CONTAINERIZING HPC APPLICATIONS WITH SINGULARITY/APPTAINER

Yucheng Zhang, Senior Life Science Scientist

ACCESS | Advancing Innovation

NSF

PURDUE UNIVERSITY | Rosen Center for Advanced Computing

What to expect from this workshop?

Objectives

- What are containers and why should we use them?
- Singularity/Apptainer basics
- How to use apptainer to containerize your own applications
- Deployed containers on RCAC clusters

Containerizing HPC applications with Singularity/Apptainer

Containers

Purdue University | Rosen Center for Advanced Computing

The struggles from our users

- My application requires glibc 2.26, but the cluster only has 2.17, could you update glibc?
- My application run without issues on another cluster, but it does not work in this cluster.
- My code worked 3 months ago, but it does not work now. I need to repeat the analysis for my paper.
- Could you give me sudo privilege? I need it to install my application.
- I can install the package easily on my laptop, why the installation on the cluster failed?
- My whole group needs to run this application, could you help us install it?

Solution - Containers

- ❖ The arrival of modern shipping containers changed our transportation industry.
- ❖ Container is a standardized way to package items together into one shipment.

1. Standard packaging
2. Isolation and efficiency
3. Separation of concerns
4. Portable

Containers

A **container** is an abstraction for a set of technologies that aim to solve the problem of how to get software to run reliably when moved from one computing environment to another.

A container **image** is simply a file (or collection of files) saved on disk that stores everything you need to run a target application or applications.

Registry: a place to store (and share) container images.

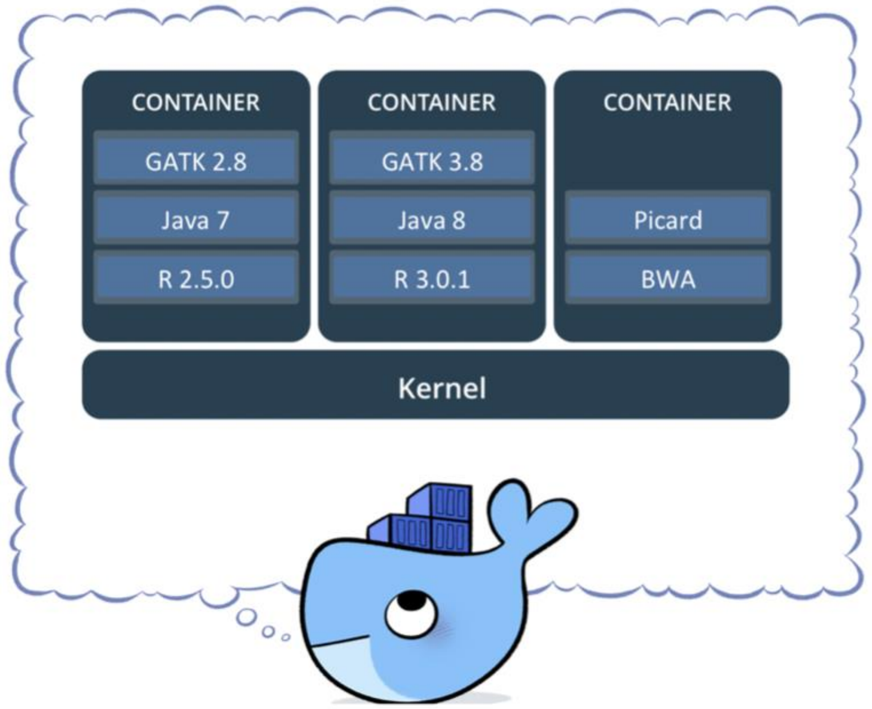
docker hub

GitHub Container Registry

BioContainers

Why use containers

- **Getting organized:** containers keep things organized by isolating programs and their dependencies inside containers.
- **Build once, run almost anywhere:** containers allow us to package up our complete software environment and ship it to numerous operating systems.
- **Reproducibility:** containers can ensure identical versions of apps, libraries, compliers, etc.



Installing applications into containers is much easier than installing them directly into our clusters.

Docker

The concept of containers emerged in 1970s, but they were not well known until the emergence of Docker containers in 2013.

Docker is an open source platform for building, deploying, and managing containerized applications.

docker

Some concerns about the security of Docker containers in HPC: Docker gives superuser privileges, but we do not want users to have full, unrestricted admin/ root access.

Adobe ARROW AT&T Blue Apron INTEGRATE

Lucent Health paloalto NETWORKS PathFactory Bitbucket_Full PayPal pipedrive

Segment snowflake spiceworks splunk> stripe

TARGET The Container Store UNIVERSITY OF CALGARY verizon VIRGINIA TECH

Yale

Singularity

❖ Singularity was developed in 2015 as an open-source project by researchers at Lawrence Berkeley National Laboratory led by Gregory Kurtzer.

❖ Singularity is emerging as the containerization framework of choice in HPC environments.

1. Enable researchers to package entire scientific workflows, libraries, and even data.
2. **Can use docker images.**
3. **Does not require root privileges to run.**

apptainer

- ❖ Singularity was renamed to Apptainer and hosted by the Linux Foundation.
- ❖ Apptainer uses the command apptainer to replace the previous command singularity.
- ❖ The singularity command on RCAC clusters also works, because it is alias for the command apptainer.
- ❖ The variable prefixes are APPTAINER_ and APPTAINERENV_ instead of the previous SINGULARITY_ and SINGULARITYENV_.

```
$ singularity --version
   3.8.5-2.el8
```

- Anvil
- Bell
- Brown
- Workbench

```
$ singularity --version
   apptainer version 1.1.6-1
```

- Negishi
- Gilbreth
- Scholar

Containerizing HPC applications with Singularity/Apptainer

Singularity/apptainer basics

Singularity/apptainer pull

Download or build a container from a given URI.

singularity/apptainer pull [output file] URI

Example: **singularity pull** blast_2.13.0.sif **docker://staphb/blast:2.13.0**

custom name *URI*

Supported URIs include:

❖ **library**: Pull an image from the currently configured library

library://user/collection/image[:tag]

❖ **docker**: Pull a Docker/OCI image from Docker Hub, or another OCI registry.

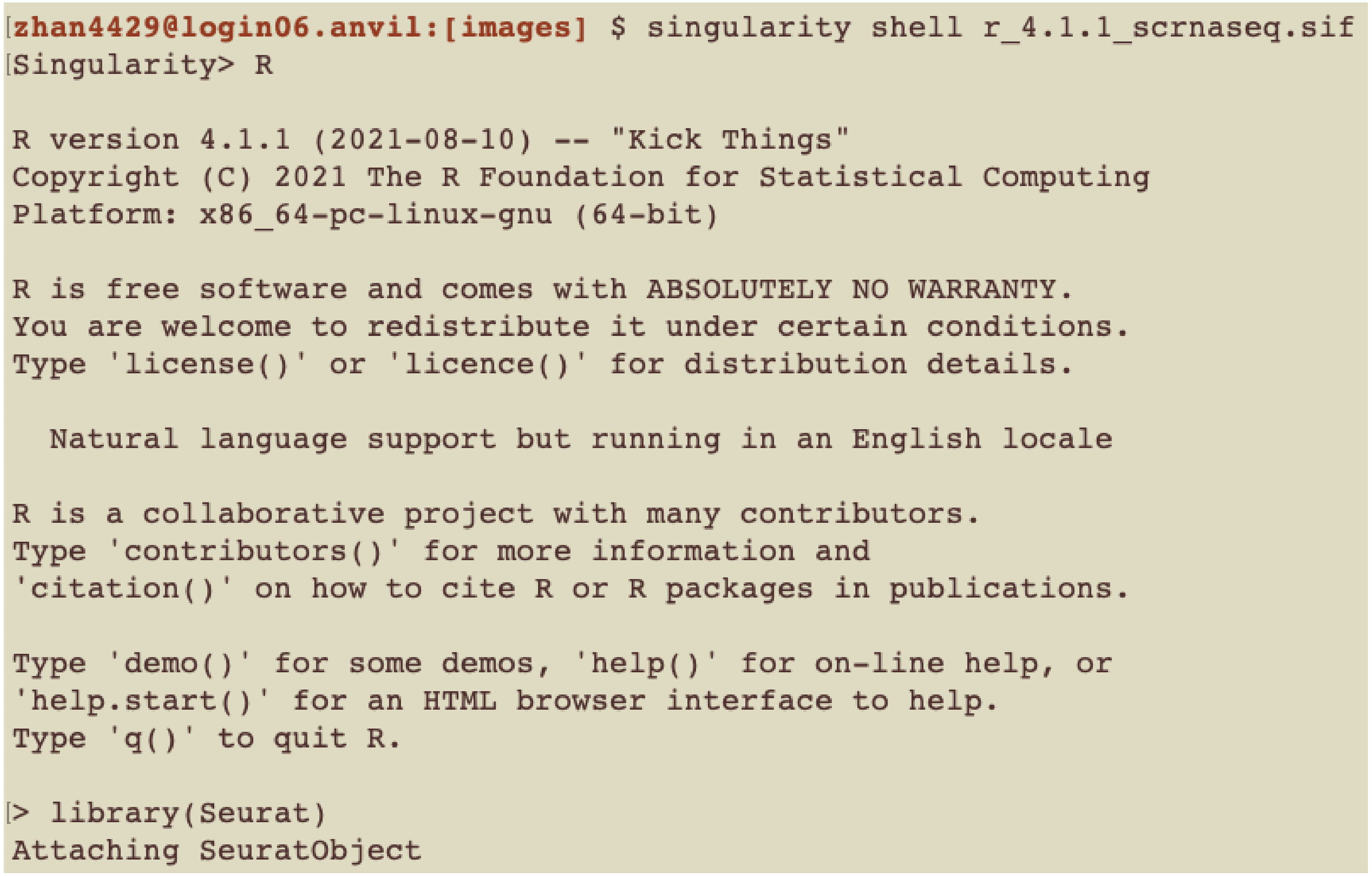
docker://user/image[:tag]

❖ **http, https**: Pull an image using the http(s?) protocol

https://depot.galaxyproject.org/singularity/hisat2%3A2.2.1--he1b5a44_2

Singularity/apptainer shell

Users can go inside the container to run interactive commands



```
[zhan4429@login06.anvil:[images] $ singularity shell r_4.1.1_scrnaseq.sif
[Singularity> R

R version 4.1.1 (2021-08-10) -- "Kick Things"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[> library(Seurat)
Attaching SeuratObject
```

Singularity/apptainer exec

A container may contain many executables/scripts.

singularity/apptainer exec can be used to select which executable/script to run.

singularity/apptainer exec image.sif command

For example:

```
singularity exec blast.2.13.0.sif blastn \
      -query nucleotide.fasta \
      -db nt -out blastn.out

apptainer exec blast.2.13.0.sif blastp \
     -query protein.fasta \
     -db nr -out blastp.out
```

Singularity build

Build using a singularity definition file

```
sudo singularity build image.sif definition.def
```

If you really need write access to a container you can use a writable sandbox.

```
sudo singularity build --sandbox container-sand definition.def
sudo singularity shell --writable container-sand
singularity> apt-get update
singularity> apt-get install –y packageName
sudo singularity build container.sif container-sand
```

apptainer build

Regular users don't have need sudo privilege to build containers.

```
$ sudo singularity build
```

```
$ singularity --version
3.8.5-2.el8
```

- Anvil
- Bell
- Brown
- Workbench

```
$ singularity build
$ apptainer build
```

```
$ singularity --version
apptainer version 1.1.6-1
```

- Negishi
- Gilbreth
- Scholar

Definition files

A **definition** file, or **def** file, is a recipe to build a container image with singularity. It is divided into two parts:

1. **Header**: the Header describes the core operating system to build within the container.
   - Bootstrap
   - From
2. **Section**: each section is defined by a **%** character followed by the name of the particular section. Different sections add different content or execute commands at different times during the build process.
   - help
   - setup
   - files
   - labels
   - **environment**
   - **post**
   - runscript
   - …

Header: Bootstrap and From

References the kind of base you want to use, and which registry stores the base image.

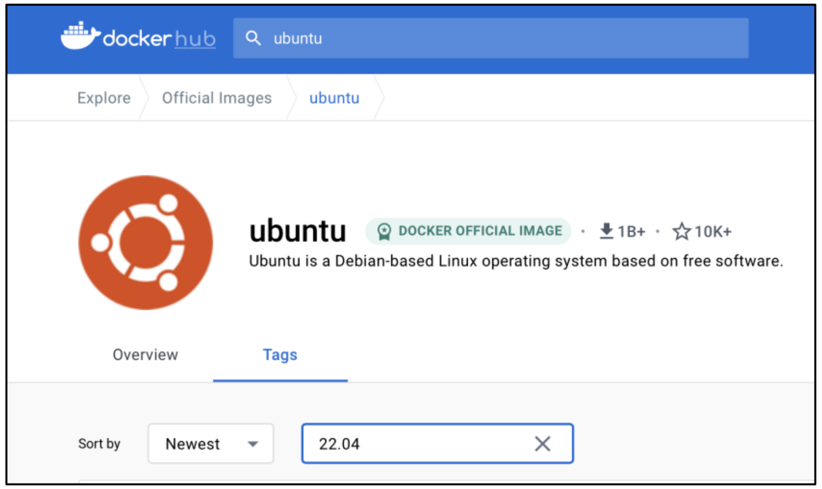
Images hosted on Docker Hub

```
Bootstrap: docker
From: ubuntu:22.04
```

```
Bootstrap: docker
From: nvidia/cuda:12.1.0-cudnn8-devel-ubuntu22.04
```

Images saved on your machine

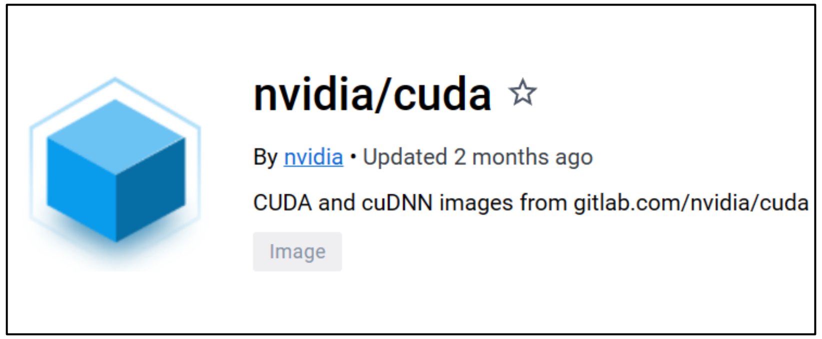
```
Bootstrap: localimage
From: /apps/biocontainers/images/mamba.sif
```



ubuntu

Ubuntu is a Debian-based Linux operating system based on free software.

22.04



nvidia/cuda

By nvidia • Updated 2 months ago

CUDA and cuDNN images from gitlab.com/nvidia/cuda

%label and %help

The **labels** section allows you to define metadata for your container.

The **help** section show the user-defined help for an image.

```
%labels
    Author "Yucheng Zhang <zhan4429@purdue.edu>“
    Version 4.3.1

%help
    This is an Ubuntu 20.04 container with R/4.3.1 and Rstudio/2023.06-2
```

```
$ singularity inspect  --labels <image path>
$ singularity run-help <image path>
```

%files and %environment

If you need to copy files from host into container, you do so in **%files**. Each line in **%files** is a pair of source and destination paths, where source is on your host, and destination is a path inside container.

%environment allows you to define environment variables for your container. These variables are available when you run the container, not during its build.

```
%files
    install.sh    /opt/install.sh
    example_data/      /opt/example_data
%environment
    export PATH=/opt/myapp/bin:$PATH
    export LD_LIBRARY_PATH=/opt/myapp/lib:$LD_LIBRARY_PATH
    export LC_ALL=C
```

%post

%post is where you can download files from the internet with tools like git and wget, install new software and libraries, write configuration files, create new directories, etc.

```
Bootstrap: docker
From: ubuntu:18.04

%post
# Update and install system libraries
apt-get -y update
apt-get -y install --no-install-recommends --no-install-suggests build-essential libssl-dev wget

# KALIGN 2.0.4
cd /opt  && mkdir kalign2
cd kalign2 && wget http://msa.sbc.su.se/downloads/kalign/current.tar.gz
tar -xvf current.tar.gz && ./configure && make

%environment
export PATH=/opt/kalign2:$PATH
```

%runscript

The **%runscript** section will be executed when the container is run. The runscript can be triggered with the **run** command, or simply by calling the container as though it were an executable.

```
%runscript
      exec python3 "$@"
```

```
%runscript
      rstudio "$@"
```

```
 %runscript
    cellphonedb "$@"
```

Inspect the runscript

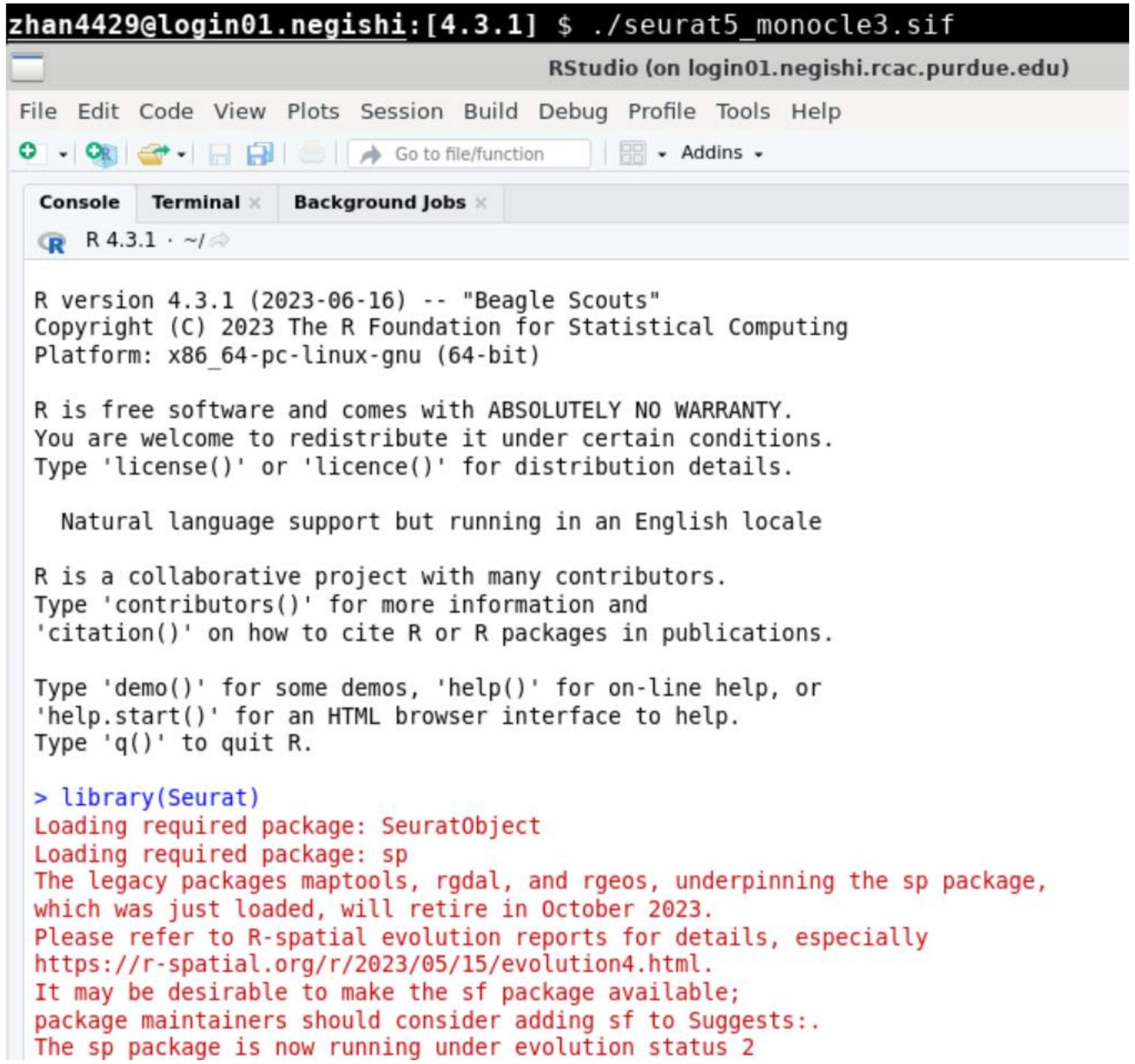
```
singularity inspect --runscript image.sif
singularity inspect -r image.sif
```

Execute runscript

```
## Using singularity run
singularity run image.sif
## Use image as exectuable
./image.sif
```

localimage: R/Rstudio container

```
Bootstrap: localimage
From: /apps/base_images/r_rstudio/r_4.3.1_rstduio_2023.06.sif

%post
## Install  dependencies for monocle3
Rscript -e "BiocManager::install(c('BiocGenerics', 'DelayedArray',\
                'DelayedMatrixStats',  'limma', 'lme4', 'S4Vectors',\
        'SingleCellExperiment', 'SummarizedExperiment', \
                        'batchelor', 'HDF5Array', 'terra', 'ggrastr'))"

## Install monocle3
Rscript -e "devtools::install_github('cole-trapnell-lab/monocle3')"

## Install Seurat5
Rscript -e "devtools::install_github('satijalab/seurat', 'seurat5', quiet = TRUE)"
```



```
zhan4429@login01.negishi:[4.3.1] $ ./seurat5_monocle3.sif
```

RStudio (on login01.negishi.rcac.purdue.edu)
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Console
Terminal
Background Jobs
R 4.3.1 · ~/

```
R version 4.3.1 (2023-06-16) -- "Beagle Scouts"
Copyright (C) 2023 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> library(Seurat)
Loading required package: SeuratObject
Loading required package: sp
The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
which was just loaded, will retire in October 2023.
Please refer to R-spatial evolution reports for details, especially
https://r-spatial.org/r/2023/05/15/evolution4.html.
It may be desirable to make the sf package available;
package maintainers should consider adding sf to Suggests:.
The sp package is now running under evolution status 2
```

localimage: mamba/conda

```
Bootstrap: localimage
From: /apps/base_images/mamba/mamba_1.4.2.sif

%labels
    Author "Yucheng Zhang <zhan4429@purdue.edu>"

%post
    mamba install -c bioconda fastqc trim-galore samtools \
            bowtie2 star subread hisat2 salmon \
            picard bedtools stringtie
```

Mamba can speed up your conda installs by 50-80%.

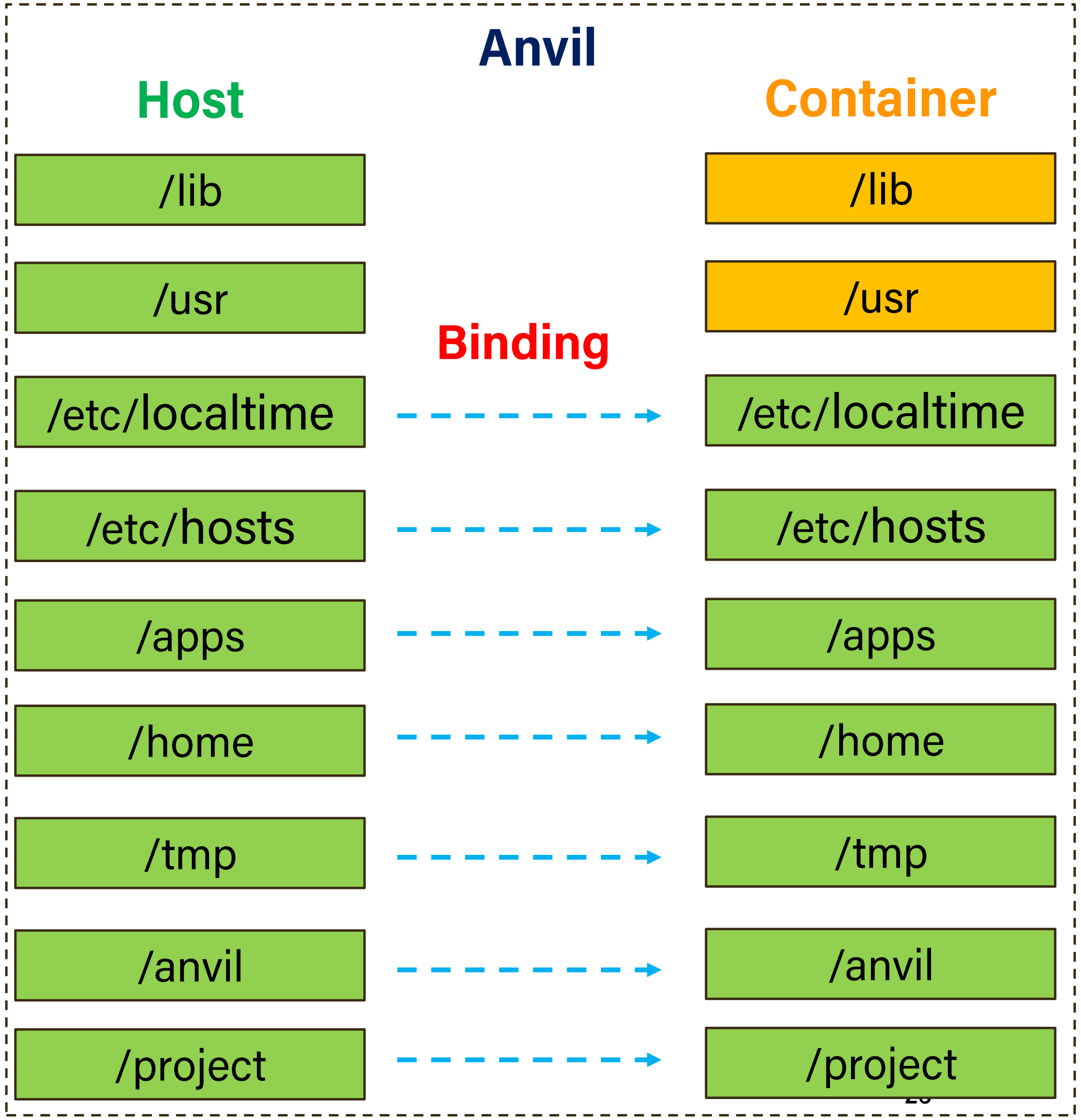
Bind host directories

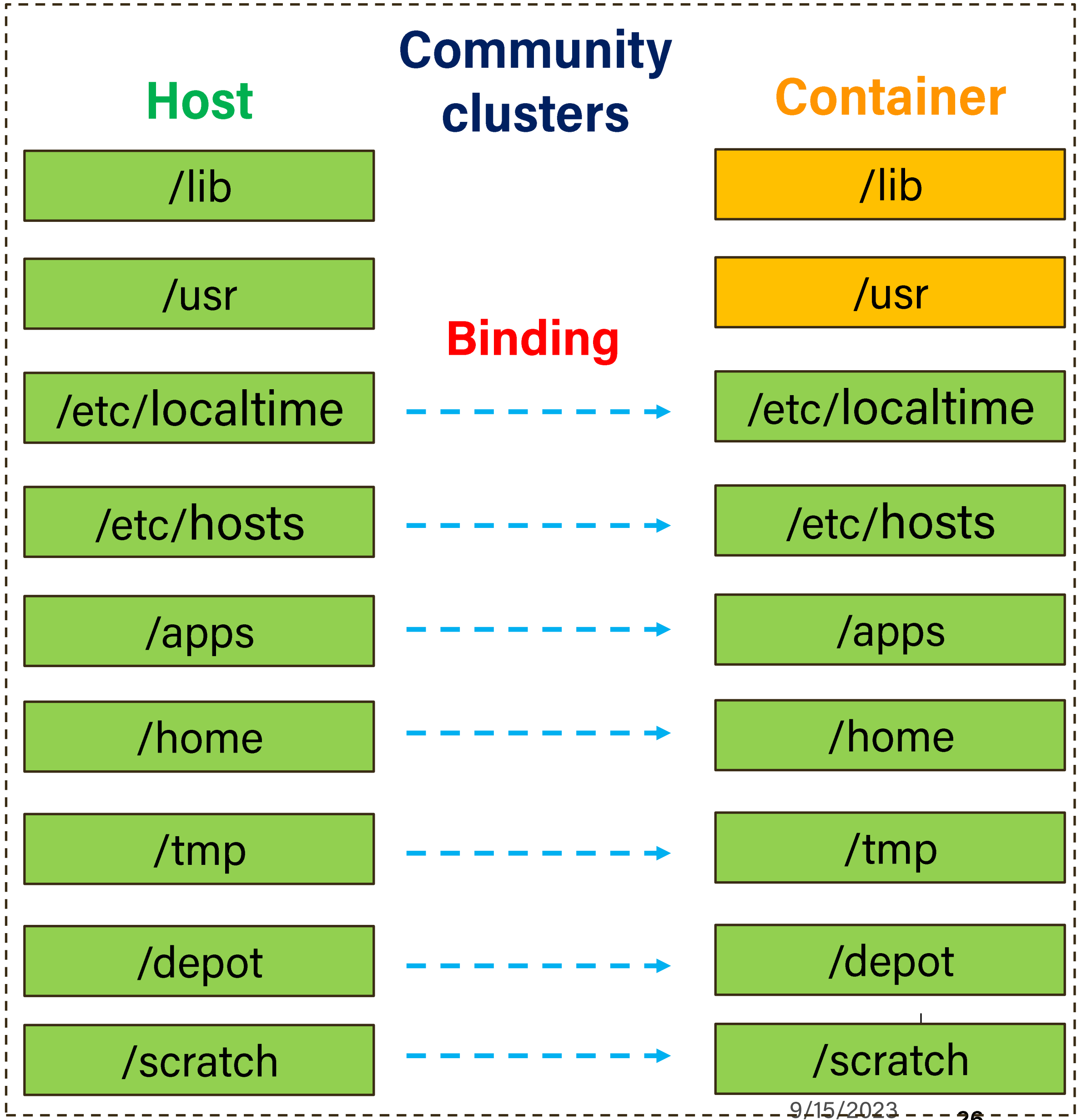
- Programs running inside a container will not have access to directories and files outside of your home and the current directory.
- Singularity allows you to map directories on your host system to directories within your container using bind mounts.

```
singularity/apptainer shell/run/exec --bind hostdir:containerdir image.sif
```

Singularity binds several directories into the container image automatically.
$HOME, **/tmp** and **$PWD** is the default list.
We also configured singularity to bind **/apps**, **/depot**, and **/scratch** on Purdue community clusters, to bind **/anvil**, and **/apps** on ACCESS Anvil.

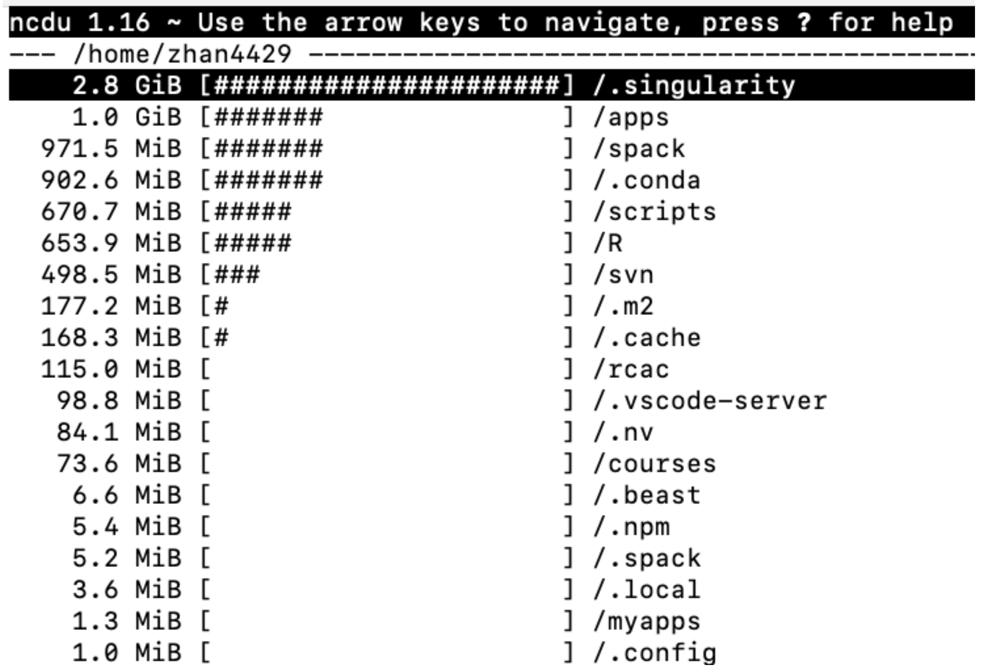
binding


Anvil
Host
Container
Binding
/lib
/lib
/usr
/usr
/etc/localtime
/etc/localtime
/etc/hosts
/etc/hosts
/apps
/apps
/home
/home
/tmp
/tmp
/anvil
/anvil
/project
/project


Community clusters
Host
Container
Binding
/lib
/lib
/usr
/usr
/etc/localtime
/etc/localtime
/etc/hosts
/etc/hosts
/apps
/apps
/home
/home
/tmp
/tmp
/depot
/depot
/scratch
/scratch

cache

```
$ ncdu $HOME
```



```
ncdu 1.16 ~ Use the arrow keys to navigate, press ? for help
--- /home/zhan4429 ----------------------------------------------
    2.8 GiB [#######################] /.singularity
    1.0 GiB [#######                ] /apps
  971.5 MiB [#######                ] /spack
  902.6 MiB [#######                ] /.conda
  670.7 MiB [#####                  ] /scripts
  653.9 MiB [#####                  ] /R
  498.5 MiB [###                    ] /svn
  177.2 MiB [#                      ] /.m2
  168.3 MiB [#                      ] /.cache
  115.0 MiB [                       ] /rcac
   98.8 MiB [                       ] /.vscode-server
   84.1 MiB [                       ] /.nv
   73.6 MiB [                       ] /courses
    6.6 MiB [                       ] /.beast
    5.4 MiB [                       ] /.npm
    5.2 MiB [                       ] /.spack
    3.6 MiB [                       ] /.local
    1.3 MiB [                       ] /myapps
    1.0 MiB [                       ] /.config
```

To mitigate this, users can either run the singularity pull command with argument --disable-cache or manually clean $HOME/.singularity/cache or $HOME/.apptainer/cache

```
singularity pull --disable-cache URI
```

Or specifying a custom cache directory

```
export SINGULARITY_CACHEDIR=/tmp/$USER
```

```
export APPTAINER_CACHEDIR=/tmp/$USER
```

GPU support

For many applications, CPU compute resources provide sufficient performance. However, for a certain class of applications, the massively parallel compute power offered by GPUs can speed up operations by orders of magnitude.

Run a container with GPU acceleration

For AMD GPUs (Bell and Negishi):

```
singularity/apptainer shell/run/exec --rocm myimage.sif [command] [argument]
```

For NVIDIA GPUs (Anvil, Gilbreth, and Scholar):

```
singularity/apptainer shell/run/exec --nv myimage.sif [command] [argument]
```

Multi-node MPI

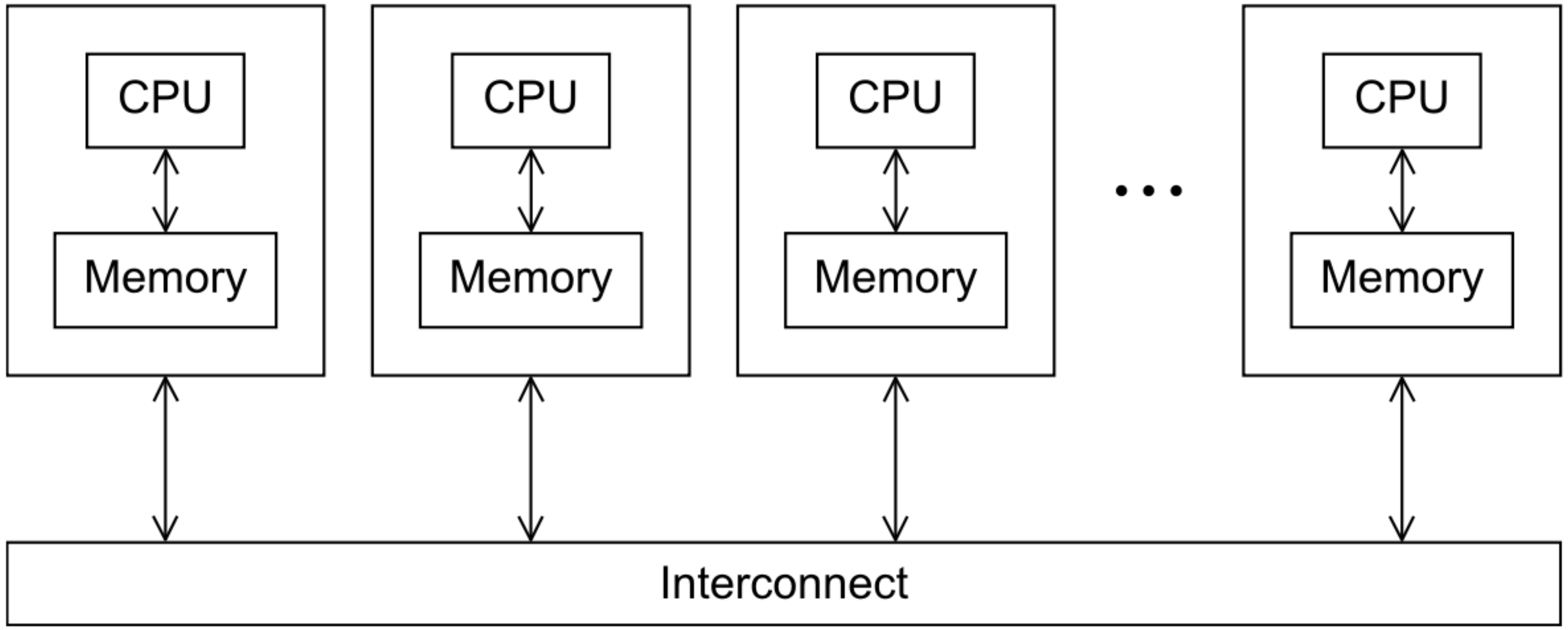
While singularity/apptainer has always supported MPI-enabled applications, it can be tricky to get them working on multiple nodes.

Two old ways (https://apptainer.org/docs/user/latest/mpi.html)**:**

1. the hybrid model
2. the bind model

The new way:

- ❖ PMI



An Introduction to Parallel Programming by Peter Pacheco and Matthew Malensek

Hybrid mode

This model is called hybrid because it requires both an MPI implementation on the host and another implementation in the container image.

The host MPI provides the **mpirun** or **mpiexec** command and start ranks or containers on computing nodes.

MPI in the container image is used to run the application.

Host and container MPI need to be “**compatible**” since they need to tightly interact with each other.

```
module load openmpi/XXX ## load the MPI module compatible with your container
mpirun -n $SLURM_NTASKS singularity exec mpi_container.sif mpi_program
```

Bind mode

Similar to the Hybrid mode, the Bind mode is to start the MPI application by calling the MPI launcher (e.g., mpirun) from the host.

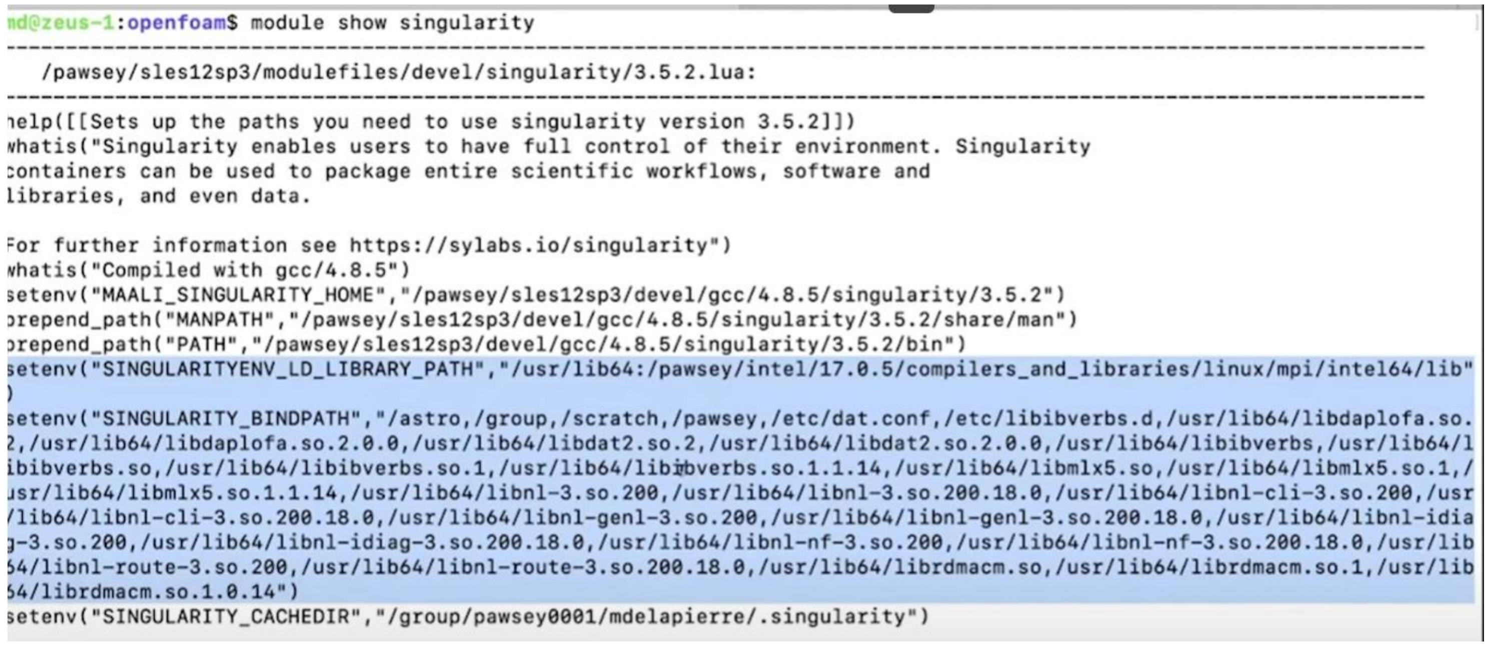
The main difference between the hybrid and bind mode is the fact that with the bind mode, the container usually does not include any MPI implementation.

This means that **singularity/apptainer needs to mount/bind the MPI and high-speed interconnect libraries available on the host into the container**.

Technically this requires two steps:

1. Know where the MPI implementation on the host is installed.
2. Mount/bind it into the container in a location where the system will be able to find libraries and binaries.

This mode sounds simple, but binding the complete paths is difficult and inconvenient.



```
nd@zeus-1:openfoam$ module show singularity
-------------------------------------------------------------------------------------------------------------------
   /pawsey/sles12sp3/modulefiles/devel/singularity/3.5.2.lua:
-------------------------------------------------------------------------------------------------------------------
help([[Sets up the paths you need to use singularity version 3.5.2]])
whatis("Singularity enables users to have full control of their environment. Singularity
containers can be used to package entire scientific workflows, software and
libraries, and even data.

For further information see https://sylabs.io/singularity")
whatis("Compiled with gcc/4.8.5")
setenv("MAALI_SINGULARITY_HOME","/pawsey/sles12sp3/devel/gcc/4.8.5/singularity/3.5.2")
prepend_path("MANPATH","/pawsey/sles12sp3/devel/gcc/4.8.5/singularity/3.5.2/share/man")
prepend_path("PATH","/pawsey/sles12sp3/devel/gcc/4.8.5/singularity/3.5.2/bin")
setenv("SINGULARITYENV_LD_LIBRARY_PATH","/usr/lib64:/pawsey/intel/17.0.5/compilers_and_libraries/linux/mpi/intel64/lib"
)
setenv("SINGULARITY_BINDPATH","/astro,/group,/scratch,/pawsey,/etc/dat.conf,/etc/libibverbs.d,/usr/lib64/libdaplofa.so.
2,/usr/lib64/libdaplofa.so.2.0.0,/usr/lib64/libdat2.so.2,/usr/lib64/libdat2.so.2.0.0,/usr/lib64/libibverbs,/usr/lib64/l
ibibverbs.so,/usr/lib64/libibverbs.so.1,/usr/lib64/libibverbs.so.1.1.14,/usr/lib64/libmlx5.so,/usr/lib64/libmlx5.so.1,/
usr/lib64/libmlx5.so.1.1.14,/usr/lib64/libnl-3.so.200,/usr/lib64/libnl-3.so.200.18.0,/usr/lib64/libnl-cli-3.so.200,/usr
/lib64/libnl-cli-3.so.200.18.0,/usr/lib64/libnl-genl-3.so.200,/usr/lib64/libnl-genl-3.so.200.18.0,/usr/lib64/libnl-idia
g-3.so.200,/usr/lib64/libnl-idiag-3.so.200.18.0,/usr/lib64/libnl-nf-3.so.200,/usr/lib64/libnl-nf-3.so.200.18.0,/usr/lib
64/libnl-route-3.so.200,/usr/lib64/libnl-route-3.so.200.18.0,/usr/lib64/librdmacm.so,/usr/lib64/librdmacm.so.1,/usr/lib
64/librdmacm.so.1.0.14")
setenv("SINGULARITY_CACHEDIR","/group/pawsey0001/mdelapierre/.singularity")
```

Pawsey Supercomputing Research Centre
https://www.youtube.com/watch?v=sV4JmbWUirc&t=4s

srun and PMI

The basic idea behind this approach is to put the entire MPI framework into a container along with the MPI application and then to use a tool (e.g. slurm) that implements one of the *PMI* standards to launch the MPI jobs.

MPI inside container need to be built with the same PMI standard support.

On RCAC clusters, **PMI-2** is recommended.

Home / CIQ Blog / Apptainer / Singularity

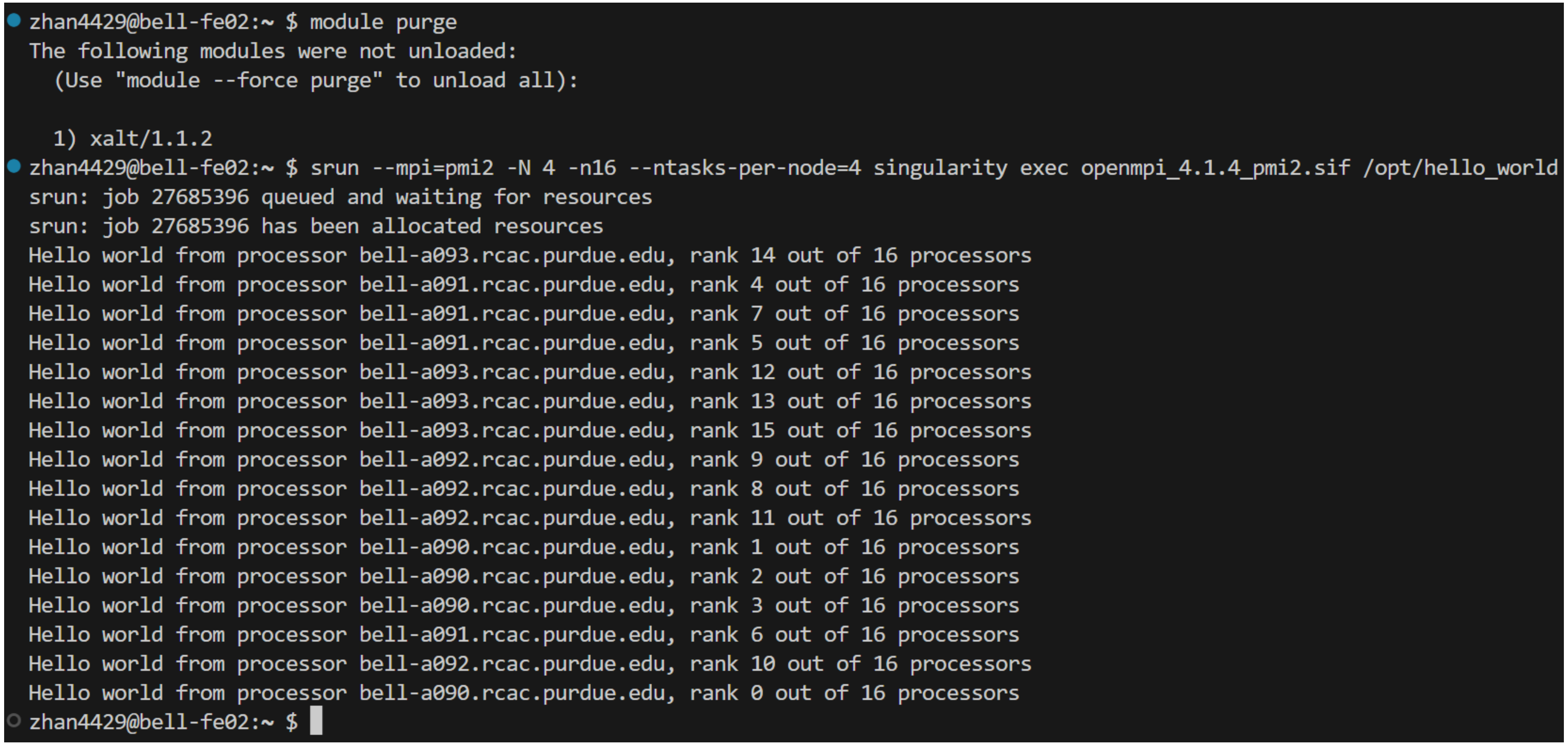
A New Approach to MPI in Apptainer

Dave Godlove • June 27, 2023

https://ciq.com/blog/a-new-approach-to-mpi-in-apptainer/

srun and PMI: usage

module purge # mainly purpose is to unload host mpi modules
srun --mpi=pmi2 singularity/apptainer exec mpi_image.sif mpi_application



```
zhan4429@bell-fe02:~ $ module purge
The following modules were not unloaded:
  (Use "module --force purge" to unload all):

  1) xalt/1.1.2
zhan4429@bell-fe02:~ $ srun --mpi=pmi2 -N 4 -n16 --ntasks-per-node=4 singularity exec openmpi_4.1.4_pmi2.sif /opt/hello_world
srun: job 27685396 queued and waiting for resources
srun: job 27685396 has been allocated resources
Hello world from processor bell-a093.rcac.purdue.edu, rank 14 out of 16 processors
Hello world from processor bell-a091.rcac.purdue.edu, rank 4 out of 16 processors
Hello world from processor bell-a091.rcac.purdue.edu, rank 7 out of 16 processors
Hello world from processor bell-a091.rcac.purdue.edu, rank 5 out of 16 processors
Hello world from processor bell-a093.rcac.purdue.edu, rank 12 out of 16 processors
Hello world from processor bell-a093.rcac.purdue.edu, rank 13 out of 16 processors
Hello world from processor bell-a093.rcac.purdue.edu, rank 15 out of 16 processors
Hello world from processor bell-a092.rcac.purdue.edu, rank 9 out of 16 processors
Hello world from processor bell-a092.rcac.purdue.edu, rank 8 out of 16 processors
Hello world from processor bell-a092.rcac.purdue.edu, rank 11 out of 16 processors
Hello world from processor bell-a090.rcac.purdue.edu, rank 1 out of 16 processors
Hello world from processor bell-a090.rcac.purdue.edu, rank 2 out of 16 processors
Hello world from processor bell-a090.rcac.purdue.edu, rank 3 out of 16 processors
Hello world from processor bell-a091.rcac.purdue.edu, rank 6 out of 16 processors
Hello world from processor bell-a092.rcac.purdue.edu, rank 10 out of 16 processors
Hello world from processor bell-a090.rcac.purdue.edu, rank 0 out of 16 processors
zhan4429@bell-fe02:~ $
```

Build your own MPI containers

raxml-ng-mpi_1.2.0.def

```
Bootstrap: localimage
From: /apps/base_images/MPI/openmpi/openmpi_4.1.4_pmi2.sif

%labels
    Author "Yucheng Zhang <zhan4429@purdue.edu>"
    Version v1.2.0

%post
    ## Install dependencies
    apt-get -y update
    apt-get -y install flex bison libgmp3-dev
    ## Download and install raxml-ng
    cd /opt
    git clone --recursive https://github.com/amkozlov/raxml-ng
    cd raxml-ng
    mkdir build && cd build
    cmake -DUSE_MPI=ON ..
    make
    make install
```

```
apptainer build raxml-ng-mpi_1.2.0.sif raxml-ng-mpi_1.2.0.def
```

```
#!/bin/bash

#SBATCH -A standby
#SBATCH -t 4:00:00
#SBATCH -N 2
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=128
#SBATCH --job-name=raxml-ng
#SBATCH --mail-type=FAIL,BEGIN,END
#SBATCH --error=%x-%J-%u.err
#SBATCH --output=%x-%J-%u.out

module purge
srun --mpi=pmi2 apptainer exec \
        raxml-ng-mpi_1.2.0.sif \
        raxml-ng-mpi --msa input.fas \
        --model  GTR+G --prefix output \
        --threads 128
```

https://github.com/amkozlov/raxml-ng

Containerizing HPC applications with Singularity/Apptainer

Deployed containers on RCAC clusters

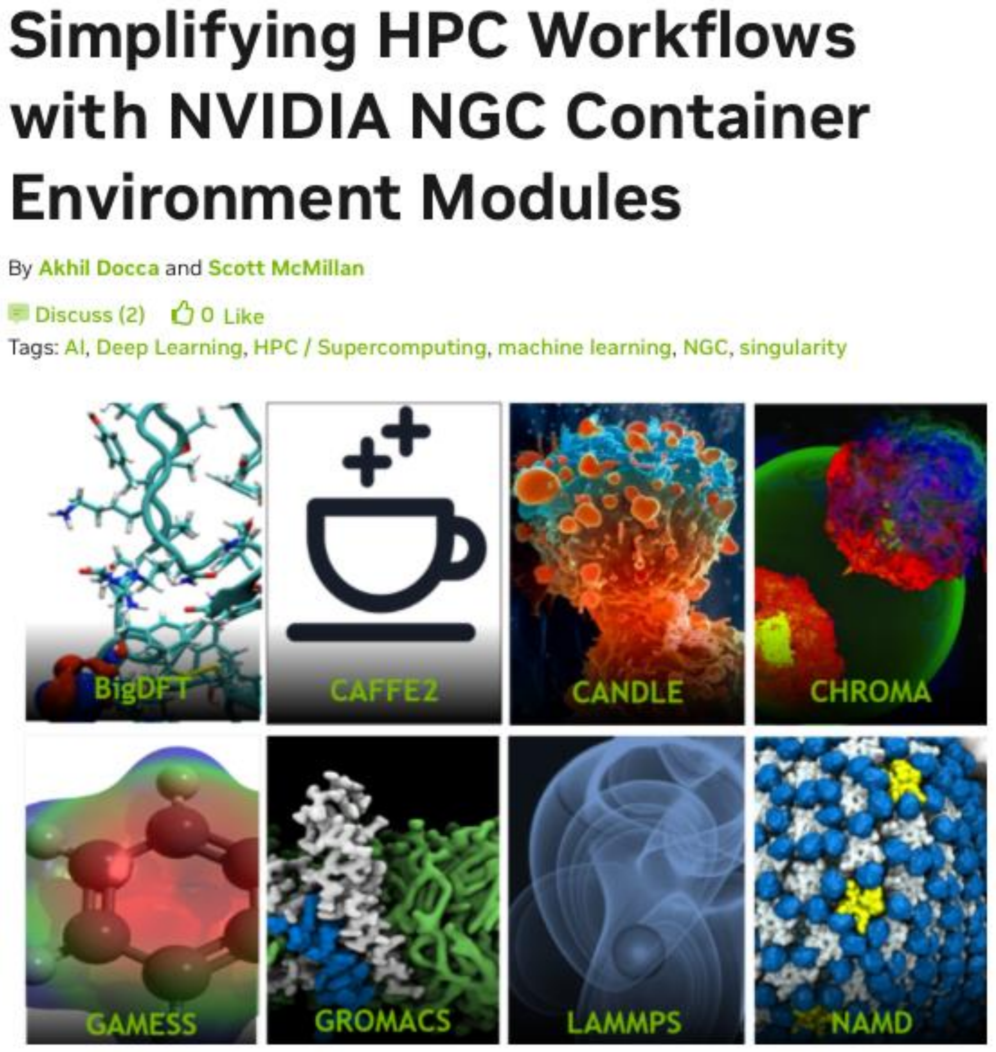
PURDUE UNIVERSITY | Rosen Center for Advanced Computing

NGC container environment modules

NGC container environment modules are lightweight wrappers that make it possible to transparently use NGC containers as environment modules.

1. Allow HPC users to utilize familiar environment module commands.
2. Leverage all the benefits of containers, including portability and reproducibility.

https://github.com/NVIDIA/ngc-container-environment-modules



Simplifying HPC Workflows with NVIDIA NGC Container Environment Modules

By Akhil Docca and Scott McMillan

Discuss (2) 0 Like

Tags: AI, Deep Learning, HPC / Supercomputing, machine learning, NGC, singularity

Nvidia GPU-optimized tools for deep learning, machine learning, and high-performance computing.

https://catalog.ngc.nvidia.com/

NGC containers are deployed on Anvil, Gilbreth and Scholar.

```
module load modtree/gpu ## This is only required on anvil

# Load ngc
    module load ngc

# Check available applications
    module avail

# Load and run specific tools
   module load tensorflow/21.09-tf2-py3
```

Note: to use NGC containers, you also need to use GPU nodes equipped with Nvidia GPUs.

AMD ROCm

AMD GPU software containers for HPC, AI & machine learnin.
https://www.amd.com/en/technologies/infinity-hub

NGC containers are deployed on Bell and Negishi.

```
# Load ngc
    module load rocmcontainers

# Check available applications
    module avail

# Load and run specific tools
   module load pytorch/1.10.0-rocm5.0-ubuntu18.04-py3.7
```

Note: to use AMD containers, you also need to use GPU nodes equipped with AMD GPUs.

Biocontainers

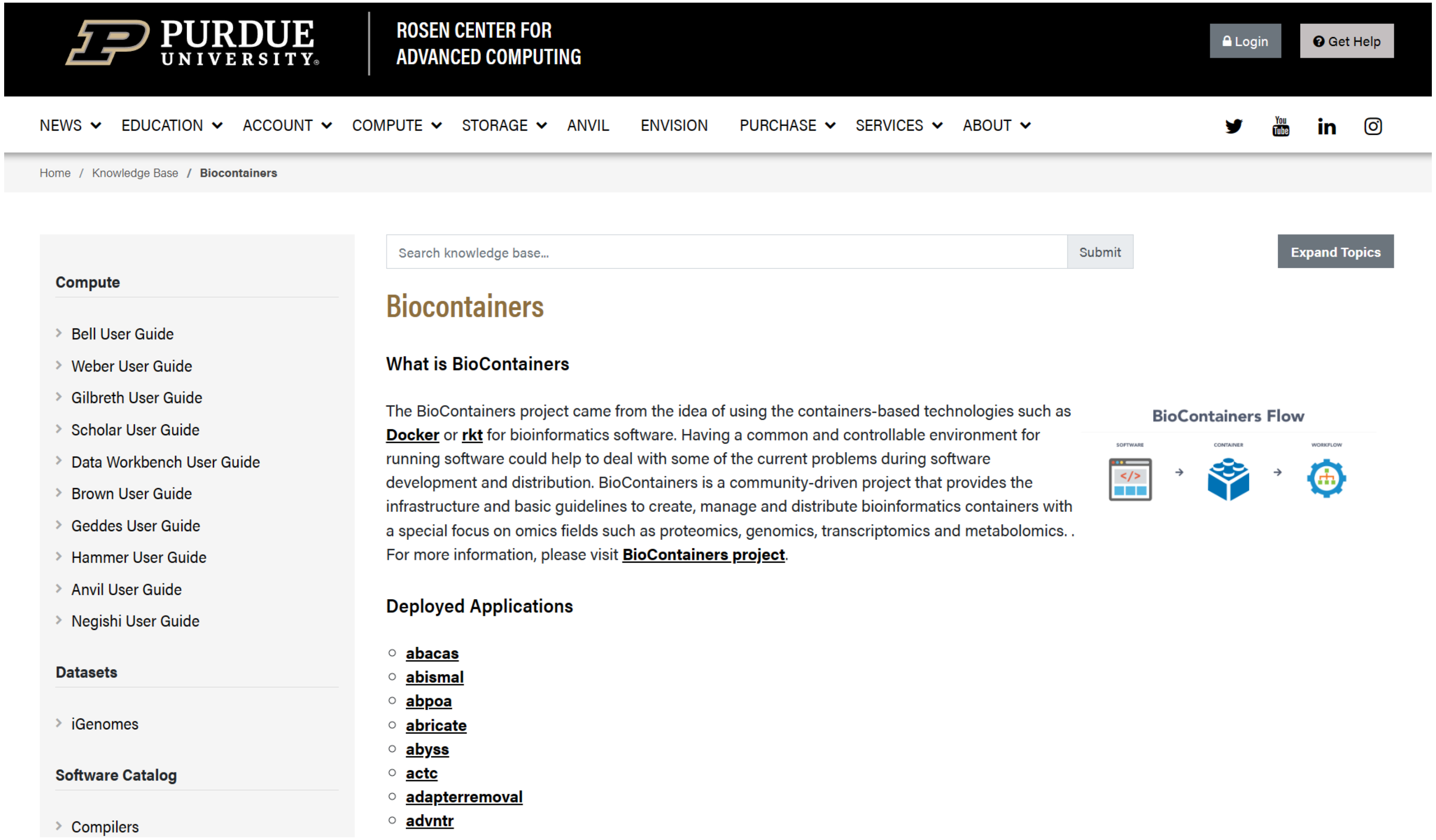
>800 modules for ~600 applications (As of August, 2023)

```
# Load biocontainers
module load biocontainers

# Check available applications
module avail

# Load and run specific tools
module load samtools/1.16
samtools idxstats input.bam
```

Biocontainers documentation

$ module load biocontainers

User guides for each biocontainer module can be found in https://www.rcac.purdue.edu/knowledge/biocontainers



Compute

- Bell User Guide
- Weber User Guide
- Gilbreth User Guide
- Scholar User Guide
- Data Workbench User Guide
- Brown User Guide
- Geddes User Guide
- Hammer User Guide
- Anvil User Guide
- Negishi User Guide

Datasets

- iGenomes

Software Catalog

- Compilers

Biocontainers

What is BioContainers

The BioContainers project came from the idea of using the containers-based technologies such as **Docker** or **rkt** for bioinformatics software. Having a common and controllable environment for running software could help to deal with some of the current problems during software development and distribution. BioContainers is a community-driven project that provides the infrastructure and basic guidelines to create, manage and distribute bioinformatics containers with a special focus on omics fields such as proteomics, genomics, transcriptomics and metabolomics. . For more information, please visit **BioContainers project**.

Deployed Applications

- abacas
- abismal
- abpoa
- abricate
- abyss
- actc
- adapterremoval
- advntr

One more thing: bash wrappers

```
cd $HOME
mkdir containers myapps
export PATH=$HOME/myapps:$PATH

## pull images to $HOME/containers
cd $HOME/containers
singularity pull docker://staphb/samtools:1.18

## create the wrapper stored in $HOME/myapps
cd $HOME/myapps
vim samtools
chmod +x samtools

## use wrapper to run the image
samtools help
```

```
#!/bin/bash

image_dir="$HOME/containers"
image_name="samtools_1.18.sif"

cmd="$(basename $0)"

args="$@"

singularity exec $image_dir/$image_name $cmd $args
```

THANK YOU

ACCESS Help Desk:
https://support.access-ci.org