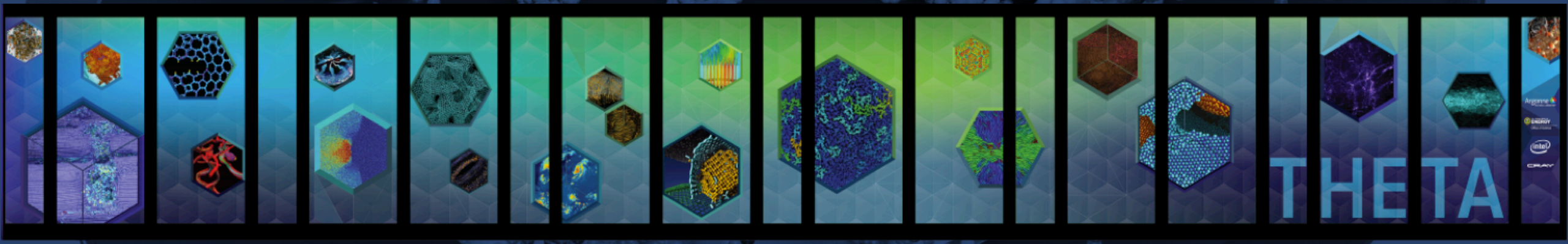
Argonne NATIONAL LABORATORY

Using Containers on Theta

Murat Keçeli

SIMULATION.DATA.LEARNING WORKSHOP '19



Quick Survey

- **Have you ever used containers?**

Quick Survey

- **Have you ever used containers?**

- **Try Singularity on Theta:**

```
/projects/SDL_Workshop/training/Singularity/run.sh
```

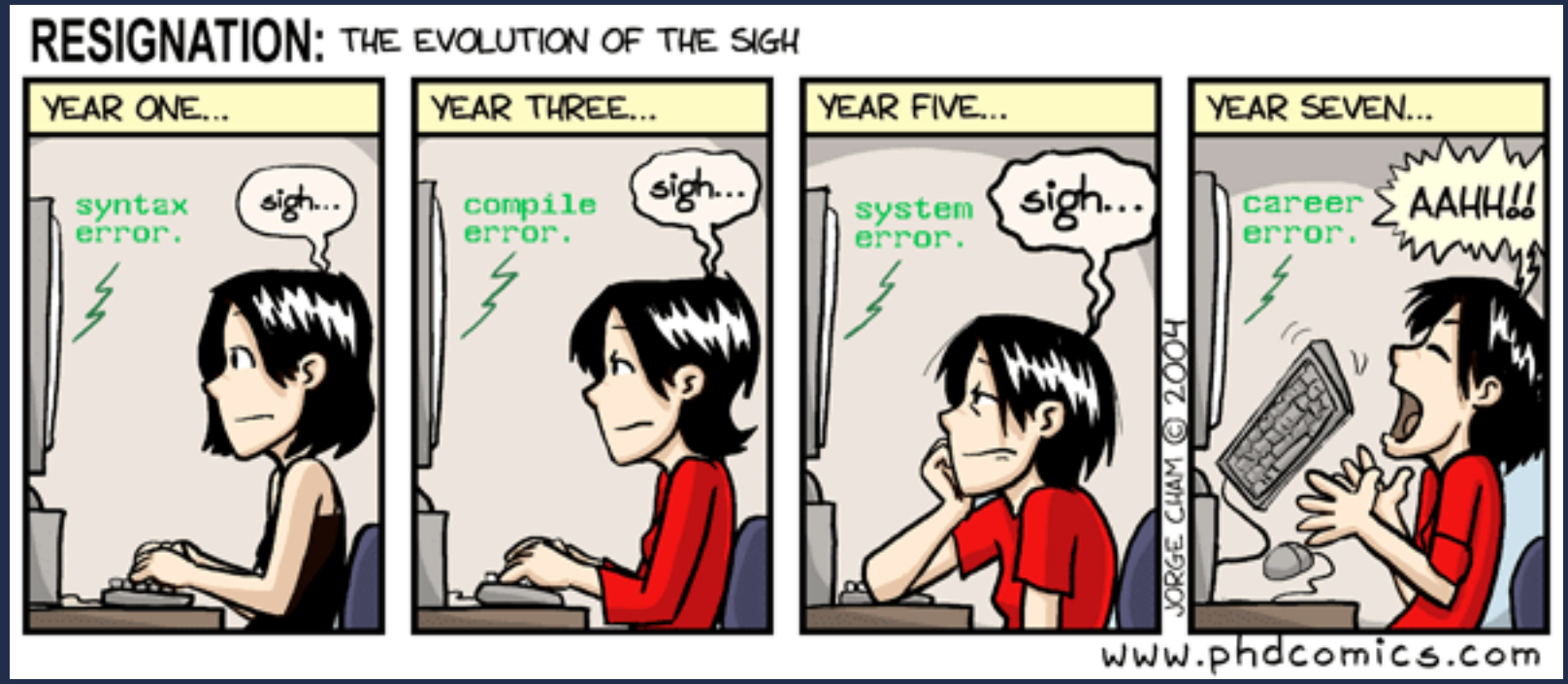
Argonne NATIONAL LABORATORY

What is a container?

The definition from Docker Manual:

- A container is a runtime instance of an image--what the image becomes in memory when executed (that is, an image with state, or a user process), i.e. a container is launched by running an image, but you first build the image.
- An image is an executable package that includes everything needed to run an application--the code, a runtime, libraries, environment variables, and configuration files.

Do we need containers?



It can make your life easier.

http://phdcomics.com/comics/archive.php?comicid=531

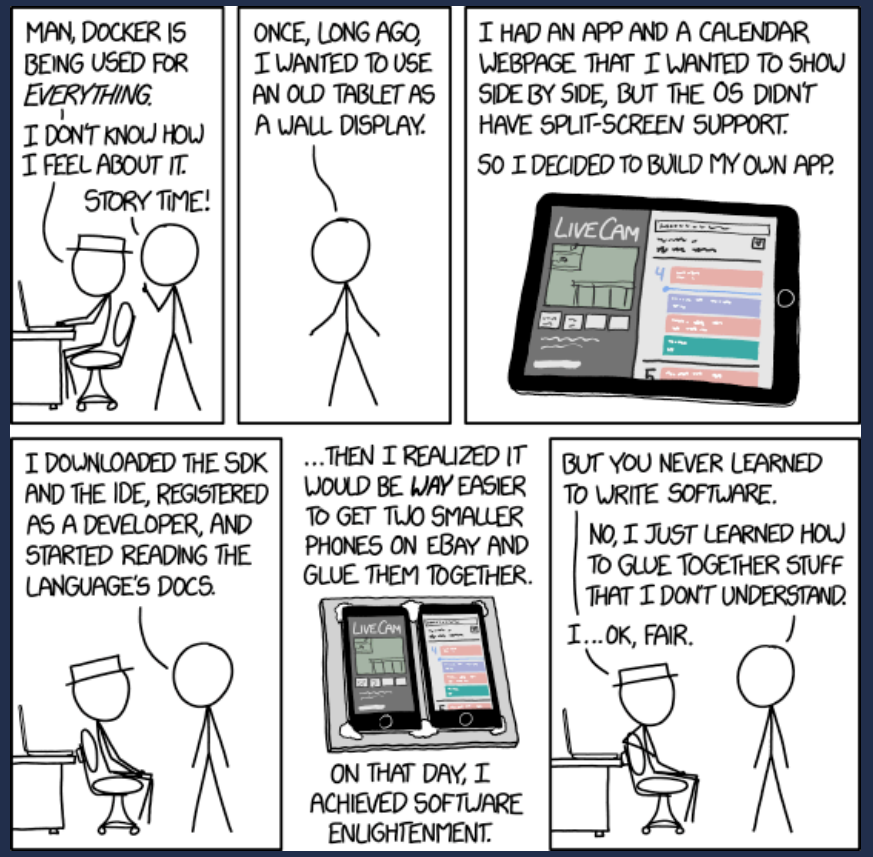
Advantages

- Portability
  - You can run your app on your laptop, local cluster or a *supercomputer*.
  - No need to ask system admins to install a library for you.

- Reproducibility
  - $28 billion per year is spent on preclinical research that is not reproducible.
  - Include all data required to run your application in the image.
  - https://www.nature.com/news/software-simplified-1.22059

- Faster development and production
  - You can build the image anywhere, no need to compile on login node.
  - You can create an image based on existing images.

A Silver Bullet?



May not be the best solution.

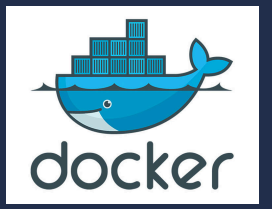
https://xkcd.com/1988/

HPC use cases

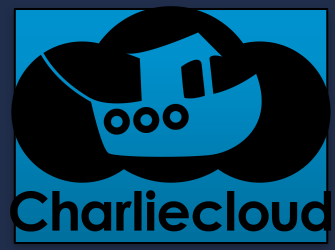
- You have been using containers
- You have installation problems
  - You need a specific version of a system library, i.e. glibc
  - You have a lot of dependencies
  - Long compilation time
- You only have a binary, which is not compatible with
- You want to share your environment with others
- You want reproducibility (i.e. avoid problems due to system changes)
- You want to run on different machines with minimal effort

Container Solutions

- Linux Containers (LXC)
  - Uses kernel namespaces and cgroups for resource management.
- Docker
  - Extension of LXC, current enterprise solution for micro-services.



- HPC containers:
  - Shifter (NERSC)
  - Charlie Cloud (LANL)
  - Singularity (LBL, Sylabs Inc.)



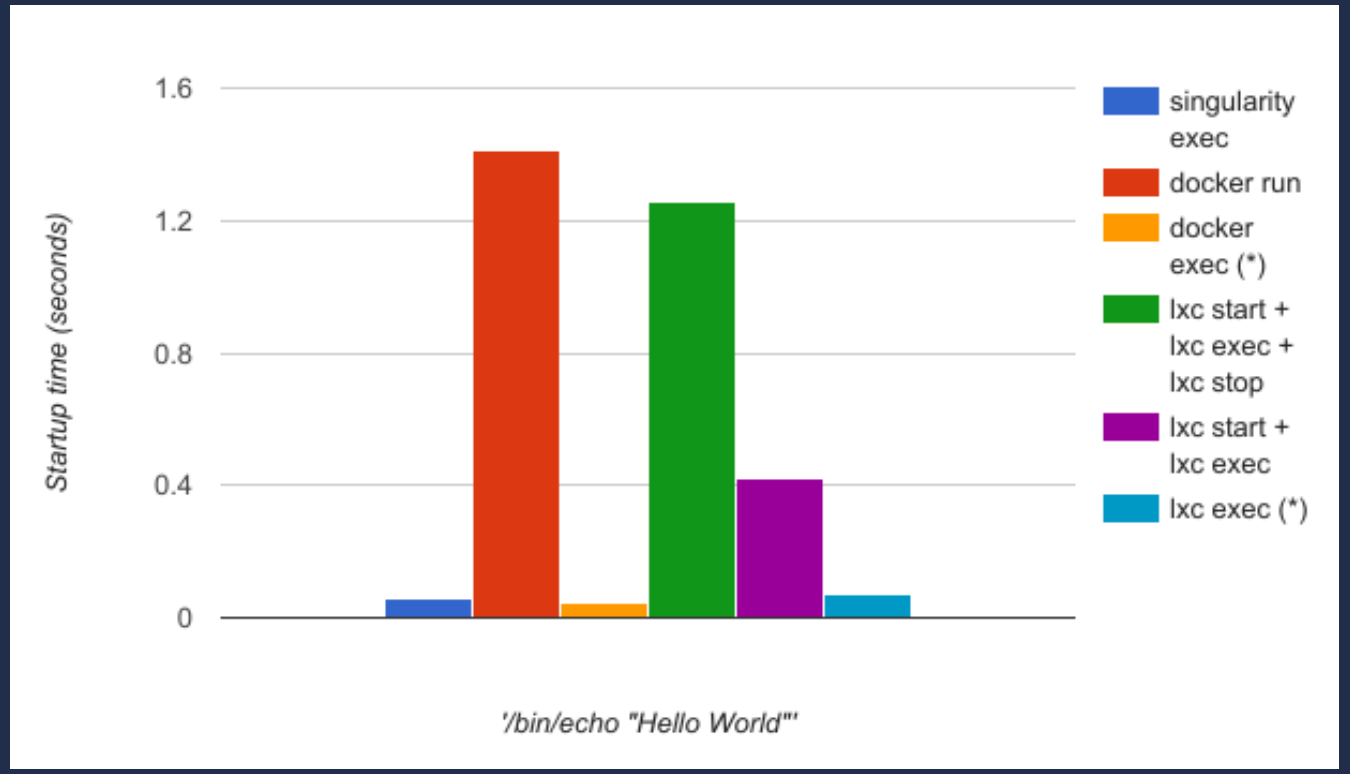
Container Comparison

	Singularity	Shifter	Charlie Cloud	Docker
Privilege model	SUID/UserNS	SUID	UserNS	Root Daemon
Supports current production Linux distros	Yes	Yes	No	No
Internal image build/bootstrap	Yes	No*	No*	No***
No privileged or trusted daemons	Yes	Yes	Yes	No
No additional network configurations	Yes	Yes	Yes	No
No additional hardware	Yes	Maybe	Yes	Maybe
Access to host filesystem	Yes	Yes	Yes	Yes**
Native support for GPU	Yes	No	No	No
Native support for InfiniBand	Yes	Yes	Yes	Yes
Native support for MPI	Yes	Yes	Yes	Yes
Works with all schedulers	Yes	No	Yes	No
Designed for general scientific use cases	Yes	Yes	No	No
Contained environment has correct perms	Yes	Yes	No	Yes
Containers are portable, unmodified by use	Yes	No	No	No
Trivial HPC install (one package, zero conf)	Yes	No	Yes	Yes
Admins can control and limit capabilities	Yes	Yes	No	No

G. M. Kurtzer, V. Sochat, and M. W. Bauer, “Singularity: Scientific containers for mobility of compute,” PLoS One, vol. 12, no. 5, pp. 1–20, 2017.

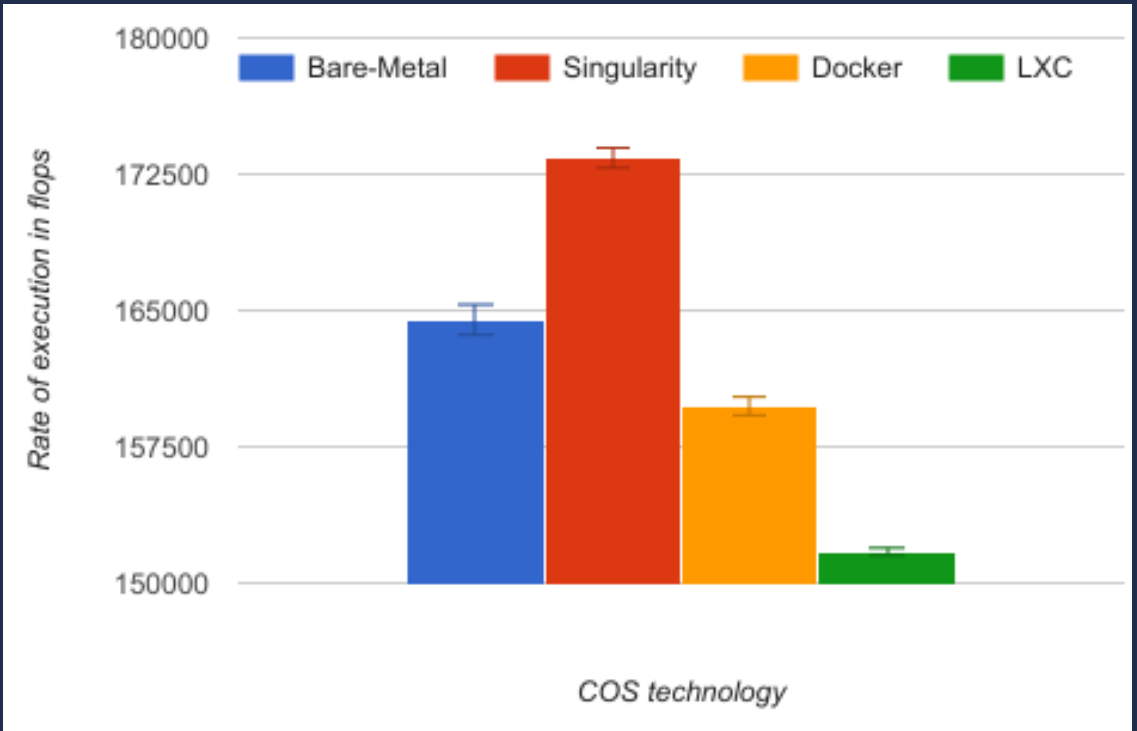
Argonne NATIONAL LABORATORY

Performance / Overhead



C. Arango, R. Dernat, and J. Sanabria, “Performance Evaluation of Container-based Virtualization for High Performance Computing Environments,” 2017.

Performance / Overhead



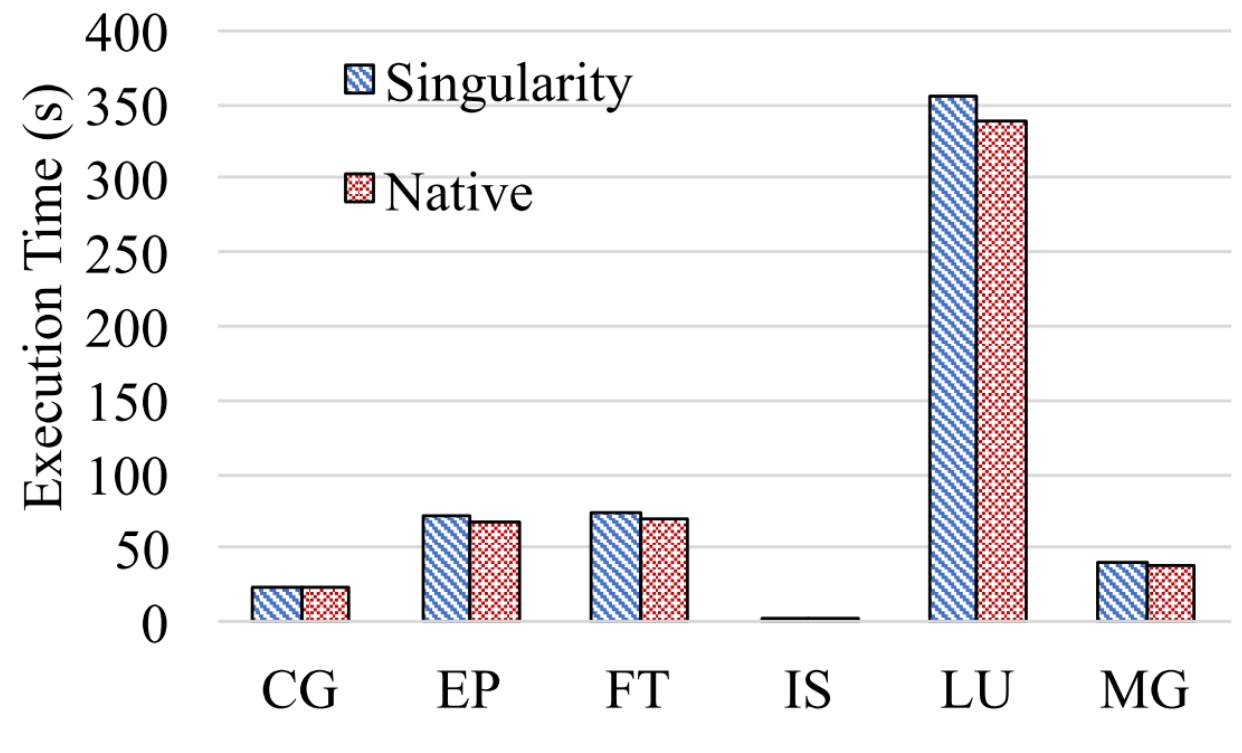
Singularity was able to achieve a **better** performance than **native** with 5.42% because it is not emulating a full hardware level virtualization (only the mount namespace) paradigm and as the image itself is only a single metadata lookup this can yield in very high performance benefits.

HPL benchmark, higher is better

C. Arango, R. Dernat, and J. Sanabria, "Performance Evaluation of Container-based Virtualization for High Performance Computing Environments," 2017.

Argonne NATIONAL LABORATORY

Performance / Overhead

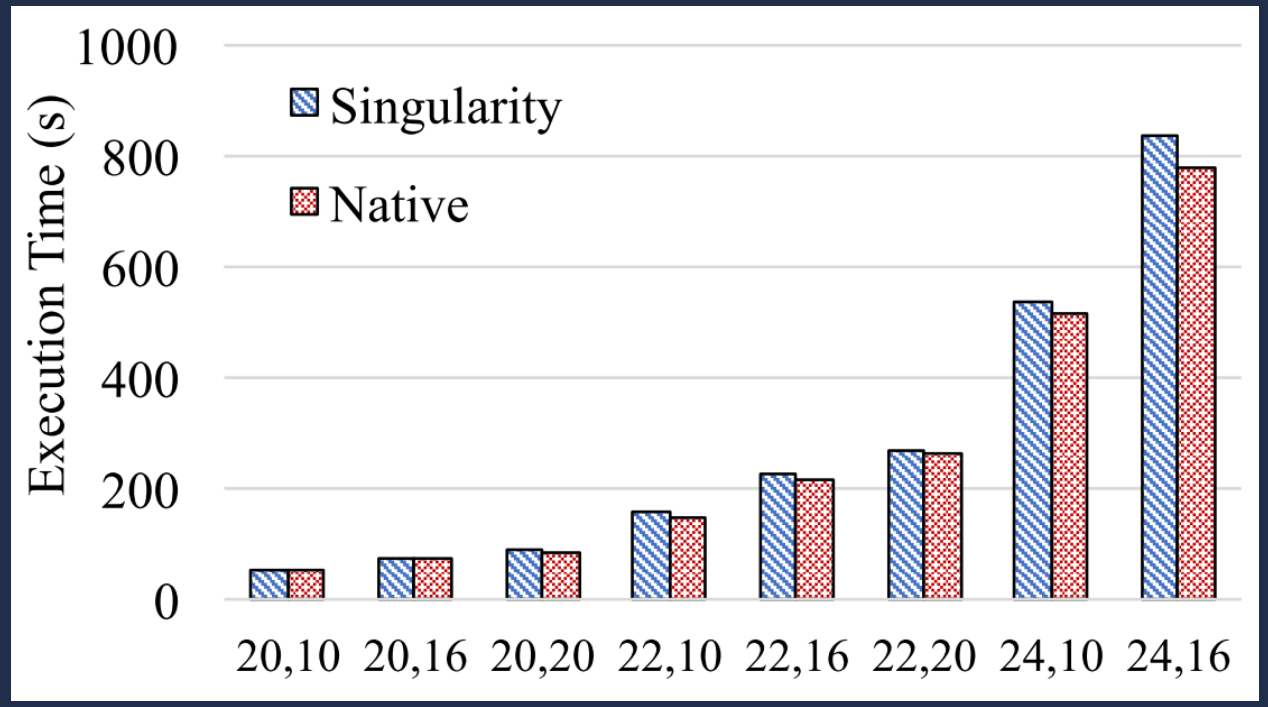


CG	Conjugate Gradient
EP	Embarrassingly Parallel
FT	3D FFT, all-to-all communication
IS	Integer Sort, random memory access
LU	Lower-Upper Gauss-Seidel solver
MG	Multi-Grid, memory intensive

X. Lu and D. K. Panda, “Is Singularity-based Container Technology Ready for Running MPI Applications on HPC Clouds?,” Proc. 10th Int. Conf. Util. Cloud Comput. (UCC ’17), pp. 151–160, 2017.

Argonne NATIONAL LABORATORY

Performance / Overhead



Graph-data analytics workload

Point-to-point communications

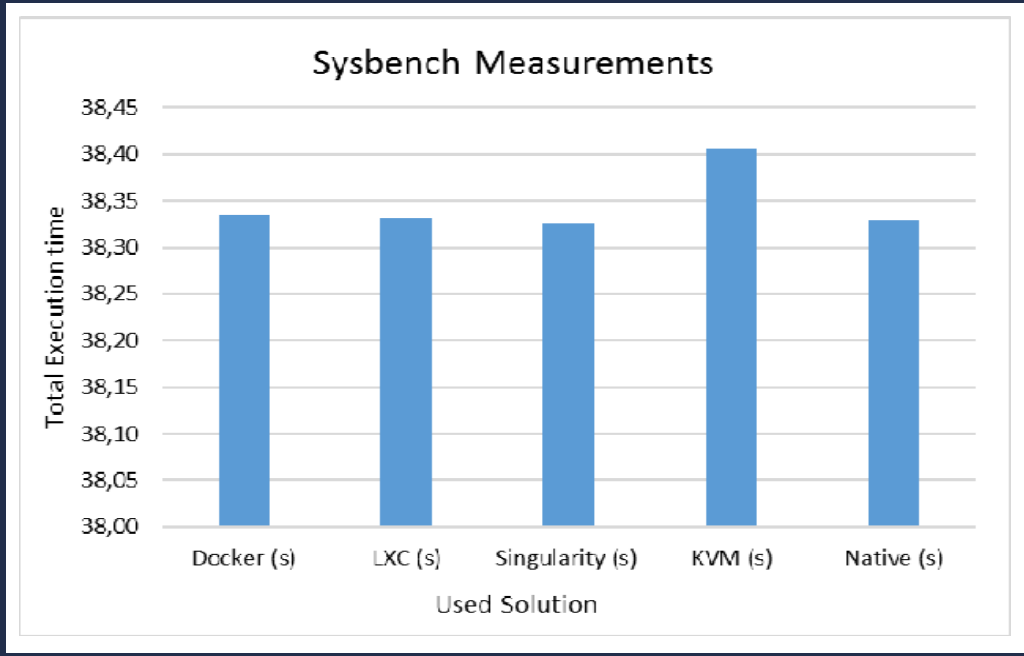
20,10 means 2^{20} vertices and 2^{10} edges

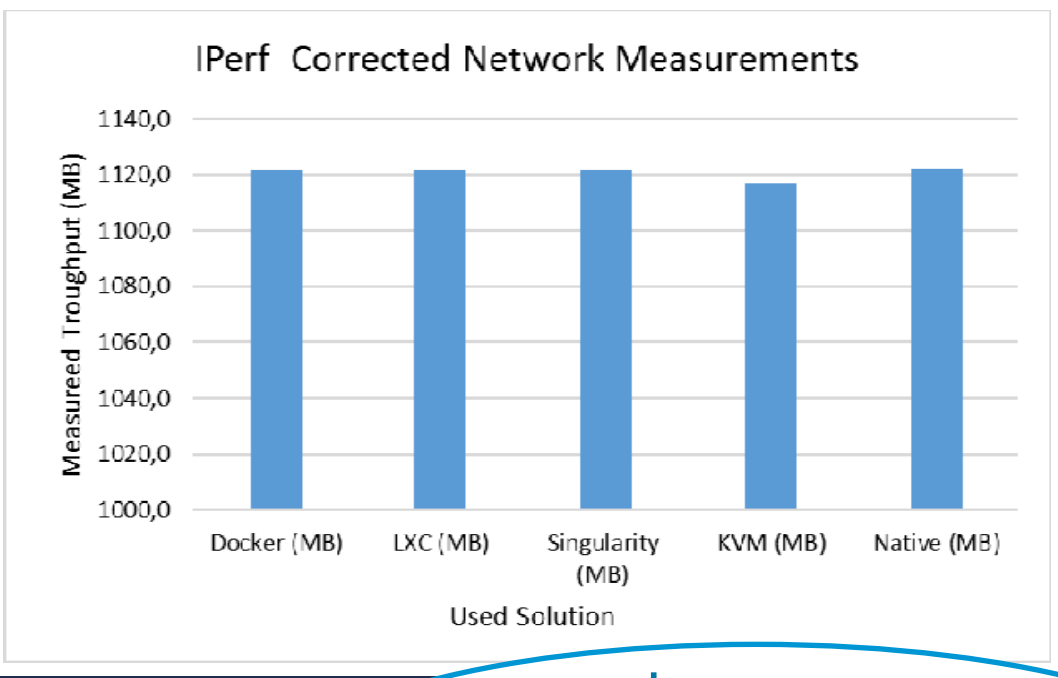
2 KNL nodes, 128 processes

Less than 8% overhead for CPU, memory, network, and IO

X. Lu and D. K. Panda, "Is Singularity-based Container Technology Ready for Running MPI Applications on HPC Clouds?," Proc. 10th Int. Conf. Util. Cloud Comput. (UCC '17), pp. 151–160, 2017.

Performance / Overhead

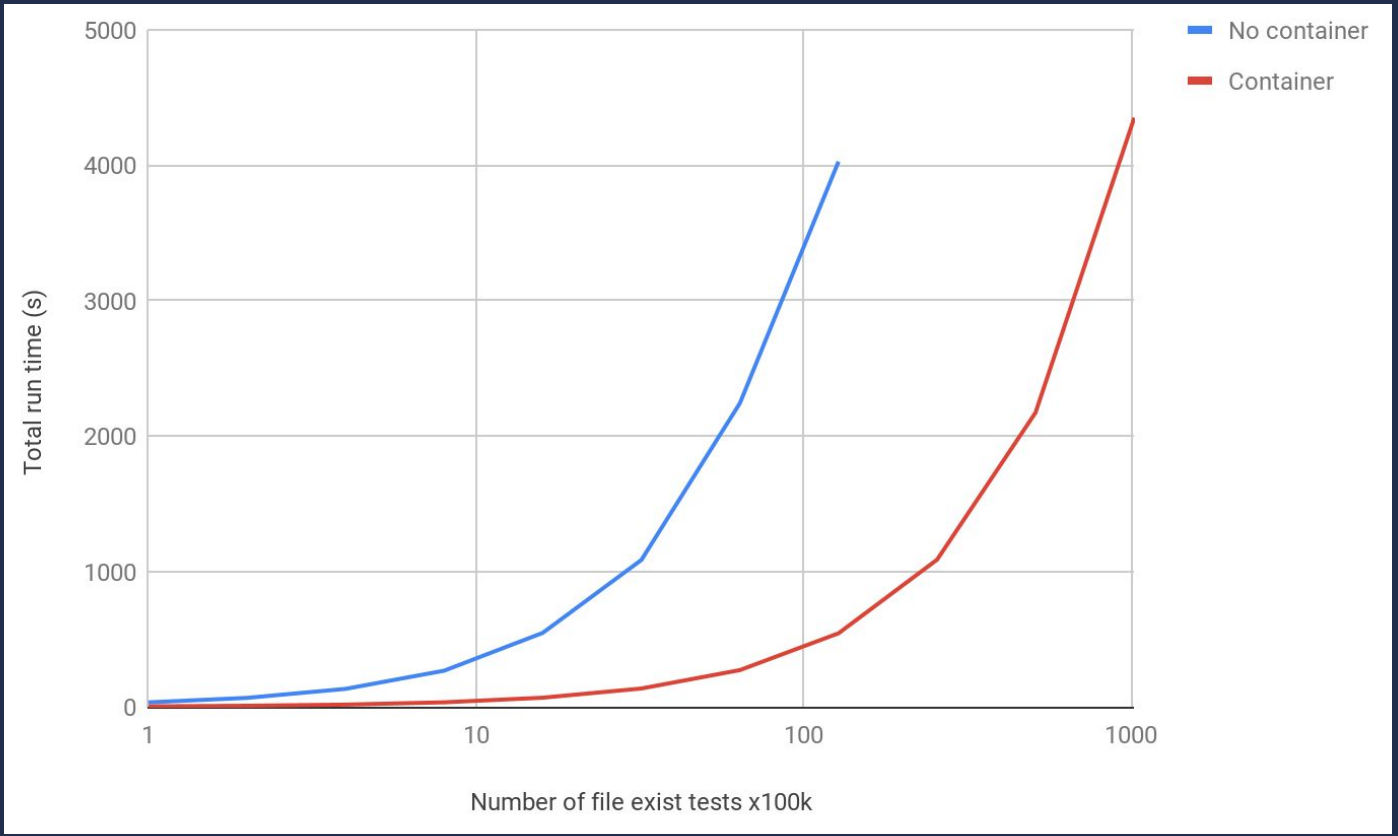




Less than 1% overhead for CPU, and network

Á. Kovács, "Comparison of different linux containers," 20 Int. Conf. Telecommun. Signal Process. TSP 2017, vol. 2017–January, pp. 47–51, 2017.

Theta Benchmarks - File checks



- One script creates *n* files for *n* MPI ranks.
- Another one checks the files if they exist
- Singularity caches its files making it ~6x faster

Spencer Williams, J. Taylor Childers
https://github.com/spencer-williams/sgww_argonne

Argonne NATIONAL LABORATORY

Installing Singularity

- You need root permissions to install Singularity.
- For Linux and Windows follow the instructions at https://sylabs.io/guides/3.4/user-guide/installation.html
- For Mac, Singularity Desktop beta release is available. You can simply download and install a DMG file.
  - Desktop version allows only remote builds.
  - You need to create an account at https://cloud.sylabs.io and enable a token at https://cloud.sylabs.io/auth/tokens to be able to build containers remotely.

How to use Singularity

- **You can pull and build Singularity images or execute them through the command line interface**

```
  Important Commands: singularity [cmd]
    build       Build a Singularity image
    exec        Run a command within a container
    help        Help about any command
    pull        Pull an image from a URI
    run         Run the user-defined default command within a container
    search      Search a Container Library for images
    shell       Run a shell within a container
    version     Show the version for Singularity
```

```
$singularity help
$singularity help exec
$singularity help pull
```


Argonne NATIONAL LABORATORY

How to use Singularity on Theta

- If you have seen the following error:

```
ERROR: ld.so: object '/soft/buildtools/trackdeps/$LIB/trackdeps.so' from LD_PRELOAD cannot
be preloaded (cannot open shared object file): ignored.
```

- Delete trackdeps module

```
>module delete trackdeps
```

- You can pull the image into your directory:

```
>singularity pull docker://godlovedc/lolcow
```

Host vs container

- **Get OS info for Theta:**

```
> cat /etc/os-release
NAME="SLES"
VERSION="12-SP3"
VERSION_ID="12.3"
PRETTY_NAME="SUSE Linux Enterprise Server 12 SP3"
ID="sles"
ANSI_COLOR="0;32"
CPE_NAME="cpe:/o:suse:sles:12:sp3"
```

- **Get OS info for the container:**

```
>singularity exec lolcow_latest.sif cat /etc/os-release
NAME="Ubuntu"
VERSION="16.04.3 LTS (Xenial Xerus)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 16.04.3 LTS"
VERSION_ID="16.04"
HOME_URL="http://www.ubuntu.com/"
SUPPORT_URL="http://help.ubuntu.com/"
BUG_REPORT_URL="http://bugs.launchpad.net/ubuntu/"
VERSION_CODENAME=xenial
UBUNTU_CODENAME=xenial
```


Argonne NATIONAL LABORATORY

Environment variables

- **Singularity passes all the environment variables in the shell as is except:**
  - `PATH –> USERPATH`
  - `SHLVL` **(shell level increases by 1)**
  - **Singularity libs are added to** `LD_LIBRARY_PATH`

```
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/.singularity.d/libs
```

- **Singularity adds the following variables**
  - `SINGULARITY_APPNAME=`
  - `SINGULARITY_CONTAINER=/PATH/TO/IMAGE.sif`
  - `SINGULARITY_NAME=IMAGE.sif`
- **You can start with a clean environment by using** `--clean-env` **(not recommended)**

Argonne NATIONAL LABORATORY

Binding directories

- Singularity allows you to bind directories on your host to the container.
- Default bind points are

`$HOME` , `$PWD`, `/sys` , `/proc`, `/tmp`, `/var/tmp`, `/etc/resolv.conf`, and `/etc/passwd`.

- Usage:

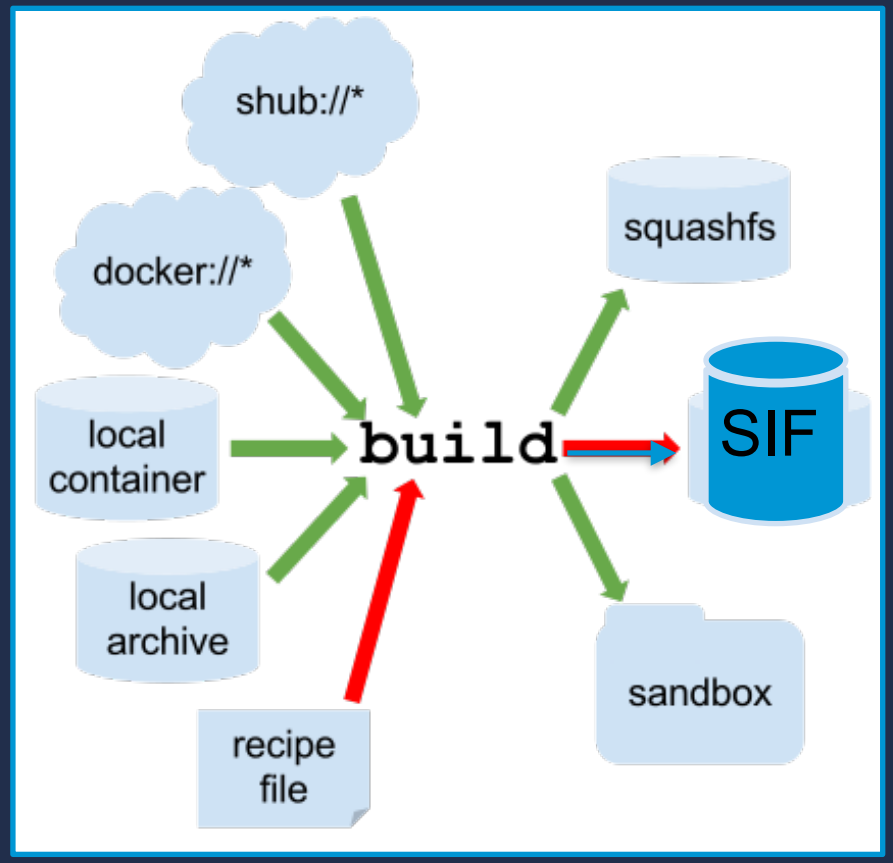
```
-B src[:dest[:opts]]
--bind src[:dest[:opts]]
```

- Options are 'rw' (read/write default) or 'ro' (read-only) . Multiple bind paths can be given by a comma separated list.

```
singularity shell -B /opt,/data:/mnt:rw my_container.sif
```


Argonne NATIONAL LABORATORY

Building container images



Build based on an image on a hub:
(Does not require sudo)

```
$>singularity build <OPT> <IMG> library://xxx/yy:z
$>singularity build <OPT> <IMG> shub://xxx/yy:z
$>singularity build <OPT> <IMG> docker://xxx/yy:z
```

Build based on a recipe file (Requires sudo)

```
$>sudo singularity build <OPT> <IMG> Singularity.tag
```

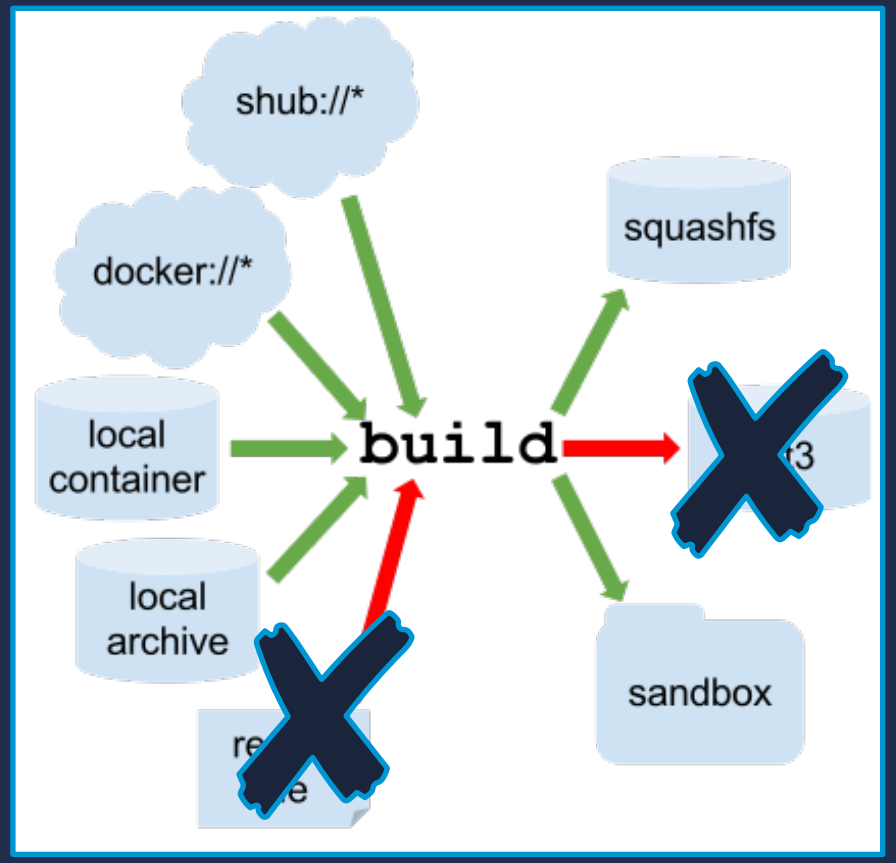
By default created image <IMG> is read only in squashfs format.
Writable image can be created: (Requires sudo)

```
--sandbox:  Writable container within a directory
--writable: Legacy writable image format (ext3)
```

https://www.sylabs.io/guides/2.6/user-guide/build_a_container.html

Argonne NATIONAL LABORATORY

Singularity on Theta



If you are not going to run the container in parallel, you can use *any* images from Singularity or Docker hub.

```
$> singularity build quip.simg shub://libAtoms/QUIP
```

You can run the image like any other application.

```
$> singularity run quip.simg
```

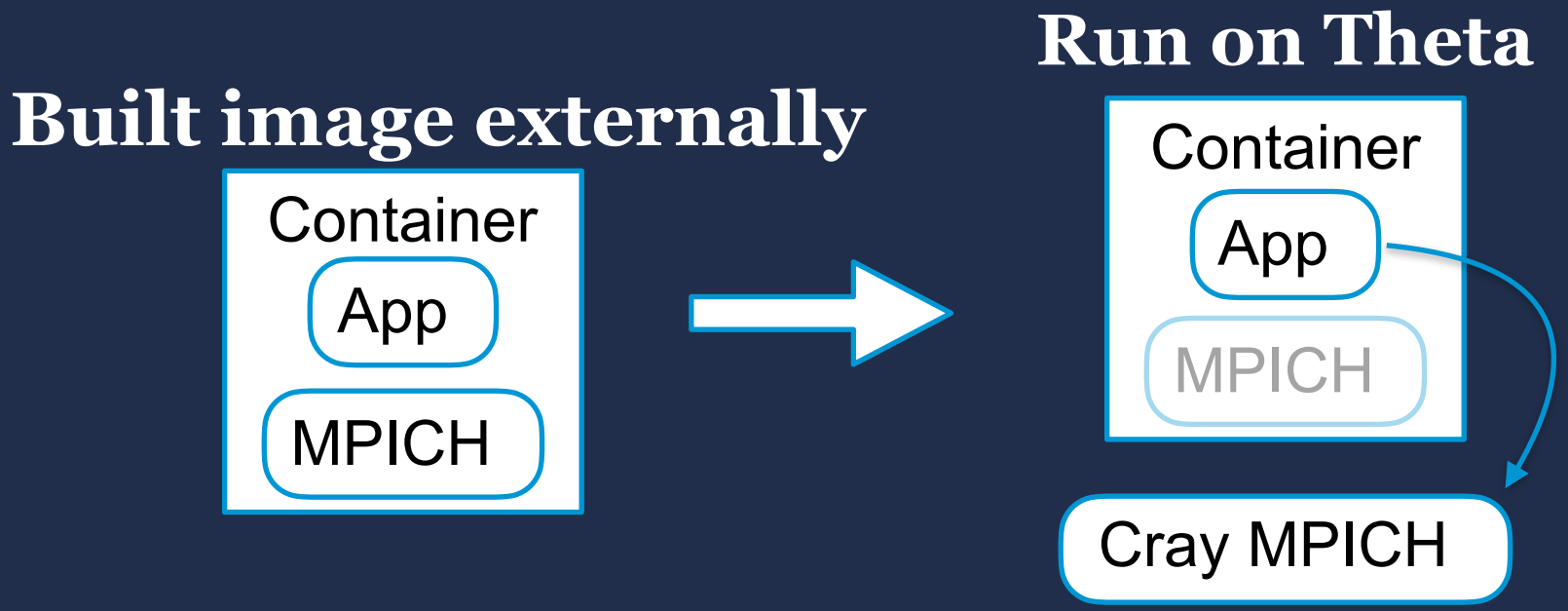
You can bind a directory on Theta to your container with -b <host_path>:<container_path>:<opt>
<opt> = ro for read-only,
<opt> = rw for read/write

```
$> singularity run -B ./mydata:/data:rw quip.simg
```

Singularity on Theta (MPI applications)

We need to use Cray MPI on Theta, so we cannot use any image.

We can build our special image with a Singularity recipe file



Source of base image

Make working directory.
Copy files from into image.

During the ‘setup’ phase, the image does not yet exist and is still on the host filesystem at the path `SINGULARITY_ROOTFS`
This creates app directory at ‘/myapp’ in the image

```
1  Bootstrap: docker
2  From: centos
3
4  %setup
5      echo ${SINGULARITY_ROOTFS}
6      mkdir ${SINGULARITY_ROOTFS}/myapp
7      cp pi.c ${SINGULARITY_ROOTFS}/myapp/
8
9  %post
10     yum update -y
11     yum groupinstall -y "Development Tools"
12     yum install -y gcc
13     yum install -y gcc-c++
14     yum install -y wget
15     cd /myapp
16     # install MPICH
17     wget http://www.mpich.org/static/downloads/3.2.1/mpich-3.2.1.tar.gz
18     tar xf mpich-3.2.1.tar.gz
19     cd mpich-3.2.1
20     # disable the addition of the RPATH to compiled executables
21     # this allows us to override the MPI libraries to use those
22     # found via LD_LIBRARY_PATH
23     ./configure --prefix=$PWD/install --disable-wrapper-rpath
24     make -j 4 install
25     # add to local environment to build pi.c
26     export PATH=$PATH:$PWD/install/bin
27     export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$PWD/install/lib
28     cd ..
29     mpicc -o pi -fPIC pi.c
30
31 %runscript
32     /myapp/pi
```

Source of base image

Make working directory.
Copy files from into image.

Commands required for installing your application.

Specify the executable to run with container is called

```
1  Bootstrap: docker
2  From: centos
3
4  %setup
5     echo ${SINGULARITY_ROOTFS}
6     mkdir ${SINGULARITY_ROOTFS}/myapp
7     cp pi.c ${SINGULARITY_ROOTFS}/myapp/
8
9  %post
10    yum update -y
11    yum groupinstall -y "Development Tools"
12    yum install -y gcc
13    yum install -y gcc-c++
14    yum install -y wget
15    cd /myapp
16    # install MPICH
17    wget http://www.mpich.org/static/downloads/3.2.1/mpich-3.2.1.tar.gz
18    tar xf mpich-3.2.1.tar.gz
19    cd mpich-3.2.1
20    # disable the addition of the RPATH to compiled executables
21    # this allows us to override the MPI libraries to use those
22    # found via LD_LIBRARY_PATH
23    ./configure --prefix=$PWD/install --disable-wrapper-rpath
24    make -j 4 install
25    # add to local environment to build pi.c
26    export PATH=$PATH:$PWD/install/bin
27    export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$PWD/install/lib
28    cd ..
29    mpicc -o pi -fPIC pi.c
30
31 %runscript
32    /myapp/pi
```

Source of base image

Make working directory.
Copy files from into image.

Commands to install my image with the application.

Typically containers are built to run one executable.

```
singularity run myapp.img
```

Specify the executable to run with container is called

```
1  Bootstrap: docker
2  From: centos
3
4  %setup
5     echo ${SINGULARITY_ROOTFS}
6     mkdir ${SINGULARITY_ROOTFS}/myapp
7     cp pi.c ${SINGULARITY_ROOTFS}/myapp/
8
9  %post
10    yum update -y
11    yum groupinstall -y "Development Tools"
12    yum install -y gcc
13    yum install -y gcc-c++
14    yum install -y wget
15    cd /myapp
16    # install MPICH
17    wget http://www.mpich.org/static/downloads/3.2.1/mpich-3.2.1.tar.gz
18    tar xf mpich-3.2.1.tar.gz
19    cd mpich-3.2.1
20    # disable the addition of the RPATH to compiled executables
21    # this allows us to override the MPI libraries to use those
22    # found via LD_LIBRARY_PATH
23    ./configure --prefix=$PWD/install --disable-wrapper-rpath
24    make -j 4 install
25    # add to local environment to build pi.c
26    export PATH=$PATH:$PWD/install/bin
27    export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$PWD/install/lib
28    cd ..
29    mpicc -o pi -fPIC pi.c
30
31 %runscript
32    /myapp/pi
```

Using mpich installed Image as the base

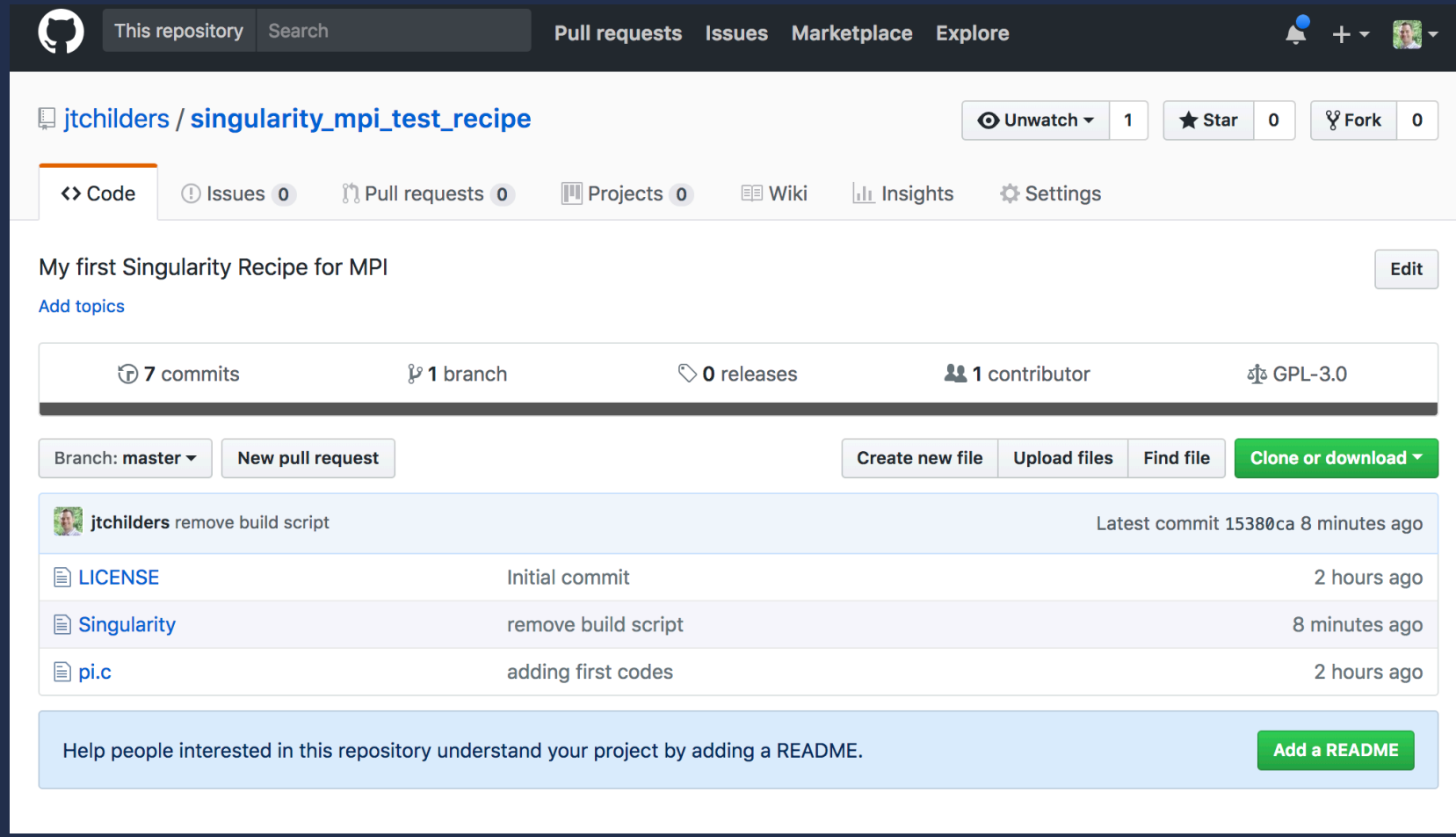
Commands to install PETSc

```
Bootstrap: shub
From: keceli/mpi_benchmark:theta

%post
   export PATH=$PATH:/mpich-3.2.1/install/bin/
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/mpich-3.2.1/
install/lib
   export PETSC_DIR=/container/petsc
   export PETSC_ARCH=arch-container
   export PETSC_VERSION=v3.11.1
   yum update -y
   yum install -y openssh-server openssh-clients python-
devel
   cd /container
   git clone https://bitbucket.org/petsc/petsc petsc
   cd $PETSC_DIR
   git checkout $PETSC_VERSION
   ./configure --with-shared-libraries=1 --with-
debugging=1 --download-fblaslapack --with-cc=mpicc --with-
cxx=mpicxx --with-fc=mpif90
   make all
   cd ./src/ksp/ksp/examples/tutorials
   make ex5
%environment
   export PETSC_DIR=/container/petsc
```

Create new Github Repository

- https://github.com/jtchilders/singularity_mpi_test_recipe
- Need to add recipe file inside with filename 'Singularity'
- Add file pi.c from previous link

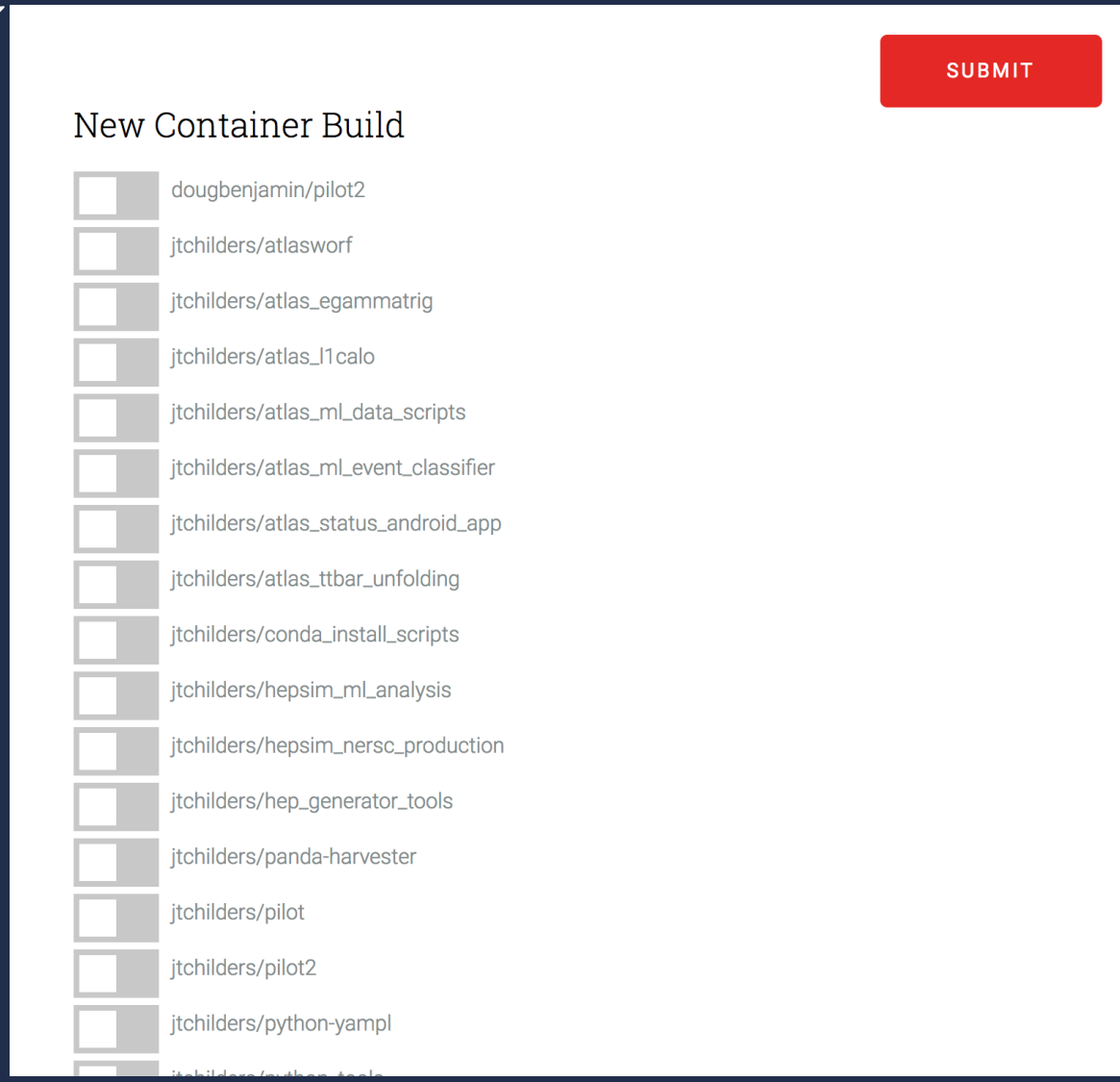


Create Singularity Hub Account

- Goto: https://www.singularity-hub.org/login/
- Authenticate using your Github account
- You can then add github repositories to your container collection.
- Click the big red button

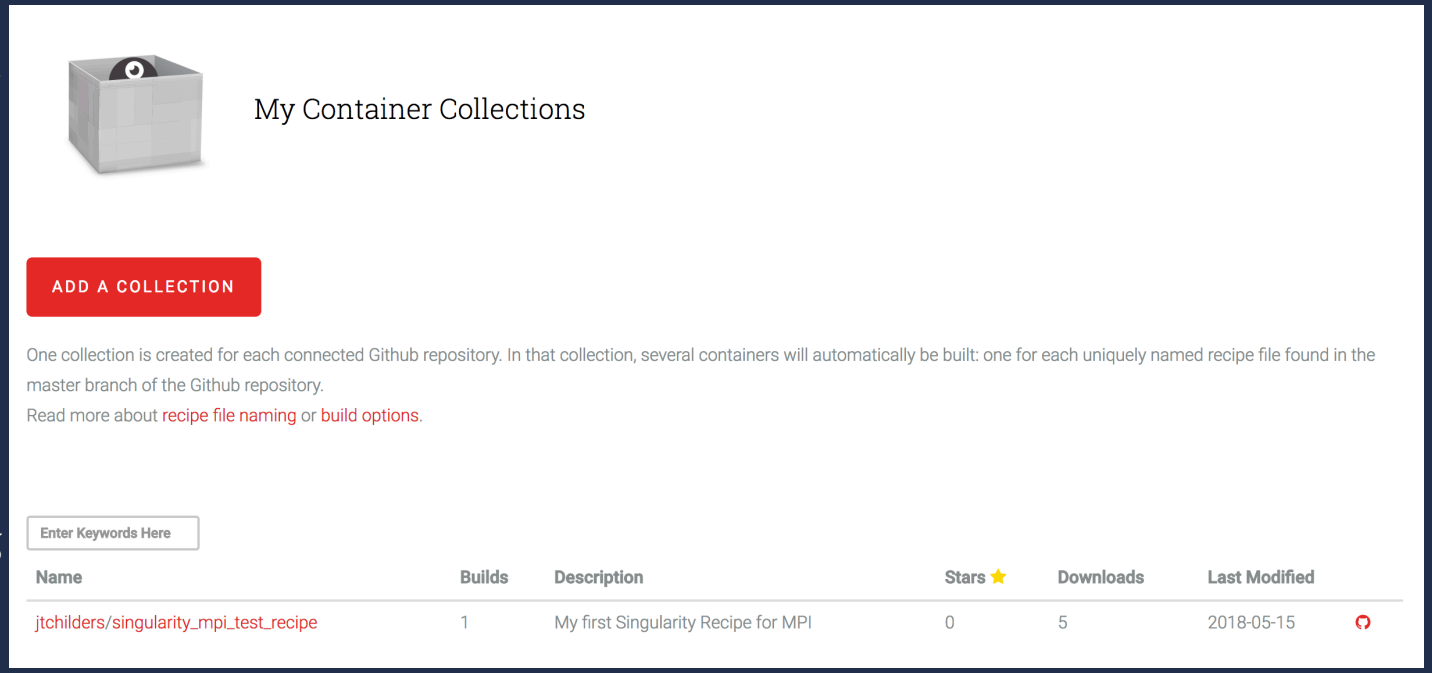
Create Singularity Hub Account

- Goto: https://www.singularity-hub.org/login/
- Authenticate using your Github account
- You can then add github repositories to your container collection.
- Click the big red button
- Select your new repository and click the big red button



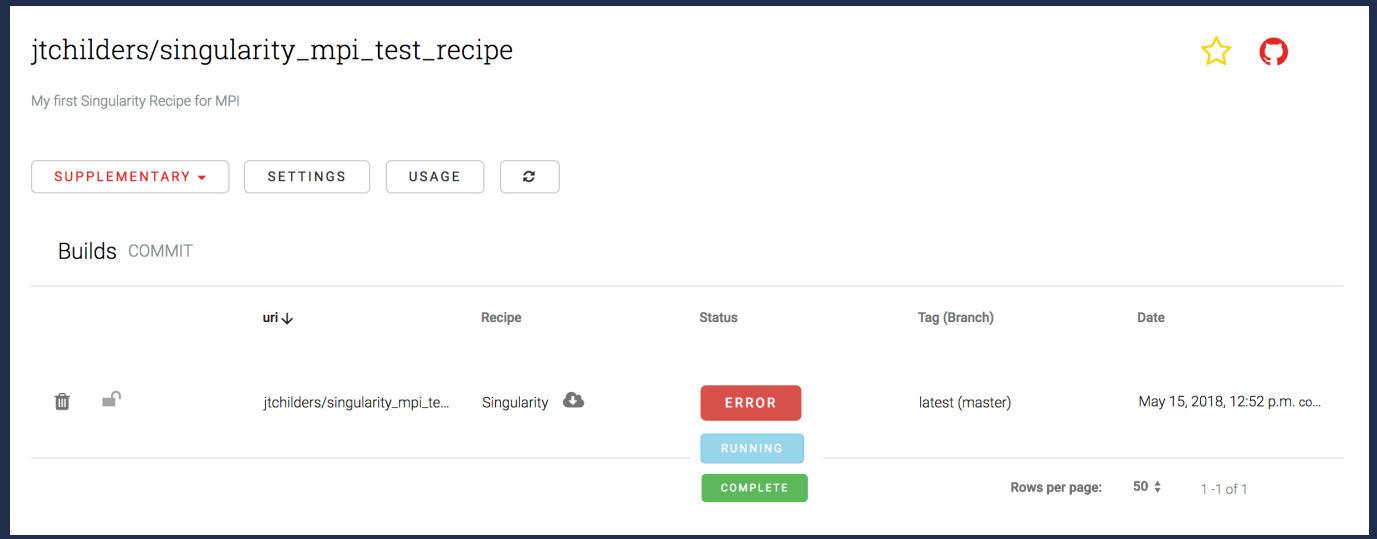
Create Singularity Hub Account

- Go to: https://www.singularity-hub.org/login/
- Authenticate using your Github account
- You can then add github repositories to your container collection.
- Click the big red button
- Select your new repository and click the big red button
- Now you have your recipe listed and Singularity Hub will begin recursively searching the repo for any files named 'Singularity' and building those recipes
- Our example only has 1 recipe
- Click on the recipe



Create Singularity Hub Account

- Goto: https://www.singularity-hub.org/login/
- Authenticate using your Github account
- You can then add github repositories to your container collection.
- Click the big red button
- Select your new repository and click the big red button
- Now you have your recipe listed and Singularity Hub will begin recursively searching the repo for any files named 'Singularity' and building those recipes
- Our example only has 1 recipe
- Click on the recipe to see it's build status
- Error messages during build can be seen by clicking the big red button
- Otherwise it will list the container as COMPLETE



Job submission script

```
$> singularity pull shub://keceli/mpi_benchmark:theta
```

```
#!/bin/bash
#COBALT -t 30
#COBALT -q training
#COBALT -n 2
#COBALT -A SDL_Workshop

module swap cray-mpich cray-mpich-abi
module delete trackdeps
export LD_LIBRARY_PATH=/opt/cray/wlm_detect/default/lib64/:/opt/cray/diag/lib/:$CRAY_LD_LIBRARY_PATH:
$LD_LIBRARY_PATH
export LD_PRELOAD=/soft/datascience/libfake/libfake.so.1
MOUNTEDVOLUMES=/opt/cray,/etc/alternatives,/soft/datascience/libfake
MYIMG=/projects/SDL_Workshop/training/Singularity/mpibenchmark.sif
aprun -n 8 -N 4 singularity run -B $MOUNTEDVOLUMES $MYIMG
```

Summary

- Containers can be helpful for
  - Portability
    - MPI stack requires special care.
  - Reproducibility
  - Faster development cycles
- Reasonable performance
  - There might be additional performance penalties due to dynamic linking, fat images, moderate optimization
- Very useful for complicated software stacks with many dependencies.

Argonne NATIONAL LABORATORY

Quick Reference

- Official Singularity documentation
  - https://www.sylabs.io/docs/
  - singularity-container.slack.com
- How to use Singularity on Theta and Cooley
  - https://www.alcf.anl.gov/user-guides/singularity
  - https://www.alcf.anl.gov/user-guides/singularity-cooley
- Similar tutorials from other HPC centers:
  - http://www.sdsc.edu/support/user_guides/tutorials/singularity.html
  - https://github.com/NIH-HPC/Singularity-Tutorial
  - https://ulhpc-tutorials.readthedocs.io/en/latest/containers/singularity/
- Github repo to check Singularity source and issues
  - https://github.com/sylabs/singularity
- Singularity hub
  - https://www.singularity-hub.org/
- Singularity container library (new)
  - https://cloud.sylabs.io/library
- Docker hub
  - https://hub.docker.com/

Argonne NATIONAL LABORATORY

Resources

- Official Singularity documentation
  - https://www.sylabs.io/docs/
  - singularity-container.slack.com
- How to use Singularity on Theta
  - https://www.alcf.anl.gov/user-guides/singularity
- Similar tutorials from other HPC centers:
  - http://www.sdsc.edu/support/user_guides/tutorials/singularity.html
  - https://github.com/NIH-HPC/Singularity-Tutorial
  - https://ulhpc-tutorials.readthedocs.io/en/latest/containers/singularity/
- Github repo to check Singularity source and issues
  - https://github.com/sylabs/singularity
- Singularity registry
  - https://www.singularity-hub.org/
- Docker registry
  - https://hub.docker.com/
- http://geekyap.blogspot.com/2016/11/docker-vs-singularity-vs-shifter-in-hpc.html

Argonne NATIONAL LABORATORY

Any Questions?

References

- "A Tale of Two Systems: Using Containers to Deploy HPC Applications on Supercomputers and Clouds"
- "Charliecloud: unprivileged containers for user-defined software stacks in HPC"
- "Singularity: Scientific containers for mobility of compute"
- "Contain This, Unleashing Docker for HPC"
- "Performance Evaluation of Container-Based Virtualization for High Performance Computing Environments" **(There is a 2013 paper on IEEE with the same title)**
- "Comparison of Different Linux Containers"

References

[1] G. M. Kurtzer, V. Sochat, and M. W. Bauer, “Singularity: Scientific containers for mobility of compute,” PLoS One, vol. 12, no. 5, pp. 1–20, 2017
[2] R. Priedhorsky and T. Randles, “Charliecloud,” Proc. Int. Conf. High Perform. Comput. Networking, Storage Anal. - SC ’17, pp. 1–10, 2017.
[3] Á. Kovács, “Comparison of different linux containers,” 2017 40th Int. Conf. Telecommun. Signal Process. TSP 2017, vol. 2017–Janua, pp. 47–51, 2017.
[4] A. J. Younge, K. Pedretti, R. E. Grant, and R. Brightwell, “A Tale of Two Systems: Using Containers to Deploy HPC Applications on Supercomputers and Clouds,” Proc. Int. Conf. Cloud Comput. Technol. Sci. CloudCom, vol. 2017–Decem, pp. 74–81, 2017.
[5] C. Arango, R. Dernat, and J. Sanabria, “Performance Evaluation of Container-based Virtualization for High Performance Computing Environments,” 2017.
[6] X. Lu and D. K. Panda, “Is Singularity-based Container Technology Ready for Running MPI Applications on HPC Clouds?,” Proc. 10th Int. Conf. Util. Cloud Comput. (UCC ’17), pp. 151–160, 2017.