Investigation of Storage Systems for use in Grid Applications

Overview

- Test bed and testing methodology
- Tests of general performance
- Tests of the metadata servers
- Tests of root-based applications
- HEPiX storage group result
- Operations and fault tolerance

ISGC 2012
Feb 27, 2012

Keith Chadwick for Gabriele Garzoglio
Grid & Cloud Computing dept.,
Computing Sector, Fermilab

Fermilab

Acknowledgements

- *Ted Hesselroth, Doug Strain – IOZone Perf. measurements*
- *Andrei Maslennikov* – HEPiX storage group
- *Andrew Norman, Denis Perevalov* – Nova framework for the storage benchmarks and HEPiX work
- *Robert Hatcher, Art Kreymer* – Minos framework for the storage benchmarks and HEPiX work
- *Steve Timm, Neha Sharma, Hyunwoo Kim* – FermiCloud support
- *Alex Kulyavtsev, Amitoj Singh* – Consulting

- This work is supported by the U.S. Department of Energy under contract No. DE-AC02-07CH11359

Context

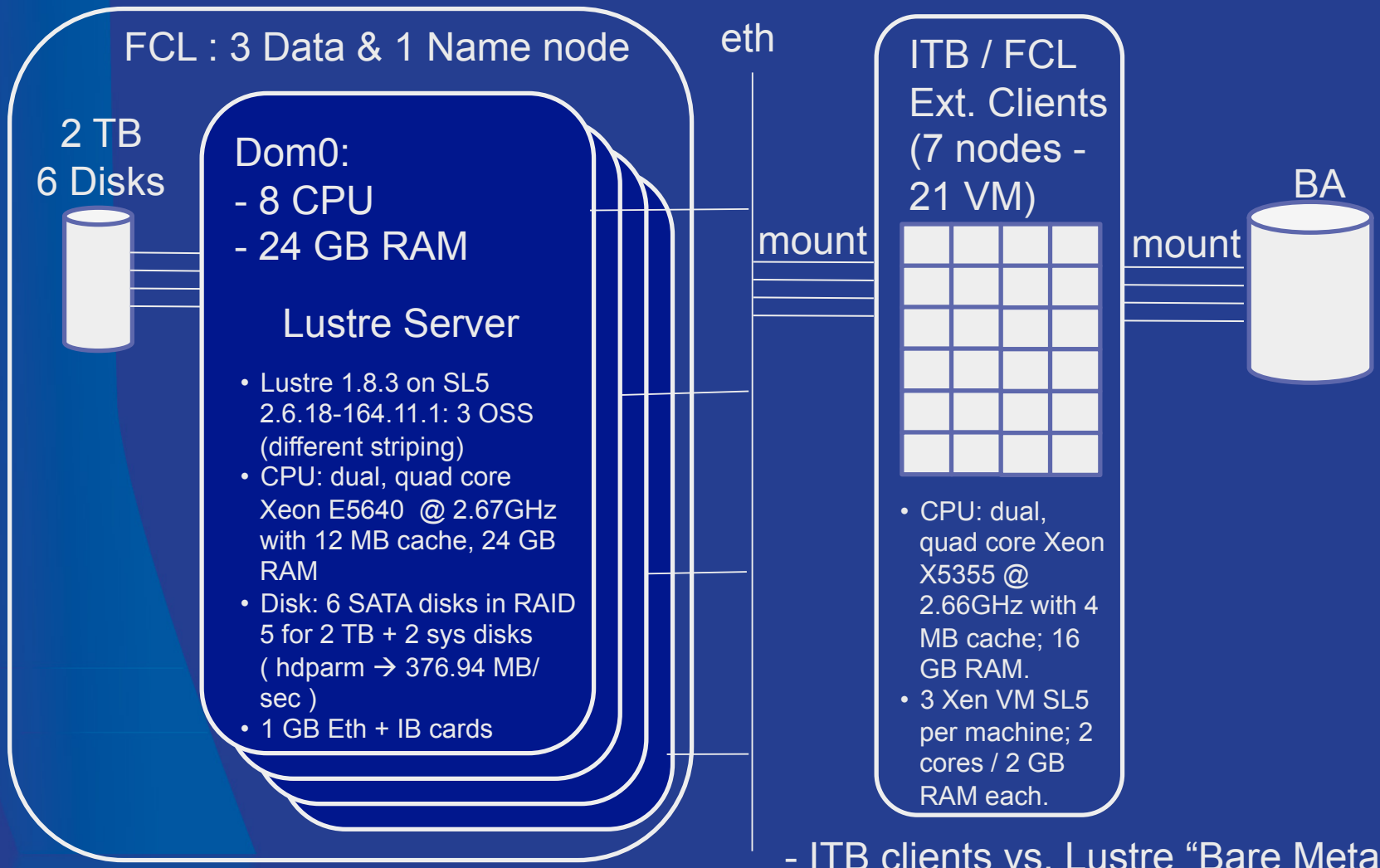
- Goal
  - Evaluation of storage technologies for the use case of data intensive jobs on Grid and Cloud facilities at Fermilab.
- Technologies considered
  - Lustre (v1.8.3 / kernel 2.6.18)
  - Hadoop Distributed File System (HDFS v0.19.1-20)
  - Blue Arc (BA)
  - Orange FS (PVFS2 v2.8.4)
- Targeted infrastructures:
  - FermiGrid, FermiCloud, and the General Physics Computing Farm.
- Collaboration at Fermilab:
  - FermiGrid / FermiCloud, Open Science Grid Storage team, Data Movement and Storage, Running Experiments
- More info at

  http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=3532

Evaluation Method

- **Set the scale**: measure storage metrics from running experiments to set the scale on expected bandwidth, typical file size, number of clients, etc.
  - http://home.fnal.gov/~garzogli/storage/dzero-sam-file-access.html
  - http://home.fnal.gov/~garzogli/storage/cdf-sam-file-access-per-app-family.html
- **Install storage solutions on FermiCloud testbed**
- **Measure performance**
  - run standard benchmarks on storage installations
  - study response of the technology to real-life (skim) applications access patterns (root-based)
  - use HEPiX storage group infrastructure to characterize response to IF applications
- **Fault tolerance**: simulate faults and study reactions
- **Operations:** comment on potential operational issues

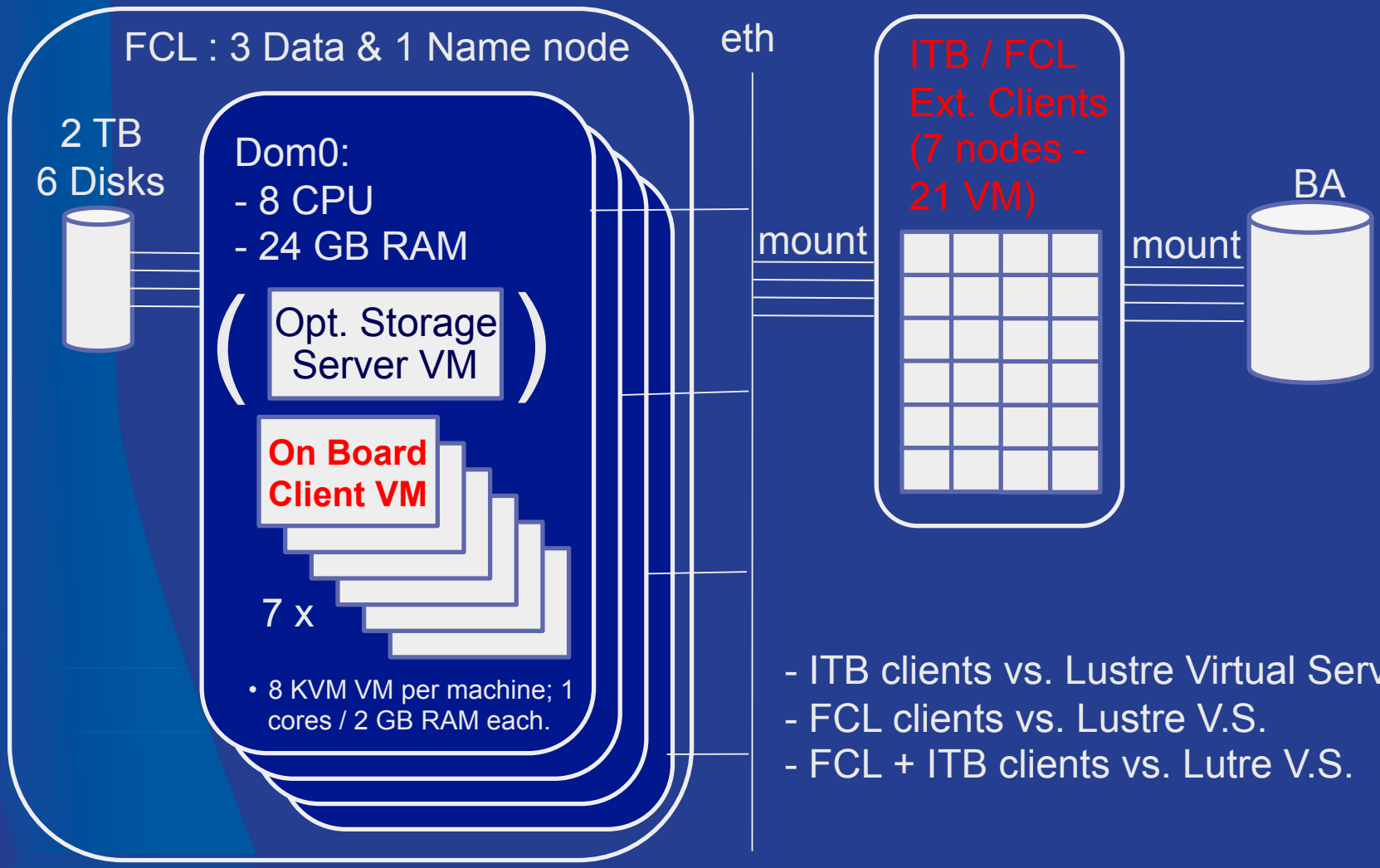
Test Bed: “Bare Metal” Clnt / Srvr



- ITB clients vs. Lustre “Bare Metal”

Fermilab

Test Bed: On-Board vs. External Clnts



- ITB clients vs. Lustre Virtual Server
- FCL clients vs. Lustre V.S.
- FCL + ITB clients vs. Lutre V.S.

Data Access Tests

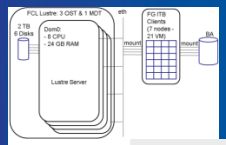
- **IOZone**
  - Writes (2GB) file from each client and performs read/write tests.
  - **Setup:** 3-60clients on Bare Metal (BM) and 3-21 VM/nodes.

- **Root-based applications**
  - Nova: ana framework, simulating skim app – read large fraction of all events → disregard all (read-only)
  - Minos: loon framework, simulating skim app – data is compressed → access CPU-bound (does NOT stress storage) NOT SHOWN
  - Writes are CPU-bound: NOT SHOWN
  - **Setup:** 3-60clients on Bare Metal and 3-21 VM/nodes.

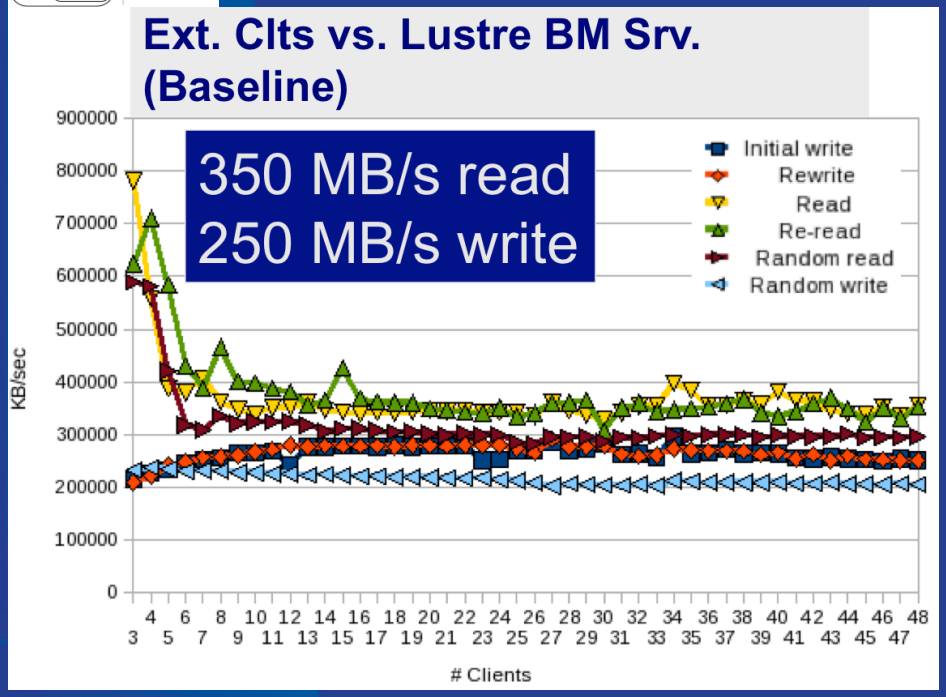
- **MDTest**
  - Different FS operations on up to 50k files / dirs using different access patterns.
  - **Setup:** 21-504 clients on 21 VM.

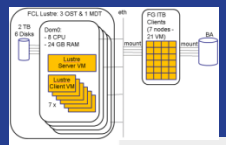
Lustre Performance (iozone)

How well does Lustre perform with servers on Bare Metal vs. VM ?

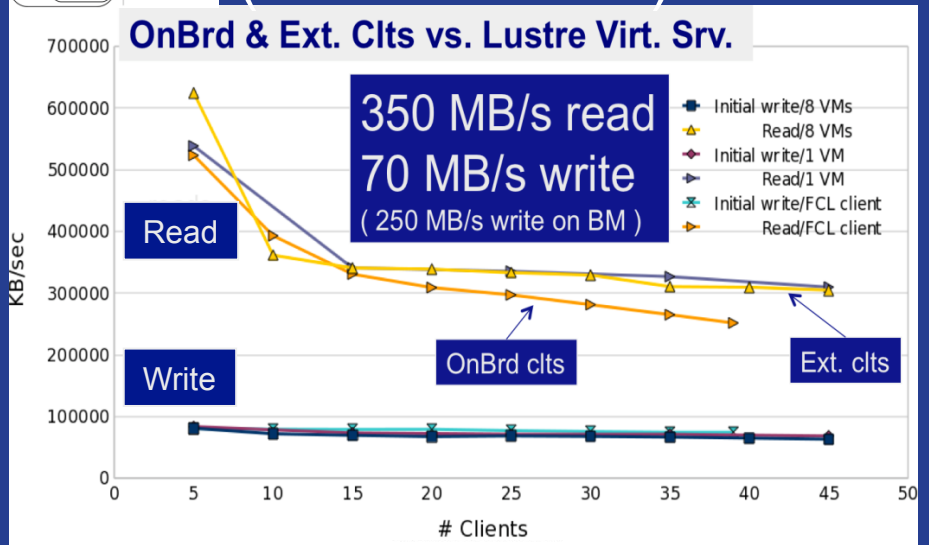


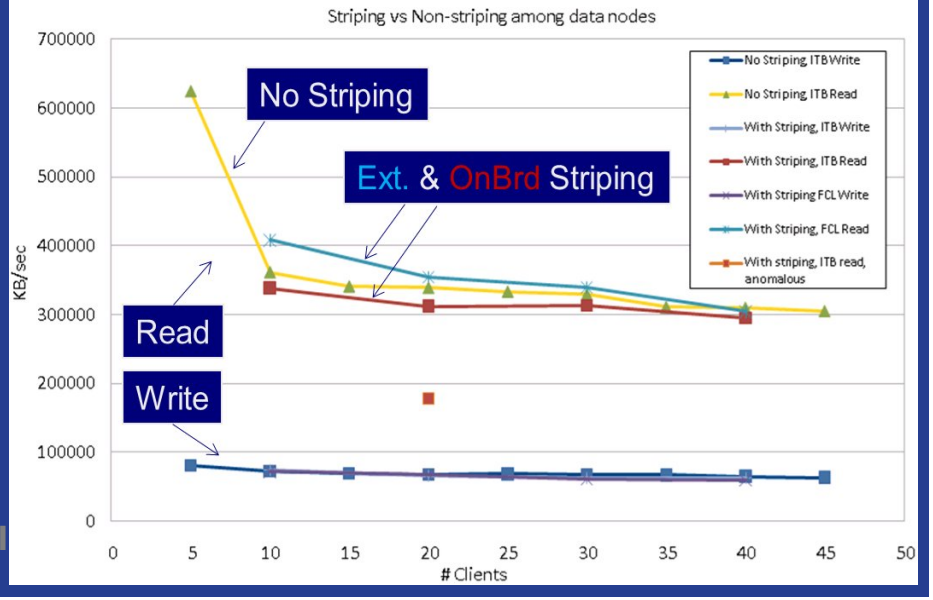
Lustre Srv. on Bare Metal





Lustre Srv. on VM (SL5 and KVM)



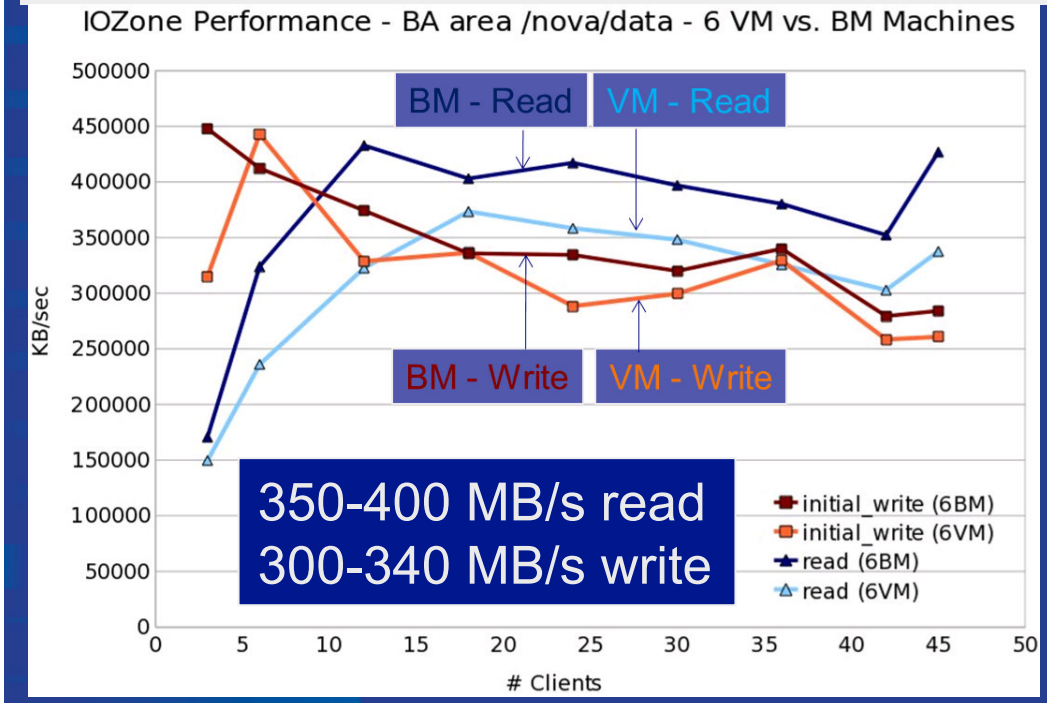


- **Read: same for Bare Metal and VM srv (w/ virtio net drv.)**
- **Read: OnBrd clts 15% slower than Ext. clts (not significant)**
- **Write: Bare Metal srv 3x faster than VM srv**
- **Striping has a 5% effect on reading, none on writing.**
- **No effect changing number of cores on Srv VM**

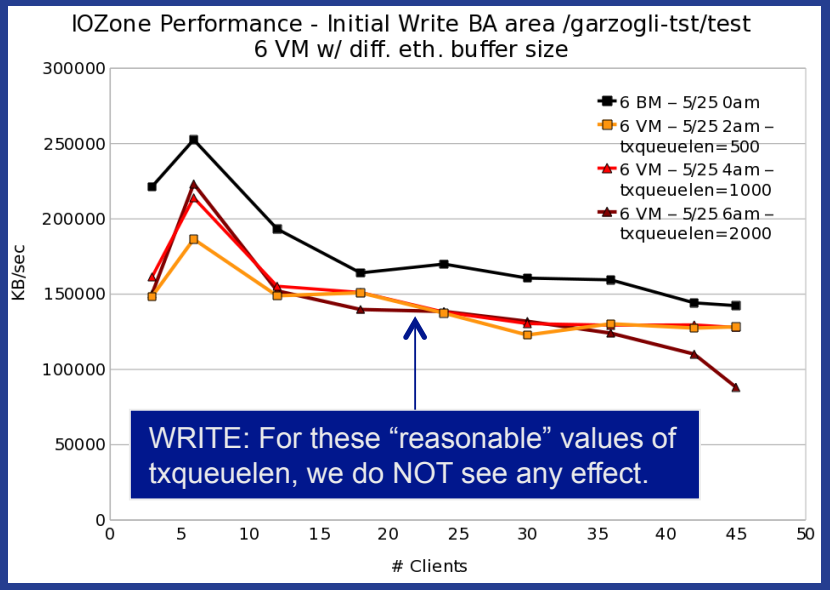
Note: SL6 may have better write performance

BA Performance (iozone)

How well do VM clients perform vs. Bare Metal clients?

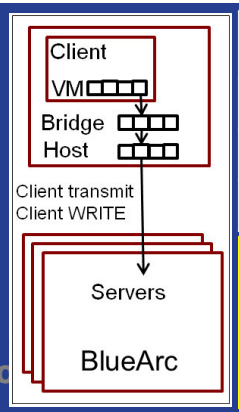


Do we need to optimize transmit eth buffer size (txqueuelen) for the client VMs (writes) ?



- **Bare Metal Reads are (~10%) faster than VM Reads.**
- **Bare Metal Writes are (~5%) faster than VM Writes.**

Note: results vary depending on the overall system conditions (net, storage, etc.)

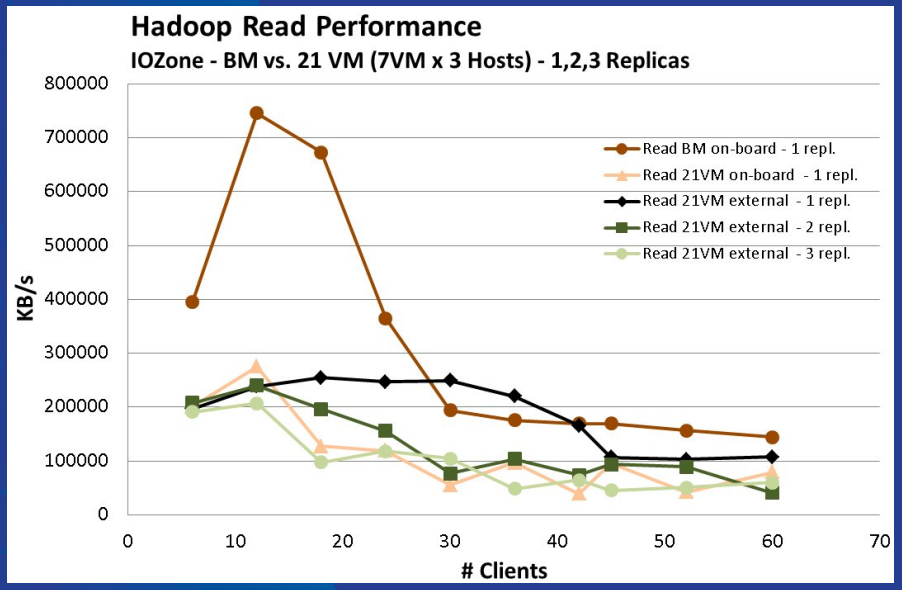


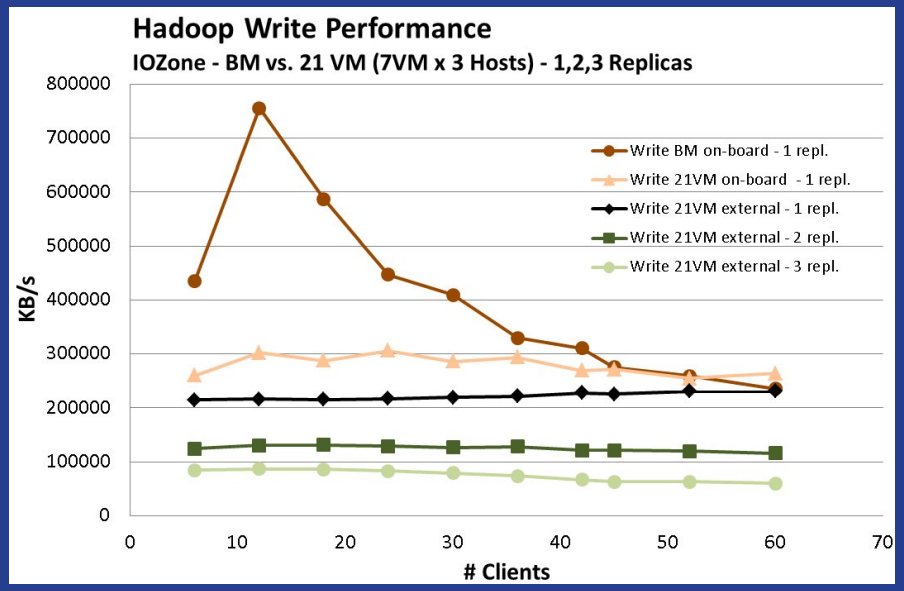
Eth interface	txqueuelen
Host	1000
Host / VM bridge	500, 1000, 2000
VM	1000

Varying the eth buffer size for the client VMs does not change read / write BW.

Hadoop Performance (iozone)

How well do VM clients perform vs. Bare Metal clients?
Is there a difference for External vs. OnBoard clients?
How does number of replica change performance?





On-Brd Bare Metal cl. reads gain from kernel caching, for a few clients. For many clients, same or ~50% faster than On-Brd VM & ~100% faster than Ext. VM clients.

On-Brd VM Cl. 50%-150% faster than Ext. VM clients.

Multiple replicas have little effect on read BW.

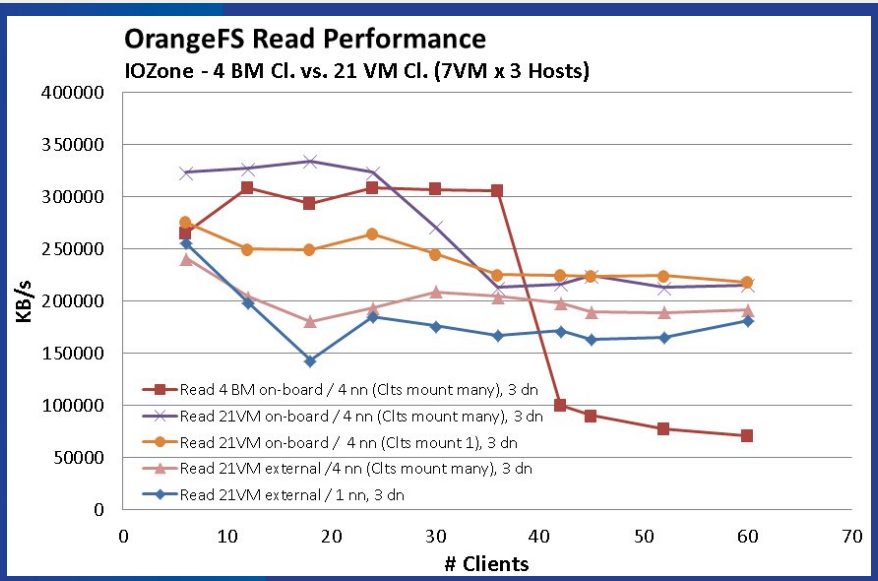
On-Brd Bare Metal cl. Writes gain from kernel caching: generally faster than VM cl.

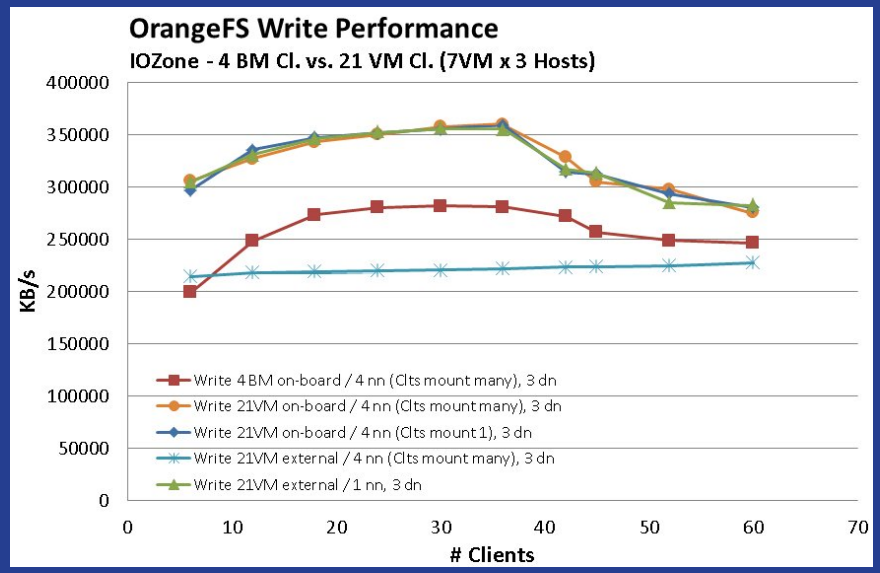
On-Brd VM cl. 50%-200% faster than Ext. VM clients. All VM write scale well with number of clients.

For Ext. VM cl., write speed scales ~linearly with number of replicas.

OrangeFS Performance (iozone)

How well do VM clients perform vs. Bare Metal clients?
Is there a difference for External vs. OnBoard clients?
How does number of name nodes change performance?





On-Brd Bare Metal cl. read almost as fast as OnBrd VM (faster config.), but 50% slower than VM cl. for many processes: possibly too many procs for the OS to manage.

On-Brd VM Cl. read 10%-60% faster than Ext. VM clients.

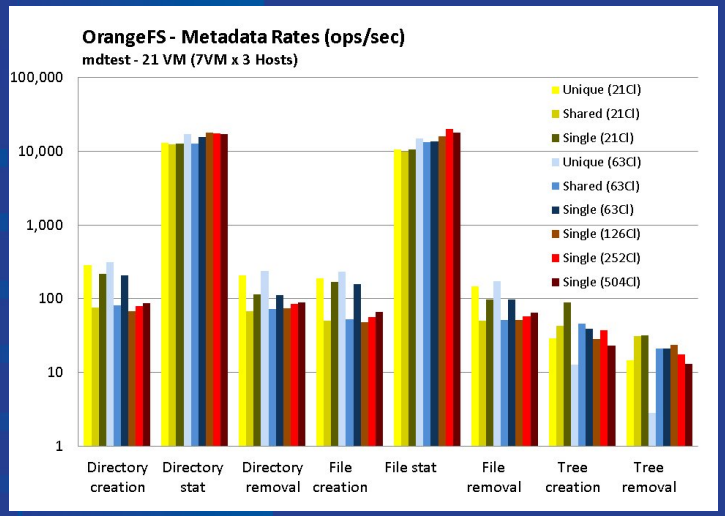
Using 4 name nodes improves read performance by 10%-60% as compared to 1 name node (different from write performance). Best performance when each name node serves a fraction of the clients.

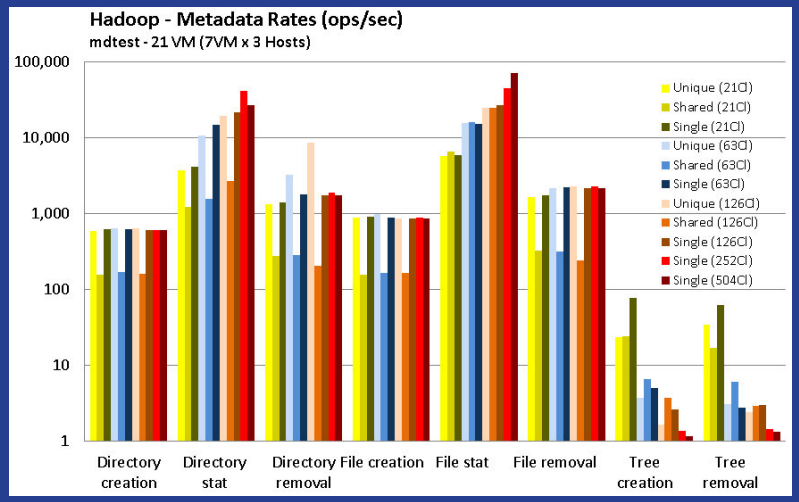
On-Brd Bare Metal cl. write are 80% slower than VM cl.: possibly too many processes for the OS to manage.

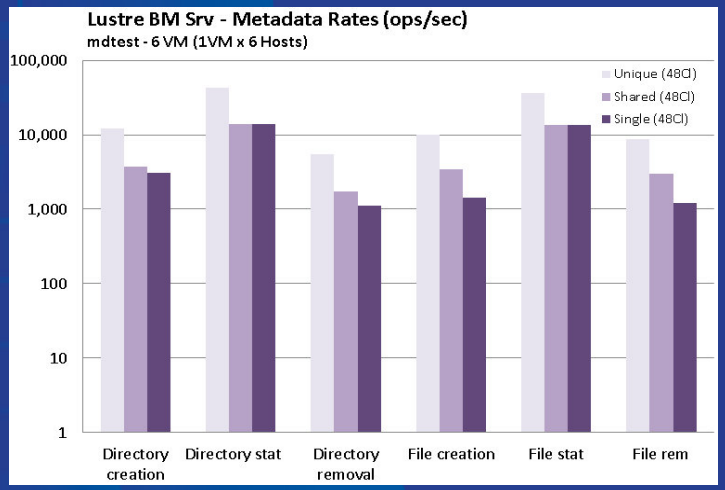
Write performance NOT consistent. On-Brd VM cl. generally have the same performance as Ext. VM clients. One reproducible 70% slower write meas. for Ext. VM (4 name nodes when each nn serves a fraction of the cl.)

Using 4 name nodes has 70% slower perf. for Ext. VM (reproducible). Different from read performance.

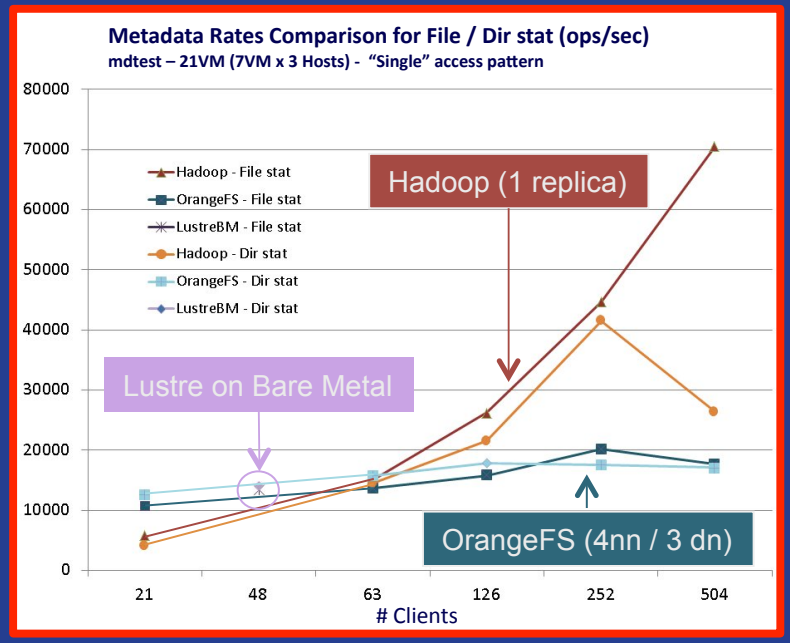
MetaData Srv Performance Comparison







Hadoop Name Node scales better than OrangeFS



Using 3 meta-data testing access patterns

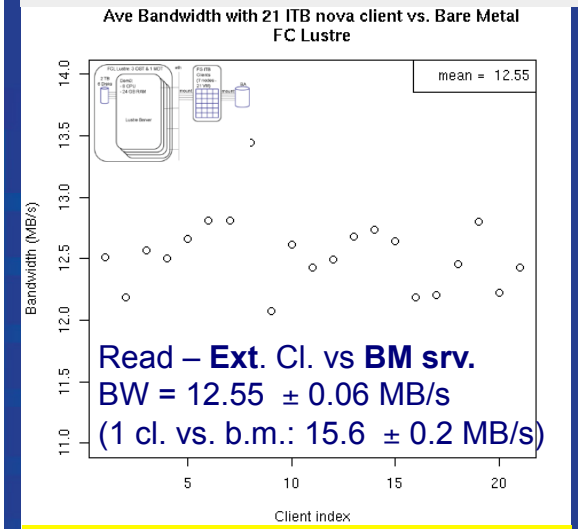
Unique
each client performs metadata operations on a directory and files unique for that client, all at the same time

Shared
one client creates a set of test files, then all clients perform operations on those shared files at the same time

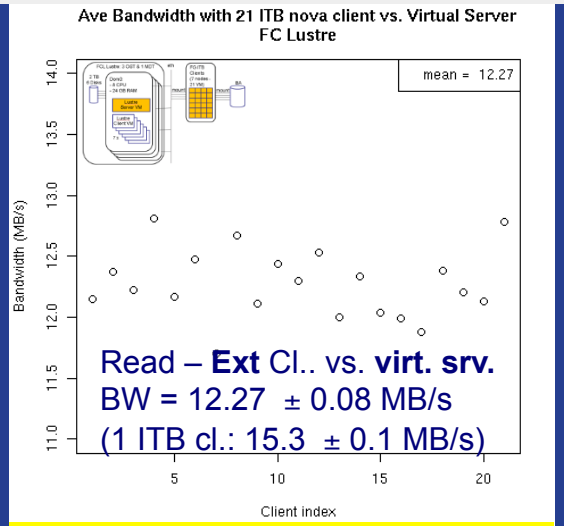
Single
all clients perform operations on files unique for that client, but all files are located in a single shared directory.

Lustre Perf. – Root-based Benchmark (21 clts)

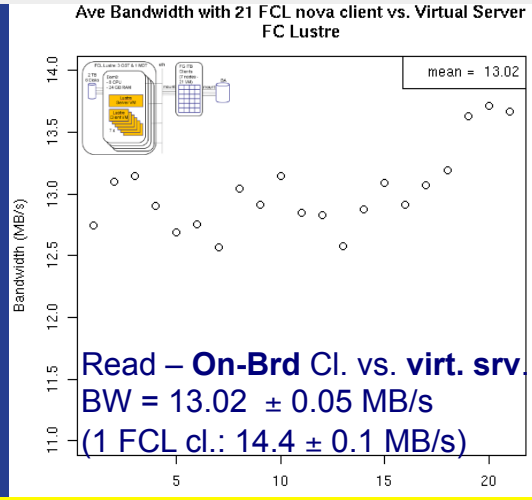
Read performance: how does Lustre Srv. on VM compare with Lustre Srv. on Bare Metal for External and On-Board clients?



Non-Striped Bare Metal (BM) Server: baseline for read

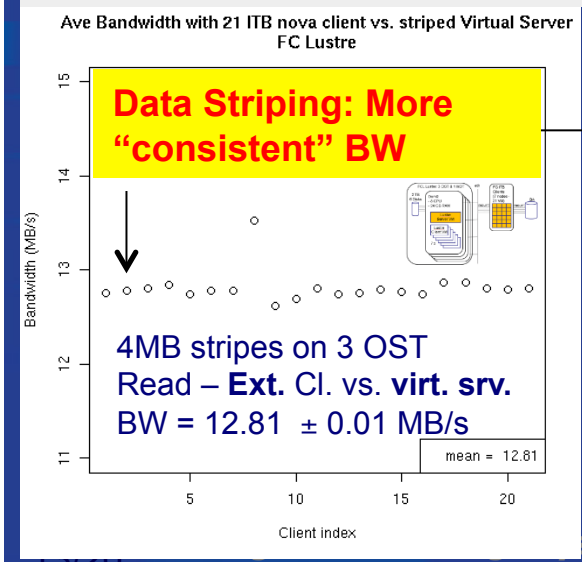


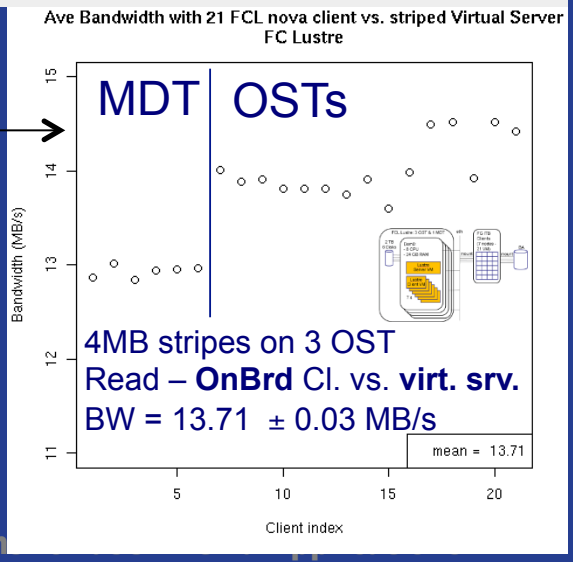
Virtual Server is almost as fast as Bare Metal for read



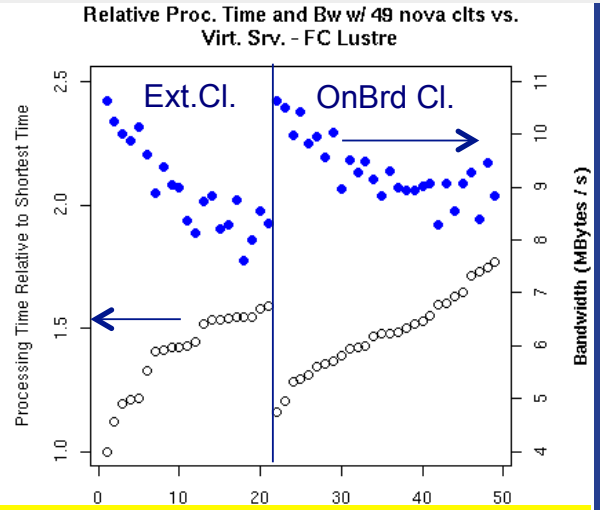
Virtual Clients as fast as BM for read. On-Brd Cl. 6% faster than Ext. cl. (Opposite as Hadoop & OrangeFS)

Does Striping affect read BW?





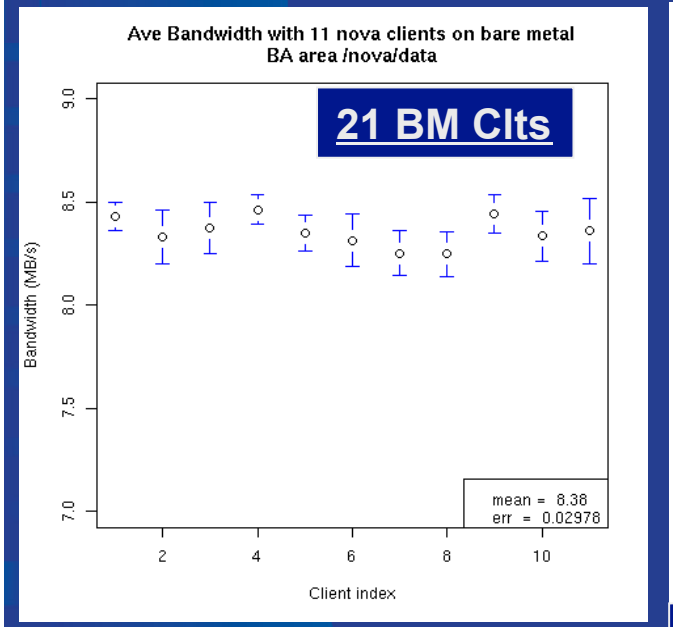
49 clts saturate the BW: is the distrib. fair?

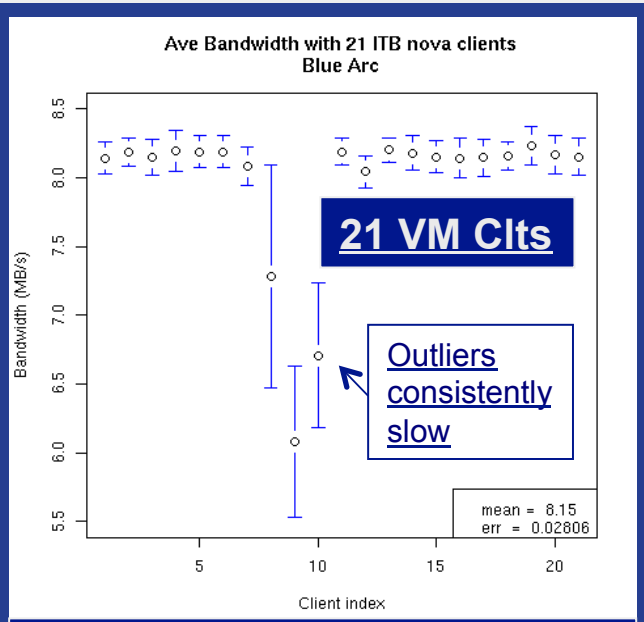


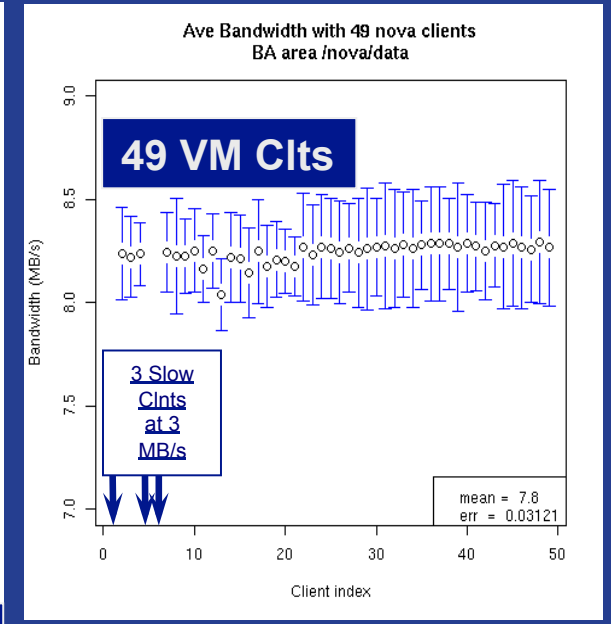
Clients do NOT all get the same share of the bandwidth (within 20%).

BA Perf. – Root-based Benchmark

How well do VM clients perform vs. Bare Metal (BM) clients?







Root-app Read Rates:
21 Clts: 8.15 ± 0.03 MB/s
(Lustre: 12.55 ± 0.06 MB/s
Hadoop: ~7.9 ± 0.1 MB/s)

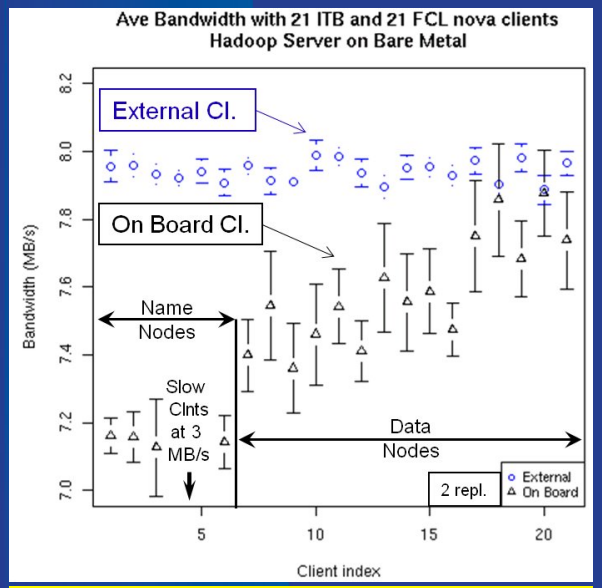
Read BW is essentially the same on Bare Metal and VM.

Note: NOvA skimming app reads 50% of the events by design. On BA, OrangeFS, and Hadoop, clients transfer 50% of the file. On Lustre 85%, because the default read-ahead configuration is inadequate for this use case.

Fermilab

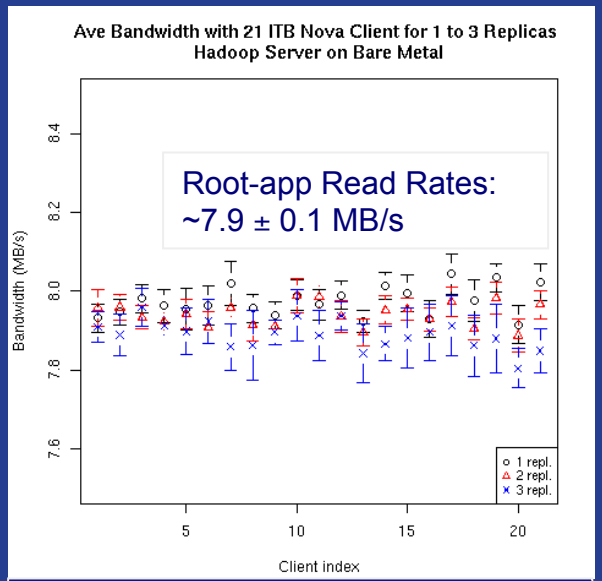
Hadoop Perf. – Root-based Benchmark

How does read BW vary for On-Brd vs. Ext. clnts?



Ext (ITB) clients read ~5% faster then on-board (FCL) clients.

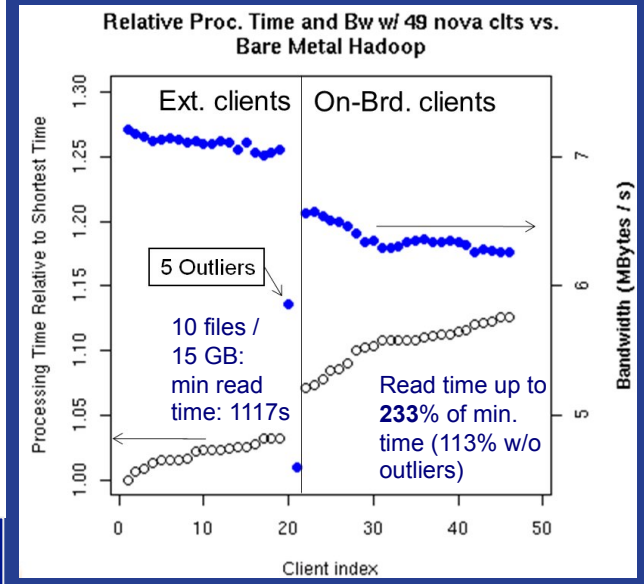
How does read bw vary vs. number of replicas?



(Lustre on Bare Metal was 12.55 ± 0.06 MB/s Read)

Number of replicas has minimal impact on read bandwidth.

49 clts (1 proc. / VM / core) saturate the BW to the srv. Is the distribution of the BW fair?



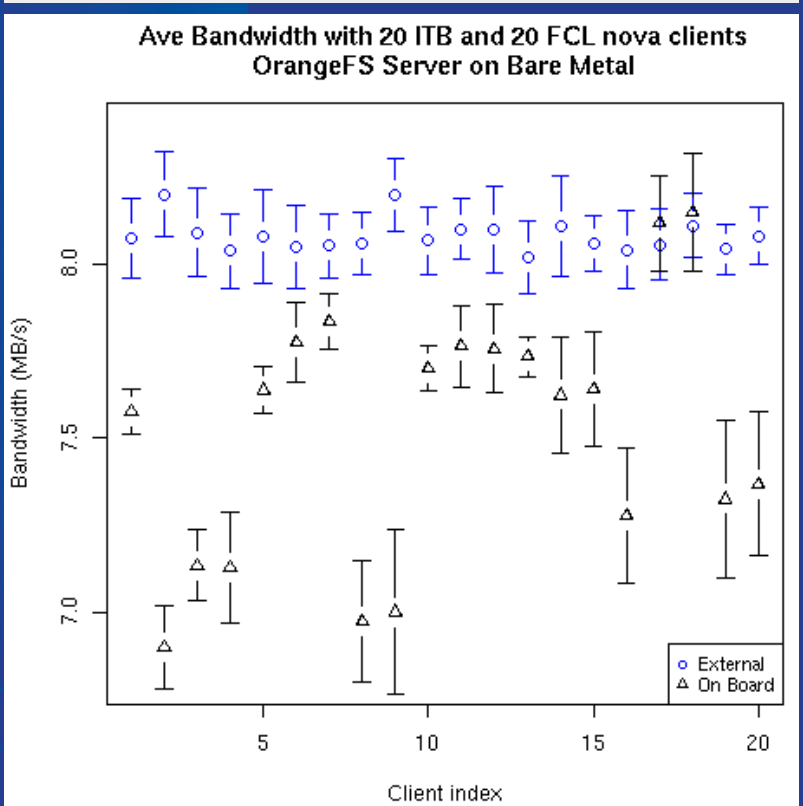
At saturation, Ext. cl. read ~10% faster than On-Brd cl. (Same as OrangeFS. Different from Lustre)

Ext and On-Brd cl. get the same share of the bw among themselves (within ~2%).

Fermilab

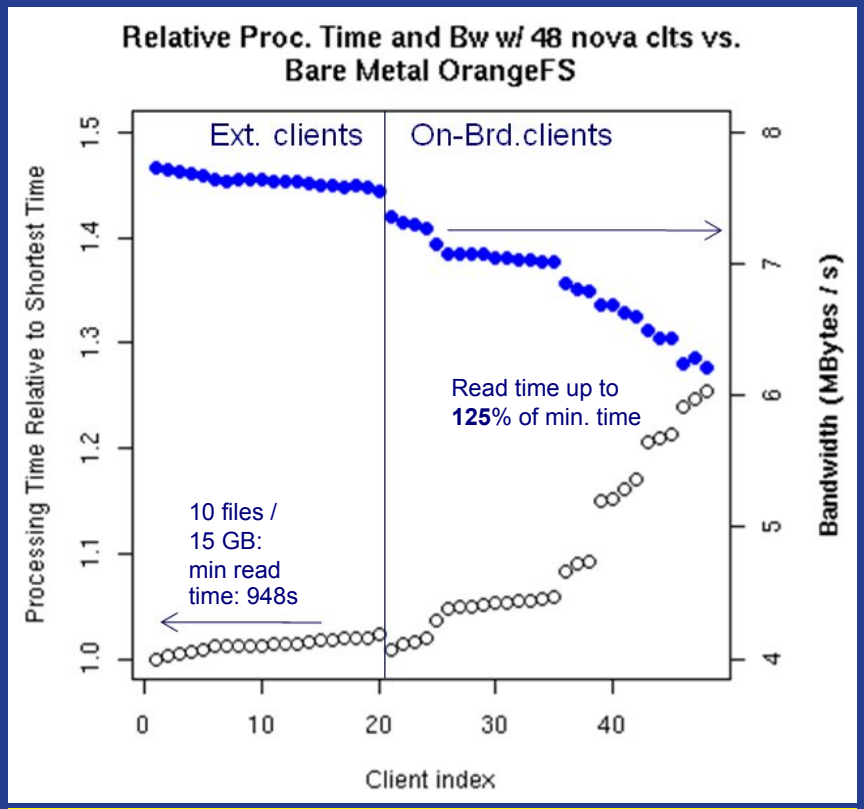
OrangeFS Perf. – Root-based Benchmark

How does read BW vary for On-Brd vs. Ext. clnts?



Ext (ITB) clients read ~7% faster then on-board (FCL) clients
(Same as Hadoop. Opposite from Lustre virt. Srv.).

49 clts (1 proc. / VM / core) saturate the BW to the srv. Is the distribution of the BW fair?

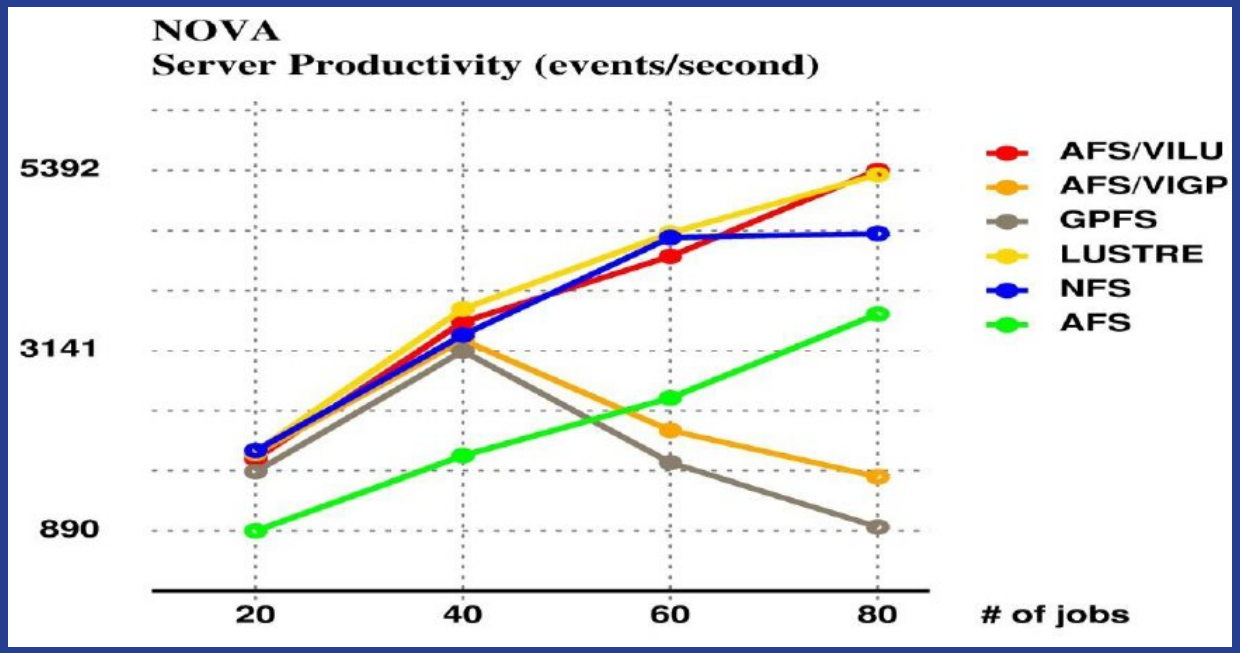


At saturation, on average Ext. cl. read ~10% faster than On-Brd cl.
(Same as Hadoop. Different from Lustre virt. Srv.)

Ext cl. get the same share of the bw among themselves (within ~2%) (as Hadoop).
On-Brd. cl. have a larger spread (~20%)
(as Lustre virt. Srv.).

HEPiX Storage Group

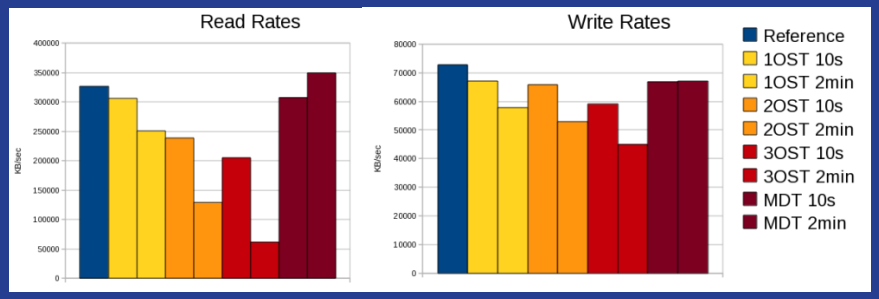
- Collaboration with Andrei Maslennikov (*)
- Nova offline skim app. used to characterize storage solutions. Clients read and write events to storage (bi-directional test).
- Run 20,40,60,80 jobs per 10-node cluster (2,4,6,8 jobs per node); each of the jobs is processing a dedicated non-shared set of event files;
- Start and stop all jobs simultaneously after some predefined period of smooth running;
- Lustre with AFS front-end for caching has best performance (AFS/VILU), although similar to Lustre



* HEPiX Storage Working Group – Oct 2011, Vancouver - Andrei Maslennikov

Comparison of fault tolerance / operations

- **Lustre (v1.8.3 / kernel 2.6.18)**
  - Server: special Kernel w/ special FS. Client: Kernel module → **operational implications**.
  - FS configuration managed through MGS server. Need look-ahead tuning when using root files.
  - Fault tolerance: turning off 1,2,3 OST or MDT for 10 or 120 sec on Lustre Virt. Srv. during iozone tests
    - 2 modes: Fail-over vs. Fail-out. **Lost data when transitioning.**
    - **Graceful degradation**: If OST down → access suspended; If MDT down → ongoing access is NOT affected



- **Hadoop (v0.19.1-20)**
  - Server: Java process in user space on ext3. Client: fuse kernel module: **not fully POSIX**.
  - Configuration managed through files. Some issues configuring number of replicas for clients.
  - Block-level replicas make the system fault tolerant.
- **OrangeFS (v2.8.4)**
  - Server process in user space on ext3. Client: Kernel module.
  - Configuration managed through files.
  - **Limited documentation and small community**. Log files have error messages hard to interpret
  - Data spread over data nodes w/o replicas: no fault tolerance if one node goes down.
- **BlueArc**
  - Server: dedicated HW with RAID6 arrays. Client: mounts via NFS.
  - Professionally managed at Fermilab. More HW / better scalability than test bed for the other solutions.

Results Summary

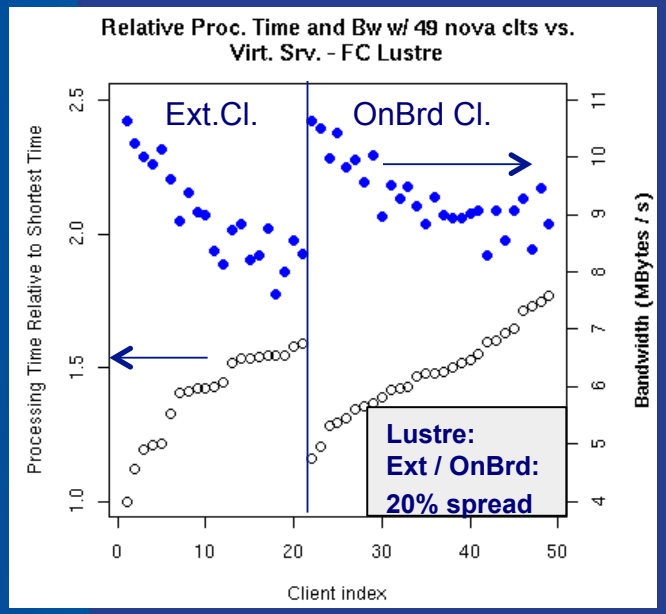
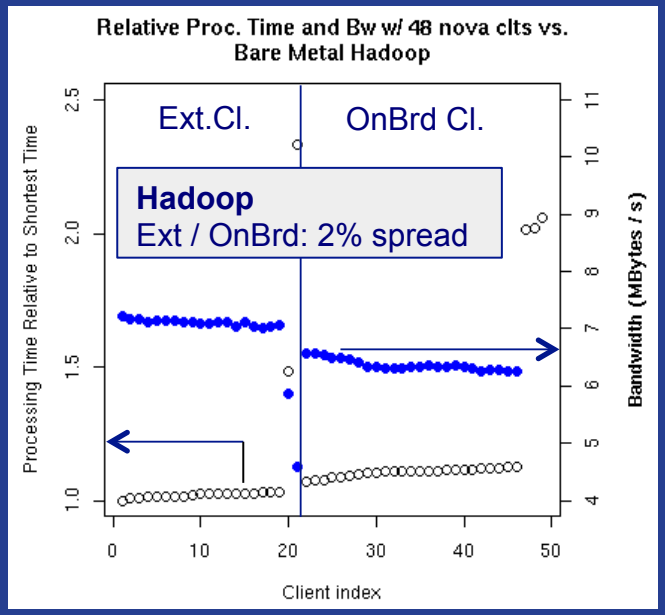
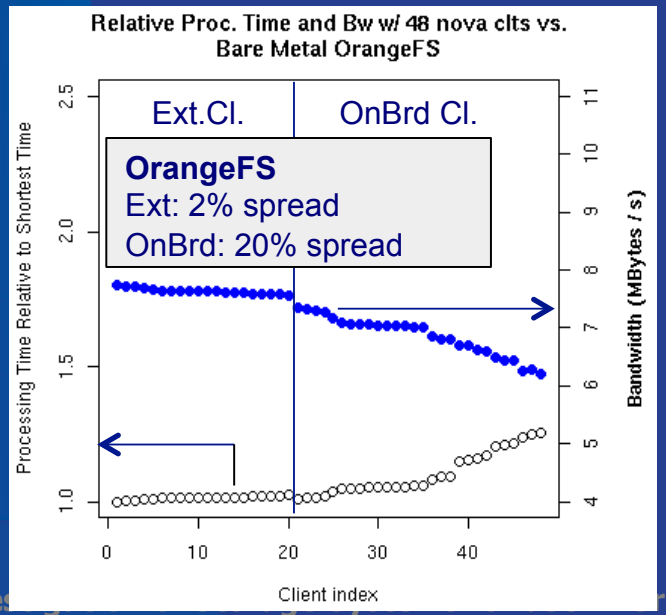
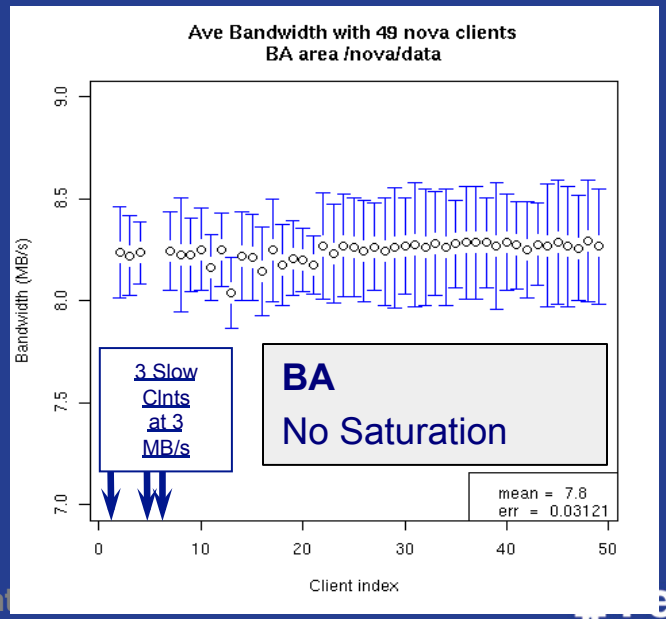
Storage	Benchmark	Read (MB/s)	Write (MB/s)	Notes
Lustre	IOZone	350	250 (70 on VM)	
	Root-based	12.6	-	
Hadoop	IOZone	50 - 240	80 - 300	Varies on num of replicas
	Root-based	7.9	-	
BlueArc	IOZone	300	330	Varies on sys. conditions
	Root-based	8.4	-	
OrangeFS	IOZone	150-330	220-350	Varies on num name nodes
	Root-based	8.1	-	

Conclusions

- Lustre on Bare Metal has the best performance as an external storage solution for the root skim app use case (fast read / little write). Consistent performance for general operations (tested via iozone).
  - Consider operational drawback of special kernel
  - On-board clients only via virtualization, but Srv. VM allows only slow write.
- Hadoop, OrangeFS, and BA have equivalent performance for the root skim use case.
- Hadoop has good operational properties (maintenance, fault tolerance) and a fast name srv, but performance is NOT impressive.
- BA at FNAL is a good alternative for general operations since it is a well known solution professionally managed.

EXTRA SLIDES

U.S. DEPARTMENT OF ENERGY

Fermilab

Bandwidth Fairness - Comparison