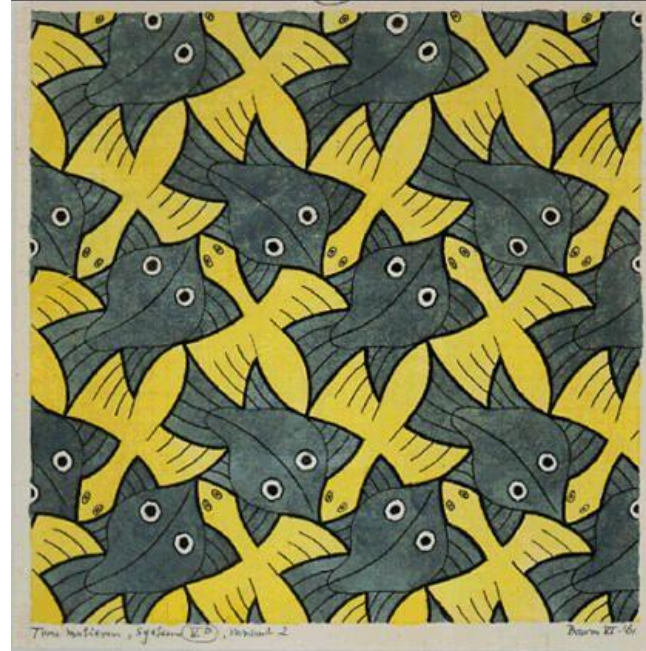


# Sharing server nodes for storage and compute **Batch on EOS Extra Resources**

CHEP 2018, SOFIA, July 10



# BEER Contributors

- ❑ IT-CM
  - Tim Bell, **Ben Jones**, Domenico Giordano,
  - **Gavin McCance**, Jaroslava Schovancova, Havard Tollefsen
- ❑ IT-ST
  - Dirk Duellmann, **Massimo Lamanna**, Herve Rousseau
- ❑ IT-DI
  - Markus Schulz, Andrea Sciaba, Andrea Valassi, **David Smith**
- ❑ Petersburg Nuclear Physics Institute
  - **Andrey Kirianov**
- ❑ **ATLAS for using the resource**

# Background

- ❑ EOS is CERN's large storage management system
  - <https://eos.web.cern.ch/>
  - Frequent reports at CHEP ☺
- ❑ The hardware architecture follows CERN's commodity paradigm
  - Disk servers are based on standard servers with shelves of disks

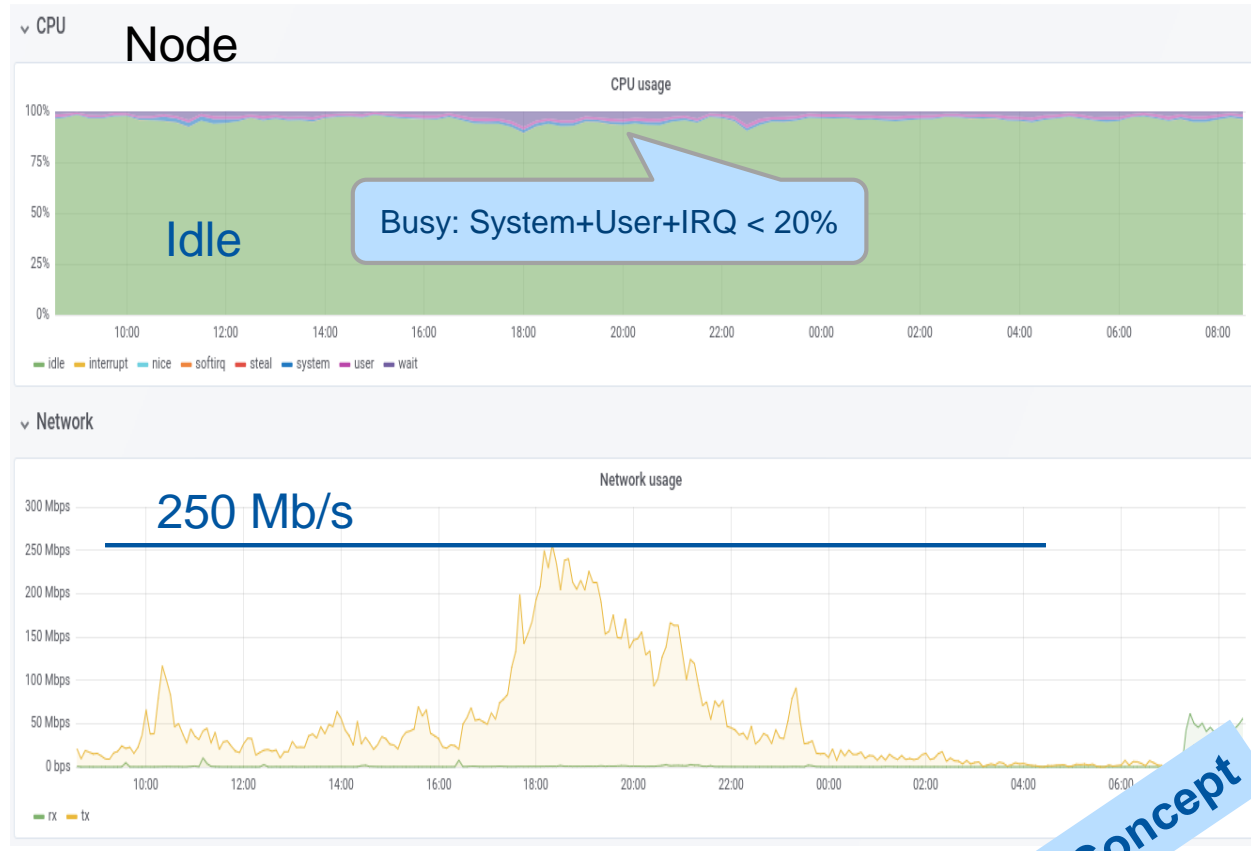
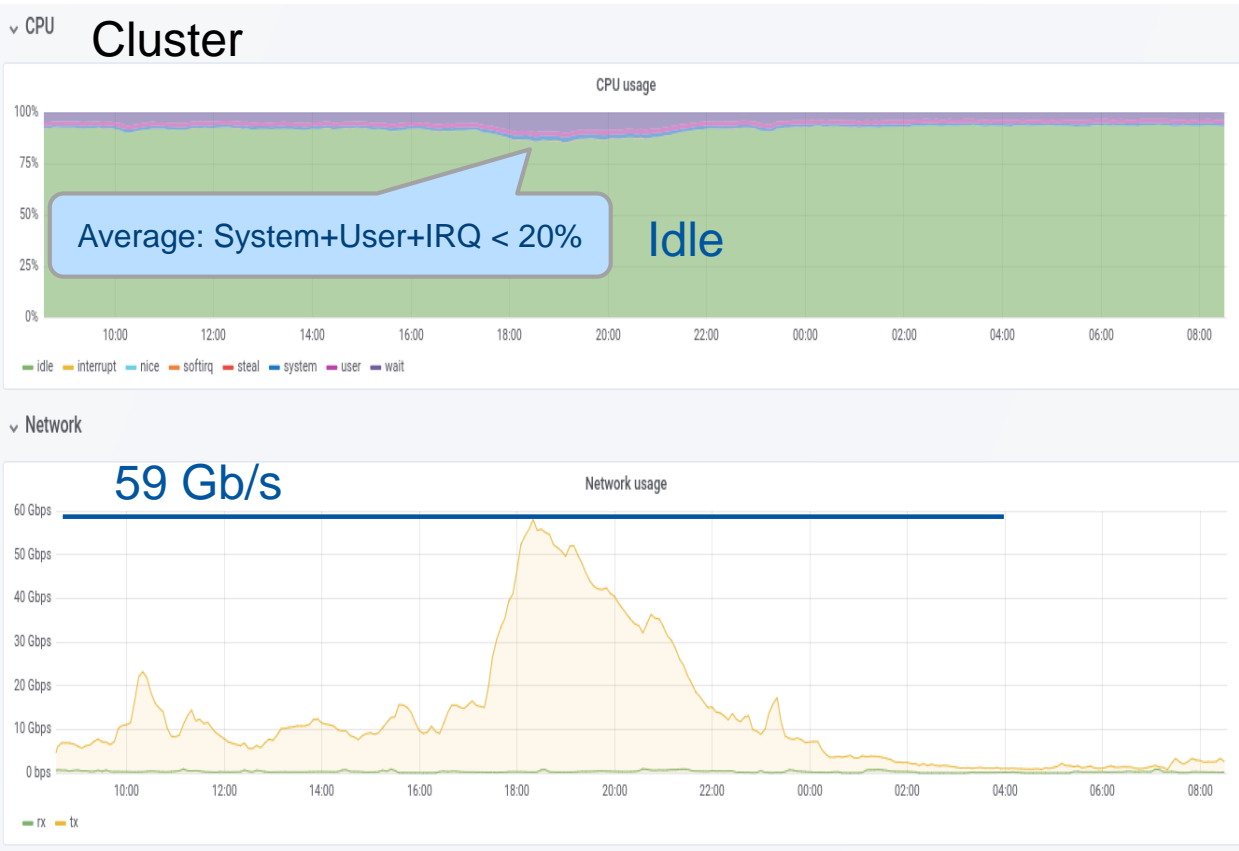


# Motivation

- ❑ Observation of relative low **average** loads on our storage systems
  - Sometimes I/O bound (internally and network)
    - Most nodes are not CPU saturated even when they are I/O saturated
- ❑ Average usage per server node (snapshot):
  - Read: **35MB/sec - 67MB/sec**, write: <10MB/sec
  - IOPS: read 290-500 Hz, write: 28Hz
- ❑ CERN has a relatively large storage system, **1339** nodes with data
  - Several have 40 cores with 3.2 GB/core
- ❑ **Questions:**
  - Can we make use of some of these cores?
  - What value does this correspond to?
  - Proof of concept tests where done by Andrey using some older nodes.

# Recent Loads (spring 2018)

- ALICE EOS **cluster average** CPU utilization is  $\leq 20\%$  – left plot
- ALICE EOS **disk server** CPU utilization is  $\leq 20\%$  – right plot
  - Some **IOWait**, which can be used by other processes → Significant Potential



# First Steps

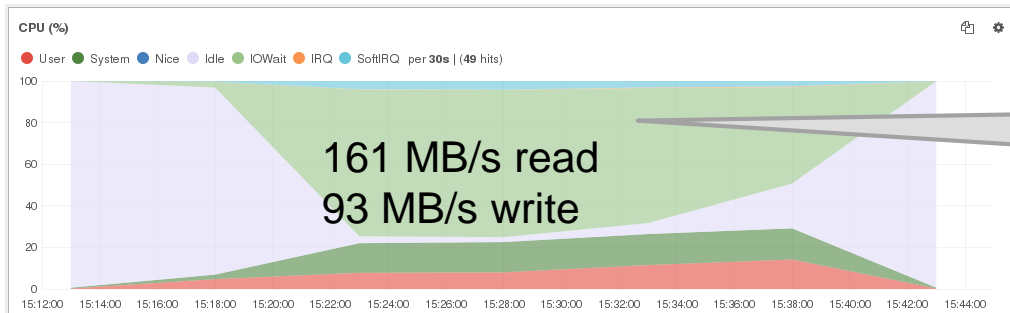
- ❑ Proof of concept tests by Andrey Kiryanov
  - Testbed (puppetized)
    - Small EOS system(s)
    - Client/load-generator cluster
    - Condor based test with computational loads
      - Using `lhc@home`
      - Boinc based system
      - Using *nice* to limit impact on EOS



Proof of Concept

# Test systems

- First system
  - Disk servers with 1Gb interface
  - Local IO generator also used
- Second system
  - Disk server with 10Gb interface

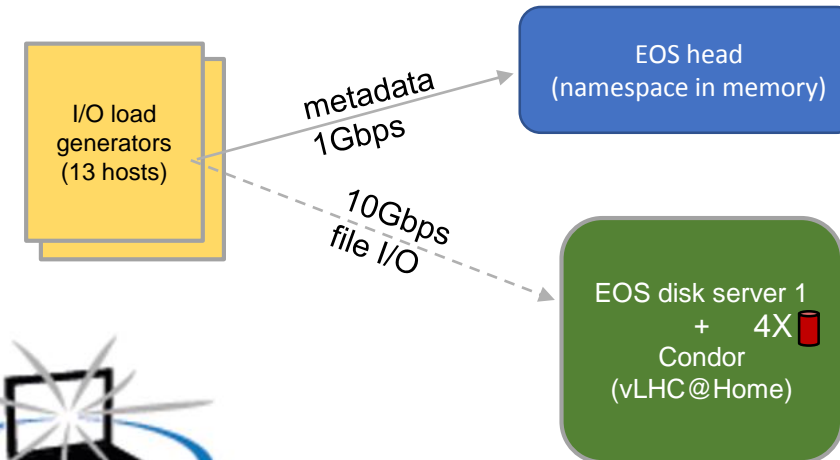
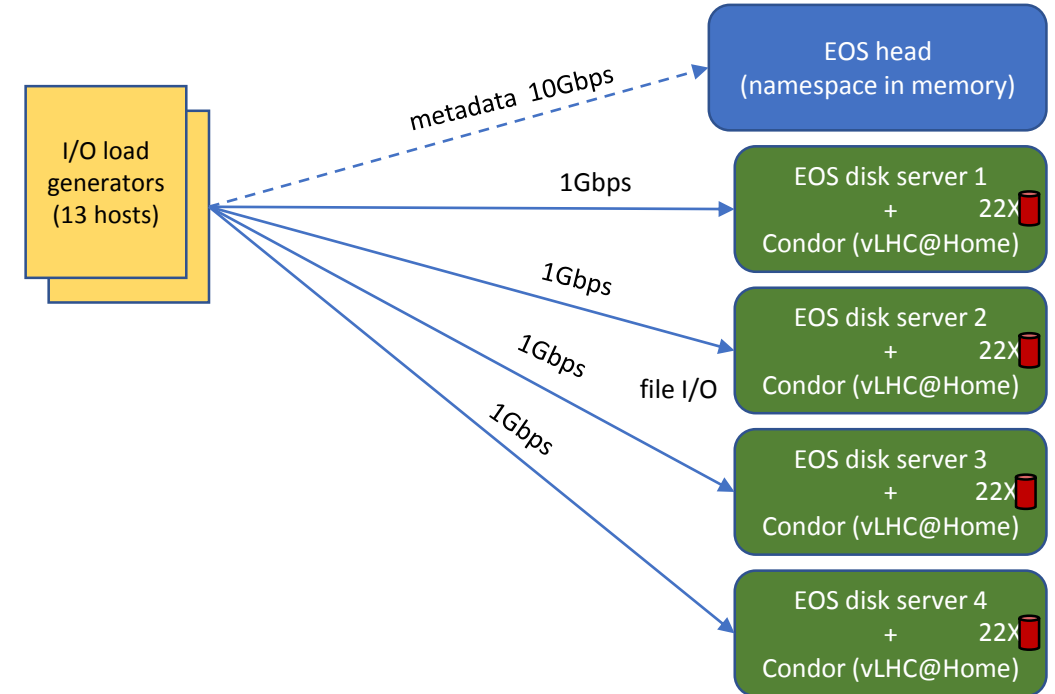


EOS only



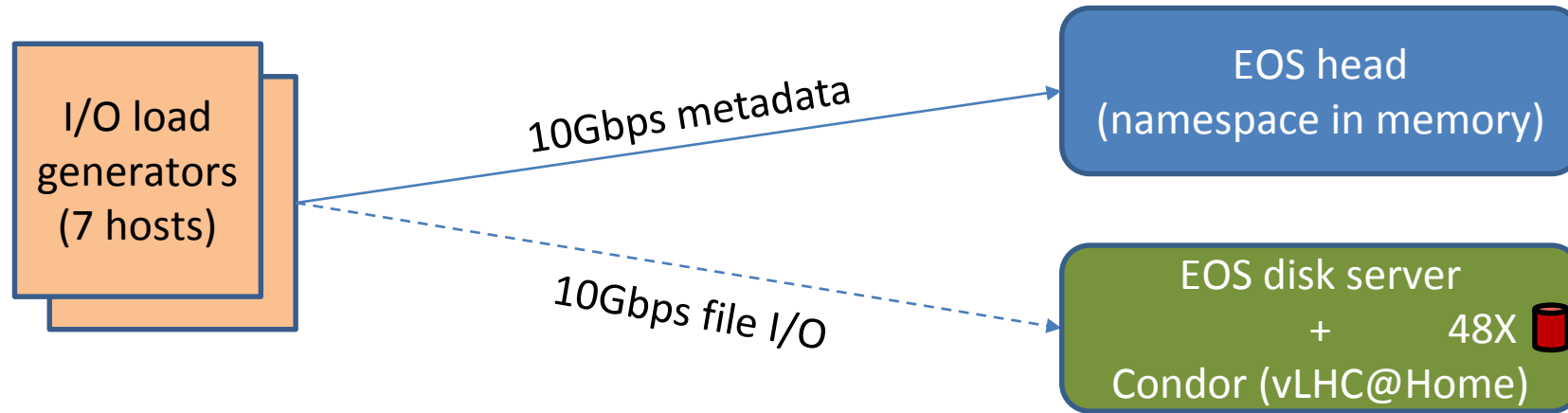
EOS +  
vLHC@home

7



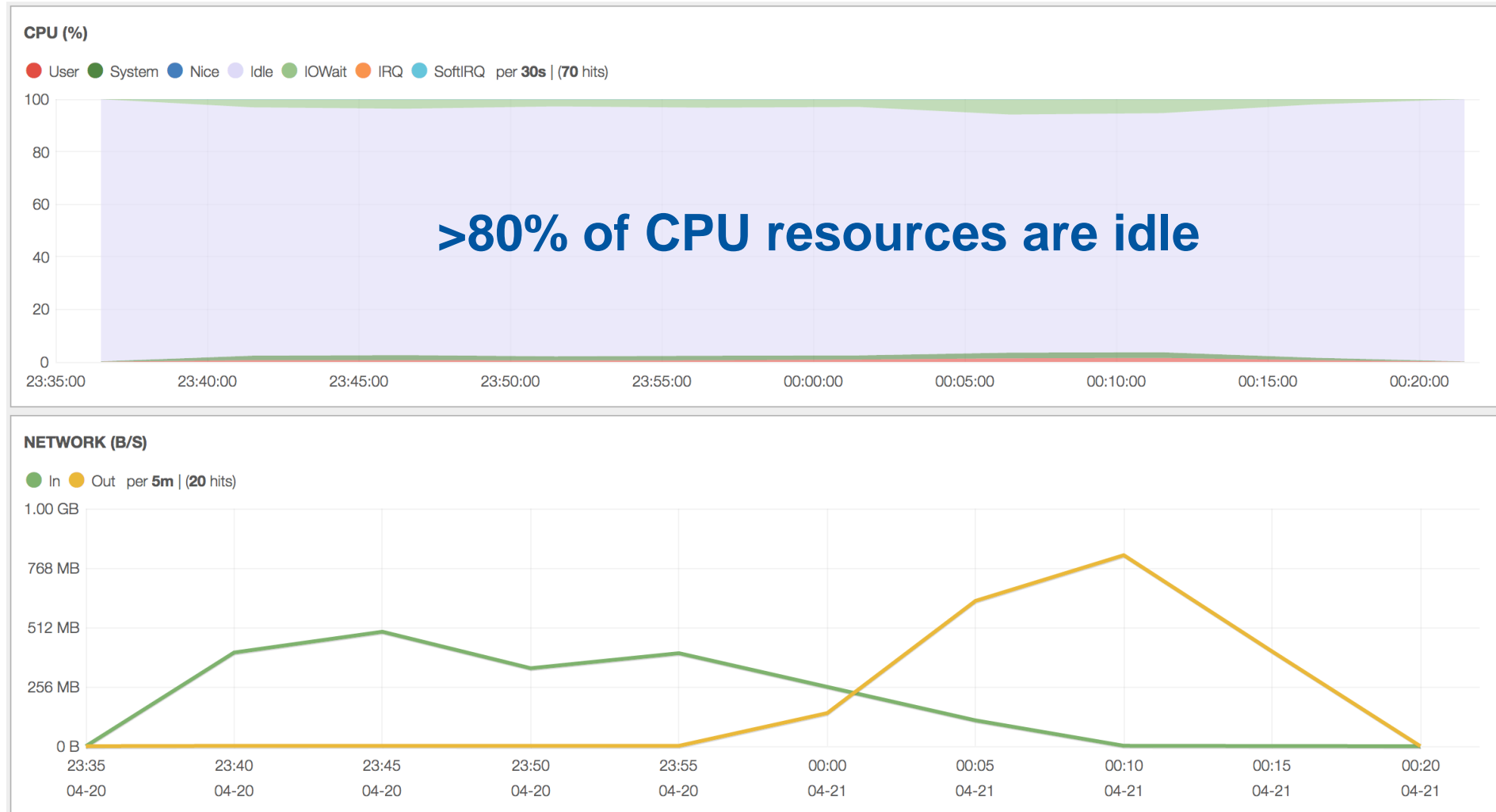
7

# Tests with more recent hardware



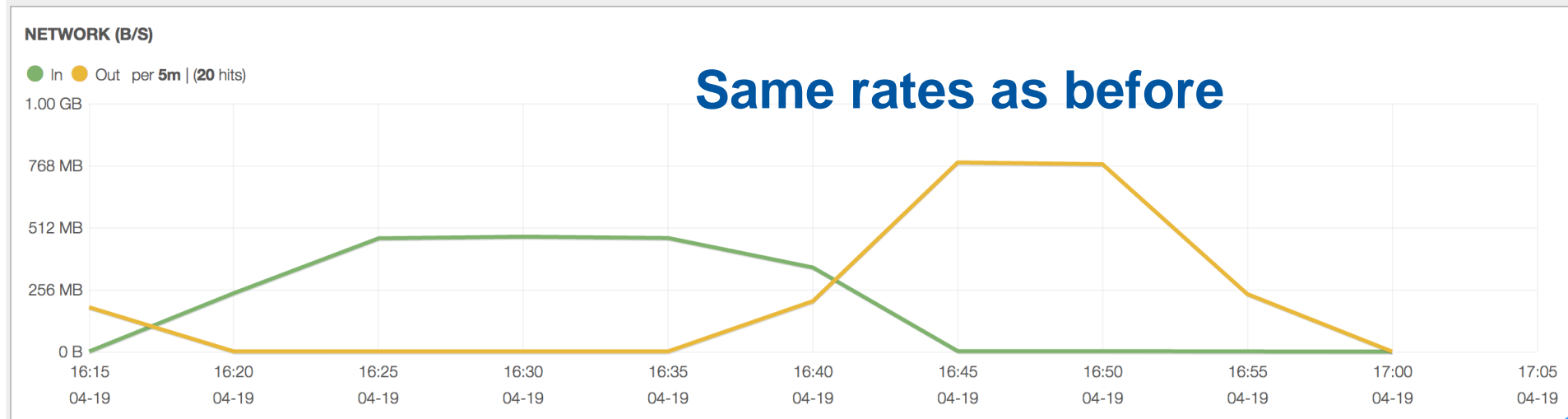
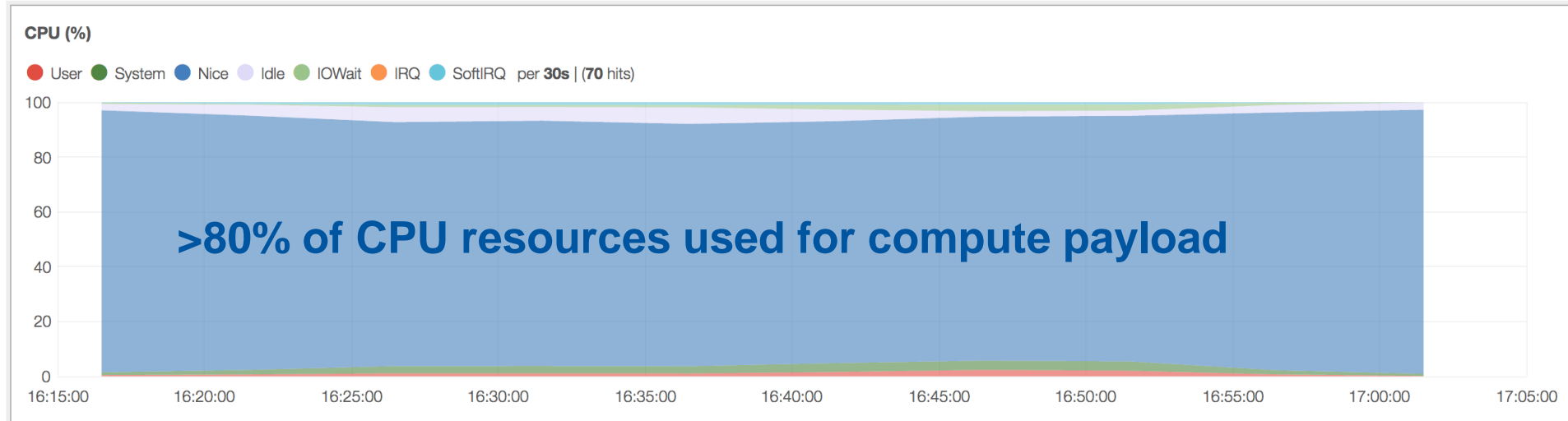
- New disk server with decent hardware
  - Dual E5-2630 v3 @ 2.40GHz (32 vcores with HT)
  - 64 GB RAM
  - 48 disks 6TB each + 2 SSD (OS + swap)
  - 10 Gbps network

# Running with no payload



Proof of Concept II

# Running with compute payload



Almost identical results

Proof of Concept II

# What did we learn?

- ❑ With a simple setup **Storage and Computing** can be run without much interference
  - HTCondor + *nice*
- ❑ For typical I/O loads in production we can expect to use **>80%** of the CPU for non storage tasks
  - Worst case would be about 50%
- ❑ Interesting..... But....

# How to turn this into production?

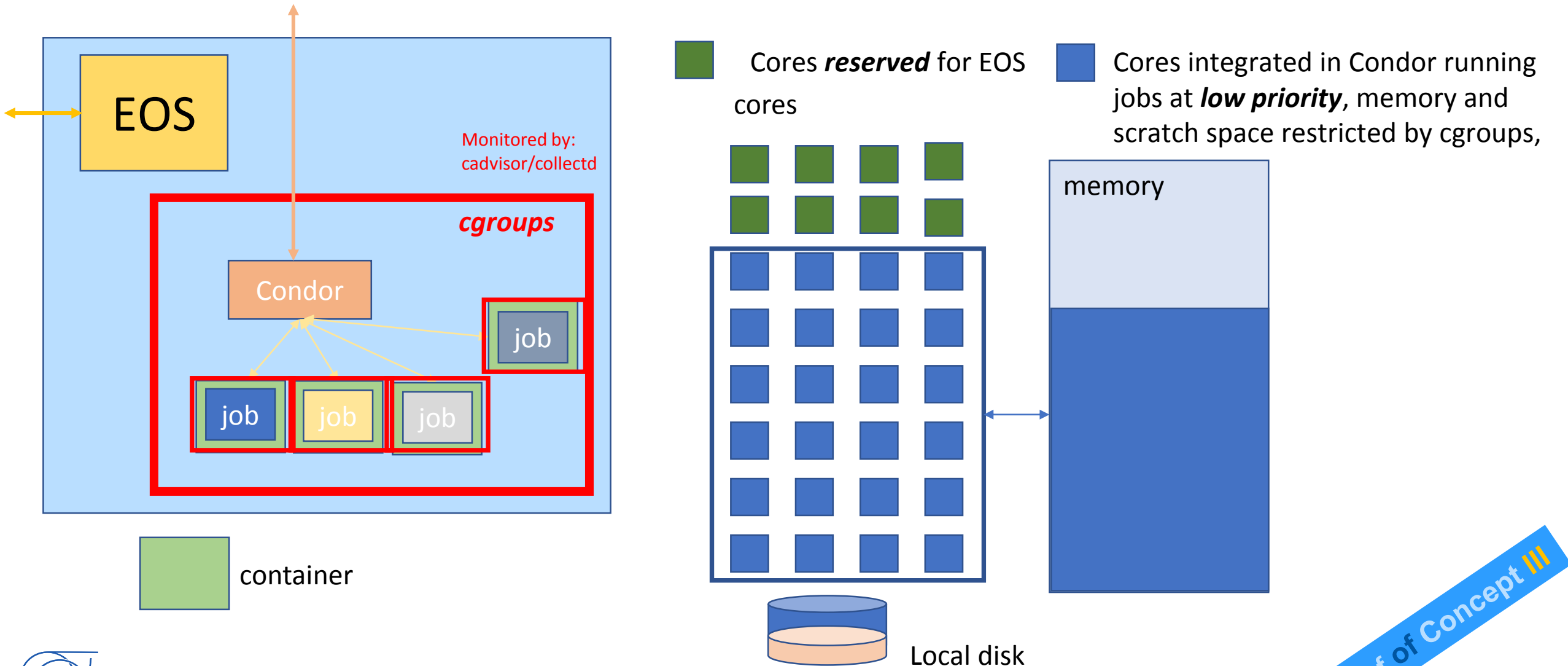
- ❑ Needs to be deployable
  - Configuration Management challenge
    - Two services on the same node
      - Responsibility split between two teams
- ❑ EOS service must not be compromised
  - Resources (CPUs, memory .....
  - Halting the computational tasks on demand
- ❑ Monitoring

# A model emerged

- ❑ “Partition” the resources
  - To guarantee that storage performance is not crippled
  - To provide accountable resources (not like lhc@home)
  - Control groups (**Cgroups**)
  - Run Condor jobs in Containers
    - Using Cgroups to limit resource usage
  - One puppet configuration for the node
- ❑ Integrate resources in CERN’s Condor batch system
  - Queues for suitable workloads
- ❑ BEER Pilot to explore this approach
  - Participation from storage and batch team
  - JIRA for ticketing



# Condor + Containers



Proof of Concept III

# Testbed #3

- ❑ Three disk servers: each
  - 48 6TB HDD (1 HDD apparently failed on one server)
  - 2 x E5-2630 v3 (haswell; 2 x 8 physical cores => 32 w/ SMT)
  - 2 x 800GB SSD (Intel DC S3510 series; 0.3 DWPD for 5 yrs)
  - 10Gbit network
- ❑ Centos7; EOS 0.3.240 (Aquamarine)
  - Using puppet, hostgroup based on eos hostgroup and using modified eos module and cerncondor module (beer branch)
- ❑ Local Disc Setup:
  - 1 ssd set aside for the batch work (including addition of 96GB swap)
  - CVMFS installed
  - cerncondor module used to install and run condor
- ❑ Changes in the batch environment (possibly amongst others):
  - set `memory.memsw.limit_in_bytes` and add condor to the cgroup controller: `cpuset.cpus` and `cpuset.memsw`
  - (systemd already set mem limit, but no support for memsw or cpuset)

# Limits

- ❑ Memory limit
  - is set as a parameter in /etc/systemd/system/condor.service and systemd uses the setting when setting up the **cgroup** for the condor service: 48GB
  - But does not limit swap usage
  - Modified condor.service to also set **memsw** limit (ram + swap) to 96GB
- ❑ Add condor to another **cgroup** using **cpuset**
  - cpus 2-7,10-15,18-23,26-31
    - leaves 4 physical cores entirely excluded covering both sockets
  - Later only number of cpus has been limited.
- ❑ Condor configured to offer 24 job slots and 96GB ram
- ❑ Number of processes has been limited to 8000
- ❑ **blkio** is used to control the I/O scheduling
  - Blkio.weight == 50
- ❑ Network traffic limits can be set via iptables on the docker level, but are currently not used.
- ❑ See detailed setup description at the end

# Security?

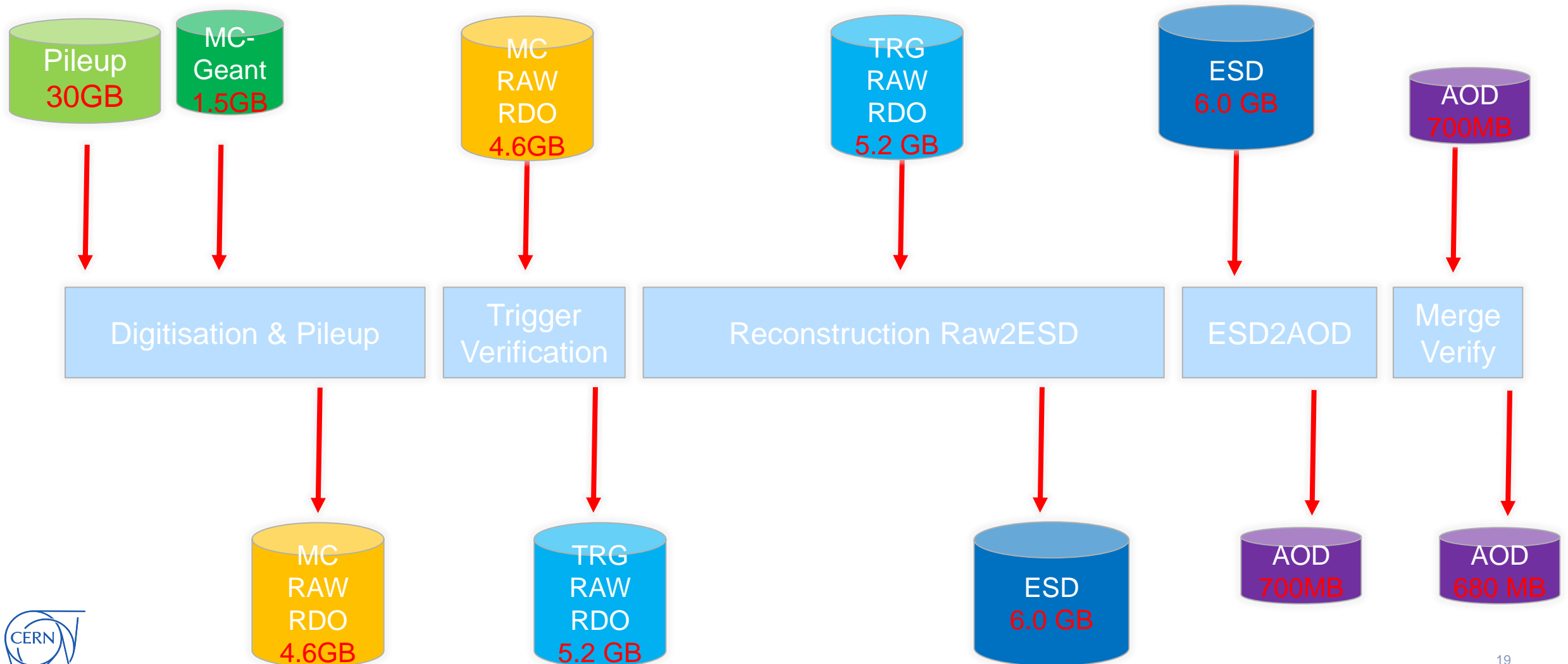
- ❑ Data on the node is protected by UNIX ownership
- ❑ Separation between users is done by CONDOR
- ❑ In addition jobs are run in containers
  - EOS data disks not visible within container

# Load generation

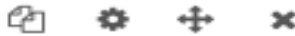
- ❑ For load run 10 instances of xrdstress
  - 4 jobs; 20 files; rw; size 1GB +- 256MB on 4 hosts
  - No difference between 3 and 4 hosts → saturation
- ❑ Run 3 ATLAS Pile job (8 cores per job)
  - Staging pileup data on node
  - Signal + min bias mixing;
  - Digitisation
  - Trigger simulation
  - Reconstruction
  - Convert ESD to AOD
  - Start jobs with short delay
  - Similar, shorter, job has been added to HammerCloud
    - For generating steady stream of jobs

# Test Job:

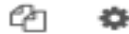
2000 MC Events, 8 cores



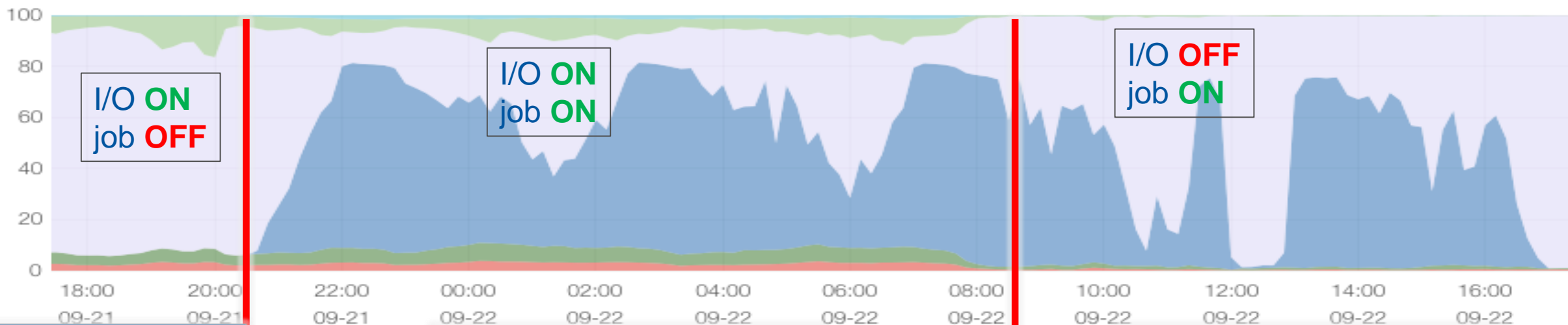
# UTILISATION



## CPU (%)



● User ● System ● Nice ● Idle ● IOWait ● IRQ ● SoftIRQ per 10m | (6531 hits)



Phase 1

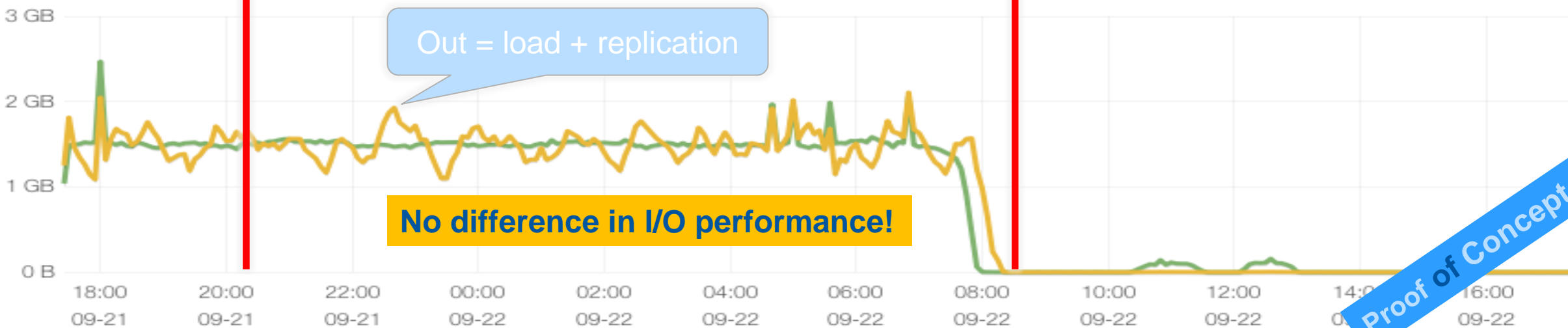
Phase 2

Phase 3

## NETWORK (B/S)



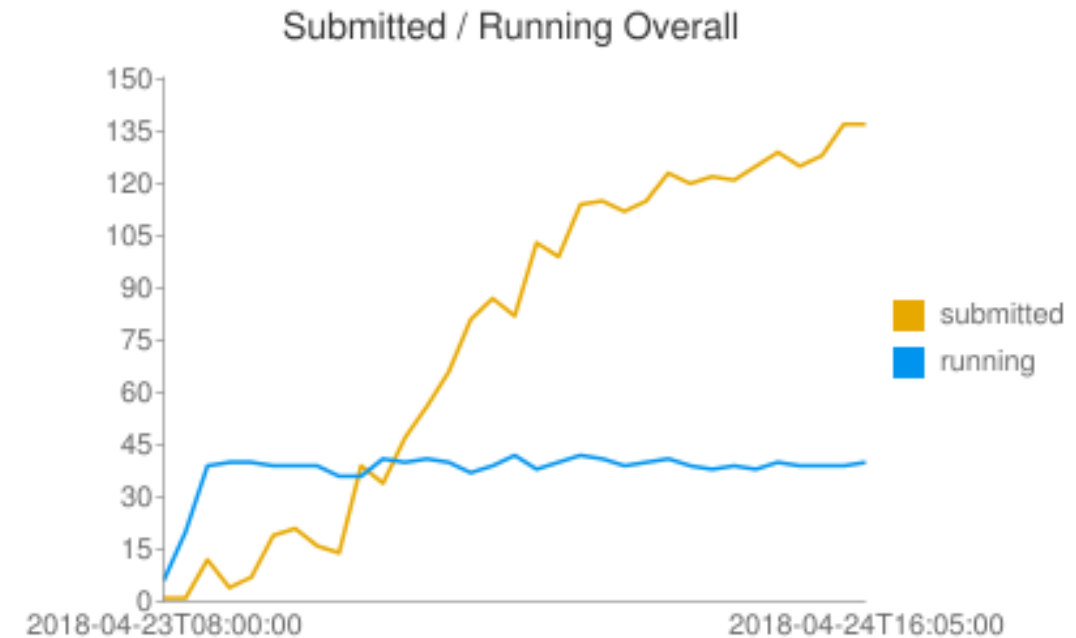
● In ● Out per 5m | (1792 hits)



Proof of Concept III

# Preproduction

- ❑ 3 test servers
- ❑ Stopped using cpusets, only cpu shares
- ❑ 4 nodes from EOS pre-production cluster
- ❑ HammerCloud (ATLAS Pile job)
  - Hammercloud submits to Condor, Condor schedules and starts the job → like a standard Grid job
  - ATHENA MP (i.e. each job starts 4 processes)
    - About 30 mins runtime (choose a small number of events / job)

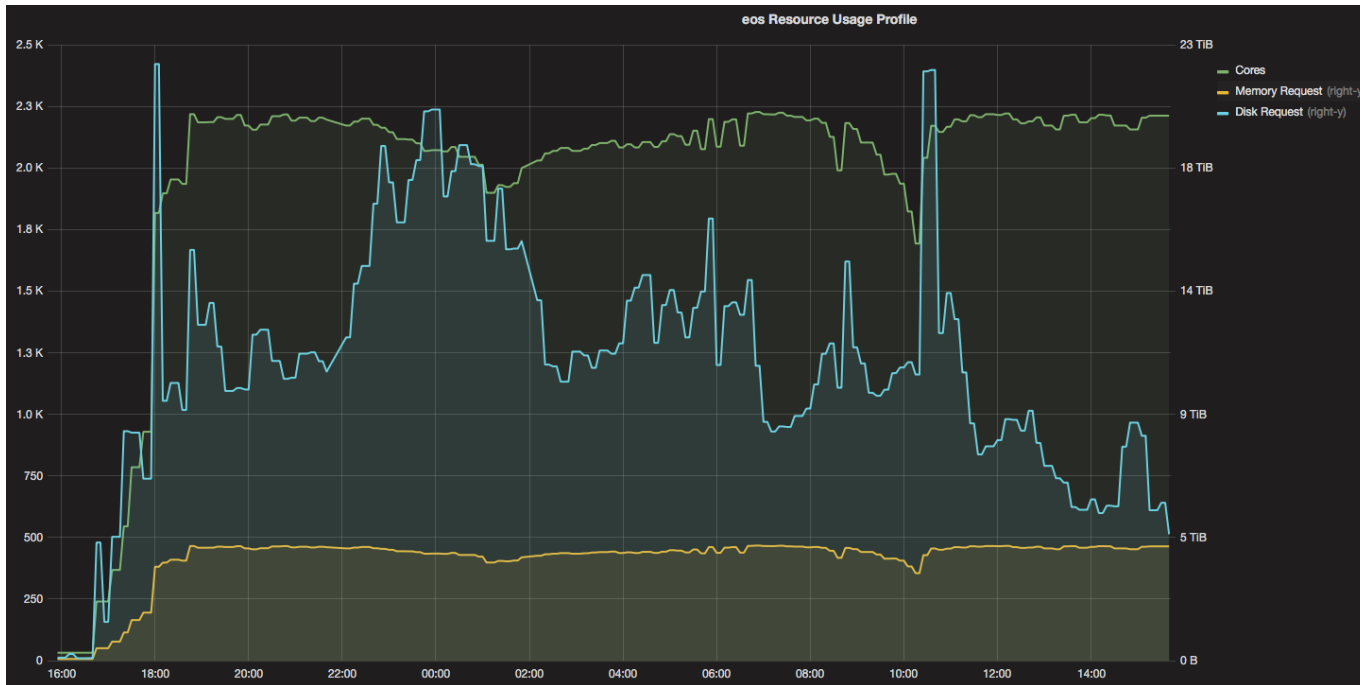


# BEER on Production Nodes

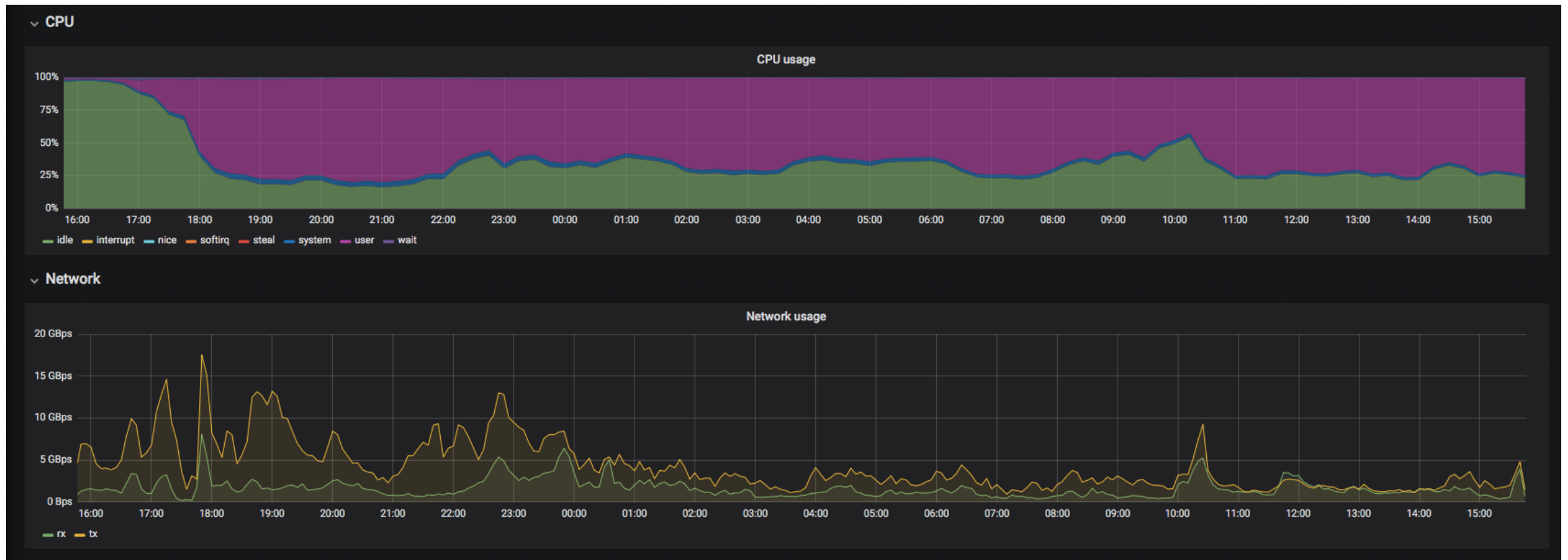
- ❑ 70 nodes with 40 cores
- ❑ Disk Servers from ATLAS Production
  - Initial loads from ATLAS
    - Later will become part of CERN resources
- ❑ Jobs experience so far
  - Production jobs started last week

# Plots from monitoring: production

- 32 1-core job slots available per 40 core machine
  - No hard limit on CPU usage; but uses cgroup cpu shares to reduce job priority to EOS service processes
- 70 disc servers involved



# Plots from monitoring: production



# Next Steps

- ❑ More detailed monitoring of the situation
  - Already increased logging on the CONDOR level
  - I/O and network related monitoring also needed
- ❑ Study of mixed workloads
  - Different experiments
  - Different processing steps
- ❑ Scaling UP!

# What could be gained for CERN?

*precision at best 10 – 20 %, but based on conservative assumptions  
roughly 10 HEP-SPECs / core*

## □ Near future:

- Assuming 40|80% of 350 machines can be used:
  - Our measurements show that this is far less than what can be done when the node is fully saturated
- $350 * 32 * 0.4|0.8 = 4480|8960$  cores ~ 44,800|89,600 HEP-SPEC06
- ~3.8|7.6% of the 2017 T0 pledge

## □ Long term (somewhat optimistic):

- Assuming 40|80% of 1200 machines:
- $1200 * 32 * 0.4|0.8 = 15360|30720$  cores ~ 153,600|307,200 HEP-SPEC06
- ~13|26% of the 2017 T0 pledge



# Estimated effort

- ❑ Activity from **P**roof **o**f **C**oncept to production readiness took about 2.5 years
- ❑ Several people have been involved. All intermittent and at small overall percentages.
  - PoC work about 3 person months over 2 years
  - Planning and coordination of production mode required intense communication
- ❑ Total effort is difficult to estimate
  - $3\text{PM} + 6\text{PM} = 9\text{PM}$  ← rough estimate assuming 5%/person
  - Some of the work was foreseen for other reasons
    - HTCondor + containers