Introdution to Singularity containers

Jan P. Buchmann
jan.buchmann@sydney.edu.au

The University Of Sydney

2018-11-29

Just in case...

Download	https://www.sylabs.io/singularity/get-singularity/
Manual	https://www.sylabs.io/guides/3.0/user-guide.pdf
Paper	https://www.sylabs.io/guides/3.0/user-guide.pdf

Singularity: Containers for HPC

Containers are encapsulated system enviroments

Not a microservice: Scientific focus, e.g. whole pipelines

Single file: The image is a single file easily share, archive, reproduce, good for parallel file sytems, e.g. Lustre

Run as user: root to create, user to run

Access HPC resources: MPI, GPUs, InfiniBand/Network, file systems

Biggest difference to Docker

Privileges

You run the container as the user who invokes singularity. You can only be root in the container if you run it as root. Not your usual HPC experience.

No daemon

There is no deamon required, Singularity image is mounted as a loopback. Docker swarms need a DockerEngine on each node or instance they run.

Runs closer to the host

Running a singularity container bind mount your $HOME, /dev, /sys, and /proc automatically by default.

Singularity Image Format (SIF) ($\geq$3.0)

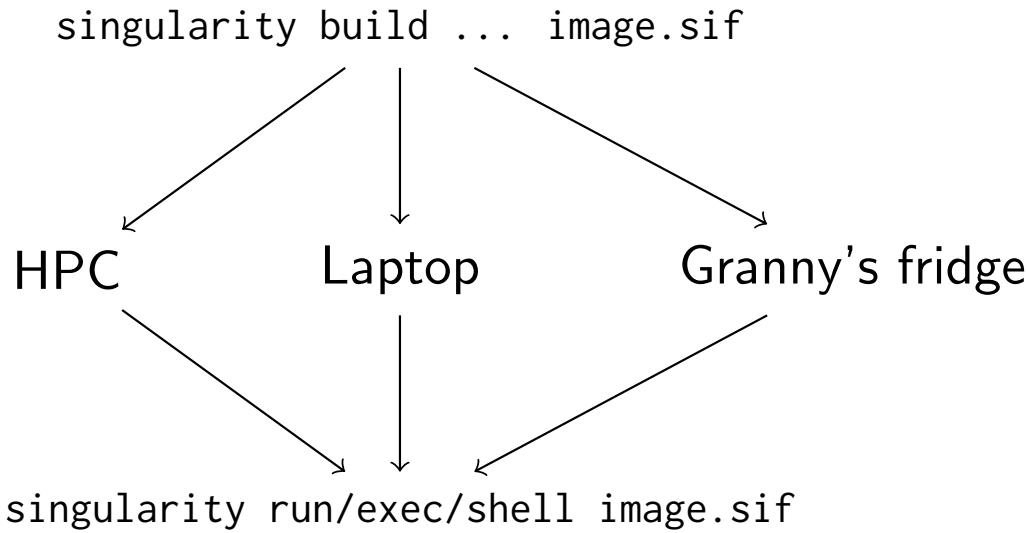
Image container format resembling a general file system whoch will allow PGP signing, block encryption, partitions accomodating multiple OSes, fast metadata access.

Docker layers change

Docker image layers can change, i.e. a docker image has likely changed layers when pulled a couple of months later.

Speed

Overall singularity workflow



Building singularity containers

Docker Hub (docker://) singularity build lolcow.simg
docker://godlovedc/lolcow

Container Library (library://) singularity build lolcow.simg
library://sylabs-jms/testing/lolcow

Singularity Hub (shub://) singularity build demo.simg
shub://jasongallant/singularity_demosingularity

Singularity receipe files Roll your own

Invoking singularity

```
singularity [global options] command [command options]
singularity -v build --sandbox /tmp/ubuntu docker://ubuntu:latest
```

Workflow: Build enviroment (root)

Interactive

```
sudo singularity build --sandbox /tmp/ubuntu    \
  docker://ubuntu:latest  # In container
sudo singularity exec --writable  /tmp/ubuntu/  \
  apt-get update          # In container
sudo singularity exec --writable  /tmp/ubuntu/  \
  mkdir  /mnt/builds      # In container
sudo singularity shell -B hostdir:/mnt/builds -w /tmp/ubuntu/
#-> compile, copy, etc. tools from hostdir in the container
sudo singularity build raxml.sif /tmp/ubuntu/
sudo singularity shell raxml.sif
```

Example: `../commands/commands.md`

Workflow: Receipe file

- Preferred way to crate images. Use interactive to get details right, test depencdencies, etc. than fixeverything in a receipe file.

Example: `../receipes/raxml.ubuntu.def`

Singularity on HPC

- Host and container OS need same OpenMPI/MPI version
- Adjust image for binding directories
- Adjust GPU specific files, libraries

Example PBS file: `../receipes/raxml-sing.pbs`