Magic Tools to Install & Manage Software

Part 2: Singularity Container

Jason Li

HPC User Services

LSU HPC / LONI

sys-help@loni.org

Louisiana State University, Baton Rouge
Oct 30, 2024

Magic Tools to Install & Manage Software

Part 1: CONDA Virtual Environment

Part 2: Singularity Container

Outlines

1. Why Container?
2. Run an Existing Container Image
3. Get More Container Images
4. Build Your Own Container Image

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity
2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity
3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get
4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity
2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity
3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get
4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

1) Problems

- **Core problem:**

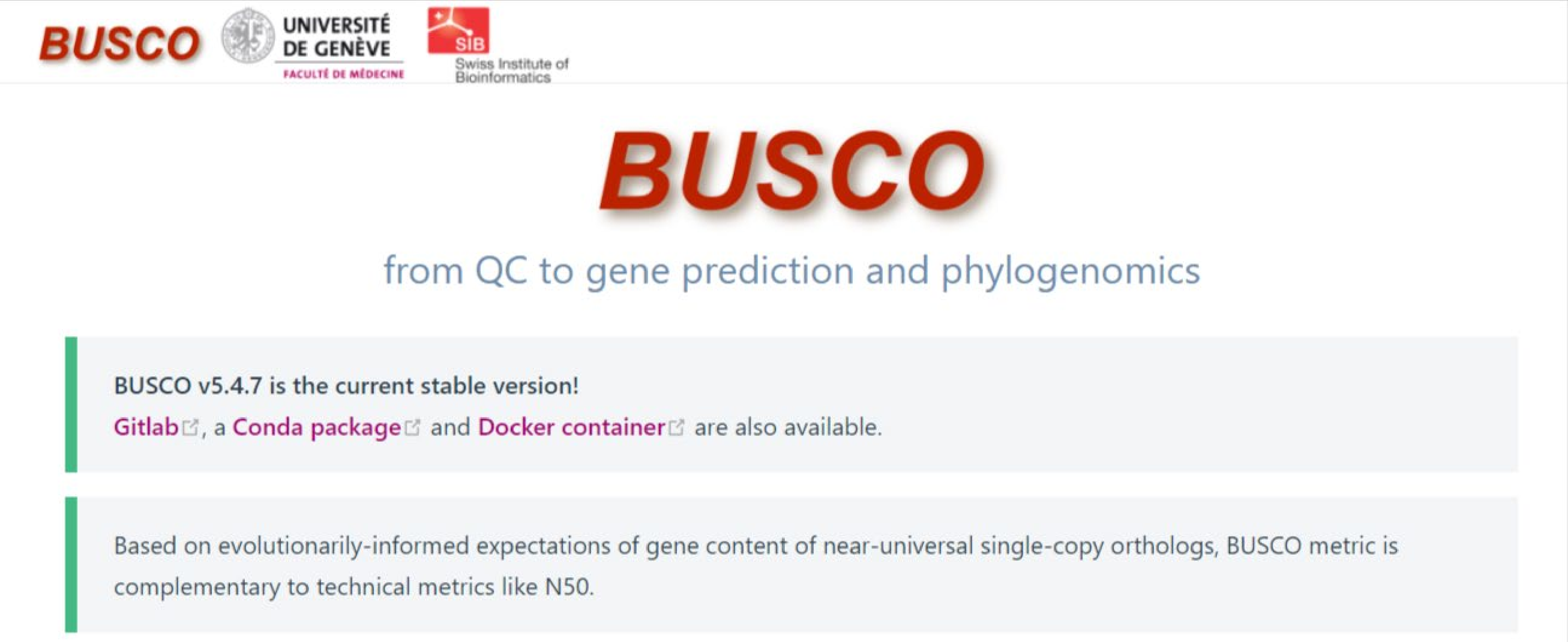
Installing software on an HPC system

1) Problems

- **Traditional Linux solution:**
  - Compiling from source code

1) Problems

a) **Dependencies** (Welcome to Linux!)



BUSCO

UNIVERSITÉ DE GENÈVE FACULTÉ DE MÉDECINE

SIB Swiss Institute of Bioinformatics

BUSCO

from QC to gene prediction and phylogenomics

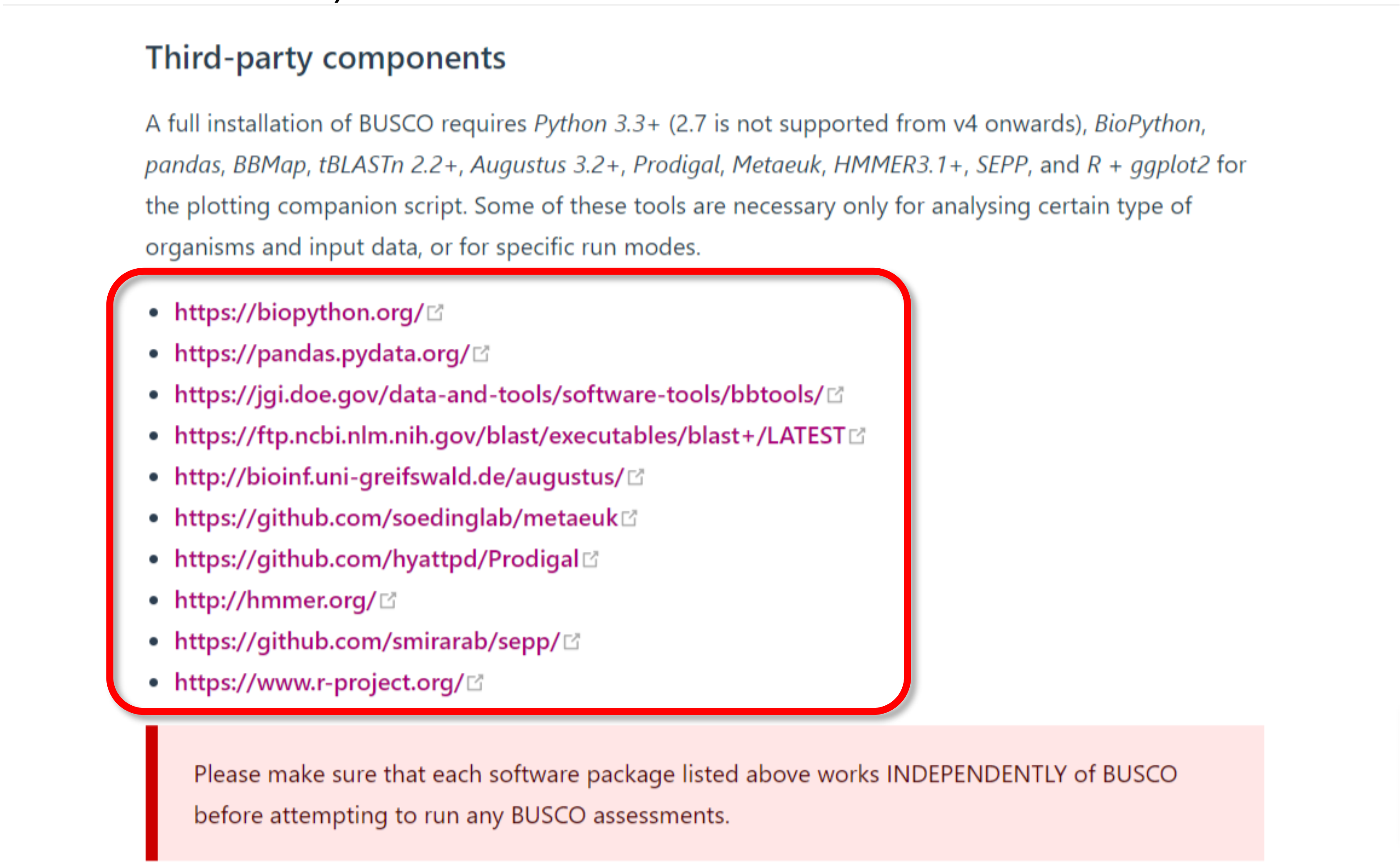
BUSCO v5.4.7 is the current stable version!

Gitlab, a Conda package and Docker container are also available.

Based on evolutionarily-informed expectations of gene content of near-universal single-copy orthologs, BUSCO metric is complementary to technical metrics like N50.

1) Problems

a) **Dependencies** (Welcome to Linux!)



Third-party components

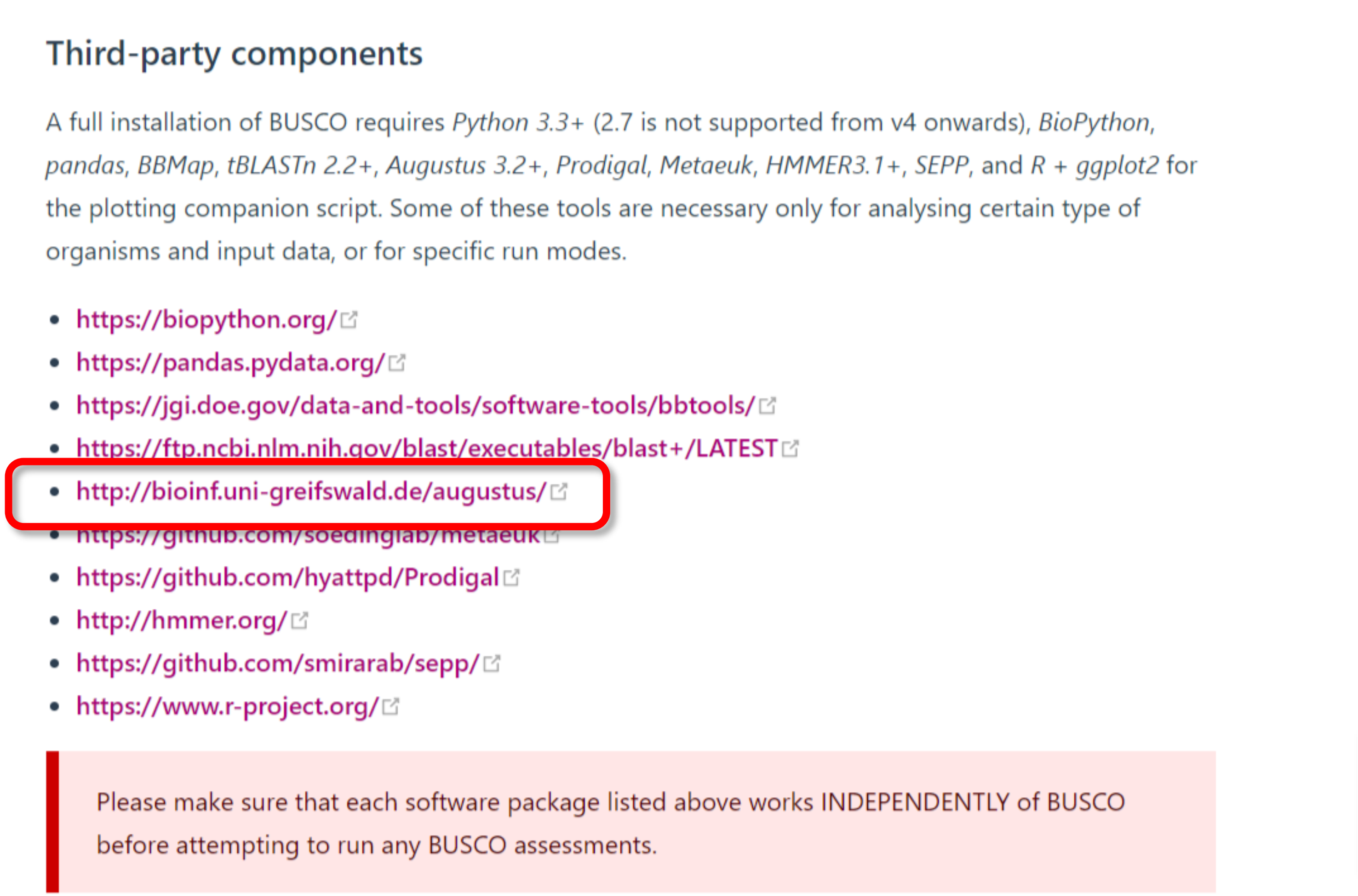
A full installation of BUSCO requires *Python 3.3+* (2.7 is not supported from v4 onwards), *BioPython*, *pandas*, *BBMap*, *tBLASTn 2.2+*, *Augustus 3.2+*, *Prodigal*, *Metaeuk*, *HMMER3.1+*, *SEPP*, and *R + ggplot2* for the plotting companion script. Some of these tools are necessary only for analysing certain type of organisms and input data, or for specific run modes.

- https://biopython.org/
- https://pandas.pydata.org/
- https://jgi.doe.gov/data-and-tools/software-tools/bbtools/
- https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST
- http://bioinf.uni-greifswald.de/augustus/
- https://github.com/soedinglab/metaeuk
- https://github.com/hyattpd/Prodigal
- http://hmmer.org/
- https://github.com/smirarab/sepp/
- https://www.r-project.org/

> Please make sure that each software package listed above works INDEPENDENTLY of BUSCO before attempting to run any BUSCO assessments.

1) Problems

a) **Dependencies** (Welcome to Linux!)



Third-party components

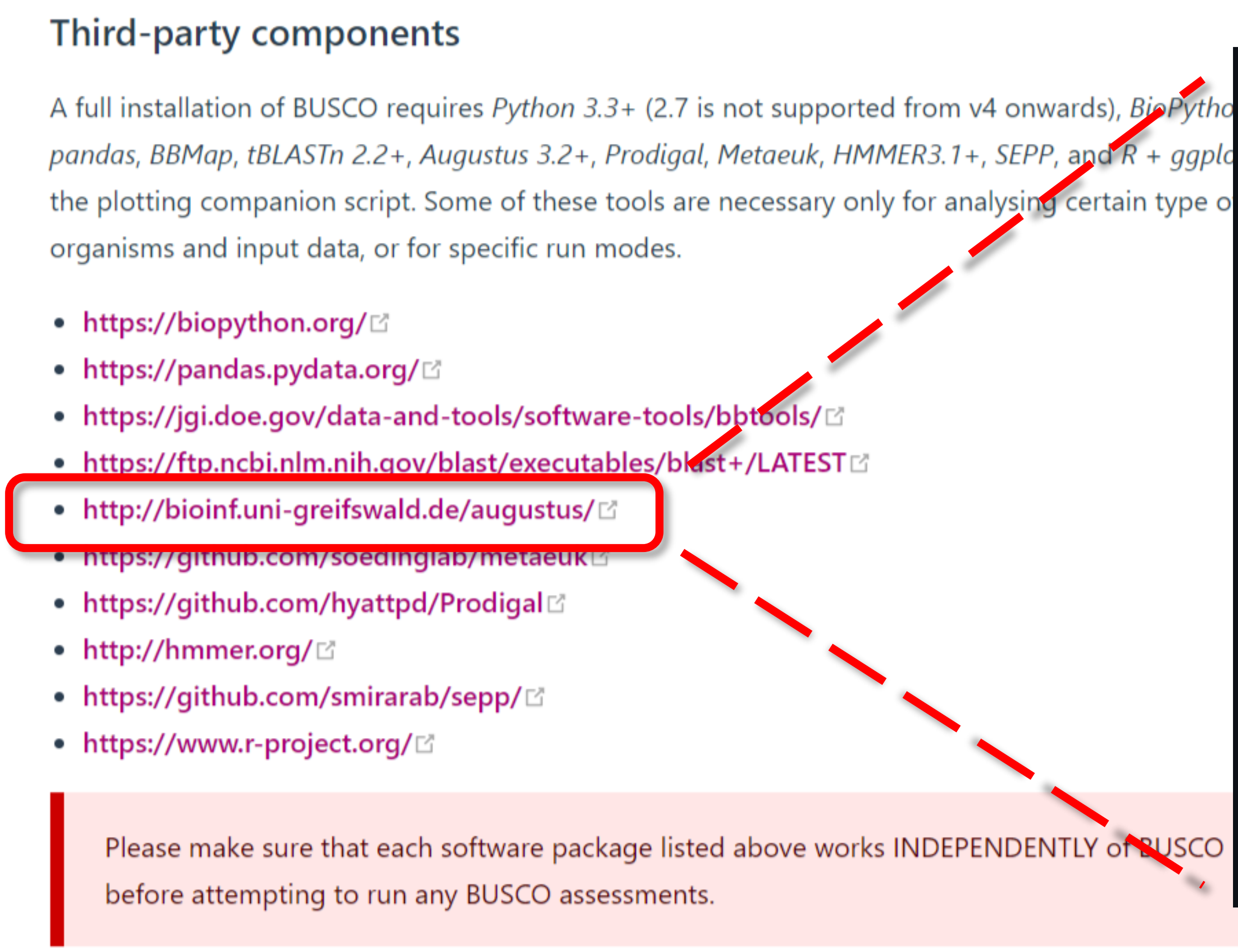
A full installation of BUSCO requires *Python 3.3+* (2.7 is not supported from v4 onwards), *BioPython*, *pandas*, *BBMap*, *tBLASTn 2.2+*, *Augustus 3.2+*, *Prodigal*, *Metaeuk*, *HMMER3.1+*, *SEPP*, and *R + ggplot2* for the plotting companion script. Some of these tools are necessary only for analysing certain type of organisms and input data, or for specific run modes.

- https://biopython.org/
- https://pandas.pydata.org/
- https://jgi.doe.gov/data-and-tools/software-tools/bbtools/
- https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST
- http://bioinf.uni-greifswald.de/augustus/
- https://github.com/soedinglab/metaeuk
- https://github.com/hyattpd/Prodigal
- http://hmmer.org/
- https://github.com/smirarab/sepp/
- https://www.r-project.org/

> Please make sure that each software package listed above works INDEPENDENTLY of BUSCO before attempting to run any BUSCO assessments.

1) Problems

a) **Dependencies** (Welcome to Linux!)

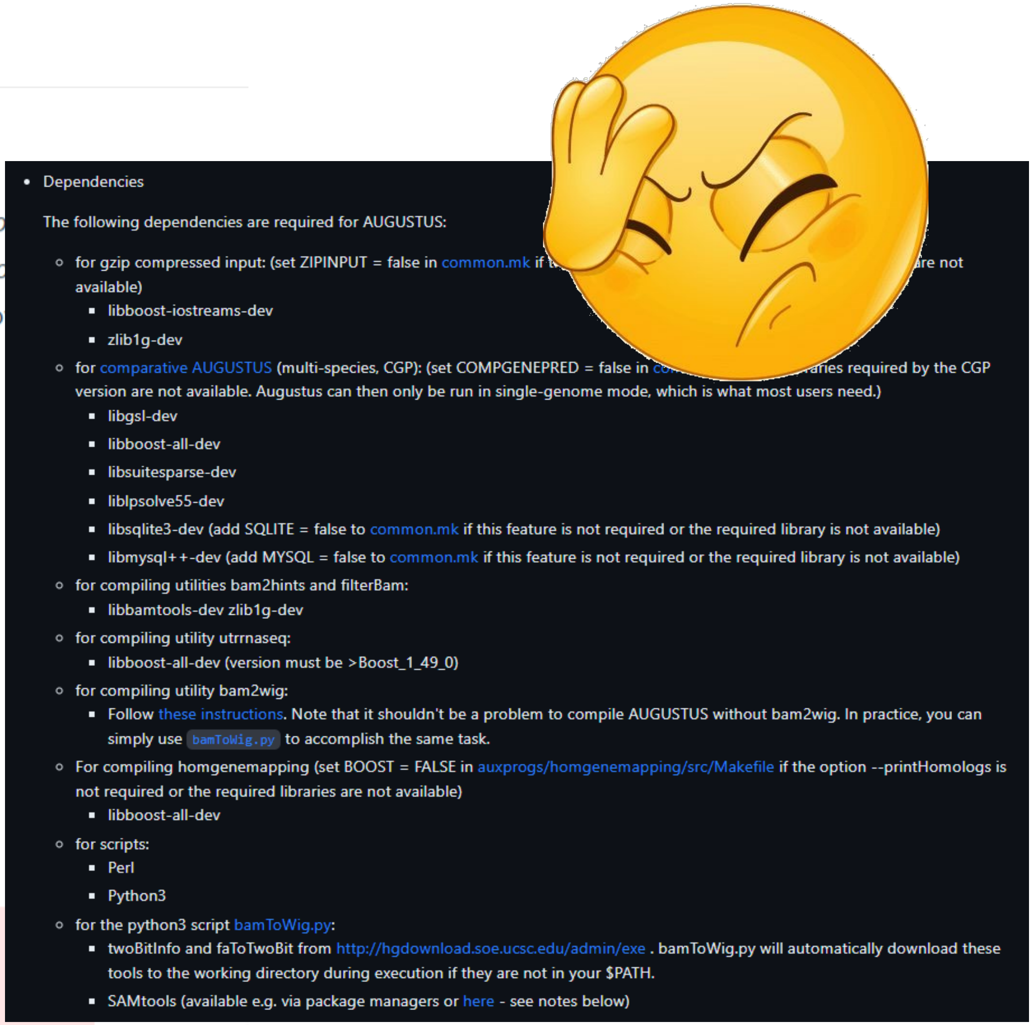


Third-party components

A full installation of BUSCO requires *Python 3.3+* (2.7 is not supported from v4 onwards), *BioPytho pandas, BBMap, tBLASTn 2.2+, Augustus 3.2+, Prodigal, Metaeuk, HMMER3.1+, SEPP,* and *R + ggplo* the plotting companion script. Some of these tools are necessary only for analysing certain type o organisms and input data, or for specific run modes.

- https://biopython.org/
- https://pandas.pydata.org/
- https://jgi.doe.gov/data-and-tools/software-tools/bbtools/
- https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST
- http://bioinf.uni-greifswald.de/augustus/
- https://github.com/soedinglab/metaeuk
- https://github.com/hyattpd/Prodigal
- http://hmmer.org/
- https://github.com/smirarab/sepp/
- https://www.r-project.org/

> Please make sure that each software package listed above works INDEPENDENTLY of BUSCO before attempting to run any BUSCO assessments.



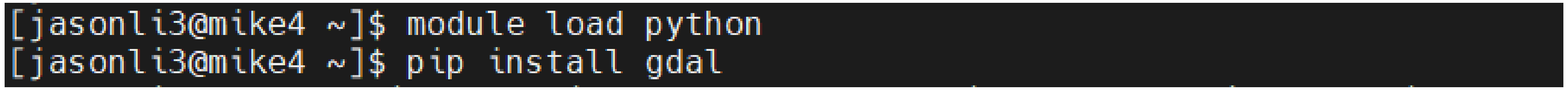
- Dependencies

  The following dependencies are required for AUGUSTUS:

  - for gzip compressed input: (set ZIPINPUT = false in common.mk if t are not available)
    - libboost-iostreams-dev
    - zlib1g-dev
  - for comparative AUGUSTUS (multi-species, CGP): (set COMPGENEPRED = false in co aries required by the CGP version are not available. Augustus can then only be run in single-genome mode, which is what most users need.)
    - libgsl-dev
    - libboost-all-dev
    - libsuitesparse-dev
    - liblpsolve55-dev
    - libsqlite3-dev (add SQLITE = false to common.mk if this feature is not required or the required library is not available)
    - libmysql++-dev (add MYSQL = false to common.mk if this feature is not required or the required library is not available)
  - for compiling utilities bam2hints and filterBam:
    - libbamtools-dev zlib1g-dev
  - for compiling utility utrrnaseq:
    - libboost-all-dev (version must be >Boost_1_49_0)
  - for compiling utility bam2wig:
    - Follow these instructions. Note that it shouldn't be a problem to compile AUGUSTUS without bam2wig. In practice, you can simply use `bamToWig.py` to accomplish the same task.
  - For compiling homgenemapping (set BOOST = FALSE in auxprogs/homgenemapping/src/Makefile if the option --printHomologs is not required or the required libraries are not available)
    - libboost-all-dev
  - for scripts:
    - Perl
    - Python3
  - for the python3 script bamToWig.py:
    - twoBitInfo and faToTwoBit from http://hgdownload.soe.ucsc.edu/admin/exe . bamToWig.py will automatically download these tools to the working directory during execution if they are not in your $PATH.
    - SAMtools (available e.g. via package managers or here - see notes below)

1) Problems

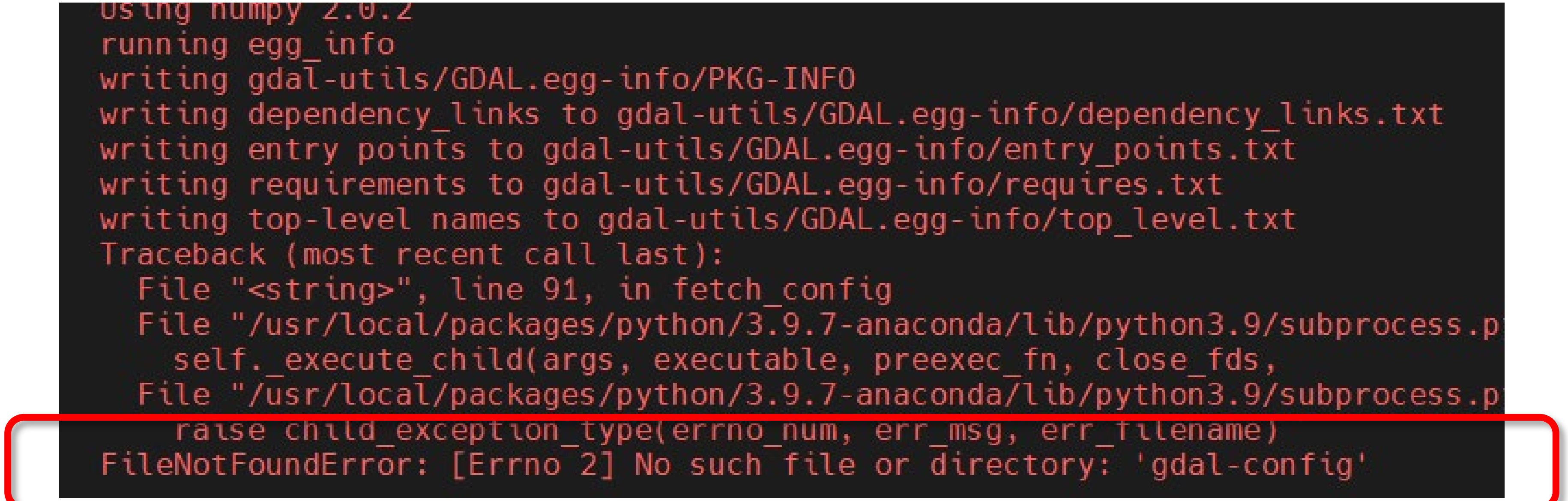
b) **Permission denied** (Welcome to HPC!)



```
[jasonli3@mike4 ~]$ module load python
[jasonli3@mike4 ~]$ pip install gdal
```

1) Problems

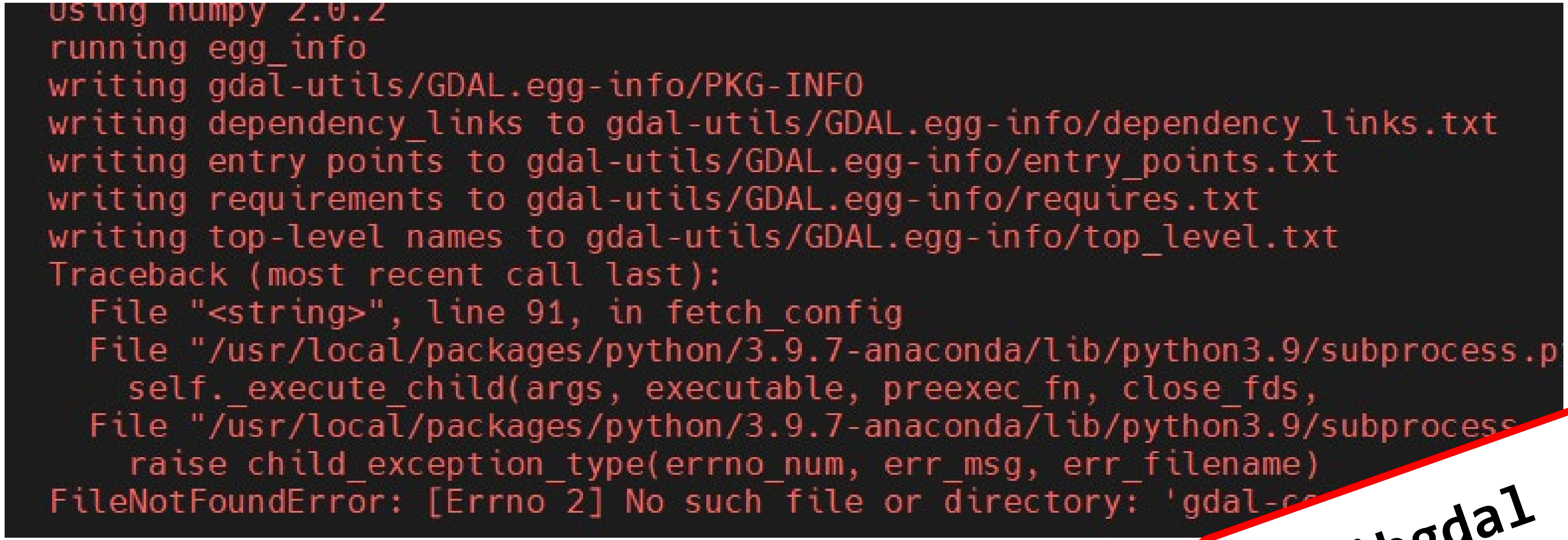
b) **Permission denied** (Welcome to HPC!)



```
Using numpy 2.0.2
running egg_info
writing gdal-utils/GDAL.egg-info/PKG-INFO
writing dependency_links to gdal-utils/GDAL.egg-info/dependency_links.txt
writing entry points to gdal-utils/GDAL.egg-info/entry_points.txt
writing requirements to gdal-utils/GDAL.egg-info/requires.txt
writing top-level names to gdal-utils/GDAL.egg-info/top_level.txt
Traceback (most recent call last):
  File "<string>", line 91, in fetch_config
  File "/usr/local/packages/python/3.9.7-anaconda/lib/python3.9/subprocess.p
    self._execute_child(args, executable, preexec_fn, close_fds,
  File "/usr/local/packages/python/3.9.7-anaconda/lib/python3.9/subprocess.p
    raise child_exception_type(errno_num, err_msg, err_filename)
FileNotFoundError: [Errno 2] No such file or directory: 'gdal-config'
```

1) Problems

b) **Permission denied** (Welcome to HPC!)



```
Using numpy 2.0.2
running egg_info
writing gdal-utils/GDAL.egg-info/PKG-INFO
writing dependency_links to gdal-utils/GDAL.egg-info/dependency_links.txt
writing entry points to gdal-utils/GDAL.egg-info/entry_points.txt
writing requirements to gdal-utils/GDAL.egg-info/requires.txt
writing top-level names to gdal-utils/GDAL.egg-info/top_level.txt
Traceback (most recent call last):
  File "<string>", line 91, in fetch_config
  File "/usr/local/packages/python/3.9.7-anaconda/lib/python3.9/subprocess.p
    self._execute_child(args, executable, preexec_fn, close_fds,
  File "/usr/local/packages/python/3.9.7-anaconda/lib/python3.9/subprocess
    raise child_exception_type(errno_num, err_msg, err_filename)
FileNotFoundError: [Errno 2] No such file or directory: 'gdal-c
```

libgdal

1) Problems

b) **Permission denied** (Welcome to HPC!)

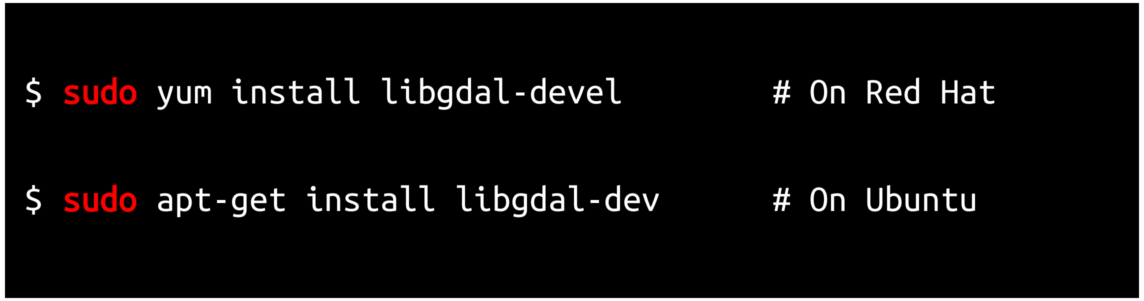
- If you ask Google / ChatGPT…

```
$ sudo yum install libgdal-devel        # On Red Hat

$ sudo apt-get install libgdal-dev      # On Ubuntu
```

1) Problems

b) **Permission denied** (Welcome to HPC!)

- If you ask Google / ChatGPT…



```
$ sudo yum install libgdal-devel        # On Red Hat

$ sudo apt-get install libgdal-dev      # On Ubuntu
```

1) Problems

b) **Permission denied** (Welcome to HPC!)

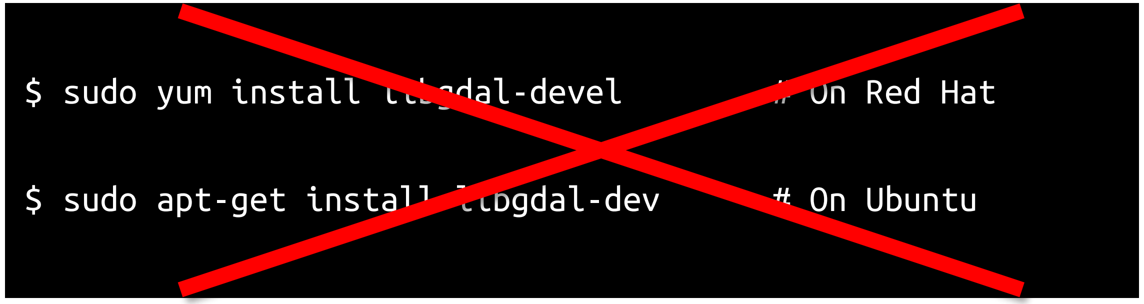
- If you ask Google / ChatGPT…

```
$ sudo yum install libgdal-devel        # On Red Hat

$ sudo apt-get install libgdal-dev      # On Ubuntu
```

1) Problems

b) **Permission denied** (Welcome to HPC!)

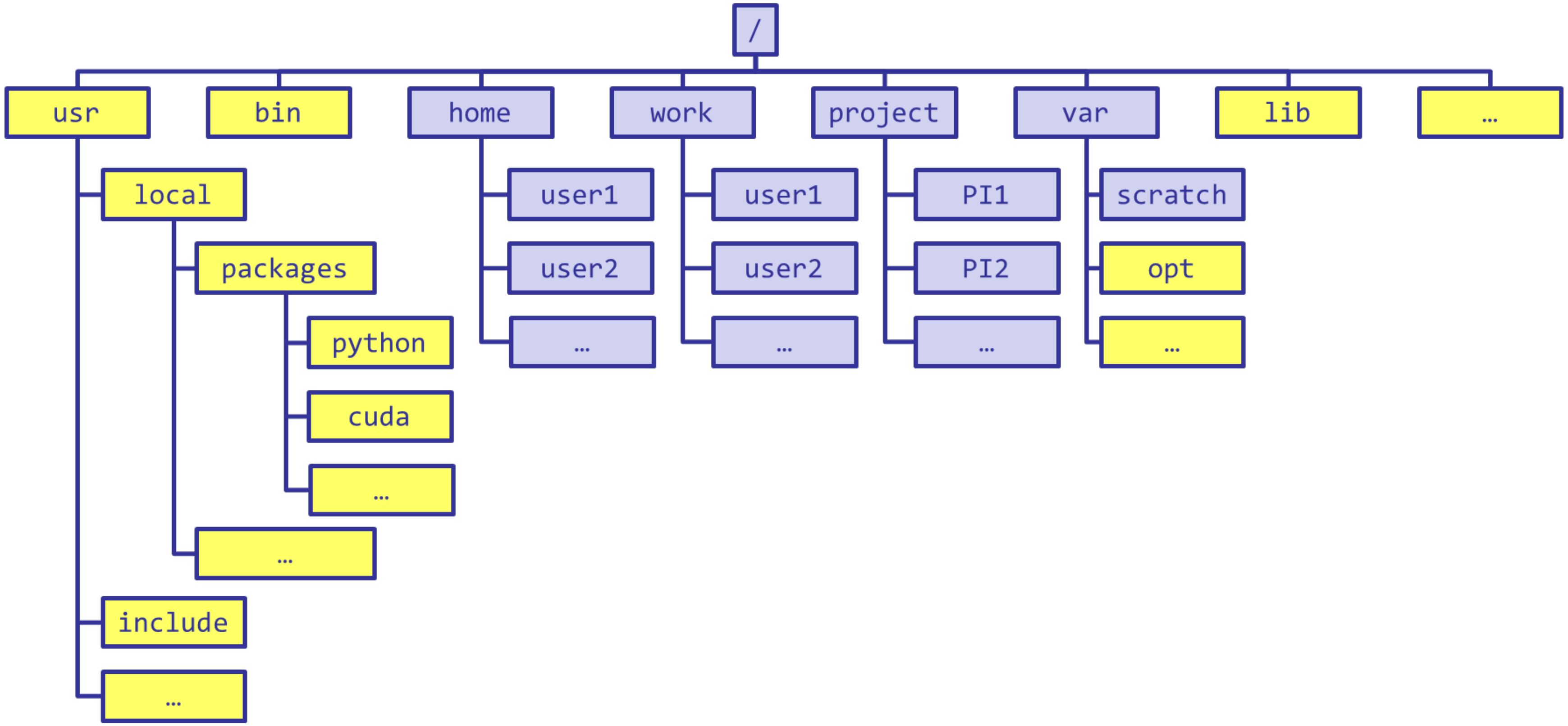
- If you ask Google / ChatGPT…



```
$ sudo yum install libgdal-devel          # On Red Hat

$ sudo apt-get install libgdal-dev        # On Ubuntu
```

1) Problems

b) **Permission denied** (Welcome to HPC!)



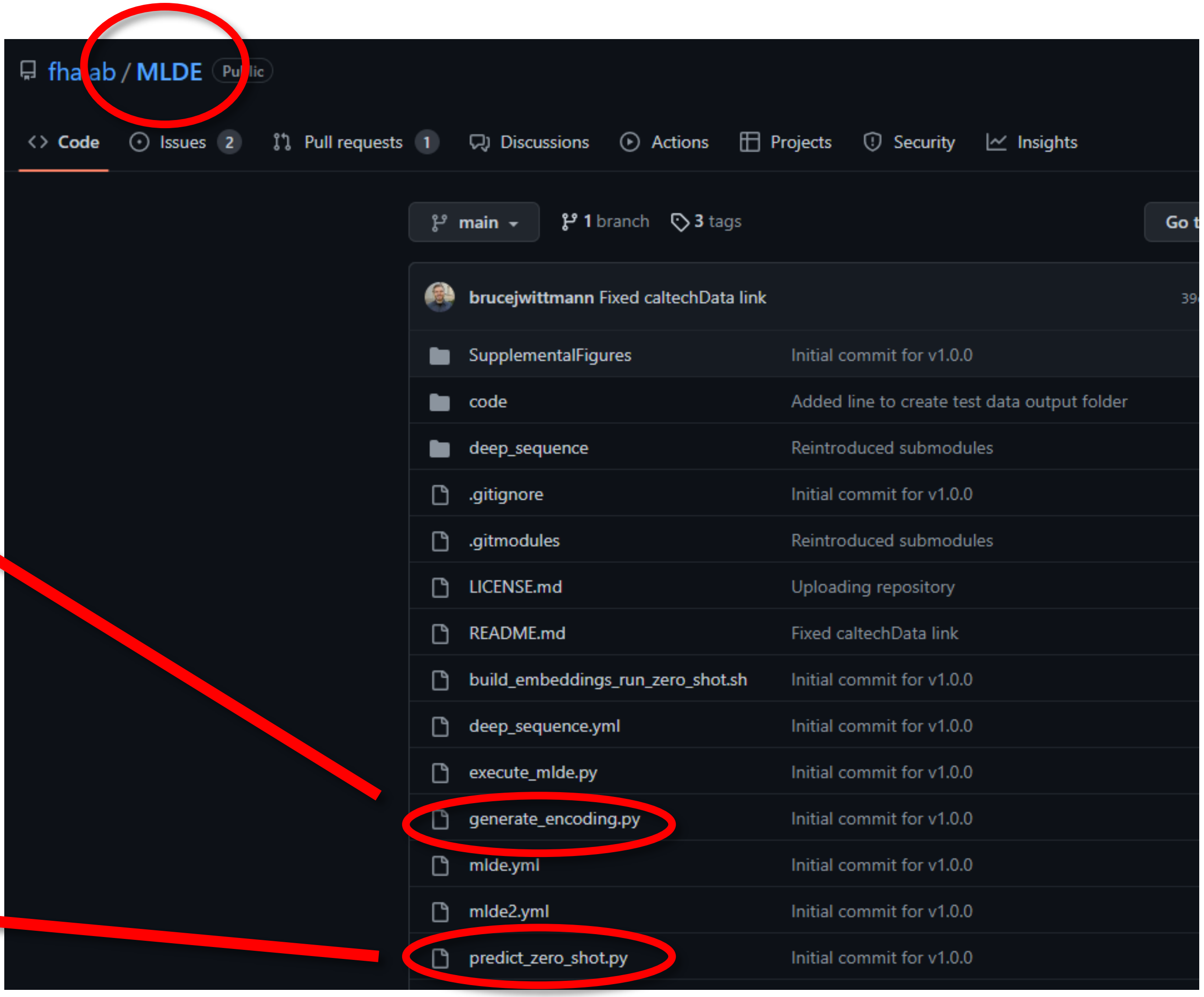
1) Problems

c) Conflicted packages

- What if I need two packages w/ conflicted dependencies?

```
Tensorflow 1.13
```

```
PyTorch > 1.5
```



fha ab / MLDE Pu lic

brucejwittmann Fixed caltechData link

SupplementalFigures	Initial commit for v1.0.0
code	Added line to create test data output folder
deep_sequence	Reintroduced submodules
.gitignore	Initial commit for v1.0.0
.gitmodules	Reintroduced submodules
LICENSE.md	Uploading repository
README.md	Fixed caltechData link
build_embeddings_run_zero_shot.sh	Initial commit for v1.0.0
deep_sequence.yml	Initial commit for v1.0.0
execute_mlde.py	Initial commit for v1.0.0
generate_encoding.py	Initial commit for v1.0.0
mlde.yml	Initial commit for v1.0.0
mlde2.yml	Initial commit for v1.0.0
predict_zero_shot.py	Initial commit for v1.0.0

1) Problems

d) Sharing / Migrating your software

- Huge effort & large disk quota to install
  - What if my colleagues want to use?
  - What if I want to migrate a different cluster?

1) Problems

Any of those apply to you?

1) Problems

Magic Tools to Install / Manage Software

Part 1: CONDA Virtual Environment

Part 2: Singularity Container

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

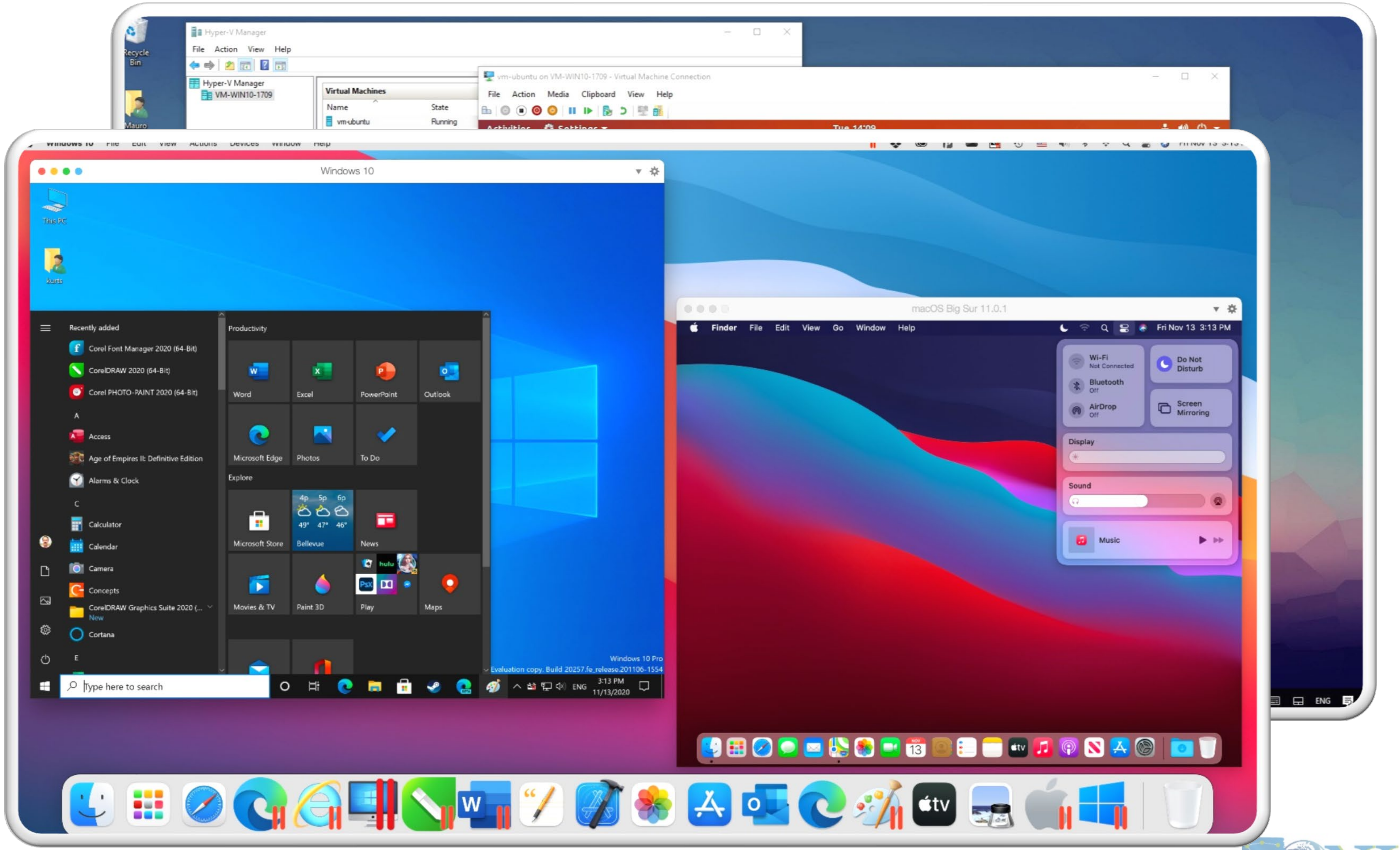
3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

2) Container & Singularity

a) What is a **container**?

- **Virtual machine**
  - "Virtualize" / "mimic" an **entire computer** on another computer
  - Virtualize both **hardware** and **software**



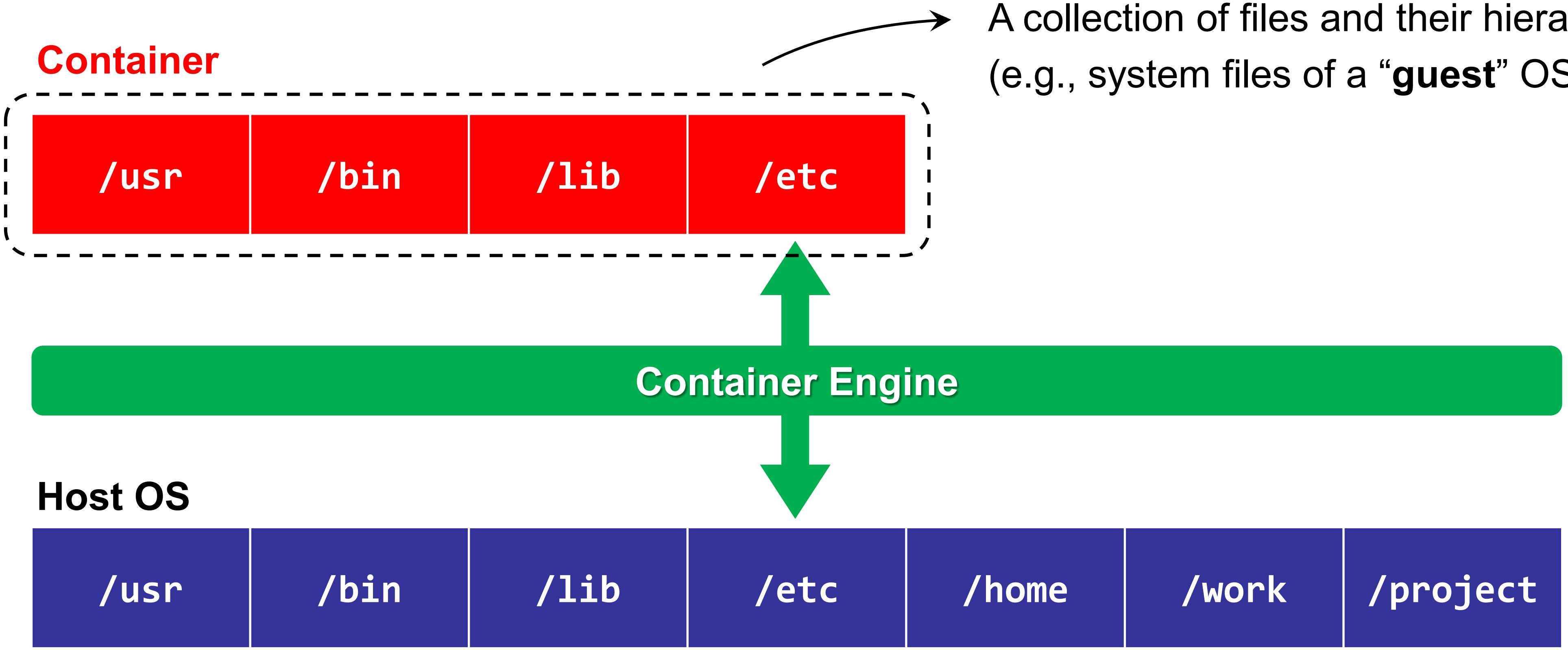
2) Container & Singularity

a) What is a container?

- **Container:**
  - A **lightweight** and **fast** virtual machine
  - Only virtualize the **Operation System** (meaning, does not virtualize hardware)
  - Only virtualize **Linux** on **Linux**

2) Container & Singularity

a) What is a container?



A collection of files and their hierarchy (e.g., system files of a “**guest**” OS)

2) Container & Singularity

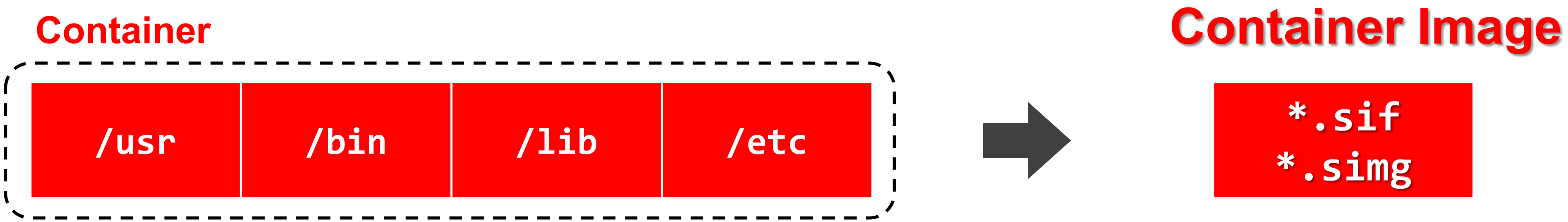
a) What is a **container**?

/usr	/bin	/lib	/etc	/home	/work	/project

- **A "chimera" system:**
  - Can virtualize **an entirely different OS** !
  - Can contain other **software packages** (inc. dependencies, environment settings, etc.) installed in the guest OS

2) Container & Singularity

a) What is a container?



2) Container & Singularity

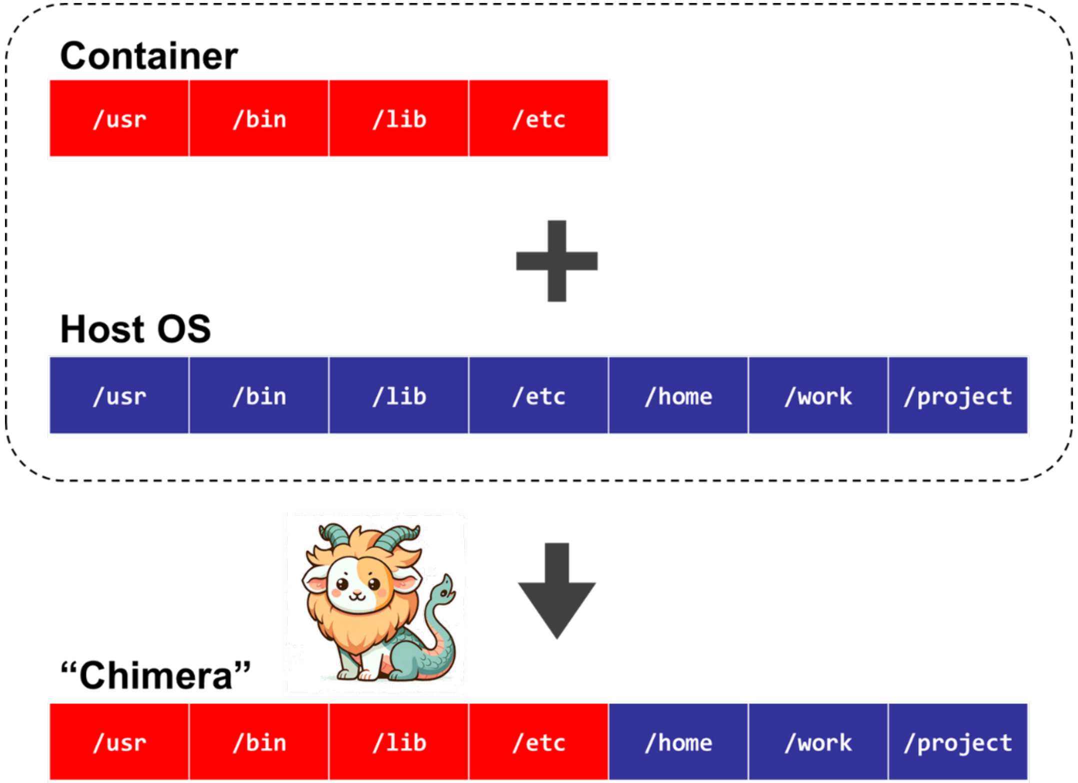
a) What is a container?

- **Properties:**
  - **Self-contained**

    All dependencies can be installed within the container

  - **Isolated**

    Whatever happens in a container stays in that container…



Container

/usr /bin /lib /etc

+

Host OS

/usr /bin /lib /etc /home /work /project

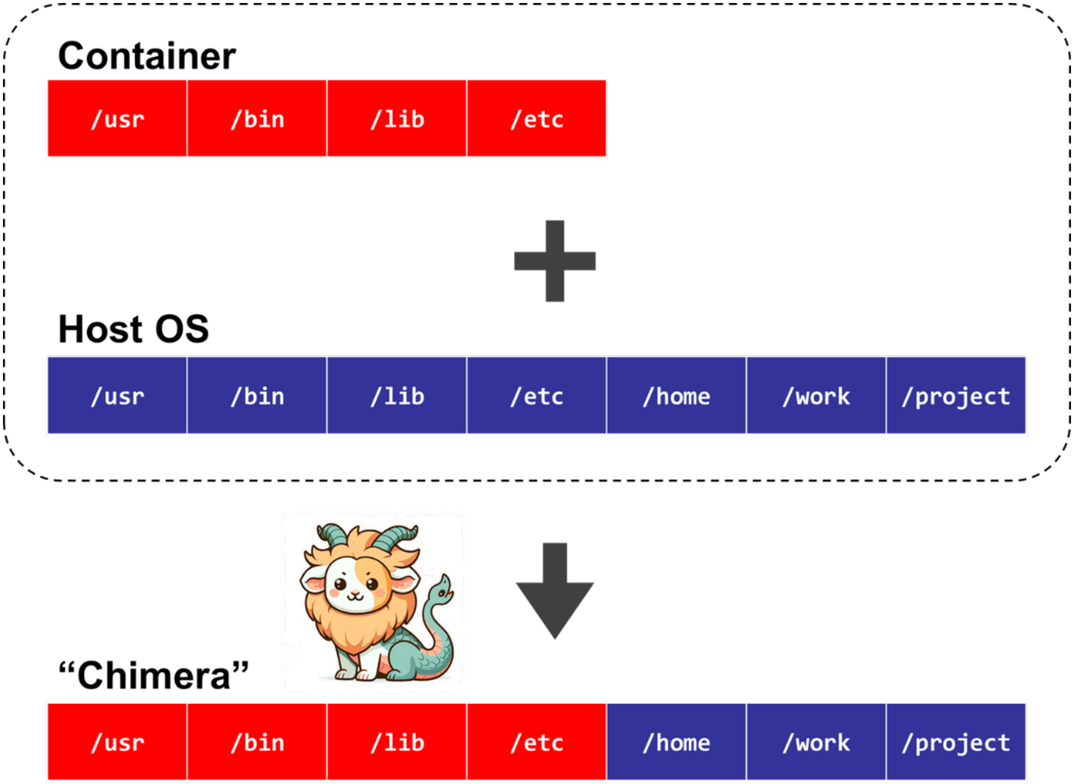
"Chimera"

/usr /bin /lib /etc /home /work /project

2) Container & Singularity

b) How does it solve my problems?

- **Dependency issue**
  - Pack all dependencies (even OS) in container
  - Can use `apt-get` or `yum`
  - **Developers now release containers!**
- **Permission issue**
  - Can't write to certain paths on HPC, but **CAN** write to them in container
- **Conflicted packages**
  - Install in different containers.
- **Share / Migrate**
  - Copy-paste a container image!



Container

/usr | /bin | /lib | /etc

+

Host OS

/usr | /bin | /lib | /etc | /home | /work | /project

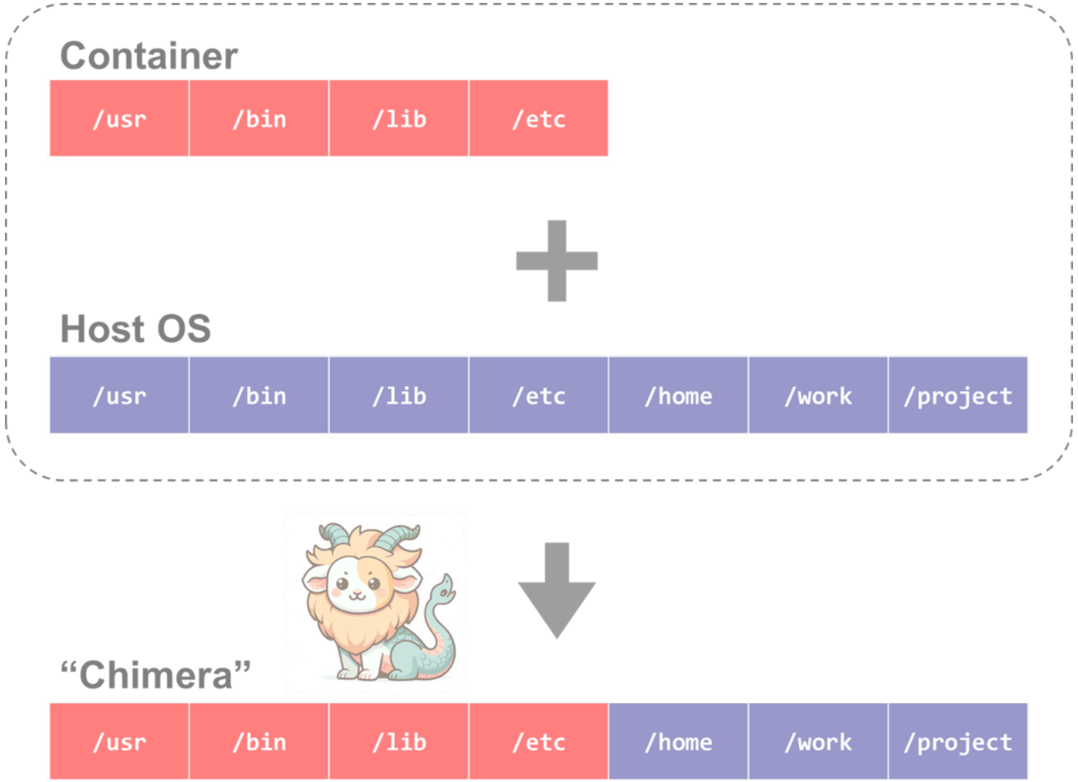
"Chimera"

/usr | /bin | /lib | /etc | /home | /work | /project

2) Container & Singularity

c) What is Singularity?

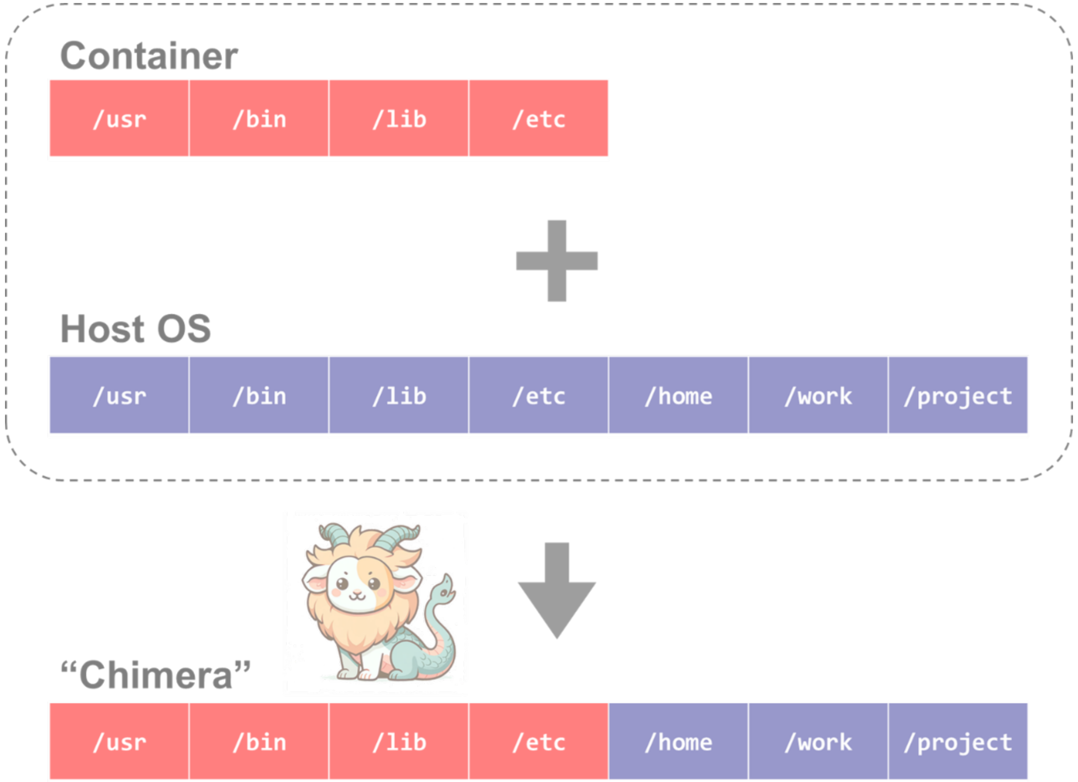
Technology →



2) Container & Singularity

c) What is Singularity?

↑ **Software** system that implements the technology



Container

/usr	/bin	/lib	/etc

+

Host OS

/usr	/bin	/lib	/etc	/home	/work	/project

“Chimera”

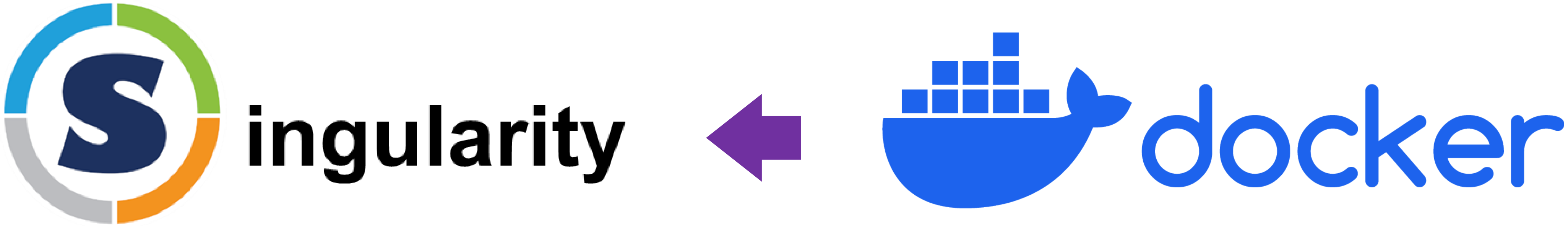
/usr	/bin	/lib	/etc	/home	/work	/project

2) Container & Singularity

c) What is **Singularity**?

2) Container & Singularity

c) What is **Singularity**?



- Does **NOT** need root privileges
- "**Container for HPC**"

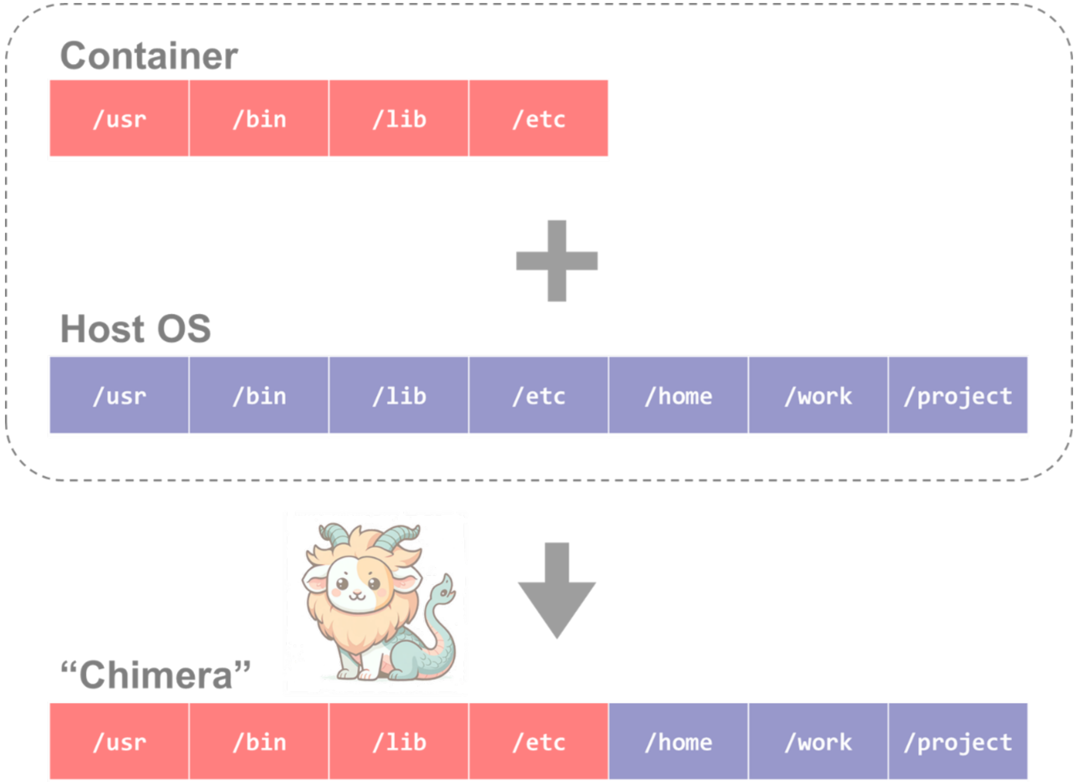
- **Needs** root privileges

Summary

Technology that helps with software installation →

↓ **Software** system that implements the technology

Singularity



Container

/usr /bin /lib /etc

+

Host OS

/usr /bin /lib /etc /home /work /project

“Chimera”

/usr /bin /lib /etc /home /work /project

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

1) What you need

- **Singularity availability**

a) On **all clusters**

✓ **LSU HPC**: SMIC, Deep Bayou, SuperMike 3

✓ **LONI**: QB3, QB4

b) Only on **computing nodes**

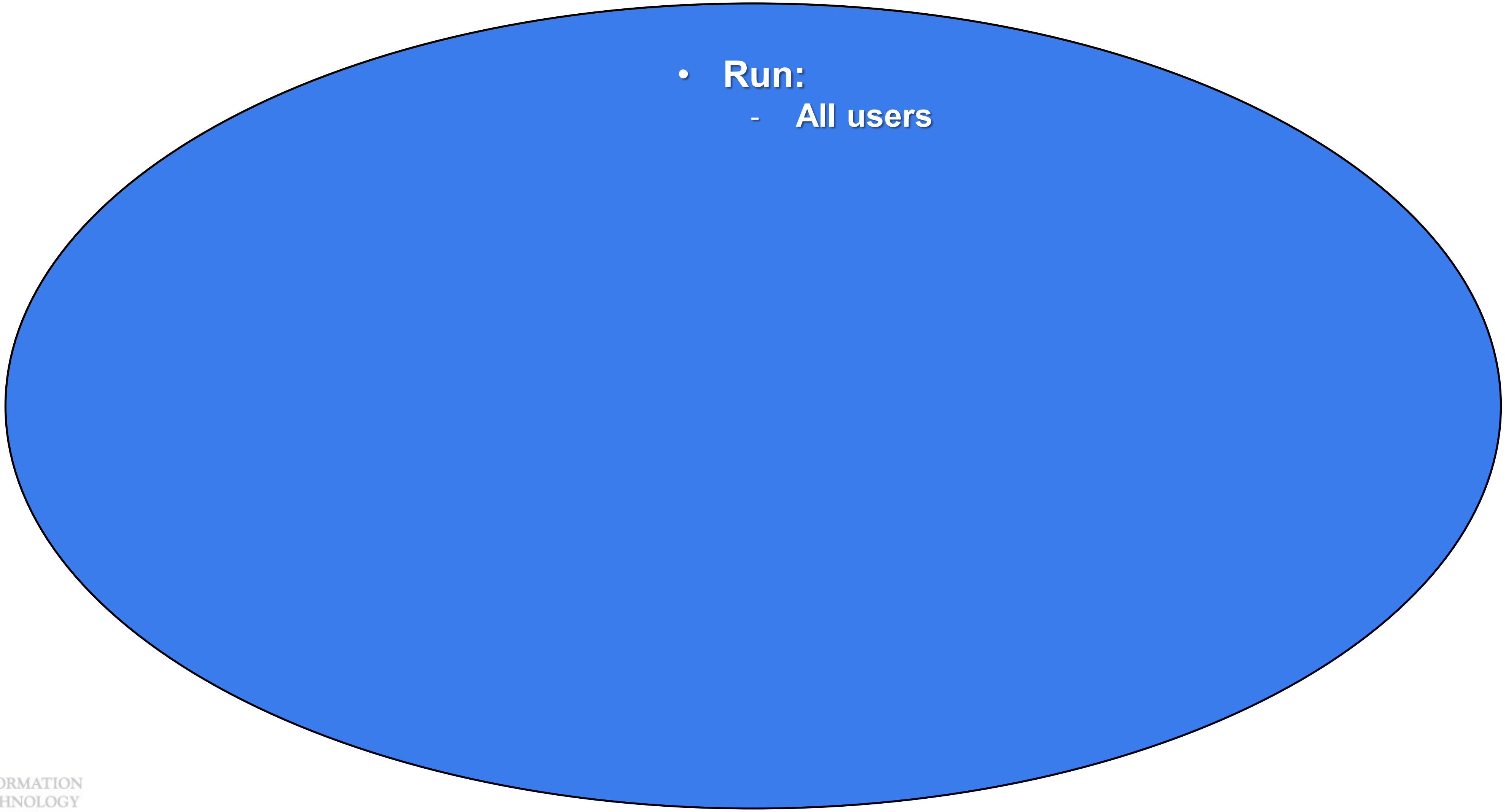
× Unavailable on head nodes

✓ Must start a job (interactive & batch)

c) To **all users**

× No additional action required

1) What you need



- **Run:**
  - **All users**

Outlines

1. Why Container?
   1) Problems
   2) Container & Singularity

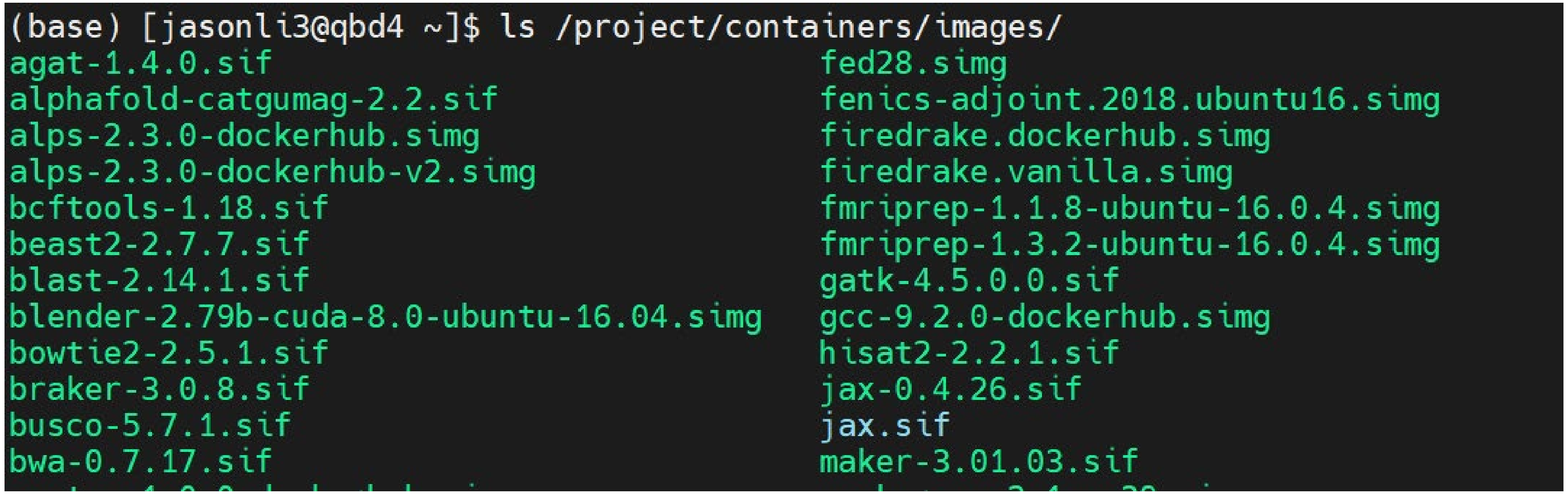
2. **Run an Existing Container Image**
   1) What you need
   2) **Basic commands**
   3) Running jobs with Singularity

3. Get More Container Images
   1) What you need
   2) Where to get
   3) How to get

4. Build Your Own Container Image
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

2) Basic commands

- **Available images**
  - On all clusters: `/project/containers/images`



```
(base) [jasonli3@qbd4 ~]$ ls /project/containers/images/
agat-1.4.0.sif                             fed28.simg
alphafold-catgumag-2.2.sif                 fenics-adjoint.2018.ubuntu16.simg
alps-2.3.0-dockerhub.simg                  firedrake.dockerhub.simg
alps-2.3.0-dockerhub-v2.simg               firedrake.vanilla.simg
bcftools-1.18.sif                          fmriprep-1.1.8-ubuntu-16.0.4.simg
beast2-2.7.7.sif                           fmriprep-1.3.2-ubuntu-16.0.4.simg
blast-2.14.1.sif                           gatk-4.5.0.0.sif
blender-2.79b-cuda-8.0-ubuntu-16.04.simg   gcc-9.2.0-dockerhub.simg
bowtie2-2.5.1.sif                          hisat2-2.2.1.sif
braker-3.0.8.sif                           jax-0.4.26.sif
busco-5.7.1.sif                            jax.sif
bwa-0.7.17.sif                             maker-3.01.03.sif
```

2) Basic commands

a) Common usage 1: Open a shell in the image

Syntax	Description
`singularity shell <container>`	Starts an interactive shell in the image

Try me: `/project/containers/images/ubuntu-training.sif`

2) Basic commands

a) Common usage 1: Open a shell in the image

Syntax		Description
`singularity shell [options] <container>`		Starts an interactive shell in the image
`[Options]`	`-B /path/to/bind`	Bind a path(s) • /home is bound by default
	`--nv`	Enable Nvidia GPU

2) Basic commands

b) Common usage 2: Execute a single command in the image

Syntax	Description
`singularity exec <container> <command>`	Execute a command in the image

Try me: `/project/containers/images/ubuntu-training.sif`

2) Basic commands

b) Common usage 2: Execute a single command in the image

Syntax		Description
`singularity exec [options] <container> <command>`		Execute a command in the image
`[Options]`	`-B /path/to/bind`	Bind a path(s) • /home is bound by default
	`--nv`	Enable Nvidia GPU

2) Basic commands

c) Another (less) common usage: Run a prewritten script

Syntax		Description
`singularity run [options] <container>`		Run a prewritten script
[Options]	`-B /path/to/bind`	Bind a path(s) • /home is bound by default
	`--nv`	Enable Nvidia GPU

2) Basic commands

- **Quick recap**

Syntax	Description
`singularity shell [options] <container>`	Starts an interactive shell in the image
`singularity exec [options] <container> <command>`	Execute a command in the image
`singularity run [options] <container>`	Run a prewritten script

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity
2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity
3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get
4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

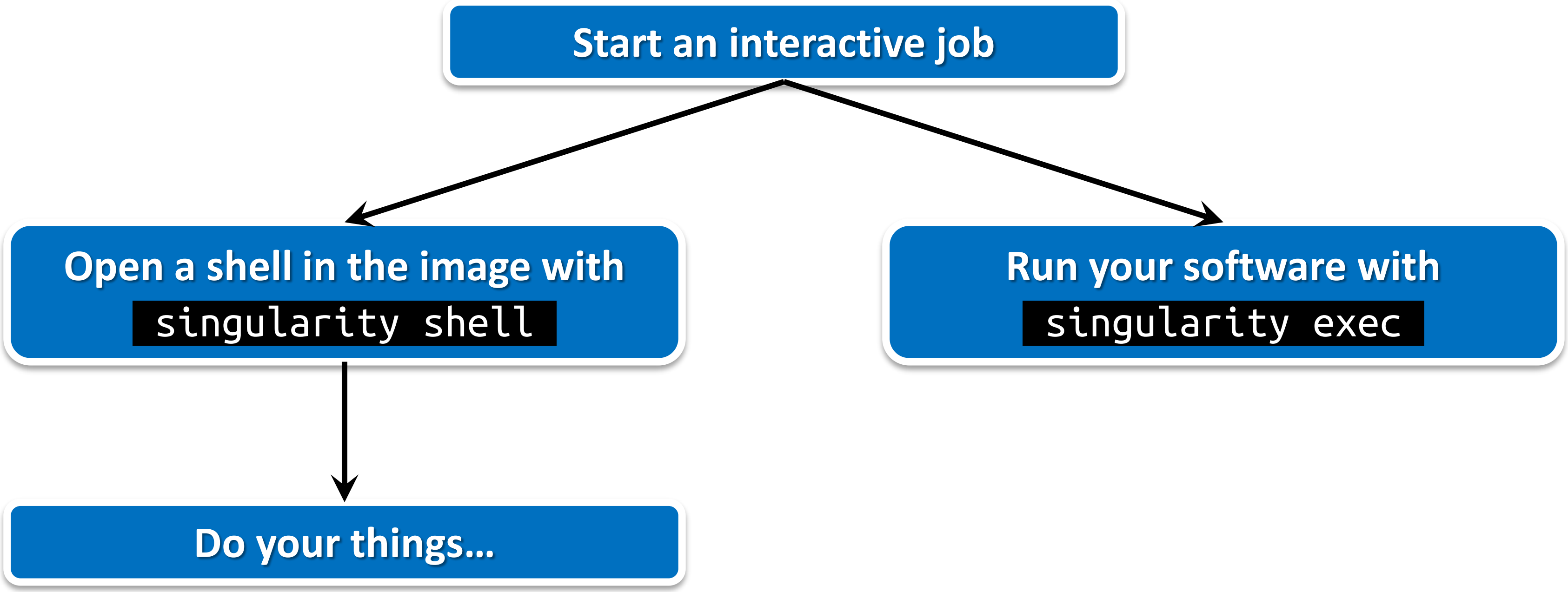
3) Run jobs with Singularity

- **Job types and commands**

Job Type	Commands	Purpose
Interactive	• `singularity shell [options] <container>` • `singularity exec  [options] <container> <command>`	• Debugging & testing
Batch	• `singularity exec [options] <container> <command>`	• Production

3) Run jobs with Singularity

a) Interactive job



Start an interactive job

Open a shell in the image with `singularity shell`

Run your software with `singularity exec`

Do your things…

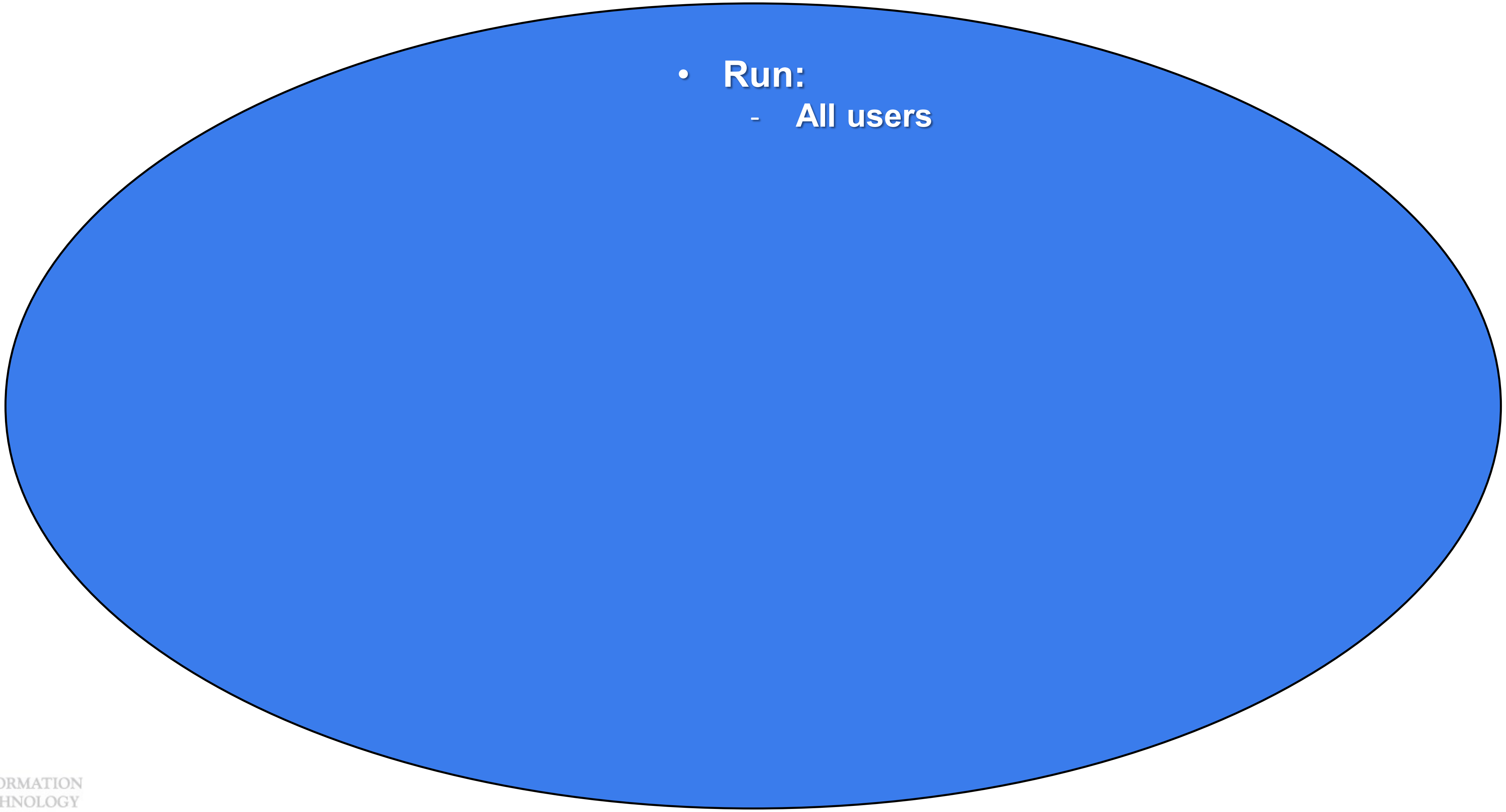
3) Run jobs with Singularity

b) Batch job

```
#!/bin/bash
#SBATCH -A <Allocation name>
#SBATCH -p workq
#SBATCH -N 1
#SBATCH -n 64
#SBATCH -t 24:00:00

cd /to/work/directory

IMG=/home/admin/singularity/ubuntu-training.sif

singularity exec -B /work,/project $IMG \
  python myjob.py
```

Example

Summary



- Run:
  - All users

Summary

Syntax	Description
`singularity shell [options] <container>`	Run a prewritten script
`singularity exec [options] <container> <command>`	Execute a command in the image
`singularity run [options] <container>`	Run a prewritten script

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

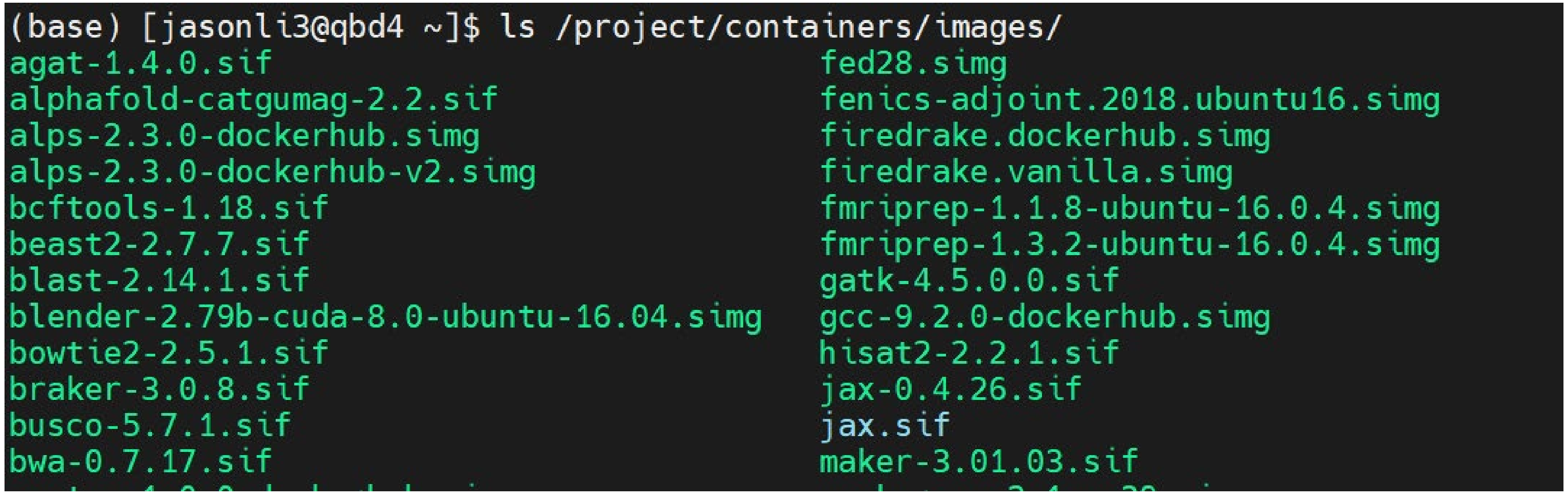
3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

3. Get More Container Images

- **Available images**

  – On all clusters: `/project/containers/images`



```
(base) [jasonli3@qbd4 ~]$ ls /project/containers/images/
agat-1.4.0.sif                              fed28.simg
alphafold-catgumag-2.2.sif                  fenics-adjoint.2018.ubuntu16.simg
alps-2.3.0-dockerhub.simg                   firedrake.dockerhub.simg
alps-2.3.0-dockerhub-v2.simg                firedrake.vanilla.simg
bcftools-1.18.sif                           fmriprep-1.1.8-ubuntu-16.0.4.simg
beast2-2.7.7.sif                            fmriprep-1.3.2-ubuntu-16.0.4.simg
blast-2.14.1.sif                            gatk-4.5.0.0.sif
blender-2.79b-cuda-8.0-ubuntu-16.04.simg    gcc-9.2.0-dockerhub.simg
bowtie2-2.5.1.sif                           hisat2-2.2.1.sif
braker-3.0.8.sif                            jax-0.4.26.sif
busco-5.7.1.sif                             jax.sif
bwa-0.7.17.sif                              maker-3.01.03.sif
```

Outlines

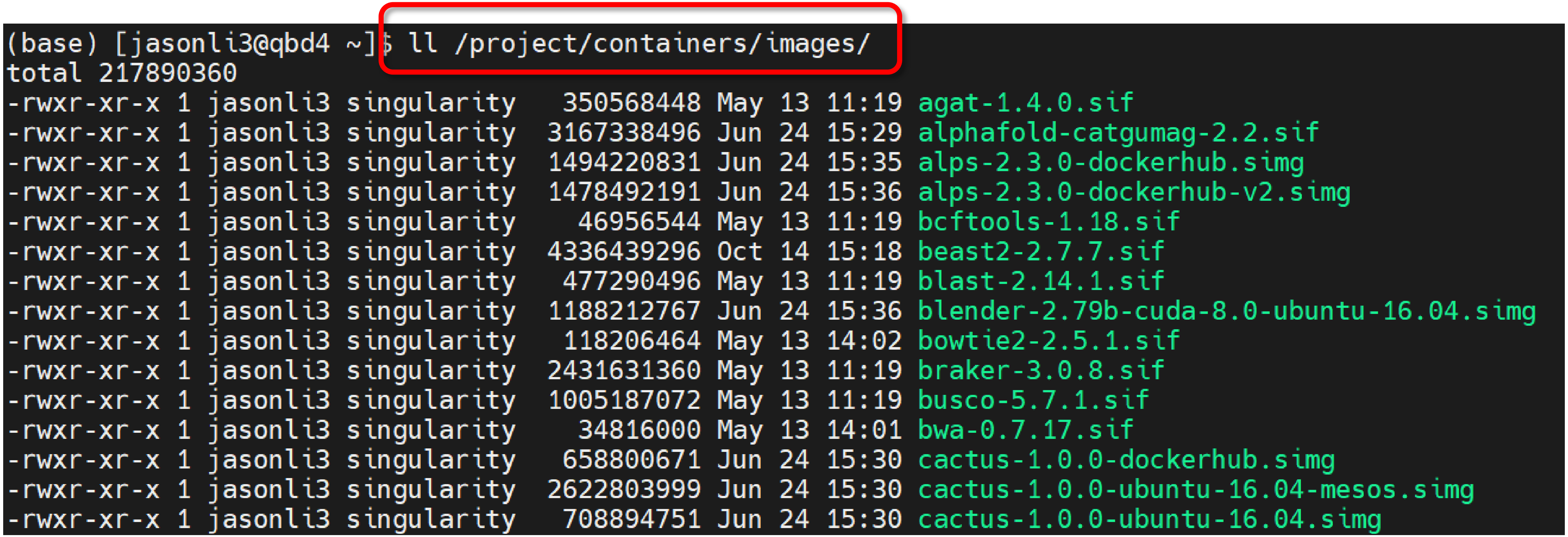
1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

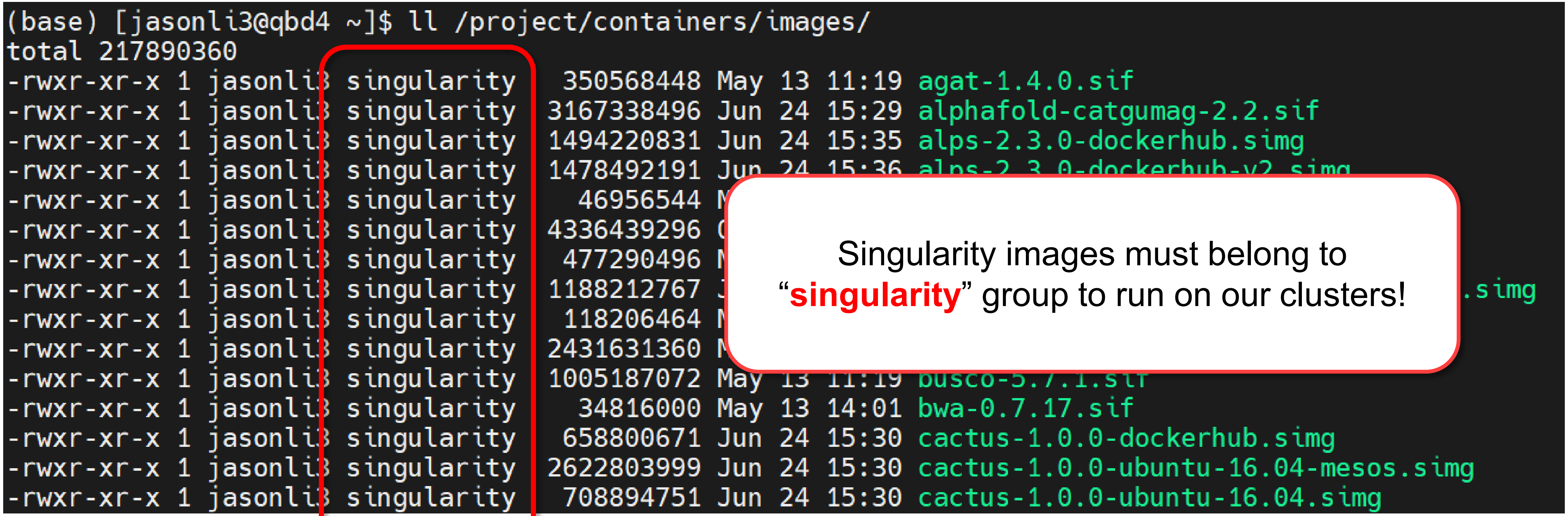
4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

1) What you need



```
(base) [jasonli3@qbd4 ~]$ ll /project/containers/images/
total 217890360
-rwxr-xr-x 1 jasonli3 singularity   350568448 May 13 11:19 agat-1.4.0.sif
-rwxr-xr-x 1 jasonli3 singularity  3167338496 Jun 24 15:29 alphafold-catgumag-2.2.sif
-rwxr-xr-x 1 jasonli3 singularity  1494220831 Jun 24 15:35 alps-2.3.0-dockerhub.simg
-rwxr-xr-x 1 jasonli3 singularity  1478492191 Jun 24 15:36 alps-2.3.0-dockerhub-v2.simg
-rwxr-xr-x 1 jasonli3 singularity    46956544 May 13 11:19 bcftools-1.18.sif
-rwxr-xr-x 1 jasonli3 singularity  4336439296 Oct 14 15:18 beast2-2.7.7.sif
-rwxr-xr-x 1 jasonli3 singularity   477290496 May 13 11:19 blast-2.14.1.sif
-rwxr-xr-x 1 jasonli3 singularity  1188212767 Jun 24 15:36 blender-2.79b-cuda-8.0-ubuntu-16.04.simg
-rwxr-xr-x 1 jasonli3 singularity   118206464 May 13 14:02 bowtie2-2.5.1.sif
-rwxr-xr-x 1 jasonli3 singularity  2431631360 May 13 11:19 braker-3.0.8.sif
-rwxr-xr-x 1 jasonli3 singularity  1005187072 May 13 11:19 busco-5.7.1.sif
-rwxr-xr-x 1 jasonli3 singularity    34816000 May 13 14:01 bwa-0.7.17.sif
-rwxr-xr-x 1 jasonli3 singularity   658800671 Jun 24 15:30 cactus-1.0.0-dockerhub.simg
-rwxr-xr-x 1 jasonli3 singularity  2622803999 Jun 24 15:30 cactus-1.0.0-ubuntu-16.04-mesos.simg
-rwxr-xr-x 1 jasonli3 singularity   708894751 Jun 24 15:30 cactus-1.0.0-ubuntu-16.04.simg
```

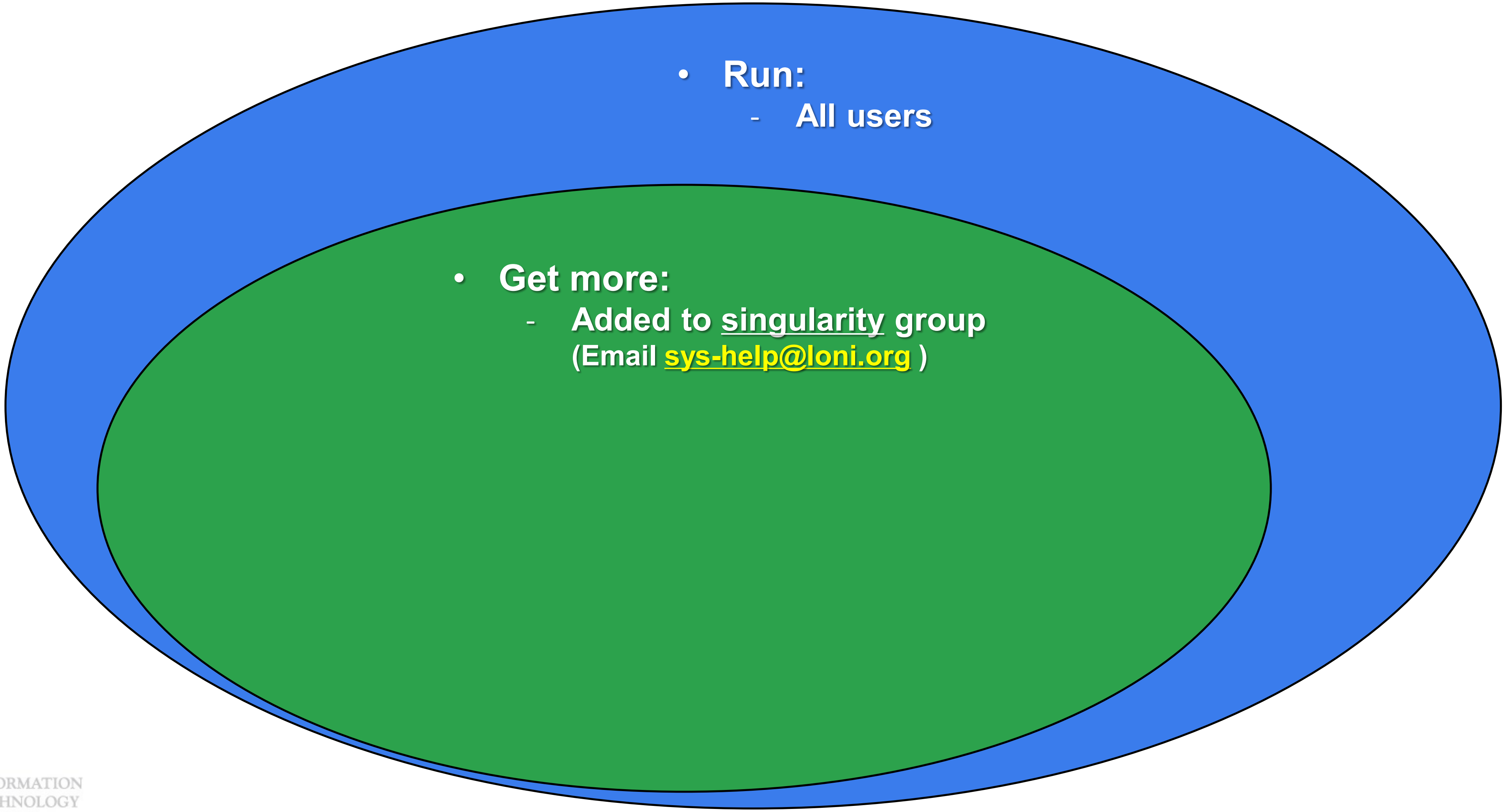
1) What you need



```
(base) [jasonli3@qbd4 ~]$ ll /project/containers/images/
total 217890360
-rwxr-xr-x 1 jasonli3 singularity  350568448 May 13 11:19 agat-1.4.0.sif
-rwxr-xr-x 1 jasonli3 singularity 3167338496 Jun 24 15:29 alphafold-catgumag-2.2.sif
-rwxr-xr-x 1 jasonli3 singularity 1494220831 Jun 24 15:35 alps-2.3.0-dockerhub.simg
-rwxr-xr-x 1 jasonli3 singularity 1478492191 Jun 24 15:36 alps-2.3.0-dockerhub-v2.simg
-rwxr-xr-x 1 jasonli3 singularity   46956544 M
-rwxr-xr-x 1 jasonli3 singularity 4336439296 O
-rwxr-xr-x 1 jasonli3 singularity  477290496 M
-rwxr-xr-x 1 jasonli3 singularity 1188212767 J                                          .simg
-rwxr-xr-x 1 jasonli3 singularity  118206464 M
-rwxr-xr-x 1 jasonli3 singularity 2431631360 M
-rwxr-xr-x 1 jasonli3 singularity 1005187072 May 13 11:19 busco-5.7.1.sif
-rwxr-xr-x 1 jasonli3 singularity   34816000 May 13 14:01 bwa-0.7.17.sif
-rwxr-xr-x 1 jasonli3 singularity  658800671 Jun 24 15:30 cactus-1.0.0-dockerhub.simg
-rwxr-xr-x 1 jasonli3 singularity 2622803999 Jun 24 15:30 cactus-1.0.0-ubuntu-16.04-mesos.simg
-rwxr-xr-x 1 jasonli3 singularity  708894751 Jun 24 15:30 cactus-1.0.0-ubuntu-16.04.simg
```

Singularity images must belong to "**singularity**" group to run on our clusters!

1) What you need



- Run:
  - All users
- Get more:
  - Added to singularity group (Email sys-help@loni.org)

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

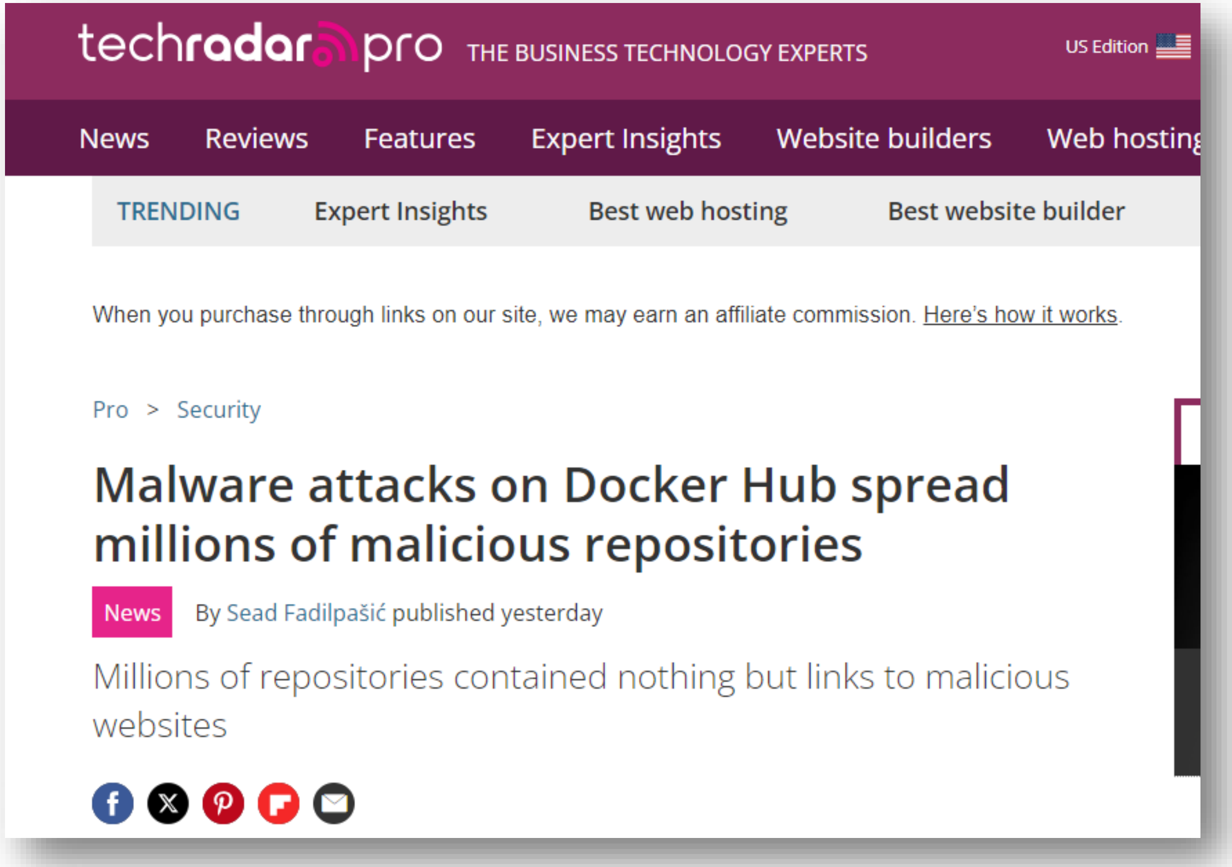
2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

2) Where to get

- You can get container images from a lot of places
  - ➢ Not that you should!
- Concerns?
  - – Reliability
    - Some third-party or untested images may not work
  - – Security risk
    - Some untrustworthy publishers may pack something you don't know about
- Solution
  - – Always get from a source that you can trust !


techradar pro THE BUSINESS TECHNOLOGY EXPERTS
US Edition
News Reviews Features Expert Insights Website builders Web hosting
TRENDING Expert Insights Best web hosting Best website builder

When you purchase through links on our site, we may earn an affiliate commission. Here's how it works.

Pro > Security

Malware attacks on Docker Hub spread millions of malicious repositories

News By Sead Fadilpašić published yesterday

Millions of repositories contained nothing but links to malicious websites

[1] https://www.techradar.com/pro/security/malware-attacks-on-docker-hub-spread-millions-of-malicious-repositories

2) Where to get

- **Tier 1: Developer release (official release)**
  - On software's official website, look for "**Docker**" / "**Singularity**" / "**Container**" / etc.
  - E.g., Tensorflow, Trinity, Salmon

- **Tier 2: Trustworthy third party**

Name	Notes
Biocontainers	• https://biocontainers-edu.readthedocs.io/en/latest/ • For biology
Nvidia NGC	• https://catalog.ngc.nvidia.com/containers • For Nvidia GPU
Bitnami	• https://bitnami.com/stacks/containers • By VmWare
Docker Hub **Quay.io**	• https://hub.docker.com/ & https://quay.io/ • Don't just trust everything you see there! • Look for trustworthy icons like Docker Official Image or ✓ Verified Publisher • Avoid third-party publishers that you don't know

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

3) How to get

- **Steps:**

  a) Step 1: Pull the image

  b) Step 2: Change group ownership

3) How to get

a) Step 1: Pull the image

Syntax		Description
`singularity pull <source>`		Pull an image from source
`<source>`	`docker://container[:tag]` • `(Compare to Docker command)` `docker pull container[:tag]`	Pull a Docker container and convert to Singularity • Many software’s official container release is in Docker form (may or may not on Docker Hub)
	`http://www.myexample.com/container_image.sif`	Download an image file from a web source

3) How to get

a) Step 1: Pull the image

Syntax		Description
`singularity` **`build`** `<target> <source>`		Build an image from source (Advanced)
`<source>`	`docker://container[:tag]`	Build from a Docker container
	`container_image.sif`	Build from a local image file
	`container_sandbox/`	Build from a local **sandbox** (A directory form of a container)
	`container_recipe.def`	Build from a **recipe** (an instruction script of how to build an image)

3) How to get

a) Step 1: Pull the image

Syntax	Description
`singularity pull [options] [target] <source>`	Simple pull
`singularity build [options] <target> <source>`	Advanced build command

3) How to get

b) Step 2: Change group ownership

- What if you do not?

```
FATAL:   singularity image is not owned by required group(s)
```

- To solve it, run this:

```
$ chgrp singularity <container>
```

* You must be added to singularity group to finish this step

3) How to get

- **BONUS: Hot packages!**

i. **PyTorch** (**2.5.0**, w/ GPU support)

```
$ singularity pull docker://pytorch/pytorch:2.5.0-cuda12.4-cudnn9-runtime
```

ii. **Tensorflow** (**2.18.0**, w/ GPU support)

```
$ singularity pull docker://tensorflow/tensorflow:2.18.0-gpu-jupyter
```

3) How to get

- **BONUS: Hot packages!**

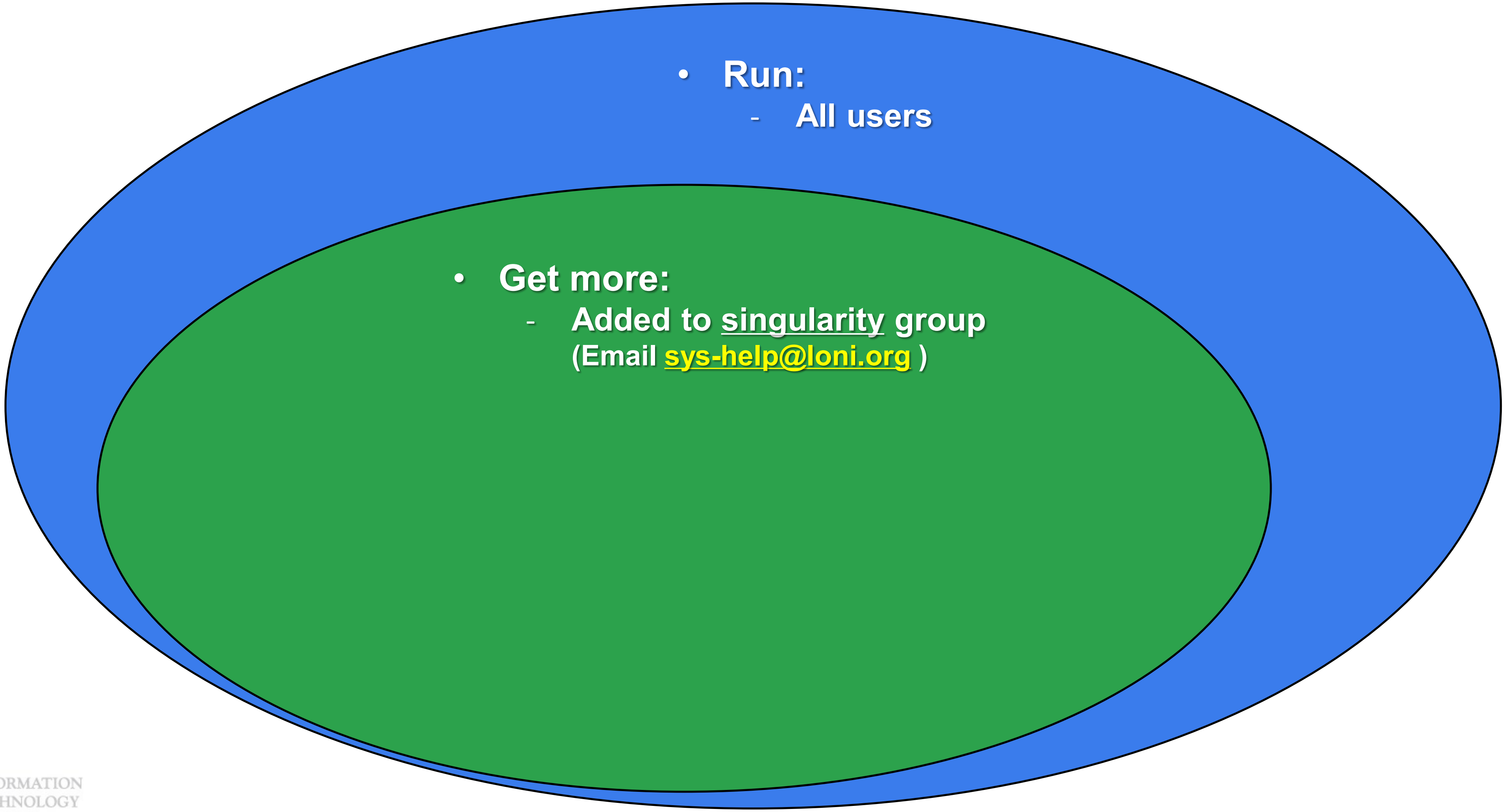
i. **PyTorch** (**2.5.0**, w/ GPU support)

```
$ module load pytorch
```

ii. **Tensorflow** (**2.18.0**, w/ GPU support)

```
$ module load tensorflow
```

Summary



- **Run:**
  - **All users**
- **Get more:**
  - **Added to singularity group (Email sys-help@loni.org)**

Summary

- **Steps:**

a) Step 1: Pull the image

Syntax	Description
`singularity pull [options] [target] <source>`	Simple pull
`singularity build [options] <target> <source>`	Advanced build command

b) Step 2: Change group ownership

Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

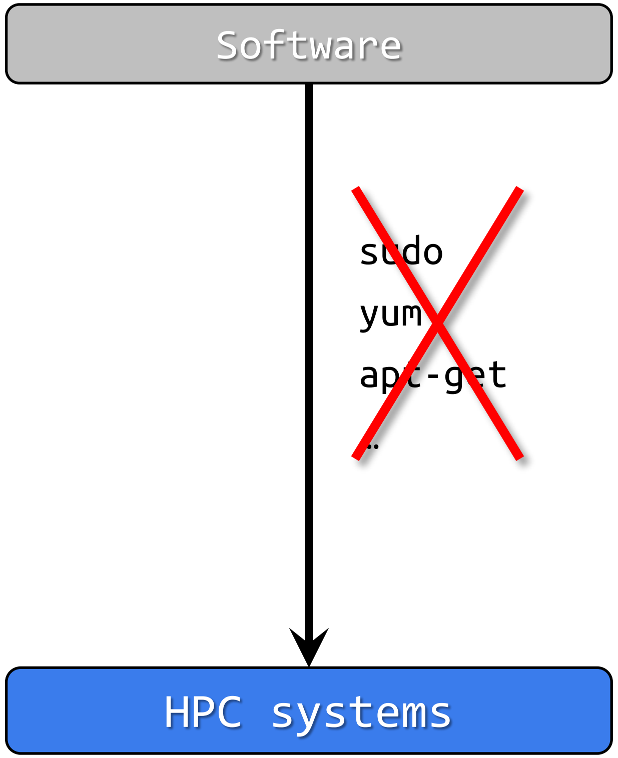
4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

4. Build Your Own Container Image

- **Scenarios:**
  - I did not find any container release. Need to DIY.
  - Installation can be easily done using `sudo apt` or `sudo yum` if I have the permission.
  - I found a container, but need to make changes (e.g., adding something else).

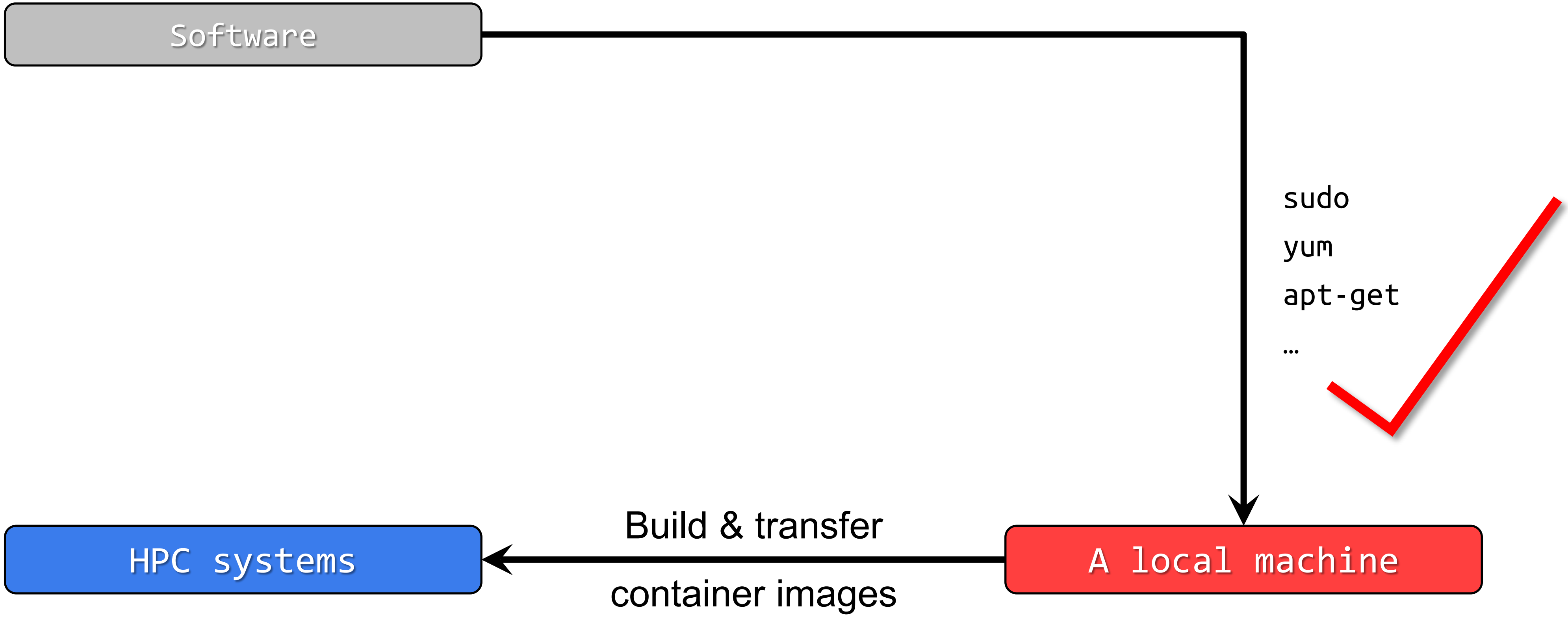
4. Build Your Own Container Image

- **Idea**



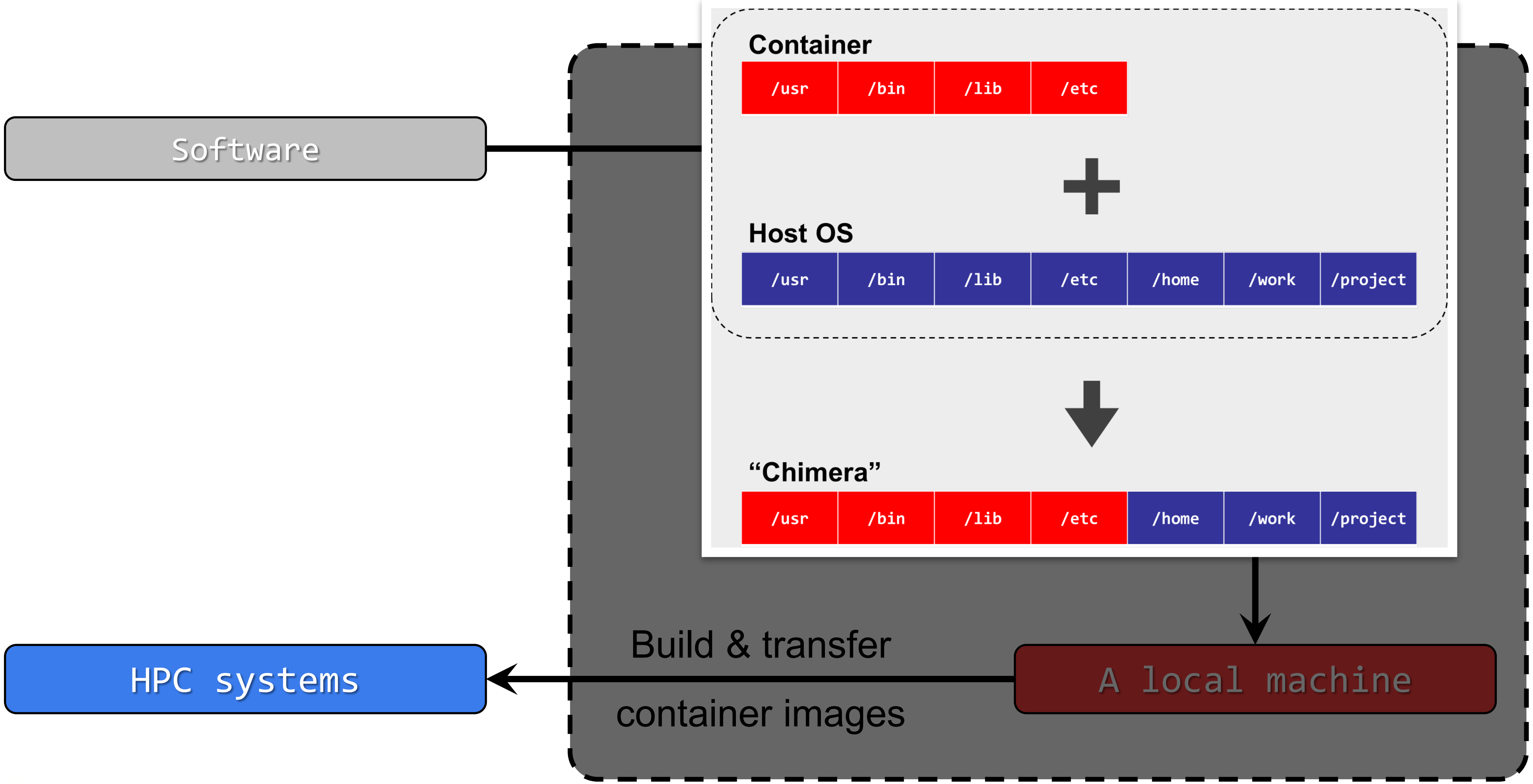
4. Build Your Own Container Image

- **Idea**



4. Build Your Own Container Image

- **Idea**



sudo

yum

apt-get

…

Outlines

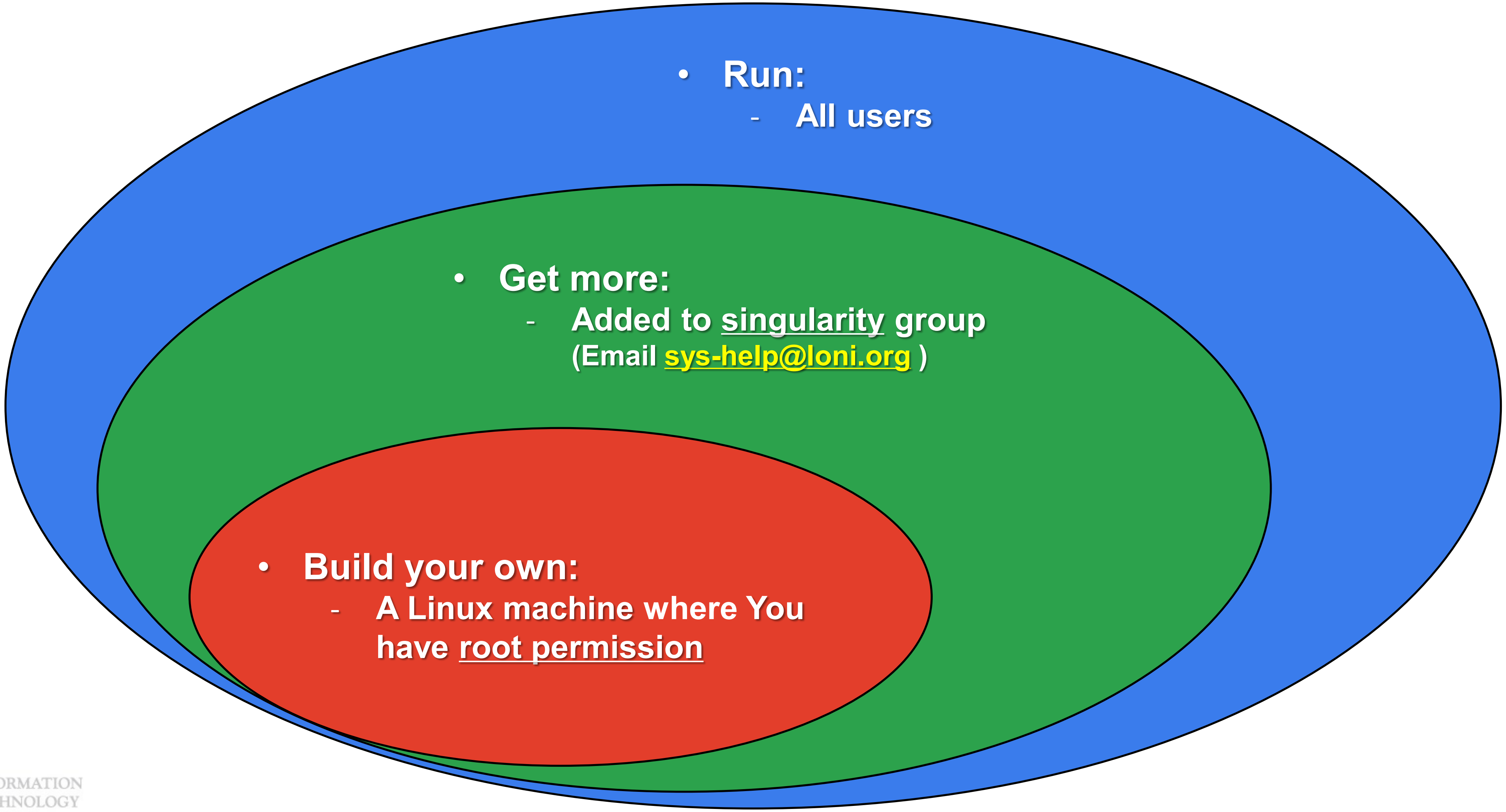
1. Why Container?
   1) Problems
   2) Container & Singularity

2. Run an Existing Container Image
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. Get More Container Images
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) **What you need**
   2) Typical workflow
   3) Make it easier - Recipe

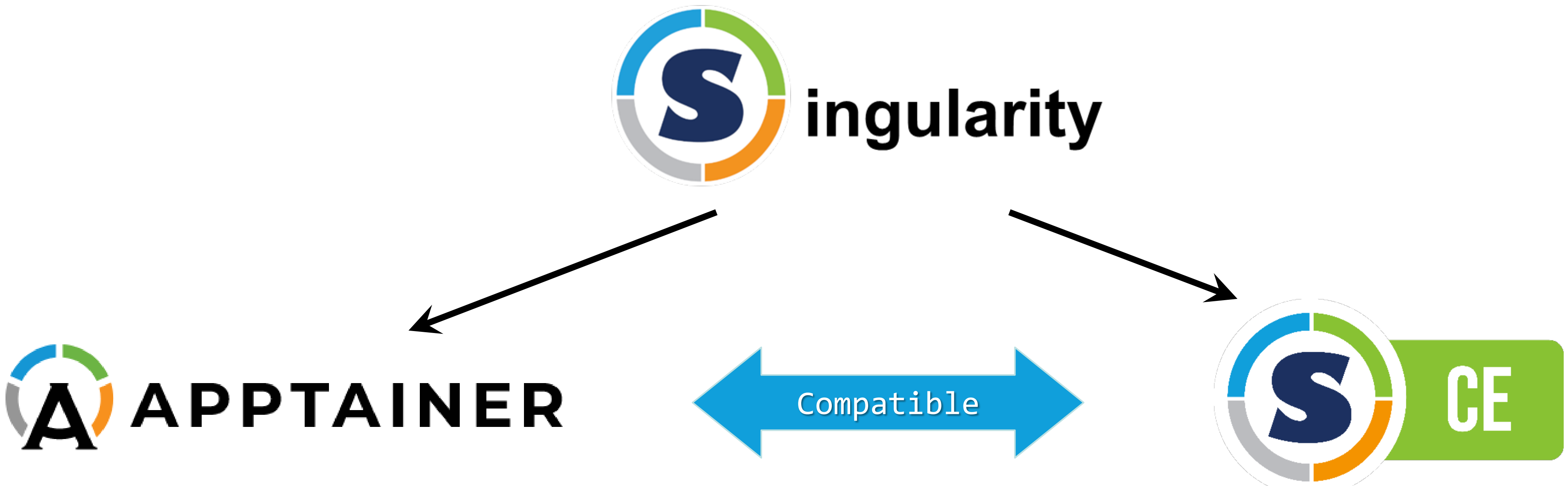
1) What you need



- Run:
  - All users
- Get more:
  - Added to singularity group (Email sys-help@loni.org)
- Build your own:
  - A Linux machine where You have root permission

1) What you need

- **Install Singularity**



- Joined Linux Foundation
- Easier installation

- Community supported
- Installed on our clusters

[1] https://apptainer.org/docs/admin/main/installation.html
[2] https://docs.sylabs.io/guides/3.8/admin-guide/installation.html

Outlines

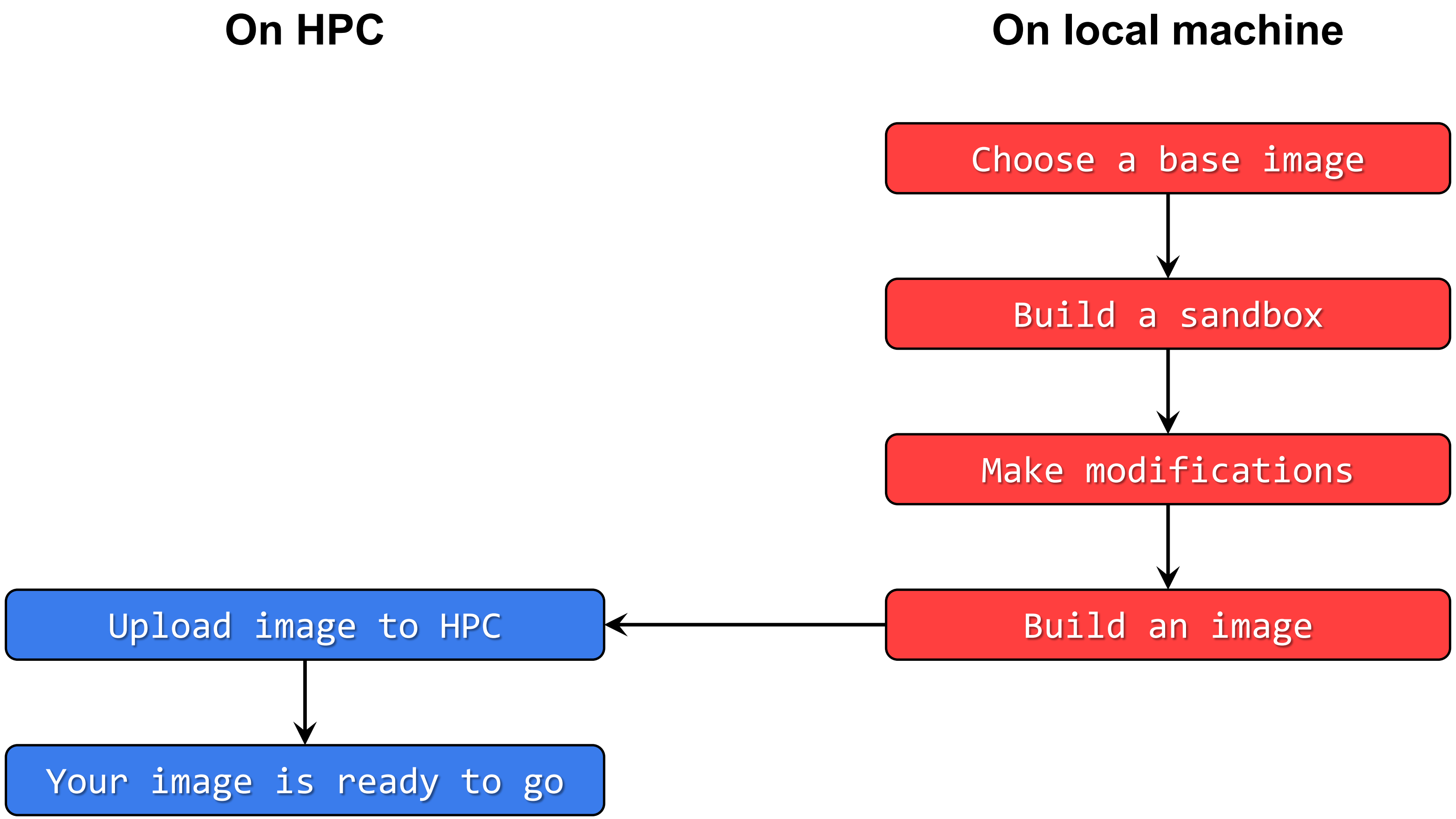
1. **Why Container?**
   1) Problems
   2) Container & Singularity

2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

2) Typical workflow



2) Typical workflow

a) Choose a base image

Common choices	Typical scenarios
A minimum, “mint” OS (e.g., Ubuntu, Rocky, Debian, …)	• You cannot find an existing image with the software you need, and need to install from the scratch. • You need to build a minimum size image
A container with software already installed (e.g., TensorFlow, PyTorch, …)	• You need to modify an existing working image (e.g., add a Python module to Tensorflow image)

2) Typical workflow

b) Build a sandbox

- What's a **sandbox** ?
  - A **directory** form (unlike a single image file) of a container
  - Allows modification

2) Typical workflow

b) Build a sandbox

```
$ singularity build               [options] <target> <source>
```

`<source>`	`docker://container[:tag]`	Build from a Docker container
	`container_image.sif`	Build from a local image file
	`container_sandbox/`	Build from a local **sandbox** (A directory form of a container)
	`container_recipe.def`	Build from a **recipe** (an instruction script of how to build an image)

2) Typical workflow

b) Build a sandbox

```
$ singularity build --sandbox [options] <target> <source>
```

<source>	docker://container[:tag]	Build from a Docker container
	container_image.sif	Build from a local image file
	container_sandbox/	Build from a local **sandbox** (A directory form of a container)
	container_recipe.def	Build from a **recipe** (an instruction script of how to build an image)

2) Typical workflow

c) Make modifications

```
$       singularity shell            [options] <container>
```

2) Typical workflow

c) Make modifications

```
$       singularity shell --writable [options] <container>
```

i. Allows **writing** to the sandbox

   – Without it, just like running a regular container image

2) Typical workflow

c) Make modifications

```
$ sudo singularity shell --writable [options] <container>
```

ii. Run the container as **root**

- Grants root privilege in container
- Needed in most cases
- Technically not required, but cannot run things like `sudo apt` or `sudo yum` without it

i. Allows **writing** to the sandbox

- Without it, just like running a regular container image

2) Typical workflow

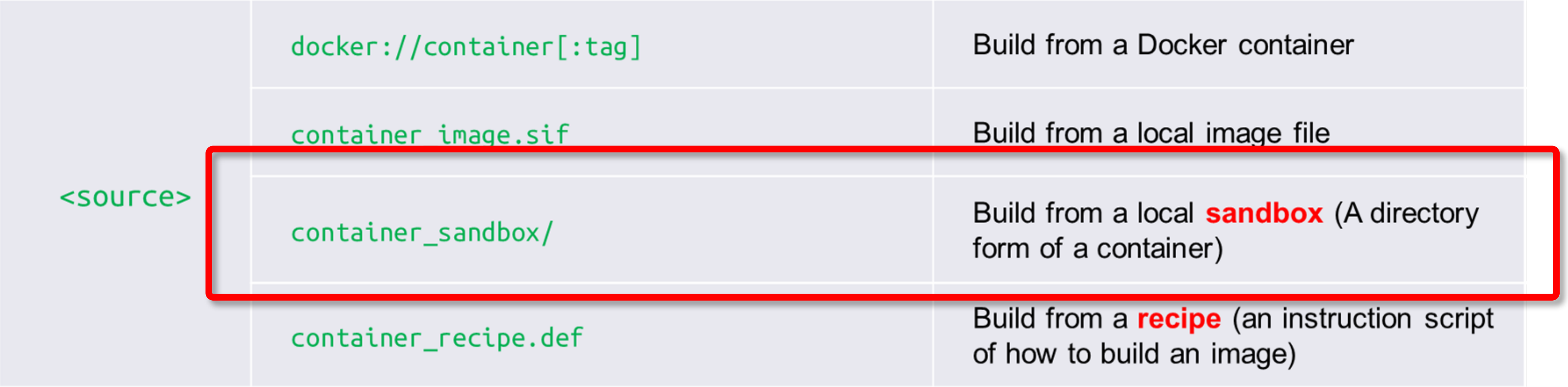
c) Make modifications

```
$ sudo singularity shell --writable [options] <container>
Singularity>
Singularity> apt update
Singularity> apt install ...
```

2) Typical workflow

d) Build an image from sandbox

```
$      singularity build [options] <target> <source>
```



<source>	docker://container[:tag]	Build from a Docker container
	container_image.sif	Build from a local image file
	container_sandbox/	Build from a local **sandbox** (A directory form of a container)
	container_recipe.def	Build from a **recipe** (an instruction script of how to build an image)

2) Typical workflow

d) Build an image from sandbox

```
$ sudo singularity build [options] <target> <source>
```

Modify with “**sudo**”? Build with “**sudo**”!

2) Typical workflow

- **Quick recap**

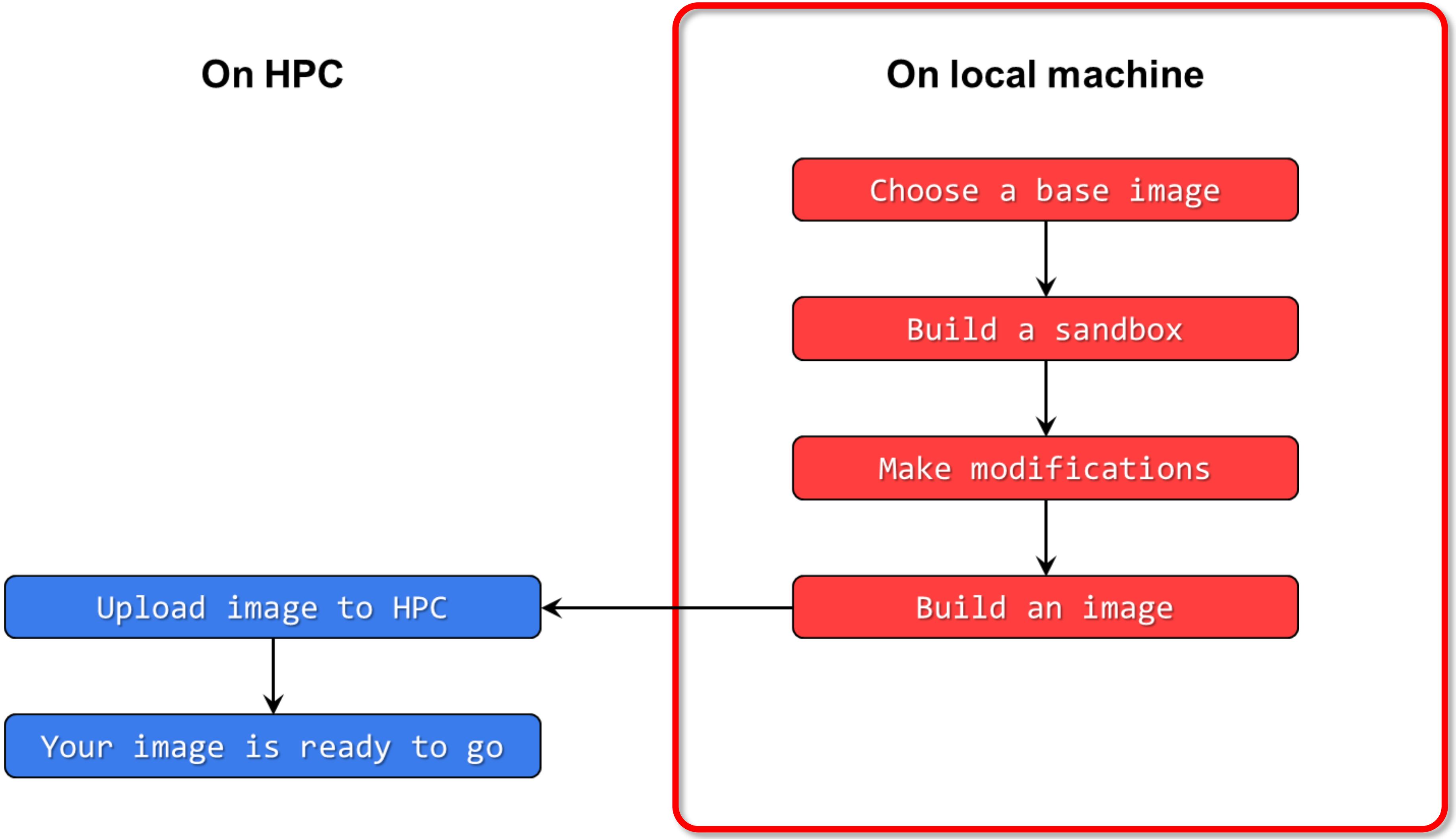
To …	You need to …
Build a sandbox	`$  singularity build --sandbox ...`
Modify a sandbox	`$  sudo singularity shell --writable ...`
Build an image from sandbox	`$  sudo singularity build ...`

2) Typical workflow

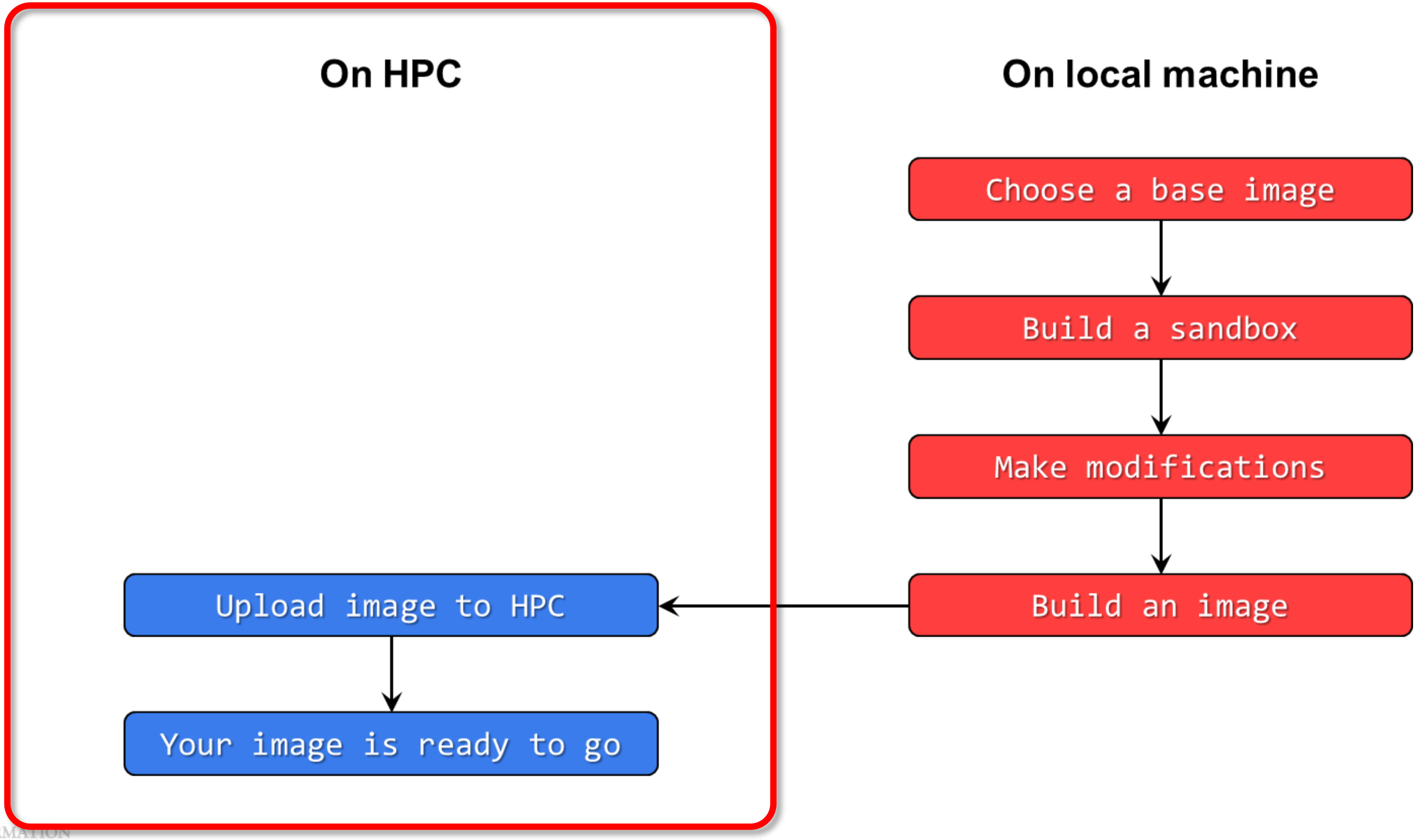
e) Upload image to HPC and run

Now! The moment of truth!

2) Typical workflow



2) Typical workflow



Outlines

1. **Why Container?**
   1) Problems
   2) Container & Singularity

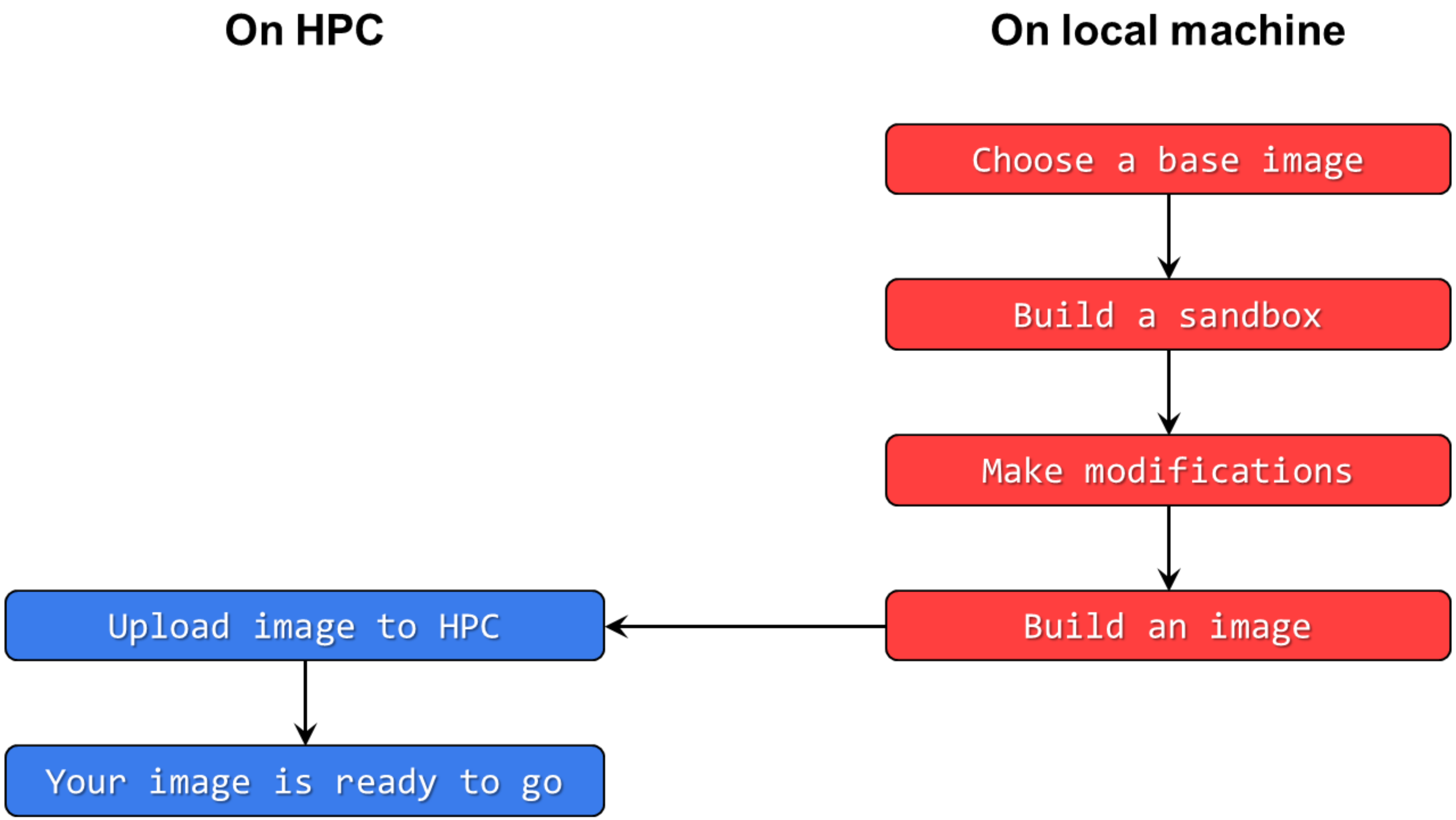
2. **Run an Existing Container Image**
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

3. **Get More Container Images**
   1) What you need
   2) Where to get
   3) How to get

4. **Build Your Own Container Image**
   1) What you need
   2) Typical workflow
   3) Make it easier - Recipe

3) Make it easier - Recipe

- **Why?**



3) Make it easier - Recipe

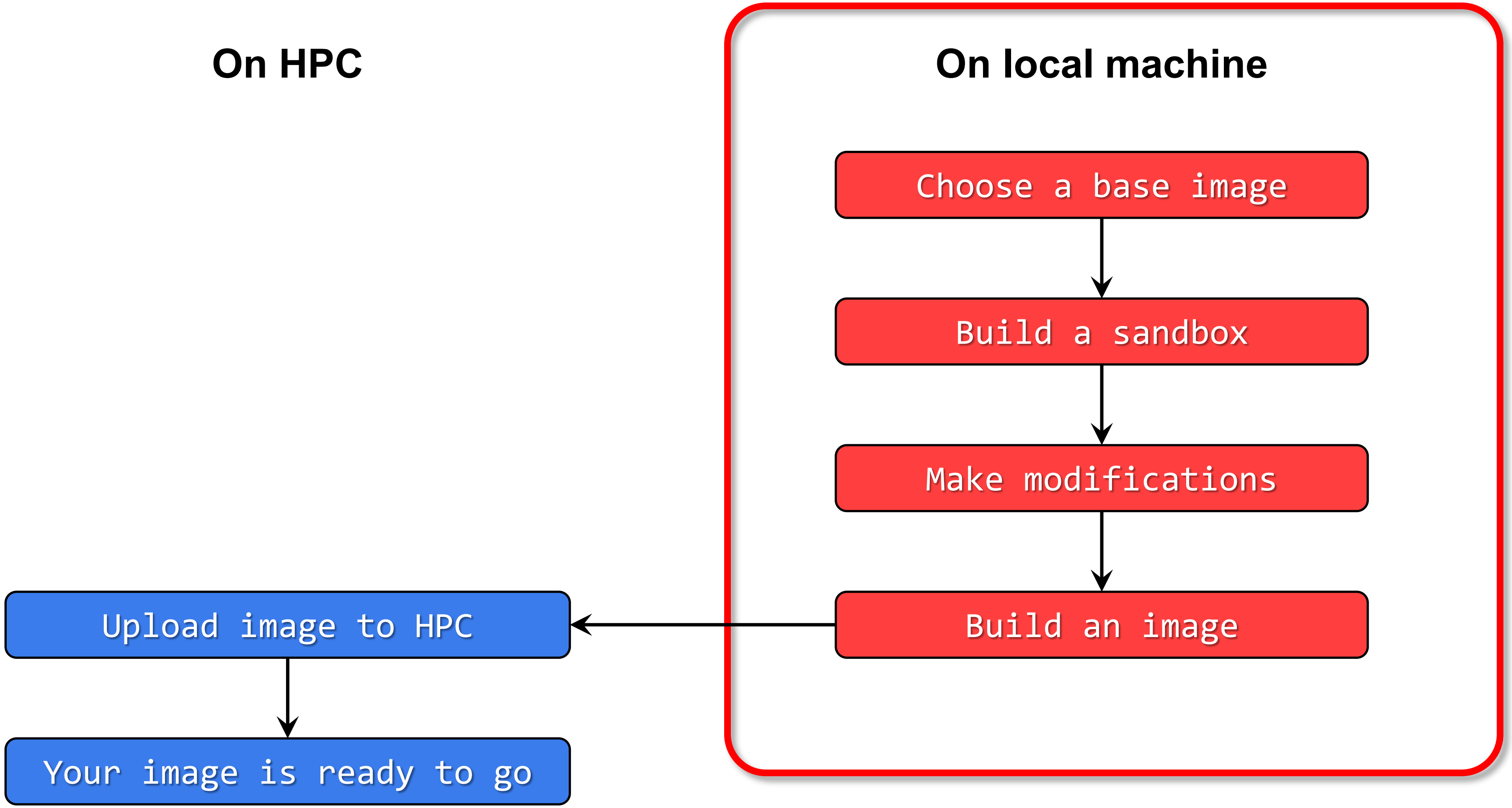
- **Why?**

Pros	Cons
• Flexibility	• Repeatability • Minimizing image size

- **Solution:**
  - **Recipe**: A text file containing instructions to build a container

3) Make it easier - Recipe

- **Why?**



On HPC

On local machine

Choose a base image

Build a sandbox

Make modifications

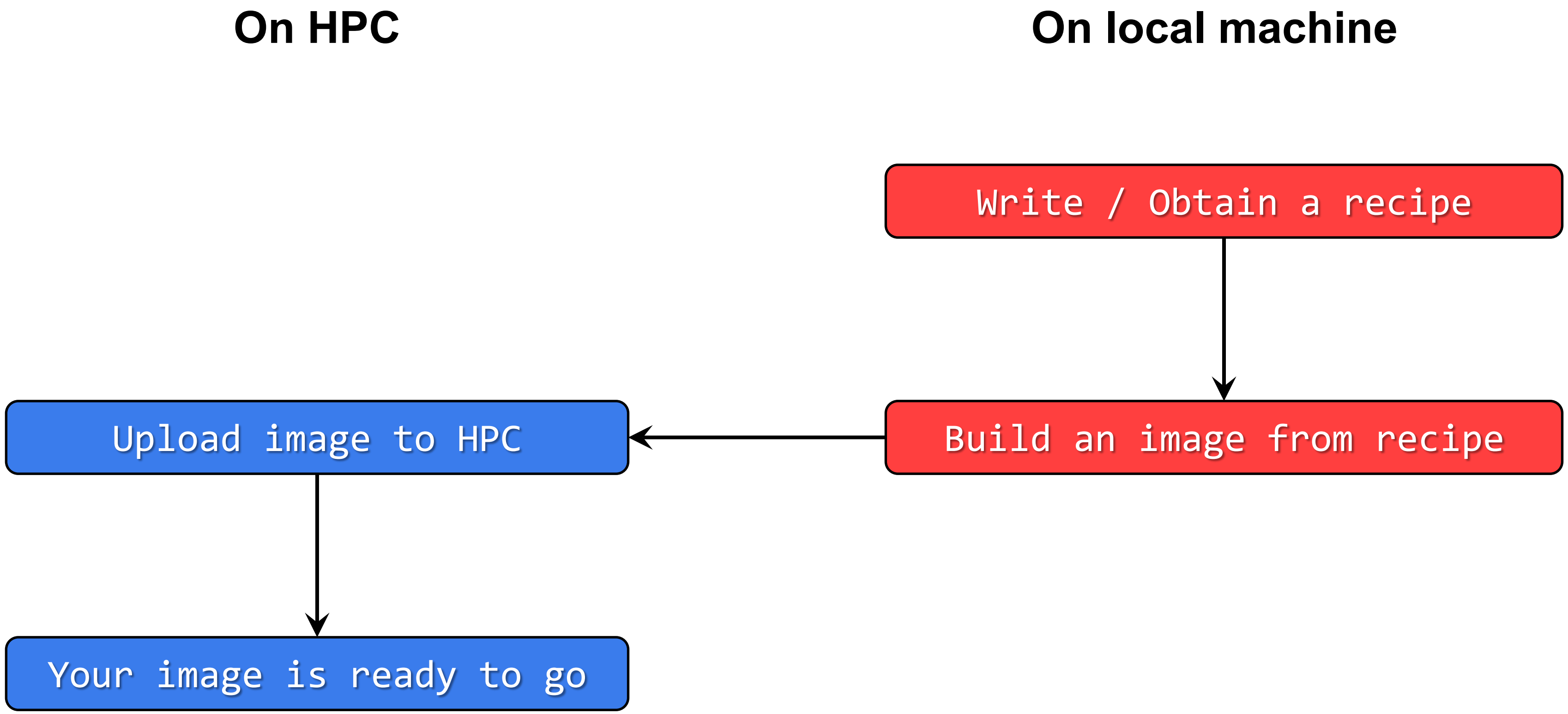
Build an image

Upload image to HPC

Your image is ready to go

3) Make it easier - Recipe

- **Why?**



3) Make it easier - Recipe

a) What does a recipe look like?

ruby.def

```
BootStrap: docker
From: ubuntu:latest

%labels
Author       Jason Li
Description  A container with Ruby installed

%post
apt update
apt install -y ruby

%environment
export MYENV="Some environmental variable"

%runscript
ruby -e "puts 'Hello from container!'"
```

3) Make it easier - Recipe

a) What does a recipe look like?

ruby.def

```
BootStrap: docker
From: ubuntu:latest

%labels
Author       Jason Li
Description  A container with Ruby installed

%post
apt update
apt install -y ruby

%environment
export MYENV="Some environmental variable"

%runscript
ruby -e "puts 'Hello from container!'"
```

Header

- Base image info (how, where, what to pull)

3) Make it easier - Recipe

a) What does a recipe look like?

ruby.def

```
BootStrap: docker
From: ubuntu:latest

%labels
Author       Jason Li
Description  A container with Ruby installed

%post
apt update
apt install -y ruby

%environment
export MYENV="Some environmental variable"

%runscript
ruby -e "puts 'Hello from container!'"
```

Label

- Container information (write whatever you want)

3) Make it easier - Recipe

a) What does a recipe look like?

ruby.def

```
BootStrap: docker
From: ubuntu:latest

%labels
Author       Jason Li
Description  A container with Ruby installed

%post
apt update
apt install -y ruby

%environment
export MYENV="Some environmental variable"

%runscript
ruby -e "puts 'Hello from container!'"
```

Post

- Commands to execute after the base image is pulled

3) Make it easier - Recipe

a) What does a recipe look like?

ruby.def

```
BootStrap: docker
From: ubuntu:latest

%labels
Author       Jason Li
Description  A container with Ruby installed

%post
apt update
apt install -y ruby

%environment
export MYENV="Some environmental variable"

%runscript
ruby -e "puts 'Hello from container!'"
```

Environment
- Define environmental variables every time the container is executed

3) Make it easier - Recipe

a) What does a recipe look like?

ruby.def

```
BootStrap: docker
From: ubuntu:latest

%labels
Author        Jason Li
Description   A container with Ruby installed

%post
apt update
apt install -y ruby

%environment
export MYENV="Some environmental variable"

%runscript
ruby -e "puts 'Hello from container!'"
```

Runscript

- Commands to be run with `singularity run`

3) Make it easier - Recipe

a) What does a recipe look like?

ruby.def

```
BootStrap: docker
From: ubuntu:latest

%labels
Author        Jason Li
Description   A container with Ruby installed

%post
apt update
apt install -y ruby

%environment
export MYENV="Some environmental variable"

%runscript
ruby -e "puts 'Hello from container!'"
```

3) Make it easier - Recipe

a) What does a recipe look like?

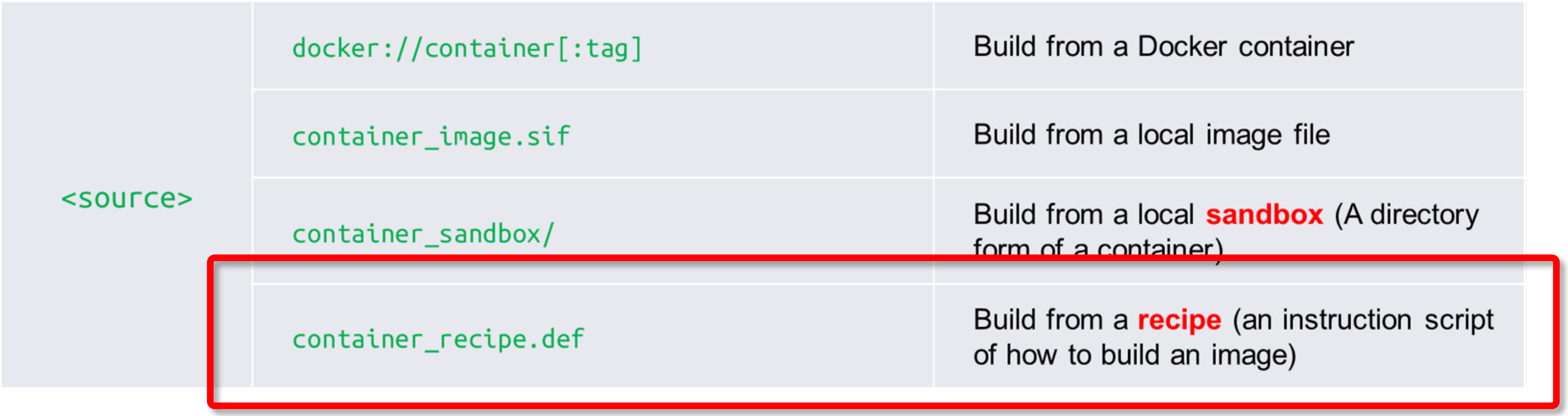
ruby.def

```
BootStrap: docker
From: ubuntu:latest

%labels
Author       Jason Li
Description  A container with Ruby installed

%post
apt update
apt install -y ruby

%environment
export MYENV="Some environmental variable"

%runscript
ruby -e "puts 'Hello from container!'"
```

3) Make it easier - Recipe

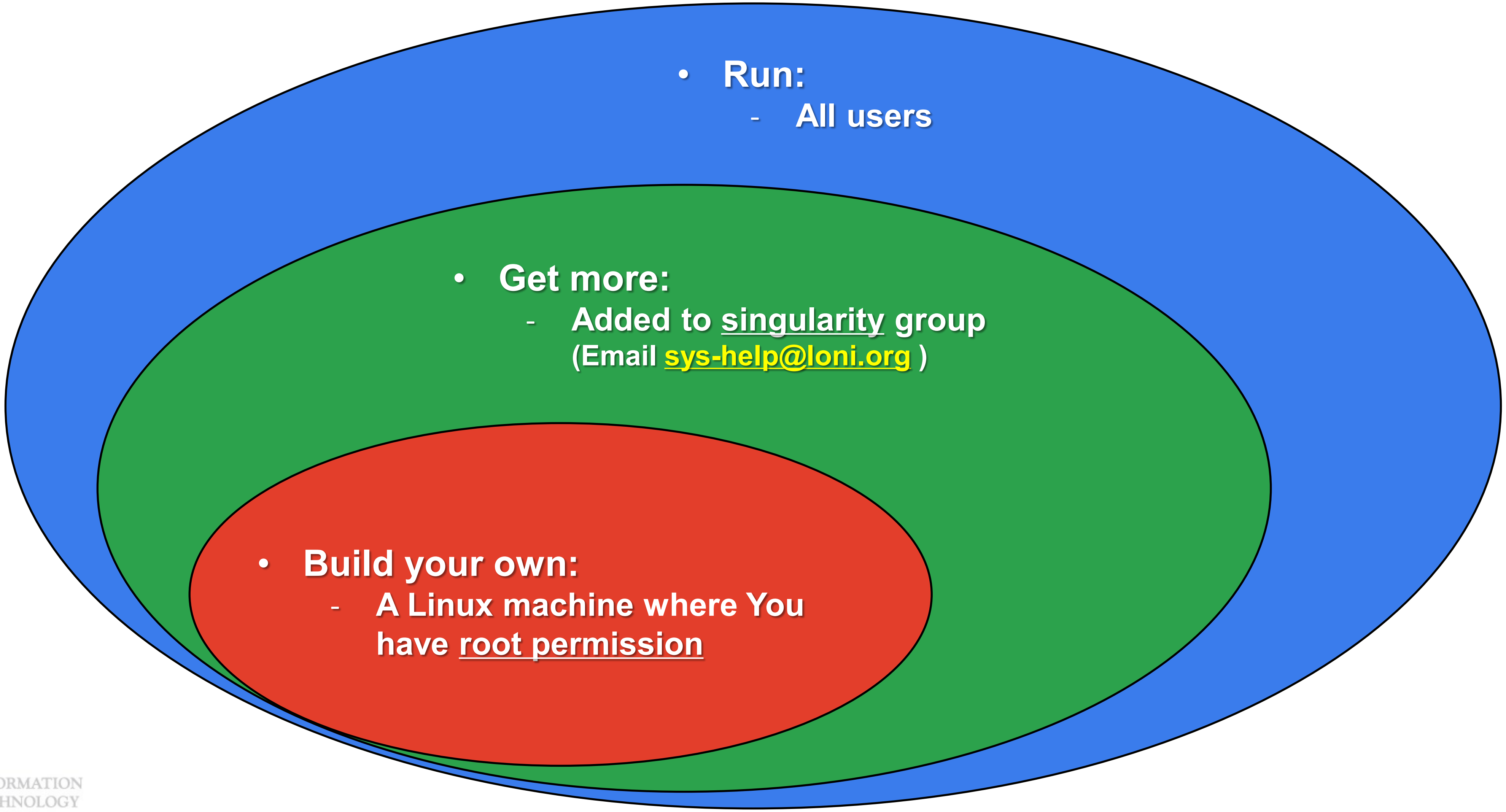
b) Build the recipe

```
$ singularity build [options] <target> <source>
```



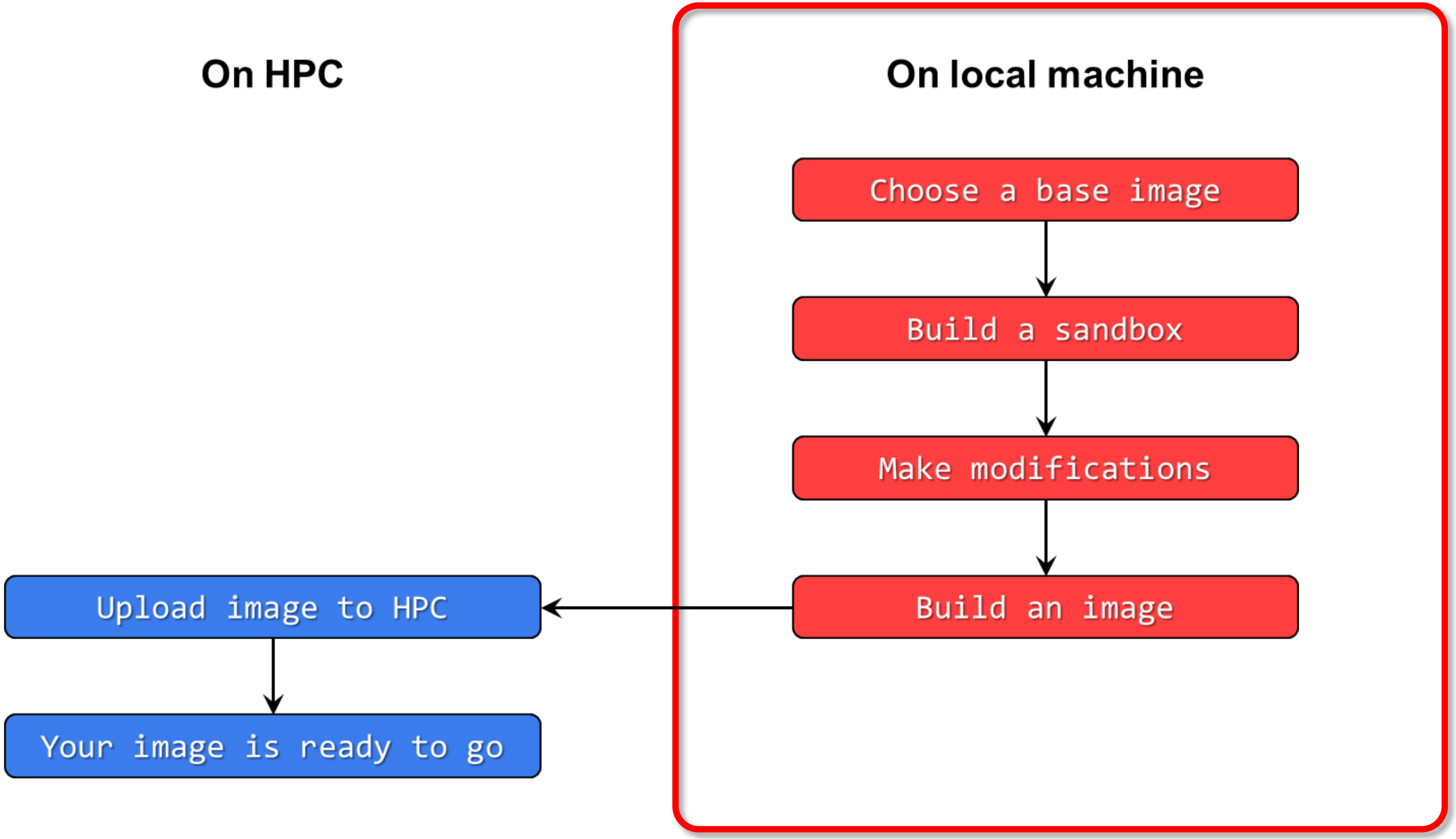
<source>	docker://container[:tag]	Build from a Docker container
	container_image.sif	Build from a local image file
	container_sandbox/	Build from a local **sandbox** (A directory form of a container)
	container_recipe.def	Build from a **recipe** (an instruction script of how to build an image)

Summary



- Run:
  - All users
- Get more:
  - Added to singularity group (Email sys-help@loni.org)
- Build your own:
  - A Linux machine where You have root permission

Summary



On HPC

On local machine

Choose a base image

Build a sandbox

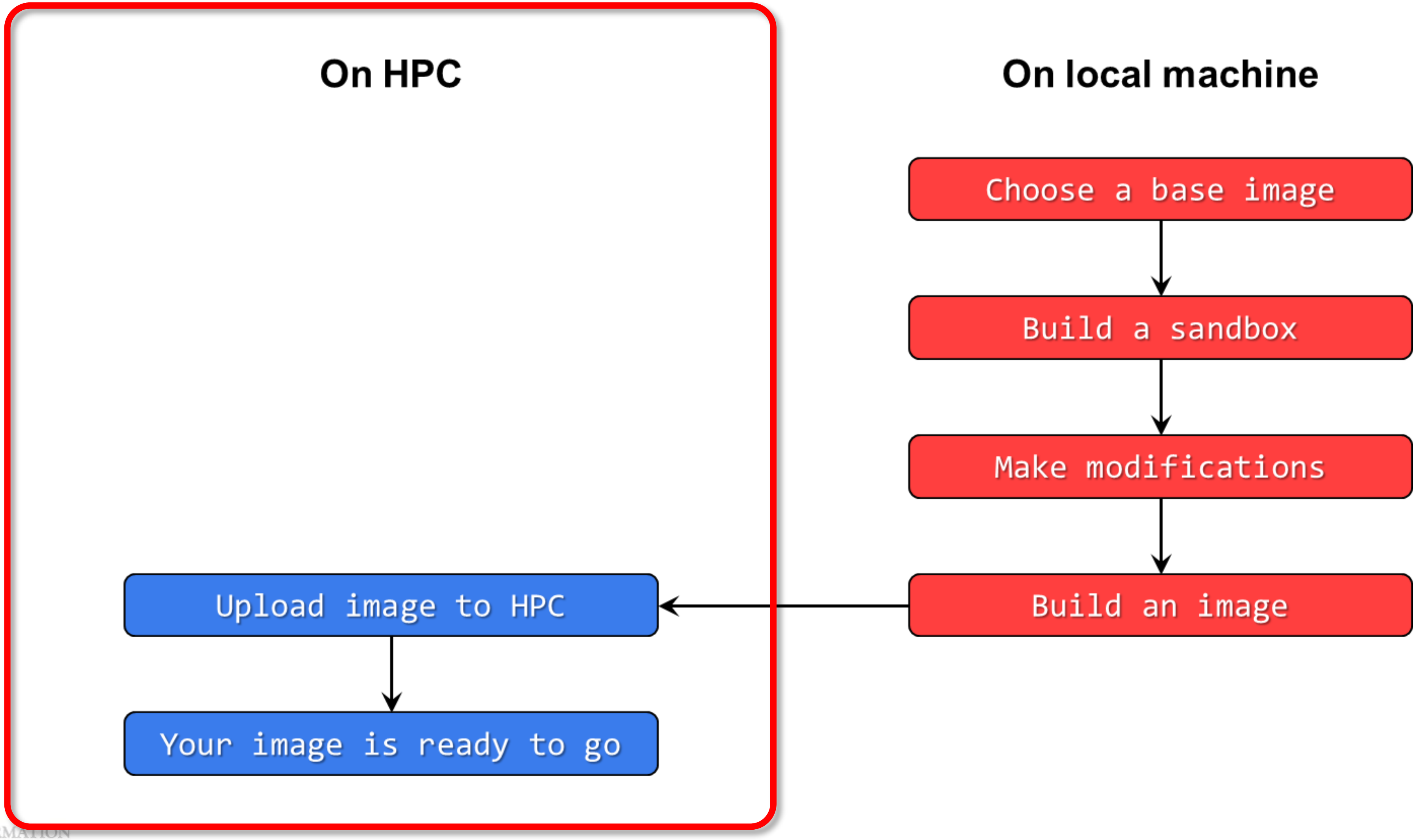
Make modifications

Build an image

Upload image to HPC

Your image is ready to go

Summary

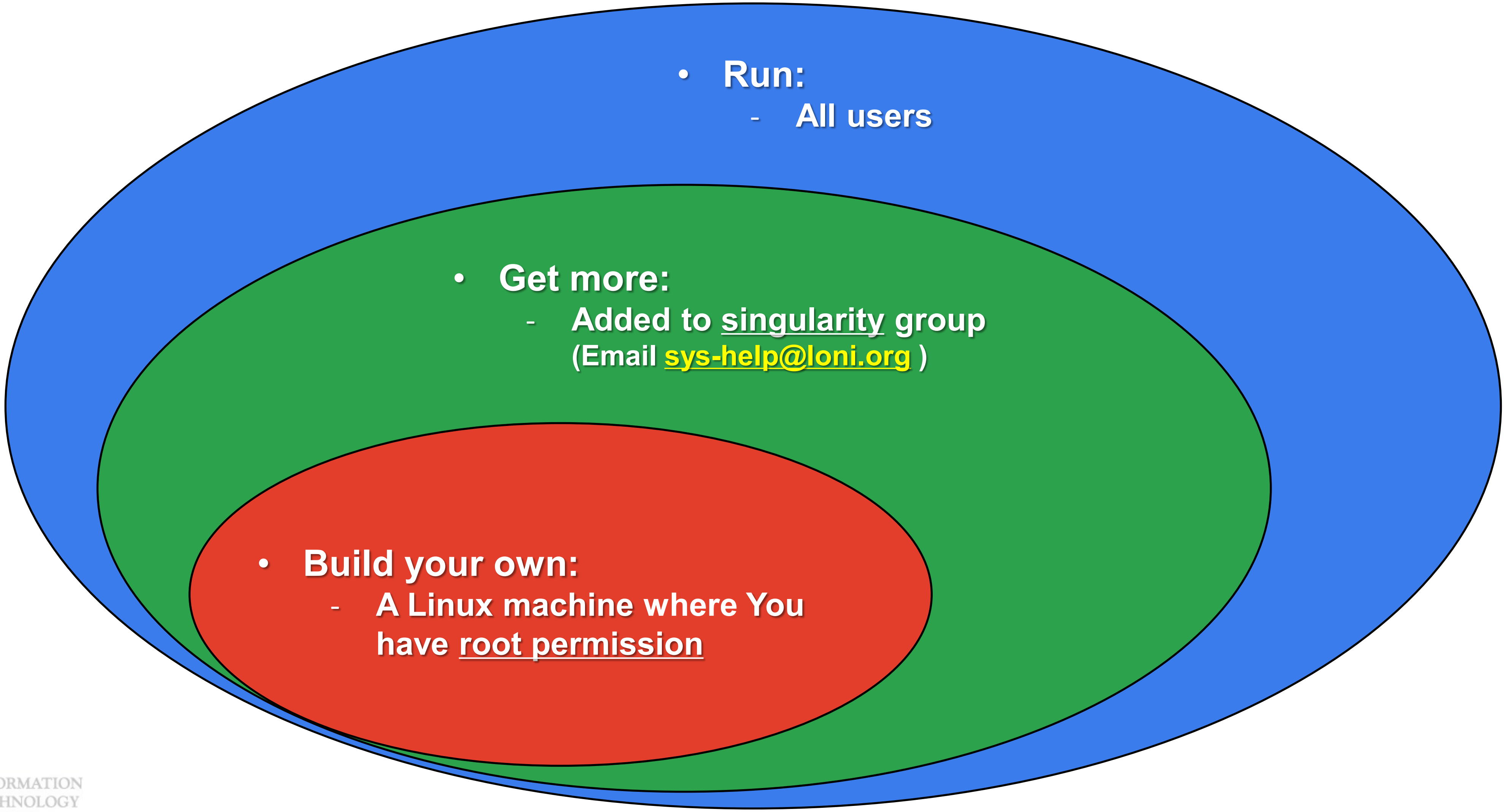

On HPC
On local machine
Choose a base image
Build a sandbox
Make modifications
Build an image
Upload image to HPC
Your image is ready to go

Conclusion

Conclusion

1. **Why Container?**
    1) Problems
    2) Container & Singularity
2. **Run an Existing Container Image**
    1) What you need
    2) Basic commands
    3) Running jobs with Singularity
3. **Get More Container Images**
    1) What you need
    2) Where to get
    3) How to get
4. **Build Your Own Container Image**
    1) What you need
    2) Typical workflow
    3) Make it easier - Recipe

Take home message

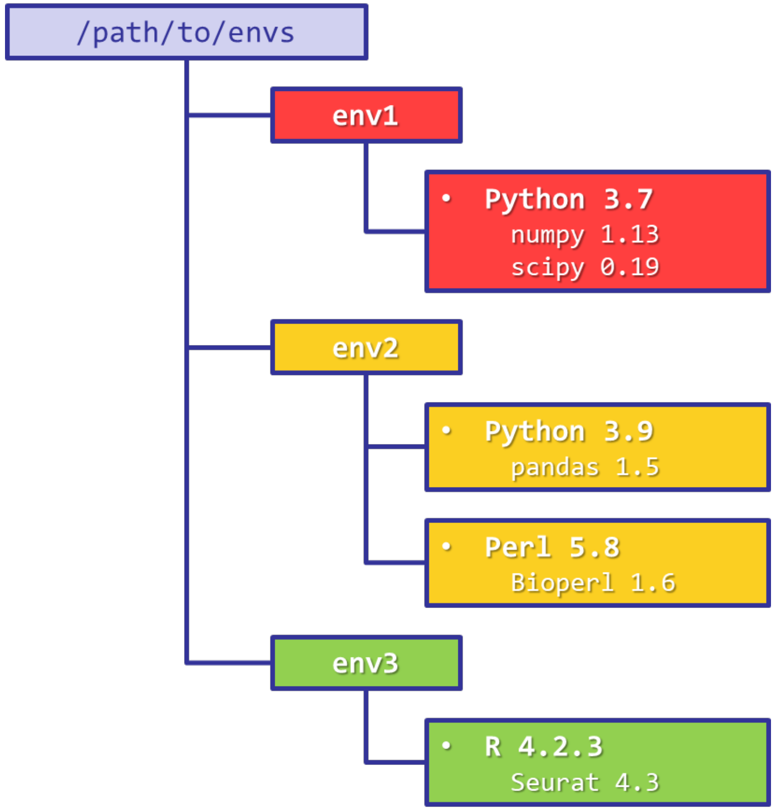


- Run:
  - All users
- Get more:
  - Added to singularity group (Email sys-help@loni.org)
- Build your own:
  - A Linux machine where You have root permission

To conclude our mini series…

Conda vs Singularity

- **Virtual Environment** v.s. **Container** ?



/path/to/envs

env1

- Python 3.7
  numpy 1.13
  scipy 0.19

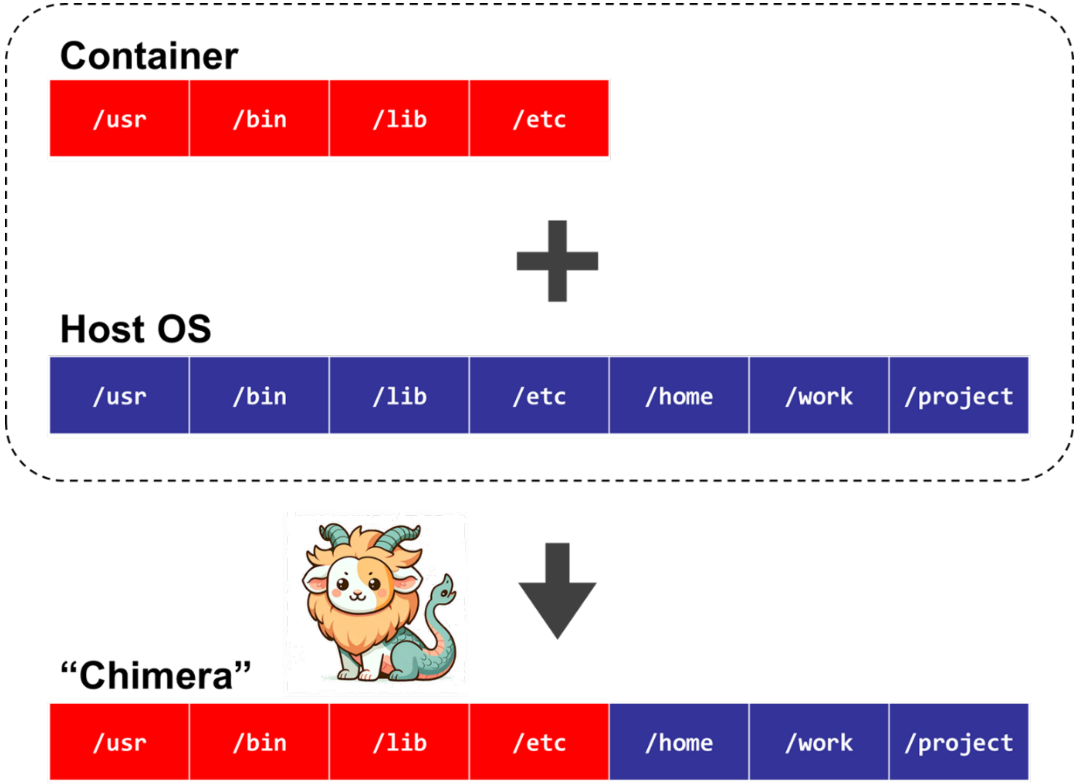
env2

- Python 3.9
  pandas 1.5

- Perl 5.8
  Bioperl 1.6

env3

- R 4.2.3
  Seurat 4.3



Container

/usr /bin /lib /etc

Host OS

/usr /bin /lib /etc /home /work /project

“Chimera”

/usr /bin /lib /etc /home /work /project

Conda vs Singularity

	Conda / Virtual Environments	Singularity / Containers
Availability	All users	All users, but may need additional things
Self-contained	Yes	Yes
Isolated	Yes (but still accessible from outside)	Perfect (completely isolated from outside)
Editability	Yes	No (Must create a new image)
Disk usage	Large	Smaller
Portability	Possible (but .yml may not work)	Great (just copy-paste one file)
Security	Fair	Good
Ease of use	Good	May require a little more understanding

Conda vs Singularity

	Conda / Virtual Environments	Singularity / Containers
Good for	• Less hassle to create and install software from scratch • If you need to frequently make modifications	• Less hassle if the developer releases a working container • If you don’t need to make changes after it is created • Portability • Reduce disk usage • Your system admins yelled at you about security risk

Contact us

- **Contact user services**
  - Email Help Ticket: sys-help@loni.org
  - Telephone Help Desk: +1 (225) 578-0900

“Commercial” time!

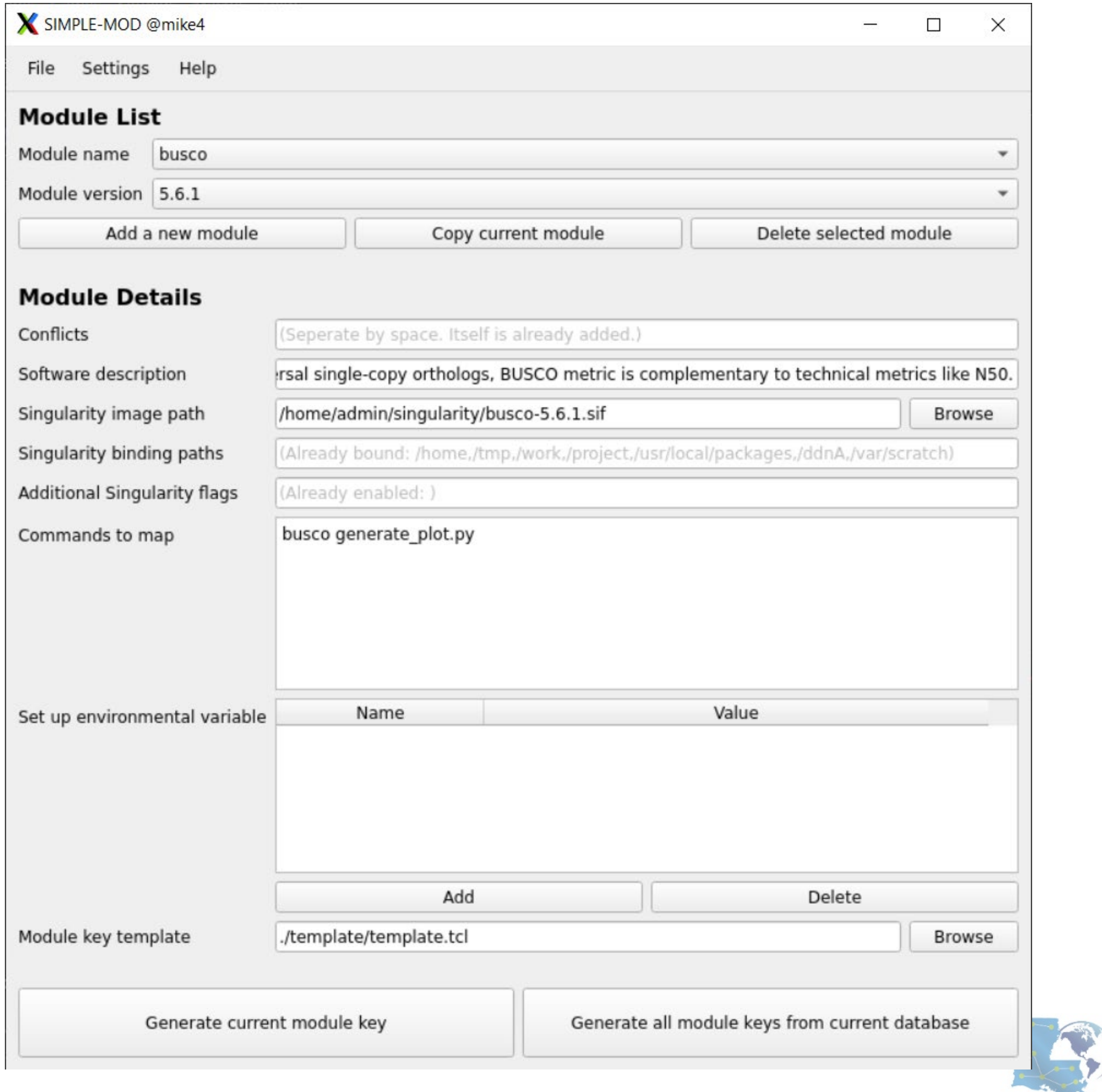
- **Are you tired of wring the long, tedious singularity commands?**

```
$ singularity exec --nv -B /work,/project,/usr/local/package \
    /home/admin/singularity/ubuntu-training.sif \
    python helloworld.py
```

“Commercial” time!

- Try **SIMPLE-MOD** !
  - https://github.com/lsuhpchelp/SIMPLE-MOD
  - A GUI tool to create module key from container-based software.
  - Using the software in containers is as easy as:

```
$ module load busco
$ busco --version
  BUSCO 5.6.1
```



Module List

Module name: busco

Module version: 5.6.1

Module Details

Conflicts: (Seperate by space. Itself is already added.)

Software description: rsal single-copy orthologs, BUSCO metric is complementary to technical metrics like N50.

Singularity image path: /home/admin/singularity/busco-5.6.1.sif

Singularity binding paths: (Already bound: /home,/tmp,/work,/project,/usr/local/packages,/ddnA,/var/scratch)

Additional Singularity flags: (Already enabled:)

Commands to map: busco generate_plot.py

Set up environmental variable:

Name	Value

Module key template: ./template/template.tcl

LSU INFORMATION TECHNOLOGY SERVICES