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
Nov 1, 2023
Magic Tools to Install & Manage Software

Part 1: CONDA Virtual Environment

Part 2: ingularity Container
Outlines

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

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

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

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

4. Build Your Own 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 Image
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

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

4. Build Your Own 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 Image
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

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

4. Build Your Own 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

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, tBLASTx 2.2+, Augustus 3.2+, Prodigal, MetaEuk, HHMER3.1+, 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/
- 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, tBLASTx 2.2+, Augustus 3.2.2+, Prodigal, Metaeur, HMMER3.1+, SEPP, and R + ggrep 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/
- http://bioinf.uni-greifswald.de/augustus/
- https://github.com/soedinglab/metaeuk
- https://github.com/lyattpd/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, tBLASTx 2.2+, Augustus 3.2+, Prodigal, Metaevk, HMMER3.1+, SEPP, and R + ggplot for the plotting companion script. Some of these tools are necessary only for analysing certain types of organisms and input data, or for specific run modes.

- https://biopython.org/
- https://pandas.pydata.org/
- http://bioinf.uni-greifswald.de/augustus/
- https://github.com/soedinglab/metaevk/
- https://github.com/hyytpd/Prodigal/
- http://hmmer.org/
- https://github.com/smirarak/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

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

```
[jasonli3@smic2 ~]$ module load python/3.6.2-anaconda-tensorflow
[jasonli3@smic2 ~]$ module list
Currently Loaded Modulefiles:
1) python/3.6.2-anaconda-tensorflow
```
b) **Permission denied** (Welcome to HPC!)

```
[jasonli3@smic2 ~]$ module load python/3.6.2-anaconda-tensorflow
[jasonli3@smic2 ~]$ module list
Currently Loaded Modules:
1) python/3.6.2-anaconda-tensorflow
[jasonli3@smic2 ~]$ pip install geos
```
b) Permission denied (Welcome to HPC!)

```
[jasonli3@smic2 ~]$ module load python/3.6.2-anaconda-tensorflow
[jasonli3@smic2 ~]$ module list
Currently Loaded Modulefiles:
  1) python/3.6.2-anaconda-tensorflow
[jasonli3@smic2 ~]$ pip install geos
Collecting geos
  Downloading https://files.pythonhosted.org/packages/49/5b/b8acf74c01187a36aa41b6523de9b3a59c1080a175f1a7a205bb8534/geos-2.2.2-cp36-cp36m-manylinux1_x86_64.whl (409kB 3.0MB/s)

File “/usr/local/packages/python/3.6.2-anaconda/lib/python3.6/os.py”, line 220, in makedirs
    os.makedirs(path)
  File “/usr/local/packages/python/3.6.2-anaconda/lib/python3.6/os.py”, line 220, in makedirs
    mkdir(name, mode)
You are using pip version 9.0.1, however version 20.0.1 is available.
You should consider upgrading via the ‘pip install --upgrade pip’ command.
[jasonli3@smic2 ~]$
```
b) **Permission denied** (Welcome to HPC!)

- If you ask Google / ChatGPT...

```plaintext
$ sudo yum install ...
$ sudo apt-get install ...
$ sudo make install
```
1) Problems

1. Why Container?

2. Run

3. Get More

4. Build your own

c) Conflicted packages

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

Tensorflow 1.13

PyTorch > 1.5
1) Problems

d) Sharing / Migrating your environment

- 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?
1) Problems

Magic Tools to Install / Manage Software

Part 1: **CONDA** Virtual Environment

Part 2: **ingularity** Container
# Outlines

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

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

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

## 4. Build Your Own 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
  - Only virtualize Linux on Linux
2) Container & Singularity

a) What is a container?

- **Contained**
  
  All dependencies can be installed within the container

- **Isolated**
  
  Whatever happens in a container stays in that container…
2) Container & Singularity

a) What is a container?

• Container image:
  - One single binary file that packs everything
  - *.sif / *.simg
2) Container & Singularity

a) What is a **container**?

What the user sees when running the container:

- `/etc`
- `/usr`
- `/lib`
- `/home`
- `/work`
- `/project`

Inside a container:

- `/etc`
- `/usr`
- `/lib`

On host OS:

- `/etc`
- `/usr`
- `/lib`
- `/home`
- `/work`
- `/project`
2) Container & Singularity

b) How does it solve my problems?

- **Dependency issue**
  - Pack all dependencies 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

- **Share / Migrate**
  - Copy-paste image
c) What is **Singularity**?

Technology →
2) Container & Singularity

c) What is Singularity?

↑ Software system that implements the technology
2) Container & Singularity

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

---

**Summary**

1. Why Container?
2. Run
3. Get More
4. Build your own
1. Why Container?
   1) Problems
   2) Container & Singularity

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

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

4. Build Your Own 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 Image
   1) What you need
   2) Basic commands
   3) Running jobs with Singularity

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

4. Build Your Own 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: QB2, QB3

  b) Only on computing nodes
     × Not available on head nodes
     ✓ Must start a job (interactive & batch)

  c) To all users
     × No additional action required
## 1) What you need

<table>
<thead>
<tr>
<th>To …</th>
<th>What you need …</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run an existing image</td>
<td>• Access to our HPC systems</td>
</tr>
<tr>
<td></td>
<td>- An active account</td>
</tr>
<tr>
<td></td>
<td>- An active allocation</td>
</tr>
</tbody>
</table>
1) What you need

- Run:
  - All users
Outlines

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

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

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

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

a) Available images

- On all clusters: /home/admin/singularity

```
(base) [jasonli3@mike1 ~]$ ls /home/admin/singularity
abinit.v9.8.3.sif       jax-0.4.13-gpu-jupyter.sif
alphafold-catgumag-2.2.sif MuST-1.8.7.sif
bowtie2-2.5.1.sif       openpose.sif
busco-5.4.7.sif         pytorch-2.0.1-gpu-jupyter.sif
clara.400-1.sif         rstudio-2023.03.1-446-rocky8.sif
deepmd-kit_2.0.3_cuda11.3_gpu.sif salmon-1.10.2.sif
deepmd-kit_2.2.1_cuda11.6_gpu.sif tensorflow-2.13.0-gpu-jupyter.sif
delft3dfm_r142632_05032023.sif trinity-2.15.1.sif
delft3dfm_r142632.sif    ubuntu-training.sif
delft3d_r142586.sif
```
## 2) Basic commands

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

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>singularity shell &lt;container&gt;</code></td>
<td>Starts a shell in the image</td>
</tr>
</tbody>
</table>

**Try me:** `/home/admin/singularity/ubuntu-training.sif`
## 2) Basic commands

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

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>singularity shell [options] &lt;container&gt;</td>
<td>Starts a shell in the image</td>
</tr>
</tbody>
</table>

**[Options]**

- `-B /path/to/bind`
  - Bind a path(s)
  - /home is bound by default
- `--nv`
  - Enable Nvidia GPU
2) Basic commands

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

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>singularity exec &lt;container&gt; &lt;command&gt;</td>
<td>Execute a command in the image</td>
</tr>
</tbody>
</table>

Try me: /home/admin/singularity/ubuntu-training.sif
2) Basic commands

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

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>singularity exec [options] &lt;container&gt; &lt;command&gt;</td>
<td>Execute a command in the image</td>
</tr>
<tr>
<td>[Options] -B /path/to/bind</td>
<td>Bind a path(s)</td>
</tr>
<tr>
<td></td>
<td>• /home is bound by default</td>
</tr>
<tr>
<td>--nv</td>
<td>Enable Nvidia GPU</td>
</tr>
</tbody>
</table>

2. Run

3. Get More

4. Build your own
### 2) Basic commands

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

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>singularity run [options] &lt;container&gt;</code></td>
<td>Run a prewritten script</td>
</tr>
<tr>
<td>[Options]</td>
<td>Bind a path(s)</td>
</tr>
<tr>
<td><code>-B /path/to/bind</code></td>
<td>• /home is bound by default</td>
</tr>
<tr>
<td><code>--nv</code></td>
<td>Enable Nvidia GPU</td>
</tr>
</tbody>
</table>
2) Basic commands

- Quick recap

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>singularity run [options] &lt;container&gt;</code></td>
<td>Run a prewritten script</td>
</tr>
<tr>
<td><code>singularity exec [options] &lt;container&gt; &lt;command&gt;</code></td>
<td>Execute a command in the image</td>
</tr>
<tr>
<td><code>singularity run [options] &lt;container&gt;</code></td>
<td>Run a prewritten script</td>
</tr>
</tbody>
</table>
1. Why Container?
   1) Problems
   2) Container & Singularity

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

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

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

- Job types and commands

<table>
<thead>
<tr>
<th>Job Type</th>
<th>Commands</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| 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

<table>
<thead>
<tr>
<th>PBS</th>
<th>Slurm</th>
</tr>
</thead>
</table>
| #!/bin/bash  
PBS -A <Allocation name>  
PBS -q workq  
PBS -l nodes=1:ppn=20  
PBS -l walltime=24:00:00  

cd /to/work/directory  

IMG=/home/admin/singularity/ubuntu-training.sif  
singularity exec -B /work,/project $IMG \  
python myjob.py | 

#!/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
Summary

• Run:
  - All users
Outlines

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

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

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

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

- Available images

  On all clusters: `/home/admin/singularity`

```
(base) [jasonli3@mike1 ~]$ ls /home/admin/singularity
abinit.v9.8.3.sif
alphafold-catgumag-2.2.sif
bowtie2-2.5.1.sif
busco-5.4.7.sif
clara.400-1.sif
deqmd-kit_2.0.3_cuda11.3_gpu.sif
deqmd-kit_2.2.1_cuda11.6_gpu.sif
delft3dfm_r142632_05032023.sif
delft3dfm_r142632.sif
delft3d_r142586.sif
jax-0.4.13-gpu-jupyter.sif
MuST-1.8.7.sif
openpose.sif
pytorch-2.0.1-gpu-jupyter.sif
rstudio-2023.03.1-446-rocky8.sif
salmon-1.10.2.sif
tensorflow-2.13.0-gpu-jupyter.sif
trinity-2.15.1.sif
ubuntu-training.sif
```
Outlines

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

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

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

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

2. Run

```
(base) [jasonli3@mike4]
~$ ll /home/admin/singularity
```

1. Why Container?

3. Get More

4. Build your own
1. Why Container?

2. Run

3. Get More

4. Build your own

---

Singularity images must belong to “singularity” group to run on our clusters!

```bash
(base) [jasonli3@mike4 ~]$ ll /home/admin/singularity
```

- total 78260932
- singularity
- abinit.v9.8.3.sif
- alphafold-catgumag-2.2.sif
- bowtie2-2.5.1.sif
- busco-5.4.7.sif
- clara.400-1.sif
- pytorch-2.0.1-gpu-jupyter.sif
- rstudio-2023.03.1-446-rocky8.sif
- salmon-1.10.2.sif
- tensorflow-2.13.0-gpu-jupyter.sif
- trinity-2.15.1.sif
- ubuntu-training.sif
# 1) What you need

<table>
<thead>
<tr>
<th>To …</th>
<th>What you need …</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run an existing image</td>
<td>• Access to our HPC systems</td>
</tr>
<tr>
<td></td>
<td>- An active account</td>
</tr>
<tr>
<td></td>
<td>- An active allocation</td>
</tr>
<tr>
<td>Get more images</td>
<td>• (Everything above)</td>
</tr>
<tr>
<td></td>
<td>• Added to <strong>Singularity</strong> group</td>
</tr>
<tr>
<td></td>
<td>- Needed to change <strong>group ownership</strong></td>
</tr>
<tr>
<td></td>
<td>- Send email to <a href="mailto:sys-help@loni.org">sys-help@loni.org</a> to request</td>
</tr>
</tbody>
</table>
1) What you need

• Run:
  - All users

• Get more:
  - Added to singularity group
Outlines

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

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

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

4. Build Your Own 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
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**

<table>
<thead>
<tr>
<th>Name</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- For biology</td>
</tr>
<tr>
<td>Nvidia NGC</td>
<td>- <a href="https://catalog.ngc.nvidia.com(containers)">https://catalog.ngc.nvidia.com(containers)</a></td>
</tr>
<tr>
<td></td>
<td>- For Nvidia GPU</td>
</tr>
<tr>
<td>Bitnami</td>
<td>- <a href="https://bitnami.com/stacks/containers">https://bitnami.com/stacks/containers</a></td>
</tr>
<tr>
<td></td>
<td>- By VmWare</td>
</tr>
<tr>
<td><strong>Docker Hub</strong></td>
<td><strong>Quay.io</strong></td>
</tr>
<tr>
<td></td>
<td>- Don’t just trust everything you see there!</td>
</tr>
<tr>
<td></td>
<td>- Look for Docker Official Image or Verified Publisher</td>
</tr>
<tr>
<td></td>
<td>- Avoid third-party publishers that you don’t know</td>
</tr>
</tbody>
</table>
1. Why Container?
   1) Problems
   2) Container & Singularity

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

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

4. Build Your Own 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

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>singularity pull [options] [target] &lt;source&gt;</code></td>
<td>Pull an image from source</td>
</tr>
<tr>
<td><code>&lt;source&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>docker://container[:tag]</code></td>
<td>Pull a Docker container and convert to Singularity</td>
</tr>
<tr>
<td></td>
<td>• (Compare to Docker command)</td>
</tr>
<tr>
<td></td>
<td><code>docker pull container[:tag]</code></td>
</tr>
<tr>
<td><code>http://www.myexample.com/container_image.sif</code></td>
<td>Download an image file from a web source</td>
</tr>
</tbody>
</table>

**Syntax**:

- `singularity pull [options] [target] <source>`
- `docker://container[:tag]`
- `docker pull container[:tag]`
- `http://www.myexample.com/container_image.sif`
3) How to get

a) Step 1: Pull the image

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>singularity build [options] &lt;target&gt; &lt;source&gt;</td>
<td>Build an image from source (Advanced)</td>
</tr>
<tr>
<td>docker://container[:tag]</td>
<td>Build from a Docker container</td>
</tr>
<tr>
<td>container_image.sif</td>
<td>Build from a local image file</td>
</tr>
<tr>
<td>container_sandbox/</td>
<td>Build from a local sandbox (A directory form of a container)</td>
</tr>
<tr>
<td>container_recipe.def</td>
<td>Build from a recipe (an instruction script of how to build an image)</td>
</tr>
</tbody>
</table>
b) Step 2: Change group ownership

- What if you do not?

```
FATAL: failed to retrieve group information for cvmfs: group: unknown group cvmfs
```

- To solve it, run this:

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

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

1. Why Container?

2. Run
   - All users

3. Get More
   - Added to singularity group

4. Build your own
Summary

- **Steps:**
  
a) Step 1: Pull the image

b) Step 2: Change group ownership
1. Why Container?
   1) Problems
   2) Container & Singularity

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

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

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

- Scenarios:
  - I did not find any container release. Need to DIY.
  - It’s too complicated to install on cluster, but 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 Image

- Idea

```
Software

sudo
yum
apt-get

HPC systems

A local machine
```
4. Build Your Own Image

- **Idea**

![Diagram showing the process of building and transferring container images from HPC systems to a local machine using commands like `sudo`, `yum`, `apt-get`, etc.](image-url)
4. Build Your Own Image

- Idea

- Software

- HPC systems

- Build & transfer container images

- A local machine

- sudo
- yum
- apt-get

Build & transfer container images
1. Why Container?
   1) Problems
   2) Container & Singularity

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

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

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

<table>
<thead>
<tr>
<th>To …</th>
<th>What you need …</th>
</tr>
</thead>
</table>
| Run an existing image | • Access to our HPC systems  
  - An active account  
  - An active allocation |
| Get more images | • (Everything above)  
  • Added to **Singularity** group  
  - Needed to change **group ownership**  
  - Send email to [sys-help@loni.org](mailto:sys-help@loni.org) to request |
| Build your own image | • (Everything above)  
  • A local Linux machine, where:  
  - You have **root permission**  
  - **Singularity** is installed |
1) What you need

- **Run:**
  - All users

- **Get more:**
  - Added to singularity group

- **Build your own:**
  - A Linux machine where:
    - You have root permission
    - Singularity is installed
1) What you need

- Install Singularity
  
  1) What you need
  
  - Joined Linux Foundation
  - Easier installation
  
  2) Community supported
  - Installed on our clusters

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

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

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

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

Typical workflow:

On HPC:
1. Choose a base image
2. Build a sandbox
3. Make modifications
4. Test the sandbox (local test)
5. Build an image
6. Upload image to HPC
7. Test the image (on HPC, w/ jobs)
8. Your image is ready to go

On local machine:
1. Choose a base image
2. Build a sandbox
3. Make modifications
4. Test the sandbox (local test)
5. Build an image

(If needed)

1. Why Container?
2. Run
3. Get More
4. Build your own
2) Typical workflow

a) Choose a base image

<table>
<thead>
<tr>
<th>Common choices</th>
<th>Typical scenarios</th>
</tr>
</thead>
</table>
| **A fresh, minimum 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

![screenshot of singularity build command with options, target, and source placeholders]

<table>
<thead>
<tr>
<th>&lt;source&gt;</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docker://container[:tag]</td>
<td>Build from a Docker container</td>
</tr>
<tr>
<td>container_image.sif</td>
<td>Build from a local image file</td>
</tr>
<tr>
<td>container_sandbox/</td>
<td>Build from a local sandbox (a directory form of a container)</td>
</tr>
<tr>
<td>container_recipe.def</td>
<td>Build from a recipe (an instruction script of how to build an image)</td>
</tr>
</tbody>
</table>
### 2) Typical workflow

**b) Build a sandbox**

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

<table>
<thead>
<tr>
<th><code>&lt;source&gt;</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>docker://container[:tag]</code></td>
<td>Build from a Docker container</td>
</tr>
<tr>
<td><code>container_image.sif</code></td>
<td>Build from a local image file</td>
</tr>
<tr>
<td><code>container_sandbox/</code></td>
<td>Build from a local sandbox (A directory form of a container)</td>
</tr>
<tr>
<td><code>container_recipe.def</code></td>
<td>Build from a recipe (an instruction script of how to build an image)</td>
</tr>
</tbody>
</table>
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

d) Build an image from sandbox

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

<table>
<thead>
<tr>
<th>&lt;source&gt;</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docker://container[:tag]</td>
<td>Build from a Docker container</td>
</tr>
<tr>
<td>container_image.sif</td>
<td>Build from a local image file</td>
</tr>
<tr>
<td>container_sandbox/</td>
<td>Build from a local sandbox (a directory form of a container)</td>
</tr>
<tr>
<td>container_recipe.def</td>
<td>Build from a recipe (an instruction script of how to build an image)</td>
</tr>
</tbody>
</table>
d) Build an image from sandbox

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

Modify with “sudo”? Build with “sudo”!
2. Run

- Quick recap

<table>
<thead>
<tr>
<th>To ...</th>
<th>You need to ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build a sandbox</td>
<td><code>$ singularity build --sandbox ...</code></td>
</tr>
<tr>
<td>Modify a sandbox</td>
<td><code>$ sudo singularity shell --writable ...</code></td>
</tr>
<tr>
<td>Build an image from sandbox</td>
<td><code>$ sudo singularity build ...</code></td>
</tr>
</tbody>
</table>
2) Typical workflow

e) Upload image to HPC and run

Now! The moment of truth!
2) Typical workflow

On HPC

(If needed)

Upload image to HPC

Test the image (on HPC, w/ jobs)

Your image is ready to go

On local machine

Choose a base image

Build a sandbox

Make modifications

Test the sandbox (local test)

Build an image

(If needed)
1. Why Container?
   1) Problems
   2) Container & Singularity

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

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

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

- Why?

On HPC
- Upload image to HPC
- Test the image (on HPC, w/ jobs)
- Your image is ready to go

On local machine
1. Choose a base image
2. Build a sandbox
3. Make modifications
4. Test the sandbox (local test)
5. Build an image

(If needed)
### 3) Make it easier - Recipe

#### Why?

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Flexibility</td>
<td>• Repeatability</td>
</tr>
<tr>
<td></td>
<td>• Minimizing image size</td>
</tr>
</tbody>
</table>

#### Solution:

- **Recipe**: A text file containing instructions to build a container
3) Make it easier - Recipe

- Why?

On HPC

(If needed)

Upload image to HPC

Test the image (on HPC, w/ jobs)

Your image is ready to go

On local machine

Choose a base image

Build a sandbox

Make modifications

Test the sandbox (local test)

Build an image

(If needed)

2. Run

3. Get More

4. Build your own
3) Make it easier - Recipe

- Why?

On HPC

Build image from recipe

On local machine

(If needed)

Upload image to HPC

Test the image (on HPC, w/ jobs)

Your image is ready to go
3) Make it easier - Recipe

a) What does a recipe look like?

```ruby
class ruby_def:
    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

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

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
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)
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

**b) Build the recipe**

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

<table>
<thead>
<tr>
<th>&lt;source&gt;</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docker://container[:tag]</td>
<td>Build from a Docker container</td>
</tr>
<tr>
<td>container_image.sif</td>
<td>Build from a local image file</td>
</tr>
<tr>
<td>container_sandbox/</td>
<td>Build from a local <strong>sandbox</strong> (A directory form of a container)</td>
</tr>
<tr>
<td>container_recipe.def</td>
<td>Build from a <strong>recipe</strong> (an instruction script of how to build an image)</td>
</tr>
</tbody>
</table>

Summary

1. Why Container?
2. Run
   - All users
3. Get More
   - Added to singularity group
4. Build your own
   - A Linux machine where:
     - You have root permission
     - Singularity is installed
Summary

1. Why Container?
2. Run
3. Get More
4. Build your own

On HPC

(If needed)

Upload image to HPC

Test the image (on HPC, w/ jobs)

Your image is ready to go

On local machine

Choose a base image

Build a sandbox

Make modifications

Test the sandbox (local test)

Build an image

(If needed)
Summary

1. Why Container?

2. Run

3. Get More

4. Build your own

On HPC

(If needed)

Upload image to HPC

Test the image (on HPC, w/ jobs)

Your image is ready to go

On local machine

Choose a base image

Build a sandbox

Make modifications

Test the sandbox (local test)

Build an image

(If needed)
Conclusion
Conclusion

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

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

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

4. Build Your Own 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

• Build your own:
  - A Linux machine where:
    - You have root permission
    - Singularity is installed
To conclude our mini series...
- 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
  env3
    • Perl 5.8
        Bioperl 1.6
    • R 4.2.3
        Seurat 4.3
```

Container Engine

Host OS

Infrastructures

- Container 1
  - App 1
  - Sys files (OS 1)

- Container 2
  - App 2
  - Sys files (OS 2)

- Container 3
  - App 3
  - Sys files (OS 3)
# Conda vs Singularity

<table>
<thead>
<tr>
<th></th>
<th>Conda / Virtual Environments</th>
<th>Singularity / Containers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Availability</strong></td>
<td>All users</td>
<td>All users, but may need additional things</td>
</tr>
<tr>
<td><strong>Functionality</strong></td>
<td>Good (usually)</td>
<td>Better (more likely to work)</td>
</tr>
<tr>
<td><strong>Self-contained</strong></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Isolation</strong></td>
<td>Yes (but still accessible from outside)</td>
<td>Perfect</td>
</tr>
<tr>
<td><strong>Editability</strong></td>
<td>Yes</td>
<td>No (image file itself) / Yes (a detour with sandbox)</td>
</tr>
<tr>
<td><strong>Disk usage</strong></td>
<td>Large</td>
<td>Smaller</td>
</tr>
<tr>
<td><strong>Portability</strong></td>
<td>Possible (with .yml file)</td>
<td>Great (copy-paste one file)</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td>Fair</td>
<td>Good</td>
</tr>
<tr>
<td><strong>Ease of use</strong></td>
<td>Good</td>
<td>May require more effort</td>
</tr>
</tbody>
</table>
## Conda vs Singularity

<table>
<thead>
<tr>
<th>Good for</th>
<th>Conda / Virtual Environments</th>
<th>Singularity / Containers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Less hassle (usually?) to create and install software from scratch</td>
<td>• Less hassle if the developer releases a working container</td>
</tr>
<tr>
<td></td>
<td>• If you need to frequently make modifications</td>
<td>• If you don’t or don’t want to make changes after it is created</td>
</tr>
<tr>
<td></td>
<td>• If you need to access files from outside of the environment (e.g., compiling a code that uses some files in the virtual environment as dependencies)</td>
<td>• Portability</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Reduce disk usage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Your system admins yelled at you about security risk</td>
</tr>
</tbody>
</table>
Contact user services

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