# Singularity

CONTAINERS FOR HPC @ CAMBRIDGE

A WORKSHOP ON SINGULARITY AND CONTAINERS IN HPC AND CLOUD

# Bio: Gregory M. Kurtzer

- CEO of SingularityWare, LLC.
- Senior Architect at RStor, Inc.
- Scientific Advisor for LBNL/DOE/UC
- Open Source Work:
  - Founder: Centos Linux
  - Founder and project lead: Warewulf
  - Technical Steering Committee for OpenHPC
  - Founder and project lead: Singularity

# Quote:

Institut Pasteur

“singularity+warewulf+centos: winning combo”

# Containers

# What do containers provide to science?

- Reproducible software stacks
- Computing mobility and agility, run your workflow anywhere
- The ability to easily distribute and validate your work
- A reasonable escape from “dependency hell”
- Control of your own environment

# Quote:

Washington University in St. Louis

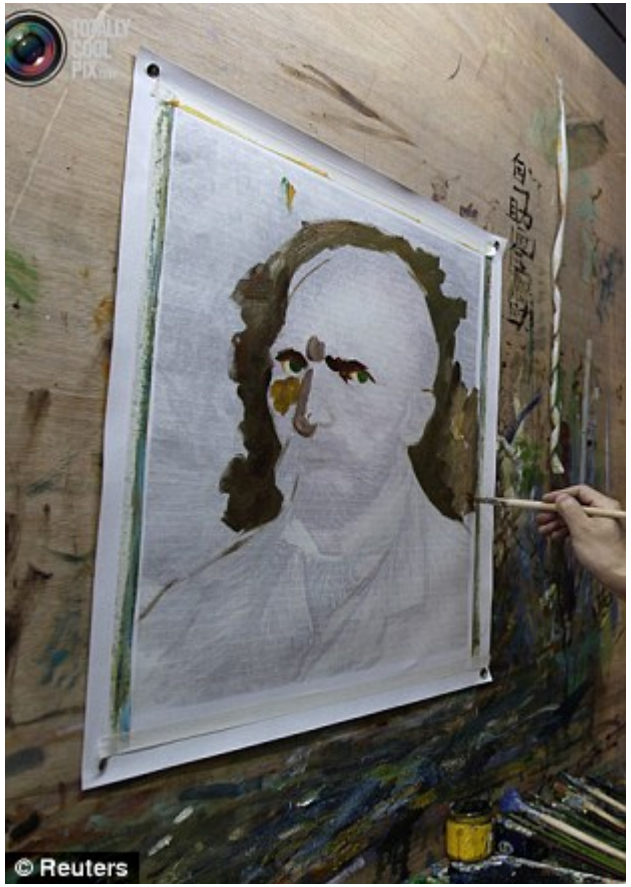
“Users have been asking for containers for years, but I've always resisted. Singularity addressed the majority of my complaints and couldn't have been easier to install”

# Reproducibility

In Science, reproducibility is of the utmost importance!

# Recreation

Without having access to the identical environment, we end up having to recreate that environment from scratch.



© Reuters

# Recreation

Sometimes the software environment is difficult and even impossible to recreate!

© Reuters

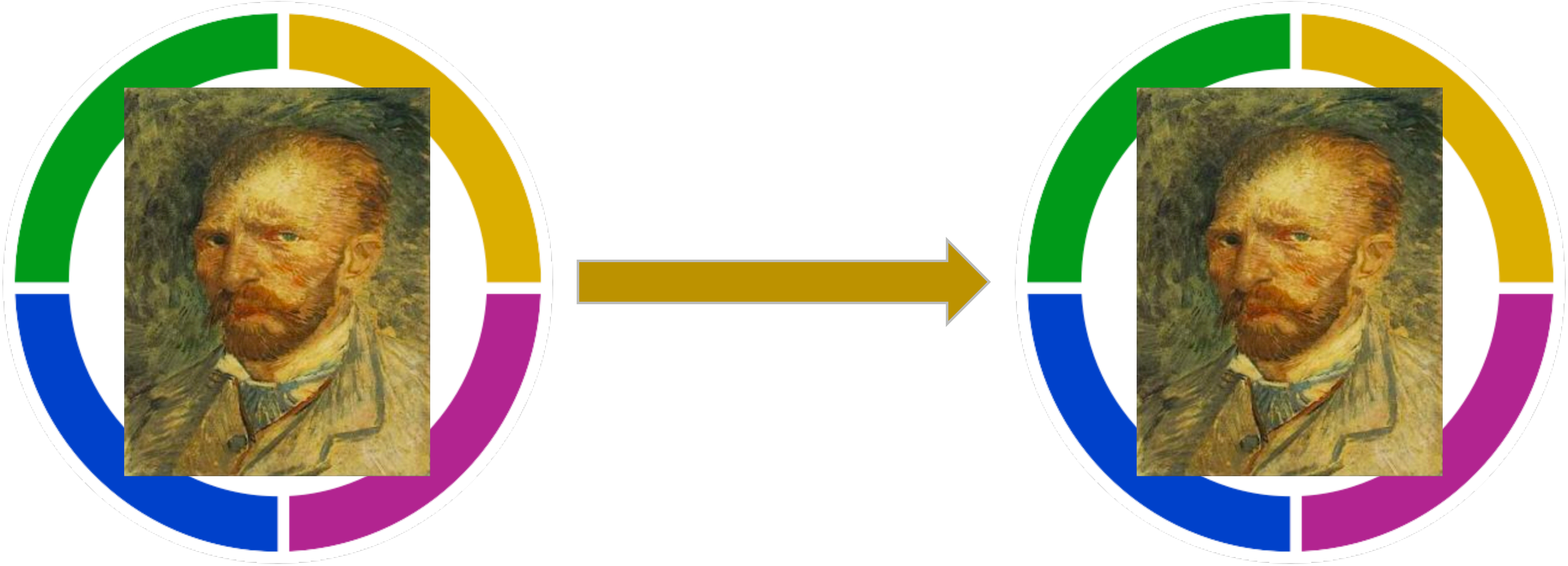
# Recreation

And sometimes you can get pretty close!

# Recreation != Reproducibility

!=

# Reproducibility Is Easy With Containers



# Quote:

THE UNIVERSITY OF CHICAGO

"Singularity allowed us to use software that was otherwise impossible to install under SL6, such as TensorFlow"

# Singularity: To the Rescue!

# Quote:

ARL

Army Research Laboratory

“Singularity is the best option among the big three considerations for HPC”

# Singularity: Overview

- Developed from necessity,… and demands, and threats, and bribes
- Built around a novel idea, talk to scientists and figure out what they need
- Designed specifically for reproducibility, mobility, computing agility, portability and ease of use
- Seamless integration with other HPC software and architectures (RMs, *MPI*, IB, Lustre, GPU)
- Limits user’s privileges, security contexts, access to data, and blurs the line of isolation
- Single file based container images, archival-able, standard permissions, controls compliant
- Docker Hub compatible

# Quote:

RUTGERS

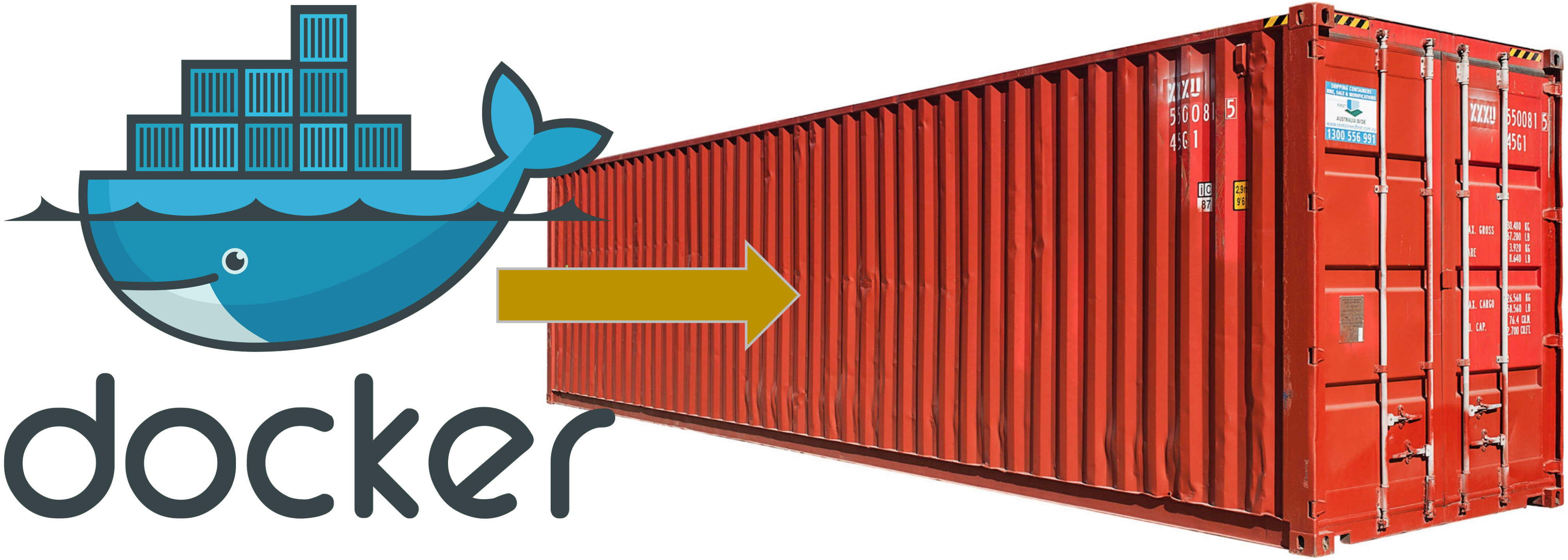
“Used to get around a GLIBC version requirement for binary distribution of the NCI GDC download tool on CentOS 6”

# Singularity: Creating a New Container

```
$ singularity create /tmp/Centos-7.img
```

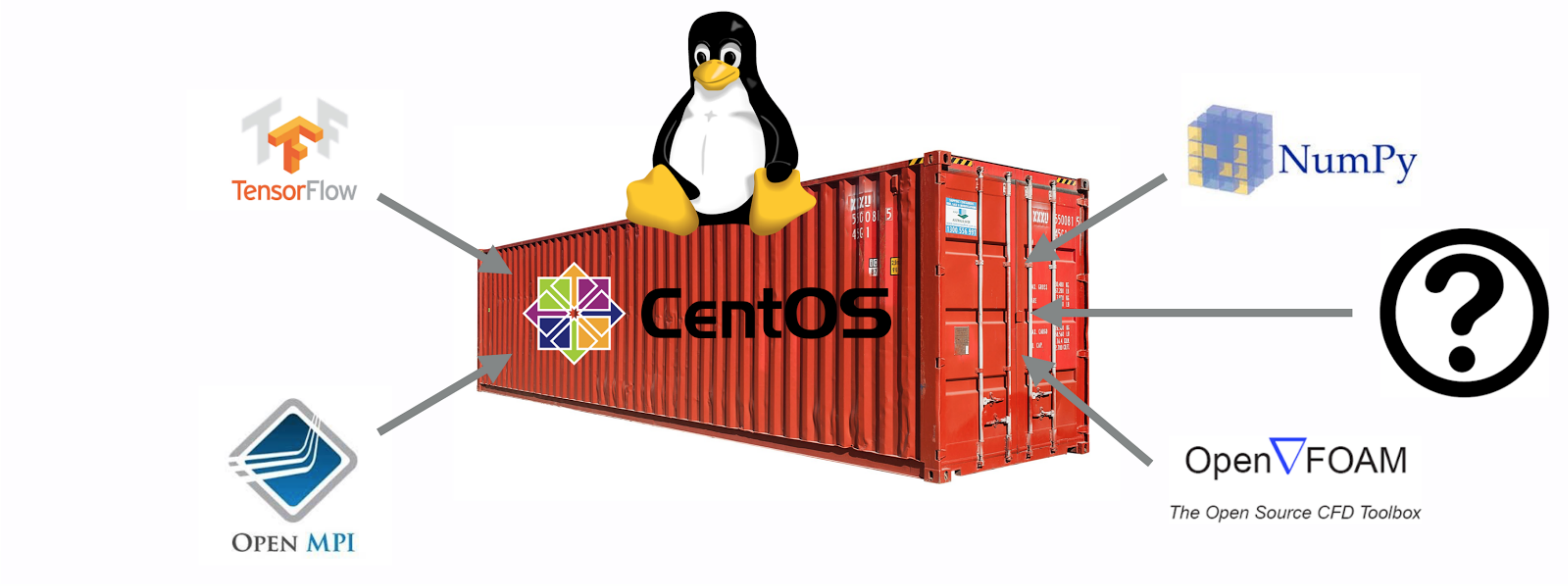
# Singularity: Importing

```
$ singularity import /tmp/Centos-7.img docker://centos:latest
```



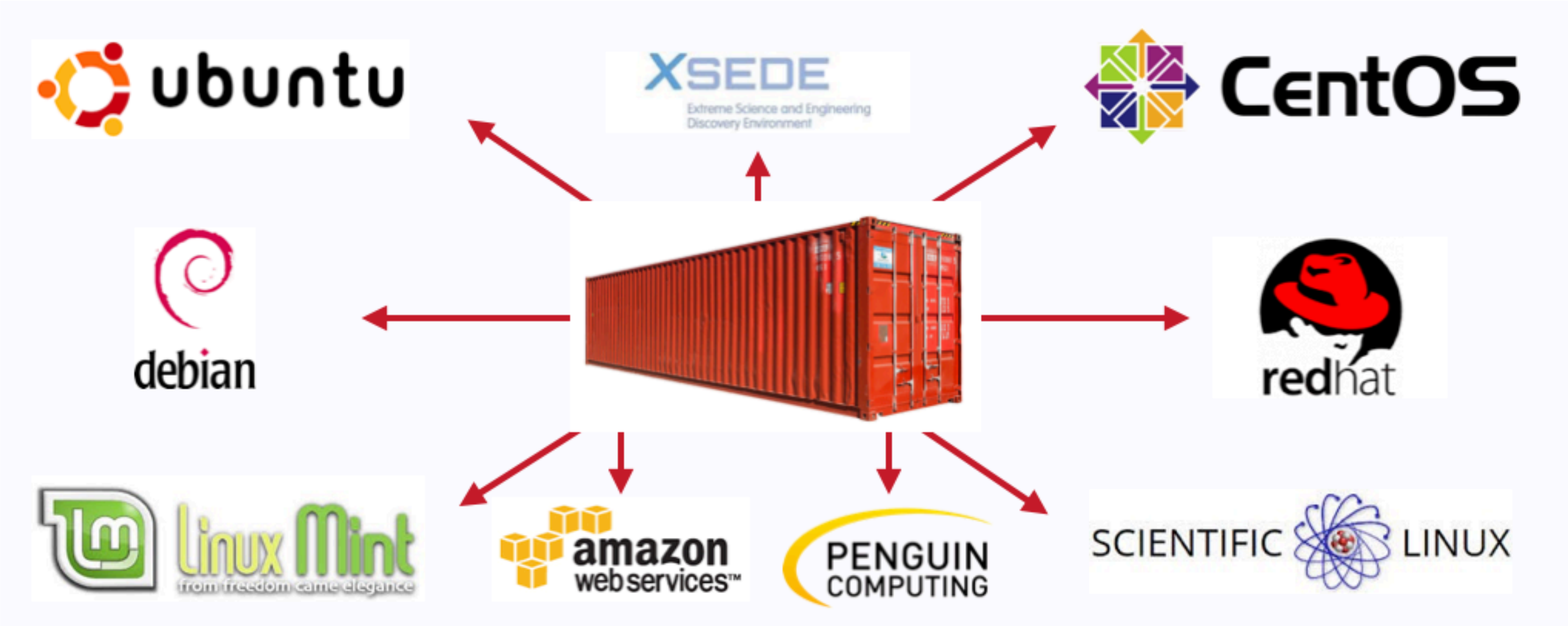
# Singularity: Bootstrapping

```
$ sudo singularity bootstrap /tmp/Centos-7.img centos.def
```



# Singularity: Final Container Image

# Singularity: Extreme Computing Agility



# Quote:

University of Michigan

“Singularity is a fabulous tool for providing forward and backward software compatibility on clusters and for reproducibility”

# Installation

```
$ git clone https://github.com/singularityware/singularity.git
$ cd singularity
$ ./autogen.sh
$ ./configure –prefix=/usr/local
$ make
$ sudo make install
```

# Initial Test Drive

```
$ singularity shell docker://centos:7
Docker image path: index.docker.io/library/centos:7
Cache folder set to /home/gmk/.singularity/docker
[1/1] |===================================| 100.0%
Creating container runtime...
Singularity: Invoking an interactive shell within container...

Singularity centos:7:~/git/singularity> cat /etc/redhat-release
CentOS Linux release 7.3.1611 (Core)
Singularity centos:7:~/git/singularity> exit
```

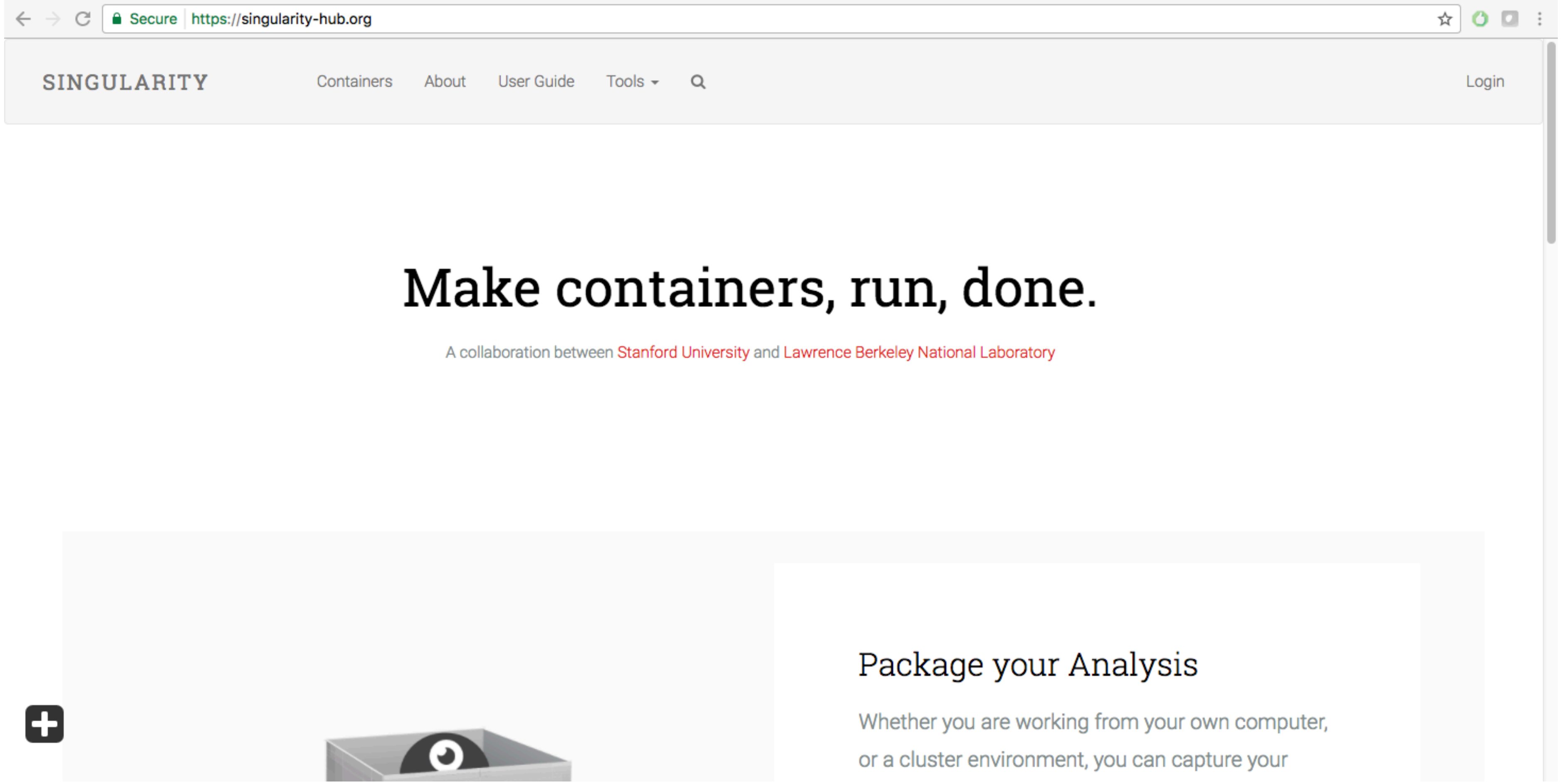
# Import

```
$ singularity create /tmp/debian.img
Creating a new image with a maximum size of 768MiB...
Executing image create helper
Formatting image with ext3 file system
Done.

$ singularity import /tmp/debian.img docker://debian:latest
Docker image path: index.docker.io/library/debian:latest
Cache folder set to /home/gmk/.singularity/docker
Importing: base Singularity environment
Importing:
/home/gmk/.singularity/docker/sha256:cd0a524342efac6edff50
0c17e625735…
Importing:
/home/gmk/.singularity/metadata/sha256:fe44851d529f465f9a
a107b32351c…
```

# Singularity Hub Pull

```
$ singularity pull shub://researchapps/quantum_state_diffusion
Progress |=====================================| 100.0%
Done. Container is at: ./researchapps-quantum_state_diffusion-master.img

$ singularity pull shub://507
Progress |=====================================| 100.0%
Done. Container is at: ./researchapps-quantum_state_diffusion-master.img

$ singularity exec researchapps-quantum_state_diffusion-master.img cat /etc/debian_version
stretch/sid
```

# Singularity: The Hub



Make containers, run, done.

A collaboration between Stanford University and Lawrence Berkeley National Laboratory

Package your Analysis

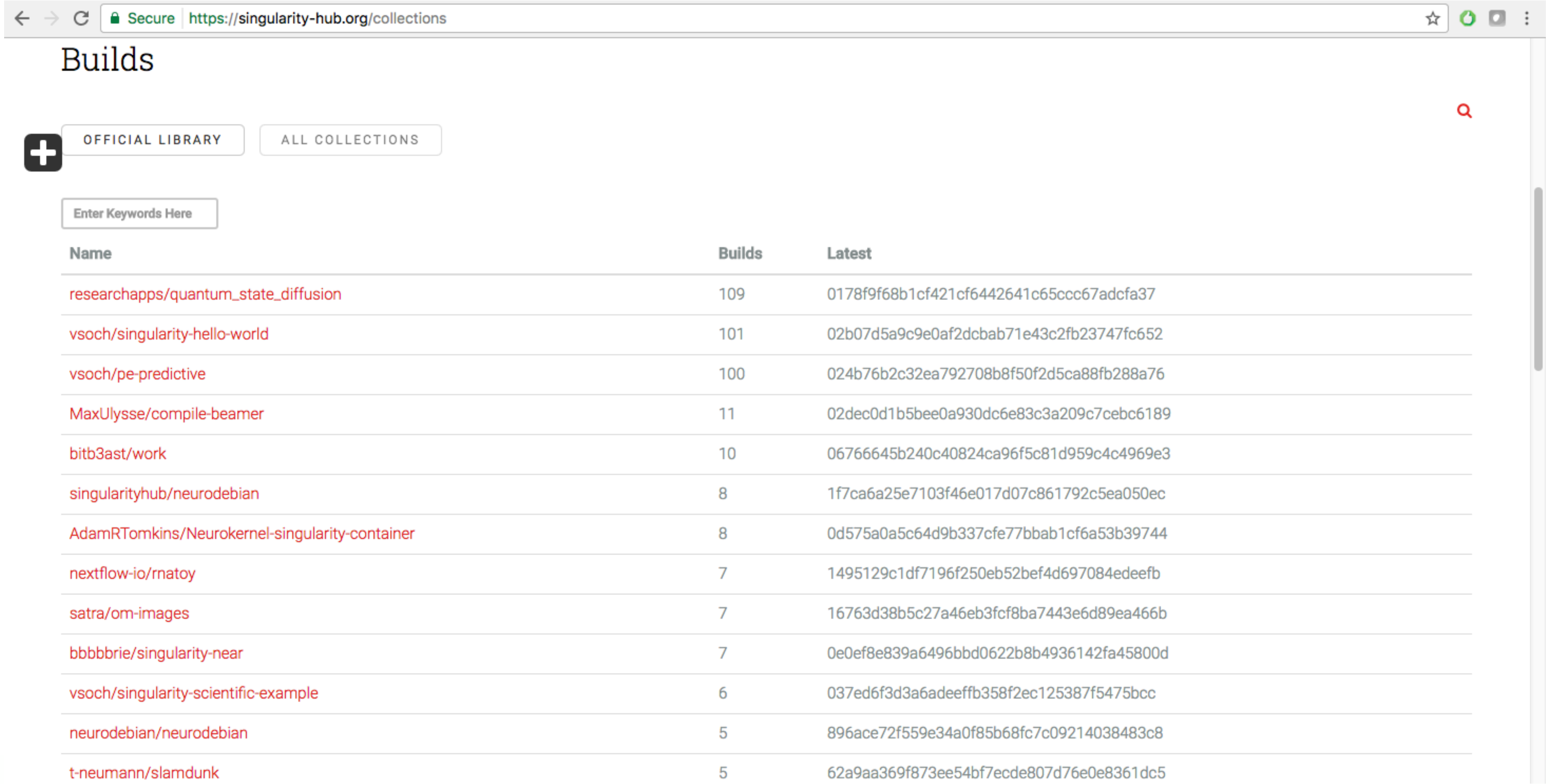
Whether you are working from your own computer, or a cluster environment, you can capture your

# Singularity Hub: The Workflow

- Have you heard of GitHub? If not, you should check it out! All the cool kids are doing it.
- In the root of your GitHub project, add a bootstrap definition file and call it “Singularity”
- Log into http://www.singularity-hub.org and setup your account
- Link your GitHub repository to Singularity Hub

Every ‘push’ to your GitHub repository will automatically trigger a container build, post it to Singularity Hub and make it available for citations!

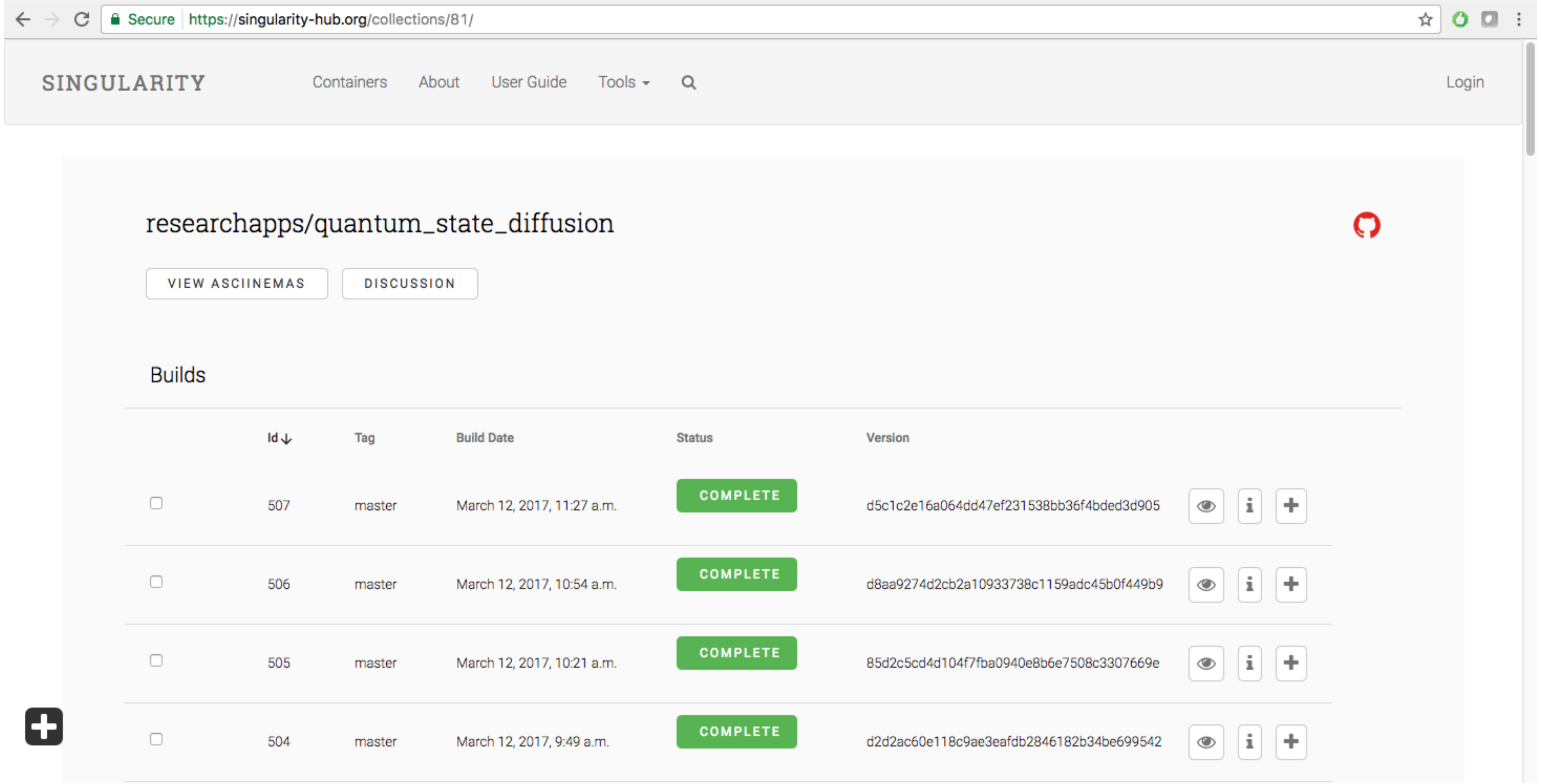
# Singularity Hub: Container Collections



## Builds

| Name | Builds | Latest |
|---|---|---|
| researchapps/quantum_state_diffusion | 109 | 0178f9f68b1cf421cf6442641c65ccc67adcfa37 |
| vsoch/singularity-hello-world | 101 | 02b07d5a9c9e0af2dcbab71e43c2fb23747fc652 |
| vsoch/pe-predictive | 100 | 024b76b2c32ea792708b8f50f2d5ca88fb288a76 |
| MaxUlysse/compile-beamer | 11 | 02dec0d1b5bee0a930dc6e83c3a209c7cebc6189 |
| bitb3ast/work | 10 | 06766645b240c40824ca96f5c81d959c4c4969e3 |
| singularityhub/neurodebian | 8 | 1f7ca6a25e7103f46e017d07c861792c5ea050ec |
| AdamRTomkins/Neurokernel-singularity-container | 8 | 0d575a0a5c64d9b337cfe77bbab1cf6a53b39744 |
| nextflow-io/rnatoy | 7 | 1495129c1df7196f250eb52bef4d697084edeefb |
| satra/om-images | 7 | 16763d38b5c27a46eb3fcf8ba7443e6d89ea466b |
| bbbbbrie/singularity-near | 7 | 0e0ef8e839a6496bbd0622b8b4936142fa45800d |
| vsoch/singularity-scientific-example | 6 | 037ed6f3d3a6adeeffb358f2ec125387f5475bcc |
| neurodebian/neurodebian | 5 | 896ace72f559e34a0f85b68fc7c09214038483c8 |
| t-neumann/slamdunk | 5 | 62a9aa369f873ee54bf7ecde807d76e0e8361dc5 |

# Singularity Hub: Container Builds

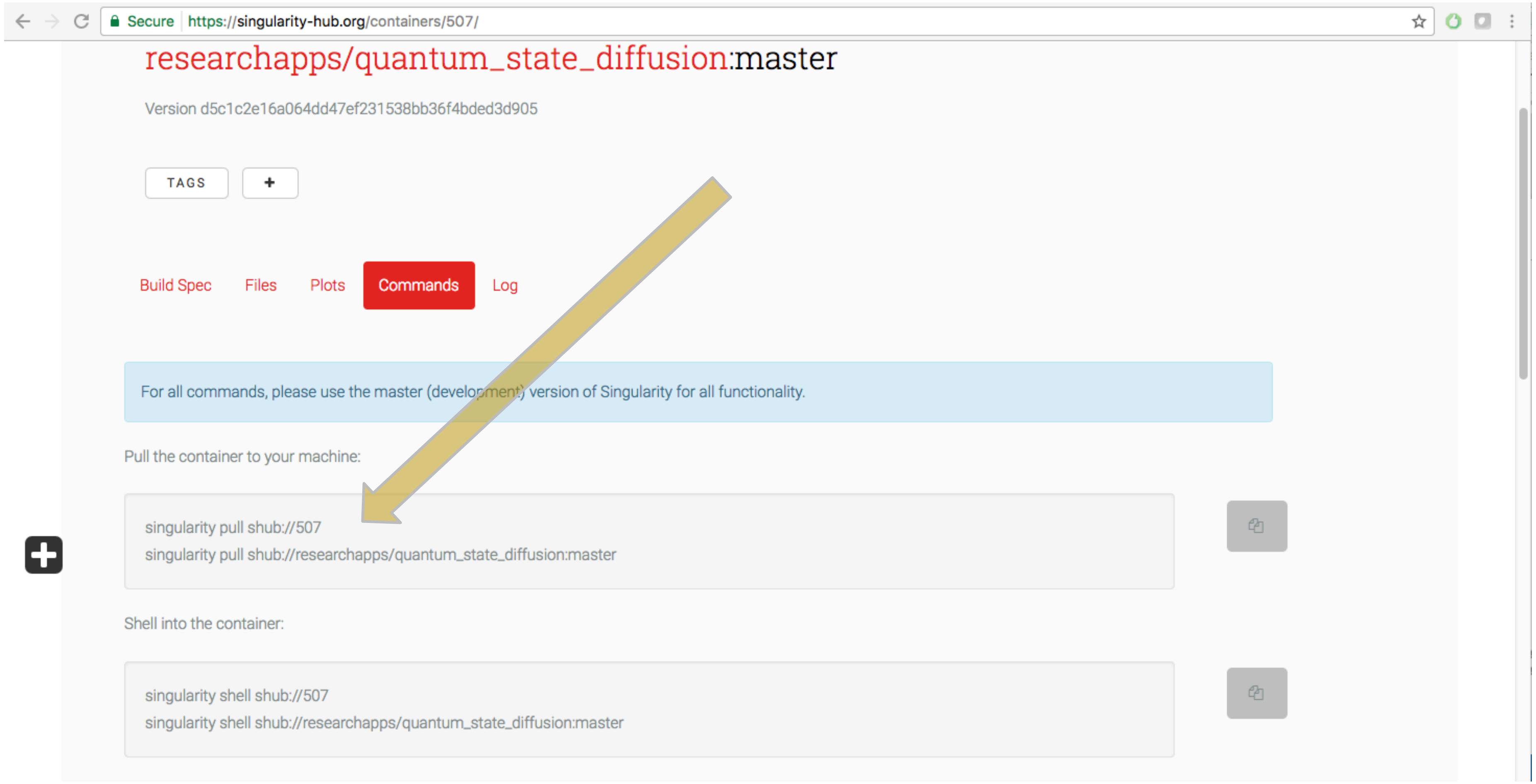


researchapps/quantum_state_diffusion

Builds

| | Id ↓ | Tag | Build Date | Status | Version |
|---|---|---|---|---|---|
| □ | 507 | master | March 12, 2017, 11:27 a.m. | COMPLETE | d5c1c2e16a064dd47ef231538bb36f4bded3d905 |
| □ | 506 | master | March 12, 2017, 10:54 a.m. | COMPLETE | d8aa9274d2cb2a10933738c1159adc45b0f449b9 |
| □ | 505 | master | March 12, 2017, 10:21 a.m. | COMPLETE | 85d2c5cd4d104f7fba0940e8b6e7508c3307669e |
| □ | 504 | master | March 12, 2017, 9:49 a.m. | COMPLETE | d2d2ac60e118c9ae3eafdb2846182b34be699542 |

# Singularity Hub: Container Commands



researchapps/quantum_state_diffusion:master

Version d5c1c2e16a064dd47ef231538bb36f4bded3d905

For all commands, please use the master (development) version of Singularity for all functionality.

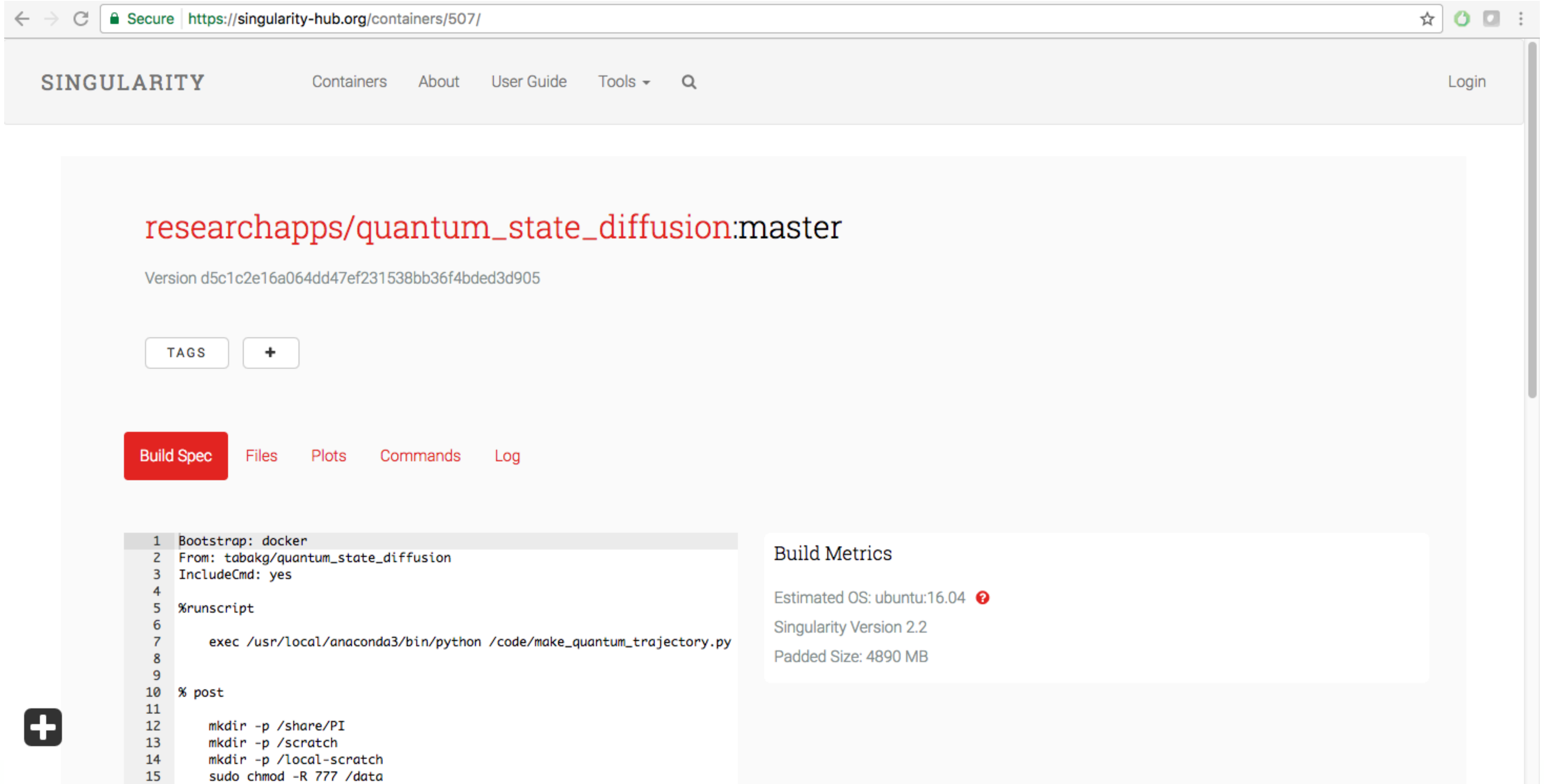
Pull the container to your machine:

```
singularity pull shub://507
singularity pull shub://researchapps/quantum_state_diffusion:master
```

Shell into the container:

```
singularity shell shub://507
singularity shell shub://researchapps/quantum_state_diffusion:master
```

# Singularity Hub: Container Bootstrap



researchapps/quantum_state_diffusion:master

Version d5c1c2e16a064dd47ef231538bb36f4bded3d905

```
Bootstrap: docker
From: tabakg/quantum_state_diffusion
IncludeCmd: yes

%runscript

    exec /usr/local/anaconda3/bin/python /code/make_quantum_trajectory.py


% post

    mkdir -p /share/PI
    mkdir -p /scratch
    mkdir -p /local-scratch
    sudo chmod -R 777 /data
```

Build Metrics

Estimated OS: ubuntu:16.04

Singularity Version 2.2

Padded Size: 4890 MB

# Singularity: Bootstrap Recipe Definitions

```
BootStrap: docker
From: ubuntu:latest

%post
   apt-get update
   apt-get -y install python3-pip locales
   pip3 install asciinema
   locale-gen en_US.UTF-8

%environment
   LANG=en_US.UTF-8
   LANGUAGE=en_US:en
   LC_ALL=en_US.UTF-8
   export LANG LANGUAGE LC_ALL

%runscript
   exec asciinema "$@"
```

Base operating system definition

Install programs and workflows

Setup environment

How to “run” container

# RHEL / YUM Bootstrap Definition

```
BootStrap: yum
OSVersion: 7
MirrorURL: http://mirror.centos.org/centos-%{OSVERSION}/%{OSVERSION}/os/$basearch/
Include: yum

%post
   yum update
   yum –y install vim-minimal

%runscript
   echo “Hello world: ‘$*’”
```

# Debian / Ubuntu Bootstrap Definition

```
BootStrap: debootstrap
OSVersion: trusty
MirrorURL: http://us.archive.ubuntu.com/ubuntu/

%post
   sed -i 's/$/ universe/' /etc/apt/sources.list
   apt-get update
   apt-get –y –force-yes install vim

%runscript
   echo “Hello world: ‘$*’”
```

# Bootstrap Process

```
$ singularity create –F /tmp/debian.img
Creating a new image with a maximum size of 768MiB...
Executing image create helper
Formatting image with ext3 file system
Done.

$ sudo singularity bootstrap /tmp/debian.img debian.def
Bootstrap initialization
Checking bootstrap definition
Executing Prebootstrap module
Executing Bootstrap 'debootstrap' module
...
update-alternatives: using /usr/bin/vim.basic to provide
/usr/bin/ex (ex) in auto mode
update-alternatives: using /usr/bin/vim.basic to provide
/usr/bin/editor (editor) in auto mode
Processing triggers for libc-bin (2.19-0ubuntu6) ...
Done.
```

# Shell Usage

```
$ singularity shell /tmp/debian.img
Singularity: Invoking an interactive shell within container...

Singularity debian.img:~/git/singularity> apt-get --version
apt 1.0.9.8.4 for amd64 compiled on Dec 11 2016 09:48:19
Supported modules:
*Ver: Standard .deb
 Pkg:  Debian APT solver interface (Priority -1000)
*Pkg:  Debian dpkg interface (Priority 30)
 S.L: 'deb' Standard Debian binary tree
 S.L: 'deb-src' Standard Debian source tree
 Idx: EDSP scenario file
 Idx: Debian Source Index
 Idx: Debian Package Index
 Idx: Debian Translation Index
 Idx: Debian dpkg status file

Singularity debian.img:~/git/singularity> cat /etc/debian_version
8.7

Singularity debian.img:~/git/singularity> exit
```

# Exec Usage

```
$ singularity exec /tmp/debian.img cat /etc/debian_version
8.7

$ singularity exec /tmp/centos.img cat /etc/redhat-release
CentOS Linux release 7.3.1611 (Core)

$ singularity exec /tmp/debian.img python
/.singularity.d/actions/exec: 8: exec: python: not found

$ sudo singularity exec --writable /tmp/debian.img apt-get install python
…

$ singularity exec /tmp/debian.img python
Python 2.7.9 (default, Jun 29 2016, 13:08:31)
[GCC 4.9.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

# Run Usage

```
$ singularity run /tmp/debian.img
Hello World:

$ singularity run /tmp/debian.img Testing 123
Hello World: Testing 123

$ ls -l /tmp/debian.img
-rwxr-xr-x 1 gmk gmk 805306400 May  4 17:32 /tmp/debian.img
```

-rwxr-xr-x

```
$ /tmp/debian.img opt1 opt2
Hello World: opt1 opt2
```

# Singularity: Process Invocation

- The command `singularity` is invoked, and shell code evaluates the ’verb’ and options
- The shell code hands off to the backend binary bits via execv()
- Linux kernel namespaces are created depending on configuration and system requirements
- The Singularity image is checked, parsed and mounted in the ‘CLONE_NEWNS’ namespace
- Bind mount points, additional file systems, and hooks into the host operating system are setup
- Singularity calls execv() again to the appropriate processes within the container
- Singularity is no longer running! It has exec’ed itself out of existance!
- Container process runs in the foreground, same PID as original `singularity` process
- When contained processes exit, all namespaces collapse leaving a clean system

# Singularity: The power of the sys-admin!

- Singularity configuration file controls what users can and can not do
- Configuration file must be root owned for it to be trusted (or it fails)
- Enforced system specific limitations, rules and boundaries
- Can specify system specific file system bind points
- Allows or disallows devices to be available within the container
- Controls the user’s ability to specify user requested bind points
- Paths, session directories, etc.. all controlled via configuration

# Singularity: Security Model

User contexts are always maintained when the container is launched.

When launched by any user, the programs inside the container will all be running as that user. Any escalation pathways inside the container are blocked. Thus…

If you want to be root inside the container,

you must first be root outside of the container!

# Conservation of Privilege

```
$ whoami
gmk
$ singularity shell /tmp/debian.img
Singularity: Invoking an interactive shell within container…

Singularity debian.img:~/git/singularity> whoami
gmk

Singularity debian.img:~/git/singularity> sudo whoami
sudo: effective uid is not 0, is /usr/bin/sudo on a file system with
the 'nosuid' option set or an NFS file system without root
privileges?

Singularity debian.img:~/git/singularity> ls -l /usr/bin/sudo
-rwsr-xr-x. 1 root root 136808 Aug 17 13:20 /usr/bin/sudo

Singularity debian.img:~/git/singularity> exit

$ sudo singularity exec /tmp/debian.img whoami
root
```

# Singularity: The Blurred Lines

- As mentioned, user’s contexts are strictly maintained and enforced
- This means we can safely blur the line between container and host
- Host/node resources can be just as tangible from within the container as outside
- This includes devices, file systems and paths, networks, X11, etc.

This allows containers to run appropriately on HPC resources!

# Welcome to the Matrix, Neo.

```
$ singularity exec /tmp/debian.img whoami
$ singularity exec /tmp/debian.img pwd
$ singularity exec /tmp/debian.img ls –l
$ singularity exec /tmp/debian.img touch ~/test_file

$ singularity exec /tmp/debian.img ps auxf
$ singularity exec –pid /tmp/debian.img ps auxf

$ singularity shell –bind /usr:/usr_host /tmp/debian.img
$ singularity shell –contain /tmp/debian.img
$ singularity shell –home ~/virt_home /tmp/debian.img

$ singularity exec –cleanenv /tmp/debian.img env
$ SINGULARITYENV_HELLO=WORLD singularity exec –cleanenv  \
> /tmp/debian.img env

$ singularity exec /tmp/debian.img python my_program.py
$ cat my_program.py | singularity exec /tmp/debian.img python

$ mpirun singularity exec /tmp/my_container.img
/path/to/mpi_program
```

# MPI example

```
$ mpirun singularity exec /tmp/mycontainer.img \
> /usr/bin/mpi_ring
Process 0 sending 10 to 1, tag 201 (4 processes in ring)
Process 0 sent to 1
Process 0 decremented value: 9
Process 0 decremented value: 8
Process 0 decremented value: 7
Process 0 decremented value: 6
Process 0 decremented value: 5
Process 0 decremented value: 4
Process 0 decremented value: 3
Process 0 decremented value: 2
Process 0 decremented value: 1
Process 0 decremented value: 0
Process 0 exiting
Process 1 exiting
Process 2 exiting
Process 3 exiting
```

# OpenMPI (from GitHub)

```
BootStrap: yum
OSVersion: 7
MirrorURL: http://mirror.centos.org/centos-%{OSVERSION}/%{OSVERSION}/os/$basearch/
Include: yum

%post
    echo "Installing Development Tools YUM group"
    yum -y groupinstall "Development Tools"
    echo "Installing OpenMPI into container..."
    mkdir /tmp/git
    cd /tmp/git
    git clone https://github.com/open-mpi/ompi.git
    cd ompi
    ./autogen.pl
    ./configure --prefix=/usr/local
    make
    make install
    /usr/local/bin/mpicc examples/ring_c.c -o /usr/bin/mpi_ring
    cd /
    rm -rf /tmp/git
    exit 0
```

# OpenFoam

```
BootStrap: debootstrap
OSVersion: trusty
MirrorURL: http://archive.ubuntu.com/ubuntu/
Include: bash


%post
   apt-get -y install wget apt-transport-https
   sed -i 's/main/main restricted universe/g' /etc/apt/sources.list
   echo 'deb http://download.openfoam.org/ubuntu trusty main' >> /etc/apt/sources.list
   wget -O - http://dl.openfoam.org/gpg.key | apt-key add -
   apt-get update
   apt-get -y install openfoam4
   echo ". /opt/openfoam4/etc/bashrc" >> /environment
```

# Tensorflow

```
BootStrap: debootstrap
OSVersion: stable
MirrorURL: http://ftp.us.debian.org/debian/

%runscript
   exec /usr/bin/python

%post
   apt-get update
   apt-get -y install vim python-pip python-dev
   pip install --upgrade https://storage.googleapis.com/tensorflow/linux/cpu/tensorflow-0.10.0-cp27-none-linux_x86_64.whl

%test
   # This runs usually less then 30 minutes depending on your host type
   python -m tensorflow.models.image.mnist.convolutional
```

# Playing With Python Versions

```
$ python hello.py
Hello World - Python version is: 2.7.6

$ singularity exec docker://python:latest python hello.py
Docker image path: index.docker.io/library/python:latest
Cache folder set to /home/gmk/.singularity/docker
[8/8] |======================================| 100.0%
Creating container runtime...
Hello World - Python version is: 3.6.1

$ cat hello.py | singularity exec docker://python:2 python
Docker image path: index.docker.io/library/python:2
Cache folder set to /home/gmk/.singularity/docker
[3/3] |======================================| 100.0%
Creating container runtime...
Hello World - Python version is: 2.7.13
```

# Pulling a Python Container

```
$ singularity pull docker://python:latest
Initializing Singularity image subsystem
Opening image file: python-latest.img
Creating 1290MiB image
Binding image to loop
Creating file system within image
Image is done: python-latest.img
Docker image path: index.docker.io/library/python:latest
Cache folder set to /home/gmk/.singularity/docker
Importing: base Singularity environment
…
Done. Container is at: python-latest.img

$ ./python-latest.img hello.py
Hello World - Python version is: 3.6.1
```

# Intel Python

```
$ **singularity pull docker://intelpython/intelpython3_core**
Initializing Singularity image subsystem
Opening image file: intelpython3_core.img
Creating 3409MiB image
Binding image to loop
Creating file system within image
Image is done: intelpython3_core.img
Docker image path: index.docker.io/intelpython/intelpython3_core:latest
Cache folder set to /home/gmk/.singularity/docker
[5/5] |====================================| 100.0%
Importing: base Singularity environment
Importing:
/home/gmk/.singularity/docker/sha256:8ad8b3f87b378cfae583fef34e47a3c9203847d779961b7351cbf786af0bc09f.tar.gz
Importing:
/home/gmk/.singularity/docker/sha256:e04db1209ac41bd39089bd10dc2d8160f01c72535f5580b03174c547dd87dcb3.tar.gz
Importing:
/home/gmk/.singularity/docker/sha256:edc7ae7e687c963bd0d8815fe7c930f6b2ab4a4a08ba2d087618c7b75f31c9a0.tar.gz
Importing:
/home/gmk/.singularity/docker/sha256:4a7b3487193b243d25027fc902c16b765776a7d02e2487f00c1fa8bcf50dc03c.tar.gz
Importing:
/home/gmk/.singularity/docker/sha256:0a02d6fdc5d036b311e926ecf9787dfadf1e0f6109b404a92910317e56e08ba4.tar.gz
Importing:
/home/gmk/.singularity/metadata/sha256:f6d87c41bba10f4649f8daf686d723d315e7a63b2e399e7e8891615a0e13fb3c.tar.gz
Done. Container is at: intelpython3_core.img
```

# Intel Python (cont)

```
$ singularity exec intelpython3_core.img python --version
Python 3.5.3 :: Intel Corporation

$ singularity exec intelpython3_core.img python hello.py
Hello World - Python version is: 3.5.3

$ singularity exec intelpython2_core.img python --version
Python 2.7.13 :: Intel Corporation

$ singularity exec intelpython2_core.img python hello.py
Hello World - Python version is: 2.7.13
```

# Intel Python: Bootstrap Definition

```
BootStrap: debootstrap
OSVersion: trusty
MirrorURL: http://us.archive.ubuntu.com/ubuntu/

%setup
   cp l_python3_pu3_2017.3.052.tgz $SINGULARITY_ROOTFS/

%post
   cd /
   ln -sf /proc/mounts /etc/mtab
   tar xvzf l_python3_pu3_2017.3.052.tgz
   cd l_python3_pu3_2017.3.052
   sed -i -e 's/^ACCEPT_EULA=.*/ACCEPT_EULA=accept/' silent.cfg
   ./install.sh -s silent.cfg

%environment
   PATH=/opt/intel/intelpython3/bin
   LD_LIBRARY_PATH=/opt/intel/intelpython3/lib
   export PATH LD_LIBRARY_PATH

%runscript
   exec /opt/intel/intelpython3/bin/python "$@"
```

# Intel Python: Building the container

```
$ singularity create –size 6144 /tmp/intelpython.img
Initializing Singularity image subsystem
Opening image file: intelpython.img
Creating 6144MiB image
Binding image to loop
Creating file system within image
Image is done: /tmp/intelpython.img

$ sudo singularity bootstrap /tmp/intelpython.img intelpython.def
Sanitizing environment
Building from bootstrap definition recipe
Adding base Singularity environment to container
I: Retrieving Release
I: Retrieving Release.gpg
…
+ cd l_python3_pu3_2017.3.052
+ sed -i -e s/^ACCEPT_EULA=.*/ACCEPT_EULA=accept/ silent.cfg
+ ./install.sh -s silent.cfg
Adding environment to container
Adding runscript
Finalizing Singularity container
```

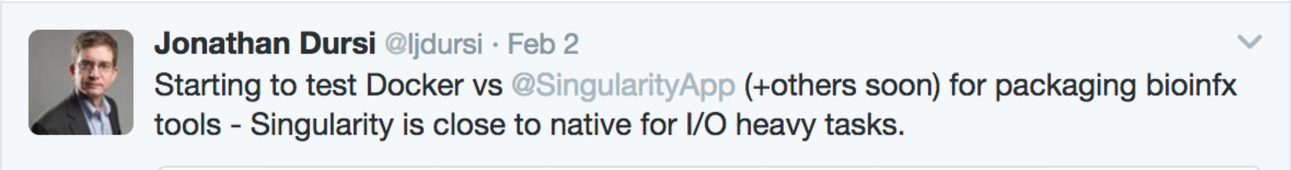
# Intel Python: Using the container

```
$ /tmp/intelpython.img --version
Python 3.5.3 :: Intel Corporation

$ /tmp/intelpython.img hello.py
Hello World - Python version is: 3.5.3

$ /tmp/intelpython.img
Python 3.5.3 |Intel Corporation| (default, Apr 27 2017, 18:08:47)
[GCC 4.8.2 20140120 (Red Hat 4.8.2-15)] on linux
Type "help", "copyright", "credits" or "license" for more information.
Intel(R) Distribution for Python is brought to you by Intel Corporation.
Please check out: https://software.intel.com/en-us/python-distribution
>>>
```

# Native GPU Support with Tensorflow

```
$ singularity run --nv docker://tensorflow/tensorflow:latest-gpu
Docker image path: index.docker.io/tensorflow/tensorflow:latest-gpu
Cache folder set to /home/gmk/.singularity/docker
[19/19] |================================| 100.0%
Creating container runtime...
[I 01:35:30.575 NotebookApp] Writing notebook server cookie secret to
/run/user/1000/jupyter/no…
[I 01:35:30.618 NotebookApp] Serving notebooks from local directory:
/home/gmk/git/singularity
[I 01:35:30.618 NotebookApp] 0 active kernels
[I 01:35:30.618 NotebookApp] The Jupyter Notebook is running at:
http://localhost:8888/?token=0dc…
[I 01:35:30.618 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice
to…
[W 01:35:30.618 NotebookApp] No web browser found: could not locate runnable browser.
[C 01:35:30.618 NotebookApp]

    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        http://localhost:8888/?token=0dc9bf2714ebc928562497aa17aad43b8844efd60a985209
```

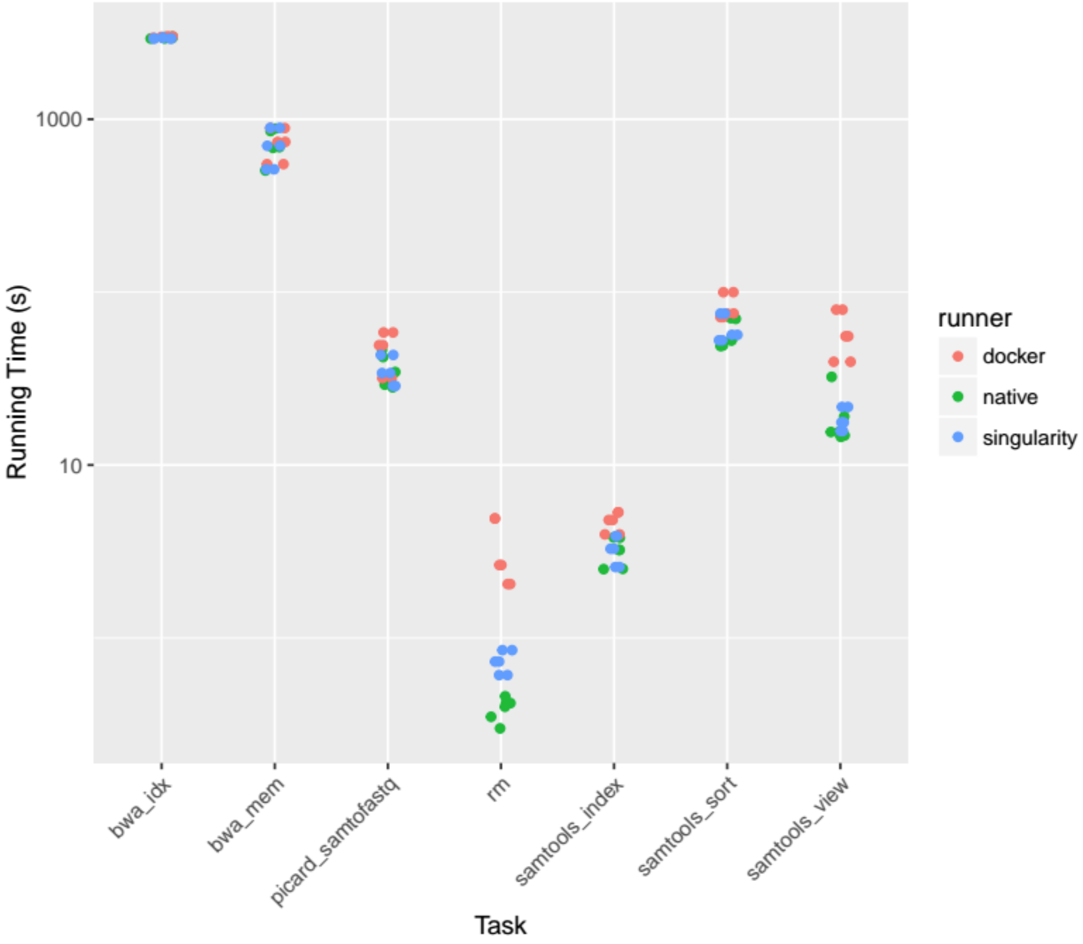
# Performance of Bio-apps



**Jonathan Dursi** @ljdursi · Feb 2

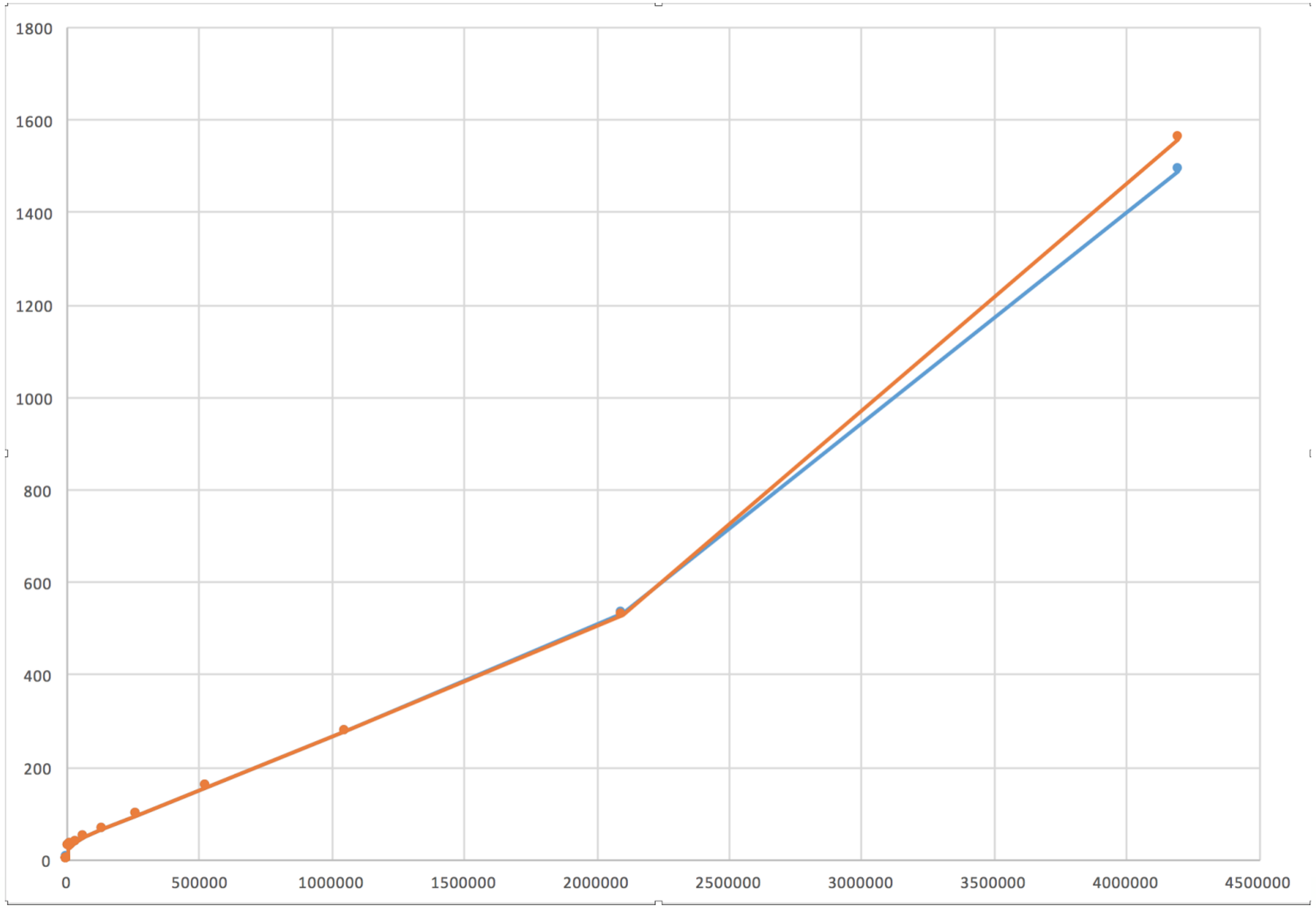
Starting to test Docker vs @SingularityApp (+others soon) for packaging bioinfx tools - Singularity is close to native for I/O heavy tasks.

- BWA Index and to a lesser extent BWA Mem are, in these cases, CPU/memory bound
- Samtools Index and sort are partially I/O bound (large quantities of small file)
- Samtools View is IOPS heavy

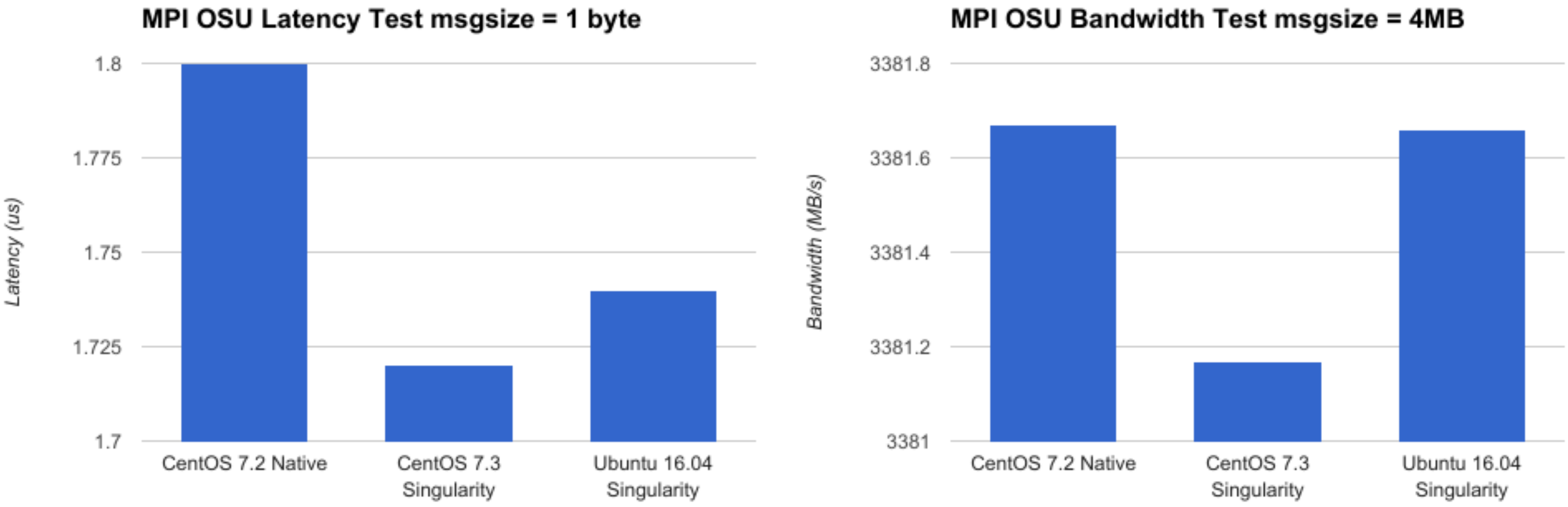


# Shared memory MPI latency between containers

- Same OS image and libraries inside and outside of container
- Tested with Open MPI (2.0)
- Both perform closely and only subtly diverage on large messages



# Containerized MPI Latency comparison



Open MPI 2.0.1 with OSU Micro Benchmarks 5.3.2

# Quote:

UNIVERSITY OF CAPE TOWN

IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

“This is a brilliant containerization technology which will gain traction very quickly”

# Use Case

BERKELEY LAB

LAWRENCE BERKELEY NATIONAL LABORATORY

- Lawrence Berkeley Labs and UC Berkeley use Singularity to support the ”long tail of science”
- LBNL is able to run a much wider breadth of jobs then what was possible before
- Non-traditional HPC scientists are able to easily run their workflows on our HPC resources
- Additionally, some workflows have been resurrected from their graves!
    - RedHat Linux 8.0 based system, installed late 2002
    - Software was written by a postdoc, and was let go after funding ended
    - Can’t be rewritten, scientists can’t recompile it on new systems, requires libraries no longer supported
    - The hardware lasted for 15 years, but alas it finally failed
    - Hard drive contents were converted to a Singularity image
    - This 15 year old workflow runs on the latest version of Centos now!

# Use Case

nextflow

- Nextflow is a workflow management language for data-driven computational pipelines
- Nextflow uses Singularity to deploy large-scale distributed scientific workflows
- Commonly used in genomics pipelines
- Supports both HPC cluster and cloud based resources in a portable manner
- Used by:
  - Center for Genomic Regulation (CRG)
  - Pasteur Institute (France)
  - SciLifeLab (Sweden)
  - Sanger Institute (UK)

# Use Case

NIH National Institutes of Health
*Turning Discovery Into Health*

- The NIH uses Singularity to provide programs like TensorFlow and OpenCV3 which are difficult or impossible to run with their current operating system
- With Singularity they can create ”portable reproducible data analysis pipelines”
- Singularity allows the NIH to provide this functionality to their users in a secure environment
- The system-admins found it easy and intuitive to use Singularity
- Some applications have been installed into Singularity containers and used as standalone programs via environment modules for the users

# Use Case

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

- Among standard HPC use cases…
- Researchers are using iPython Notebooks via JupyterHub
- iPhython JupyterHub kernels were deployed in Singularity containers
- Once the container is deployed via JupyterHub, the job runs within the container while maintaining access to local node resources
- This is a multi-user environment so Docker is a non-starter

# Use Case

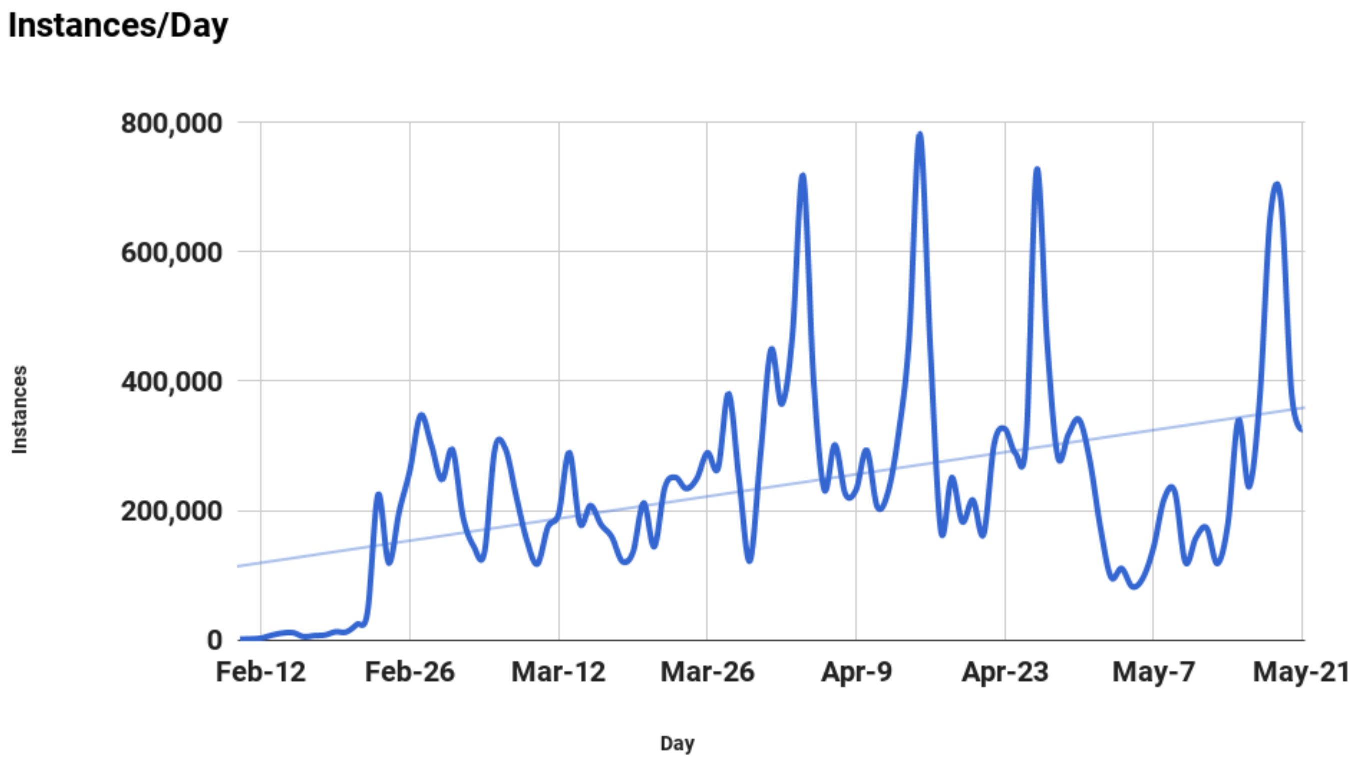
- ALICE jobs are packaged into Singularity containers
- Jobs are executed via Singularity through a modified SLURM script
- At any given moment in time, there are about 2000 Singularity containers active on the system

GSI Green Cube
6 stories tall
30,000 sqft
12 MegaWatts
PUE = 1.07 (world record)

# Use Case

Open Science Grid

- The OSG uses Singularity to provide a consistent runtime environment across heterogeneous resources worldwide
- Container images are distributed via CVMFS to all sites
- About half a million jobs are run through Singularity per day



# Singularity: Current Status

- Current version 2.3.1

- Paper has been published at PLOS: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0177459
- Growth has been hard to keep up with, massive uptake within world wide HPC community!
- SingularityWare, LLC. has been created to help bridge OSS to industry and fund development
- Singularity will remain (now and always) as open source BSD licensed software!
- Aside from having money for development, nothing else is changing

## I am hiring!

Are you an awesome person with a background in C, systems knowledge, security and/or kernel?

Join the Singularity team and send me your resume!

# Coming Soon: Data Containers

- Containers built specifically to support data portability and reproducibility
- Similar to data archives, but are container file systems and can be mounted on the host
- Designed for object store compatibility
- Optimized for local/direct IO access
- Works fantastic for consolidation of massive numbers of small files
- Integrates with Singularity natively on the host file system or from within a container
- Use cases: Any applications that utilize direct POSIX based IO and need compatibility with object stores, parallel storage, or RDM

# Coming Soon: RDM

- Research Data Management (RDM) will ensure accessibility to scientific data and runtimes
- Everybody in science is looking for an RDM solution
- It is becoming commonplace to have RDM listed as a requirement in grants and allocations
- We (SingularityWare) areworking on this problem with a very strategic partner
- With Singularity, you can deal with your container as you would any other scientific data
- The scientific runtime environment (container) then becomes a component of your research data management plan
- Use cases: almost all scientific institutions everywhere (academia, government, corporate)

# Coming Soon: Trusted Computing

- Trusted computing is environment, application and library verification
- Singularity’s single image containers are uniquely optimized for easy verification
- Verification could be as easy as a container file checksum
- Or as “paranoid” as a TPM verification
- Additionally passing the open file descriptor to the container runtime allows verification from within the container
- Use cases: highly secure environments, medical, government, financial, enterprise

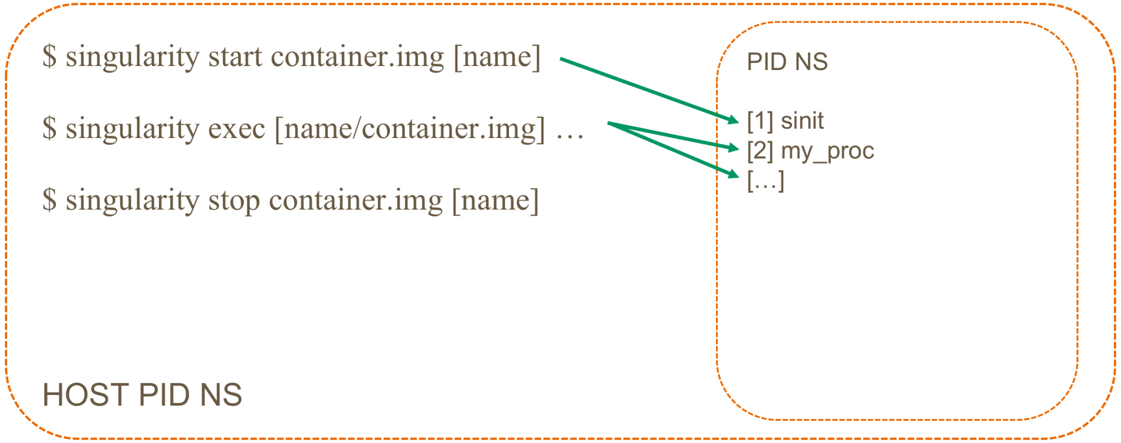
# Coming Soon: Optimized Image Format

- At present the ”Singularity Image” format is basically an embedded POSIX file system with a header offset
- This forces POSIX semantics everywhere, when it is not necessary and static image sizes
- Singularity v3 (maybe a year out) will support it’s own image file system, designed specifically for the agile, trusted, container usage
- This will allow us to do more things without privilege and support more features

# Coming Soon: Daemon Process Support

- Supporting background running daemon processes requires joining/leaving containers
- ”Singularity Init” will manage the container’s PID and network namespace
- Parent/child process relationships are always maintained
- New Singularity verbs: start, stop, status

```
$ singularity start container.img
$ singularity status container.img
Container ‘container.img’ is running
$ singularity exec container.img service start httpd
$ singularity exec container.img ps aux
```



```
$ singularity start container.img [name]
$ singularity exec [name/container.img] …
$ singularity stop container.img [name]
```

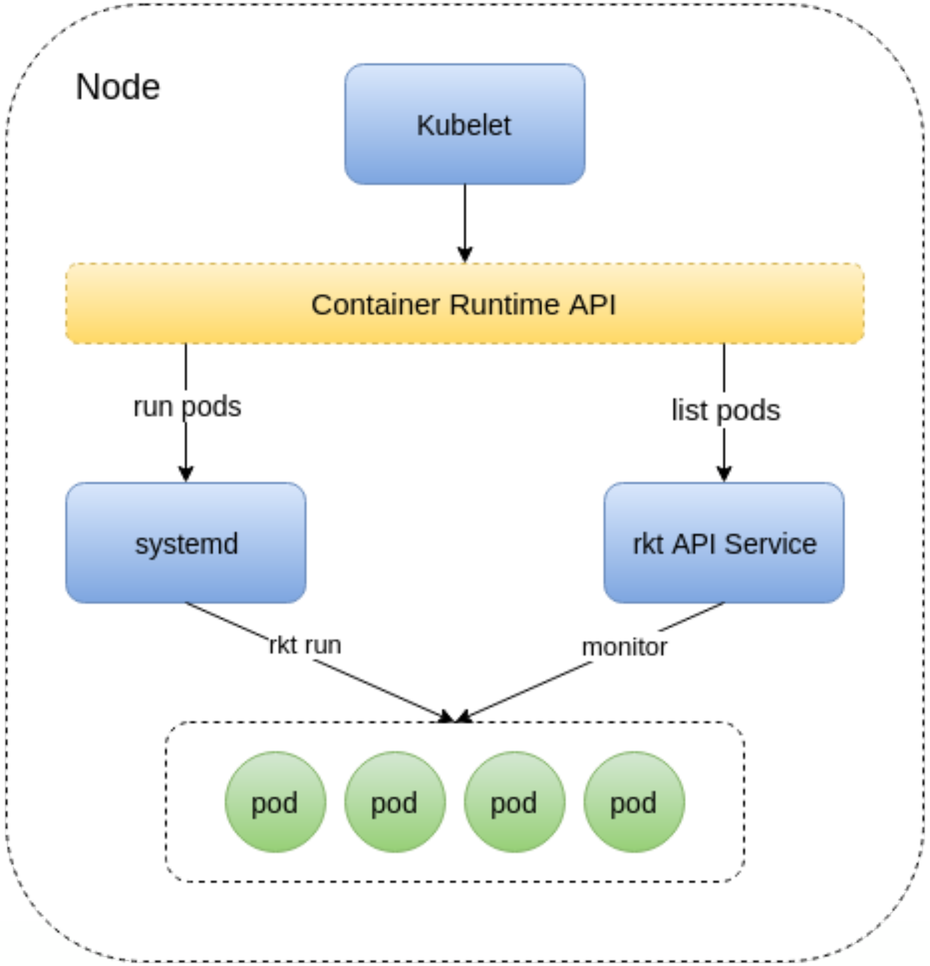
PID NS

[1] sinit
[2] my_proc
[…]

HOST PID NS

# Coming Soon: Orchestration Support

- Additional scheduling paradigms have been highly requested
- Namely: Kubernetes and Mesos
- We have funding by an outside entity (US .gov) for specifically this support
- Deploy Kubernetes services via Singularity with start/stop functionality
- Use cases: Cloud based science, HTC/serial computing



# Singularity: Contributors and Thanks!

BERKELEY LAB
Lawrence Berkeley National Laboratory

VE RI TAS HARVARD UNIVERSITY

GSI

GHENT UNIVERSITY

Lenovo

Berkeley
UNIVERSITY OF CALIFORNIA

INDIANA UNIVERSITY

Dartmouth

Stanford University

ONTROPOS

Institut Pasteur

NIH National Institutes of Health

nextflow

TACC
TEXAS ADVANCED COMPUTING CENTER

NVIDIA

amazon web services

UNIVERSITY OF Nebraska Lincoln

intel

SDSC
SAN DIEGO SUPERCOMPUTER CENTER

UNIVERSITY OF MICHIGAN

# Singularity

| | |
|---|---|
| Home page: | http://singularity.lbl.gov |
| GitHub: | https://github.com/singularityware/singularity |
| Twitter: | https://twitter.com/SingularityApp |
| Slack: | https://singularity-container.slack.com/ |

# Singularity

CONTAINERS FOR SCIENCE

Gregory M. Kurtzer
SingularityWare, LLC., Rstor, Inc.
Lawrence Berkeley National Laboratory
gmkurtzer@gmail.com