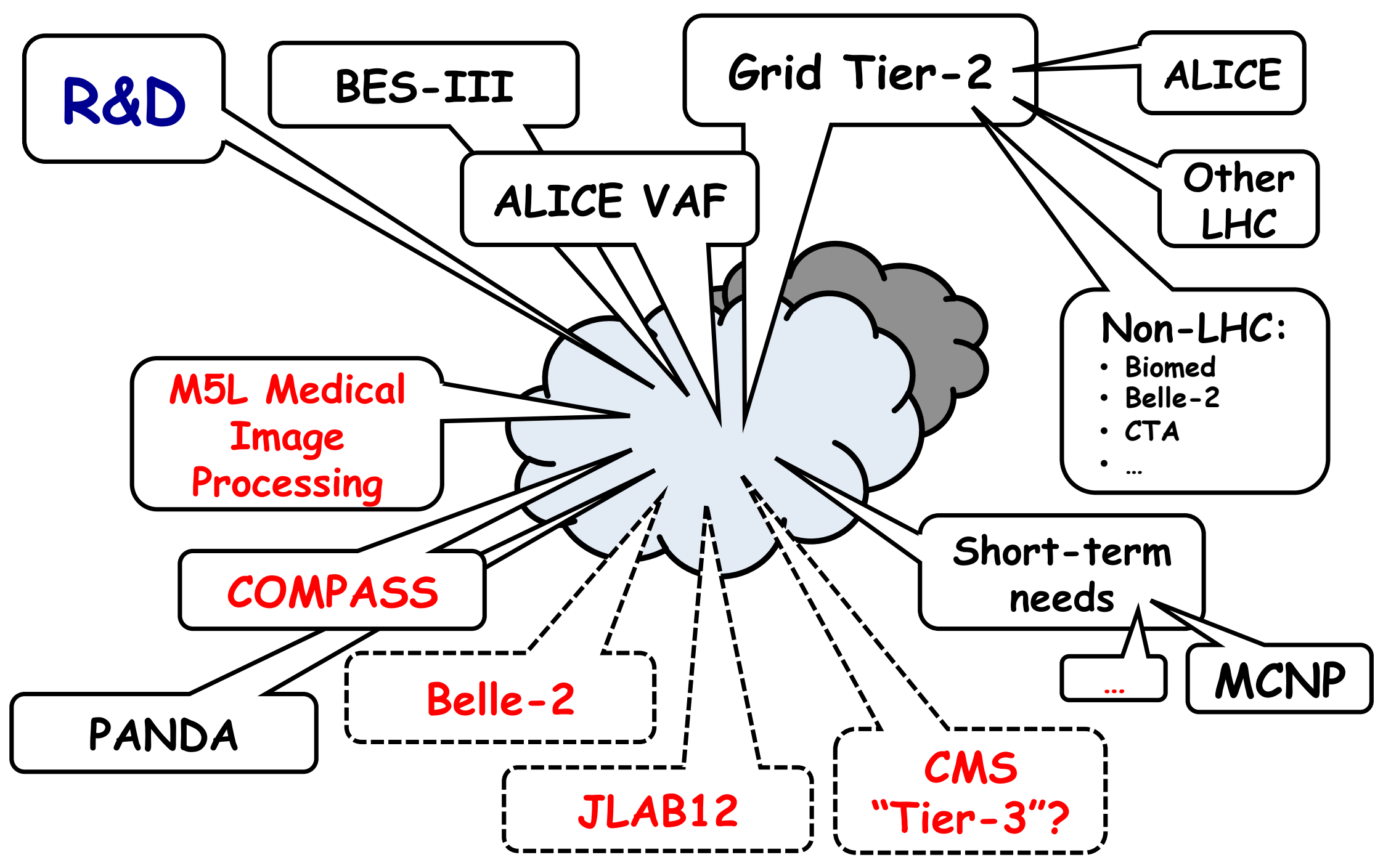
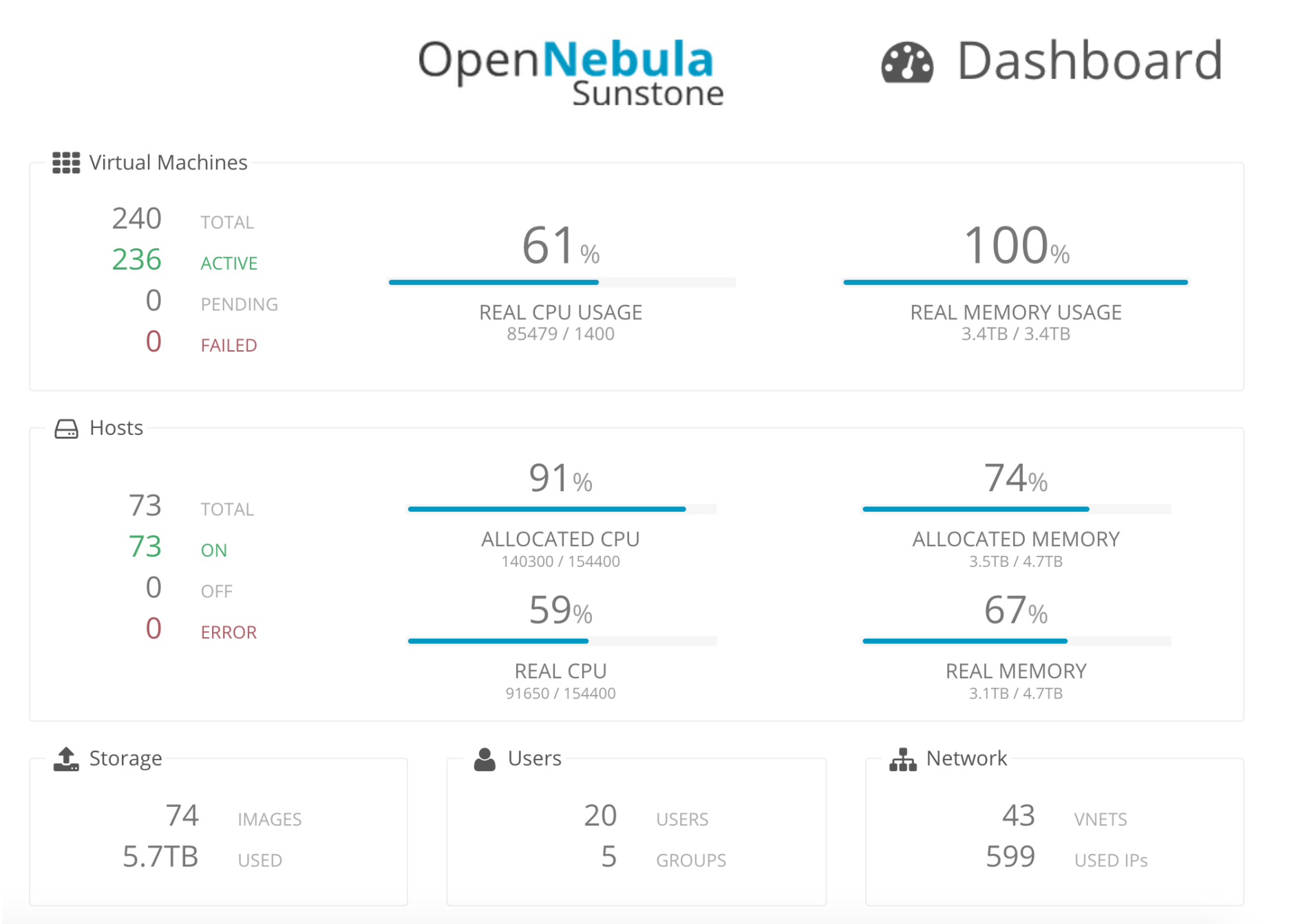
MANAGING COMPETING ELASTIC GRID AND CLOUD SCIENTIFIC COMPUTING APPLICATIONS USING OPENNEBULA

The INFN-Torino Computer Centre

- Born as a WLCG Tier-2 site for the ALICE experiment at the LHC
- Then become a Tier-2 site for the BES-III experiment at IHEP, Beijing
- Now a fully virtualized cloud infrastructure comprising ~75 hosts in two clusters managed by the OpenNebula cloud controller
- Currently providing computing power to a number of applications:
    - WLCG Tier-2 sites (LHC VOs, biomed, PANDA and others)
    - BES-III Tier-2 site (a separate middleware instance)
    - Interactive Virtual Analysis Facility for ALICE
    - Theoretical computation batch farm
    - On-demand remote medical image processing
    - Several smaller application-specific "Virtual Farms"





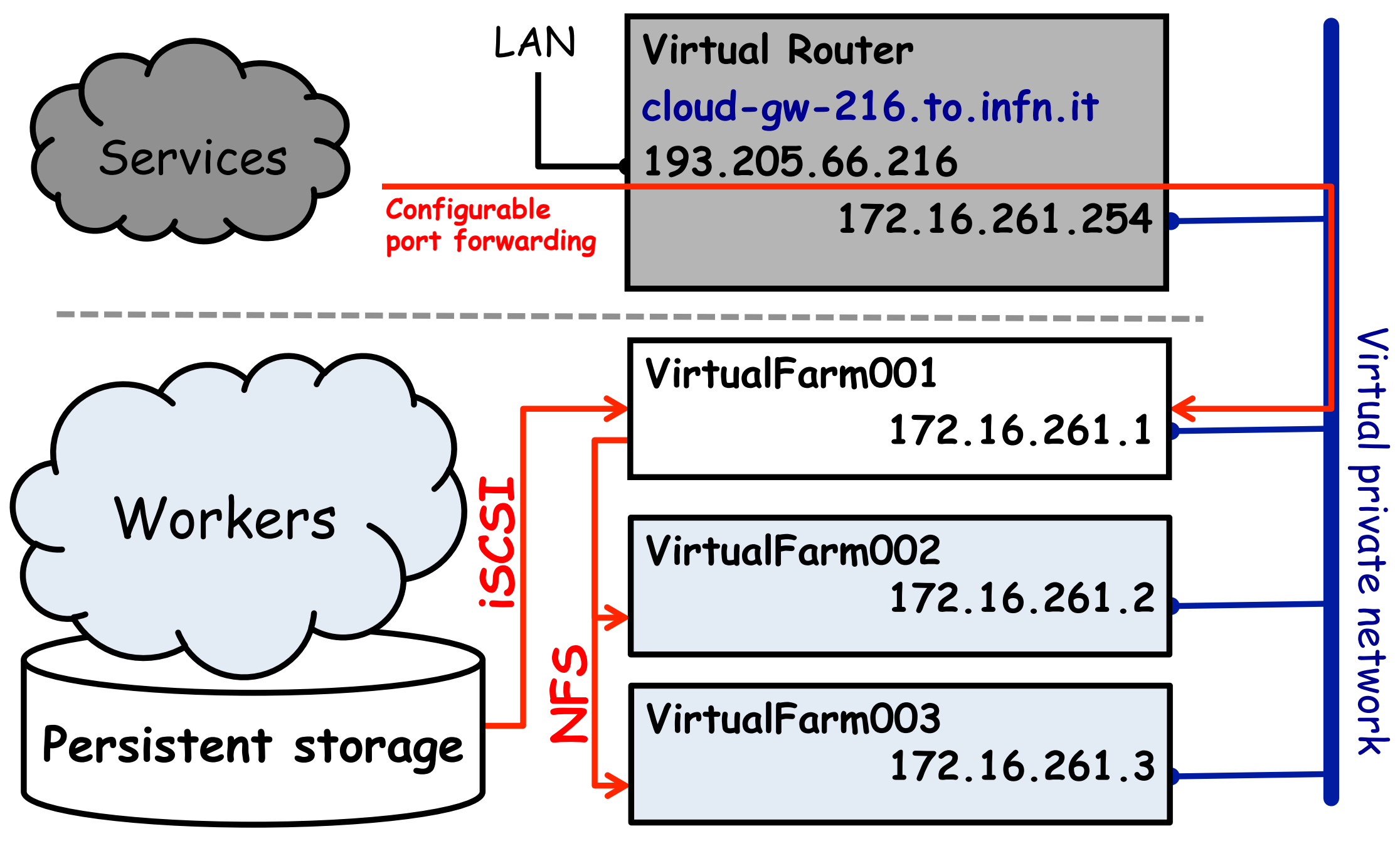
Virtual Machines
240 TOTAL
236 ACTIVE
0 PENDING
0 FAILED
61% REAL CPU USAGE 85479 / 1400
100% REAL MEMORY USAGE 3.4TB / 3.4TB

Hosts
73 TOTAL
73 ON
0 OFF
0 ERROR
91% ALLOCATED CPU 140300 / 154400
74% ALLOCATED MEMORY 3.5TB / 4.7TB
59% REAL CPU 91650 / 154400
67% REAL MEMORY 3.1TB / 4.7TB

Storage
74 IMAGES
5.7TB USED

Users
20 USERS
5 GROUPS

Network
43 VNETS
599 USED IPs



Elastic applications: Virtual Farms

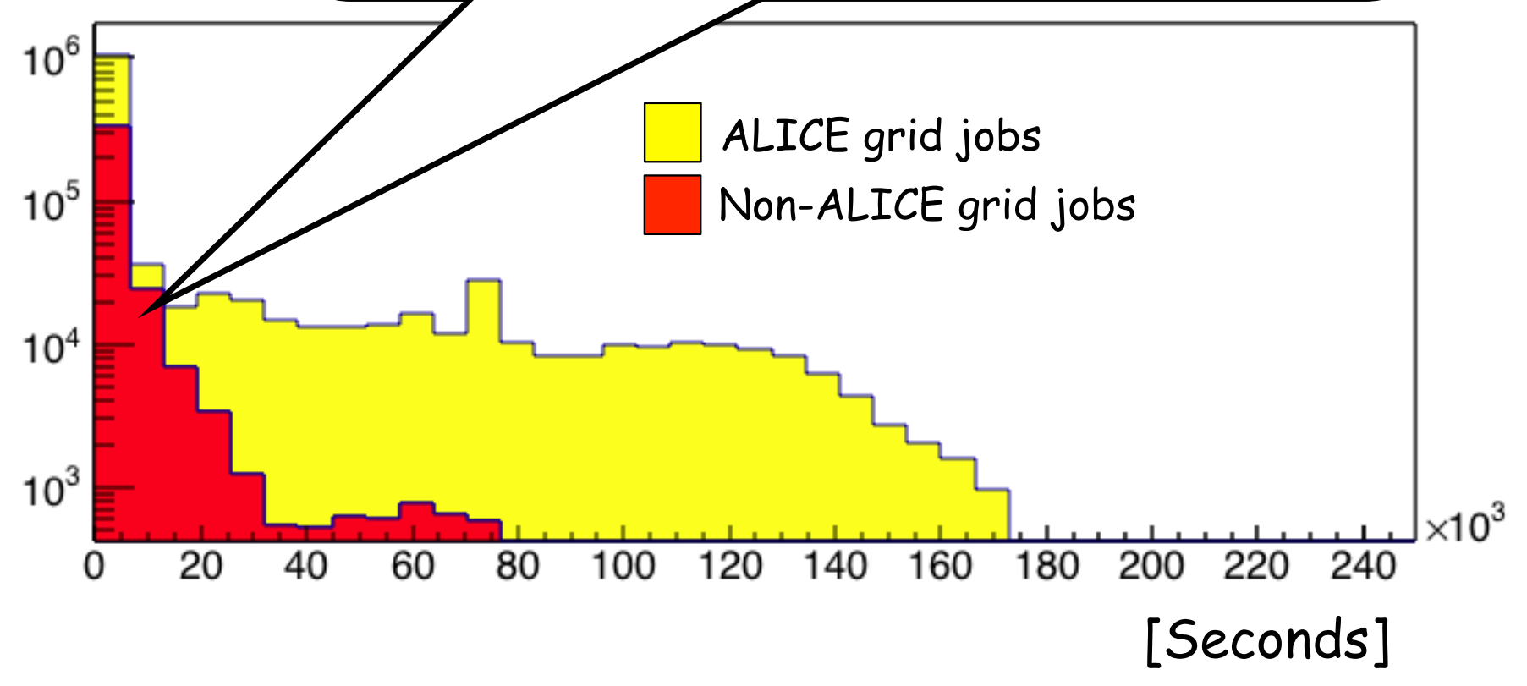
- Usage changes with time (e.g. in bursts)
- Easy to locate idle nodes to undeploy

Anelastic applications: Grid Farms

- Work in saturated regime
- Nodes are never idle

Job duration distribution show no clear pattern

No easy way to choose multi-core VMs to undeploy



TWO PATHS TO ELASTICITY: ELASTIQ AND ONEFLOW

- **Elastiq** is a custom Python daemon [https://github.com/dberzano/elastiq]
- uses the EC2 interface to communicate with the cloud-controller (can work on any cloud)
- plugin implemented for HTCondor LRMS (cloud-aware)
- SCALE UP: when jobs in queue
- SCALE-DOWN: when specific VM is idle

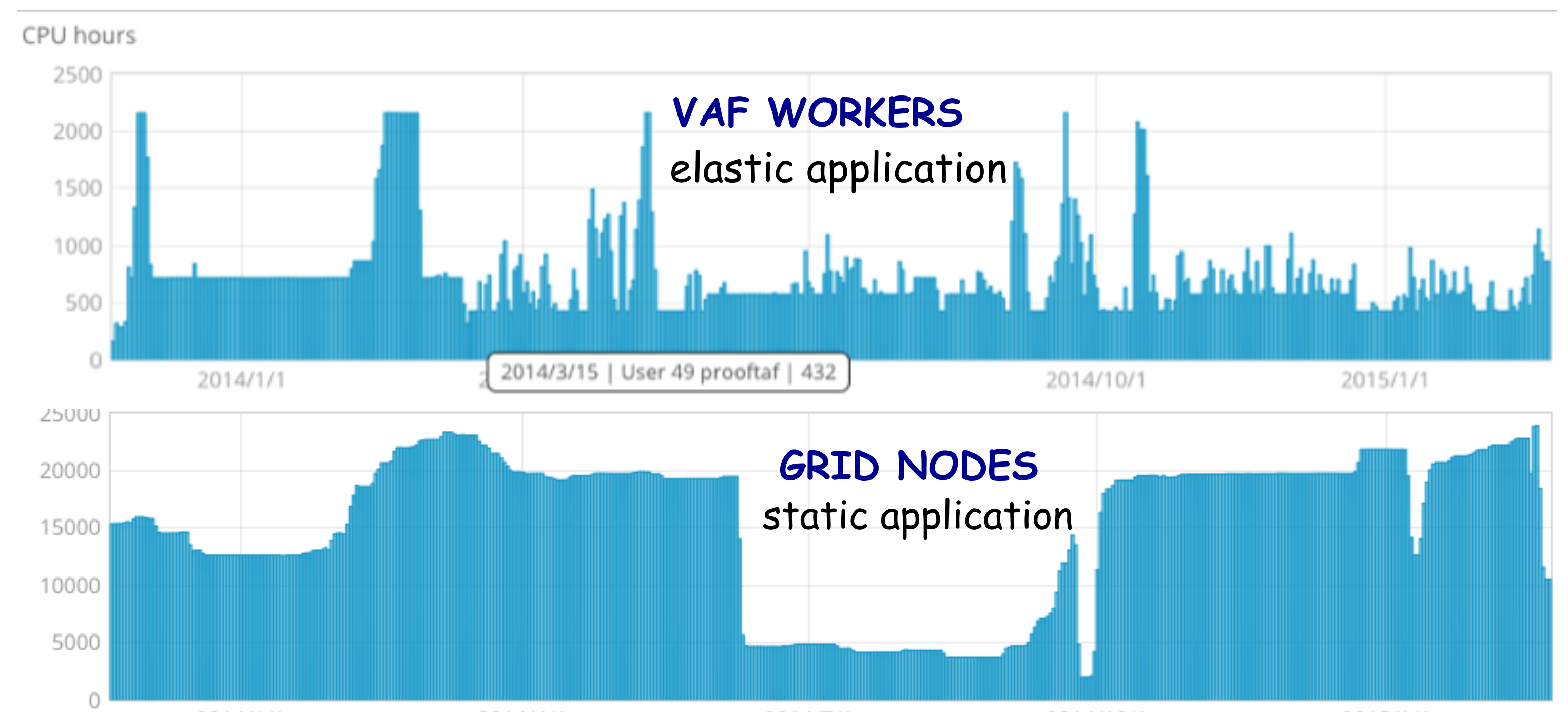
Example use-case: The ALICE Virtual Analysis Facility (VAF) [J. Phys.: Conf. Ser. 368 (2012) 012019]

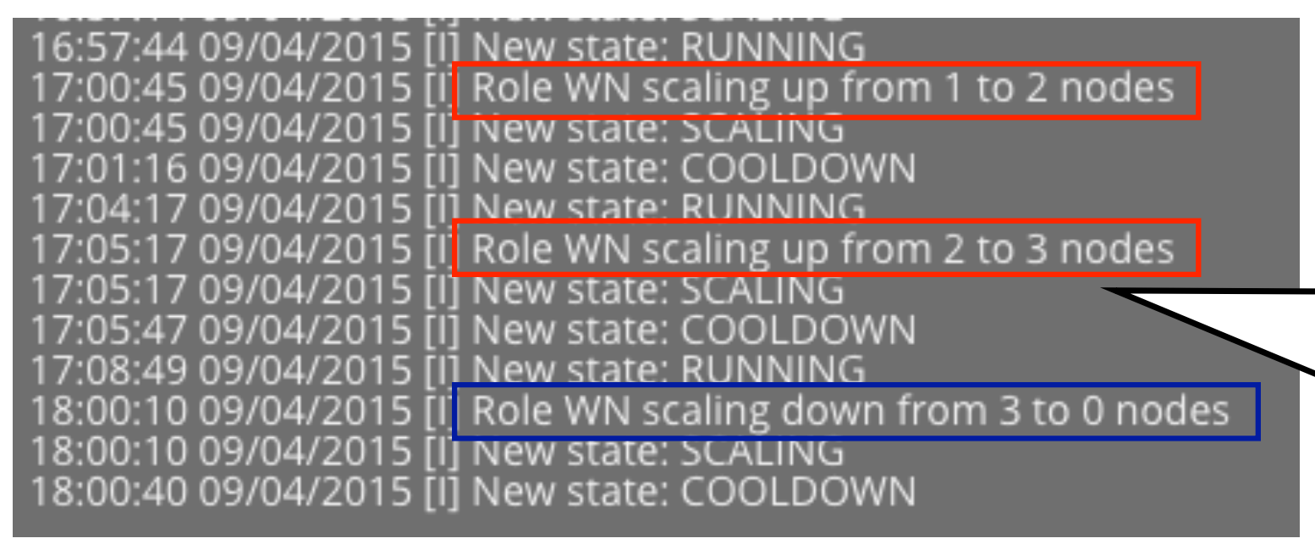
- the tenant deploys 1 single VM (the master)
- Elastiq configuration and workers configuration specified in master context

- **OneFlow** is an OpenNebula tool to deploy clusters of VMs with dependencies
- designed for load balancing applications (user cannot currently decide which VMs to undeploy)
- SCALE UP: 1 VM at the time when there are queued jobs
- SCALE DOWN: when all jobs are finished

Example use-case: BESIII GRID Tier2

- master service is a CREAM CE
- slaves are DIRAC GRID worker-nodes
- LRMS is PBS (not cloud-aware)
- worker nodes publish the number of queued/running jobs to OneGate

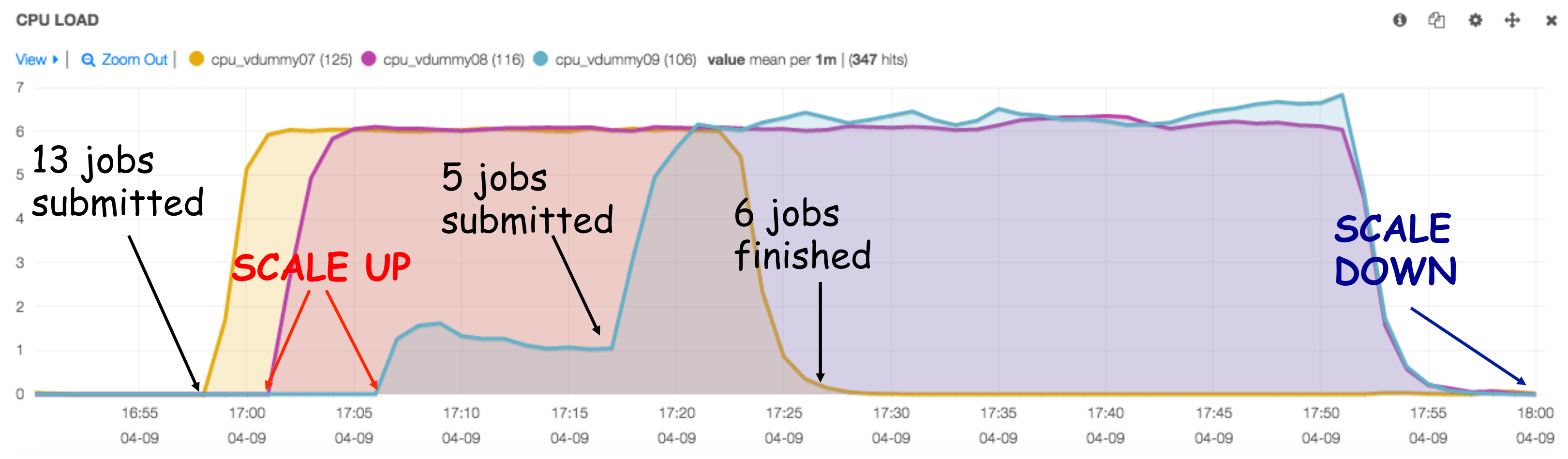




```
16:57:44 09/04/2015 [I] New state: RUNNING
17:00:45 09/04/2015 [I] Role WN scaling up from 1 to 2 nodes
17:00:45 09/04/2015 [I] New state: SCALING
17:01:16 09/04/2015 [I] New state: COOLDOWN
17:04:17 09/04/2015 [I] New state: RUNNING
17:05:17 09/04/2015 [I] Role WN scaling up from 2 to 3 nodes
17:05:17 09/04/2015 [I] New state: SCALING
17:05:47 09/04/2015 [I] New state: COOLDOWN
17:08:49 09/04/2015 [I] New state: RUNNING
18:00:10 09/04/2015 [I] Role WN scaling down from 3 to 0 nodes
18:00:10 09/04/2015 [I] New state: SCALING
18:00:40 09/04/2015 [I] New state: COOLDOWN
```

Pros of the OneFlow approach:

- easy to configure a cluster as a single service from the OpenNebula GUI
- scale up/down manually
- change worker-node context on the fly



OUTLOOK

VM Management tools

- OneFlow in its current implementation is not optimal for this use case
- Most LRMSs used in grid sites (e.g. PBS/Torque) are not cloud-aware and cannot easily cope with nodes appearing and disappearing
- HTCondor is a better candidate
- OneFlow for large saturated use cases, Elastiq for smaller virtual farms

Scale down policies

- Large 6-8 core Virtual Worker Nodes are not ideal for this use case
- No hint from job statistics means wasted resources while the node waits for longer jobs to finish
- Need to keep some (small) WNs in draining mode all the time

Next steps

- Split the ALICE farm: static large WNs to keep the number of VMs low, smaller WNs for the elastic component
- Deploy a separate HTCondor CE for the elastic component
- Define policies and parameters for scale up and scale down of this application

The present work is supported by the Istituto Nazionale di Fisica Nucleare (INFN) of Italy and is partially funded under contract 20108T4XTM of Programmi di Ricerca Scientifica di Rilevante Interesse Nazionale (PRIN, Italy).

Stefano Bagnasco[1], Dario Berzano[2], Stefano Lusso[1], Massimo Masera[1,3], Sara Vallero[1,3] on behalf of the STOA-LHC project

[1] Istituto Nazionale di Fisica Nucleare; [2] CERN; [3] Department of Physics, University of Torino