



Plancton

an opportunistic computing project based on Docker containers



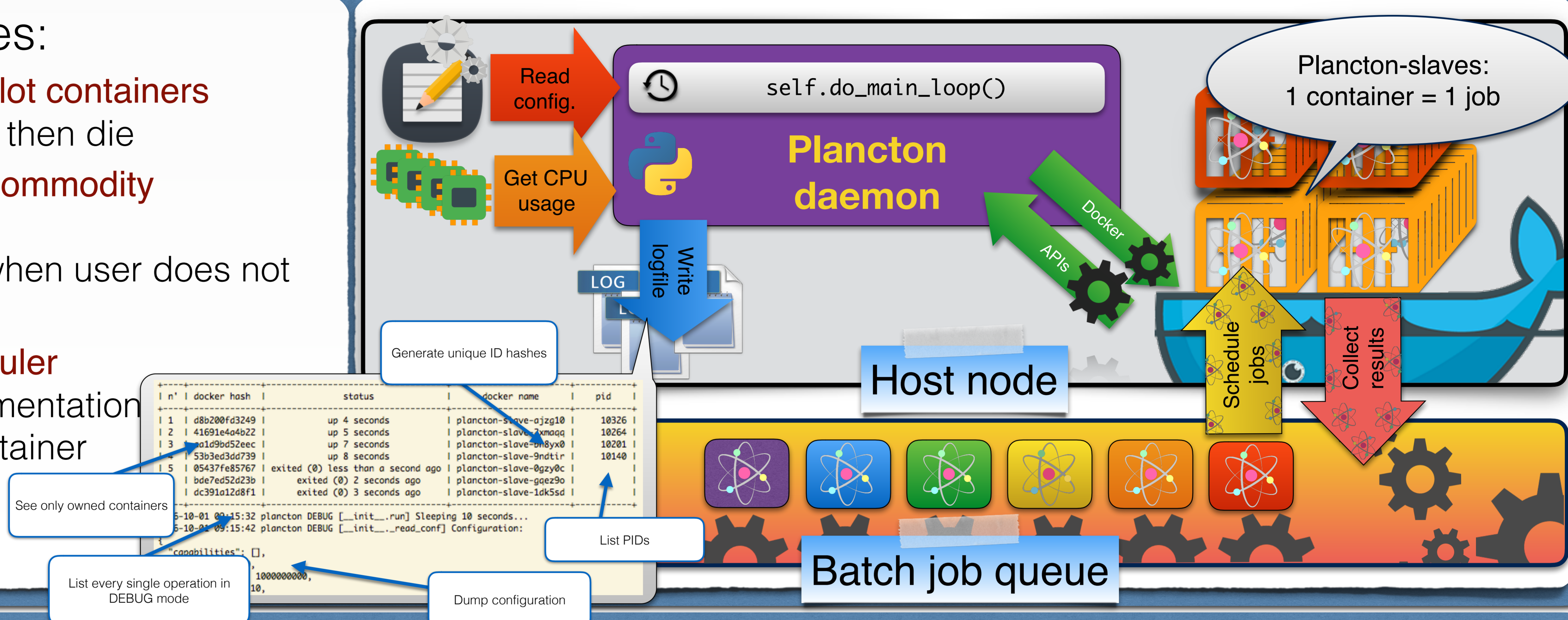
Matteo Concas¹ [matteo.concas@cern.ch], Dario Berzano² [dario.berzano@cern.ch],
Stefano Bagnasco³, Stefano Lusso³, Massimo Masera^{1,3}, Maximiliano Puccio^{1,3}, Sara Vallero³

¹Università degli Studi di Torino • ²CERN - Genève • ³INFN - Torino

The Plancton daemon

What Plancton does:

- Continuously spawn **pilot containers**
→ they execute a task then die
- Opportunistically **use commodity resources**
→ spawn containers when user does not use computer
- Just a **container scheduler**
→ full use-case implementation stays inside the container



A single tool for two use-cases

A sparse **volunteer** farm at ALICE Torino

- Execute prompt unplanned tasks (e.g. quick code testing, ...)
- Exploit commodity user workstations whose resources are shared and used by the very owners
- **Main traits:**
 - Pilot containers as worker nodes → Running **HTCondor** inside
 - **CVMFS on Parrot** → Isolated + consistent runtime environment, no need for **--privileged** (Apparmor/SELinux profiles)
 - Plancton + Docker → Enforce resource limits, continuously schedule new containers when it is possible
- **Setup:** Plancton^[1], Docker^[2], Parrot^[3], HTCondor^[4], CVMFS^[5]

A **dedicated** Grid site for **Monte Carlo**

- Carries out Monte Carlo physics productions as ALICE Grid jobs
- Running on the ALICE HLT development virtual cluster at CERN
- **Main traits:**
 - Pilot containers are Work Queue^[9] workers
 - CVMFS mounted from outside containers
 - ALICE Grid middleware (AliEn) submits to Work Queue via AliEn-WorkQueue → pure pilot approach
- **Setup:** Plancton^[1], Docker^[2], CVMFS^[5], Work Queue^[6], AliEn^[7], AliEn-WorkQueue^[8]

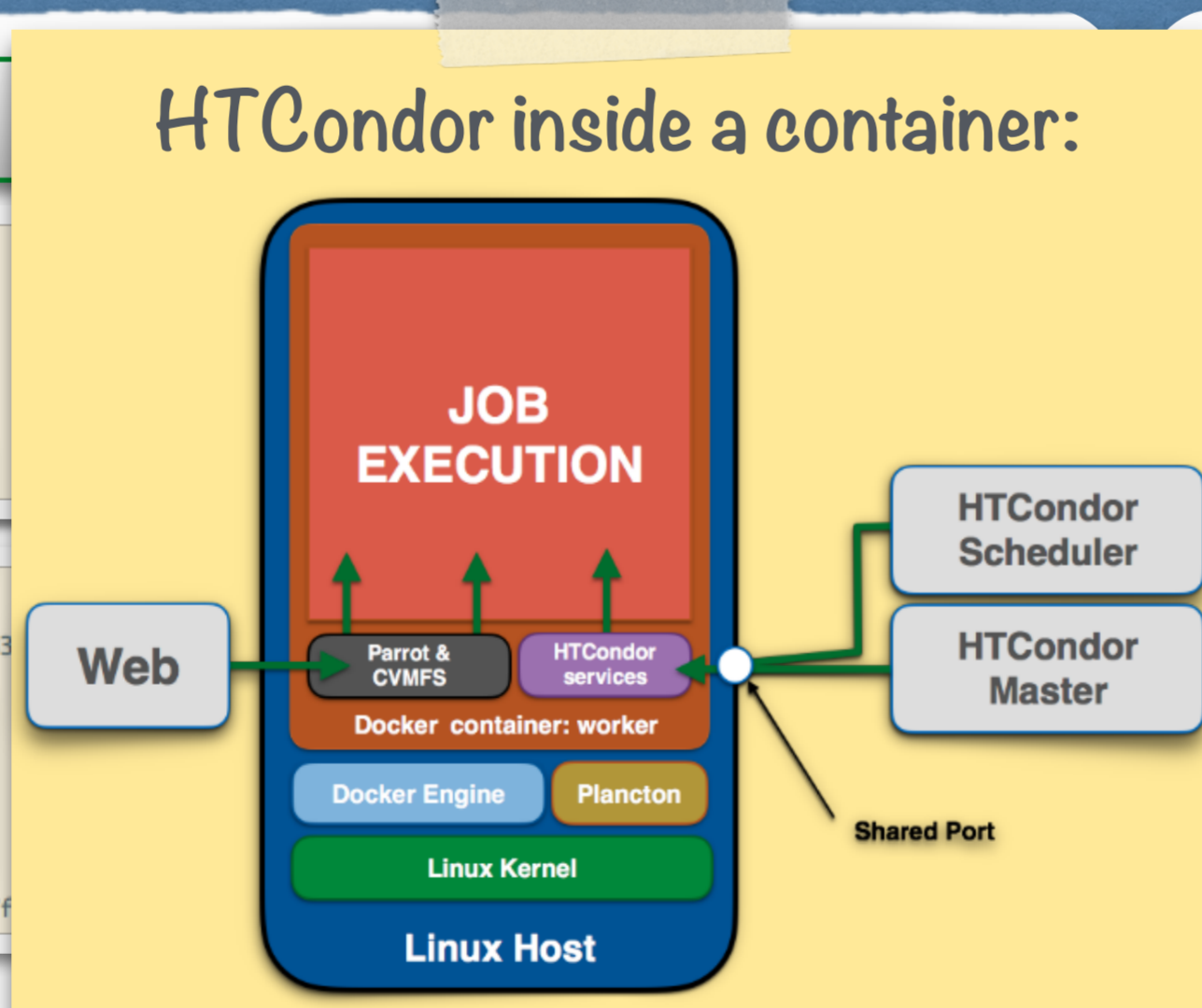
Worker nodes as containers

CVMFS not mounted in docks, accessed with Parrot, no further resources bound.

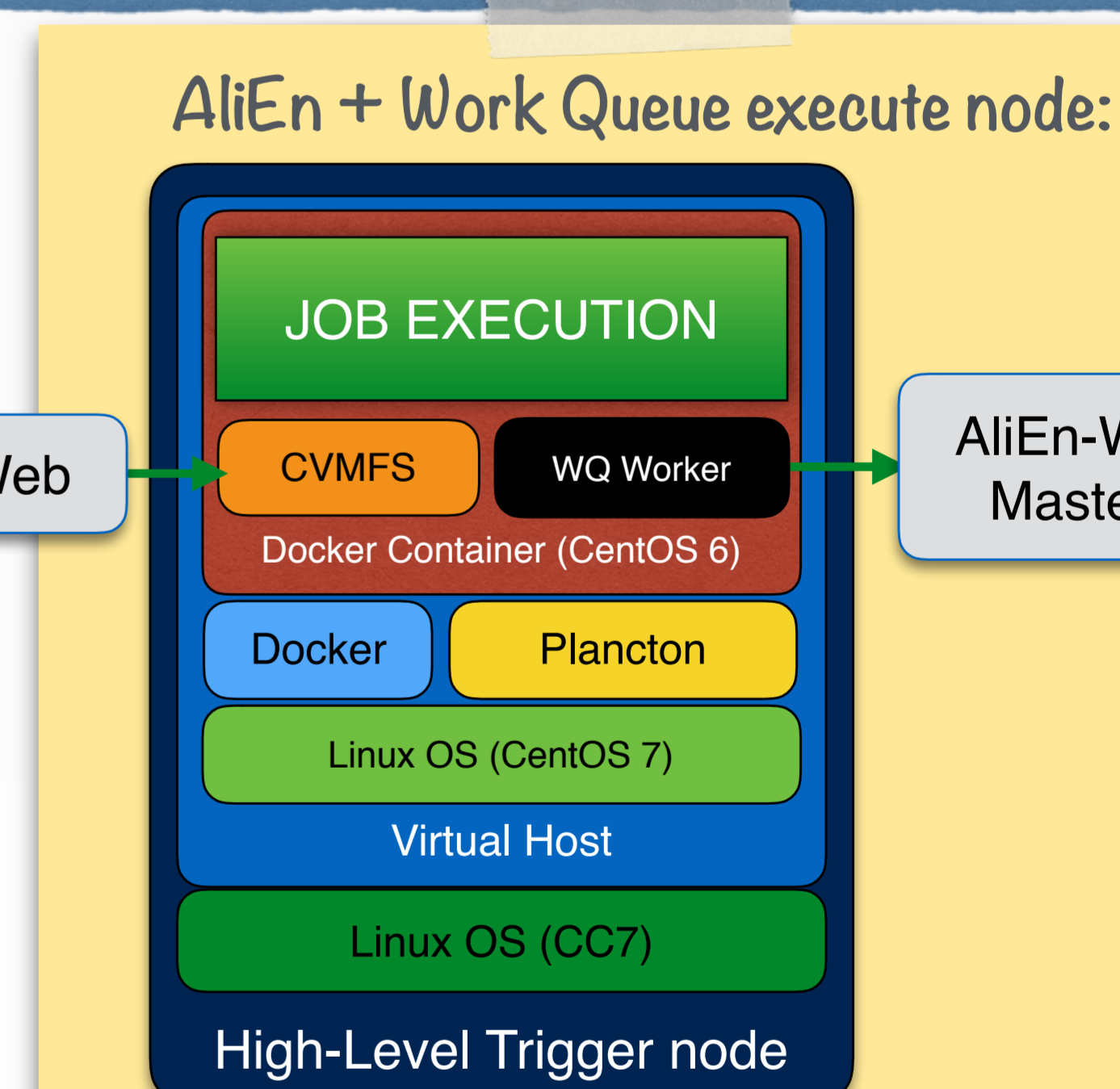
IMAGE	COMMAND	CREATED	STATUS	PORTS
mconcas/centos6-autobuild-container:latest	"/tmp/condor-centos6-"	11 seconds ago	Up 10 seconds	
mconcas/centos6-autobuild-container:latest	"/tmp/condor-centos6-"	12 seconds ago	Up 11 seconds	
mconcas/centos6-autobuild-container:latest	"/tmp/condor-centos6-"	13 seconds ago	Up 12 seconds	
mconcas/centos6-autobuild-container:latest	"/tmp/condor-centos6-"	43 seconds ago	Up 43 seconds	
mconcas/centos6-autobuild-container:latest	"/tmp/condor-centos6-"	44 seconds ago	Up 44 seconds	
mconcas/centos6-autobuild-container:latest	"/tmp/condor-centos6-"	About a minute	Up 1 minute	
mconcas/centos6-autobuild-container:latest	"/tmp/condor-centos6-"	2 minutes ago	Up 2 minutes	

Firewall: no further ports exposed

HTCondor services are subprocesses inside Docker containers



- **Minimal** configuration which can be changed at runtime
- RAM, swap and CPU are capped (cgroups + cfs)
- Containers run **inside VMs** (CentOS 7): VM layer required by HLT experts
- Jobs are run in a **single-shot** mode → container dies when done, allows Plancton to launch a new one
- ALICE Grid middleware unmodified → using AliEn-WorkQueue

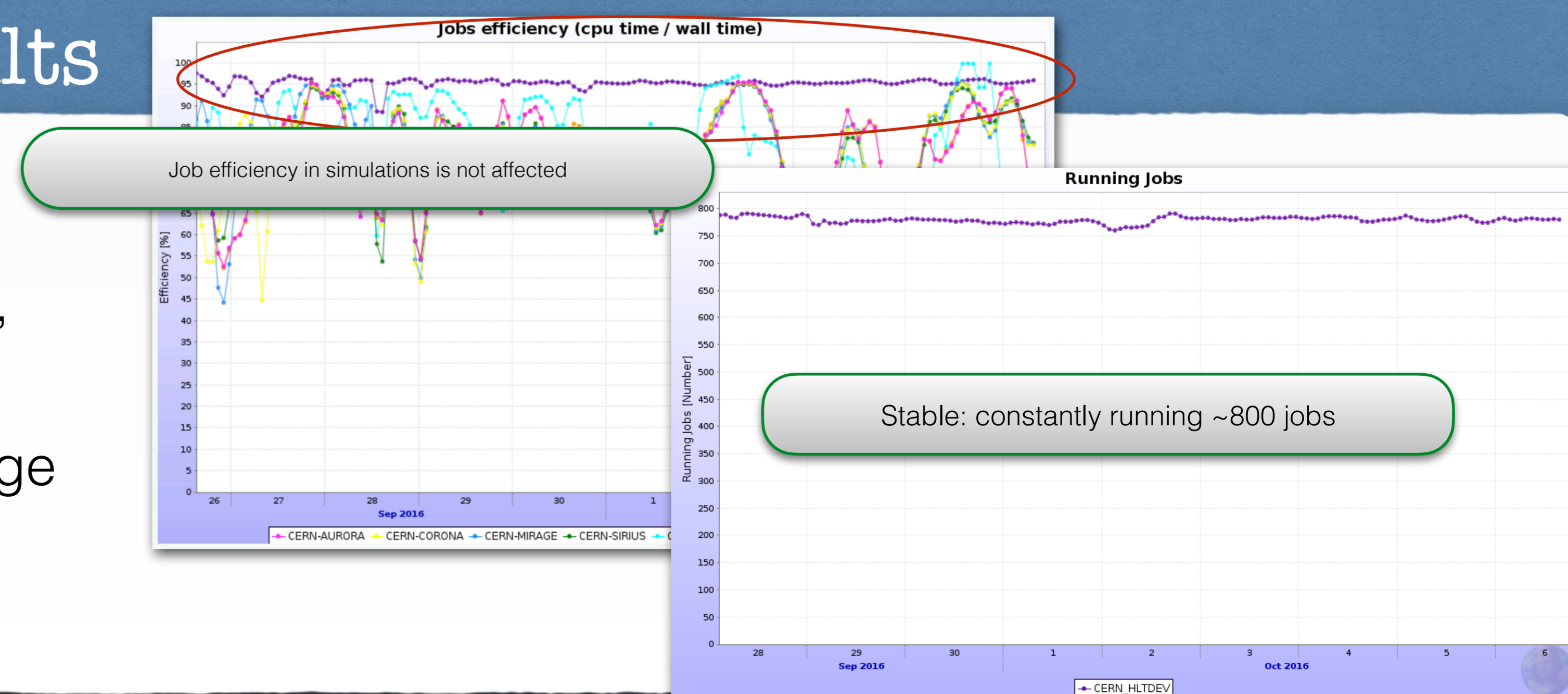


- Volunteer computing: only Docker and Plancton required
- Jobs **running on bare metal**
- Opportunistic resource utilisation (configurable) → quickly given back to user when reclaimed
- Dedicated HTCondor submission node on a static resource

Results

Service is **working perfectly**

- A lightweight scheduler for schedulers: completely independent, only takes care of container deployment
- Suitable for disposable tasks: input and output on external storage
- Plancton can be updated/restarted without affecting current running containers



[1] Plancton: github.com/mconcas/plancton
[2] Docker: docker.com
[3] Parrot (CCTools): ccl.cse.nd.edu/software/parrot

[4] HTCondor: research.cs.wisc.edu/htcondor
[5] CVMFS: cernvm.cern.ch/portal/filesystem
[6] Work Queue (CCTools): ccl.cse.nd.edu/software/manuals/workqueue.html

[7] AliEn: alien.web.cern.ch
[8] AliEn-wq: github.com/alice/alien-workqueue