ComputeOps: container for High Performance Computing

C. Cavet and the ComputeOps project members

ccavet@apc.in2p3.fr

Abstract

The High Performance Computing (HPC) domain aims to optimize code in order to use the last multicore and parallel technologies including specific processor instructions. In this computing framework, portability and reproducibility are key concepts. A way to handle these requirements is to use Linux containers. These "light virtual machines" allow to encapsulate applications within its environment in Linux processes. Containers has been recently rediscovered due to their abilities to provide both multi-infrastructure environnement for developers and system administrators and reproducibility due to image building file. Two container solutions are emerging: **Docker** for micro-services and **Singularity** for computing applications. We present here the **ComputeOps** project which has the goal to study the benefit of containers for HPC applications.

HPC

► **Keywords**:

- parallelization methods.
- specific libraries.
- portability and reproducibility.

@Li Xiang—Xinhua News Agency/Getty Images.

Goal

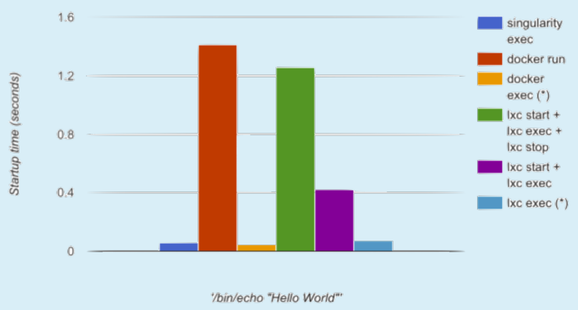
The **ComputeOps** project [2] has the goal to study and provide:

- container technology interoperability.
- portability on various architectures and infrastructures.
- repeatability of computing.
- good practices for writing recipes.

Solution comparison

Several container solutions are under evaluation:

- Singularity.
- Shifter.
- uDocker.
- CharlieCloud.



Comparaison of performance [1]: elapsed time (in s) for running hello world.

Singularity

► **Advantages**:

- Compatibility: Docker image.
- Security: unprivileged mode.
- I/O: transparent => compatibility with MPI processes and X11.
- Scheduler: native integration.
- GPU: easy GPU integration in containers.

French partners

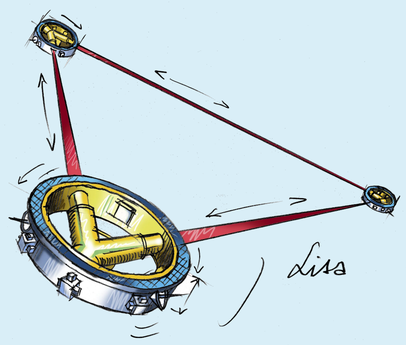
► **ComputeOps partners**:

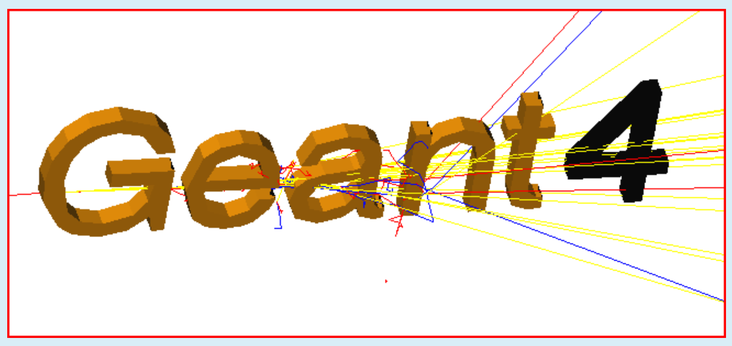
- **Aristote**: GENCI, IDRIS (french supercomputers).
- **Ecole Centrale Nantes**.
- **P2IO ACP**:
  - GPGPU: two cards of 1GB.
  - CUDA 9.2 and 9.0, last Nvidia driver.
  - TensorFlow library.

P2IO
Physique des 2 Infinis et des Origines

Pilot applications

- **GALOP**: ray tracing.
- **MEM**: analyze application running on multi-GPUs to study Higgs boson properties.
- **TensorFlow**: machine learning applications using TensorFlow.
- **LDC**: simulation pipeline for scientific challenges of the LISA mission.
- **Geant4** simulation for direct Dark-Matter detection experiments (Xenon & DarkSide).



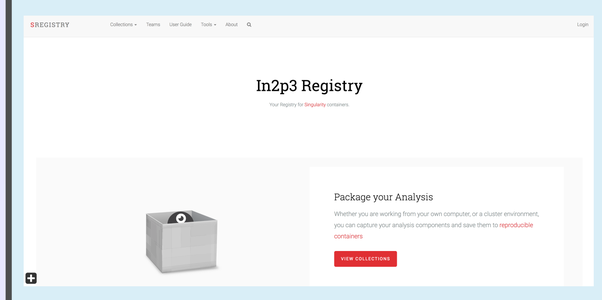


Singularity Hub

► **ComputeOps SHub** [3]:

A Singularity private Hub for Research allowing to manage Singularity images.

- a marketplace for Singularity images.
- an authentification based on a GitLab/GitHub account.
- collections of images for specific projects.
- with manageable workflows.



► **ComputeOps collection:**

- experimentation of a reviewer team for image validation.
- label processing and maintained containers.

Tutorials

► **Containerized tutorials for users:**

- Deep dive in container technologies: MaitresNageurs/EnBarque [5].
- Container in production: IN2P3 IT school 2018 [4].

Multi-ressources scheduler

A multi-ressources scheduler based on XtremWeb-HEP [6] allowing to launch jobs both on virtual machines and bare metal containers.

EDGeS

CI with Singularity

- Commit on GitLab-CI: run the Docker image creation and push it on the registry.
- Conversion or image re-build with infrastructure specification (bind volumes) on a virtual machine
- Singularity in Docker: a way to automatically build Singularity images on various OS and publish/update them on hubs.
- Run in production on a HPC cluster within the scheduler.

How to contribute?

► **Open to suggestions /contributions/collaborations:**

- More information on the wiki of the project [2].
- Send an email to `computeops-l@in2p3.fr`.

Container life cycle

Interactive Development

`sudo singularity build --sandbox tmpdir/ Singularity`

`sudo singularity build --writable container.img Singularity`

BUILD ENVIRONMENT

Build from Recipe

`sudo singularity build container.img Singularity`

Build from Singularity

`sudo singularity build container.img shub://vsoch/hello-world`

Build from Docker

`sudo singularity build container.img docker://ubuntu`

Container Execution

```
singularity run container.img
singularity shell container.img
singularity exec container.img ...
```

Reproducible Sharing

```
singularity pull shub://...
singularity pull docker://...  *
```

PRODUCTION ENVIRONMENT

* Docker construction from layers not guaranteed to replicate between pulls

References

[1] Arango et al., arXiv 1709.10140, 2017

[2] ComputeOps wiki: `https://gitlab.in2p3.fr/CodeursIntensifs/DecaLog/wikis/ComputeOps`

[3] ComputeOps Shub: `https://sregistry.in2p3.fr`

[4] IN2P3 IT school: `https://indico.in2p3.fr/event/17124/timetable/#20180607`

[5] Piscine: `https://gitlab.in2p3.fr/MaitresNageurs/EnBarque`

[6] XtremWeb: `https://xtremweb-hep.lal.in2p3.fr/`

[7] Singularity: `http://singularity.lbl.gov/`

The project is founded by the CNRS IN2P3 (french national Institut for Nuclear Physic and Particule Physic). ComputeOps is part of the DecaLog Master Project.