

Magic Tools to Install & Manage Software



Siva Prasad Kasetti

HPC User Services

LSU HPC / LONI

sys-help@loni.org

Louisiana State University, Baton Rouge

March 12, 2025

Magic Tools to Install & Manage Software



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

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. 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. 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

- Core problem:

Installing software on an HPC system

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

a) Dependencies (Welcome to Linux!)

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.

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.

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.

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.



```
• Dependencies
The following dependencies are required for AUGUSTUS:
  ◦ for gzip compressed input: (set ZIPINPUT = false in common.mk if the required library is not available)
    ▪ libboost-iostreams-dev
    ▪ zlib1g-dev
  ◦ for comparative AUGUSTUS (multi-species, CGP): (set COMPGENEPRED = false in common.mk if the required libraries 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
    ▪ liblsolve55-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 utrnanseq:
    ▪ 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)
```

b) Permission denied (Welcome to HPC!)

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

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'
```

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-co
```

libgdal

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
```


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
```

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
```

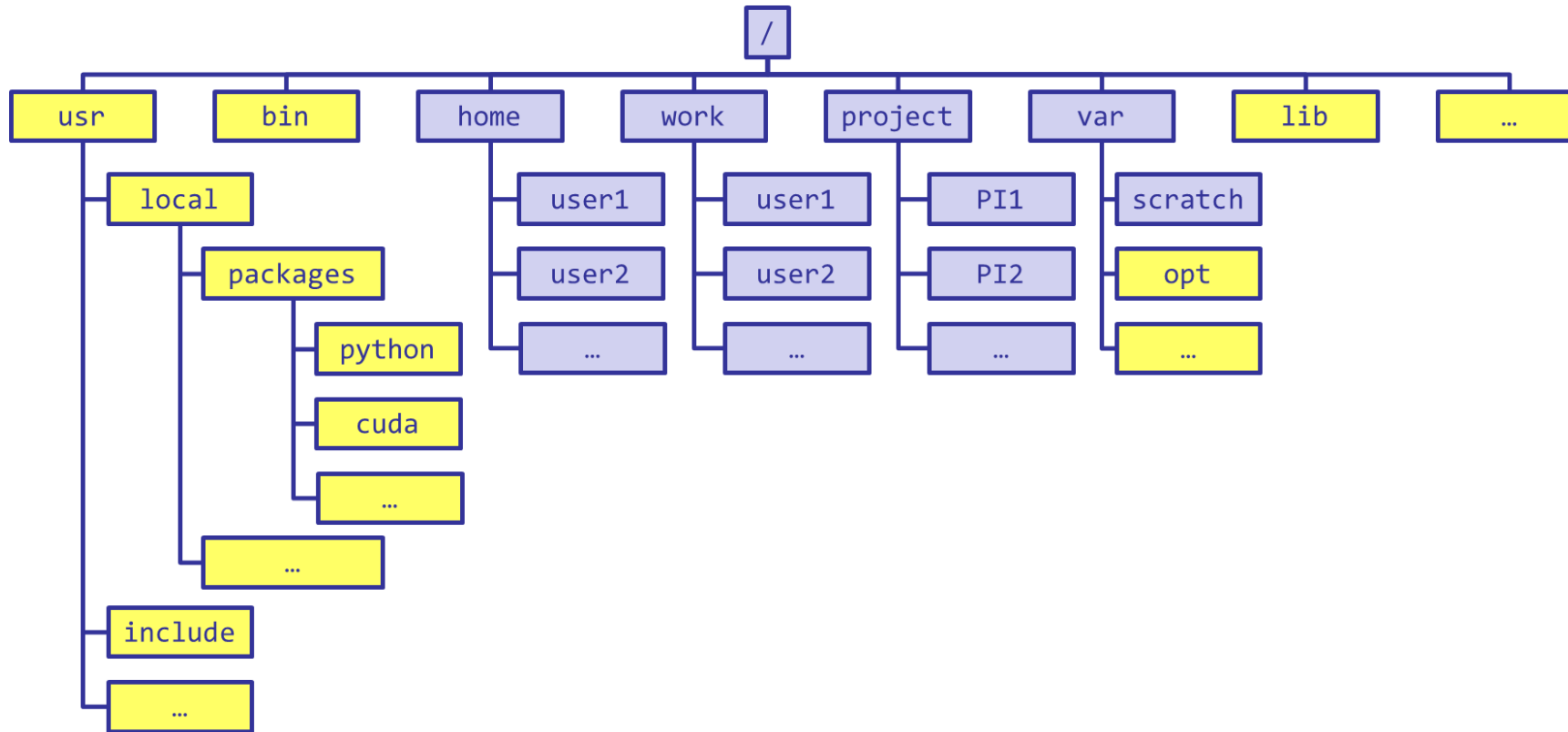
b) Permission denied (Welcome to HPC!)

- If you ask Google / ChatGPT...

```
$ sudo yum install lldpd-devel # On Red Hat  
$ sudo apt-get install lldpd-dev # On Ubuntu
```

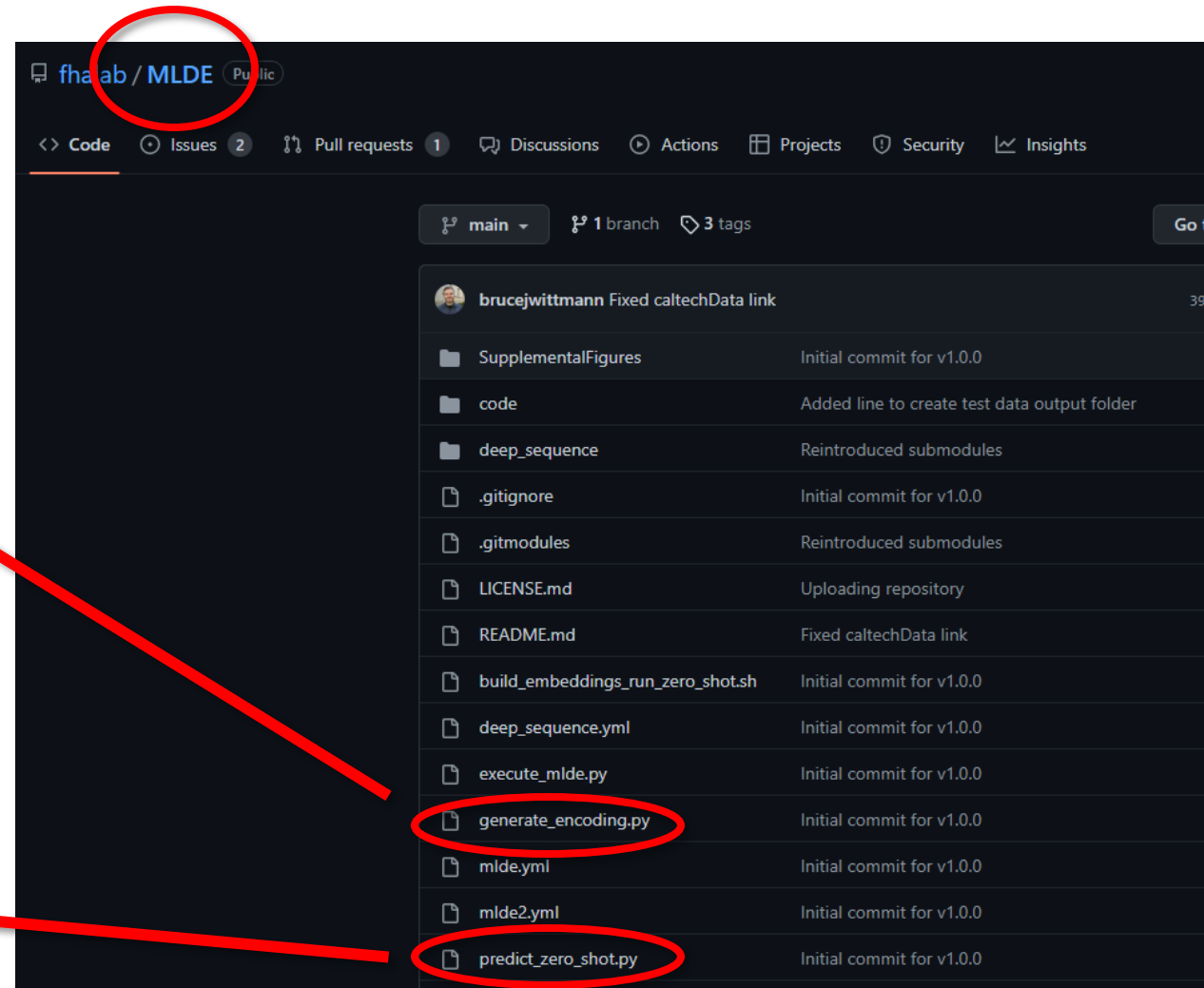
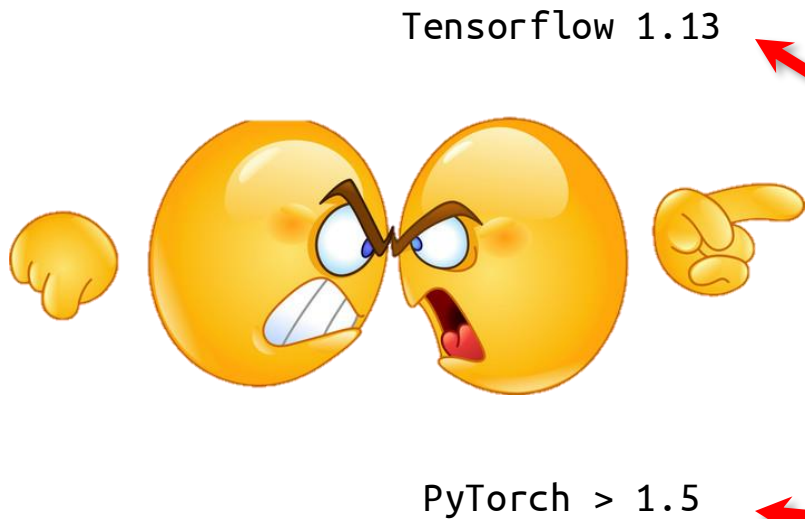


b) Permission denied (Welcome to HPC!)



c) Conflicted packages

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



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?

Any of those apply to you?

Magic Tools to Install / Manage Software



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

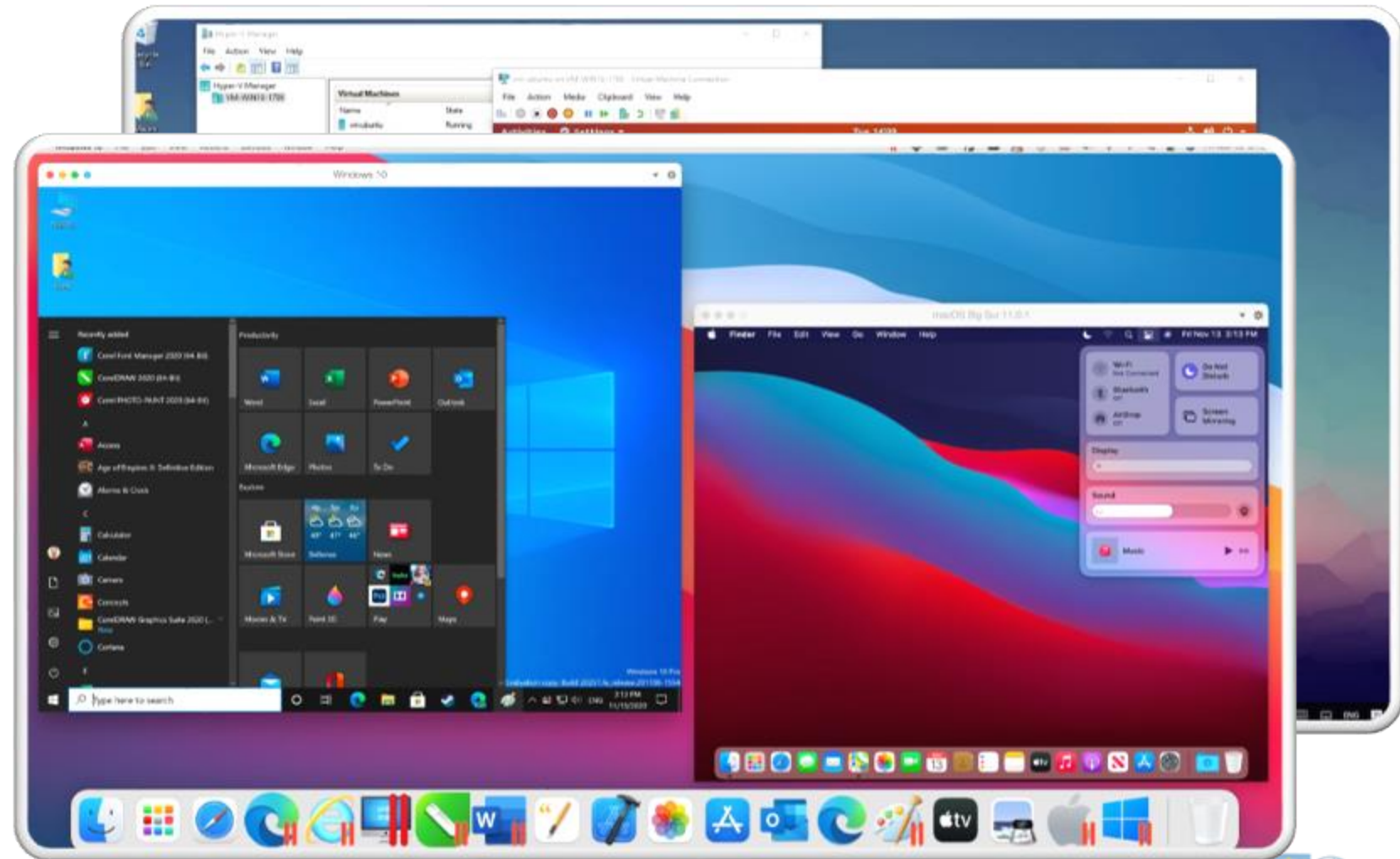
4. Build Your Own Container Image

- 1) What you need
- 2) Typical workflow
- 3) Make it easier - Recipe

2) Container & Singularity

a) What is a **container**?

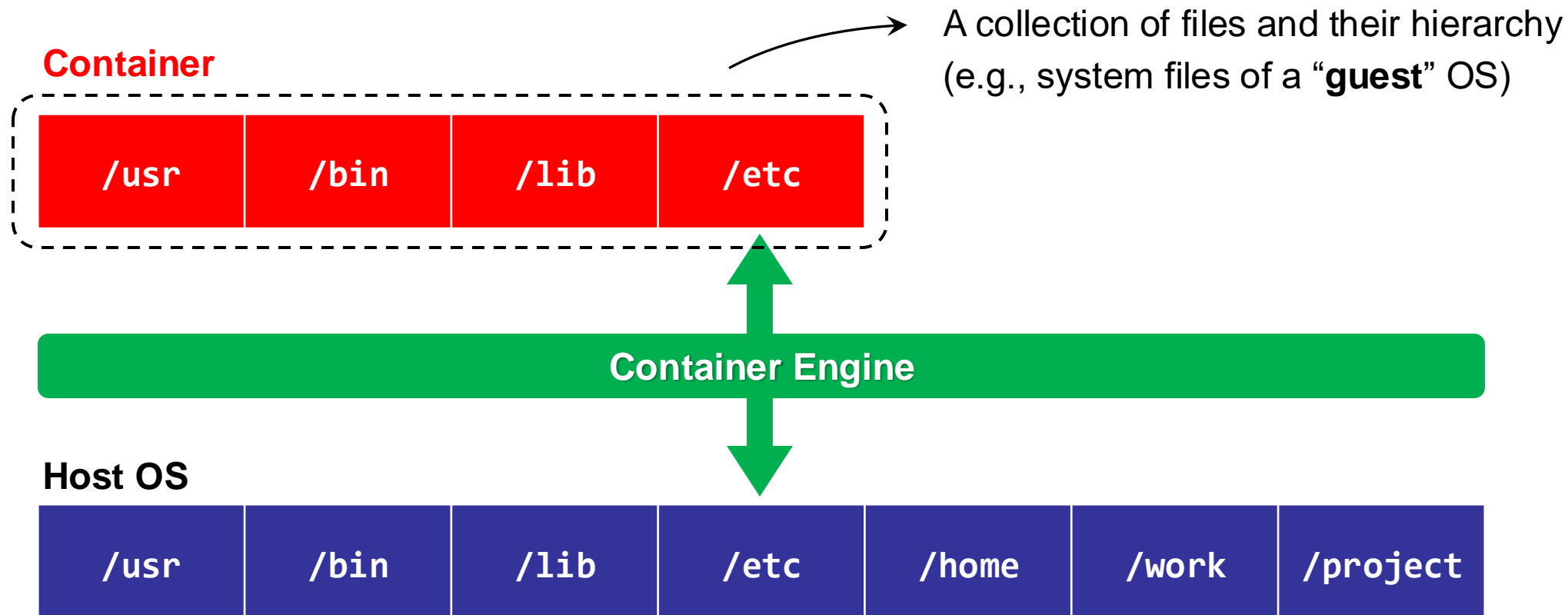
- **Virtual machine**
 - “Virtualize” / “mimic” an **entire computer** on another computer
 - Virtualize both **hardware** and **software**



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**

a) What is a **container**?



a) What is a **container**?

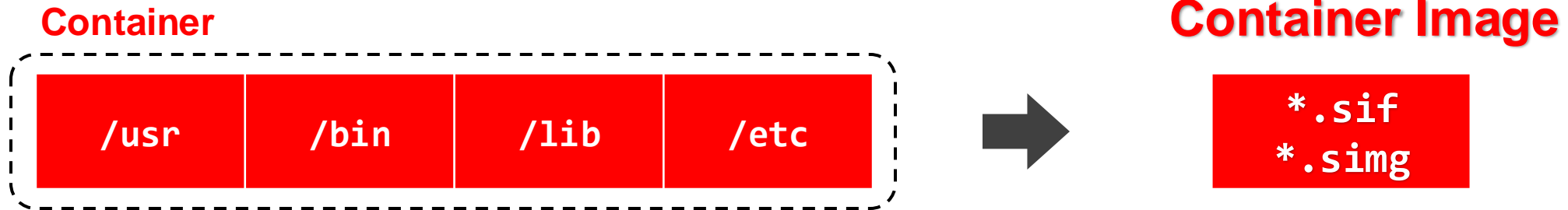


- A “chimera” system:

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



a) What is a **container**?



a) What is a **container**?

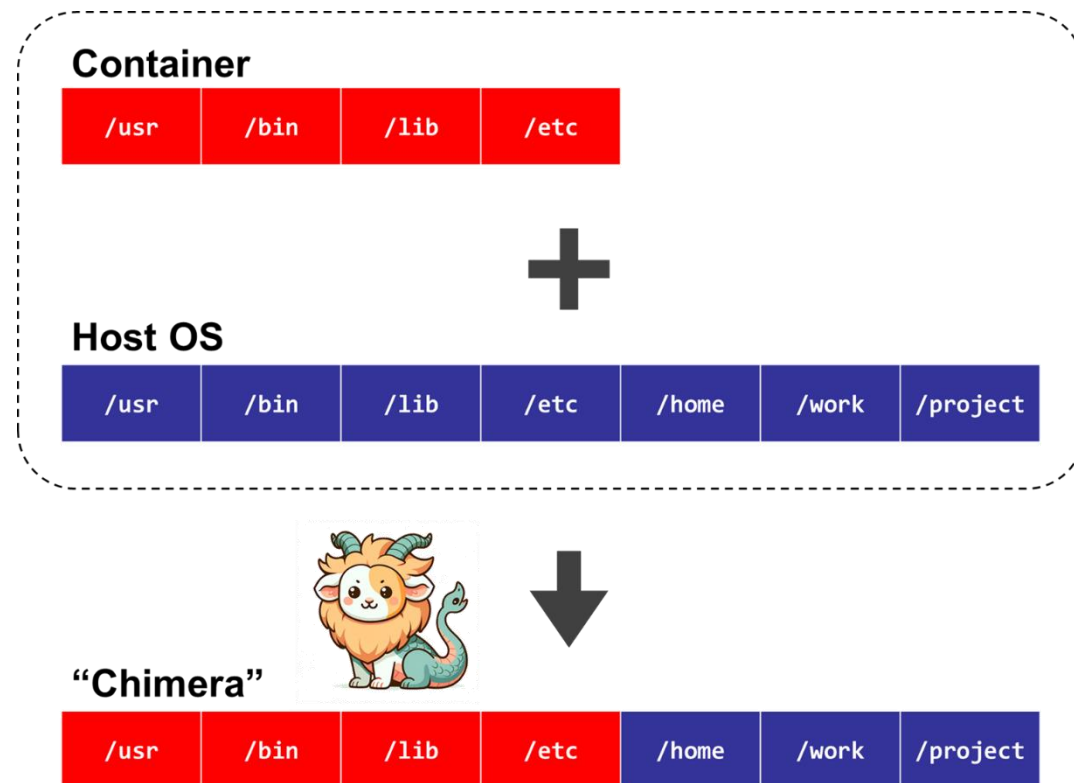
- **Properties:**

- **Self-contained**

All dependencies can be installed within the container

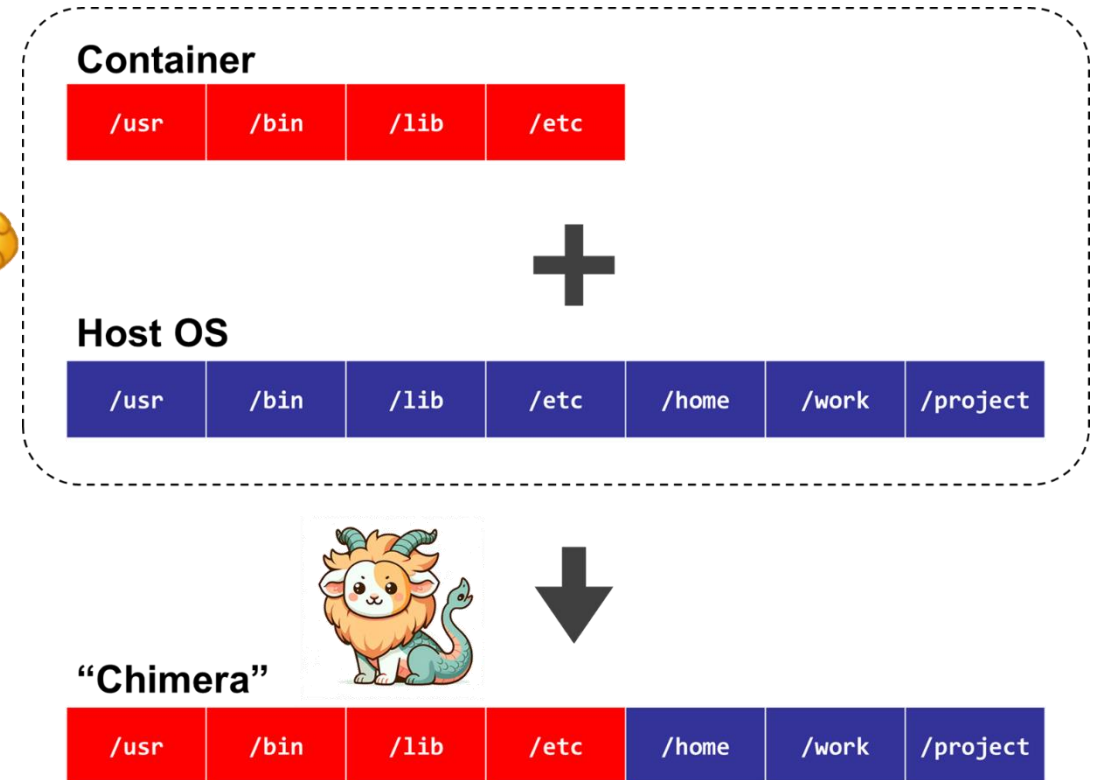
- **Isolated**

Whatever happens in a container stays in that container...



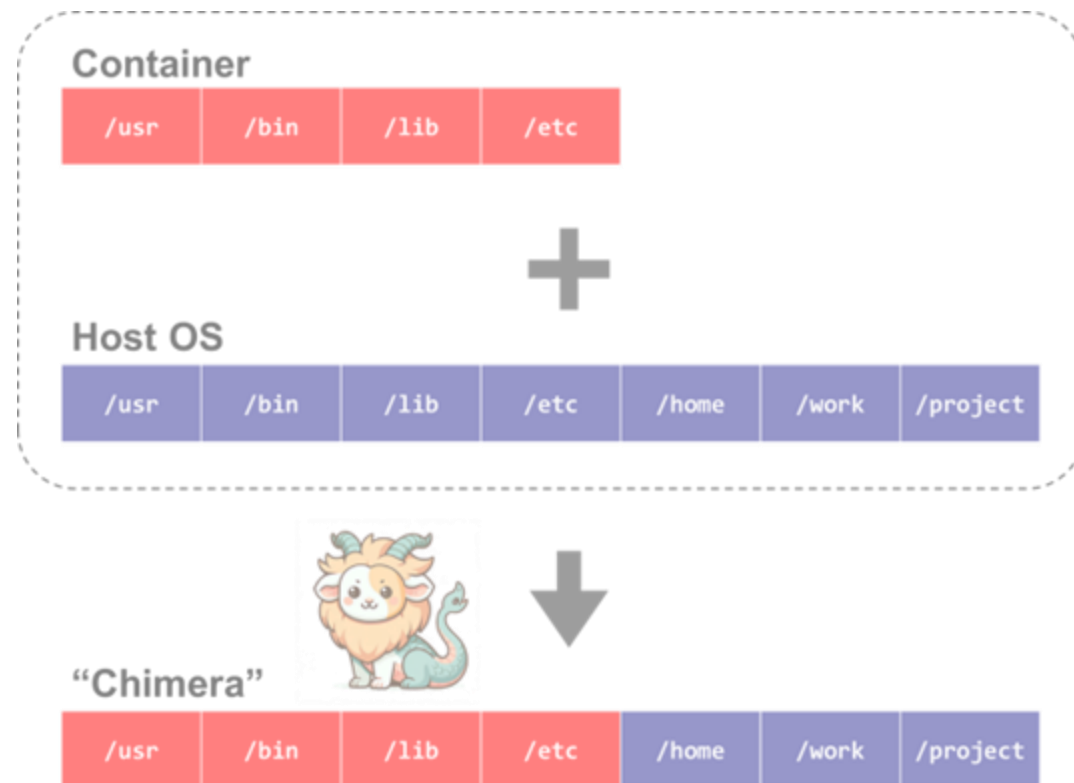
b) How does it solve my problems?

- **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!



c) What is Singularity?

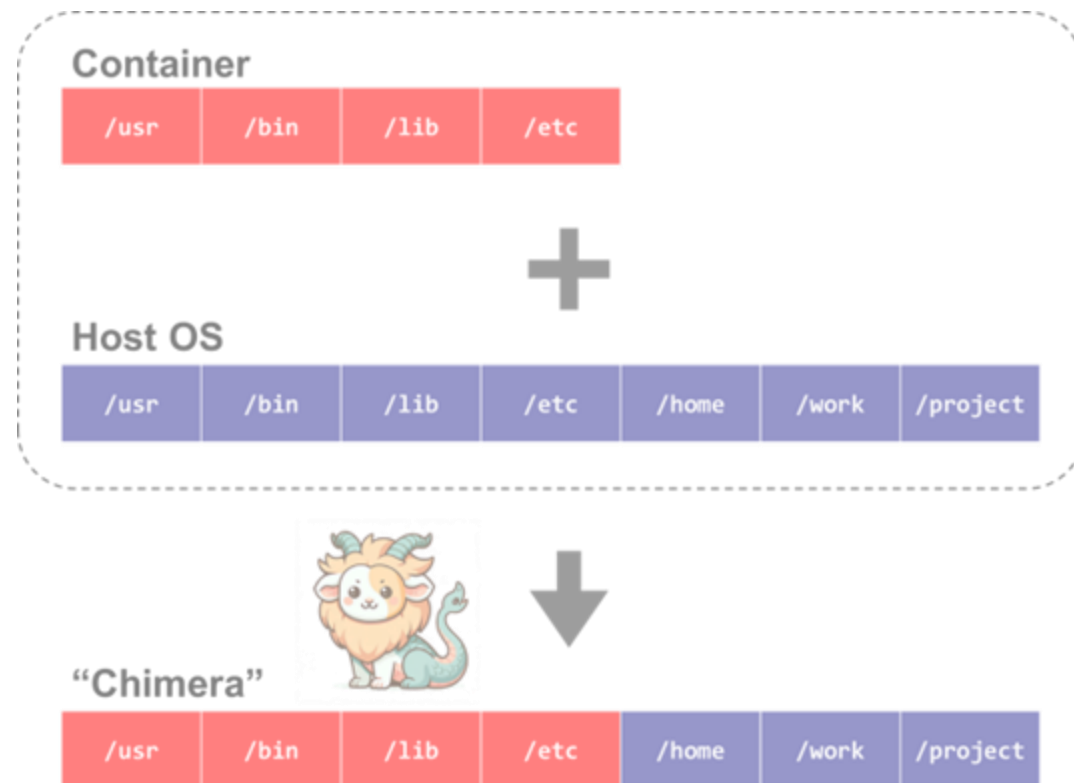
Technology →



c) What is Singularity?



↑ **Software** system that implements the technology



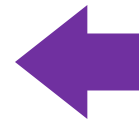
c) What is **Singularity**?



c) What is **Singularity**?



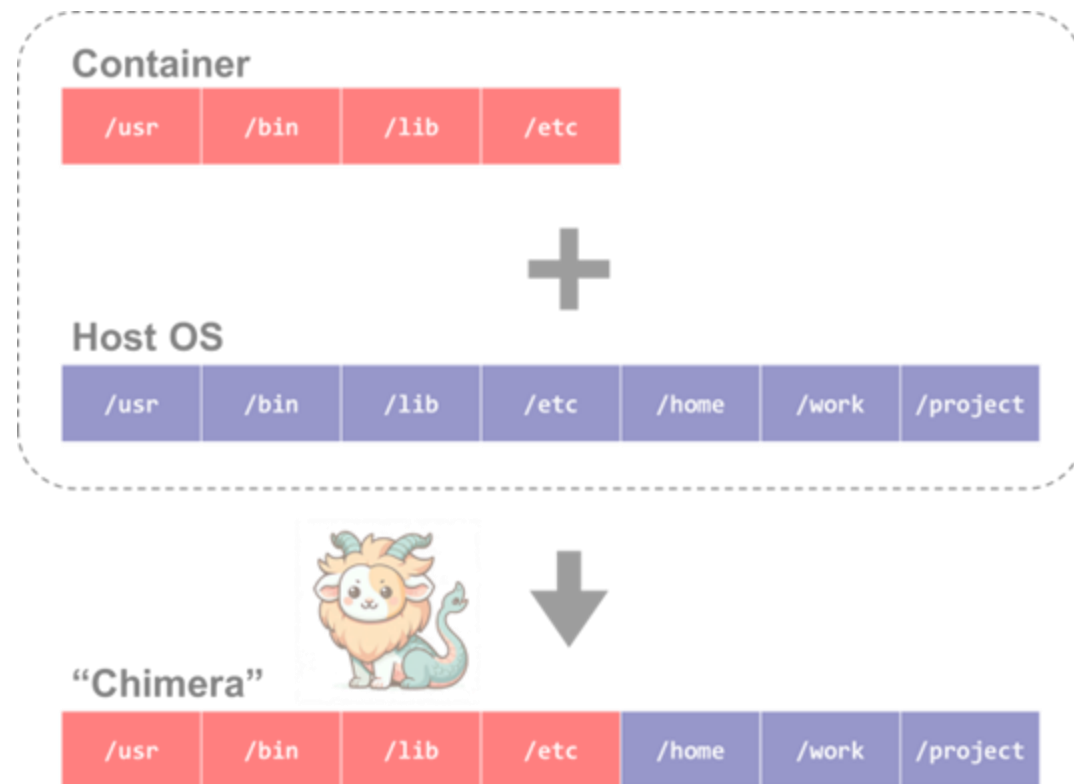
- Does **NOT** need root privileges
- “**Container for HPC**”



- **Needs** root privileges

Technology that helps with software installation →

↓ **Software** system that implements the technology



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. 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

- **Singularity availability**
 - a) On **all clusters**
 - ✓ **LSU HPC:** SMIC, Deep Bayou, SuperMike 3
 - ✓ **LONI:** QB3, QB4
 - b) Only on **computing nodes**
 - x Unavailable on head nodes
 - ✓ Must start a job (interactive & batch)
 - c) To **all users**
 - x No additional action required



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

- 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            fmrip-1.1.8-ubuntu-16.0.4.simg  
beast2-2.7.7.sif             fmrip-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  
cactus-4.0.0-dockerhub.simg  math-comp-2.4-ubuntu16.simg
```

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

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

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

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

Syntax		Description
<code>singularity shell [options] <container></code>		Starts an interactive shell in the image
[Options]	<code>-B /path/to/bind</code>	Bind a path(s) <ul style="list-style-type: none">• /home is bound by default
	<code>--nv</code>	Enable Nvidia GPU

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

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

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

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

Syntax		Description
singularity exec <i>[options]</i> <container> <command>		Execute a command in the image
<i>[Options]</i>	-B /path/to/bind	Bind a path(s) <ul style="list-style-type: none">• /home is bound by default
	--nv	Enable Nvidia GPU

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

Syntax		Description
singularity run <i>[options]</i> <container>		Run a prewritten script
<i>[Options]</i>	<i>-B /path/to/bind</i>	Bind a path(s) <ul style="list-style-type: none">• /home is bound by default
	<i>--nv</i>	Enable Nvidia GPU

- Quick recap

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

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

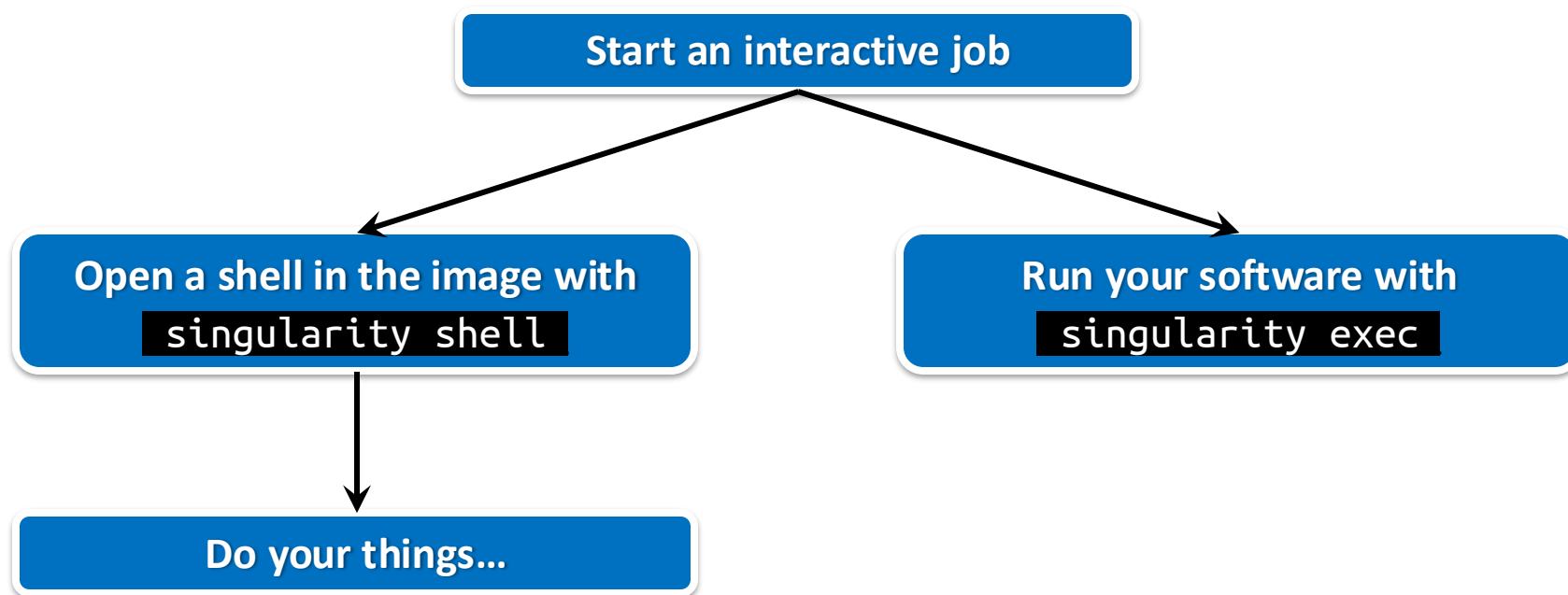
- 1) What you need
- 2) Typical workflow
- 3) Make it easier - Recipe

3) Run jobs with Singularity

- Job types and commands

Job Type	Commands	Purpose
Interactive	<ul style="list-style-type: none">• singularity shell <i>[options]</i> <container>• singularity exec <i>[options]</i> <container> <command>	<ul style="list-style-type: none">• Debugging & testing
Batch	<ul style="list-style-type: none">• singularity exec <i>[options]</i> <container> <command>	<ul style="list-style-type: none">• Production

a) Interactive job



b) Batch job

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

Example

```
cd /to/work/directory
```

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

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

- Run:
 - All users

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

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  
alphafold-catgumag-2.2.sif  
alps-2.3.0-dockerhub.simg  
alps-2.3.0-dockerhub-v2.simg  
bcftools-1.18.sif  
beast2-2.7.7.sif  
blast-2.14.1.sif  
blender-2.79b-cuda-8.0-ubuntu-16.04.simg  
bowtie2-2.5.1.sif  
braker-3.0.8.sif  
busco-5.7.1.sif  
bwa-0.7.17.sif  
brat-4.0.0-dockerhub.simg  
fed28.simg  
fenics-adjoint.2018.ubuntu16.simg  
firedrake.dockerhub.simg  
firedrake.vanilla.simg  
fmrip-1.1.8-ubuntu-16.0.4.simg  
fmrip-1.3.2-ubuntu-16.0.4.simg  
gatk-4.5.0.0.sif  
gcc-9.2.0-dockerhub.simg  
hisat2-2.2.1.sif  
jax-0.4.26.sif  
jax.sif  
maker-3.01.03.sif  
mash-2.4.0-ubuntu16.simg
```

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

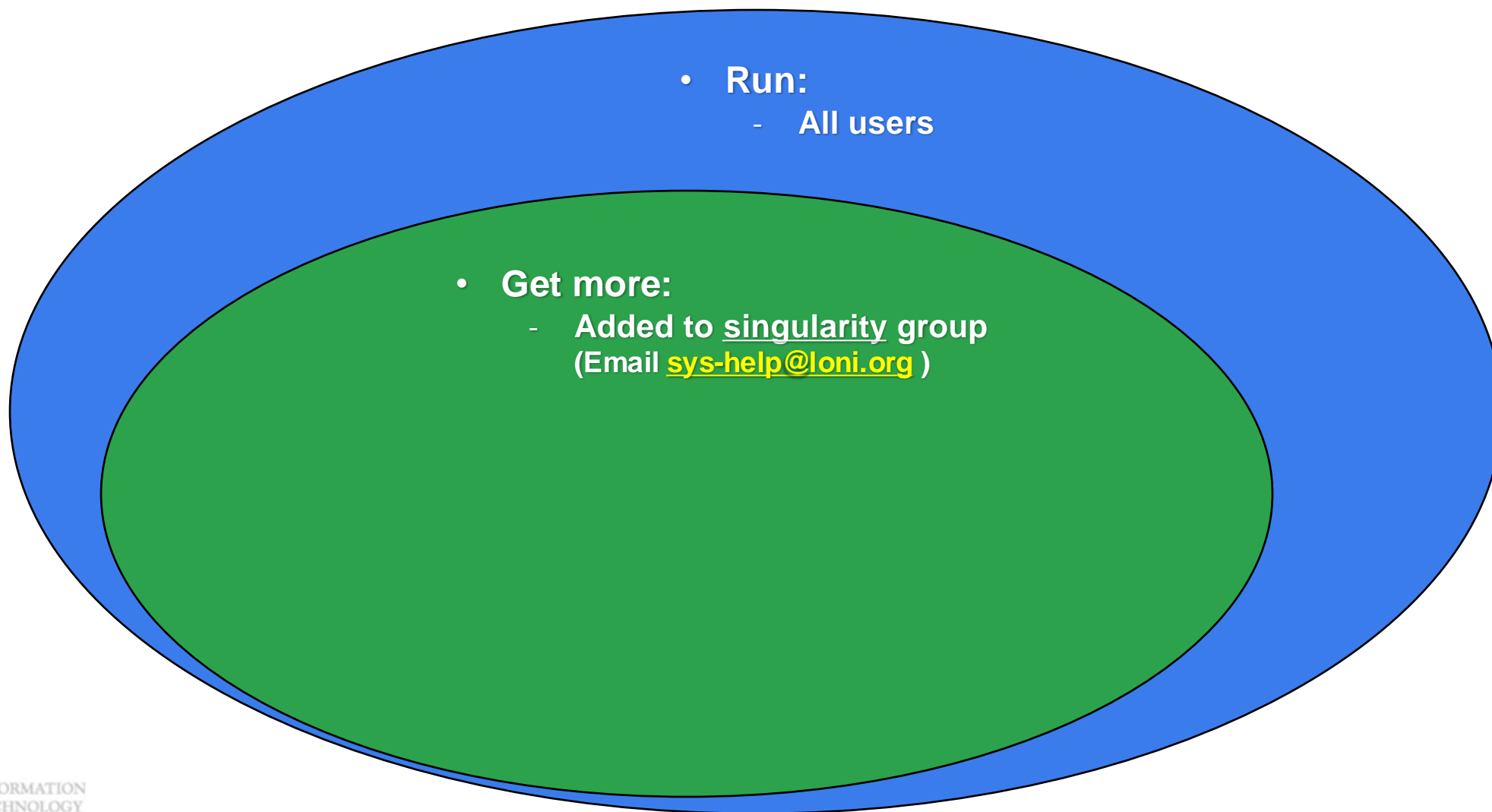
1) What you need

```
(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 May 13 11:19 agat-1.4.0.sif
-rwxr-xr-x 1 jasonli3 singularity 4336439296 Jun 24 15:29 alphafold-catgumag-2.2.sif
-rwxr-xr-x 1 jasonli3 singularity 477290496 May 13 11:19 agat-1.4.0.sif
-rwxr-xr-x 1 jasonli3 singularity 1188212767 Jun 24 15:30 cactus-1.0.0-ubuntu-16.04-mesos.simg
-rwxr-xr-x 1 jasonli3 singularity 118206464 May 13 11:19 agat-1.4.0.sif
-rwxr-xr-x 1 jasonli3 singularity 2431631360 Jun 24 15:30 cactus-1.0.0-ubuntu-16.04.simg
-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. 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 !**



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

- **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	<ul style="list-style-type: none">• https://biocontainers-edu.readthedocs.io/en/latest/• For biology
Nvidia NGC	<ul style="list-style-type: none">• https://catalog.ngc.nvidia.com/containers• For Nvidia GPU
Bitnami	<ul style="list-style-type: none">• https://bitnami.com/stacks/containers• By VmWare
Docker Hub Quay.io	<ul style="list-style-type: none">• https://hub.docker.com/ & https://quay.io/• BUT! Do NOT 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

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

- **Steps:**
 - a) Step 1: Pull the image
 - b) Step 2: Change group ownership

a) Step 1: Pull the image

Syntax		Description
singularity pull <code><source></code>		Pull an image from source
<code><source></code>	<code>docker://container[:tag]</code> <ul style="list-style-type: none">(Compare to Docker command) <code>docker pull container[:tag]</code>	Pull a Docker container and convert to Singularity <ul style="list-style-type: none">Many software's official container release is in Docker form (may or may not on Docker Hub)
	<code>http://www.myexample.com/container_image.sif</code>	Download an image file from a web source

a) Step 1: Pull the image

Syntax		Description
singularity build	<code><target></code> <code><source></code>	Build an image from source (Advanced)
	<code>docker://container[:tag]</code>	Build from a Docker container
<code><source></code>	<code>container_image.sif</code>	Build from a local image file
	<code>container_sandbox/</code>	Build from a local sandbox (A directory form of a container)
	<code>container_recipe.def</code>	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
<code>singularity pull [options] [target] <source></code>	Simple pull
<code>singularity build [options] <target> <source></code>	Advanced build command

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** (w/ GPU support)

```
$ singularity pull docker://pytorch/pytorch:2.6.0-cuda11.8-cudnn9-runtime
```

- ii. **Tensorflow** (w/ GPU support)

```
$ singularity pull docker://tensorflow/tensorflow:2.17.0-gpu-jupyter
```

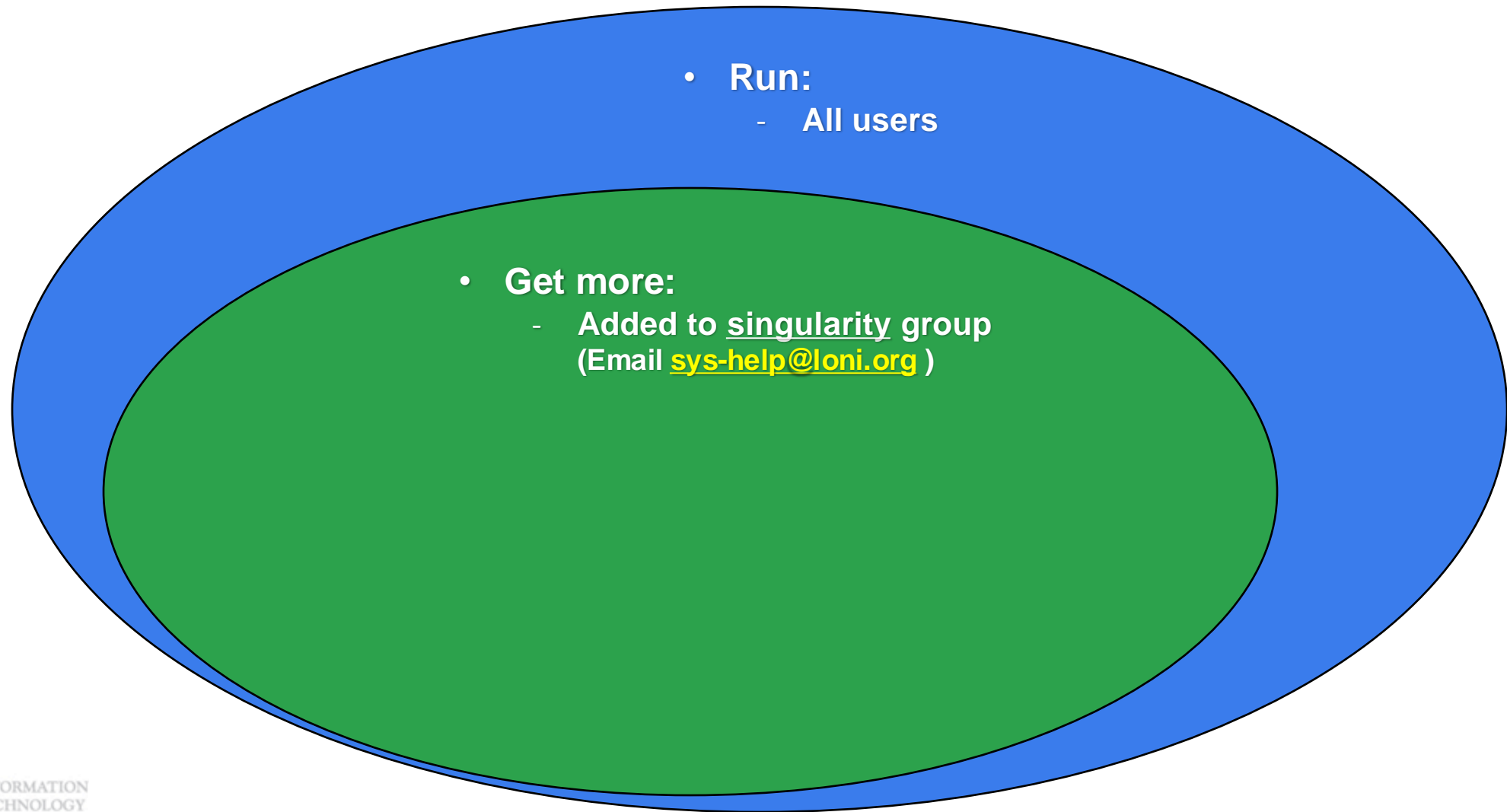
3) How to get

- **BONUS: Hot packages!**
 - i. **PyTorch** (w/ GPU support)

```
$ module load pytorch
```

- ii. **Tensorflow** (w/ GPU support)

```
$ module load tensorflow
```

- **Steps:**

a) Step 1: Pull the image

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

b) Step 2: Change group ownership

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

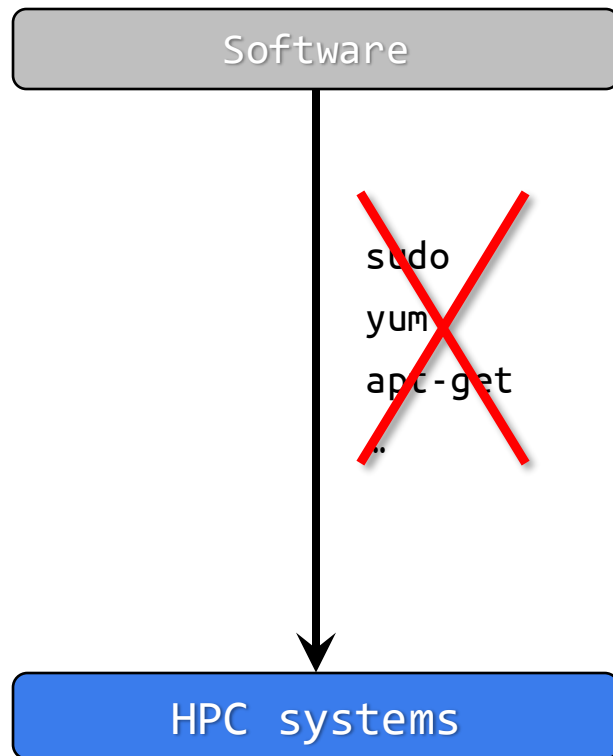
4. Build Your Own Container Image

- 1) What you need
- 2) Typical workflow
- 3) Make it easier - Recipe

- **Scenarios:**

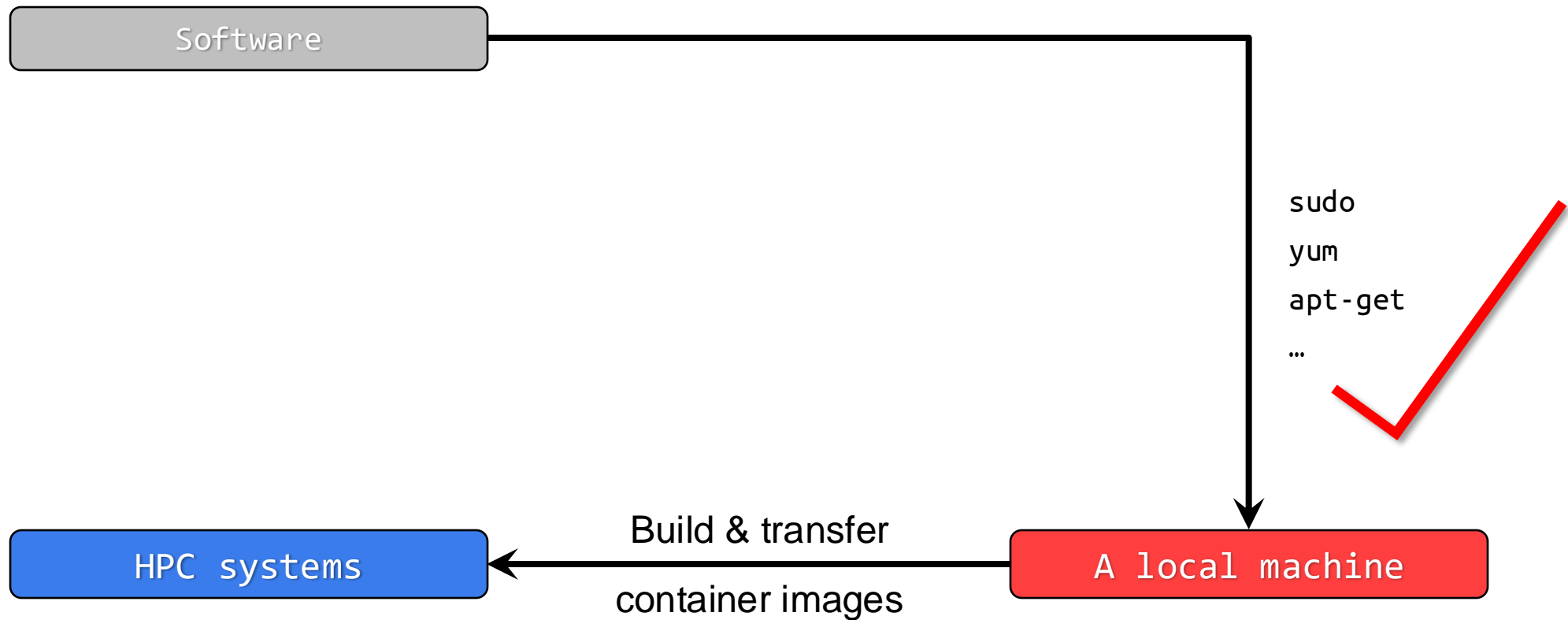
- I did not find any container release. Need to DIY.
- Installation can be easily done using `sudo apt` or `sudo yum` if I have the permission.
- I found a container, but need to make changes (e.g., adding something else).

- Idea



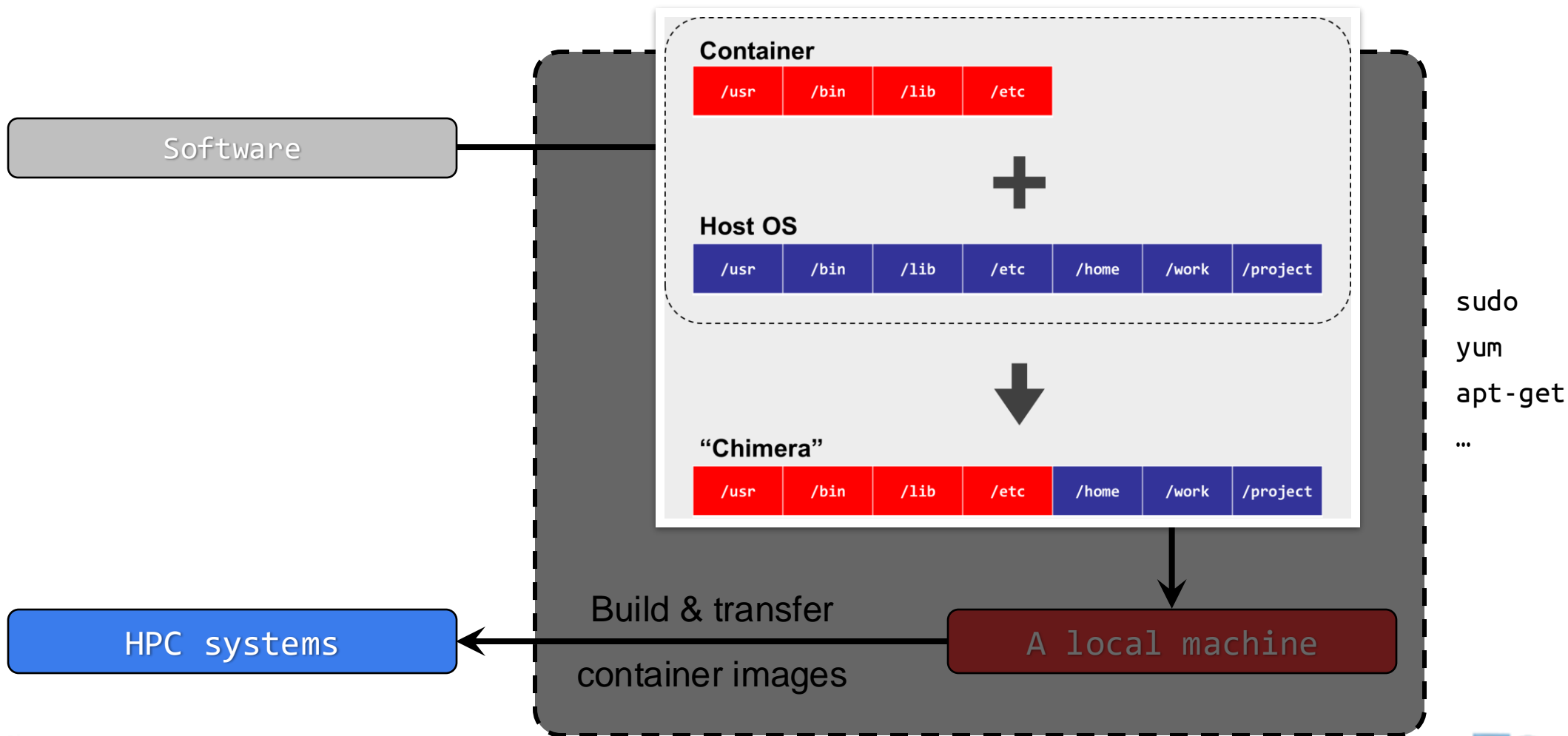
4. Build Your Own Container Image

- Idea



4. Build Your Own Container Image

- Idea



1. Why Container?

- 1) Problems
- 2) Container & Singularity

2. Run an Existing Container Image

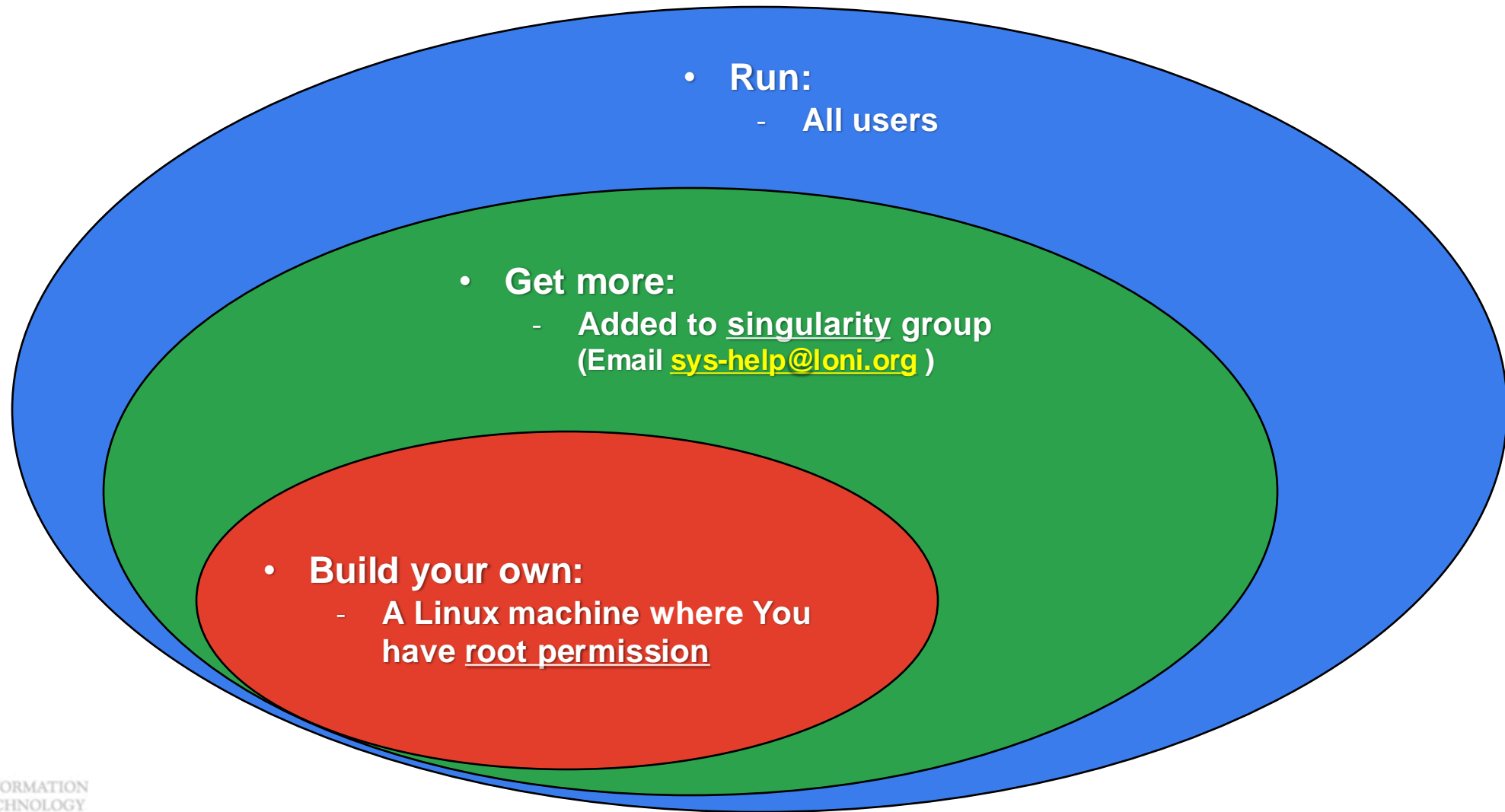
- 1) What you need
- 2) Basic commands
- 3) Running jobs with Singularity

3. Get More Container Images

- 1) What you need
- 2) Where to get
- 3) How to get

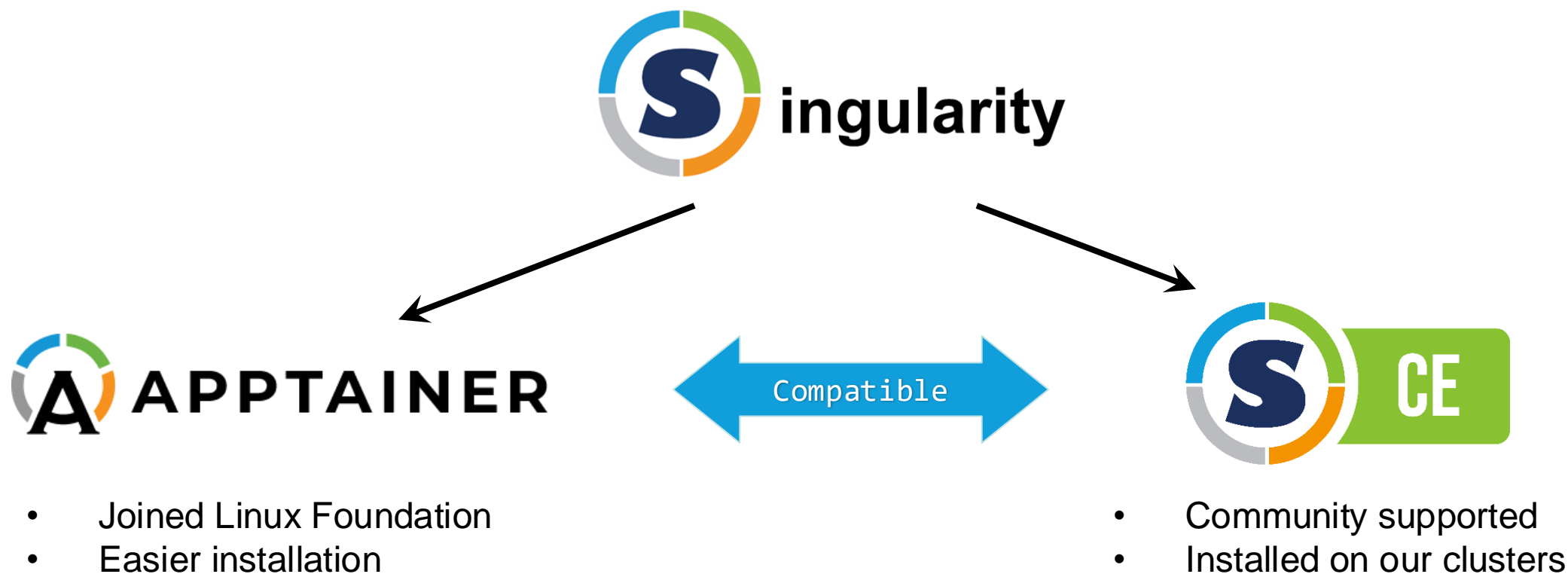
4. Build Your Own Container Image

- 1) What you need
- 2) Typical workflow
- 3) Make it easier - Recipe



1) What you need

- Install Singularity



[1] <https://apptainer.org/docs/admin/main/installation.html>

[2] <https://docs.sylabs.io/guides/3.8/admin-guide/installation.html>

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

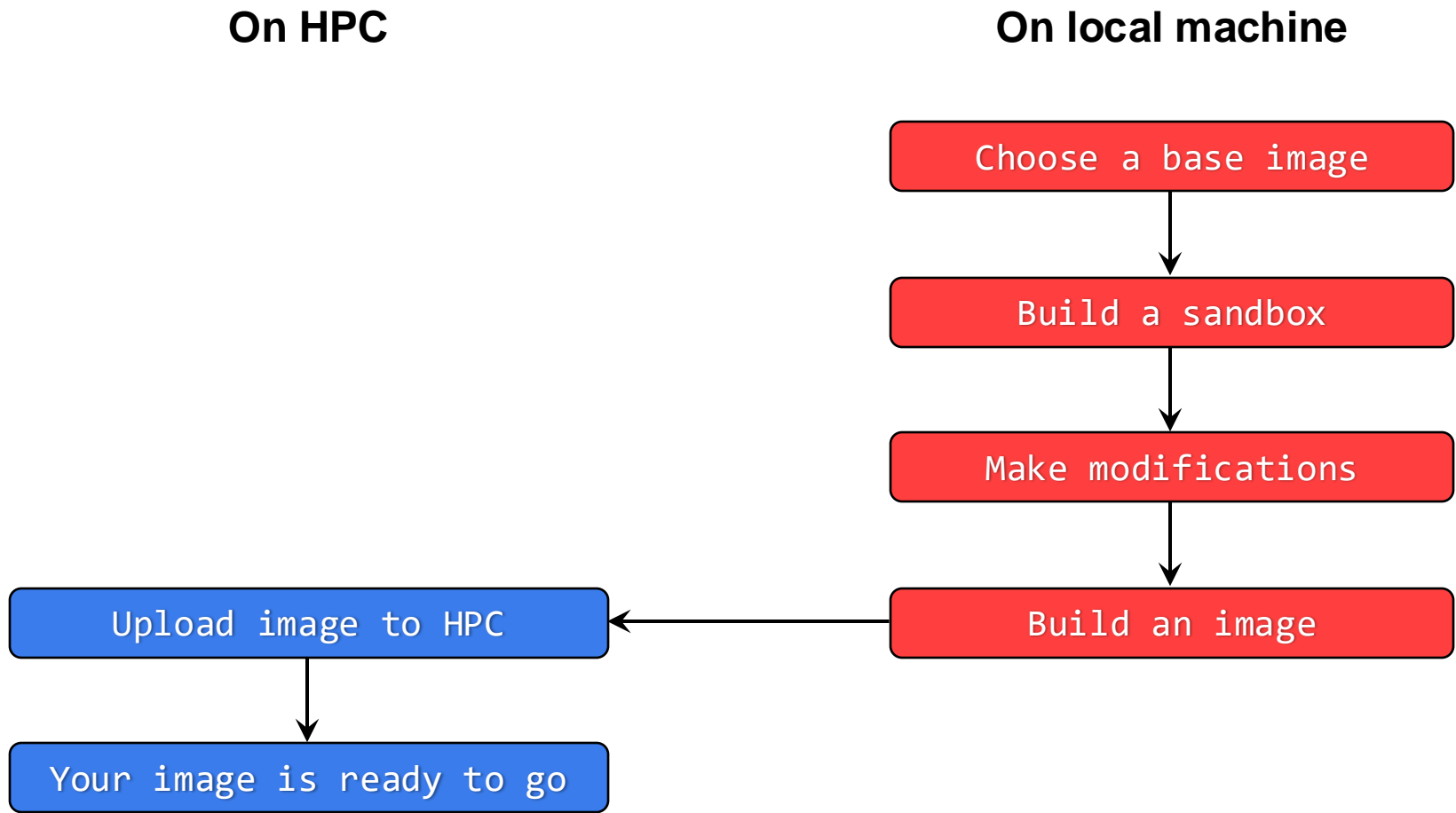
3. Get More Container Images

- 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



a) Choose a base image

Common choices	Typical scenarios
<p>A minimum, “mint” OS (e.g., Ubuntu, Rocky, Debian, ...)</p>	<ul style="list-style-type: none">• 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
<p>A container with software already installed (e.g., TensorFlow, PyTorch, ...)</p>	<ul style="list-style-type: none">• You need to modify an existing working image (e.g., add a Python module to Tensorflow image)

b) Build a sandbox

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

b) Build a sandbox

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



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

b) Build a sandbox

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



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

c) Make modifications

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

c) Make modifications

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

- i. Allows **writing** to the sandbox
 - Read-only without it

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


- Read-only without it

c) Make modifications

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

d) Build an image from sandbox

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



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

d) Build an image from sandbox

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



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

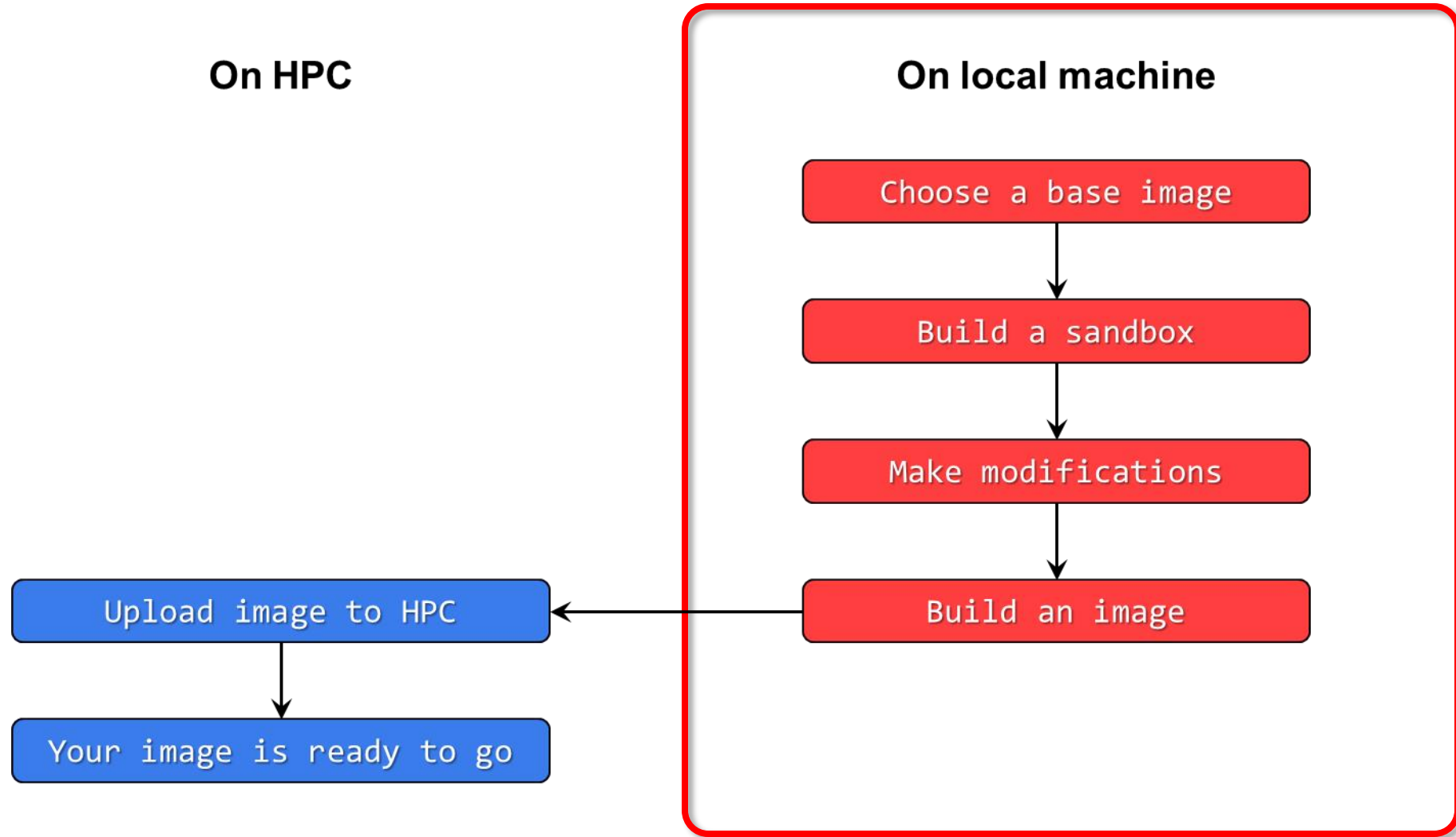
- Quick recap

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 ...

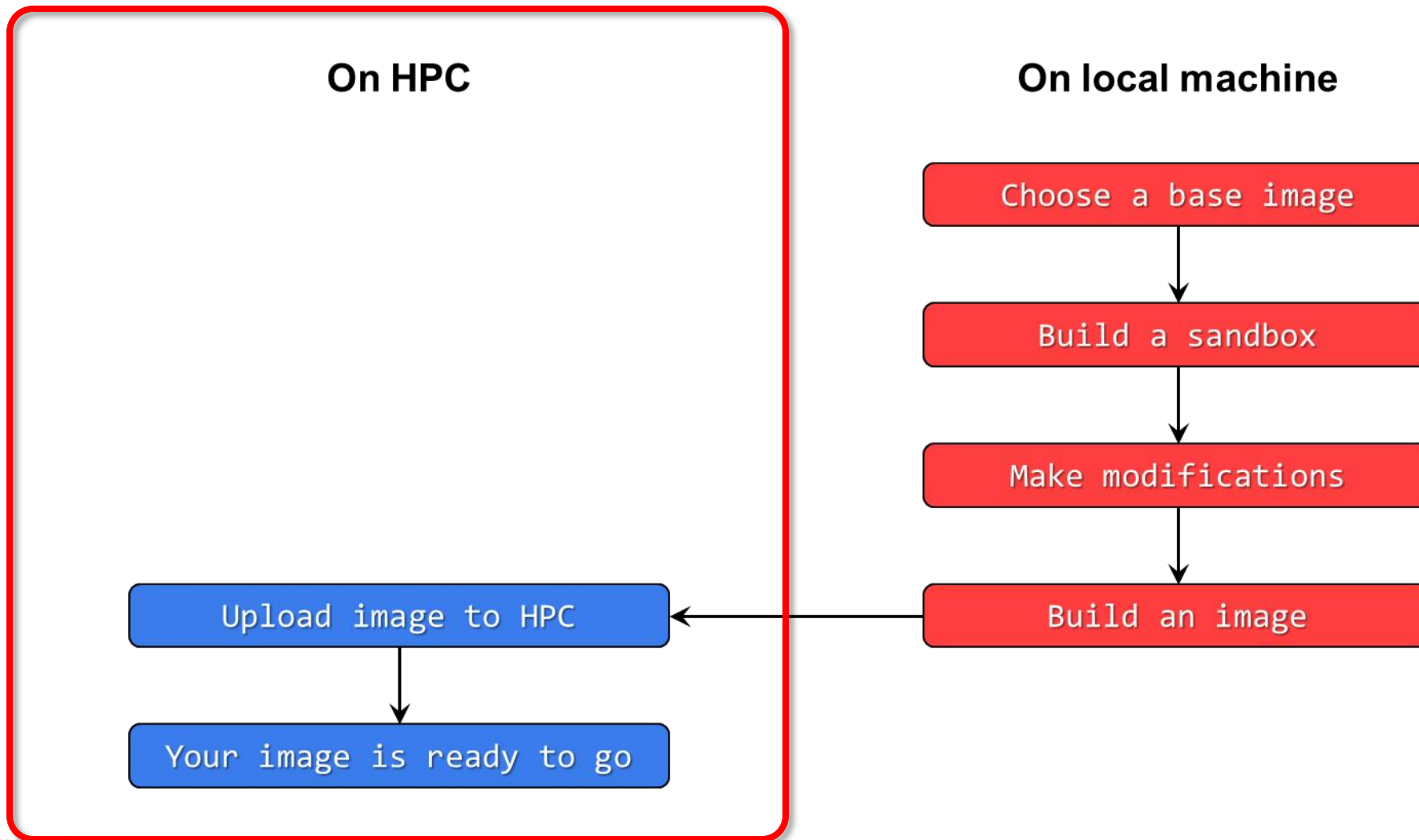
e) Upload image to HPC and run

Now! The moment of truth!

2) Typical workflow



2) Typical workflow



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

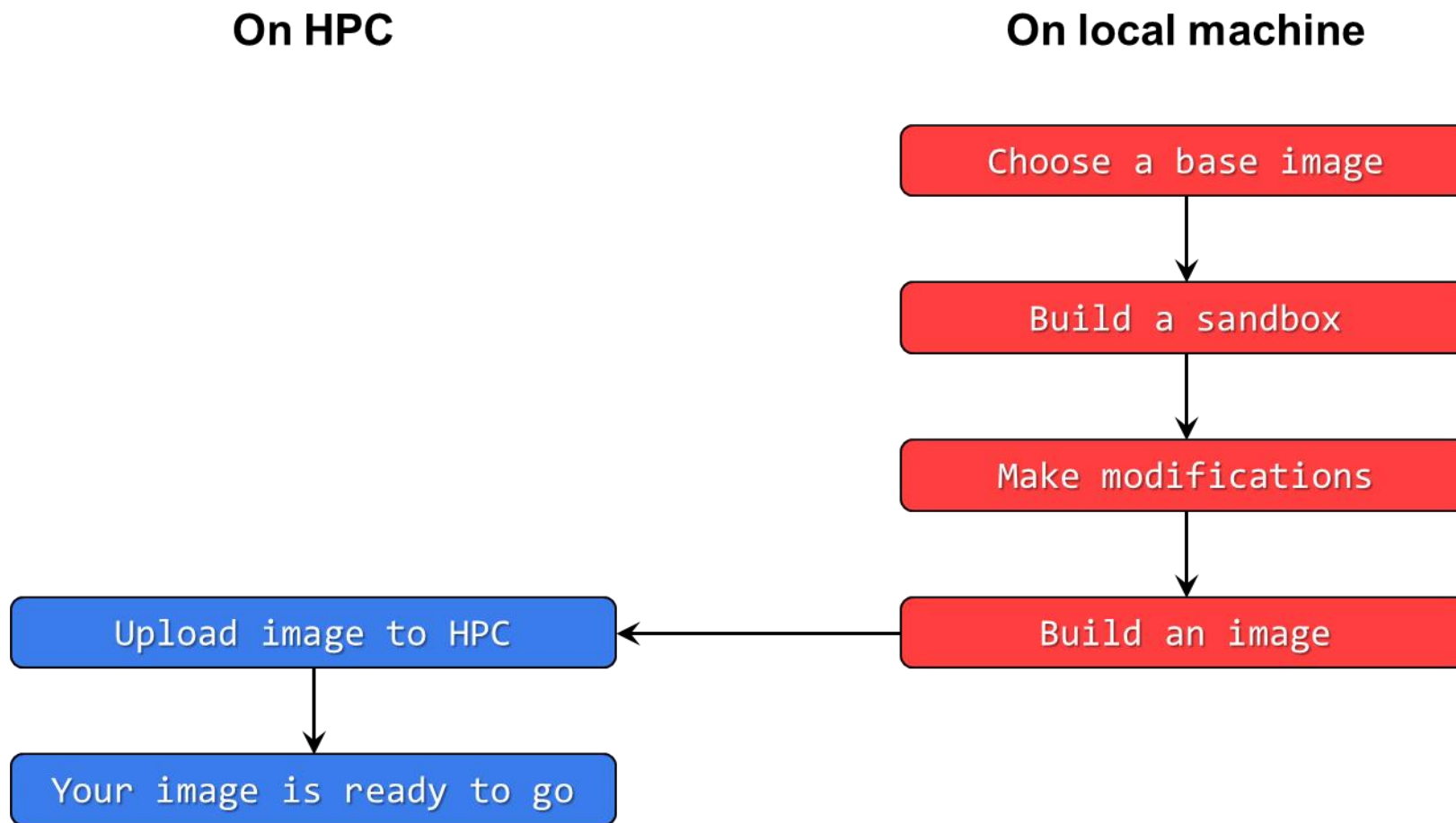
- 1) What you need
- 2) Where to get
- 3) How to get

4. Build Your Own Container Image

- 1) What you need
- 2) Typical workflow
- 3) Make it easier - Recipe

3) Make it easier - Recipe

- Why?



- Why?

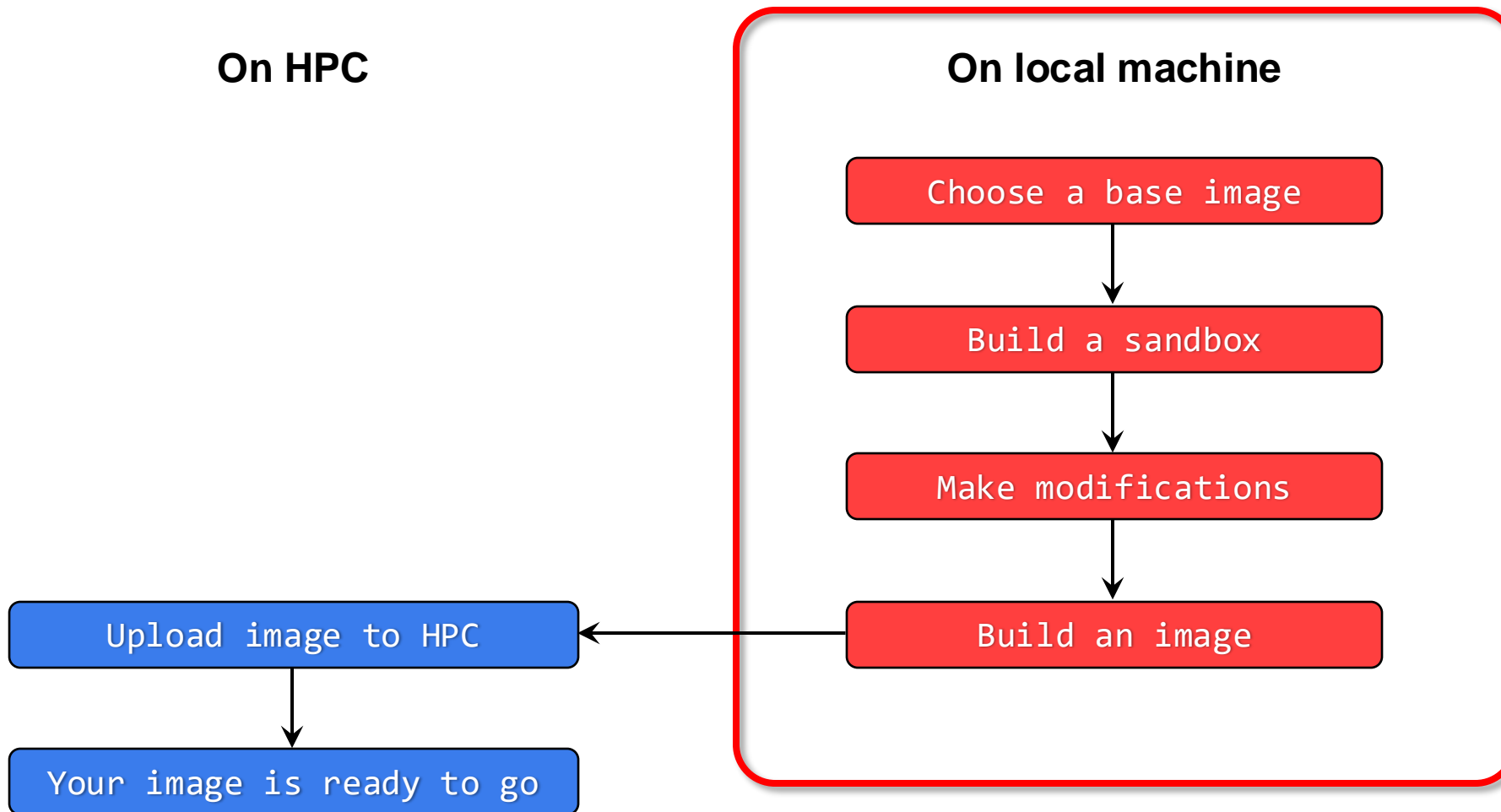
Pros	Cons
<ul style="list-style-type: none">• Flexibility	<ul style="list-style-type: none">• Repeatability• Minimizing image size

- Solution:

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

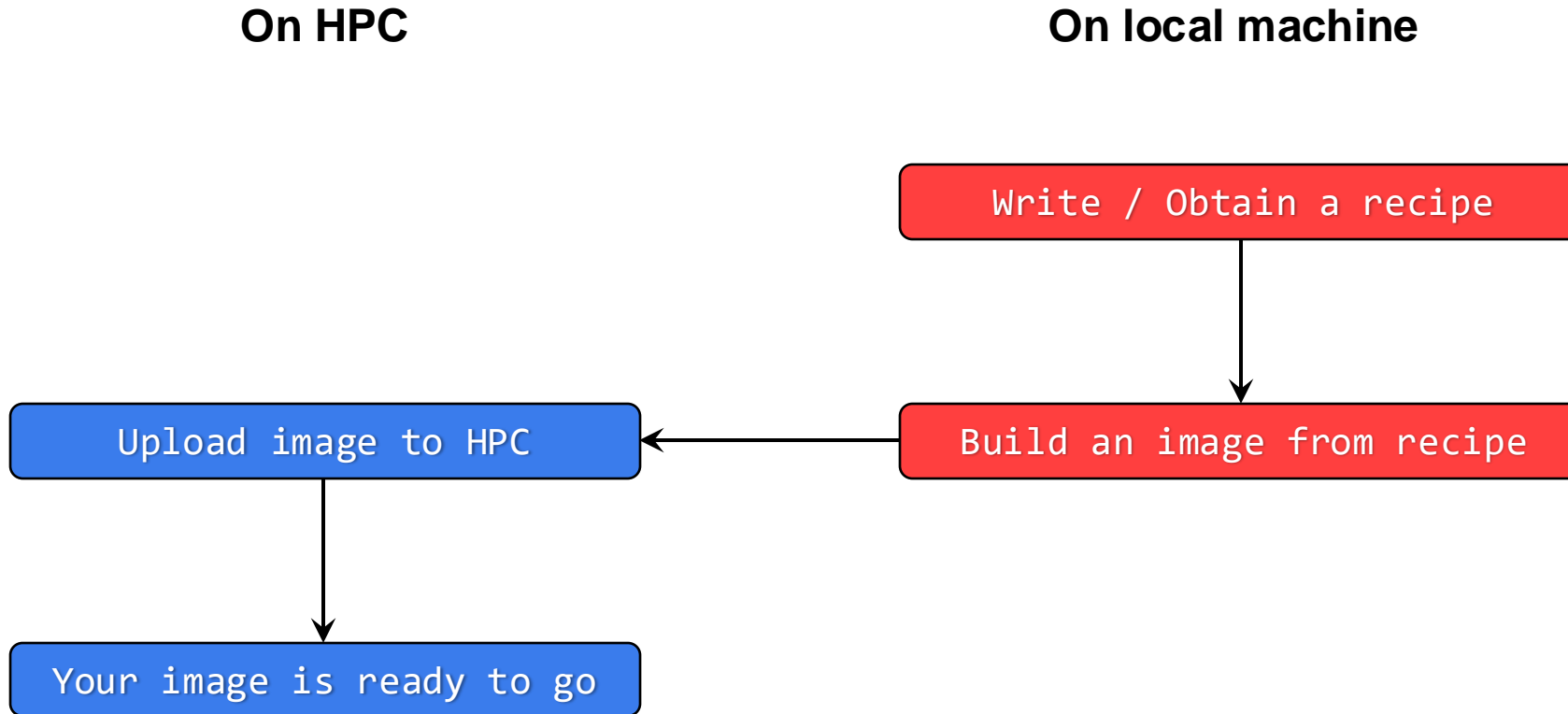
3) Make it easier - Recipe

- Why?



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!'"
```


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`

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!'"
```

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!'"
```

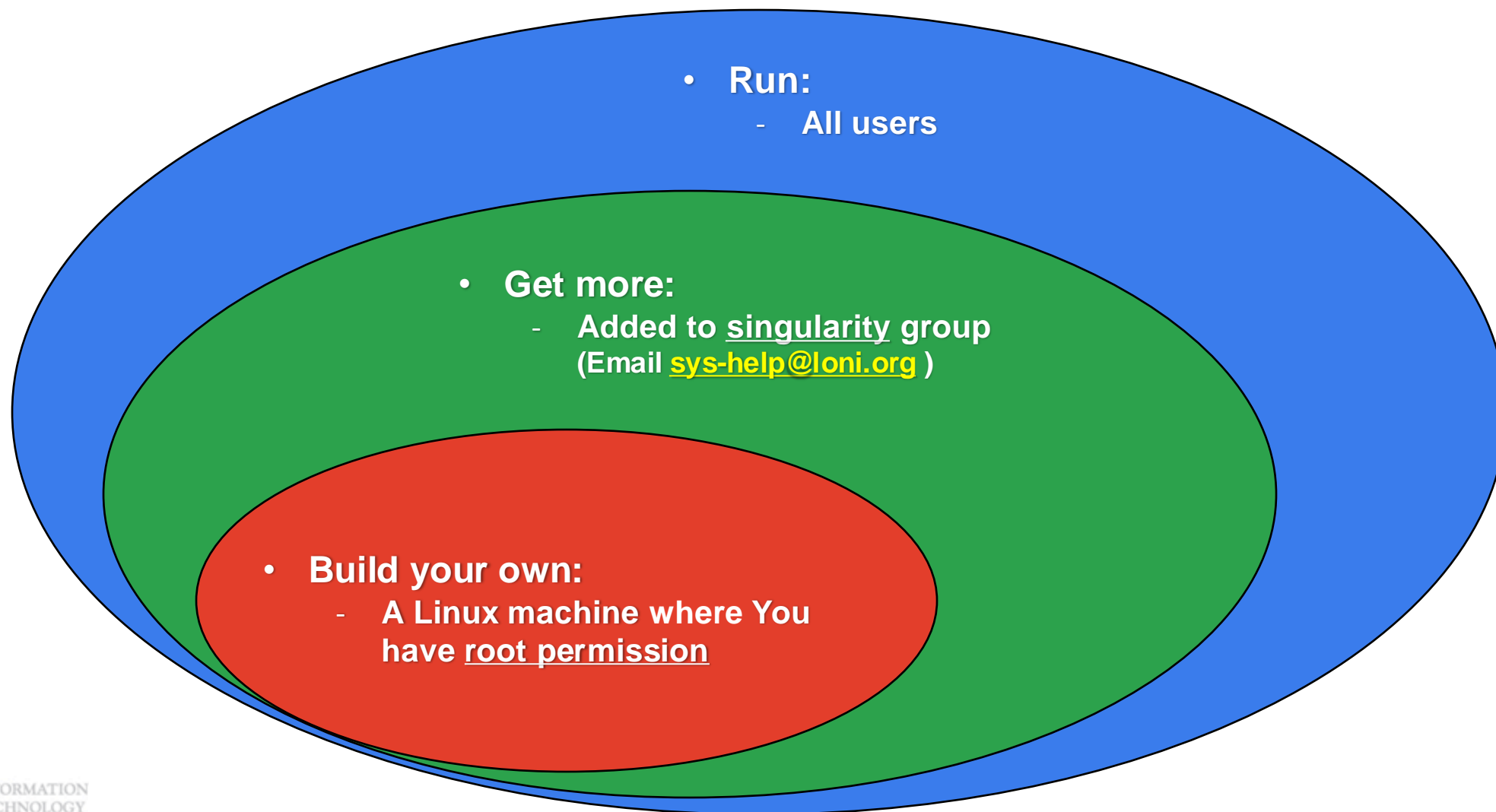
3) Make it easier - Recipe

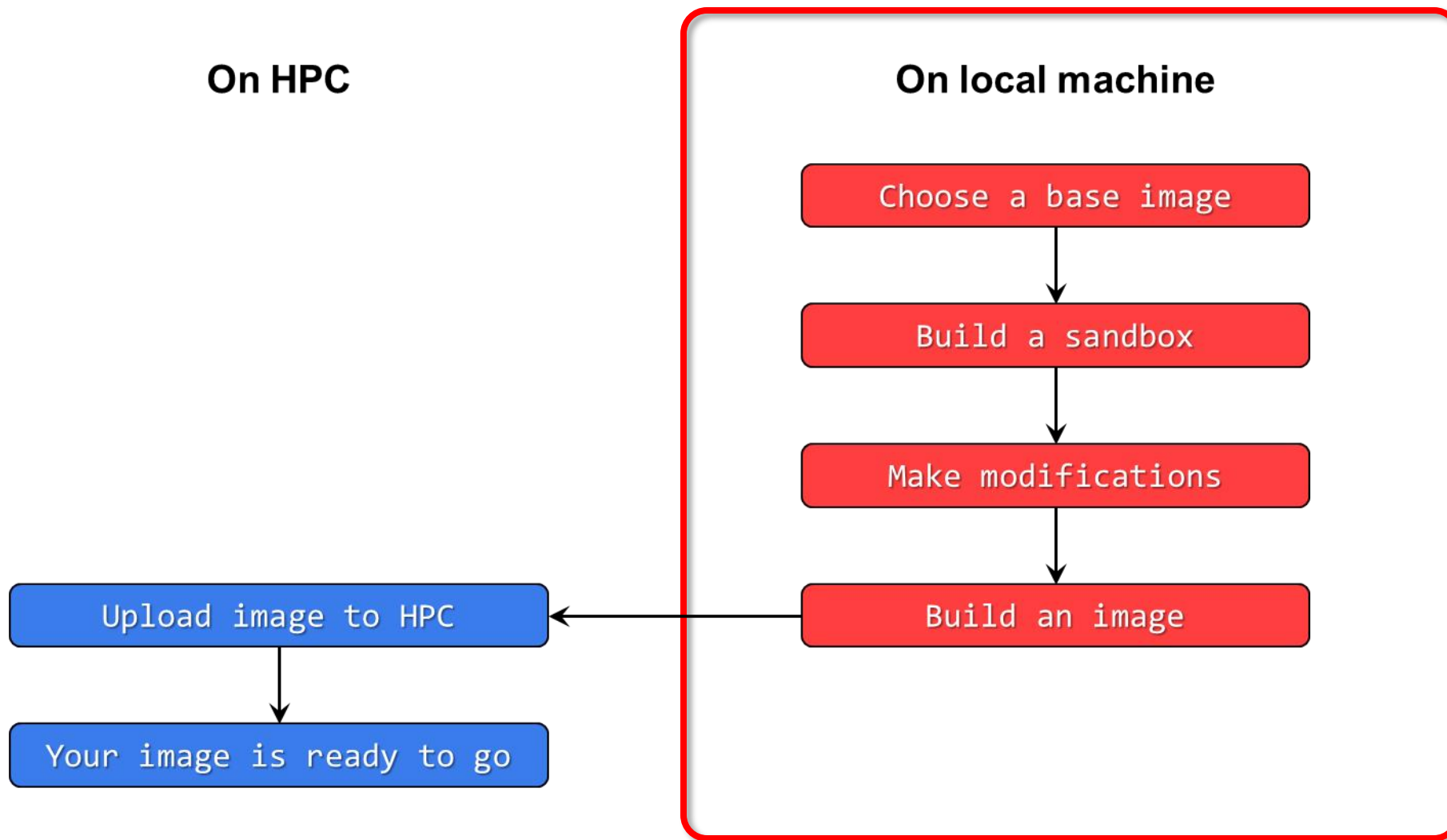
b) Build the recipe

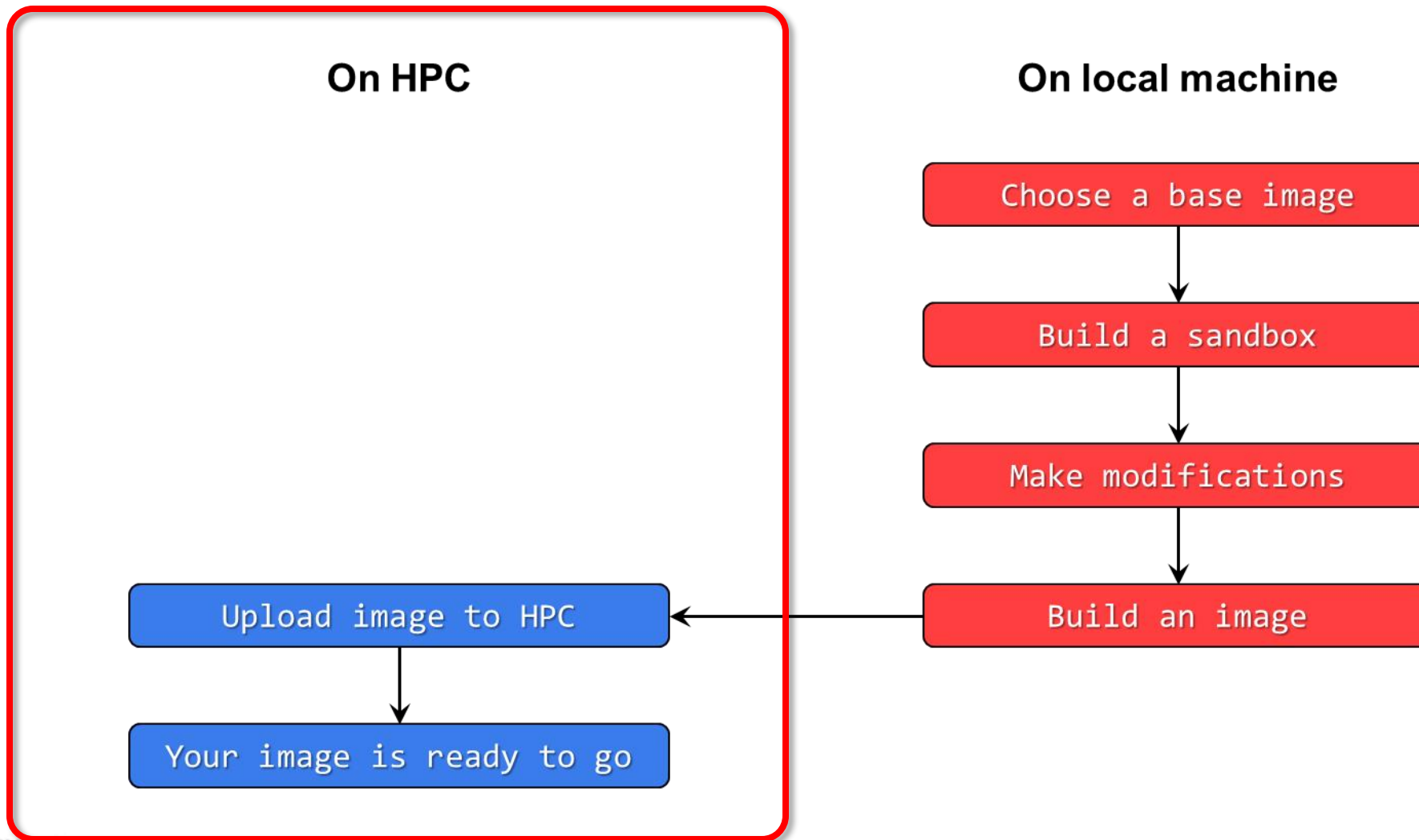
```
$ singularity build [options] <target> <source>
```



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







Conclusion

1. Why Container?

- 1) Problems
- 2) Container & Singularity

2. Run an Existing Container Image

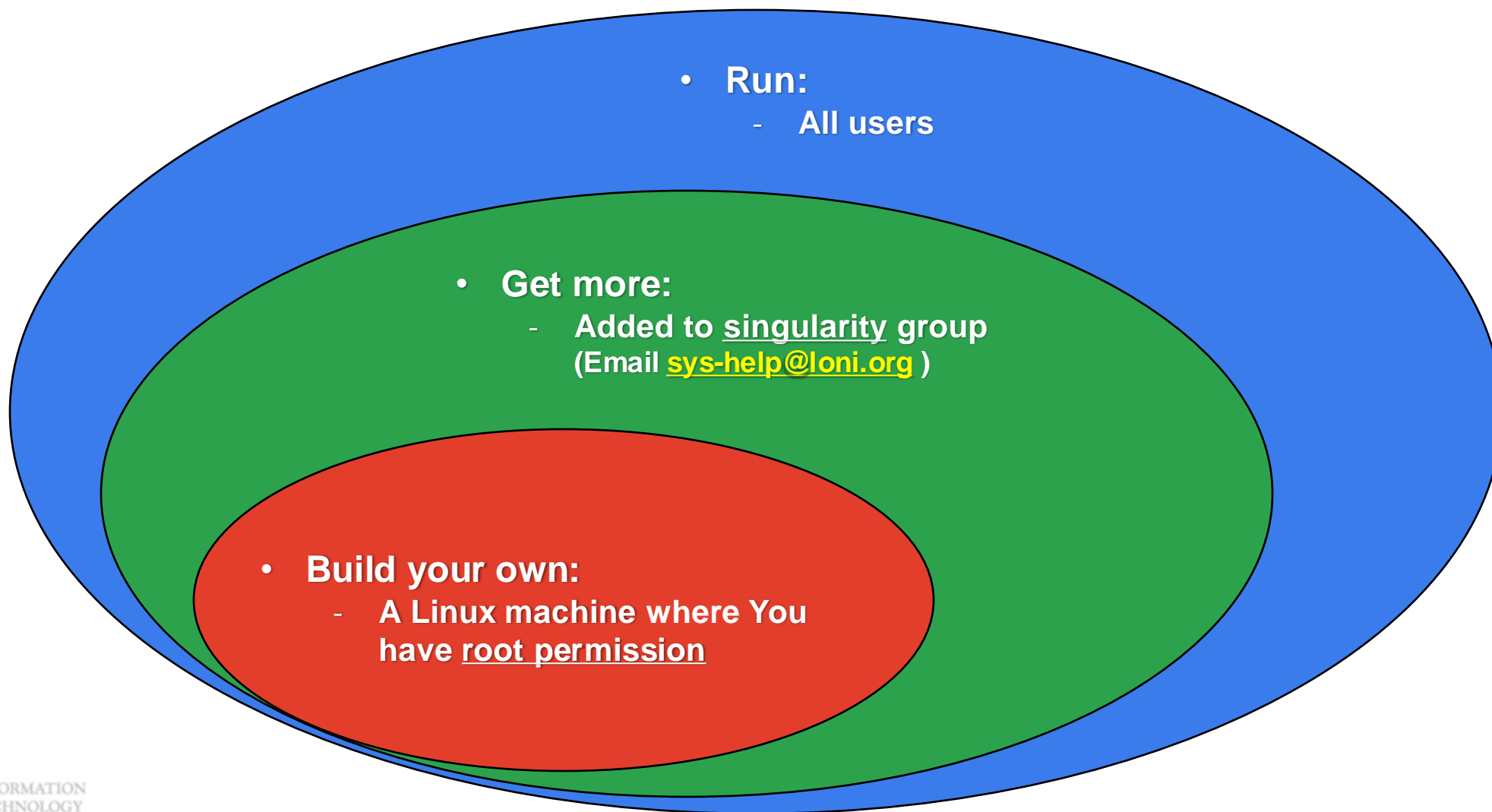
- 1) What you need
- 2) Basic commands
- 3) Running jobs with Singularity

3. Get More Container Images

- 1) What you need
- 2) Where to get
- 3) How to get

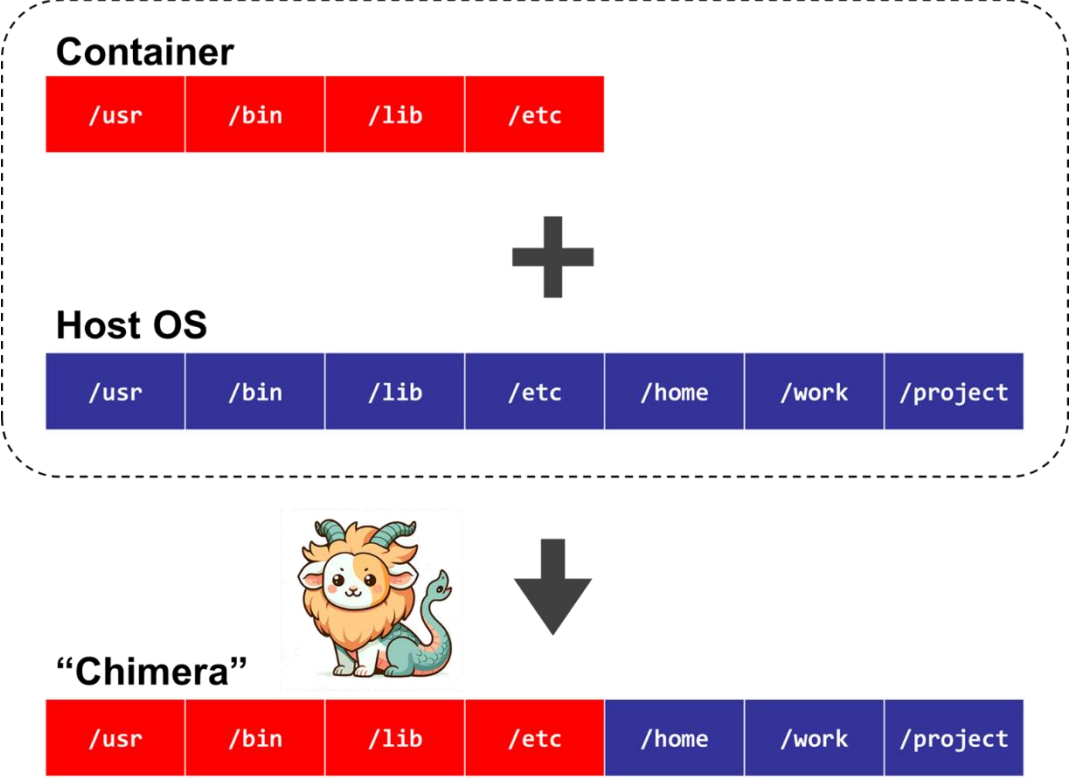
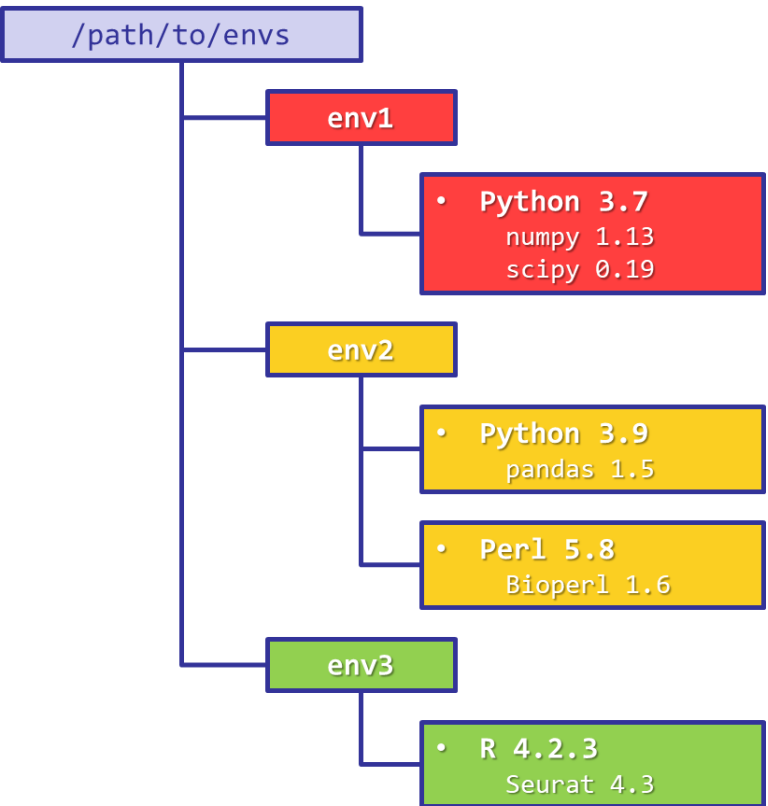
4. Build Your Own Container Image

- 1) What you need
- 2) Typical workflow
- 3) Make it easier - Recipe



To conclude our mini series...

Virtual Environment v.s. Container ?



	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 / Virtual Environments	Singularity / Containers
Good for	<ul style="list-style-type: none">• Less hassle to create and install software from scratch• If you need to frequently make modifications	<ul style="list-style-type: none">• 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 user services**

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

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

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



- 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
```



The screenshot shows the SIMPLE-MOD GUI with the following fields and buttons:

- Module List:** Module name: busco, Module version: 5.6.1. Buttons: Add a new module, Copy current module, Delete selected module.
- 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 (with a Browse button)
 - 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: A table with columns Name and Value, and buttons Add and Delete.
 - Module key template: ./template/template.tcl (with a Browse button)
- Buttons at the bottom:** Generate current module key, Generate all module keys from current database.