Creating and Running Software Containers with Singularity

Afif Elghraoui
NIH HPC
staff@hpc.nih.gov

NIH

BIOWULF
HIGH PERFORMANCE COMPUTING AT THE NIH

What are software containers?

A container is an instance of lightweight virtualization. Unlike virtual machines, containers share the operating system kernel with the host.

Use Cases

- mobility and preservation of software environments, especially for scientific workflows
- installation/distribution of poorly-integrated software

Docker

Docker is currently the most widely used container software, but it's not suitable for HPC:

Docker daemon attack surface

Running containers (and applications) with Docker implies running the Docker daemon. This daemon currently requires `root` privileges, and you should therefore be aware of some important details.

First of all, **only trusted users should be allowed to control your Docker daemon**. This is a direct consequence of some powerful Docker features. Specifically, Docker allows you to share a directory between the Docker host and a guest container; and it allows you to do so without limiting the access rights of the container. This means that you can start a container where the `/host` directory is the `/` directory on your host; and the container can alter your host filesystem without any restriction. This is similar to

https://docs.docker.com/engine/security/security/#docker-daemon-attack-surface

NIH

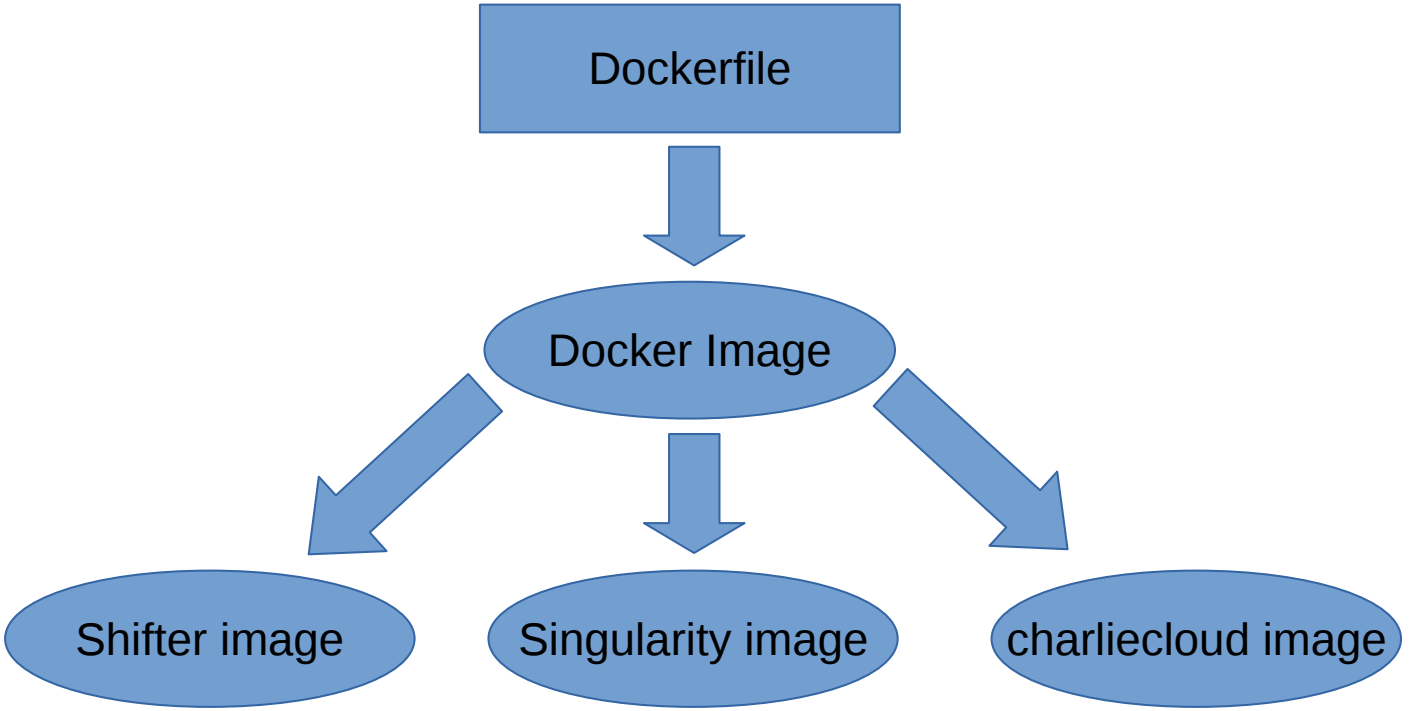
BIOWULF
HIGH PERFORMANCE COMPUTING AT THE NIH

Singularity

A user inside a Singularity container is the same user as outside the container.

...so operations that require administrative privileges inside the container require you to have administrative privileges on the host

Starting From Docker



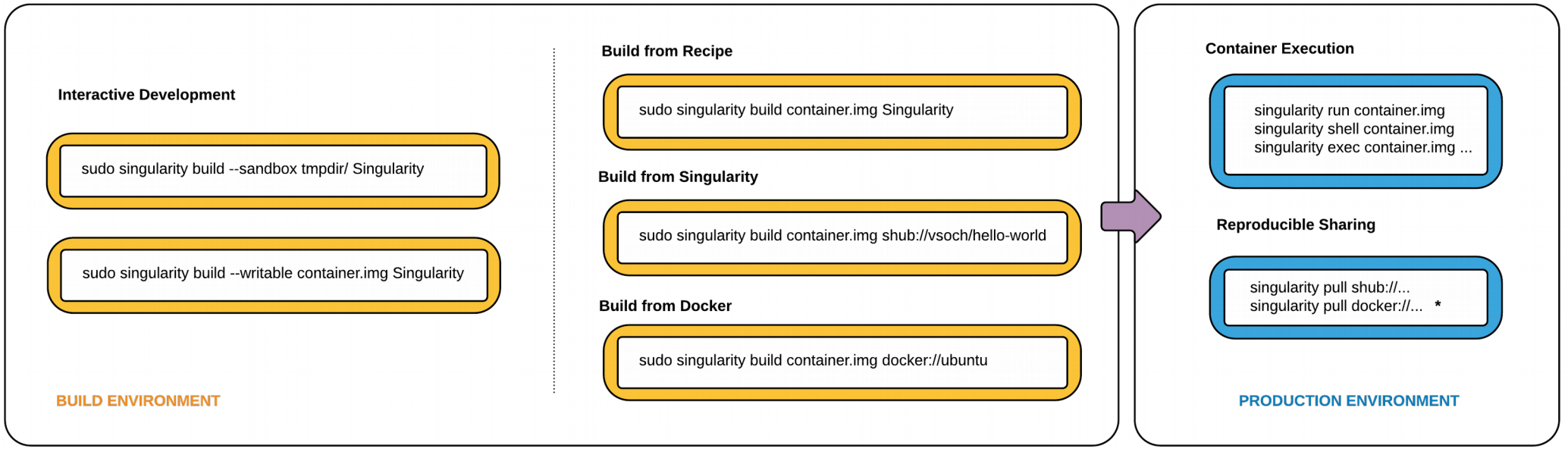
Singularity and Docker

Singularity can retrieve and convert Docker containers directly from Docker Hub, most of the time without any problems, but the Docker container should

- define all environment variables using the ENV instruction
- define an ENTRYPOINT instruction pointing to the command line interface to your pipeline and don’t use the CMD instruction
- update the shared library cache by calling ldconfig after installing shared libraries
- not use the USER instruction
- not rely on having elevated user permissions to run
- not install anything to /root, $HOME, or /tmp

http://singularity.lbl.gov/docs-docker#best-practices
https://github.com/singularityware/docker2singularity#tips-for-making-docker-images-compatible-with-singularity

Singularity Workflow



* Docker construction from layers not guaranteed to replicate between pulls

*image from http://singularity.lbl.gov/

NIH

BIOWULF
HIGH PERFORMANCE COMPUTING AT THE NIH