CERN

Automation & Monitoring in the CERN Condor pool

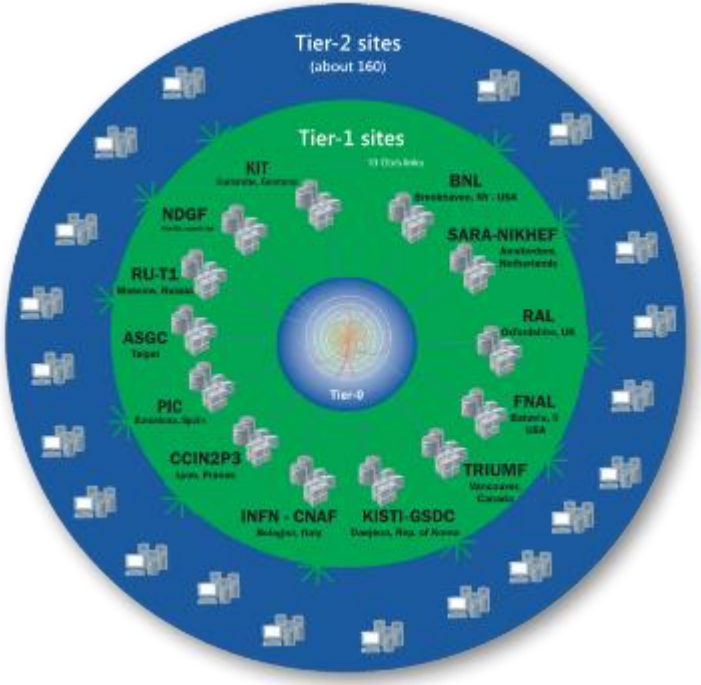
Luis Fernández Álvarez <luis.fernandez.alvarez@cern.ch>

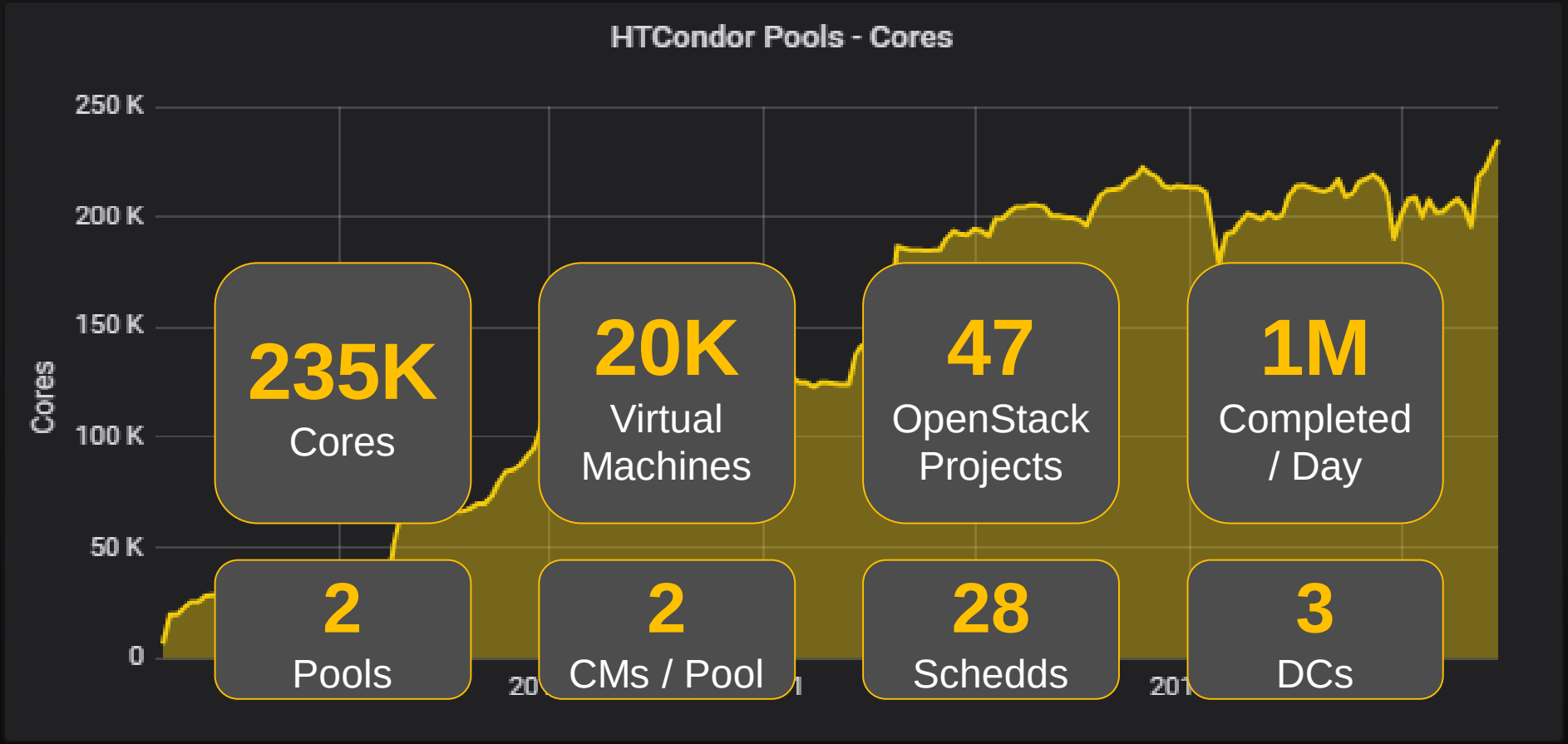
The Batch Service

The Batch Service @ CERN IT

Provides Tier-0 compute power via HTCondor to WLCG.

- Process CPU intensive workload ensuring fairshare among various user groups.
- Maximize utilization, throughput, efficiency.
- It runs jobs from the Grid and from local CERN departments.

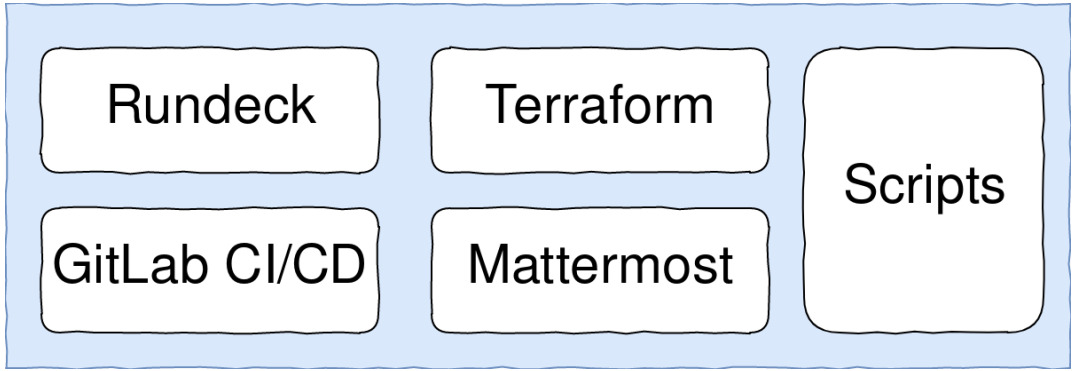



HTCondor Pools - Cores
250 K
200 K
150 K
100 K
50 K
0
Cores
235K
Cores
20K
Virtual Machines
47
OpenStack Projects
1M
Completed / Day
2
Pools
2
CMs / Pool
28
Schedds
3
DCs

Automation

Automation: why?

- Reduce the time doing tedious ops tasks.
- Avoid hidden magic scripts / procedures.
- Knowledge sharing.

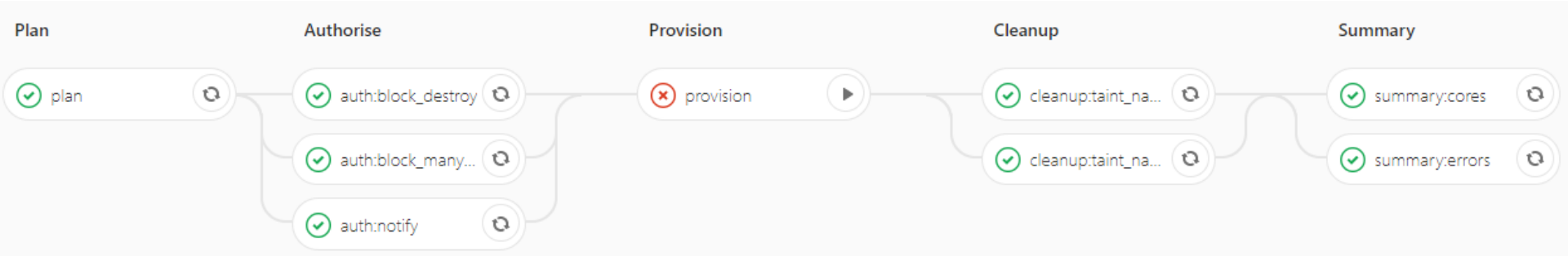


Automation: provisioning (I)

- **Goal:** keep our 47 OpenStack projects always full.
- Before: nightly script launched via Rundeck
  - Homegrown, difficult to maintain.
  - Navigate through Rundeck logs to guess state.
  - Not dealing with machines in ERROR state.
- **Solution**: Terraform + GitLabCI + Mattermost

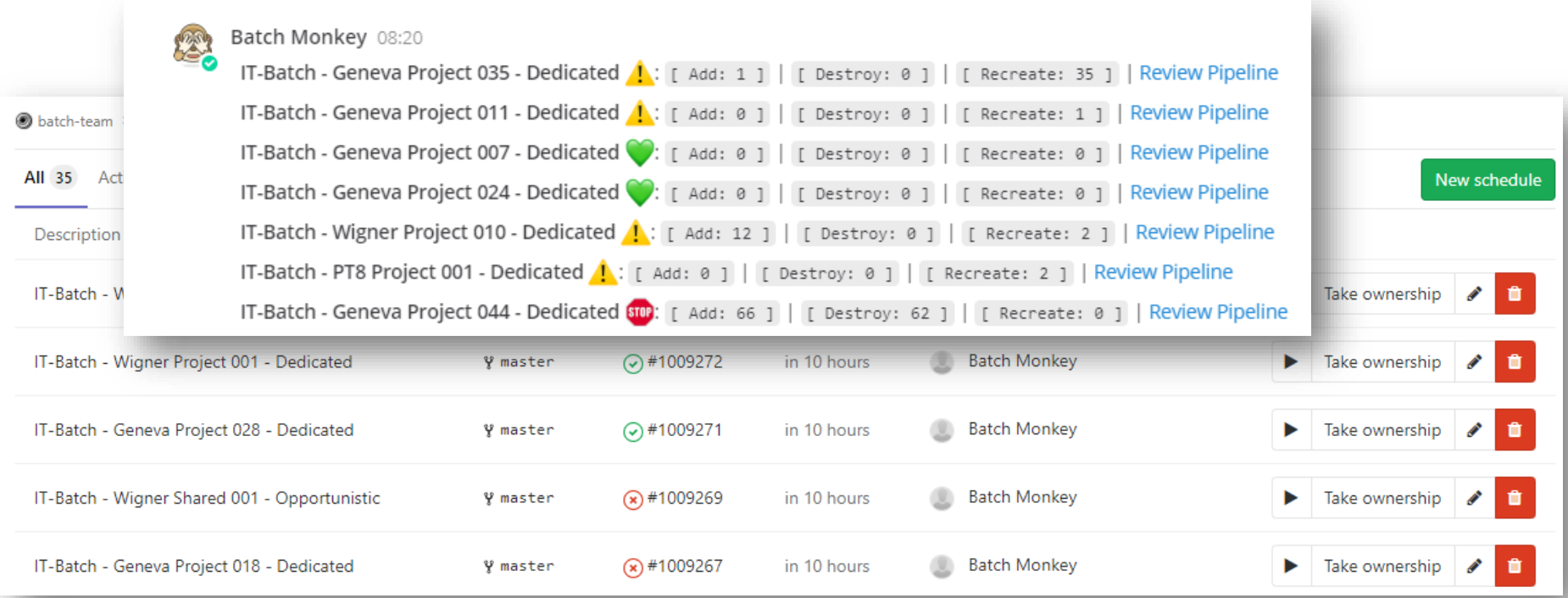
Automation: provisioning (II)

- The pipeline:



Plan	**Authorise**	**Provision**	**Cleanup**
Artifact	Don’t destroy! Don’t recreate > 10%! Notify Operators	6 hours delay	Invalidate names

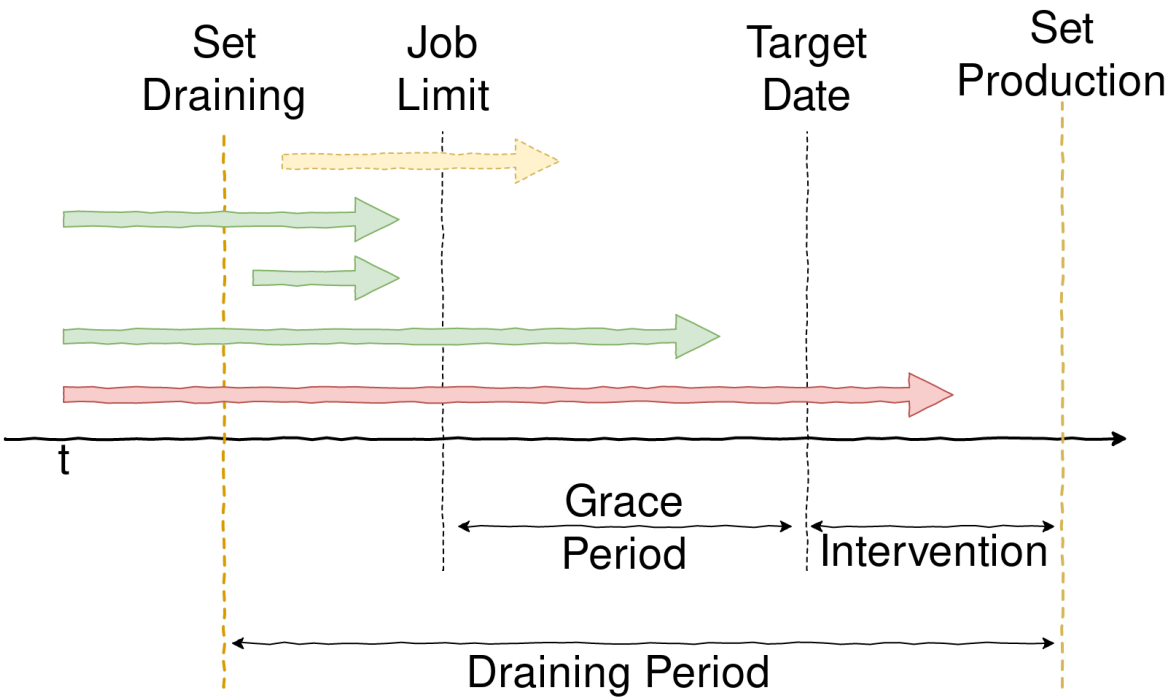
Automation: provisioning (III)



Automation: draining (I)

- Different impact:
  - Low: broken hypervisors
  - High: Spectre/Meltdown, Wigner decommissioning, CentOS7 migration,…
- Simple workflow: stop accepting jobs, wait for existing jobs to finish and reboot if needed.

Automation: draining (II)

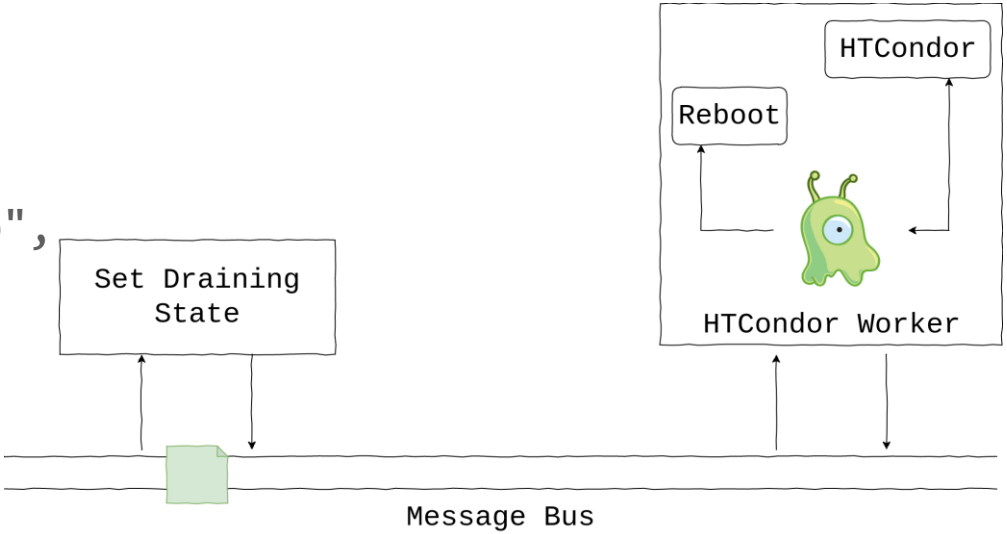


- Job limit + Target date combination for different scenarios
- Interventions: reboot, fix, upgrade, hardware...

Automation: draining (III)

- Different homegrown services involved in the implementation:
  - **roger**: state
  - **megabus**: message bus
  - **brainslug**: node agent
- Payload:

```
{
 "Reason":      "WignerDeco",
 "Reference":   "BBC-2445",
 "TargetDate": 1569218400,
 "JobDate":    1569218400,
 "Reboot":      "None"
}
```



Monitoring

Infrastructure, Nodes & Service

Monitoring: why?

- Have a better understanding of how our system works.
- Improve the quality of the service.
- Identify the misuse of the pool.
- …

Monitoring targets

Service

- *Is HTCondor running smoothly?*
- Startds, schedds, central managers…

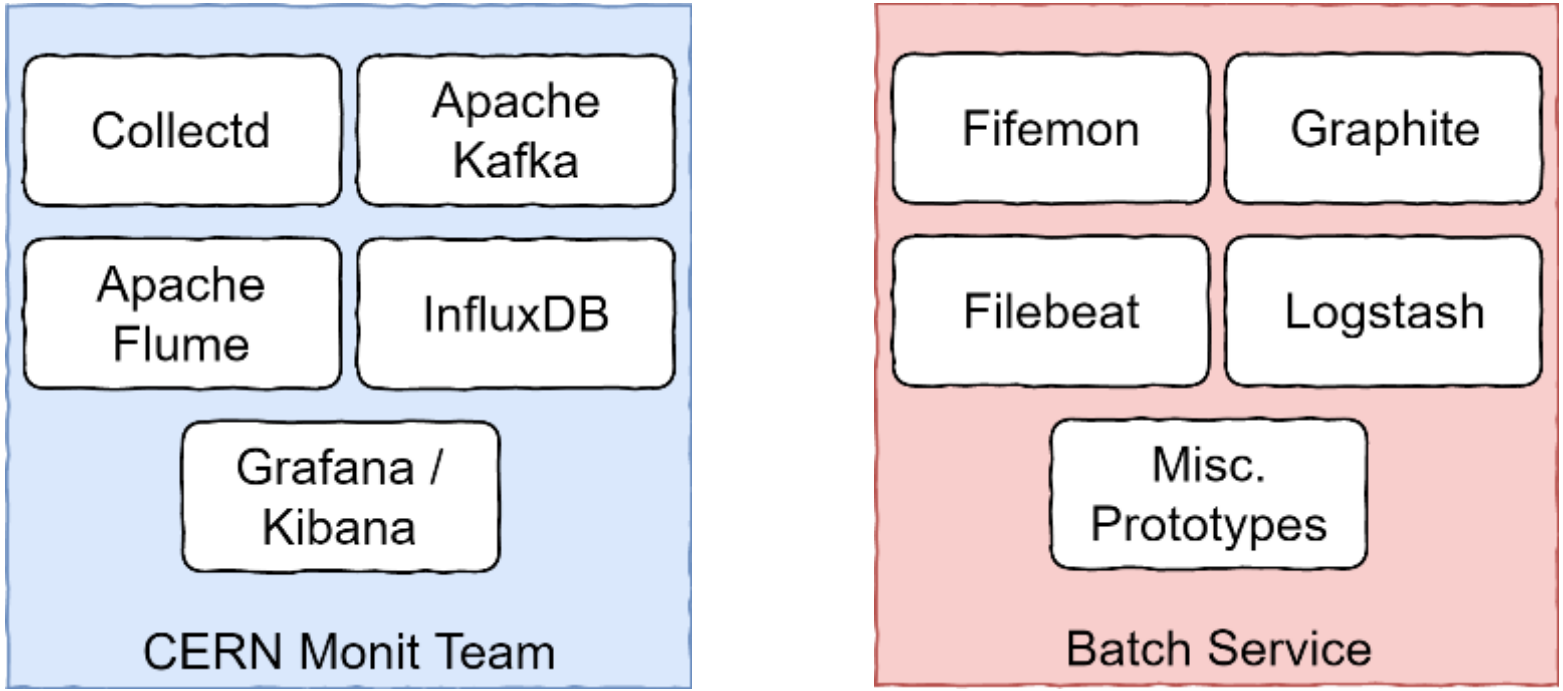
Nodes

- *Are the nodes in the pool healthy?*
- Virtual machines & physical nodes

Infrastructure

- *Is the underlying infra working fine?*
- Mainly OpenStack

Monitoring - Ecosystem



Monitoring: infrastructure (I)

- Worker nodes run on OpenStack, critical component in our pools
- Virtual machines scattered across **47 projects** and **3 datacenters**.
- Main areas adding noise:
  - Neutron vs nova-network projects.
  - Software bugs, mainly: kernel, qemu-kvm, libvirtd.
- Cloud operations impacting VM availability:
  - Replace hard disk, motherboards, opportunistic resources, etc.

Monitoring: infrastructure (II)

- Questions to answer at this stage:
  - *Are production machines going to error state?*
  - *Is our automated provisioner successfully creating new nodes?*
  - *Is there any cloud intervention affecting our resources? When and how?*
- Actions required on both sides

Monitoring: infrastructure (III)

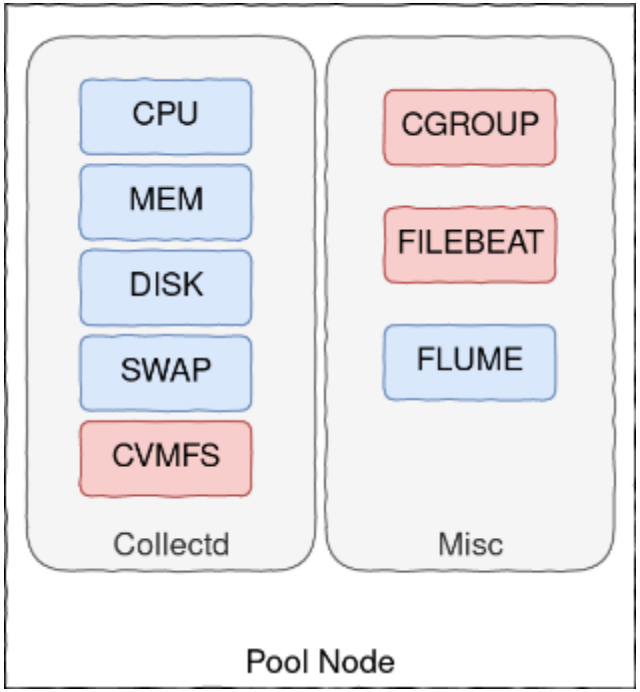
- OpenStack metrics collected by Cloud team and stored in CERN MONIT infra:
  - VMs per project and their states
  - Consumed by us via Grafana
  - Alarms defined in Grafana to spot errors
- Cloud team publishes interventions in a programmatic way via a message bus:
  - WIP: listen and take actions automatically

Usage

cell_crit_proje... 100%	cell_gva_proje... 127%	cell_gva_proje... 99%	cell_gva_proje... 98%	cell_gva_proje... 100%	cell_gva_proje... 100%	cell_gva_proje... 100%	cell_gva_proje... 50%
cell_gva_proje... 100%	cell_gva_proje... 50%	cell_gva_proje... 100%	cell_gva_proje... 100%	cell_gva_proje... 100%	cell_gva_proje... 100%	cell_gva_proje... 100%	cell_gva_proje... 100%
cell_gva_proje... 100%	cell_gva_proje... 99%	cell_gva_proje... 50%	cell_gva_proje... 83%	cell_gva_proje... 90%	cell_gva_proje... 99%	cell_gva_proje... 100%	cell_gva_proje... 100%
cell_gva_proje... 99%	cell_gva_proje... 77%	cell_gva_proje... 99%	cell_gva_proje... 100%	cell_gva_proje... 100%	cell_gva_proje... 78%	cell_pt8_proje... 100%	cell_pt8_proje... 100%
cell_pt8_proje... 99%	cell_pt8_proje... 100%	cell_pt8_proje... 100%	cell_pt8_proje... 39%	cell_pt8_proje... 100%	cell_pt8_proje... 98%	cell_wig_proje... 100%	cell_wig_proje... 99%
cell_wig_proje... 98%	cell_wig_proje... 100%	cell_wig_proje... 99%	cell_wig_proje... 100%	cell_wig_proje... 100%	cell_wig_share... 85%	cell_wig_share... 91%	

Monitoring: nodes

- What to worry about in this layer?
- > Storage not accessible, nodes running out of space, resource usage
- > Impact of: config changes or sw releases.



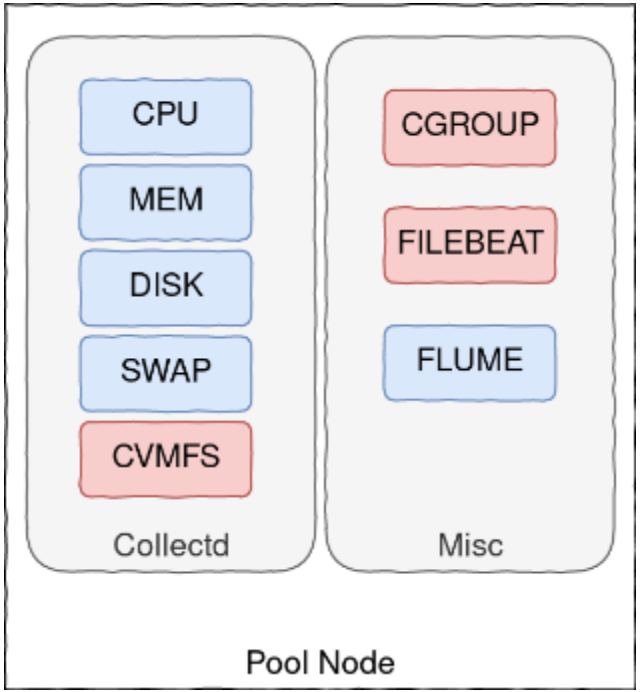
Monitoring: nodes

Base monitoring provided by CERN IT:

- Collectd + Apache Flume for common node resources: cpu, load, disk, heartbeat, etc. (InfluxDB)
- Rsyslog + Apache Flume. Shipped to ElasticSearch

Extended by us:

- CVMFS plugin for collectd: https://github.com/cvmfs/collectd-cvmfs
- Log based metrics (collectd tail) for EOS logs (WIP)
- Process monitoring: nscd, mounts,…
- Not collectd based (yet): cgroup monitoring



Monitoring: nodes (cgroups)

- Prototype: CGroups Simple (CGS)
- Fine-grained cgroup metrics to be exposed to users in the future
- Photographers, transformers & writers
- Standalone, evaluating transforming it into collectd plugin

Monitoring: nodes (heartbeat)

- Simple but interesting metric: collectd plugin + external processor
- Given its design and our scale it's a good indicator of how smoothly things are running

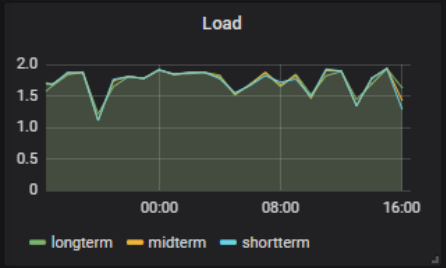
Monitoring: nodes (alarms)

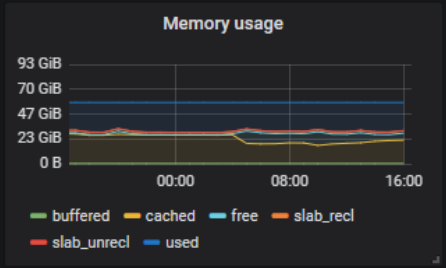
- Alarms:
  - Collectd: threshold plugin (local to the machine)
  - Grafana alerts for aggregated metrics
- Actuators for collectd alarms:
  - Based on MONIT collectd plugin.
  - Space cleanup, fix known issues, etc.

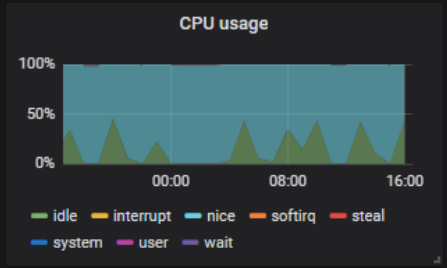
Internal > HTCondor - Worker Details

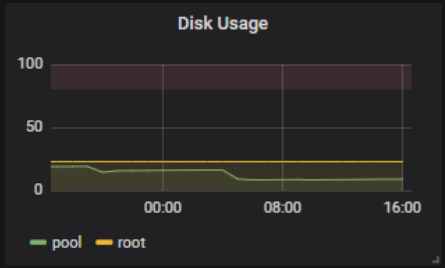
Last 24 hours | 30m

Server Metrics

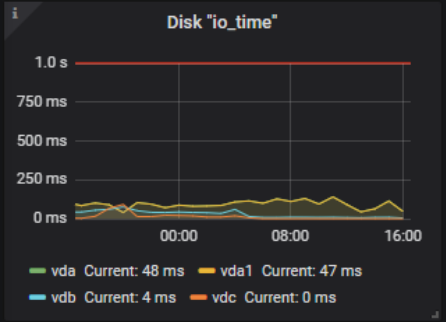


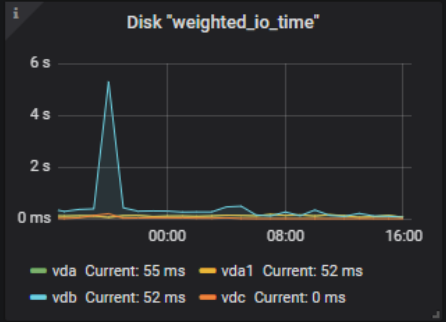


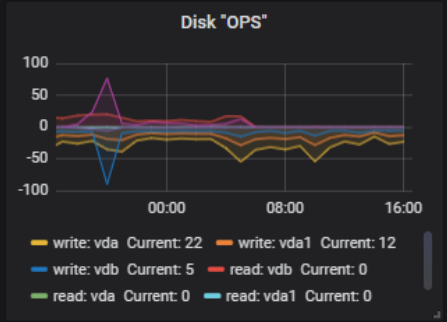


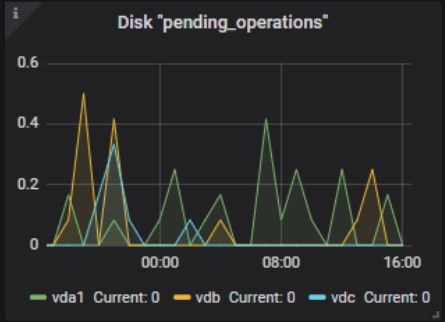


Server IO (Disk & Network)

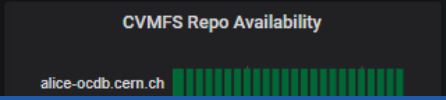


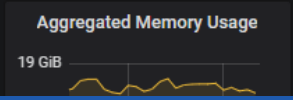


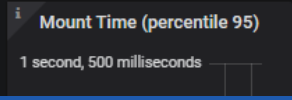


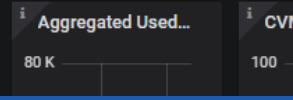


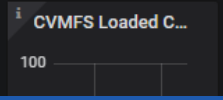
Worker Blocks (AFS, EOS, CVMFS,...)

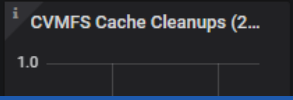












Monitoring: service

- At this level, we want to make sure that:
  - Condor is healthy, users can submit jobs, we're not hitting scalability issues,…
- Based on: http://fifemon.github.io/
- Graphite backend
  - Metrics namespaced per pool (shared and T0)
  - 90Gb of data after 2 years
- Fifemon probes running every 4 minutes
  - Extended with CERN pool specific metrics
  - TODO: include the CEs into the probes.

Monitoring: service (events)

- EventLog sent to ElasticSearch
  - Based on Kevin Retzke work
  - Local filebeat prospector & Logstash instance
- Rate ~15K events / second
- Use of `EVENT_LOG_JOB_AD_INFORMATION_ATTRS`
- Single worker
  - Revisit the usage of “aggregate” plugin
  - Increase number of workers and instances

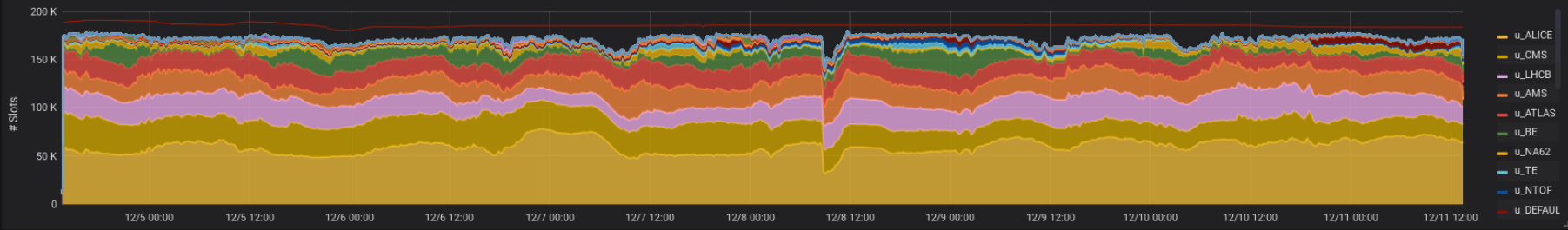
C5 Summary

Last 7 days

cluster cernprod

Average Number of Jobs Running Concurrently	Total Job Runtime (Last Week)	Average Efficiency (Last week)
132669	1155 years	74%

Running Slots By Experiment



CPU Delta (7 days)	Memory Delta (7 days)	Disk Delta (7 days)
-2364	-14 TiB	-98 TiB

Efficiencies by DataCentre (Last Week)

Metric	Min	Max	Avg	Current
wigner	52.15%	76.28%	67.22%	69.93%
meyrin	65.07%	88.72%	79.32%	86.99%

Efficiencies by Experiment/DataCentre

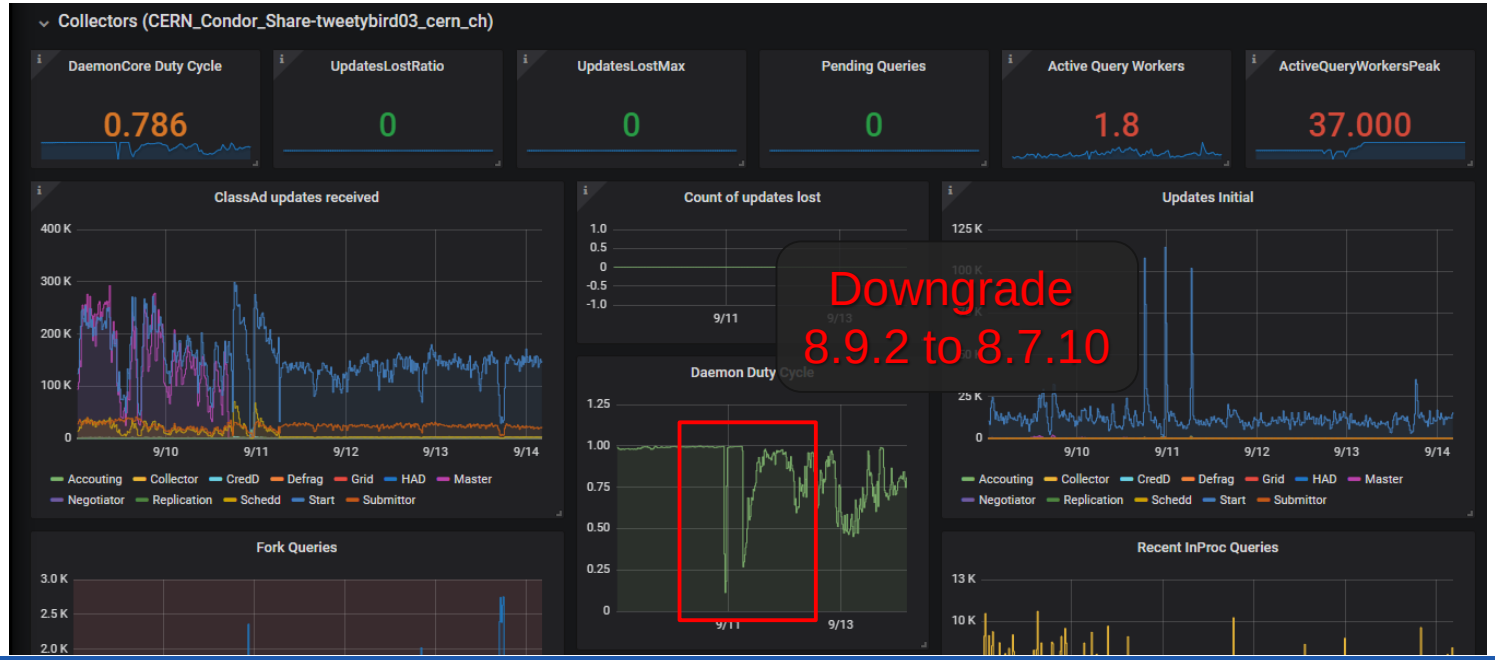
Metric	Min	Max	Avg	Current

Monitoring: scenarios

- …when is all of this worth it?

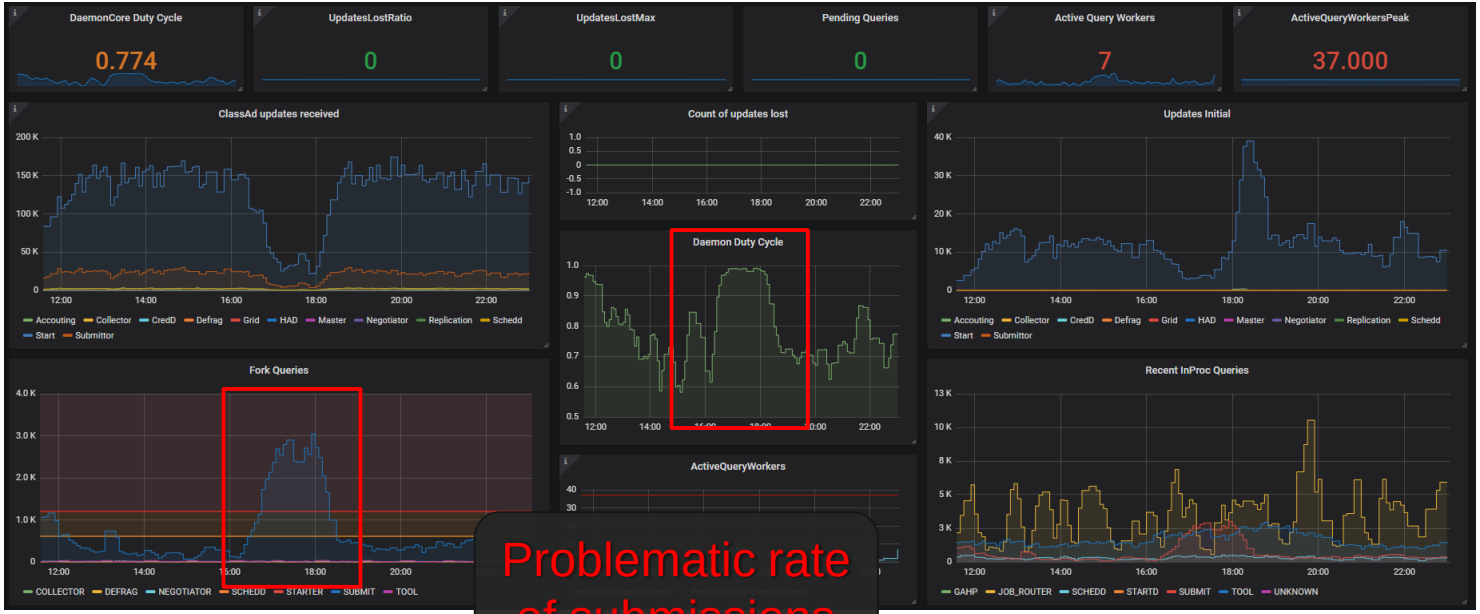
Monitoring: scenarios (I)

- Impact of software releases:



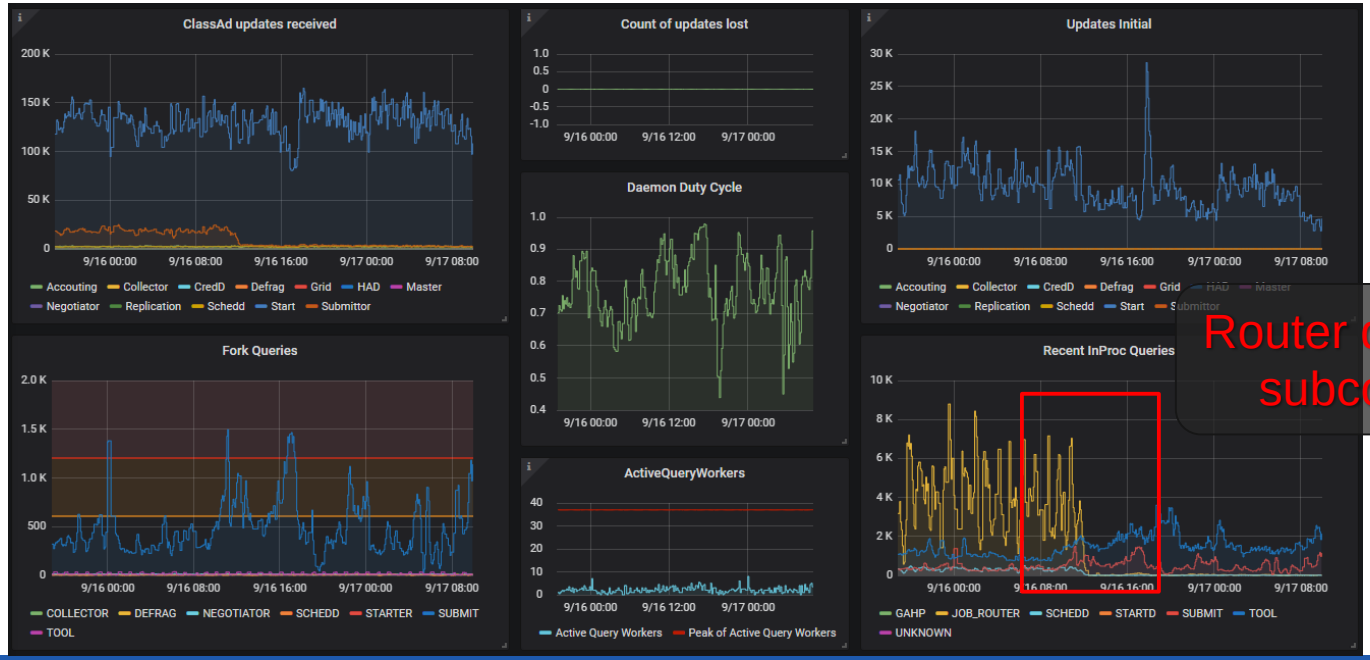
Monitoring: scenarios (II)

- Better understanding of the pool (issues, bottlenecks):



Monitoring: scenarios (III)

- Improve the pool, dedicated subcollector for schedds:



Monitoring: scenarios (IV)

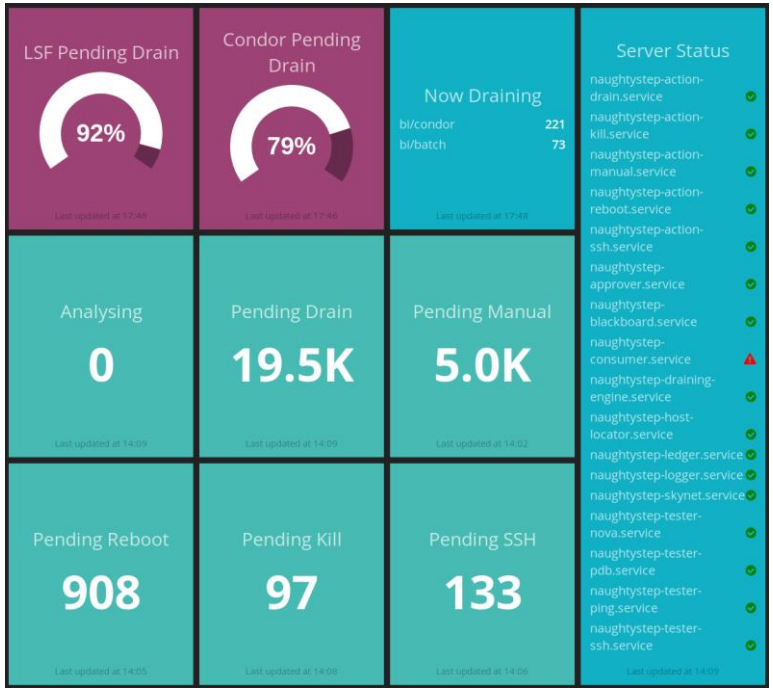
- How to optimize our resources?

Use spare power…

What’s next?

Next

- We don't want to waste our time…
  - …babysitting 20K nodes.
  - …or looking to tens of dashboards.
- Working on prototypes to orchestrate operations:
  - **Draining actions** (based on health checks, cloud interventions)
  - **Self-healing infrastructure**: take decisions based on multiple alarms.
  - **Preventive operations**: continuously drain 10% to deploy new kernel in the worker nodes.
- Keeping an eye on: consul, kafka, spark, etcd, kraken, vitrage, alertmanager.
- Open to ideas and projects… ☺



CERN