Things to do while you are waiting

- Course slides are available at: https://hprc.tamu.edu/training/intro_containers.html
- Log into TAMU VPN (if you're off campus)
- Get ready to launch a terminal on the Grace cluster for interactive exercises (ask if you don't know how).

Introduction to Containers

featuring Singularity on the Grace cluster

by Richard Lawrence

Date: 03/11/2022

Spring 2022

Outline

- Overview of Containers
- Overview of Singularity
- Getting a Container Image
- Container Usage Basics

Course Objectives

The researcher should be able to:

- Decide whether containers are right for you
- Find container images in repositories
- Use Singularity at HPRC for basic container tasks

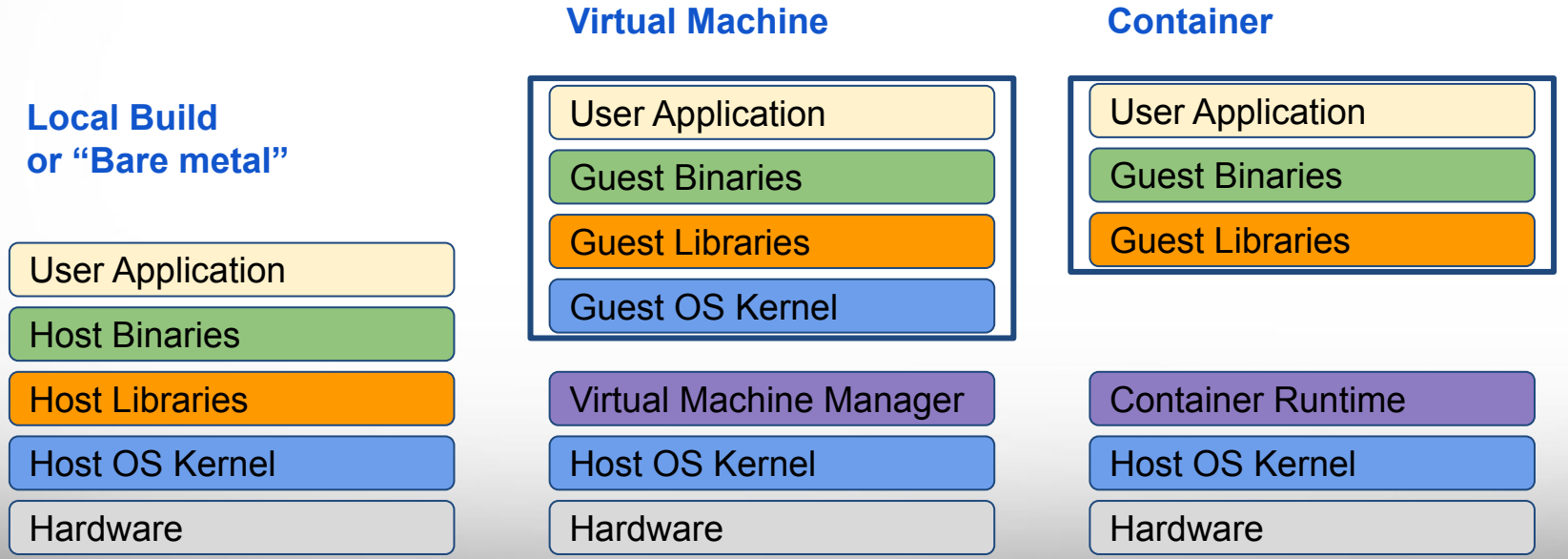
Learning Resources

- HPRC Wiki https://hprc.tamu.edu/wiki/SW:Singularity
- HPRC on Youtube https://www.youtube.com/c/TexasAMHPRC
  (video of this course will be posted)
- Singularity Manual https://apptainer.org/user-docs/3.8/
- Docker Manual https://docs.docker.com/
- Other container courses:
  NBIS https://nbis-reproducible-research.readthedocs.io/en/latest/singularity/
  Arizona https://learning.cyverse.org/projects/Container-camp-2020/
  TACC https://learn.tacc.utexas.edu/mod/page/view.php?id=95

Overview of Containers

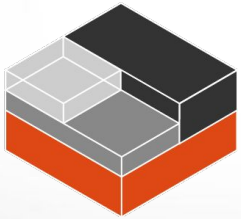
Introduction to Containers

- Containers make Applications more portable.
- Unlike in VMs, the OS Kernel is not duplicated.

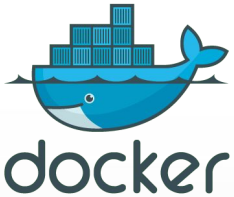


Popular Containers

Instant deployment to users on different devices!



LXC
2008



Docker
2013

Singularity
2015

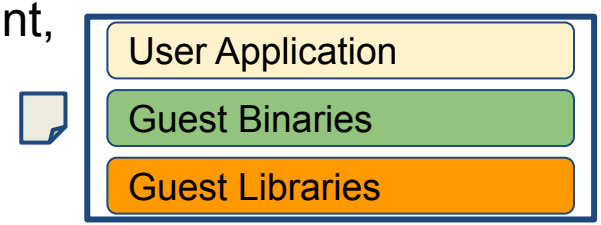
Charliecloud
2017

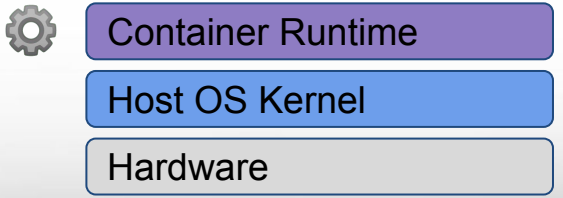
Podman
2018

Basics

Containers come in two parts:

- **Image**:
  - A file containing all the parts of an environment, libraries and applications
  - Generally built by experts
  - Found in online repositories
- **Runtime**:
  - Compatibility layer translates between the container environment and the host operating system
  - Runtime is installed by cluster administrators





Why use Containers?

- **Shareability**:
  - Share your container image file by uploading to a public repository
  - Use images shared by others
- **Portability**:
  - Use images on any computer with the same architecture (x84-64)
- **Reproducibility**:
  - Container users are largely unaffected by changes to the cluster environments

Overview of Singularity

Singularity is now Apptainer (Nov 2021)

But we will continue to refer to it as Singularity for now, because Grace has Singularity 3.8.5 installed.

Singularity Features

- Singularity is a container runtime
- Singularity can read and convert Docker images
- Filesystem inside container is isolated
- User inside is the same as the user outside
- Singularity containers are suitable for use on clusters
- Runs "close to the hardware" for speed
- Works with high-performance cluster technologies

See https://apptainer.org/user-docs/3.8/

Singularity and Security

Singularity addresses security concerns about Docker.

- **Privileges**: Singularity grants the user no additional privileges or permissions, so you can't harm the cluster by using singularity, nor can other users harm you.
- **Independence**: Singularity does not require root permission to run, so you don't need to ask your administrators for help installing anything.

Singularity at HPRC - Best Practices

- Singularity activities are cpu-intensive. You must use a compute node for singularity activities. Cannot run on a login node.
- Singularity image files (extension `.sif`) are flat - they don't share any data with other image files.
- Image files are large on disk and should be put on `/scratch` (not `/home`). File transfer takes time.

Exercises coming up next

Log into Grace ssh or Portal

Getting a Container Image

With exercises

Popular Repositories

The most common repository is,

- Docker Hub

Others repositories include,

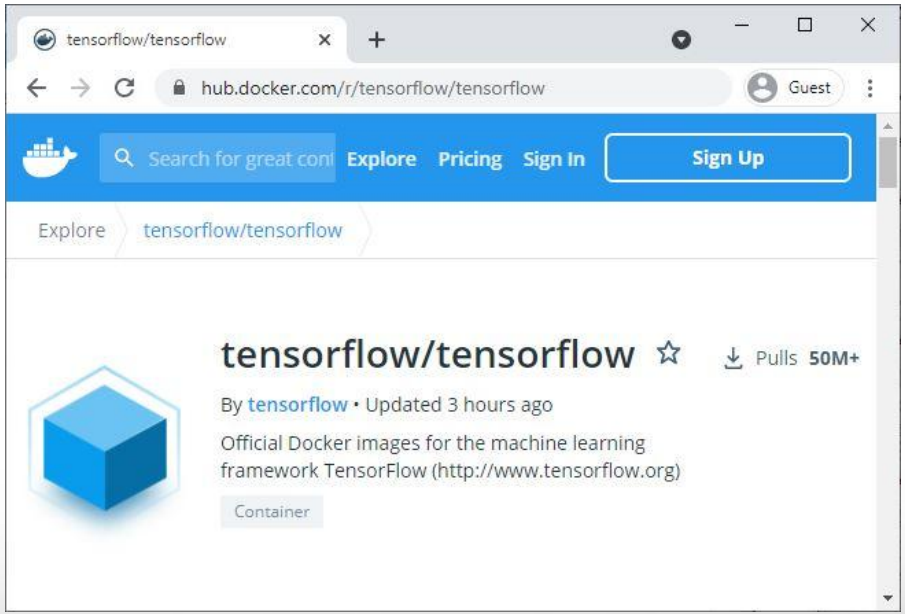
- Singularity Hub
- Singularity Library
- NVIDIA GPU Cloud
- Quay.io
- BioContainers

See https://hprc.tamu.edu/wiki/SW:Singularity:Examples

Docker Hub Example

Docker Hub repositories are named in the form `<group>/<name>` similar to GitHub.

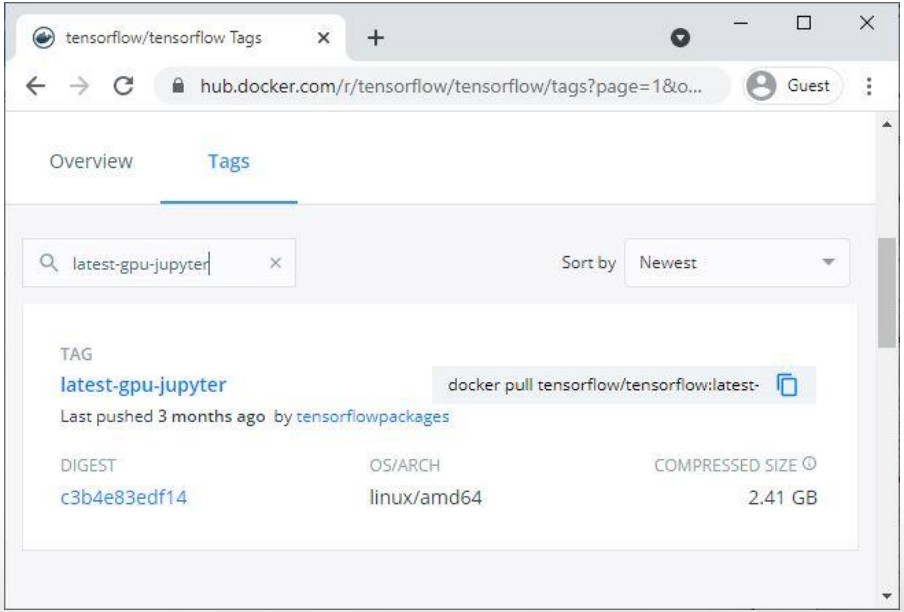
- If the group is "_", then you omit that part.



Docker Hub Example

Each image within a repository is named with a `tag` that describes how it was built.

Some repositories still work if you omit the tag, but it's best to include it if you can.



Singularity Pull

Singularity can fetch images from repositories and also convert them to the singularity file format at the same time.

```
singularity pull [target-filename] <source>
```

Where `<source>` refers to something on the internet. The syntax depends on where the source is located.

and `[target-filename]` includes the file extension.

Singularity Pull Example

The `<source>` argument for Docker images looks like
`docker://<group>/<name>[:<tag>]`

Therefore the pull command for the previous example is,

```
singularity pull tensorflow.sif \
docker://tensorflow/tensorflow:latest-gpu-jupyter
```

Singularity Pull on Terra

To get an interactive job on a compute node, use "`srun`":

```
srun --mem=512m --time=01:00:00 --pty bash -i
```

To tell Singularity to use `/scratch` instead of `/home`:

```
export SINGULARITY_CACHEDIR=$SCRATCH/.singularity
```

To get access to the internet on a compute node:

```
module load WebProxy
```

Getting an Image Exercise

The _/hello-world repository from Docker Hub is small enough to make a nice, quick exercise

```
[username@login]$ srun --nodes=1 --ntasks-per-node=4 --mem=2560M \
--time=01:00:00 --pty bash -i
```
(wait for job to start)
```
[username@compute]$ cd $SCRATCH
[username@compute]$ export SINGULARITY_CACHEDIR=$SCRATCH/.singularity
[username@compute]$ module load WebProxy
[username@compute]$ singularity pull hello-world.sif \
docker://hello-world
```
(wait for download and convert)
```
[username@compute]$ exit
```

Singularity Pull Batch Example

```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=sing_pull           #Set the job name to "sing_pull"
#SBATCH --time=01:00:00               #Set the wall clock limit to 1hr
#SBATCH --ntasks=4                    #Request 4 task
#SBATCH --mem=2560M                   #Request 2560MB (2.5GB) per node
#SBATCH --output=sing_pull.%j         #Send stdout/err to "sing_pull.[jobID]"

##OPTIONAL JOB SPECIFICATIONS
##SBATCH --account=123456             #Set billing account to 123456
##SBATCH --mail-type=ALL              #Send email on all job events
##SBATCH --mail-user=email_address    #Send all emails to email_address
```

```
# set up environment for download
cd $SCRATCH
export SINGULARITY_CACHEDIR=$SCRATCH/.singularity
module load WebProxy

# execute download
singularity pull hello-world.sif docker://hello-world
```

PULL

Pre-built Images at HPRC

HPRC provides a few images for public use, located at

`/scratch/data/Singularity/images/`

Image `Fedora28-HPRCLAB-40GB.img` contains a useable workstation. (The `.img` file extension is from an older version of Singularity.)

https://hprc.tamu.edu/wiki/SW:Singularity:Examples#Prebuilt_images

Container Usage Basics

With exercises

Interacting with the Container

A container is used to control your environment for doing computation tasks. Although the variables and files in the container may be different, the user is always the same.

Three methods:

- **Interactive**: singularity `shell`
- **Batch processing**: singularity `exec`
- **Container-as-executable**: singularity `run`

Singularity Run Exercise

Singularity run will execute the default runscript, if one was defined. You may also execute the container directly.

```
[username@login]$ srun --mem=512m --time=01:00:00 --pty bash -i
[username@compute]$ singularity run hello-world.sif
Hello from Docker!
[username@compute]$ ./hello-world.sif
Hello from Docker!
```

Docker hello-world is a minimal image. This is all it can do.

Singularity Shell Exercise

Singularity shell gives you a terminal inside the container, if the image has a working shell installed in it.
(This one is at `/scratch/data/Singularity/images/`)

```
[username@login]$ srun --mem=512m --time=01:00:00 --pty bash -i
[username@compute]$ singularity shell Fedora28-HPRCLAB-40GB.img
Singularity> whoami
username
Singularity> head -n1 /etc/os-release
NAME=Fedora
Singularity> exit
[username@compute]$ head -n1 /etc/os-release
NAME="CentOS Linux"
```

Singularity Exec Exercise

Singularity Exec lets you run executables in a container.
This is appropriate for batch jobs.
(This one is at `/scratch/data/Singularity/images/`)

```
[username@login]$ srun --mem=512m --time=01:00:00 --pty bash -i
[...]$ singularity exec Fedora28-HPRCLAB-40GB.img python3 --version
Python 3.6.6
[...]$ singularity exec Fedora28-HPRCLAB-40GB.img python3 -c \
'print("hello from python")'
hello from python
```

Working with Files

- Filesystem inside a container is isolated from the real, physical filesystem.
- To access your files, ensure the directory is *mounted*.
- By default, Singularity will mount `$HOME` and `$PWD` if it can.
- To specify additional directories, use the `SINGULARITY_BINDPATH` environment variable or the `--bind` command line option.

Working with Files Exercise

Recommended that you mount `/scratch` to get access to your data storage, and `$TMPDIR` to get access to the local disk on the node.

```
[username@login]$ srun --mem=512m --time=01:00:00 --pty bash -i
[...]$ singularity shell --bind "/scratch,$TMPDIR" <image>
Singularity> cd $SCRATCH; touch outfile; exit
[...]$ ls $SCRATCH
outfile
```

Notice that your variables like `$SCRATCH` get passed into the container by default, but the container can override them.

Singularity Batch Example

```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=sing_test          #Set the job name to "sing_test"
#SBATCH --time=00:10:00               #Set the wall clock limit to 1hr and 30min
#SBATCH --ntasks=4                    #Request 4 task
#SBATCH --mem=2560M                   #Request 2560MB (2.5GB) per node
#SBATCH --output=sing_test.%j         #Send stdout/err to "sing_test.[jobID]"

##OPTIONAL JOB SPECIFICATIONS
##SBATCH --account=123456              #Set billing account to 123456
##SBATCH --mail-type=ALL               #Send email on all job events
##SBATCH --mail-user=email_address     #Send all emails to email_address

export SINGULARITY_BINDPATH="/scratch,$TMPDIR"

# execute the default runscript defined in the container
singularity run centos6_bootstrapped.img

# execute a command within container
# the command should include absolute path if the command is not in the default search path
singularity exec centos6_bootstrapped.img /scratch/user/netid/runme.sh
```

RUN

Basic Content Complete

Optional: see “Advanced” slides

Conclusion

- Run Containers on clusters! It's easy.
- HPRC supports Singularity
- Convert Docker to Singularity!
- Expect Charlie Cloud support in the near future
- Ask for help!

Survey

Please fill out the survey to let us know how you feel about this short course. This will help us improve.

Questions

Learning Resources

- HPRC Wiki https://hprc.tamu.edu/wiki/SW:Singularity
- HPRC on Youtube https://www.youtube.com/c/TexasAMHPRC
  (video of this course will be posted)
- Singularity Manual https://apptainer.org/user-docs/3.8/
- Docker Manual https://docs.docker.com/
- Other container courses:
  NBIS https://nbis-reproducible-research.readthedocs.io/en/latest/singularity/
  Arizona https://learning.cyverse.org/projects/Container-camp-2020/
  TACC https://learn.tacc.utexas.edu/mod/page/view.php?id=95

Thank you

Contact: help@hprc.tamu.edu