

Managing Research Workflows with Singularity Containers: Part 1

Andrew Monaghan

andrew.monaghan@Colorado.edu

<https://www.rc.colorado.edu>

Slides available for download at

https://github.com/rctraining/Singularity_Short_Course_Spring_2018

*Adapted from a presentation by Martin Cuma (U. Utah; [1](#)) and materials from
<https://singularity.lbl.gov/> and <https://www.singularity-hub.org>*

Outline

- Introduction to Containers
- Singularity Commands and options
- Hands-on Examples
- Summary

Introduction to Containers



What is a container?

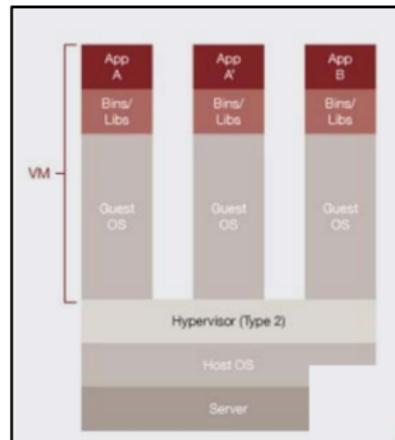
A container is a portable environment that packages some or all of the following: an operating system, software, libraries, compilers, data and workflows. Containers enable:

- Mobility of Compute
- Reproducibility (software and data)
- User Freedom

Virtualization (1)

Hardware virtualization (not used by containers!)

- Can run many OS's on same hardware (machine)
- E.g., VirtualBox, VMWare



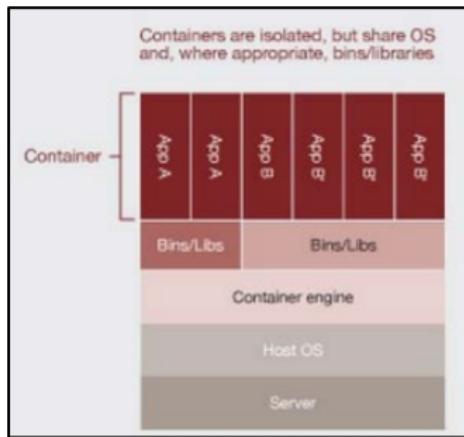
Material courtesy: M. Cuma, U. Utah

Virtualization (2)

OS-level virtualization (used by containers!)

- Can run many isolated OS instances (guests) under a server OS (host)
- Also called containers
- E.g., Docker, Singularity

Best of both worlds: isolated environment that user wants, but can leverage host OS resources (network, I/O partitions, etc.)



Material courtesy: M. Cuma, U. Utah

Why Singularity?

- Singularity is a comparably safe container solution for HPC
 - User is same inside/outside container
 - User has cannot escalate permissions without administrative privilege
- Can support MPI and GPU resources on HPC (scaling)
- Can use HPC filesystems
- Supports the use of Docker containers
- Container is seen as a file, and can be operated on as a file

Singularity commands and options

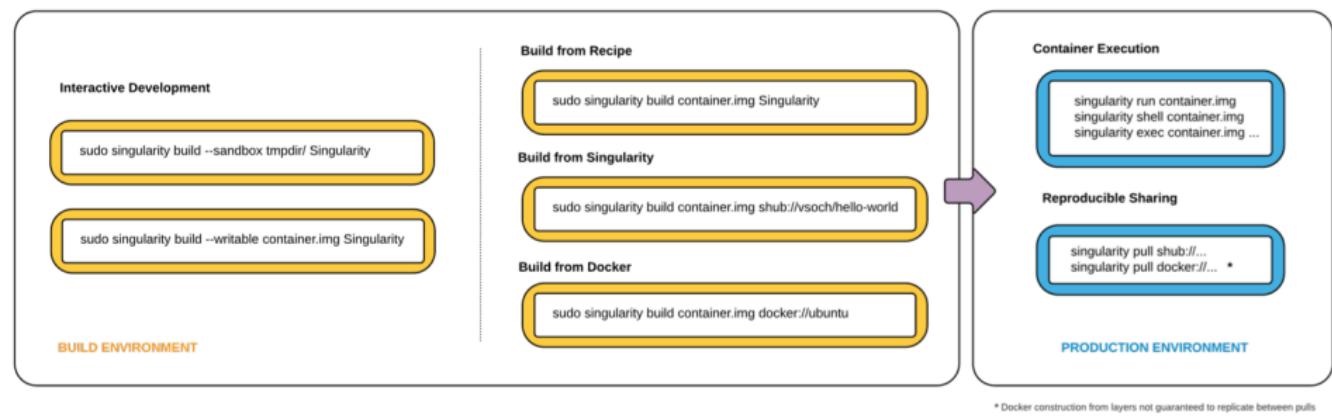
Interactive development

Building

Running



The Singularity workflow



<https://singularity.lbl.gov>

Interactive development

Within your build environment (a laptop, workstation, or a server that you control): develop and test containers

- using --sandbox (build into a writable *directory*)
 - `sudo singularity build --sandbox tmpdir/ mycont.def`
- or using --writable (build into a writable *ext3* image)
 - `sudo singularity build --writable tmp.img mycont.def`

Building containers

After development, within your build environment (a laptop, workstation, or a server that you control): build your production containers with a *squashfs* filesystem...

- Build from a “recipe” that you developed.
 - `sudo singularity build mycont.img mycont.def`
- Build from an existing container on Singularity Hub
 - `sudo singularity build mycont.img shub://somewhere`
- Build from an existing container on Docker Hub
 - `sudo singularity build mycont.img docker://somewhere`

Container Formats

- **squashfs**: the default container format is a compressed read-only file system that is widely used for things like live CDs/USBs and cell phone OS's
- **ext3**: (also called `writable`) a writable image file containing an ext3 file system that was the default container format prior to Singularity version 2.4
- **directory**: (also called `sandbox`) standard Unix directory containing a root container image
- **tar.gz**: zlib compressed tar archive
- **tar.bz2**: bzip2 compressed tar archive
- **tar**: uncompressed tar archive

<https://singularity.lbl.gov>

Running containers

Now, on **any system**, even without administrative privilege, you can retrieve and use containers:

- Download a container from Singularity Hub or Docker Hub
 - `singularity pull shub://some_image`
 - `singularity pull docker://some_image`
- Run a container
 - `singularity run mycont.img`
- Execute a specific program within a container
 - `singularity exec mycont.img pythonmyscript.py`
- “Shell” into a container to use or look around
 - `Singularity shell mycont.img`
- Inspect an image
 - `Singularity inspect --runscript mycont.img`

Singularity Commands

build: Build a container on your user endpoint or build environment

exec: Execute a command to your container

inspect: See labels, run and test scripts, and environment variables

pull: pull an image from Docker or Singularity Hub

run: Run your image as an executable

shell: Shell into your image

<https://singularity.lbl.gov>

Hands-on Examples

Building a container from
Singularity Hub

Building a container from a
recipe

Building a container from
Docker Hub

Building a container *on*
Singularity Hub



Caveats

For the sake of expediency, we will build containers based on Ubuntu Linux. Most other variants/versions of Linux are available.

We will do everything on our ‘root’ system in this session; i.e., our laptop or desktop computer. In Part 2 (next week) we will run on RMACC Summit, a system on which we do not have root access.

We don’t cover a number of topics. E.g.,

- “instances” (running services from containers in the background)
- Other containerization software that can work on HPC
 - Shifter, Charliecloud (neither presently installed on Summit)

<https://singularity.lbl.gov>



Research Computing
UNIVERSITY OF COLORADO BOULDER

16

4/17/18 Singularity Part 1

Be Boulder.

Start your Linux VM so that you can run Singularity

Once you've installed Singularity on your machine (assuming Mac or Windows):

```
$ mkdir singularity-vm  
$ cd singularity-vm  
$ vagrant init singularityware/singularity-2.4  
$ vagrant up  
$ vagrant ssh
```

If you are on Linux, you don't need to start a VM

Ex1: Build a container from Singularity Hub

...let's build a container from existing image

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

...let's run the default program in the image

```
$ singularity run hello.img
```

...let's look at the recipe

```
$ singularity inspect -d hello.img
```

...now let's shell in and modify the image

```
$ singularity exec hello.img /bin/bash /rawr.sh
```

...now let's modify the image

```
$ sudo singularity shell --writable hello.img
```

... now let's run our new program

```
$ singularity exec hello.img /bin/bash hello.sh
```

Ex2: Building a container from a recipe (1)

Header

```
Bootstrap:docker
From:ubuntu:latest
```

Metadata

```
%labels
MAINTAINER Andy M
```

Runtime
environment
variables

```
%environment
HELLO_BASE=/code
export HELLO_BASE
```

Default program
at runtime

```
%runscript
echo "This is run when you run the image!"
exec /bin/bash /code/hello.sh "$@"
```

Where software
and directories
are installed at
buildtime

```
%post
echo "This section is performed after you bootstrap to build the image."
mkdir -p /code
apt-get install -y vim
echo "echo Hello World" >> /code/hello.sh
chmod u+x /code/hello.sh
```

Ex2: Building a container from a recipe (2)

See <http://singularity.lbl.gov/docs-recipes> for more.

Now let's build and run a container from our recipe.

```
$ git clone  
https://www.github.com/monaghaa/singularity git tutorial  
  
$ cd singularity_git_tutorial  
  
$ sudo singularity build --writable mycont.img Singularity  
  
$ sudo singularity run mycont.img
```

Ex3: Build a container from Docker Hub

...let's build a container from existing image

```
$ sudo singularity build --writable python.img  
docker://python:latest
```

...let's run the default program in the image

```
$ singularity exec python.img python
```

...let's add the numpy package to the container

```
$ sudo singularity shell -w python.img
```

Ex4: Build a container on Singularity Hub (basic steps)

1. Create a github repository for your container or class of containers.
2. Create a recipe file for whatever you want your container to be
3. Name it “Singularity”
4. Upload it to your github repository
5. Log into Singularity Hub using your github username/password
6. Choose “Add Collection”
7. Select the github repository
8. The container will build automatically.
9. Additional details at:

<https://github.com/singularityhub/singularityhub.github.io/wiki>

Thank you!

Please fill out the survey:

<http://tinyurl.com/curc-survey16>

My contact information:

Andrew.Monaghan@Colorado.edu

Additional learning resources:

Slides and Examples from this course:

<https://github.com/rctraining/Singularity Short Course Spring 2018>

Web resources:

<https://singularity.lbl.gov/user-guide> (*user guide for Singularity*)

<https://www.singularity-hub.org/> (*Singularity Hub*)