Building up a high performance data centre with commodity hardware

using the example of the DESY-ZN Atlas SE

Andreas Haupt
DESY – DV –

Spring Hepix 2010
Lisbon, 2010-04-20

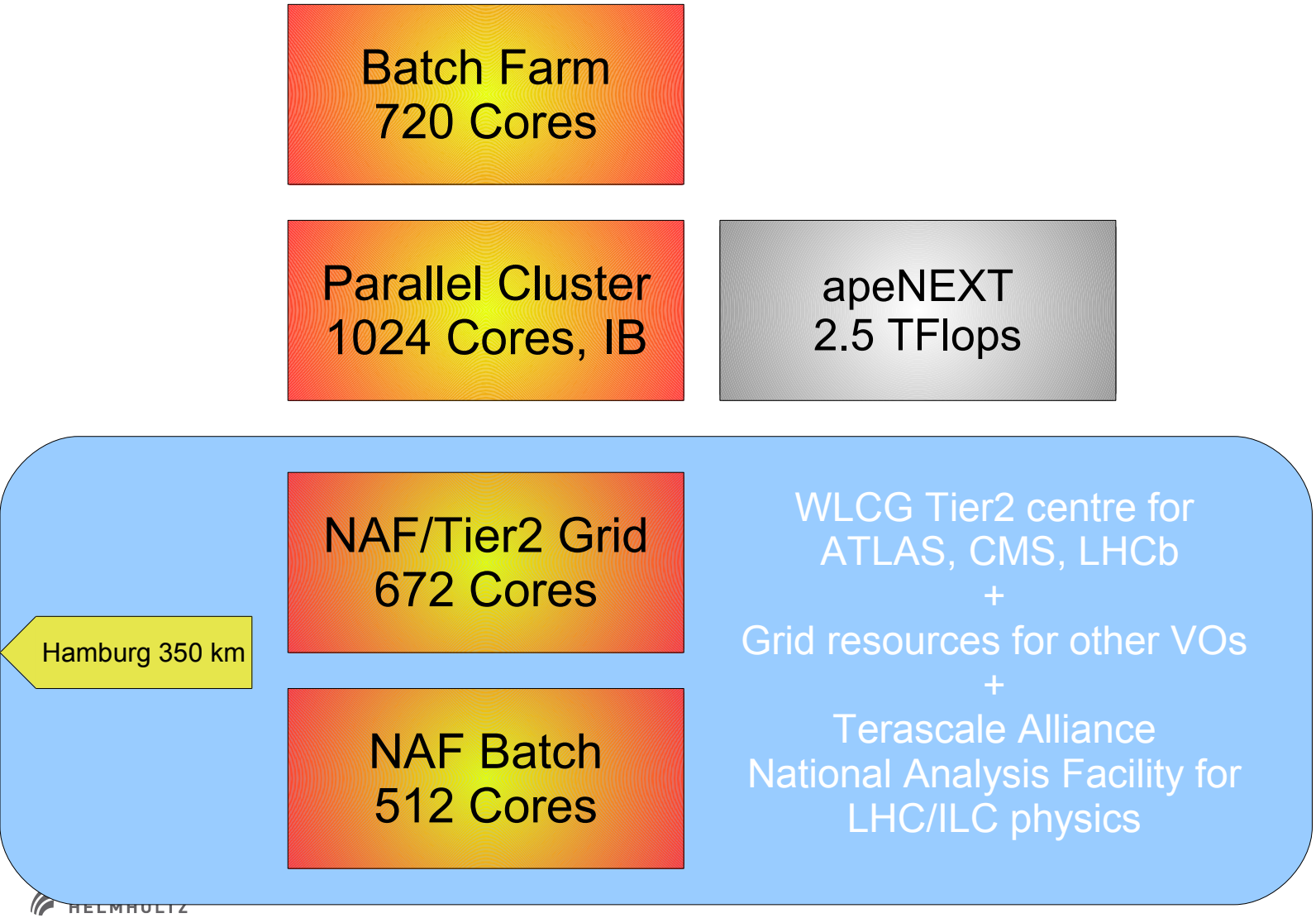
HELMHOLTZ GEMEINSCHAFT

DESY

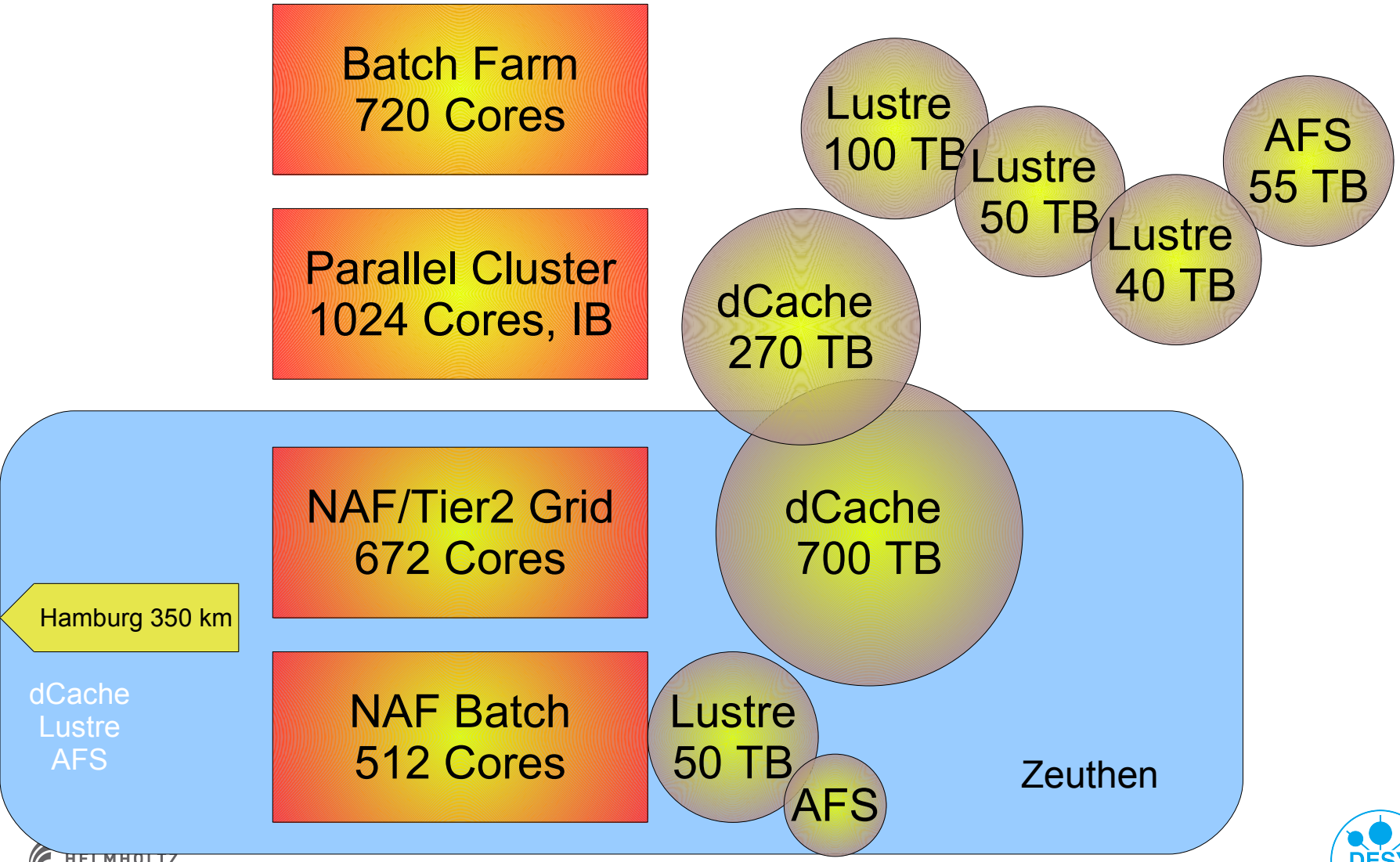
Outline

- Computing at DESY in Zeuthen
- Hardware used to build up the Atlas SE in Zeuthen
- Optimising the hardware
- Optimising the dCache service
- Example benchmark plots

Computing at DESY in Zeuthen



Computing + Disk Storage at DESY in Zeuthen



Describing the deployed hardware

- DESY-wide server advertisement two years ago
  - One winner: DELL
- DELL main server hardware vendor for DESY
  - but: comparable server models also offered by other vendors
- Hardware runs smoothly
  - Rather low defect rate
  - But: some server models show higher defect rates than other

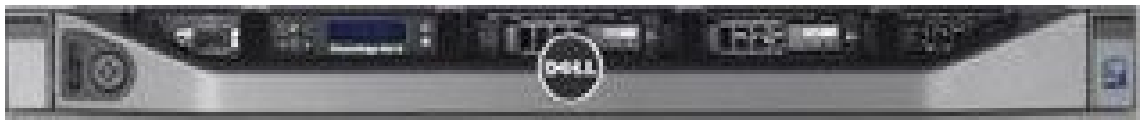
Storage bricks: PowerVault MD1200

- JBOD with 12 x 3,5” SATA / SAS disks
- 2 HU
- 2 redundant SAS connectors
- Cascadable

Storage bricks: PowerVault MD1000

- JBOD with 15 x 3,5” SAS / SATA disks
- 3 HU
- 2 SAS connectors
  - split mode: connect more than one host (but no shared storage)
  - unified mode: redundant SAS connection, static load balancing (both links active)
- Cascadable
- Currently our work horse

Storage bricks: PowerEdge R610



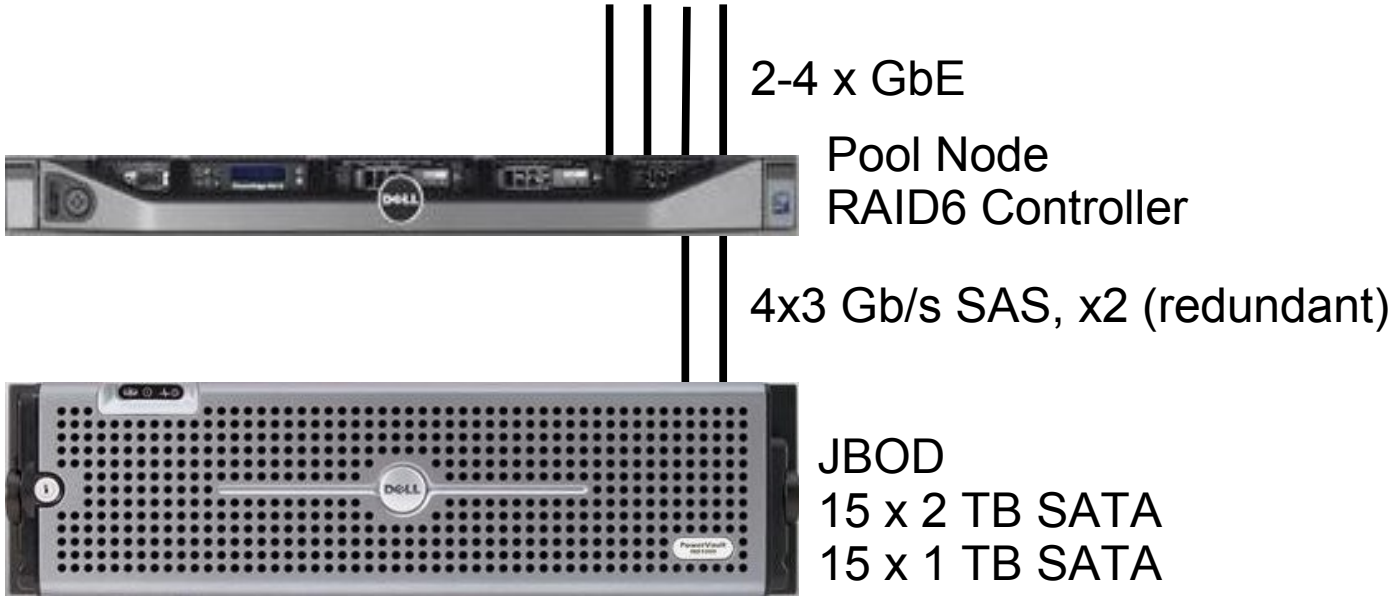
- 1 HU rack mount server
- 4 x 1GBit NIC
  - 10 GE NIC optional
- Raid-6 capable Perc6/e or H800 controller managing the MD1000/1200 storage jbods
  - Supports (adaptive) readahead
  - Dual port: able to connect more than one jbod
  - Battery backed writeback cache

Storage bricks: PowerEdge R510

- classical “storage in a box”
- 2 HU rack mount server
- 12 x 3,5” SATA / SAS disks
- H700 Raid-6 capable controller
- just 2 x 1GBit NIC
  - 10GE NIC optional
- Highest storage density of Dell's currently available standard hardware
  - 10 TB / HU net when filled with 2 TB SATA disks

The basic storage module (1)

- Direct Attached Storage: typical configuration:



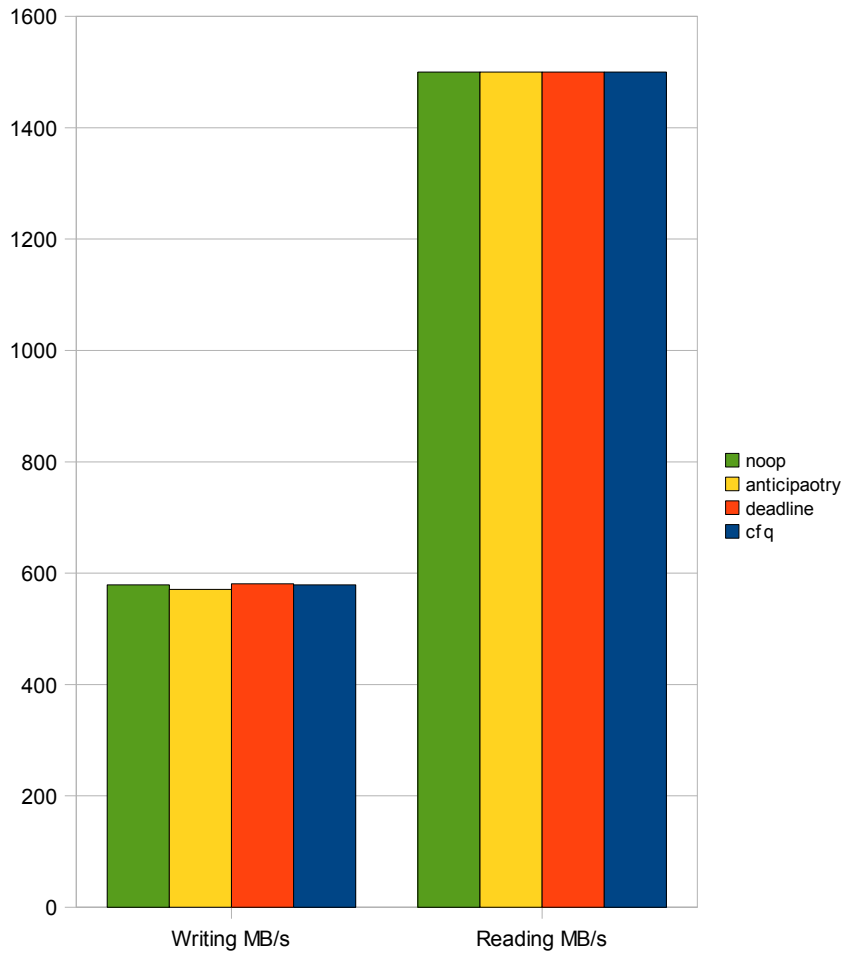
- OS: SL5 x86_64
  - xfs filesystem
  - automatic, central installation, configuration, maintenance and monitoring
  - like all other hosts (systems are fully patched)

The basic storage module (2)

- Used as dCache pool node, Lustre OSS, AFS Fileserver
- Maximum network wire speed bonded: 440 MB/s
- Maximum capacity
  - SATA (2 TB disks): 30 TB gross, 26TB net (Raid-6)
  - SAS (600 GB disks): 9 TB gross, 7,8TB net (Raid-6)
- Maximum network throughput
  - ~ 17 MB/s per TB (SATA)
  - ~ 56 MB/s per TB (SAS)
- Price: ~ 30 cent / GB net (purchase only)

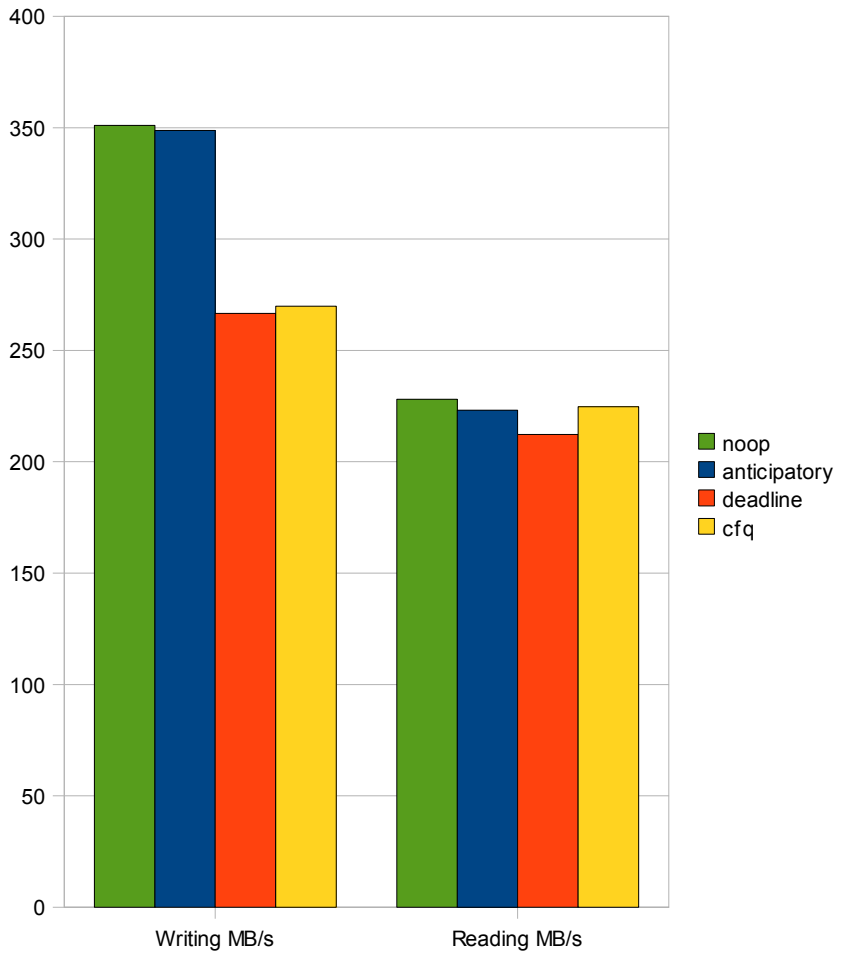
Changing the IO scheduler (1)

- Default IO scheduler (cfq - “completely fair”) was buggy and without good performance in SL5
  - This changed at some time ...
- Comparing the transfer rates at the usual access pattern can help improving the overall performance
- Statistic: compare writing / reading speed with single data stream:
  - dd if=/dev/zero of=/mnt/file1 bs=128k count=256k
  - dd if=/mnt/file1 of=/dev/null bs=128k
- With single data stream no difference

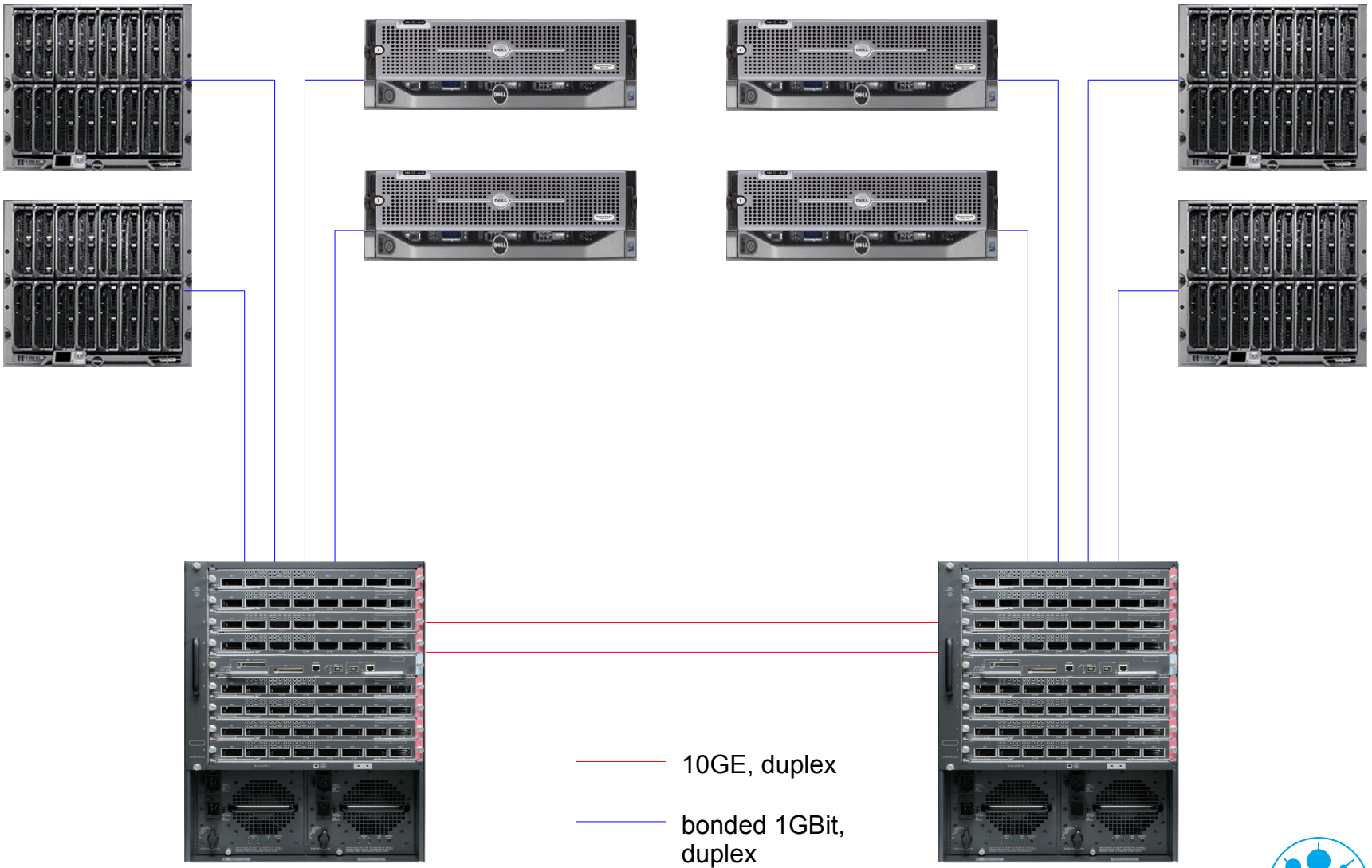


Changing the IO scheduler (2)

- Now do the same but 16x in parallel:
  - for i in {1..16}; do dd if=/dev/zero of=/mnt/file1 bs=128k count=256k & done
  - for i in {1..16}; do dd if=/dcache/data.32g.$i of=/dev/null bs=128k & done
- Breakdown in overall performance
  - Reading speed affected even worse than writing
  - When writing changing the io scheduler can result in performance gains
- SL 5.5 kernel comes with scheduler changes (at least CFQ)
  - No measurements yet



Network layout



Optimal io performance: some thoughts

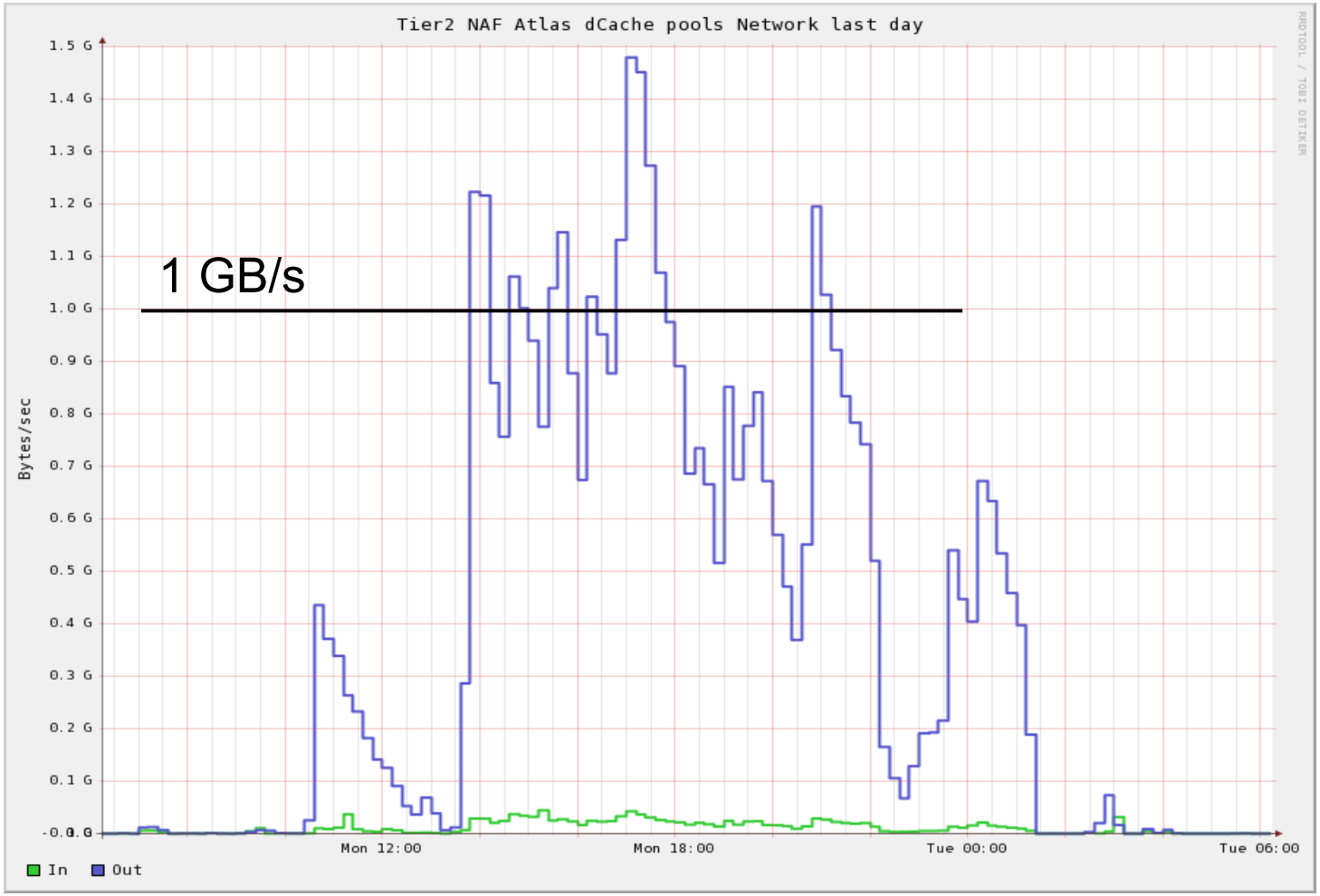
- Disk-io is the main bottleneck
  - spread the data over as much disks as possible
- Network-io is another bottleneck
  - place the data near the client it will be read from
  - avoid other network related bottlenecks like firewalls
- Data written at the same time tends to be read at the same time also
  - avoid situations when data can (or would) be written only to a little number of nodes
- In case of SAN / appliance solutions the data distribution might be managed “low level”
  - In our case this has to be done at application level (i.e. dCache)

Optimising the dCache

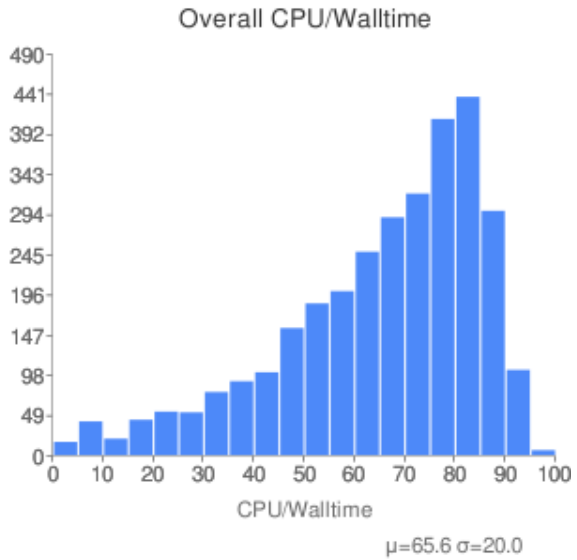
- Example: Atlas dCache SE (current available space: 550TB)
- dCache organises its data in pools
  - We set up one pool per storage unit
- Atlas space is organised is so called “Space Tokens”
  - MCDISK, DATADISK, SCRATCHDISK, ...
  - The size of the different tokens is not fix – Atlas requests change from time to time
- Choose the approach of spreading all space tokens over all dCache pools
  - Maximum performance for any of them

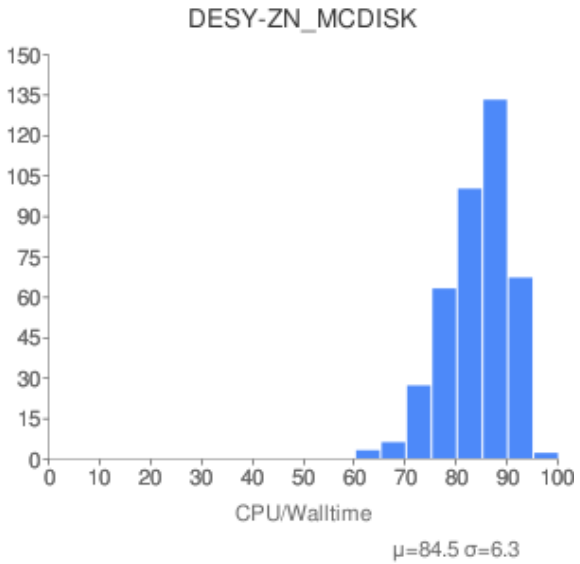
ATLAS Hammercloud GridKaCloud Test, 2nd March 2010

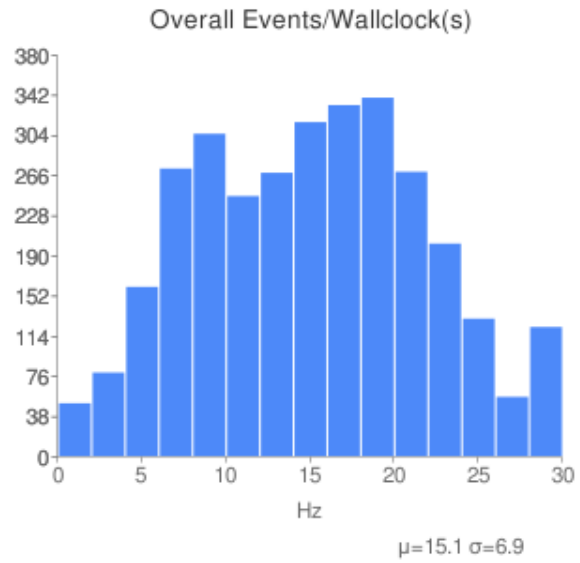
- dCache throughput

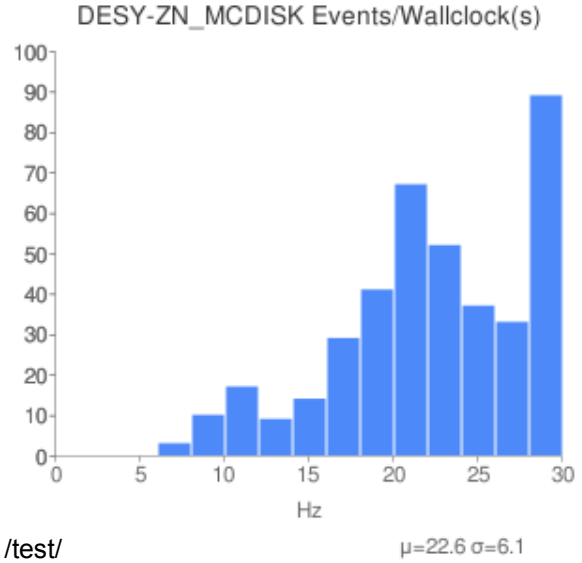


ATLAS Hammercloud GridKaCloud Test, 2nd March 2010



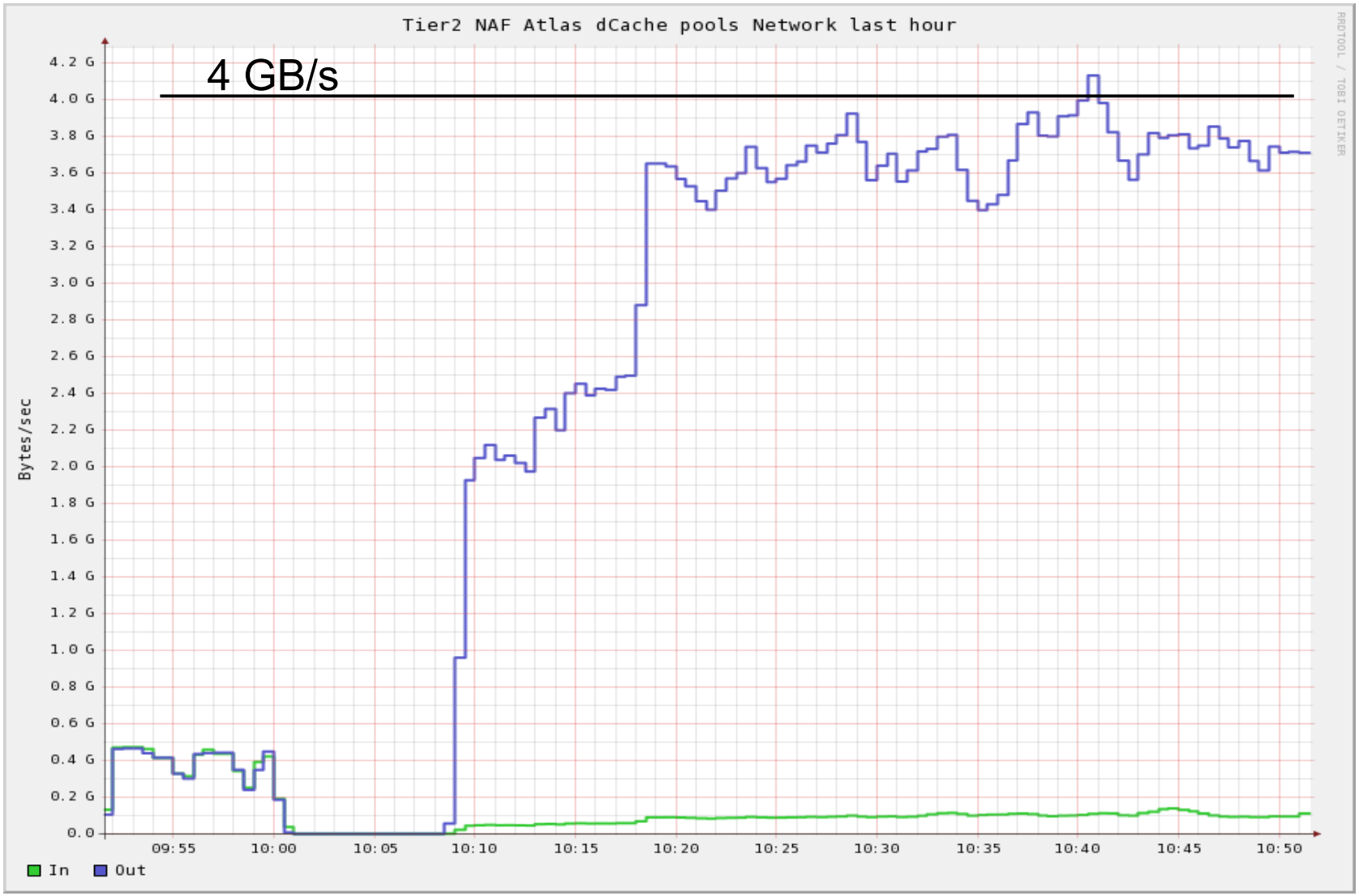






http://gangarobot.cern.ch/hc/1131/test/

dCache stress test



Summary

- DESY operates several comping farms and a range of storage systems (AFS, dCache, Lustre)
- DESY ZN currently concentrates on Dell commodity hardware
- Some low level tweaks (e.g. changing the io scheduler) may result in performance gains
- All in all the current approach scales very well

Thank you for your attention!