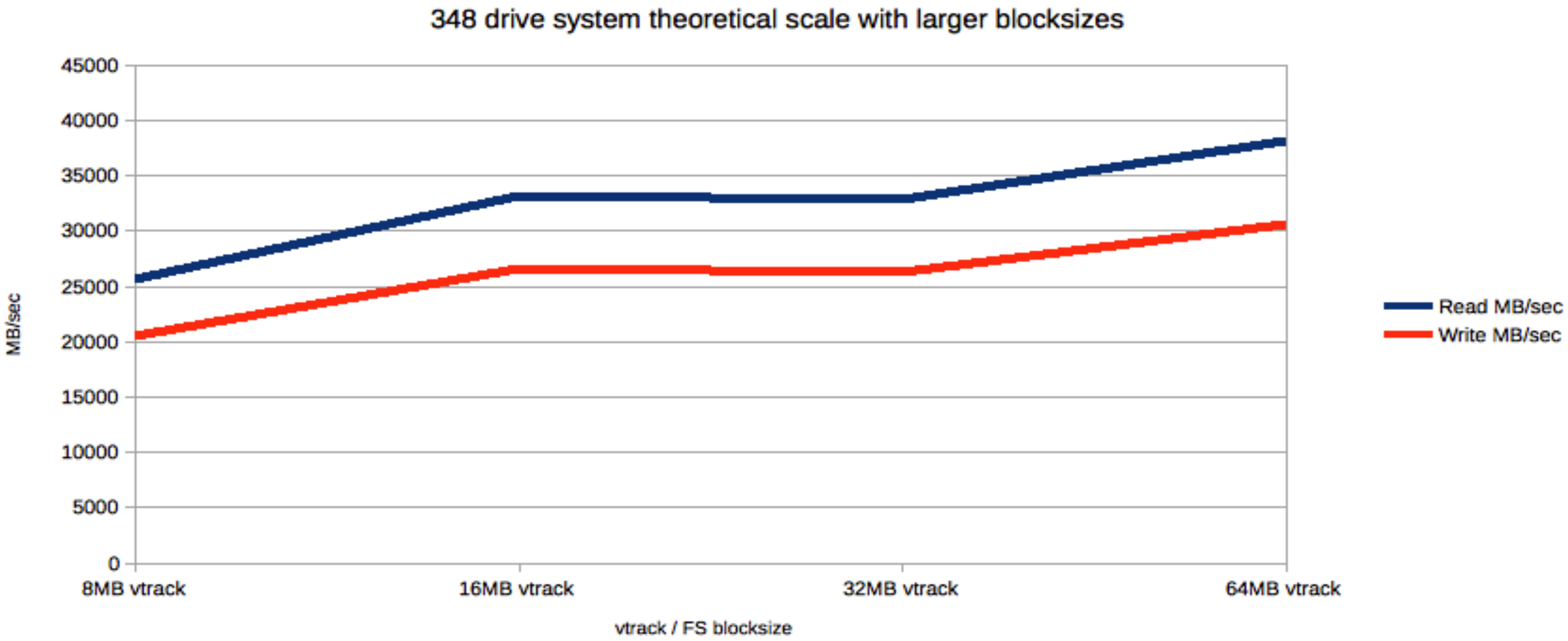
First - all access is random , lets take a look on how different models perform



Closer look at the 4TB Model with different i/o sizes



How would performance change with different Block sizes



You need to overhaul the Network communication – 4.2.1+

- Why do we need it ?
  - Keep up with the io(not capacity) density of bleeding edge Storage technology (NVMe, etc.)
  - Leverage advances in latest Network Technology (100GE/IB)
  - Single Node NSD Server 'Scale-up' limitation
  - NUMA is the norm in modern systems, no longer the exception

- What do we need to do ?
  - Implement an (almost) lock free communication code in all performance critical code path
  - Make communication code as well as other critical areas of the code NUMA aware
  - Add 'always on' instrumentation for performance critical data, don't try to add it later or design for 'occasional' collection when needed

That's what we did in 4.2.1 but there is more to come

Single client throughput enhancements

- 16 GB/sec single Node !

```
[root@p8n06 ~]# tsqosperf write seq -n 200g -r 16m -th 16 /ibm/fs2-16m-06/shared/testfile -fsync
tsqosperf write seq /ibm/fs2-16m-06/shared/testfile
  recSize 16M nBytes 200G fileSize 200G
  nProcesses 1 nThreadsPerProcess 16
  file cache flushed before test
  not using direct I/O
  offsets accessed will cycle through the same file segment
  not using shared memory buffer
  not releasing byte-range token after open
  fsync at end of test
    Data rate was 16124635.71 Kbytes/sec, thread utilization 0.938, bytesTransferred 214748364800
```

Single thread small i/o

- 50-80 usec per
- small Data RPC

```
[root@client01 mpi]# tsqosperf read seq  -n 1m -r 1k -th 1 -dio /ibm/fs2-1m-07/test
tsqosperf read seq /ibm/fs2-1m-07/test
  recSize 1K nBytes 1M fileSize 1G
  nProcesses 1 nThreadsPerProcess 1
  file cache flushed before test
  using direct I/O
  offsets accessed will cycle through the same file segment
  not using shared memory buffer
  not releasing byte-range token after open
    Data rate was 12904.76 Kbytes/sec, thread utilization 0.998, bytesTransferred 1048576
[root@client01 mpi]# mmfsadm dump iohist |less

I/O history:
```

I/O start time	RW	Buf type	disk:sectorNum	nSec	time ms	tag1	tag2	Disk UID	typ	NSD node	context	thread
09:26:46.387129	R	data	1:292536326	2	0.081	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387234	R	data	1:292536328	2	0.075	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387333	R	data	1:292536330	2	0.057	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387413	R	data	1:292536332	2	0.057	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387493	R	data	1:292536334	2	0.059	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387576	R	data	1:292536336	2	0.063	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387663	R	data	1:292536338	2	0.059	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387746	R	data	1:292536340	2	0.054	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387824	R	data	1:292536342	2	0.054	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread
09:26:46.387901	R	data	1:292536344	2	0.065	8755200	0	C0A70D06:571A90C4	cli	192.167.20.125	MBHandler	DioHandlerThread

More than 32 Sub blocks - why and what to expect ?

- Why do we have Sub blocks ?
  - Allow finer grained allocation – no space wasted
  - Allows coalescing of small files in larger blocks – raid friendly
- What Options do we have today ?
  - We can store data in inode (default <4k)
  - We can allocate a Sub block (1/32th of a Full block)
  - We support 64 KB, 128 KB, 256 KB, 512 KB, 1 MB, 2 MB, 4 MB, 8 MB and 16 MB block size today
- What's wrong with it ?
  - You have to choose between waste space for small files (>4k and <1/32th of block size) or bandwidth
  - You can never ever change it online, filesystem migration required
  - It has a significant performance penalty for small files in large block size filesystems
- So how do we fix it and what will it change ?

Best way to find out – measure it – 16 MB block size filesystem - mdtest

4.2.1 base code - SUMMARY: (of 3 iterations)

Operation		Max	Min	Mean	Std Dev
File creation	:	2296.791	2197.553	2237.644	42.695
File stat	:	3402913.848	3383139.838	3390622.546	8759.559
File read	:	452144.282	383467.565	426670.673	30712.367
File removal	:	202219.699	88486.720	160499.542	51134.019
Tree creation	:	9425.078	3138.312	6945.652	2732.932
Tree removal	:	6710.394	3063.299	5196.237	1551.879

zero-end-of-file-padding (4.2.2 + ifdef for zero padding): SUMMARY: (of 3 iterations)

Operation		Max	Min	Mean	Std Dev
File creation	:	13053.701	12570.060	12866.842	212.194
File stat	:	4077992.847	3291830.765	3600173.039	342592.742
File read	:	450592.091	408552.363	424759.494	18462.970
File removal	:	105876.511	93884.369	99224.908	4982.772
Tree creation	:	8451.948	1936.832	4123.063	3061.035
Tree removal	:	535.050	154.181	363.642	157.800

more sub blocks per block (4.2.2 + morethan32subblock code):
SUMMARY: (of 3 iterations)

Operation		Max	Min	Mean	Std Dev
File creation	:	51397.549	33005.542	40316.721	7967.608
File stat	:	3326016.821	3195765.701	3277674.290	58231.427
File read	:	616434.716	543430.803	568013.424	34240.371
File removal	:	134732.546	48867.351	86175.005	35945.588
Tree creation	:	7771.893	1039.578	3648.852	2949.535
Tree removal	:	2879.694	550.493	1859.348	972.530

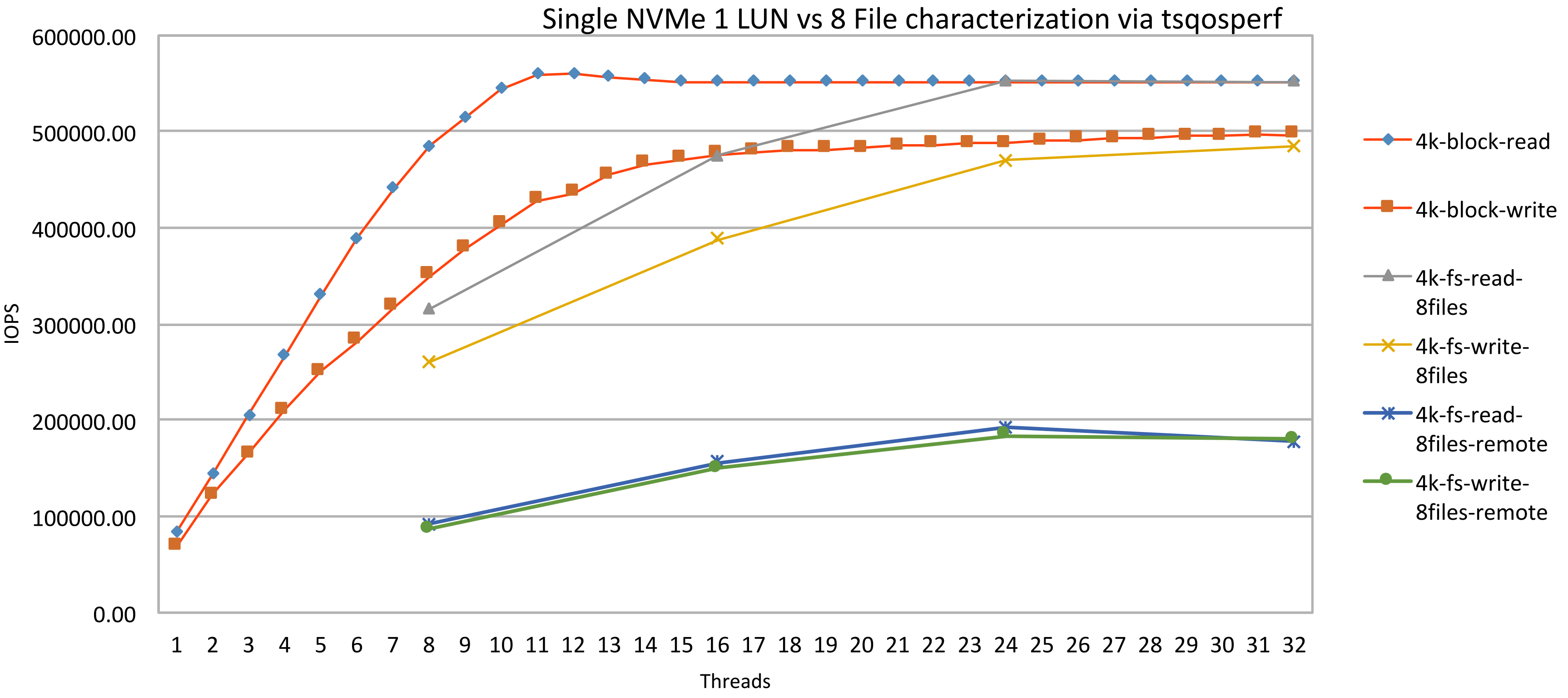
So what does all this mean to me ?

- Ask your Storage vendor for 100% random full block performance numbers, there is no sequential access, if you see 40-50 MB/sec/disk you should look for a different vendor ☺
- Don’t get confused from statements like ‘HW raid is superior’ , as you have seen on the previous slides SW raid can get the spec'd performance numbers out of a drive
- Cache matters, the i/o context matters, in order to get most efficiency you need to know the i/o context , you can only get this with embedded raid code in the filesystem
- Network matters, Bandwidth is not everything you need to be able to use it, you can’t beat RDMA today
- Latency matters, more layers don’t help, you need to condense layers
- Data distribution is important today, we will solve this soon

Agenda

- Performance engineering matters
  - Disk drive engineering – how fast is a drive really ?
  - How to get data from storage to consumer – Network overhaul
  - More than 32 Sub blocks, why and what can we expect from them
- **Spectrum Scale with NVMe**
- Spectrum Scale with NVMeoF
- ESS GF-2 first Performance results
- IOPS - does it actually mean anything ??
- Docker – Dean Hildebrand

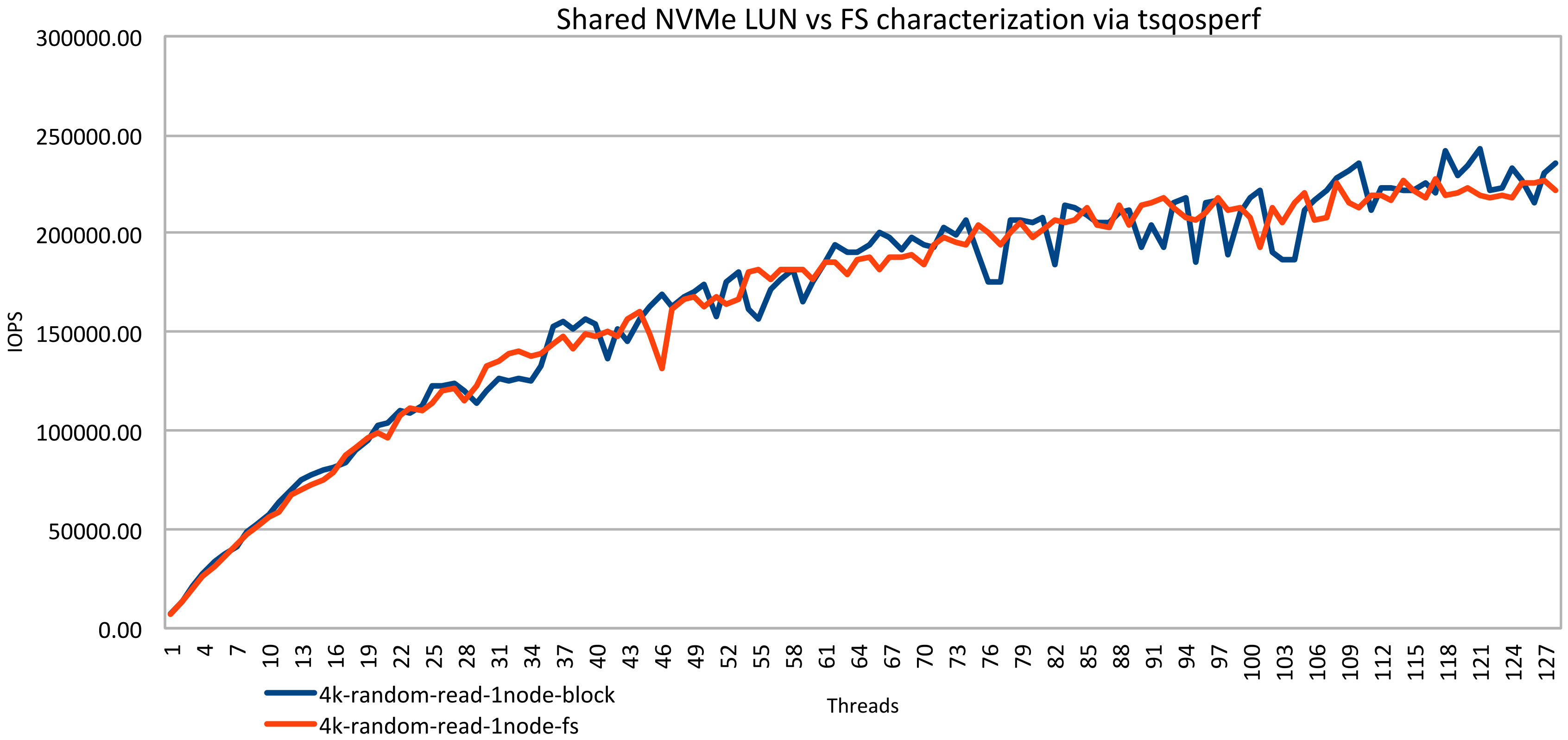
NVMe , Block , Filesystem and Remote Filesystem (NSD) access



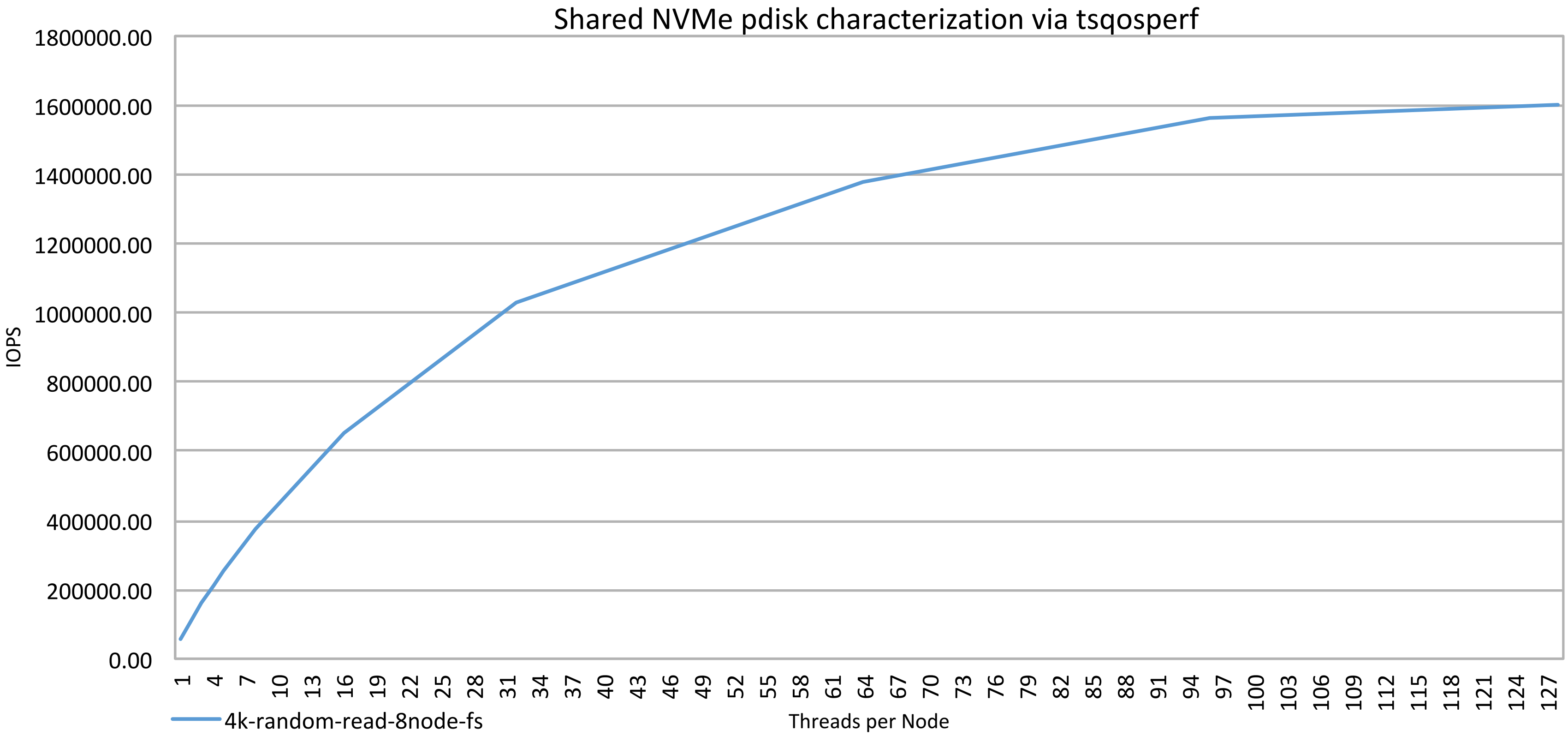
Agenda

- Performance engineering matters
  - Disk drive engineering – how fast is a drive really ?
  - How to get data from storage to consumer – Network overhaul
  - More than 32 Sub blocks, why and what can we expect from them
- Spectrum Scale with NVMe
- **Spectrum Scale with NVMeoF**
- ESS GF-2 first Performance results
- IOPS - does it actually mean anything ??
- Docker – Dean Hildebrand

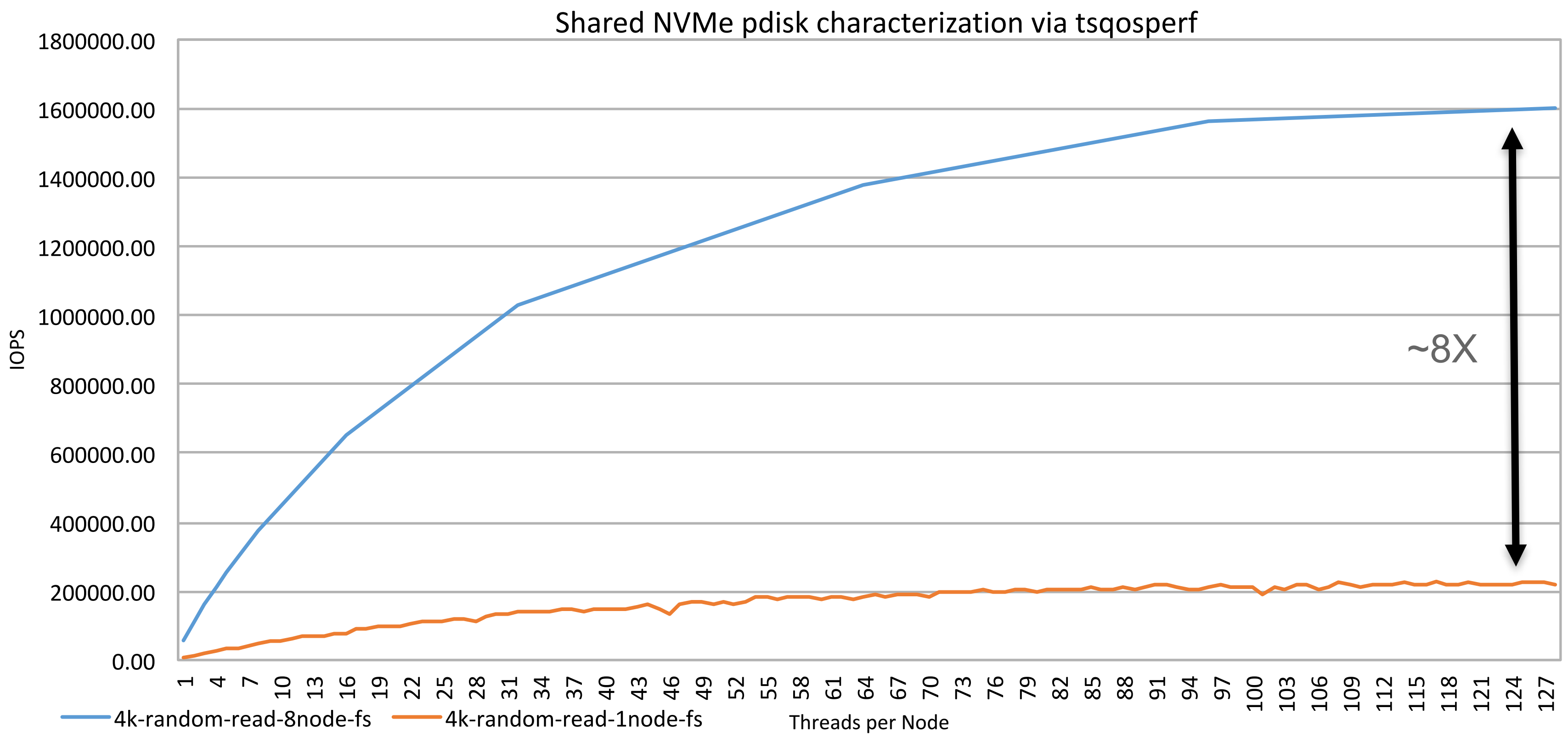
NVMeoF device – single Node – block vs filesystem – 4k RR DIO



NVMeoF device – 8 Nodes – just filesystem access – 4k RR DIO



NVMeoF device – 1 vs 8 Nodes – just filesystem access – 4k RR DIO



Agenda

- Performance engineering matters
  - Disk drive engineering – how fast is a drive really ?
  - How to get data from storage to consumer – Network overhaul
  - More than 32 Sub blocks, why and what can we expect from them
- Spectrum Scale with NVMe
- Spectrum Scale with NVMeoF
- **ESS GF-2 first Performance results**
- IOPS - does it actually mean anything ??
- Docker – Dean Hildebrand

ESS GF-1 first results

```
                  api                 = POSIX
                  test filename       = /ibm/fs2-1m-04/shared/ior//iorfile
                  access              = file-per-process
                  pattern             = segmented (1 segment)
                  ordering in a file = sequential offsets
                  ordering inter file= no tasks offsets
                  clients             = 12 (1 per node)
                  repetitions         = 2
                  xfersize            = 1 MiB
                  blocksize           = 128 GiB
                  aggregate filesize = 1536 GiB
Using Time Stamp 1476166493 (0x57fc835d) for Data Signature

access    bw(MiB/s)  block(KiB) xfer(KiB)  open(s)    wr/rd(s)   close(s)   total(s)   iter
------    ---------  ---------- ---------  --------   --------   --------   --------   ----
delaying 10 seconds . . .
write     8736       134217728  1024.00    0.194921   180.03     0.152748   180.04     0
delaying 10 seconds . . .
read      12971      134217728  1024.00    0.002787   121.26     1.14       121.26     0
remove    -          -          -          -          -          -          0.021527   0
Using Time Stamp 1476166817 (0x57fc84a1) for Data Signature
delaying 10 seconds . . .
write     8830       134217728  1024.00    0.231963   178.11     0.090353   178.12     1
delaying 10 seconds . . .
read      12981      134217728  1024.00    0.007206   121.17     2.44       121.17     1
remove    -          -          -          -          -          -          0.022912   1
```

Max Write: 8830.25 MiB/sec (9259.19 MB/sec)
Max Read: 12980.55 MiB/sec (13611.10 MB/sec)

ESS GF-2 first results

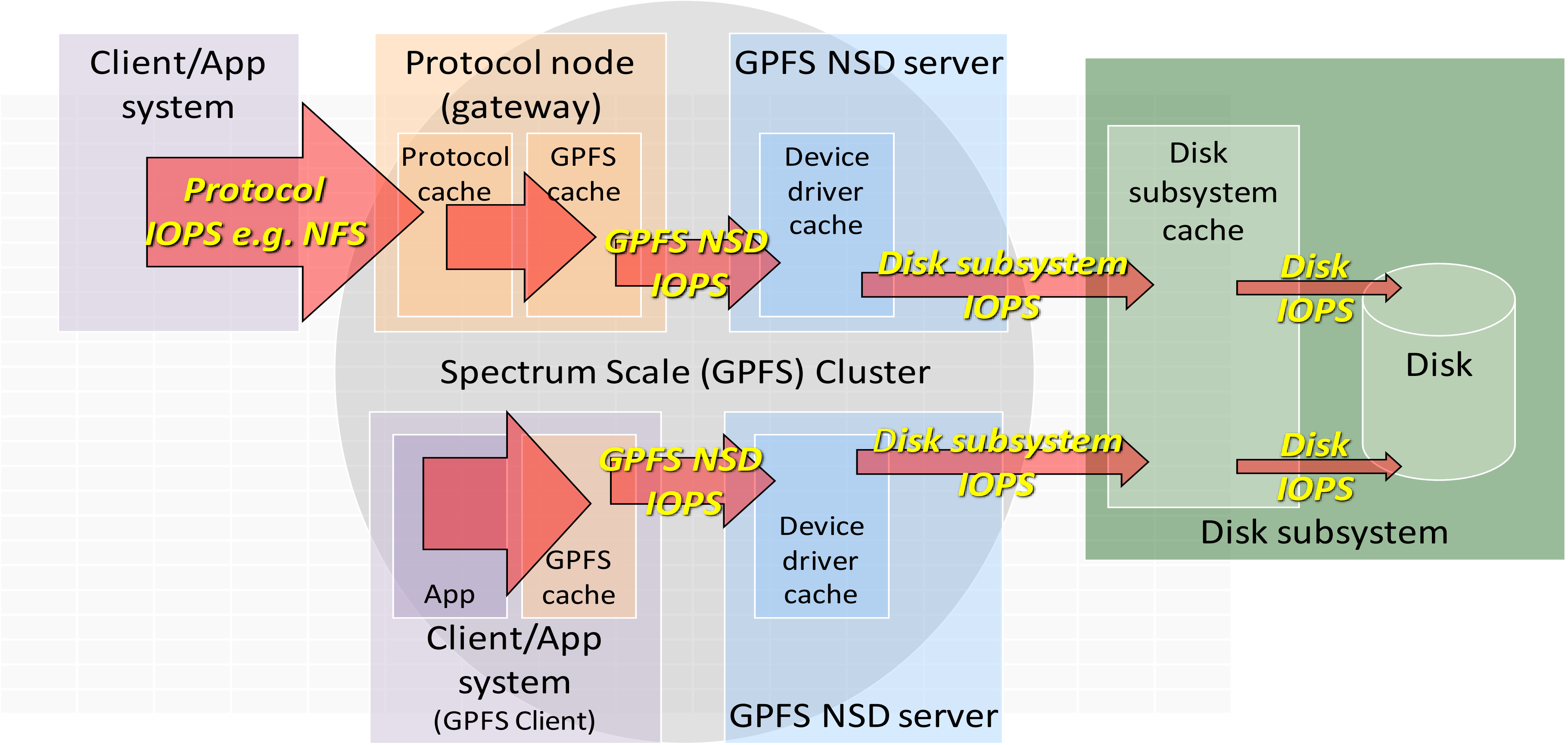
```
                api                 = POSIX
                test filename       = /ibm/fs2-1m-04/shared/ior//iorfile
                access              = file-per-process
                pattern             = segmented (1 segment)
                ordering in a file = sequential offsets
                ordering inter file= no tasks offsets
                clients             = 12 (1 per node)
                repetitions         = 2
                xfersize            = 1 MiB
                blocksize           = 128 GiB
                aggregate filesize = 1536 GiB
Using Time Stamp 1476167316 (0x57fc8694) for Data Signature

access    bw(MiB/s)  block(KiB) xfer(KiB)  open(s)    wr/rd(s)   close(s)   total(s)   iter
------    ---------  ---------- ---------  --------   --------   --------   --------   ----
delaying 10 seconds . . .
write     15729      134217728  1024.00    0.353378   99.99      0.121626   100.00     0
delaying 10 seconds . . .
read      25314      134217728  1024.00    0.001674   62.13      2.15       62.13      0
remove    -          -          -          -          -          -          0.047492   0
Using Time Stamp 1476167498 (0x57fc874a) for Data Signature
delaying 10 seconds . . .
write     15825      134217728  1024.00    0.340123   99.38      0.076954   99.39      1
delaying 10 seconds . . .
read      24863      134217728  1024.00    0.004194   63.26      2.64       63.26      1
remove    -          -          -          -          -          -          0.023176   1
```

Max Write: 15824.97 MiB/sec (16593.68 MB/sec)
Max Read: 25313.76 MiB/sec (26543.40 MB/sec)

Agenda

- Performance engineering matters
  - Disk drive engineering – how fast is a drive really ?
  - How to get data from storage to consumer – Network overhaul
  - More than 32 Sub blocks, why and what can we expect from them
- Spectrum Scale with NVMe
- Spectrum Scale with NVMeoF
- ESS GF-2 first Performance results
- **IOPS - does it actually mean anything ??**
- Docker – Dean Hildebrand

What are IOPS and why there is no answer to the "how many IOPS does a ESS GX system" question



“IOPS” run with IOZONE across 19 nodes

```
Iozone: Performance Test of File I/O
Version $Revision: 3.414 $
Compiled for 64 bit mode.
Build: linux-AMD64
Run began: Sat Mar 30 02:17:43 2013
OPS Mode. Output is in operations per second.
Record Size 4 KB
File size set to 33554432 KB
Network distribution mode enabled.
Command line used: /usr/local/bin/iozone -i 0 -i 1 -O -t 76 -r 4k -s 32g -+m
/ghome/oehmes/mpi/19.clients.76.iozone
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 Kbytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.
Throughput test with 76 processes
Each process writes a 33554432 Kbyte file in 4 Kbyte records
```

IOZONE run results

Test running:
Children see throughput for 76 initial writers = 5859945.57 ops/sec
Min throughput per process = 67595.85 ops/sec
Max throughput per process = 87392.73 ops/sec
Avg throughput per process = 77104.55 ops/sec
Min xfer = 6488345.00 ops
Test running:
Children see throughput for 76 rewriters = 5925287.16 ops/sec
Min throughput per process = 67195.85 ops/sec
Max throughput per process = 88241.34 ops/sec
Avg throughput per process = 77964.30 ops/sec
Min xfer = 6388664.00 ops
Test running:
Children see throughput for 76 readers = 7193675.62 ops/sec
Min throughput per process = 79806.80 ops/sec
Max throughput per process = 112921.42 ops/sec
Avg throughput per process = 94653.63 ops/sec
Min xfer = 5929575.00 ops
Test running:
Children see throughput for 76 re-readers = 7195287.09 ops/sec
Min throughput per process = 76148.84 ops/sec
Max throughput per process = 121510.56 ops/sec
Avg throughput per process = 94674.83 ops/sec
Min xfer = 5257923.00 ops

So what does all this mean ?

The test was performing simultaneous sequential 4k buffered read and write I/O from 19 clients connected via 2 independent Infiniband fabrics to 2 fully populated GL6 systems.

The overall working set size was 2.4 Terrabyte while the combined cache size of all components was only 5% of the working set, this was chosen to eliminate the large influence of the cache to show the performance capabilities of the I/O path.

The write test performed **5.9 million 4k iops** or 23.7GB/sec throughput on average

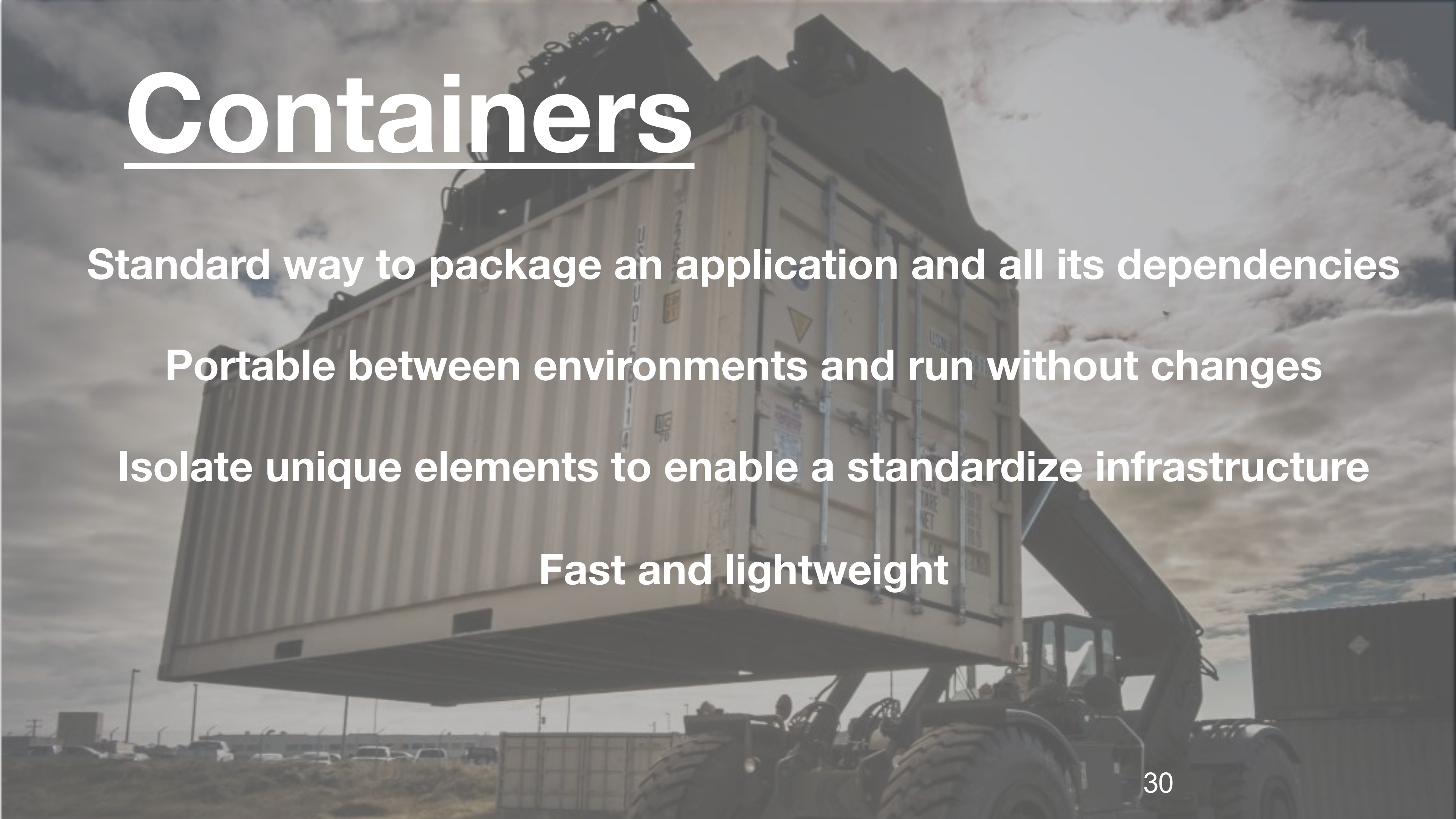
The read test performed **7.2 million 4k iops** or 28.7GB/sec throughput on average

But a word of caution : This numbers are unlikely to be achieved with real customer workloads, like other Storage Systems provides peak numbers and were only collected to demonstrate the technical capabilities of the System

Code version : This data was collected using a pre GPFS 3.5 TL3 version, but there is no expected difference when the code will be released, but worth mentioning

Agenda

- Performance engineering matters
  - Disk drive engineering – how fast is a drive really ?
  - How to get data from storage to consumer – Network overhaul
  - More than 32 Sub blocks, why and what can we expect from them
- Spectrum Scale with NVMe
- Spectrum Scale with NVMeoF
- ESS GF-2 first Performance results
- IOPS - does it actually mean anything ??
- **Docker – Dean Hildebrand**



Containers

Standard way to package an application and all its dependencies

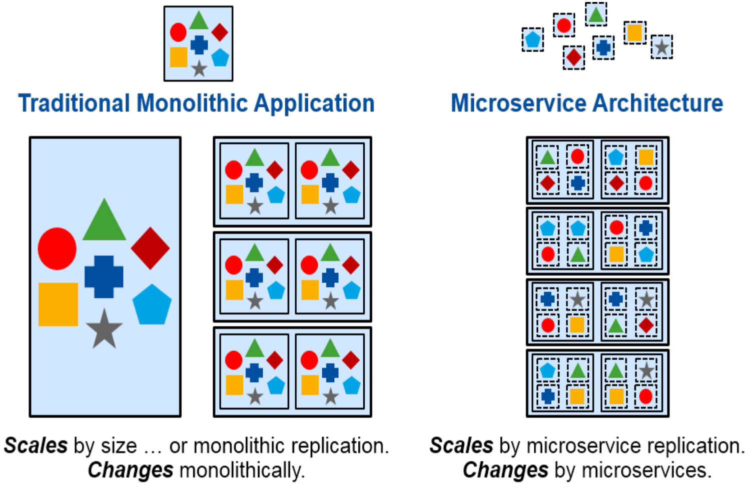
Portable between environments and run without changes

Isolate unique elements to enable a standardize infrastructure

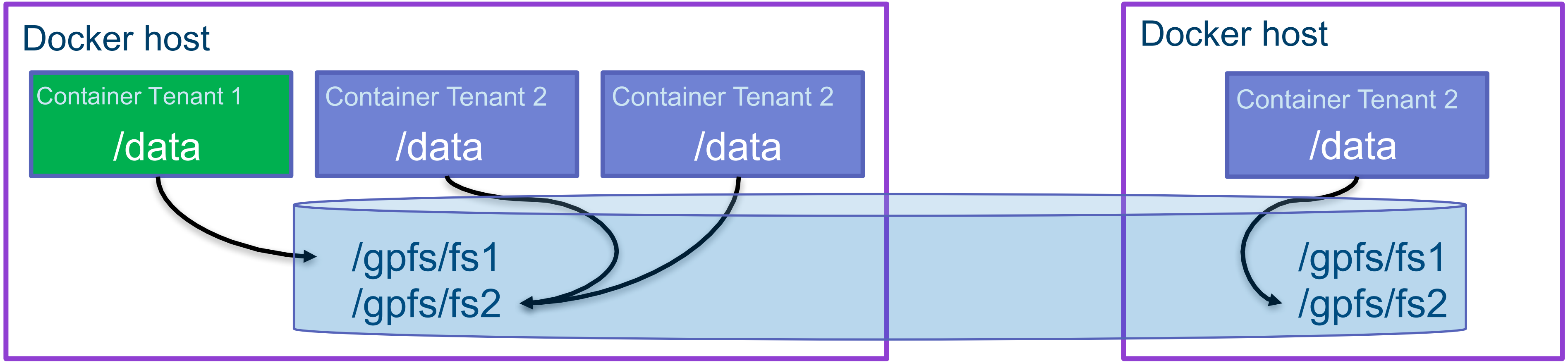
Fast and lightweight

Containers, Containers, Containers

- HPC and Scientific Computing
  - Portable and reproducible science
- On-premise Clouds
- DevOps and continuous integration
- Simplify and speed application development
- Load balancing across cloud VM services (e.g., EC2)



Spectrum Scale and Docker



Details

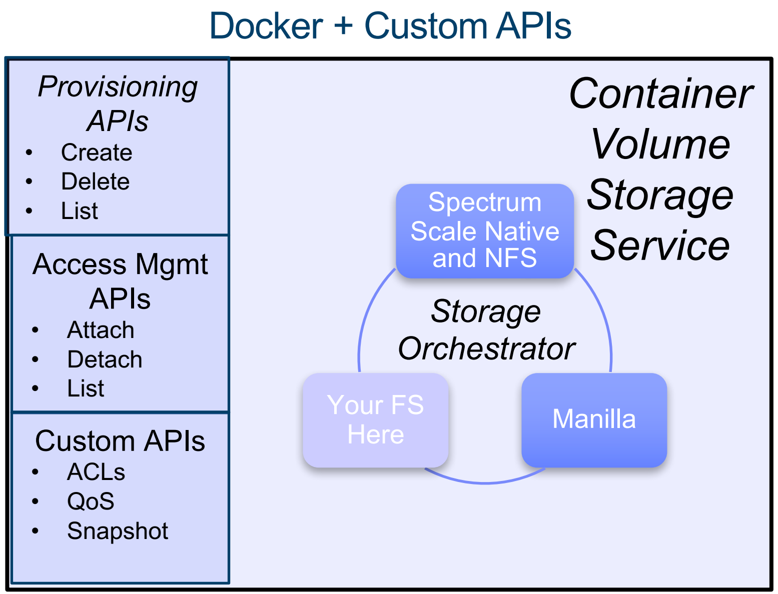
- Single or separate userid namespaces between containers and hosts
- Data sharing across containers and hosts
- All POSIX commands supported from container

A Few Benefits

- Multi-tenant access
  - Container can only access its volumes
  - Allow root access in container without allowing root access to file system
  - ACLs can add an extra level of security
- Native client performance

Spectrum Scale Container Volume Service Architecture
Provisioning persistent storage for Docker and Kubernetes

- **Storage volume drivers** abstract storage complexities while exposing key capabilities
- Docker and Kubernetes have different mechanisms for provisioning storage
- Build a single service to provision volumes for that can be shared across all frameworks
- Support a variety of features, existing or new volumes, Quota, etc
- Currently working with Spectrum Computing
- Plan is to open-source



Thank you!

ibm.com/systems/hpc

Legal notices

Information and trademarks

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.

Special notices

This document was developed for IBM offerings in the United States as of the date of publication. IBM may not make these offerings available in other countries, and the information is subject to change without notice. Consult your local IBM business contact for information on the IBM offerings available in your area.

Information in this document concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquires, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

The information contained in this document has not been submitted to any formal IBM test and is provided "AS IS" with no warranties or guarantees either expressed or implied.

All examples cited or described in this document are presented as illustrations of the manner in which some IBM products can be used and the results that may be achieved. Actual environmental costs and performance characteristics will vary depending on individual client configurations and conditions.

IBM Global Financing offerings are provided through IBM Credit Corporation in the United States and other IBM subsidiaries and divisions worldwide to qualified commercial and government clients. Rates are based on a client's credit rating, financing terms, offering type, equipment type and options, and may vary by country. Other restrictions may apply. Rates and offerings are subject to change, extension or withdrawal without notice.

IBM is not responsible for printing errors in this document that result in pricing or information inaccuracies.

All prices shown are IBM's United States suggested list prices and are subject to change without notice; reseller prices may vary.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this document may have been estimated through extrapolation. Users of this document should verify the applicable data for their specific environment.