Testing storage infrastructure for SuperB use cases

Giacinto Donvito – INFN-Bari
Armando Fella – INFN-Pisa
Silvio Pardi – INFN-Napoli

Outline

- Main objective
- Tests infrastructure
- CMS analysis jobs test
  - Test description and performance
- SuperB analysis jobs test
  - Test description and performance
- Plans and future works

Main objective

- Testing and exploiting new emerging technologies in order to find possible storage solution for SuperB data access
- Starting from experience carried on in other HEP experiments
- The results should be useful for both site admin and framework developers

Storage Systems under Test

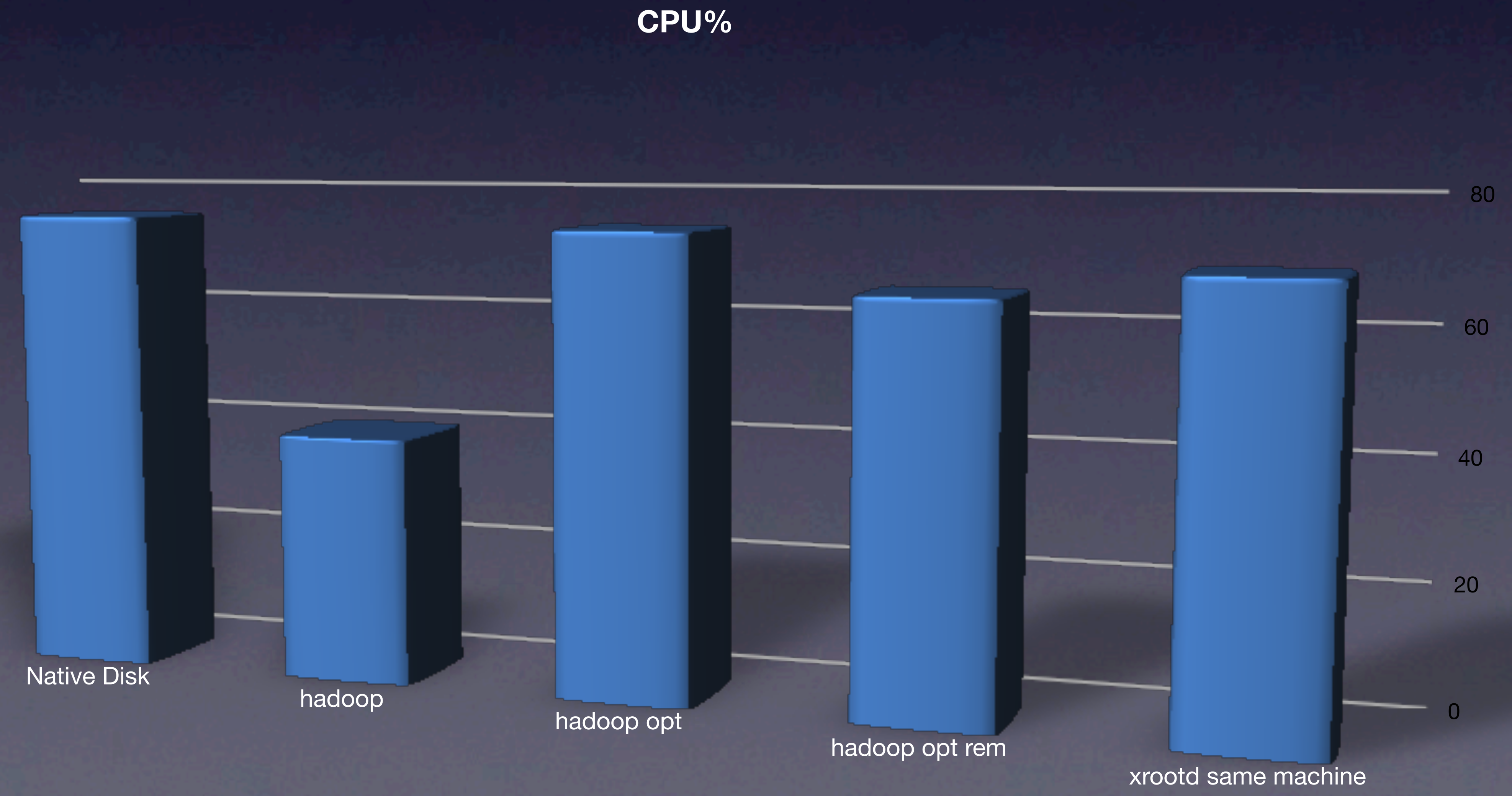
- Server:
  - Lustre 2.0:
    - 3 **RAID5 FS**. Stripe-unit size: 128 KB. **5 Data disk each**
  - Xrootd 3.0.0:
    - 13x1TB **single disks**. EXT3 FS
  - hadoop-0.20.2 (from http://newman.ultralight.org/)
    - 13x1TB **single disks**. EXT3 FS

- Clients:
  - SLC5.4 kernel 2.6.18-194.11.3
  - Fuse: fuse-libs-2.7.4-8
  - FUSE mount on the client (rdbuffer=32768)

Optimising the Single job

- “Hadoop opt”=> rdbuffer=32768
- The CMSSW (cacheHint,readHint,cacheSize) tuning parameters are always used and tested until the best result is found
- “blockdev --setra” on each drive, was tuned in order to find the best solution
- **Lustre** is not reported plot, but it was **83% of CPU efficiency**

- It is possible to obtain the same performance with up **to 4-5 concurrent job** per **single native disk**



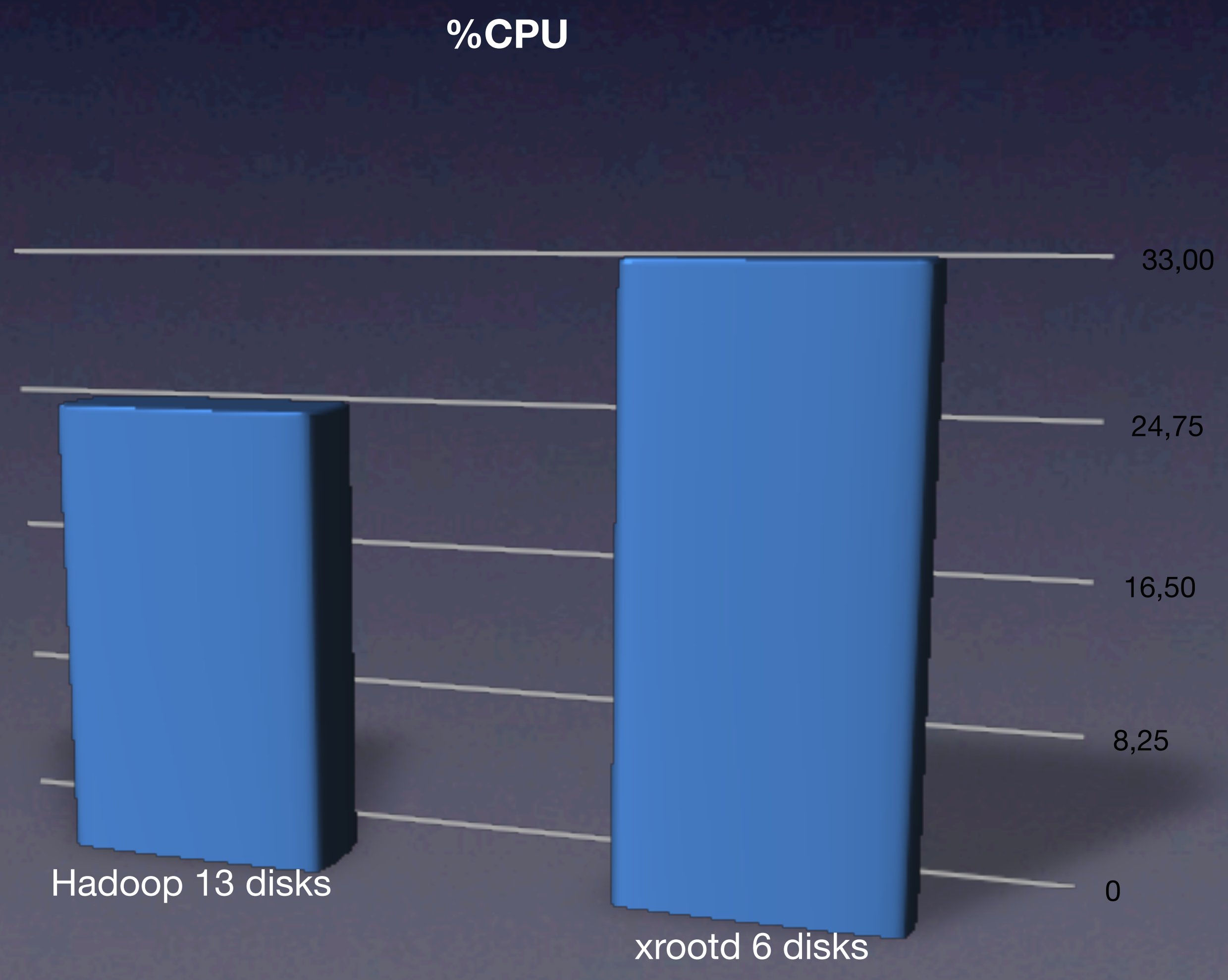
Performance Tests

- up to 116 concurrent jobs
- production farm used to run the jobs
- Each file on the server is used only by a single job
  - There is no “concurrency” on each file
- A single disk server:
  - 10Gbit/s network card
  - deep network testing to assure there are no network bottleneck
  - >400MB/s measured disk-to-network bandwidth

Performance test: hadoop vs xrootd

- Running 56 concurrent jobs
- Using **6 disks** for xrootd
- Using **13 disks** on hadoop installation
  - Reading data using "fuse optimized"
  - Single server: no "block replica"

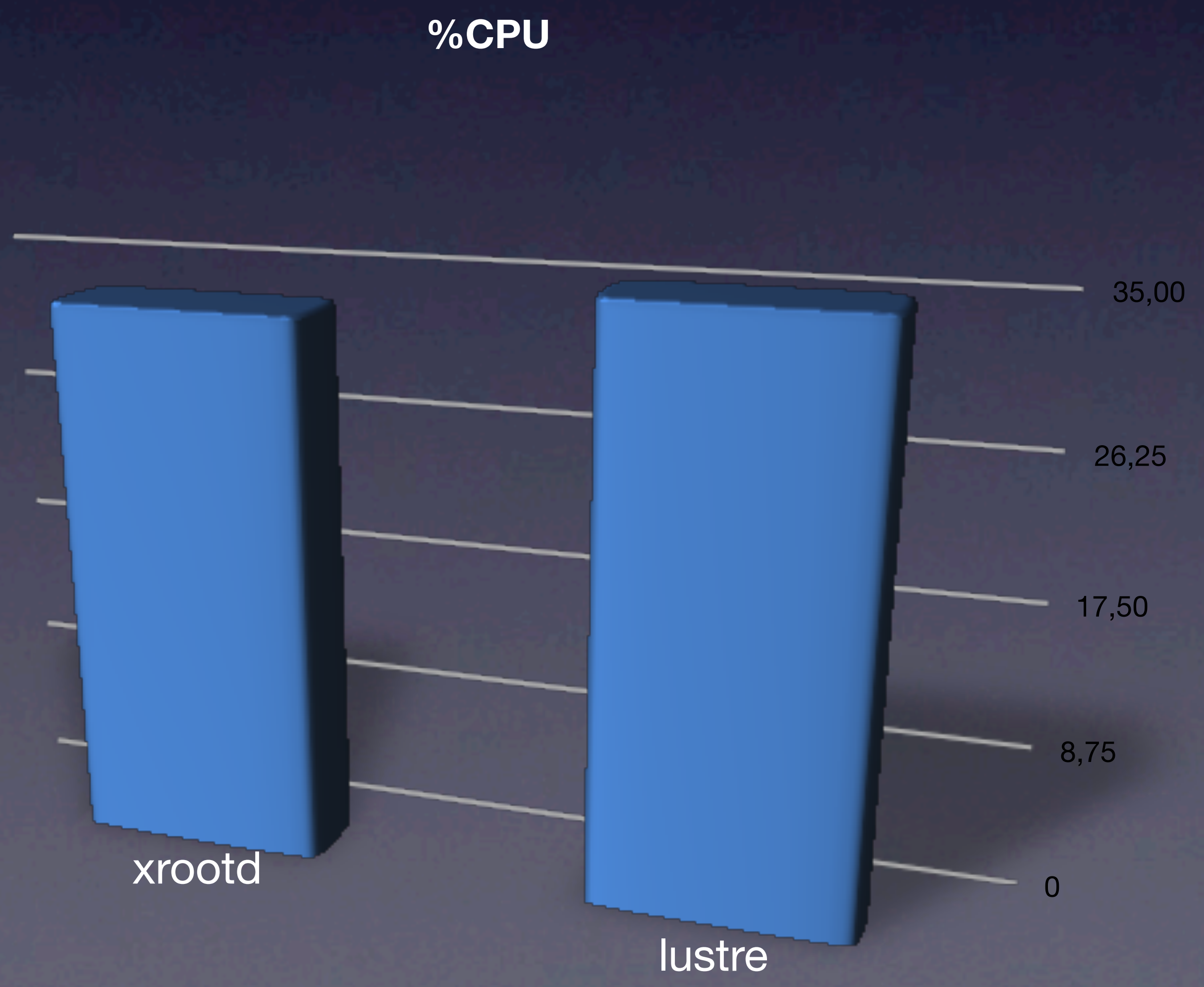
- We have observed huge load on the server while running "hadoop test"
  - tuning "blockdev setra" did not improve the situation
  - increasing the memory for java produced only small improvements



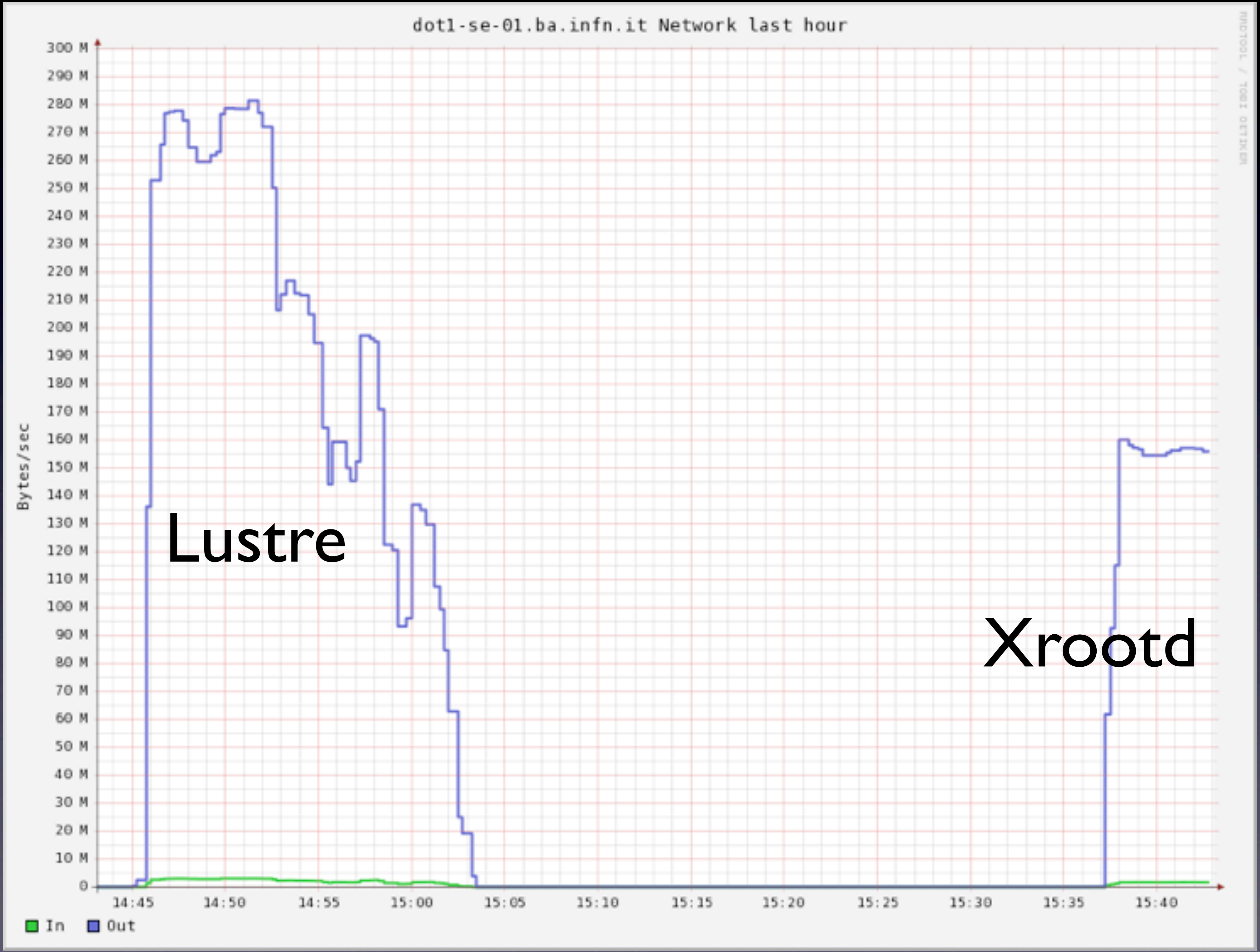
Performance Tests: lustre vs xrootd

- Running 116 concurrent jobs
- Reading ~1TB of data
- Always measuring the CPU efficiency
    - This is an interesting parameter both from user's point of view and from a site admin

- The network usage of the two solution is completely different (see next slide)
- Different configuration were tested: it looks like this is the best result we can achieve
- In both cases the disk subsystem on the server is the bottleneck



Performance Tests: lustre vs xrootd



SuperB testing description

- Marico:
  - multi analysis script (B^+/0 --> K(*)$^+$/0 nu nu, B^+ --> e^+nu)
- Submitted locally to PBS queue
- The read patterns looks like very sequential: see next slide
  - this helps to increase the bandwidth usage
- Each job is able to use ~20-30MB/s flat during the job

SuperB read pattern

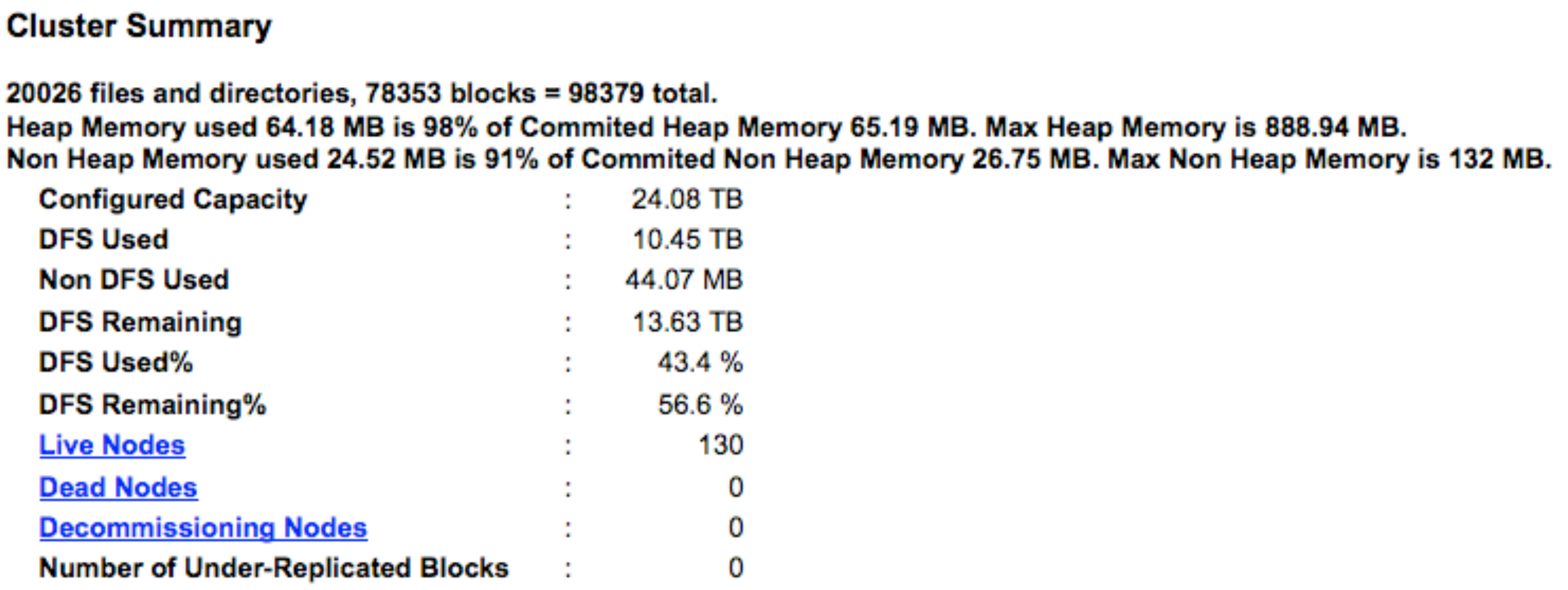
```
110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@116483825

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@116680433

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@116877041

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@117073649

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@117270257

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@117466865

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@117663473

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@117860081

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 54699@118056689

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@118111388

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdProtocol: 0400 fh=0 readV 196608@118307996

110321 18:09:45 20911 donvito.23190:18@pccms64 XrootdResponse: 0400 sending 2020955 data bytes;
status=4000
```

CMS read pattern

```
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdResponse: 2d00 sendfile 179665 data bytes; status=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2c00 req=3013 dlen=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2c00 0 fh=0 read 181178@122865227
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdResponse: 2c00 sendfile 181178 data bytes; status=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2b00 req=3013 dlen=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2b00 0 fh=0 read 176417@124779212
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdResponse: 2b00 sendfile 176417 data bytes; status=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2a00 req=3013 dlen=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2a00 0 fh=0 read 143084@126377765
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdResponse: 2a00 sendfile 143084 data bytes; status=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2900 req=3013 dlen=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2900 0 fh=0 read 146376@127338534
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdResponse: 2900 sendfile 146376 data bytes; status=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2800 req=3013 dlen=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2800 0 fh=0 read 162231@128438257
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdResponse: 2800 sendfile 162231 data bytes; status=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2700 req=3013 dlen=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2700 0 fh=0 read 172427@129646456
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdResponse: 2700 sendfile 172427 data bytes; status=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2600 req=3013 dlen=0
110330 18:37:18 20911 donvito.9576:18@pccms60 XrootdProtocol: 2600 0 fh=0 read 172671@131209955
```

Testing infrastructure

- Mixed infrastructure: classic server + WNs HD
  - 1 server: 14TB of disk space
    - Network: 10 Gbit/s
  - 130 WN with 120GB each
    - Network: 1 Gbit/s



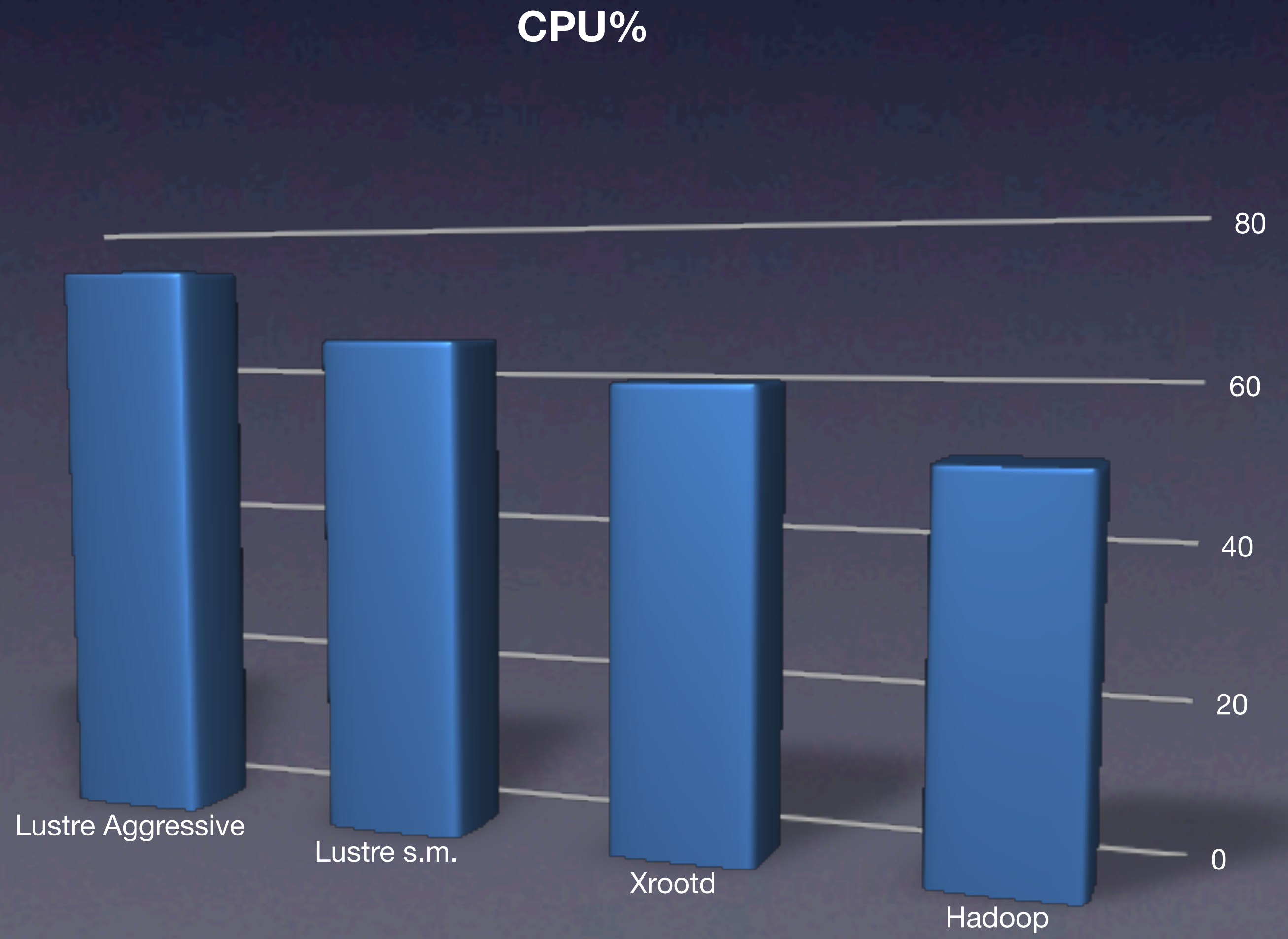
Cluster Summary

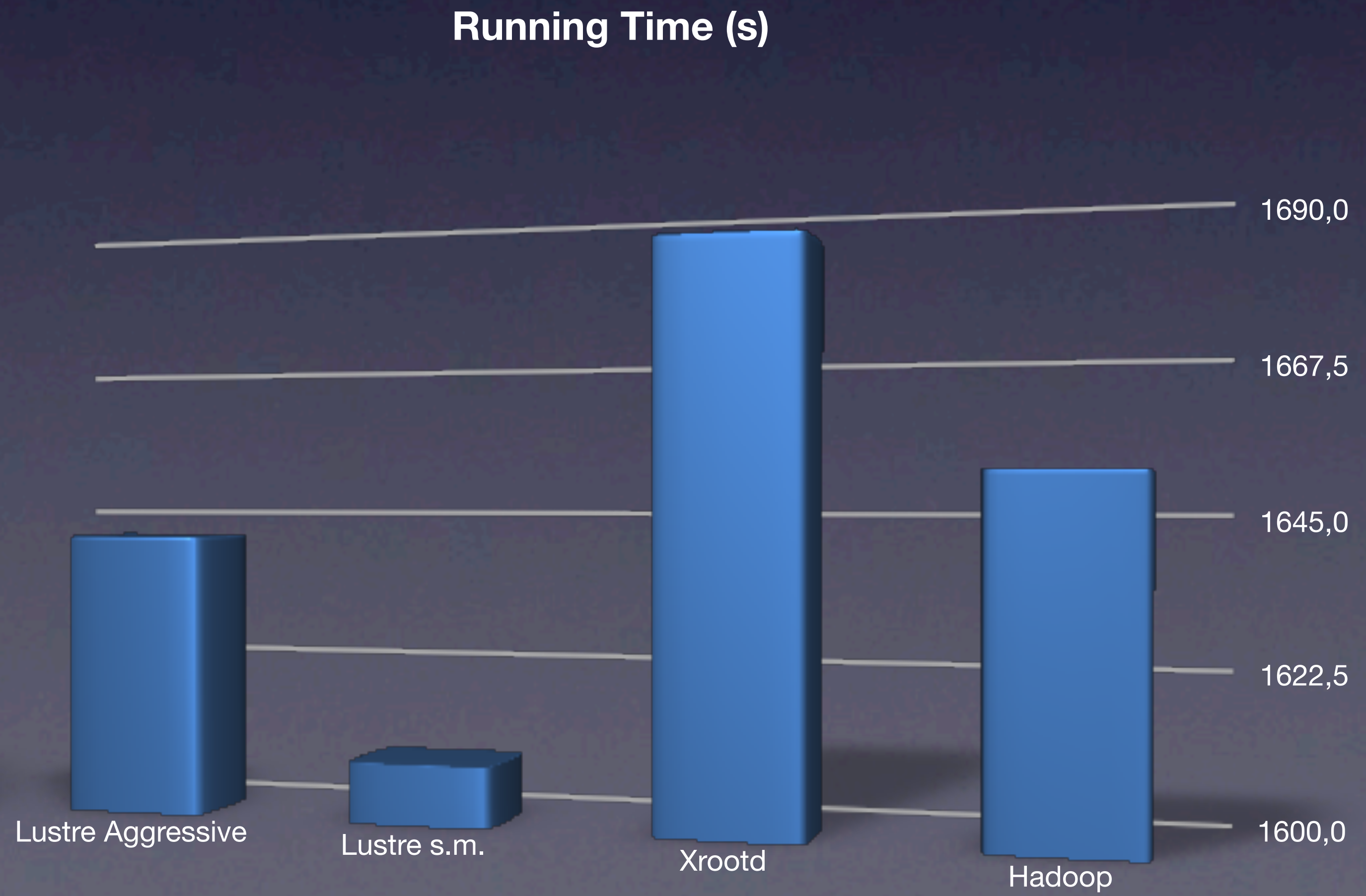
20026 files and directories, 78353 blocks = 98379 total.
Heap Memory used 64.18 MB is 98% of Commited Heap Memory 65.19 MB. Max Heap Memory is 888.94 MB.
Non Heap Memory used 24.52 MB is 91% of Commited Non Heap Memory 26.75 MB. Max Non Heap Memory is 132 MB.

Configured Capacity	:	24.08 TB
DFS Used	:	10.45 TB
Non DFS Used	:	44.07 MB
DFS Remaining	:	13.63 TB
DFS Used%	:	43.4 %
DFS Remaining%	:	56.6 %
Live Nodes	:	130
Dead Nodes	:	0
Decommissioning Nodes	:	0
Number of Under-Replicated Blocks	:	0

Running a single Marico job

- We tested different storage architectures (Lustre, Xrootd, Hadoop):
  - using always the same job and input files
- We mesure the running time and the CPU efficiency:





Optimizations

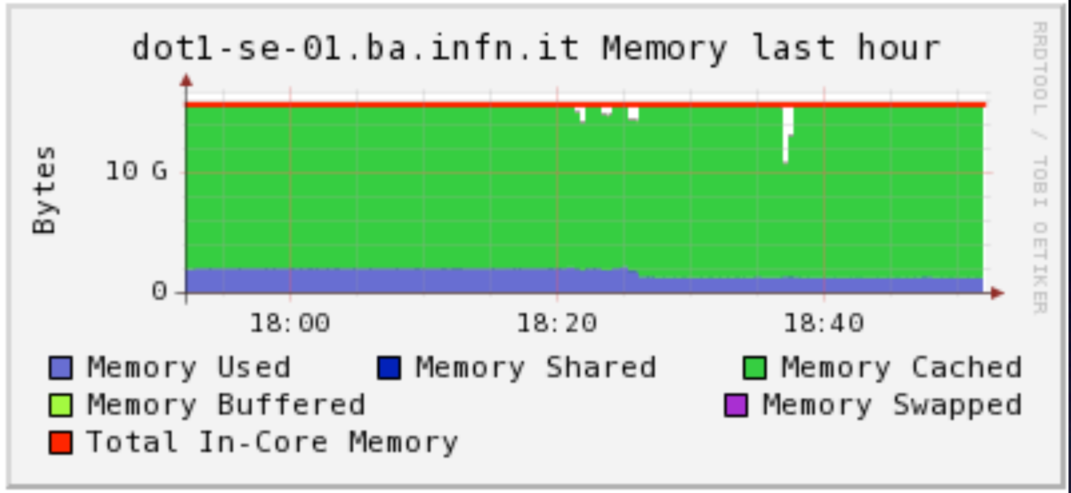
- Lustre 2.0.1:
    - Increasing the ReadHead will increase the CPU% paying with a higher network bandwidth
- Hadoop 0.21.0:
    - Tuning the ReadHeadBuffer as an option of FUSE mount point (=> rdbuffer=1M)
    - Tuning the number of thread serving data on the server

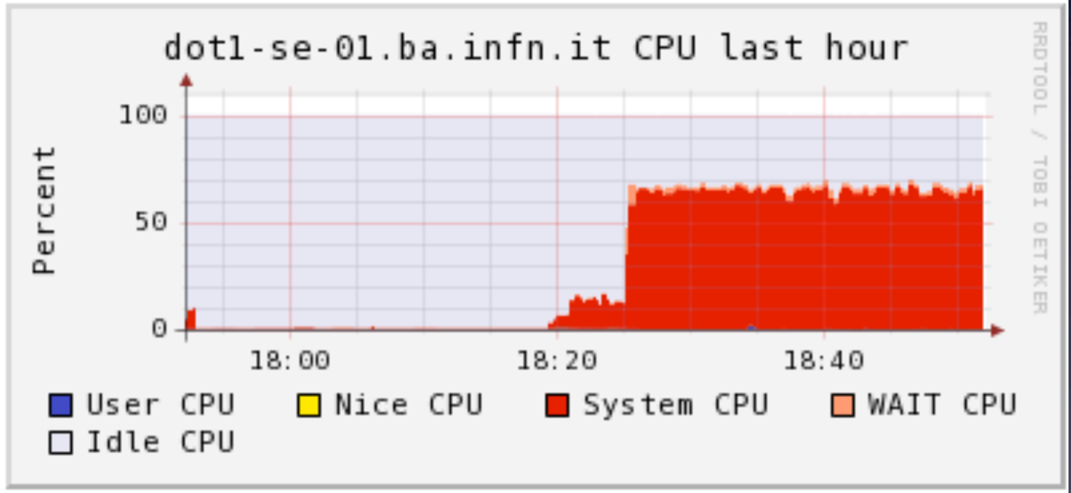
Scalability test

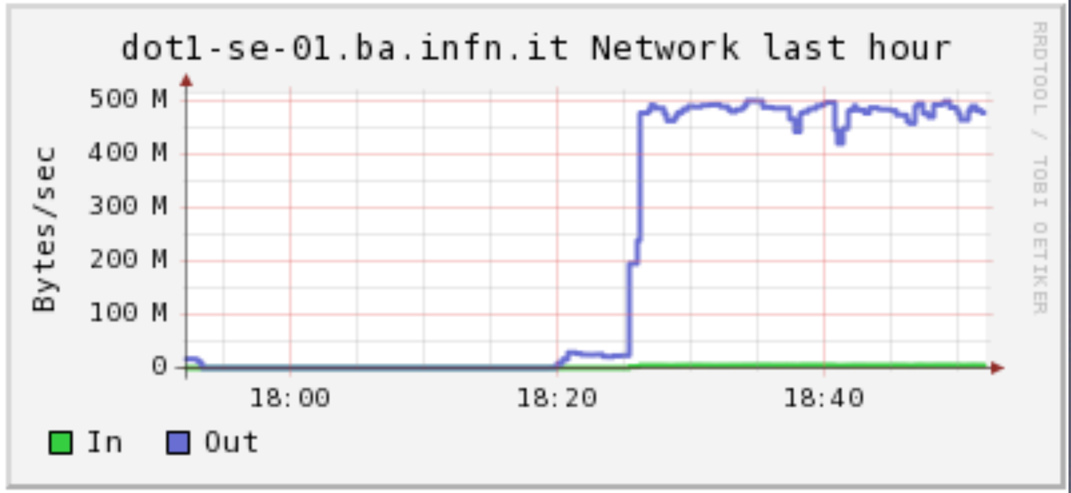
- Vertical scaling:
  - how many process per server?
- Horizontal scaling
  - how to scale on a big farm environment

Marico on Lustre

- 25 concurrent Marico jobs
- No network bottleneck on the client
- Reached the hw limit on the server
- ~50% of CPU utilization
- Lustre infrastructure scales easily adding more server
    - but it is costly (network, high performance disks, etc)







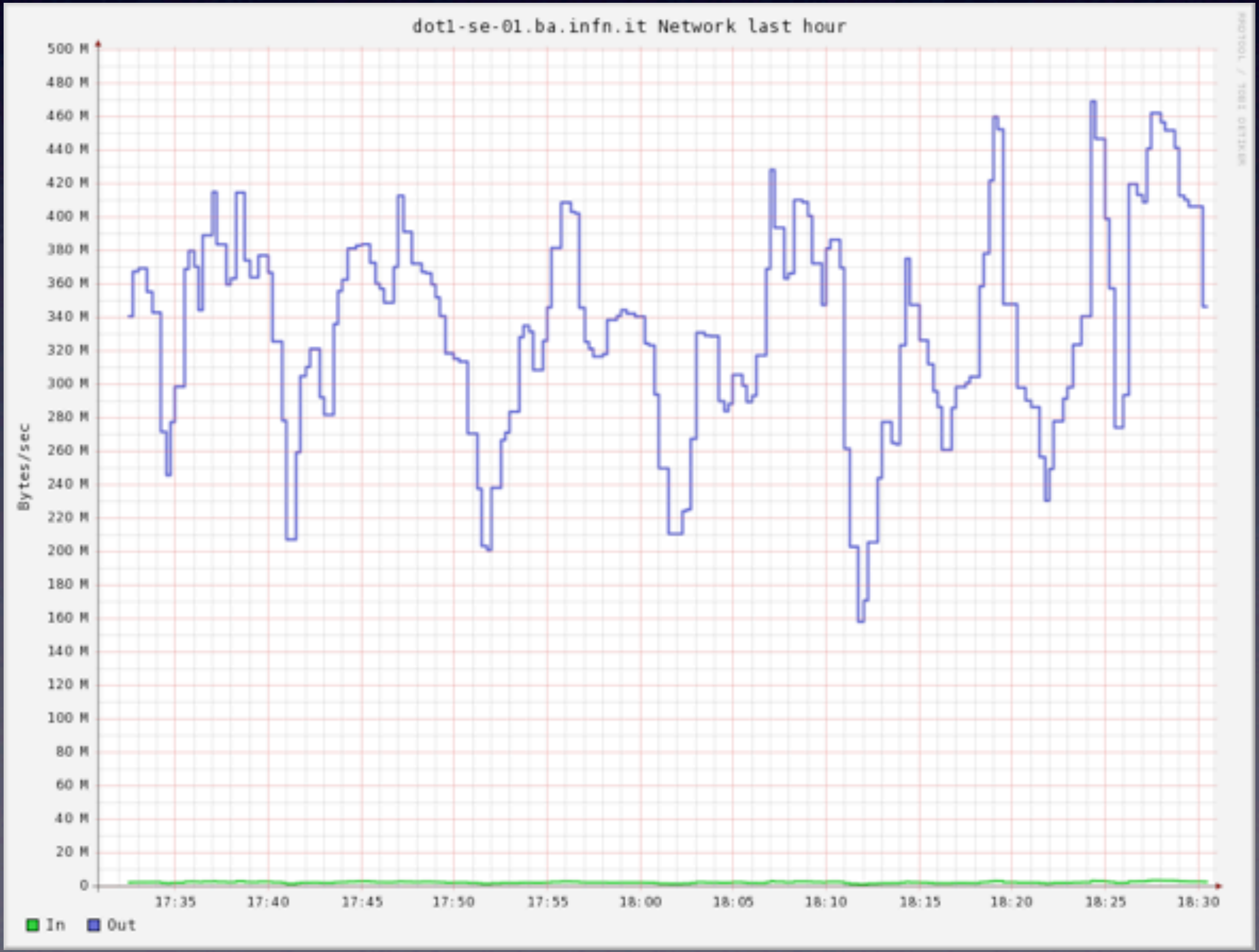
Consideration on Lustre Test

- 10Gbit/s network is a must
  - The network infrastructure should be build taking into account these requirements
- Fast raid subsystem is needed

Hadoop scalability test

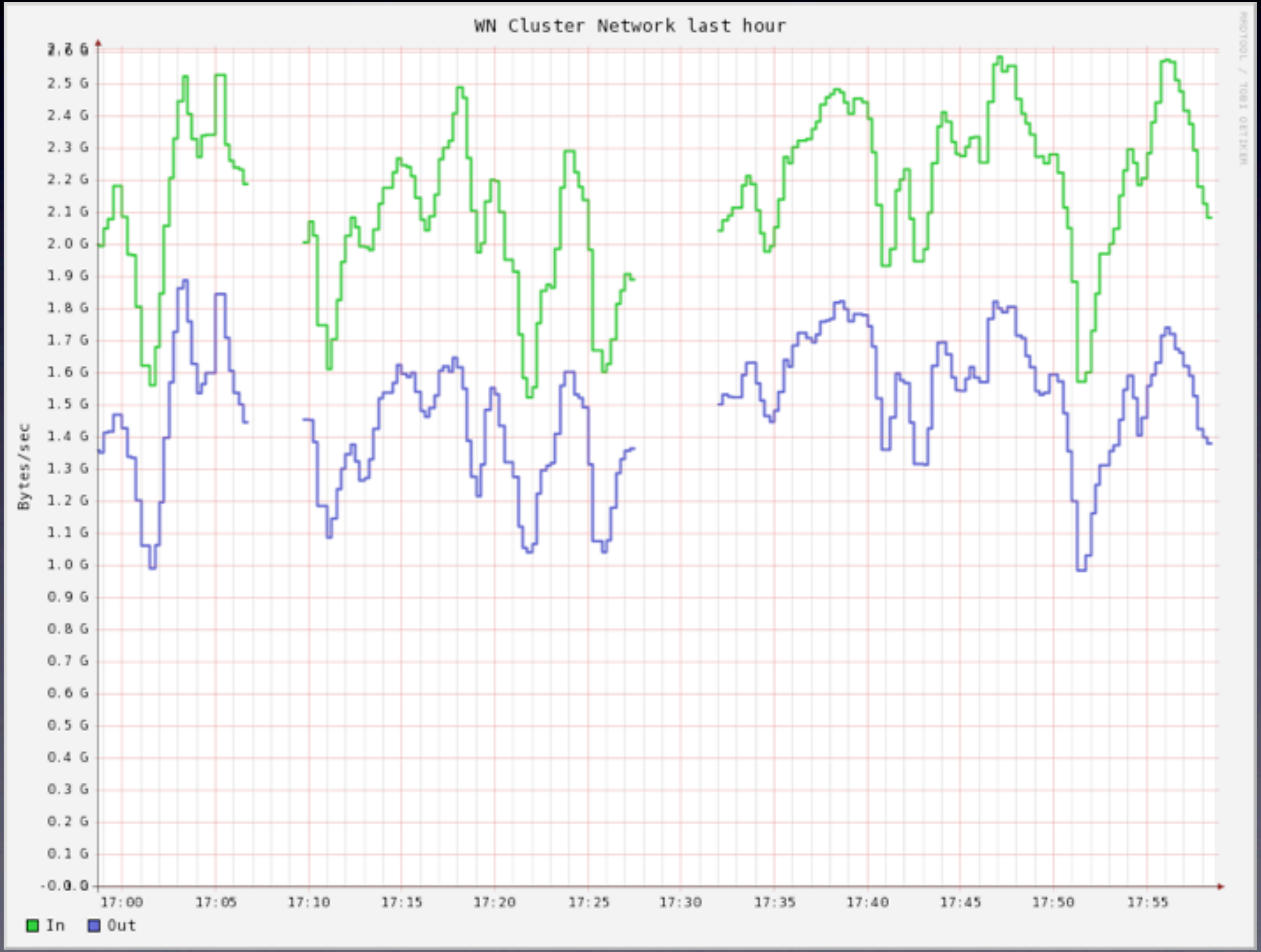
- It is evident that the single server is less performant than using Lustre on the same server:

- 25 concurrent Marico jobs
- ~30% of CPU efficiency



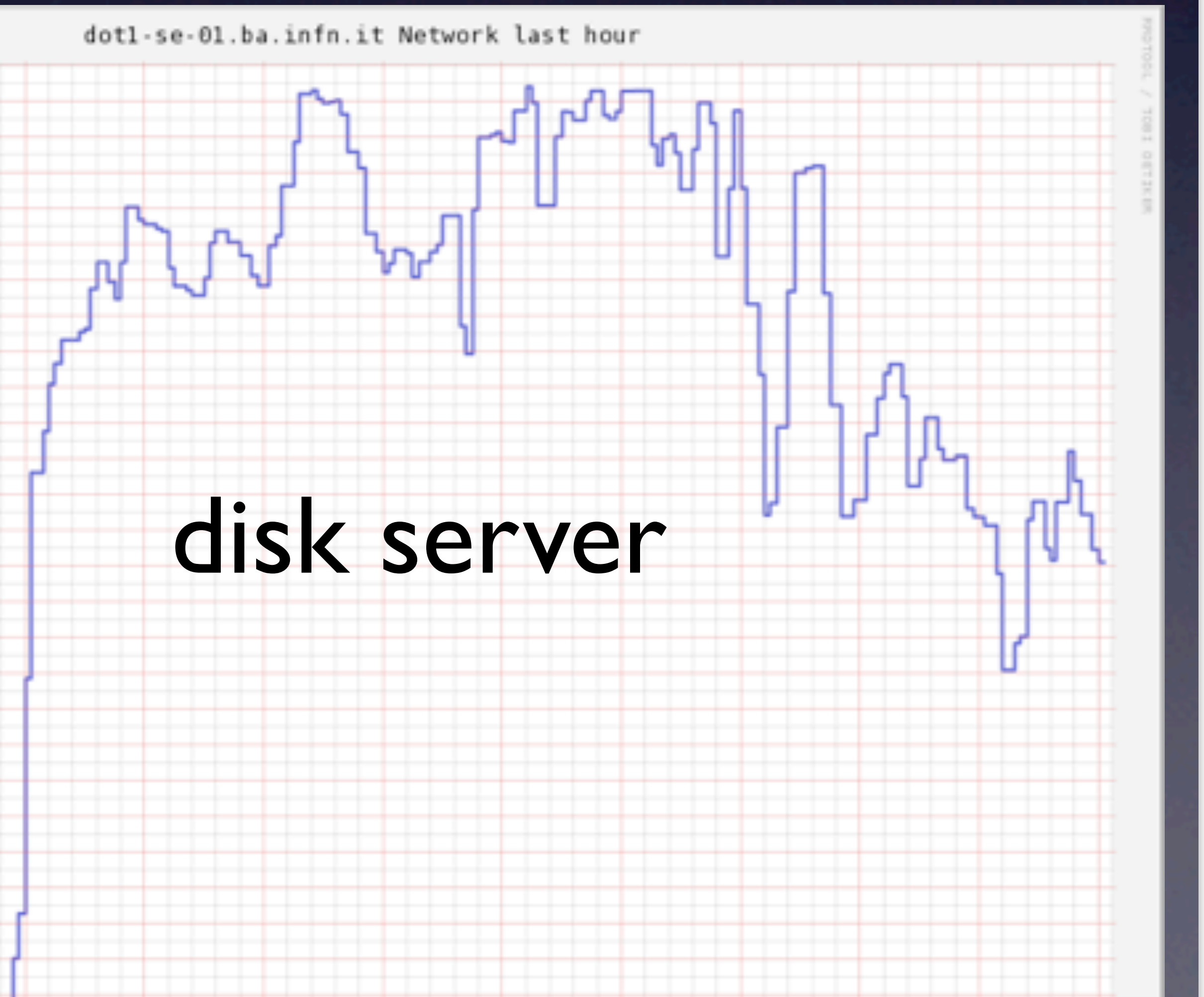
Hadoop orizontal scalability

- Using hadoop on 130WN+1 disk server
- Sequential (concurrent) read and write
- easy to go up to: 2.6GByte/s

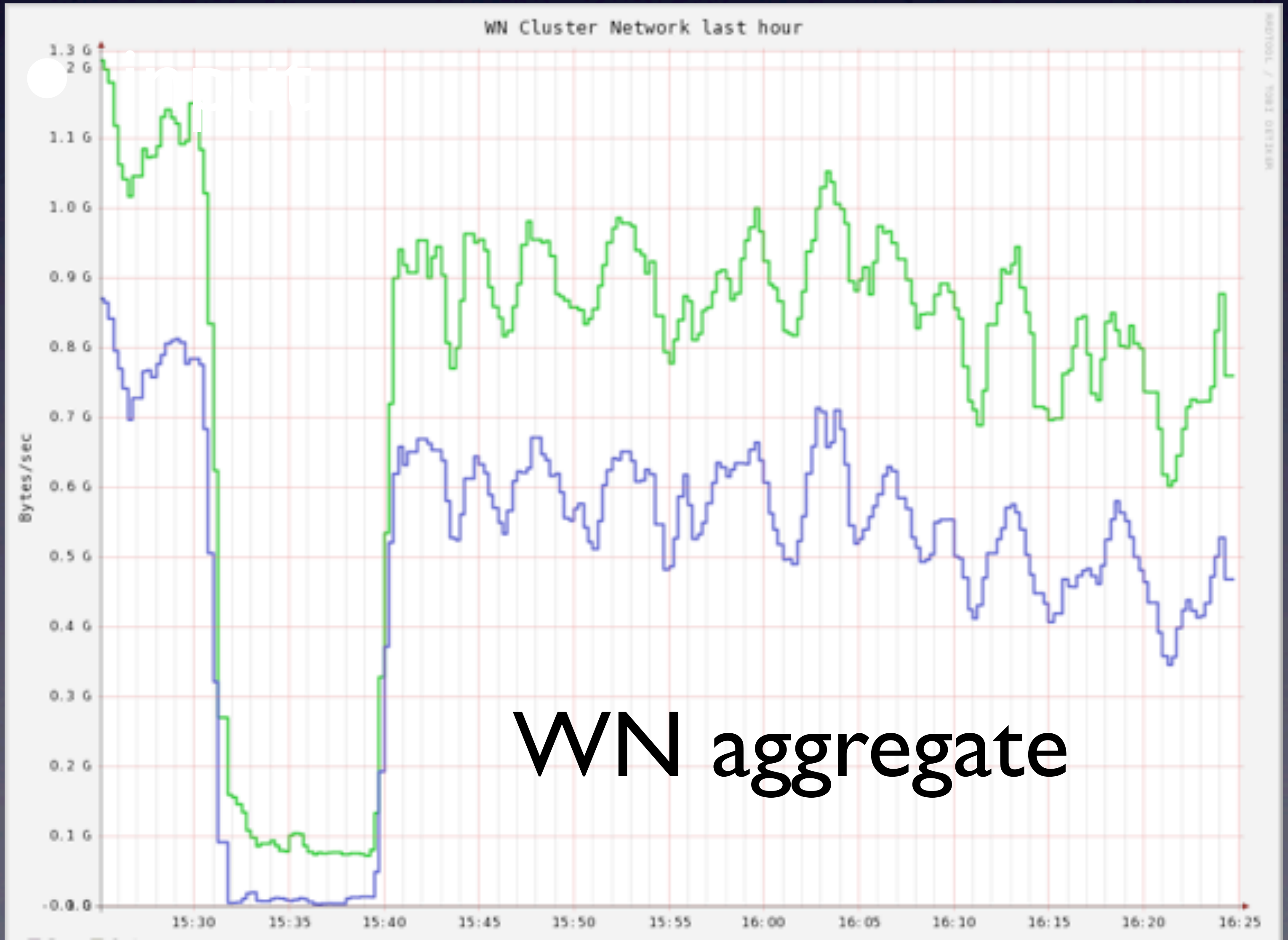


Hadoop Marico test

- 180 concurrent jobs
- 2.5TB of input data completely analyzed in about in less than 50min
- average ~20% of CPU efficiency



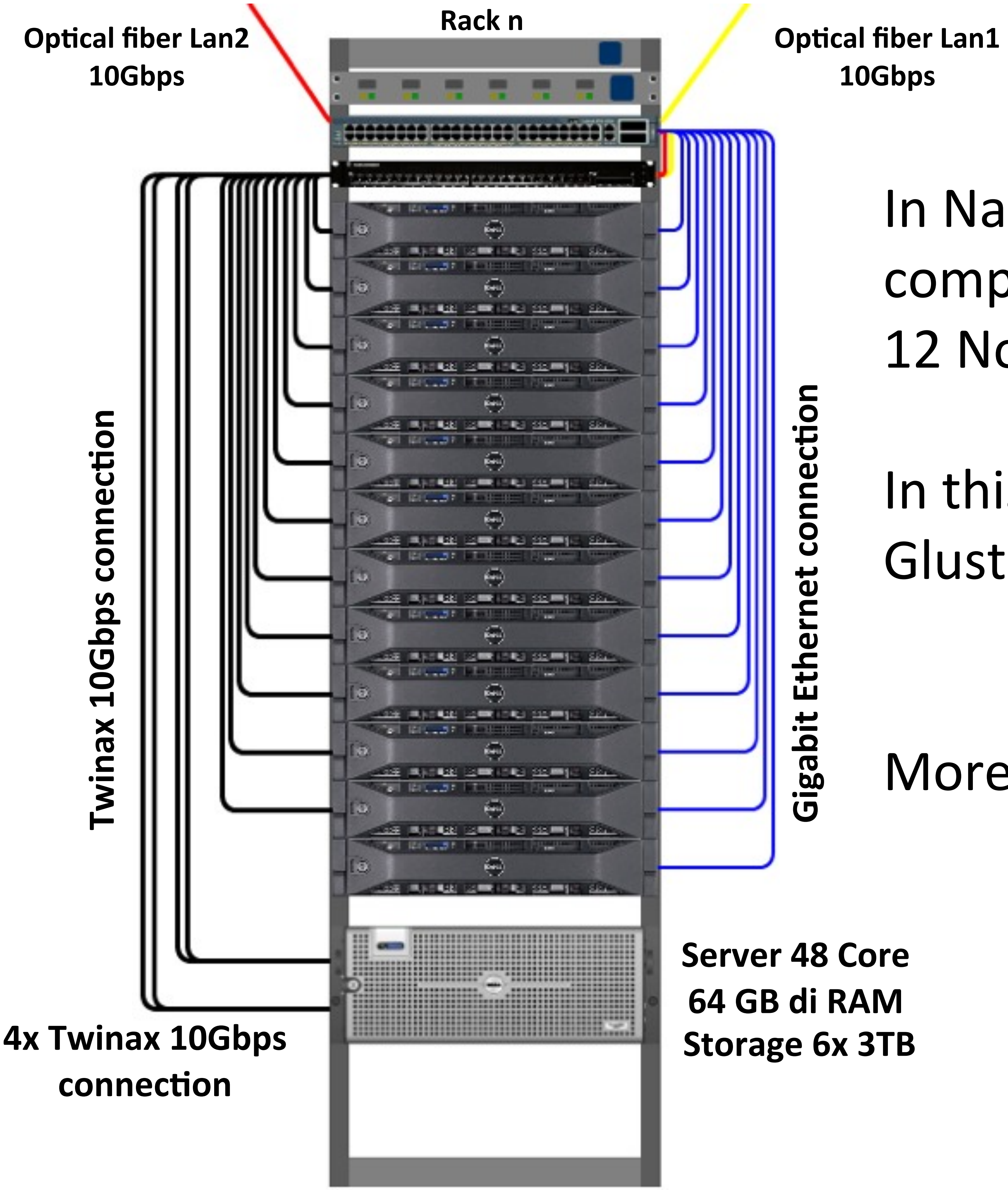
disk server



WN aggregate

Hadoop Marico test

- The performance could be easily ~doubled: adding a second disk in each WN
- Few tuning and configuration studies still needed to try to improve the performance
- The global cost of the hw infrastructure is surely cheeper than the usual client-server infrastructure
  - and both network infrastructure and disk subsystem is easier to be realized and manteined



Rack n

Optical fiber Lan2
10Gbps

Optical fiber Lan1
10Gbps

Twinax 10Gbps connection

Gigabit Ethernet connection

Server 48 Core
64 GB di RAM
Storage 6x 3TB

4x Twinax 10Gbps
connection

In Naples we have a testbed composed by
12 Nodes in a 10Gbit/s Network

In this cluster we are testing GlusterFS vs Hadoop

More references in the next talk...

Work in progress and Future works

- Testing different SuperB analysis case
  - What will happen if input files will be zipped in the future?
- Testing different storage solution: CEPH and GlusterFS
- Xrootd test
  - Involving also remote data access
- INFN-Pisa is starting testing NFSv4.1 for SuperB use cases
  - Using SL6.x (this will became the standad operating system for scientific computing facilities in the next years)
- Studying and testing WAN solution for data transfer and placement

- Providing feedback to the Framework developers

Acknowledgement

- SuperB analysis test case:
  - Elisa Manoni
  - Alejandro Perez
- Test storage infrastructure
  - Vincenzo Spinoso