TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

VIENNA SCIENTIFIC CLUSTER

# Performance Analysis of the Fraunhofer Parallel File System over ZFS

Dr. Ernst Haunschmied
Dr. Michael Alexander

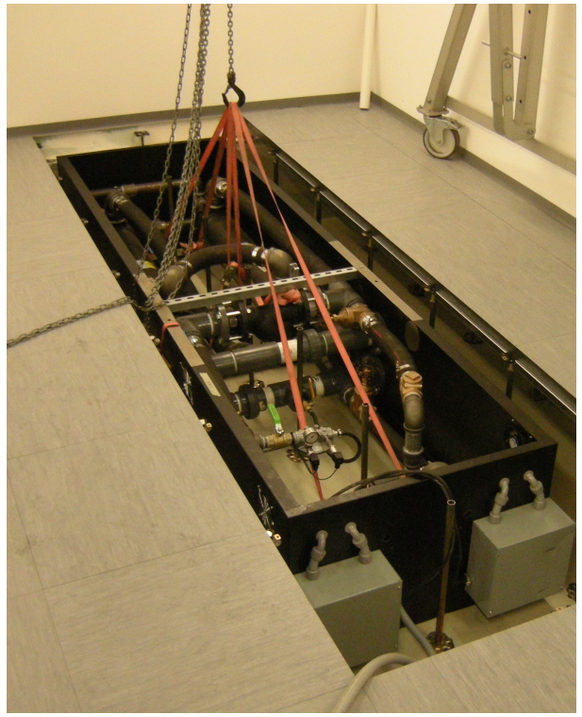
Vienna Scientific Cluster, TU Wien

# Agenda

- Who we are
- ZFS Use Cases
- Disclaimer
- Questions, Disclaimer
- Environment
- FhGFS over XFS vs. ZFS
- Multiple Test Cases
- ZFS over FhGFS with Multiple Parameters
- Capabilities of Interest
- Acknowledgements
- Conclusion and Further Work

# Who we are

- Vienna Scientific Cluster
  - Compute and Storage for multiple Austrian Universities and Research Establishments
- 3 Clusters
  - Mostly standard HPC codes over MPI
  - VSC-3 being built as-we-speak
    - >2000 nodes, largest oil-cooled deployment in Europe



# In-Scope ZFS Use Cases and ZFS Relevance

| | Reliability | Bit-Rot | Manage-ability | Compres-sion & Dedup | Perfor-mance * |
|---|---|---|---|---|---|
| Archive using spinning disk filesystem | H | H | H | H | M |
| Archive staging disk | L | H | H | L | H |
| Scratch disk at compute | L | L | L | L | H |
| User data backup | H | H | H | H | L |

Composite workload of large block, n>> small file-IOPS and multi-stream. Lower weight: non-contiguous/non-sequential and shared file metrics

# Questions

- Using FhGFS 2014.01-r5 over ZFSoL 0.6.2:
  - Is ZFSoL alone and in a stack with FhGFS stable?
  - Do first performance measurements indicate suitability to use cases?

# Disclaimer

- Non-optimized
- 2 out of 3 workloads synthetic
- Work-in-progress
- Early results as a snapshot

# Environment 1

- 1 server node 1xE5-1620 8 core/3.7 GHz/132 GB
  - Metadata, mgt., admon and
  - 1 or 2 storage daemons on the same node
  - 2xQDR IB
  - SL 6.4
  - 2+1 controller, 36xSAS, 36xSATA, 2 SATA SSDs
  - SSD Metadata store for n>> file tests
- Single and dual FhGFS storage daemons
- 8 Clients
- FhGFS 2014.01-r5, ZFSoL 0.6.2

# Environment 2

- 4 Sun Fire X2270 compute nodes
  - Node 1 Metadata, mgt., admon
  - Node 2 FhGFS over ZFS storage
  - Node 3 FhGFS over XFS storage
  - Node 4 FhGFS client, 2nd Metadata sever for out-of-scope mirroring tests
  - 1xE5-1620 8 core/3.7 GHz/132 GB
  - SL6.4
  - QDR IB same switch
  - No RAID, 2 data disks for ZFS with copies=2 (mirroring)
- FhGFS 2014.01-r1, ZFSoL 0.6.2

# Environment 3

- 1 AMD server node E5620/8 core/2.4 GHz/100GB
  - SL 6.5
  - LSI Raid 5 w/ 24 disk
- ZFS only (no FhGFS)

# FhGFS/XFS vs. FhGFS/ZFS vs. XFS, ZFS Native Single-Disk, No RAID

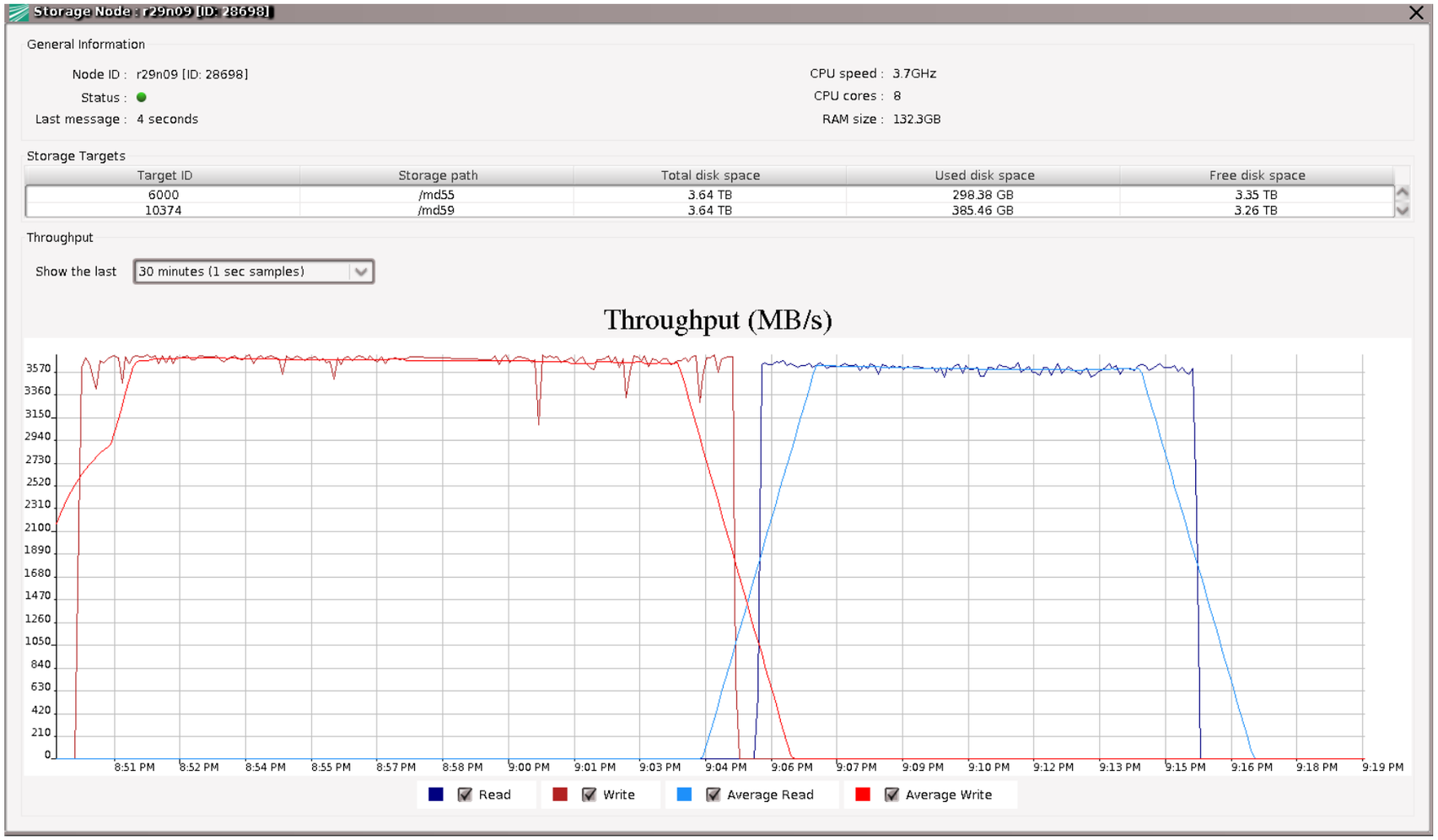
- Early gauge of combinations
- Best predictive validity on scaled tests with high file sizes

| | FHGFS/ XFS Remote | FHGFS /XFS Local Node | ZFS FHGFS/ Remote | ZFS FHGFS Local Node | ZFS Local Node | XFS Local Node | ZFS 1 Disk copies=2 Local Node | Remark |
|---|---|---|---|---|---|---|---|---|
| | MiB/s | MiB/s | MiB/s | MiB/s | MiB/s | MiB/s | MiB/s | |
| **Write 8GB** | 223 | 181 | 222 | 226 | 89,35 | 171 | 39,55 | 2E33 Byte = 8GB with base 1k=1024<br>sync; echo 3 > /proc/sys/vm/drop_caches;<br>time dd if=/dev/zero of=testfile bs=2048 count= $[2**22]; |
| **Read 8GB** | 1000 | 1000 | 1000 | 1000 | 113 | 170 | 643,5 | 2E33 Byte = 8GB with base 1k=1024<br>sync; echo 3 > /proc/sys/vm/drop_caches;<br>time dd if=testfile of=/dev/null bs=2048 count= $[2**22]; |
| **Write 10E5 1k Files** | 0,17 | 0,16 | 0,18 | 0,19 | 0,17 | 0,19 | 0,19 | mount -o remount,size=1k /dev/shm;<br>dd if=/dev/urandom of=/dev/shm/ctf1k bs=1k count=1;<br>sync; echo 3 > /proc/sys/vm/drop_caches;<br>time ( i=1;while [ $i -le 100000 ]; do \cp -f /dev/shm/ctf1k /bfszfs/ctf1k_$i; ((i++)); done ); |
| **Read 10E5 1k Files** | 0,19 | 0,18 | 0,20 | 0,16 | 0,17 | 0,22 | 0,18 | sync; echo 3 > /proc/sys/vm/drop_caches;<br>time ( i=1;while [ $i -le 100000 ]; do \cp /bfsxfs/ctf1k_$i /dev/null; ((i++)); done ); |
| **1k File creates/s** | 173,71 | 159,38 | 193,51 | 186,80 | 169,08 | 201,76 | 189,04 | 10E5/time (Write 10E5 1k Files) |

| | FHGFS /XFS Remote | FHGFS /XFS Local Node | ZFS FHGFS / Remote | ZFS FHGFS Local Node | ZFS Local Node | XFS Local Node | ZFS 1 Disk copies =2 Local Node | Remark |
|---|---|---|---|---|---|---|---|---|
| | MiB/s | MiB/s | MiB/s | MiB/s | MiB/s | MiB/s | MiB/s | |
| **Write 16GB File** | 142,33 | 175 | 140 | 104,5 | 86,8 | 147 | 70,4 | 2E34 Byte = 16GB with base 1k=1024<br>sync; echo 3 > /proc/sys/vm/drop_caches;<br>time dd if=/dev/zero of=testfile bs=2048 count=<br>$[2**23]; |
| **Read 16GB File** | 361 | 194 | 1000 | 106 | 102 | 113 | 109 | 2E34 Byte = 16GB with base 1k=1024<br>sync; echo 3 > /proc/sys/vm/drop_caches;<br>time dd if=testfile of=/dev/null bs=2048 count=<br>$[2**23]; |
| **Write 128GB** | 88,8 | 100 | 117 | 38,5 | 81,5 | 95,5 | 38,6 | 2E37 Byte = 128GB with base 1k=1024<br>sync; echo 3 > /proc/sys/vm/drop_caches;<br>time dd if=/dev/zero of=testfile bs=2048 count=<br>$[2**26]; |
| **Read 128GB** | | | 100,75 | 98,1 | 95,9 | 93,8 | 98,2 | 2E37 Byte = 128GB with base 1k=1024<br>sync; echo 3 > /proc/sys/vm/drop_caches;<br>time dd if=testfile of=/dev/null bs=2048 count=<br>$[2**26]; |
| **Write 839GB 629GB for ZFS copies=2** | 74,8 | | 87,20 | | 73,5 | 92,2 | 164 | sync; echo 3 > /proc/sys/vm/drop_caches;<br>time dd if=/dev/zero of=testfile bs=1M<br>count=800000 |
| **Read 839GB 629GB for ZFS copies=2** | | | | | 88,70 | 92,80 | 180,00 | sync; echo 3 > /proc/sys/vm/drop_caches;<br>time dd if=testfile of=/dev/null bs=1M count=800000 |

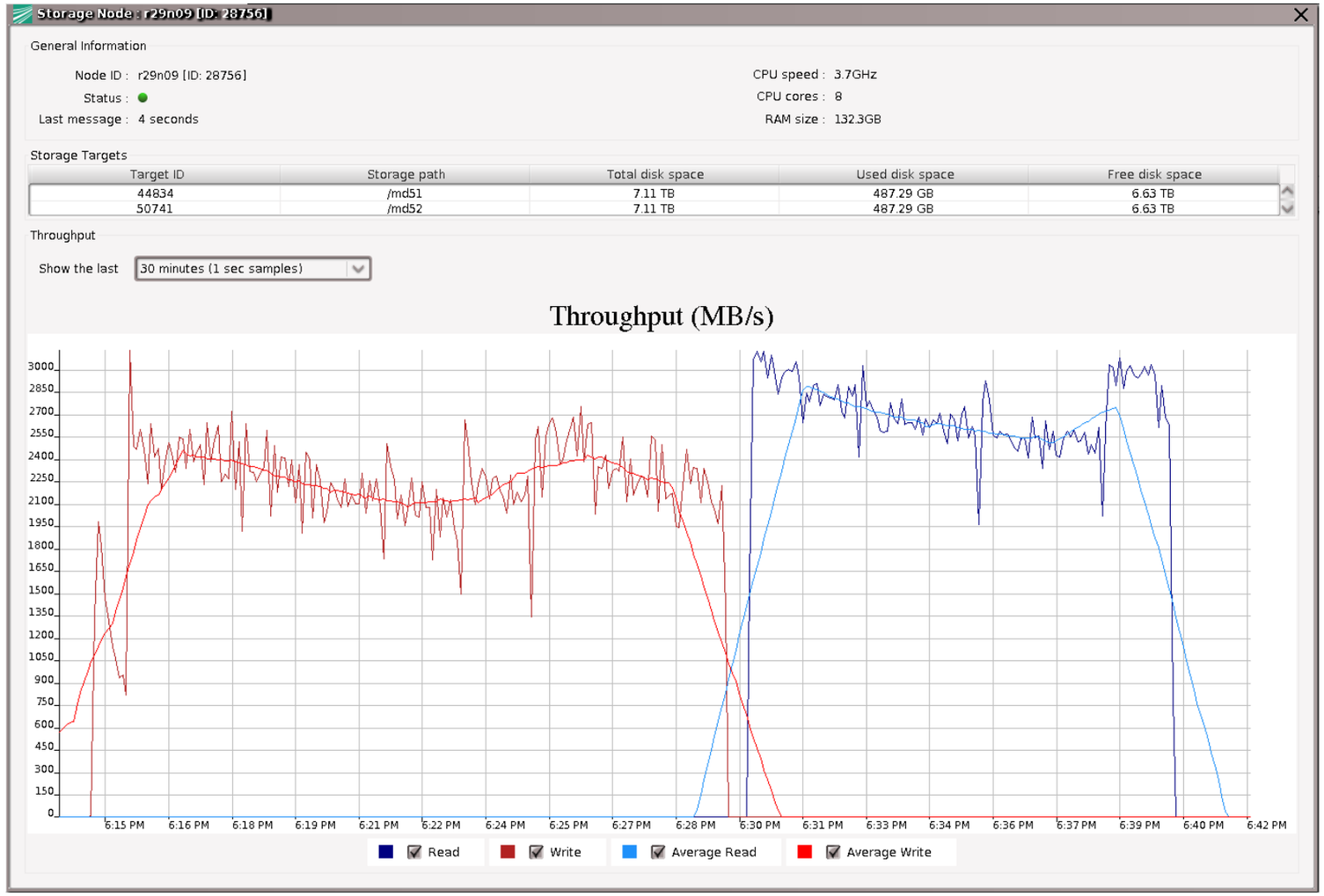
# Baseline 9 disk RAID0 XFS Straight

- Aggregate peak time-series ~3.6 GB/s write/read



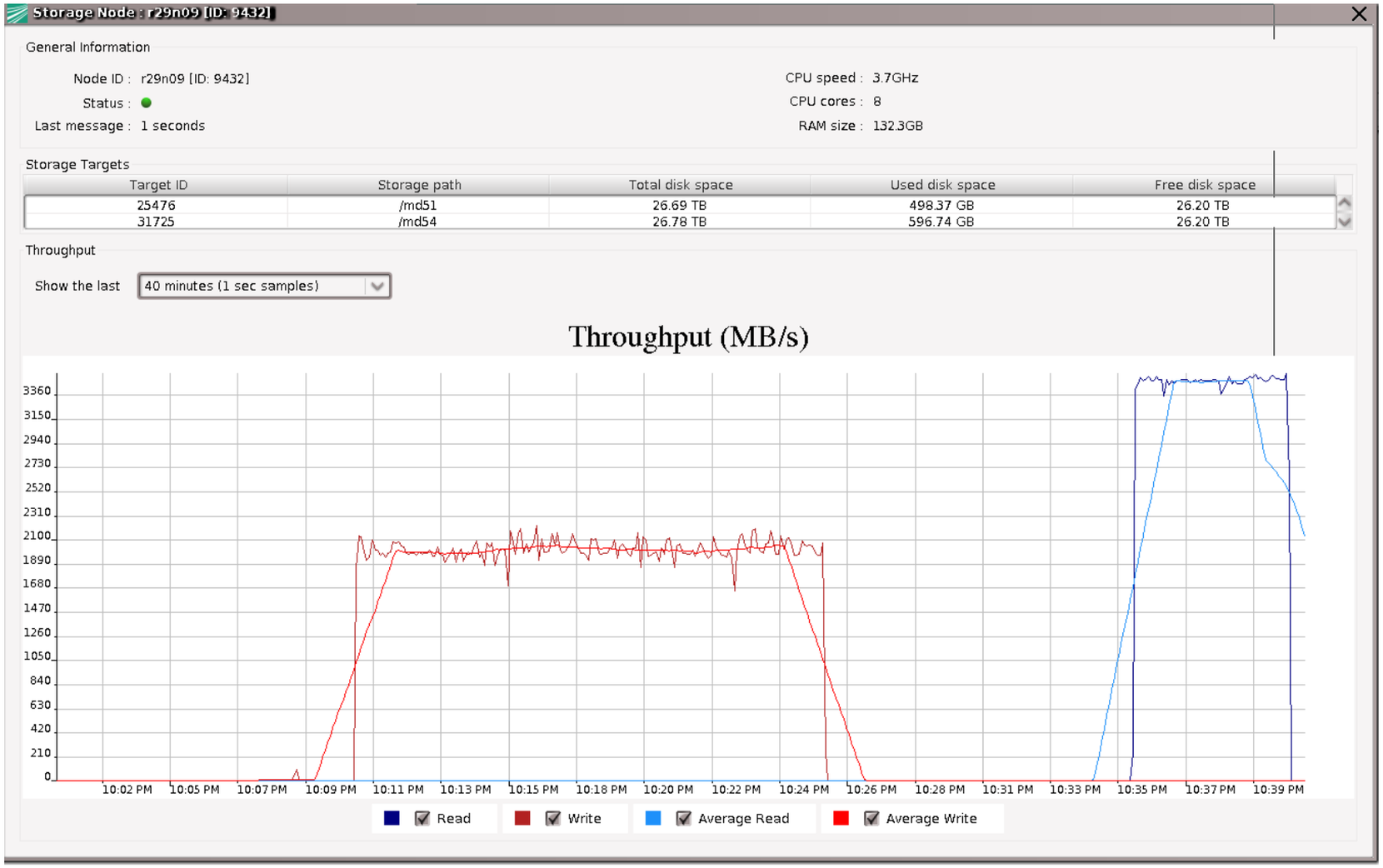
# RAIDZ(1) 4 Tanks à 9 disks

- Aggregate peak time-series ~2.7GB/s read ~2.2 GB/s write



# RAIDZ(1) 4 Tanks à 9 disks

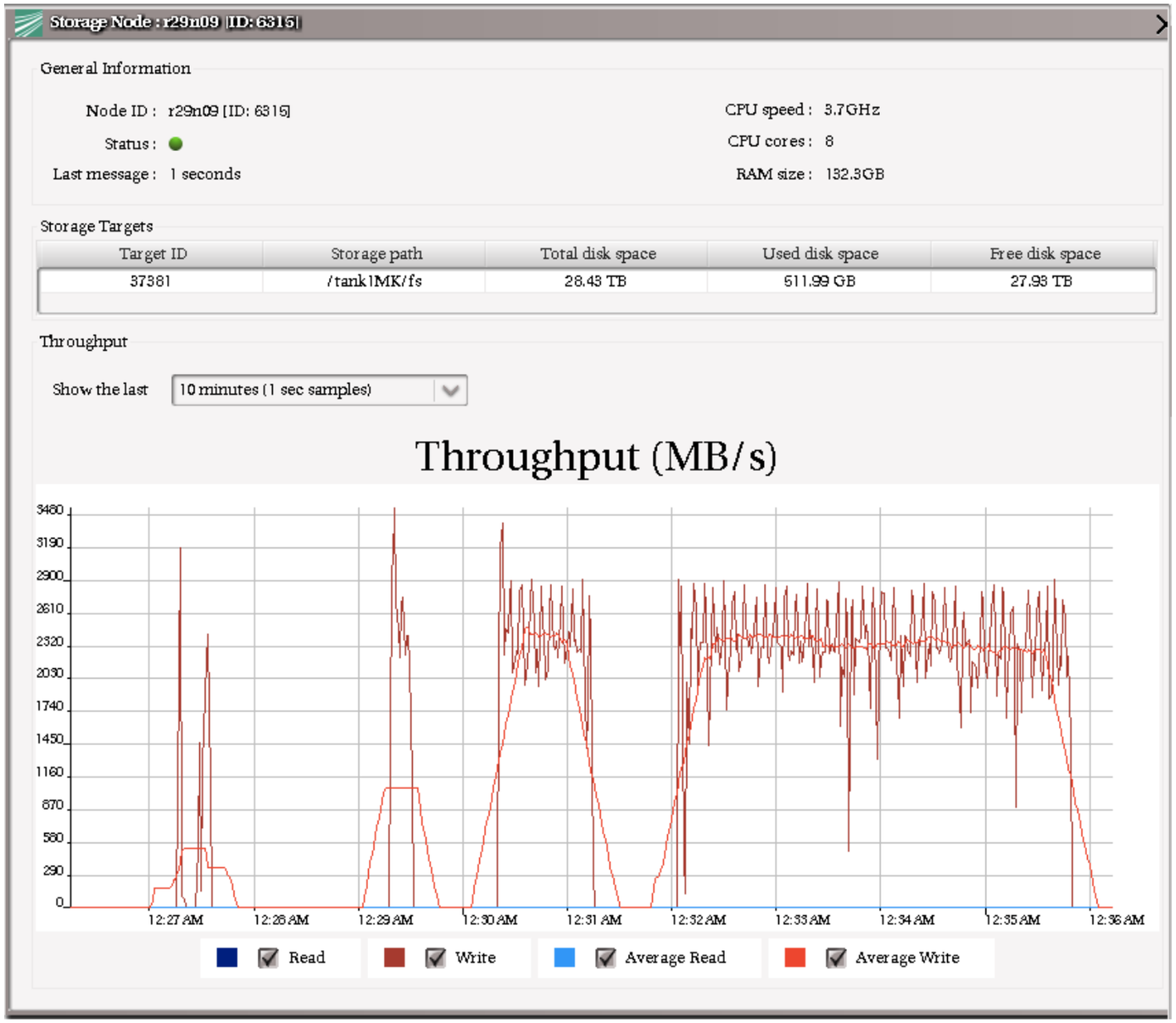
- Aggregate peak time-series ~3.4GB/s read ~2. GB/s write



# Stripe of 4 x 8+1 disk RAIDZ(1) Groups

- pdsh -w b[181-188] 'sync; echo 3 > /proc/sys/vm/drop_caches;'
- pdsh -w b[181-188] 'time dd if=/dev/zero of=/mnt/fhgfs-ib0/testfile_`hostname` bs=2048k count=$[2**13]'

# Stripe RAIDZ(1) Groups 4 x 9 disk




- 2,4,16, 64 GB x 8 clients

# Stripe RAIDZ(1) Groups 4 x 9 disk II

- ~2.3GB/s aggregate peak time-series
- fhgfs-ctl --storagebench --alltargets --write --blocksize=512K —size={2,4,16,64}G —threads=1
- fhgfs-ctl --storagebench --alltargets --status

| size [G] | 2 | 4 | 6 | 16 | 64 | 128 |
|---|---|---|---|---|---|---|
| Aggregate throughput [MB/s] | 3207 | 2605 | 2613 | 2202 | 2234 | 2259 |

# Stripe RAIDZ(1) Groups 4 x 9 disk III

- Command line used: IOR -w -N 16 -b 2m -s 150000 -t 512k -F -k -E -D 100 -d 30
- Summary:

  api = POSIX

  test filename = testFile

  access = file-per-process

  ordering in a file = sequential offsets

  ordering inter file= no tasks offsets

  clients = 1 (1 per node)

  repetitions = 1

  xfersize = 524288 bytes

  blocksize = 2 MiB

  aggregate filesize = 292.97 GiB

  Using stonewalling = 100 second(s)
- Max Write: 1057.38 MiB/sec

# Stripe RAIDZ(1) Groups 4 x 9 disk IV

- Write 10E5 1k files
  - mount -o remount,size=1k /dev/shm;
    dd if=/dev/urandom of=/dev/shm/ctf1k bs=1k count=1;
    sync; echo 3 > /proc/sys/vm/drop_caches;
    time ( i=1;while [ $i -le 100000 ]; do \cp -f /dev/shm/ctf1k /mnt/fhgfs-ib0/ctf1k_$i; ((i++)); done );
  - file write creates/s (L2ARC&ZIL SSD) 115 (150) ; 110kB/s (115kB/s)
- Read 10E5 files
  - 173 (165) file write creates/s (L2ARC&ZIL SSD) ; 170 (170) kB/s

# Large Stripes FhGFS over ZFS vs. XFS

- 72 disks, 36 SATA, 36 SAS same type
  - ST3000VX000
    - 3TB SATA spec average data rate read/write 156 MB/s
    - sync; echo 3 > /proc/sys/vm/drop_caches; hdparm -f /dev/sdal; hdparm -t /dev/sdal 198MB/s
  - MK1001TRKB
    - 1TB SAS no Toshiba spec transfer rates
    - sync; echo 3 > /proc/sys/vm/drop_caches; hdparm -f /dev/sdal; hdparm -t /dev/sdal 140MB/s
- ZFS/Kernel/FhGFS not optimized

# User Data Backup

- Approx 152M files Cluster2
- ZFS (with SSD L2ARC/ZIL) only
- LSI RAID6 w/ 24 disk SAS
- rsync create/rewrite
- Compression
  - Measured aggregate factor 1.49!!, high variance
- Deduplication
  - Preliminary partial tree measured factor 1.08, high variance - not all files deduped

# ZFS Issues Observed

- ZFS create pool using /dev/device reboot async udev device name reordering
  - known issue, probably not a bug, work-around create by e.g. UUID or
    - SL 6.4 does not create disk-by-UUID by default
- Memory leak-OOM condition on large cp ZFS-only
  - Attempt to reproduce with instrumentation

# FhGFS Issues in Conjunction with ZFS Observed

- None yet!!

# Capabilities of Interest

- End-to-end data path checksums
  - Without multiple computation
- *n* PB storage RAID economics, rebuild time issues
  - Erasure codes et al. in the distributed file system
- Enhanced joint *n*>> small file performance
  - Towards ZFS meta data store != data disks

# Acknowledgements


- **Ernst Haunschmied**
- Dieter Kvasnicka
- Jan Zabloudil

# Conclusion and Further Work

- So far **ZFS** (the test focus) over FhGFS has shown better than expected maturity for a 0.6.2 release and acceptable levels of performance
  - Against arguably low expectations!!
- More measurements needed
- Silent data corruption testing
  - Tool to be posted in June http://github.com/malexandvsc
- Much more negative testing within scope of use-cases
- Optimization
  - Queue of not applied optimizations
- Test automation