

# Tutorial: Introduction to Containers for Scientific Container-Native Workflows: **Singularity** on **ACES**

Richard Lawrence  
10/22/2024



*developed for*



# Outline

- Overview of Containers
- Overview of Singularity
- Getting Started
- Container Image Sources
- Working with Images
- Working with Containers
- Containerized Scientific Applications on ACES
  - PyTorch
  - LAMMPS

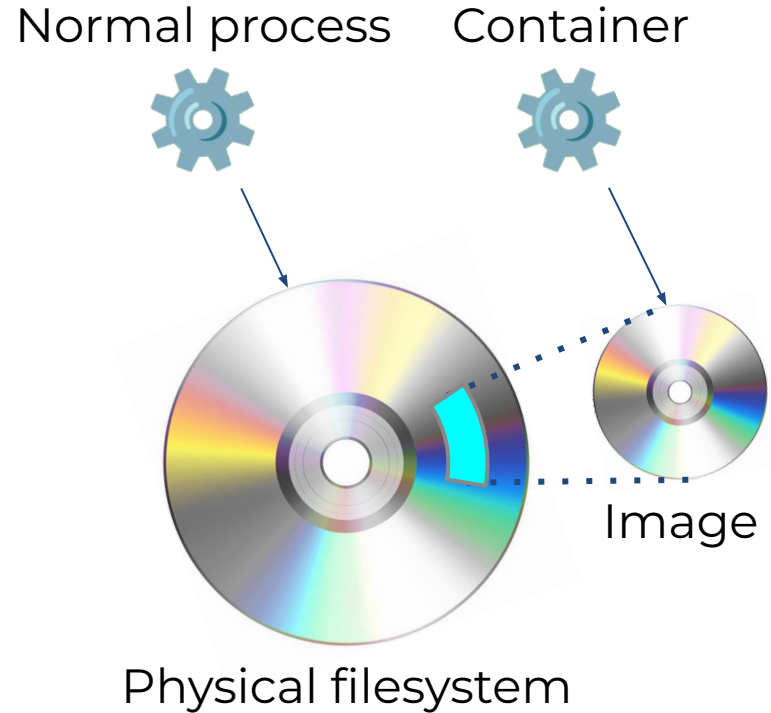
# Learning Resources

- Slides on the course web page  
[https://hprc.tamu.edu/training/aces\\_containers\\_scientific.html](https://hprc.tamu.edu/training/aces_containers_scientific.html)  
highly recommended for working along)
- HPRC's Knowledge Base  
<https://hprc.tamu.edu/kb/Software/Singularity/>
- HPRC on YouTube  
<https://www.youtube.com/c/TexasAMHPRC>
- ACCESS Links  
<https://support.access-ci.org/ci-links>

# Overview of Containers

# What Are Containers?

- A container is a process (⚙️) that has its own **view** of local resources:
  - **Filesystem**
  - User IDs
  - Network
  - etc.
- Example: this container (⚙️ on the right) sees the **image** instead of the physical filesystem



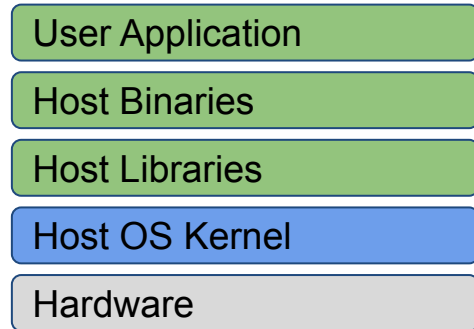
# Why Use Containers?

- **Shareability:**
  - Share your container image file by uploading to a public repository
  - Use images shared by others
- **Portability:**
  - Use images on any computer with the same architecture (x84-64)
- **Reproducibility:**
  - Container users are largely unaffected by changes to the cluster environments

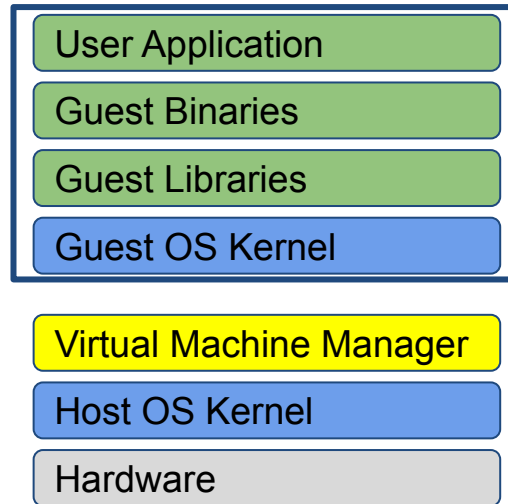
# What Goes In Container Images?

- Unlike in VMs, the OS Kernel is not duplicated
- Container images are smaller than VM images

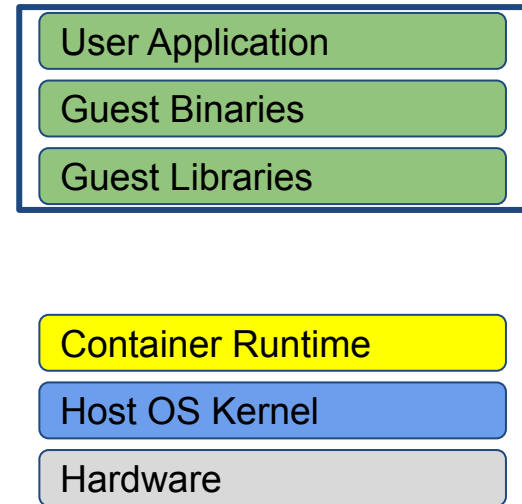
## Local Build, or “Bare metal”



## Virtual Machine

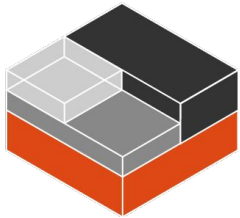


## Container

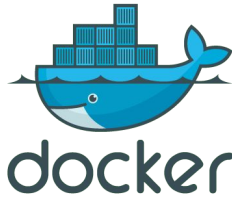


# Popular Container Runtimes

Instant deployment to users on different devices!



LXC  
2008



Docker  
2013

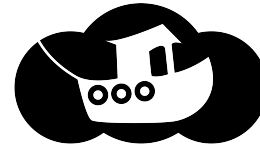


Singularity  
2015



SHIFTER

Shifter 2016



Charliecloud

Charliecloud  
2017



Podman  
2018



# Overview of Singularity

# Singularity

- An easy-to-use, high-performance container solution



**Deploying Secure Container  
Solutions from Edge to Exascale**

Presented by



# Singularity is Apptainer



# Singularity Features

- Singularity is a container runtime and an image builder
- Singularity can read and convert Docker images
- Filesystem inside container is isolated
- User inside container is the same as the user outside
- Works with high-performance cluster technologies

Read more in the Apptainer manual

<https://apptainer.org/user-docs/3.8/>

# Singularity on ACES

- Singularity is available on Compute nodes
  - Singularity activities are too cpu-intensive for login nodes.
- Singularity images can be large on disk. Be aware of your storage quota. (`/scratch` > `/home`)
- Some container activities may be too I/O-intense for the shared network filesystem. Be courteous to others and use a local filesystem for large image operations.

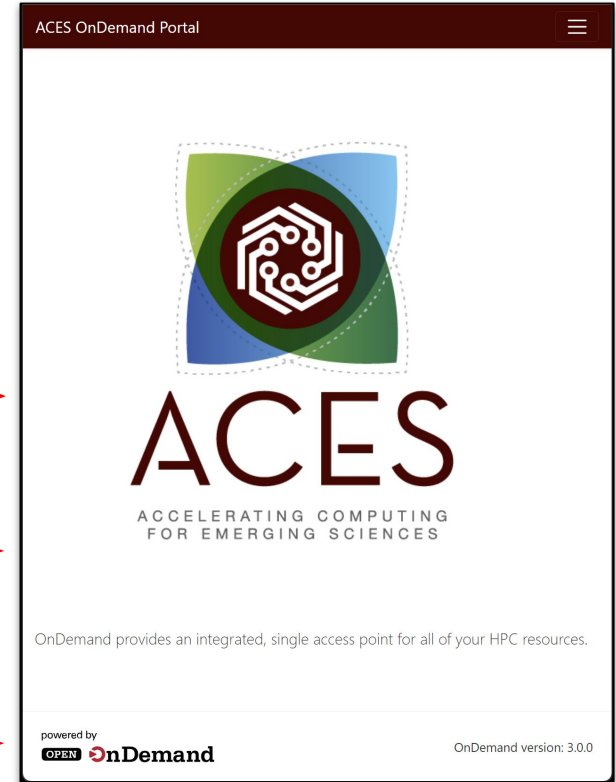
# Getting Started

# ACES Portal

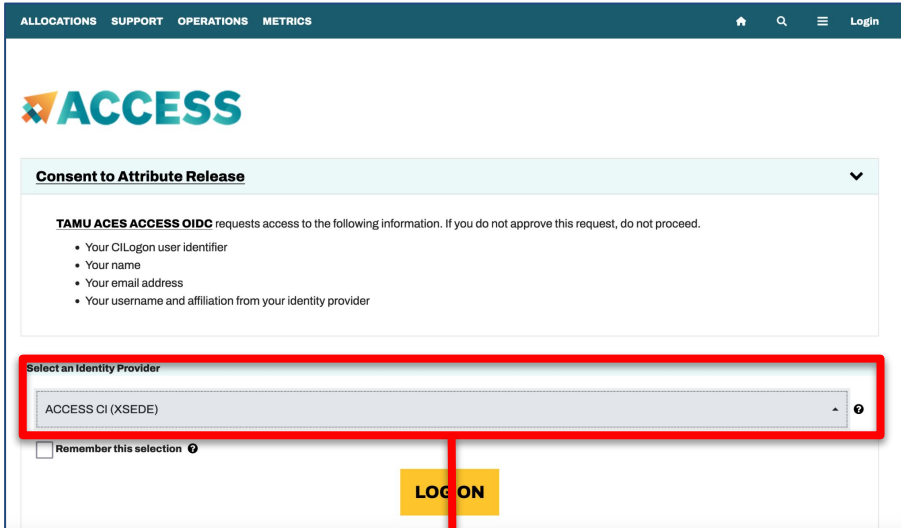


ACES Portal [portal-aces.hprc.tamu.edu](https://portal-aces.hprc.tamu.edu)  
is the web-based user interface for the ACES cluster

Open OnDemand (OOD) is an advanced web-based  
graphical interface framework for HPC users

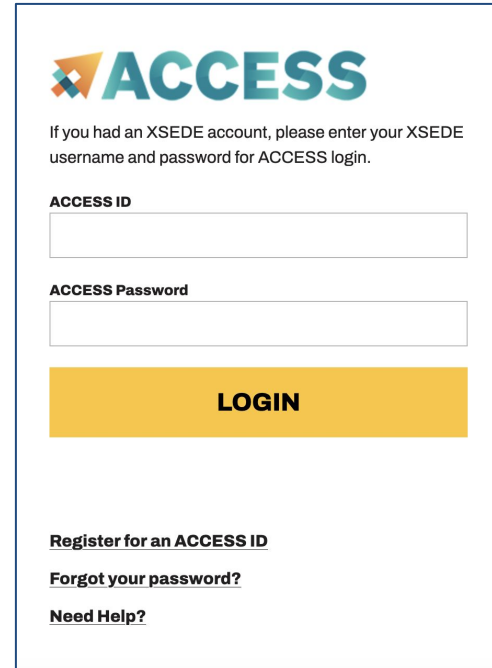


# Accessing ACES via the Portal (ACCESS)



The screenshot shows the ACCESS portal interface. At the top is a navigation bar with links: ALLOCATIONS, SUPPORT, OPERATIONS, METRICS, and a Login link. Below the navigation bar is the ACCESS logo. A section titled "Consent to Attribute Release" is expanded, showing a list of attributes requested by TAMU ACES ACCESS OIDC: Your CILogon user identifier, Your name, Your email address, and Your username and affiliation from your identity provider. Below the consent section is a "Select an Identity Provider" dropdown menu. The dropdown is open, showing "ACCESS CI (XSEDE)" as the selected option. A red rectangle highlights the dropdown menu. Below the dropdown is a checkbox labeled "Remember this selection". At the bottom of the form is a yellow "LOG ON" button. A red line points from the "LOG ON" button to the text below.

Select the Identity Provider appropriate for your account.



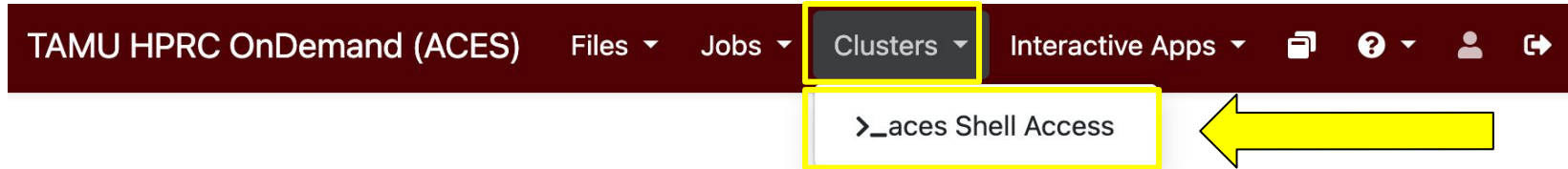
The screenshot shows the ACCESS portal login page. At the top is the ACCESS logo. Below the logo is a message: "If you had an XSEDE account, please enter your XSEDE username and password for ACCESS login." Below the message are two input fields: "ACCESS ID" and "ACCESS Password". Below the input fields is a yellow "LOGIN" button. At the bottom of the form are three links: "Register for an ACCESS ID", "Forgot your password?", and "Need Help?".

Log-in using your ACCESS or institutional credentials.



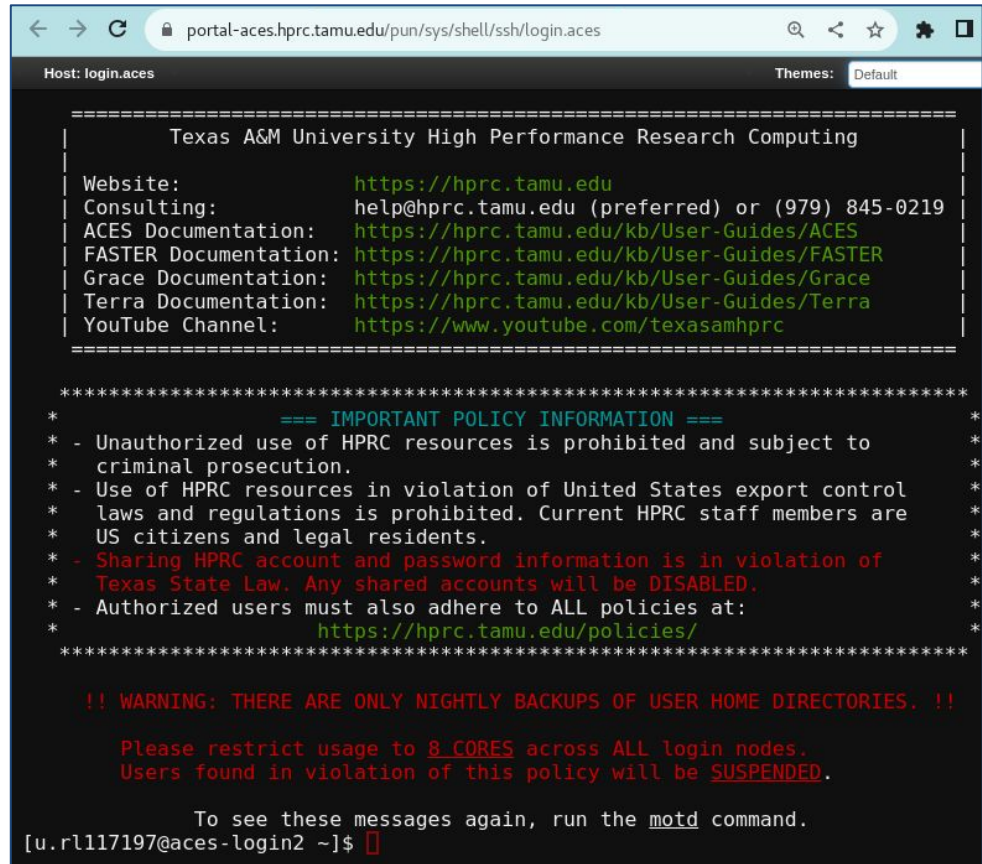
# Get a Shell on ACES

Click on “Clusters” menu → \_aces Shell Access



# Success!

Welcome to the  
ACES login node.



```
portal-aces.hprc.tamu.edu/pun/sys/shell/ssh/login.aces
Host: login.aces
Themes: Default

=====
|               Texas A&M University High Performance Research Computing               |
|=====|
| Website:                https://hprc.tamu.edu                                     |
| Consulting:             help@hprc.tamu.edu (preferred) or (979) 845-0219          |
| ACES Documentation:     https://hprc.tamu.edu/kb/User-Guides/ACES                  |
| FASTER Documentation:   https://hprc.tamu.edu/kb/User-Guides/FASTER                |
| Grace Documentation:    https://hprc.tamu.edu/kb/User-Guides/Grace                 |
| Terra Documentation:    https://hprc.tamu.edu/kb/User-Guides/Terra                 |
| YouTube Channel:        https://www.youtube.com/texasamhprc                     |
|=====|

*****
*                               === IMPORTANT POLICY INFORMATION ===                               *
* - Unauthorized use of HPRC resources is prohibited and subject to criminal prosecution. *
* - Use of HPRC resources in violation of United States export control laws and regulations *
*   is prohibited. Current HPRC staff members are US citizens and legal residents.        *
* - Sharing HPRC account and password information is in violation of Texas State Law. Any *
*   shared accounts will be DISABLED.                                                    *
* - Authorized users must also adhere to ALL policies at:                               *
*   https://hprc.tamu.edu/policies/                                                       *
*****

!! WARNING: THERE ARE ONLY NIGHTLY BACKUPS OF USER HOME DIRECTORIES. !!

Please restrict usage to 8 CORES across ALL login nodes.
Users found in violation of this policy will be SUSPENDED.

To see these messages again, run the motd command.
[u.rl117197@aces-login2 ~]$
```

# Set Up Your Tutorial Environment

```
cd $SCRATCH  
mkdir s_tutorial  
cd s_tutorial  
pwd
```

```
export TRAINING=/scratch/training/singularity  
ls $TRAINING
```

# Set Up Your Singularity Environment

Get to a compute node from the login node

```
srun --time=120 --mem=4G --pty bash -i
```

Return to your tutorial directory (if necessary)

```
cd $SCRATCH/s_tutorial
```

following along live? add:  
--reservation=containers

Set your singularity cache directory for temporary files

```
export SINGULARITY_CACHEDIR=$TMPDIR
```

Connect to the internet for fetching images

```
module load WebProxy
```

# Your First Singularity Container

Singularity can fetch an image *and* launch a shell in one line.

```
singularity shell --help
```

Fetch an image and launch a shell from it

```
singularity shell docker://almalinux:8  
cat /etc/redhat-release  
exit
```

The ACES compute nodes also have Red Hat linux installed.

```
cat /etc/redhat-release
```



**Congratulations!**

**Welcome to containers**

[WWW.FUNIMADA.COM](http://WWW.FUNIMADA.COM)

# Container Image Sources

# Popular Repositories

The most common repository is:

- Docker Hub

Others repositories include:

- Singularity Hub
- Singularity Library
- NVIDIA GPU Cloud
- Quay.io
- BioContainers

See

<https://hprc.tamu.edu/kb/Software/Singularity/Examples/#popular-repositories>

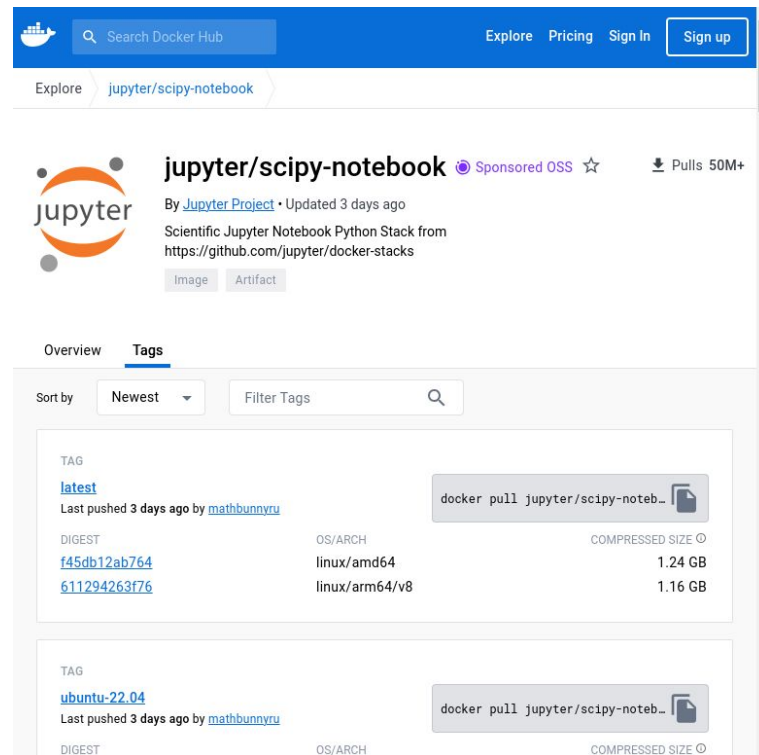


# Docker Hub Example

Docker Hub repositories are named in the form `<group>/<name>` similar to GitHub.

Each image within a repository has a `<tag>` that describes how and when it was built.

This example is  
`jupyter/scipy-notebook:latest`



The screenshot shows the Docker Hub interface for the `jupyter/scipy-notebook` repository. The page header includes a search bar and navigation links. The repository page shows the Jupyter logo, the repository name, and a 'Sponsored OSS' badge. It indicates the repository was updated 3 days ago and has over 50M pulls. The 'Tags' tab is selected, showing a list of tags. The 'latest' tag is highlighted, with a 'docker pull' button. Below the tag, a table lists the digests and OS/architectures for the latest tag.

DIGEST	OS/ARCH	COMPRESSED SIZE
<a href="#">f45db12ab764</a>	linux/amd64	1.24 GB
<a href="#">611294263f76</a>	linux/arm64/v8	1.16 GB

# Singularity Pull

Singularity can fetch images from repositories and also convert them to the singularity file format at the same time.

```
singularity pull [target-filename] <source>
```

Where <source> refers to something on the internet. The syntax depends on where the source is located.

and [target-filename] includes the file extension.

# Singularity Pull Example

The <source> argument for Docker images looks like

```
docker://<group>/<name>[:<tag>]
```

Therefore the pull command for the Jupyter example is,

```
singularity pull docker://jupyter/scipy-notebook:latest
```

(Download now or copy from \$TRAINING; we will need this later)

The default filename will be `scipy-notebook_latest.sif`

# Working with Images

# Singularity Image Formats

- Singularity container images come in two main formats:
  1. Directory
  2. Single file. Singularity uses the SIF format for single file images. This is the default.
- The `singularity build` tool can convert images in both formats.  
`singularity build --help`
- The `--sandbox` option is used to create directory-format images.

# Singularity Image Exercise

Singularity pull can fetch an image and write to either file format.  
*(note the order of the arguments)*

```
singularity pull almalinux.sif docker://almalinux:8
```

Singularity can convert an image to the directory file format.  
Use the --sandbox argument to specify the directory type.  
*(note the order of the arguments)*

```
singularity build --sandbox $TMPDIR/almalinux almalinux.sif
```

# Singularity Write Exercise

Directory images are writable. Simply add the `--writable` flag to your container command.

```
singularity shell --writable $TMPDIR/almalinux  
mkdir /my_dir  
exit
```

Are the changes still there?

```
singularity shell $TMPDIR/almalinux  
ls /
```

# Singularity Read-only Exercise

SIF files are safe for network file system /scratch.

```
singularity build --fakeroot my_almalinux.sif $TMPDIR/almalinux
```

Are the changes still there?

```
singularity shell my_almalinux.sif  
ls /  
exit
```

What about the --writeable flag?

```
singularity shell --writeable my_almalinux.sif  
no.
```



# Working with Containers

# Launching Processes

Singularity has three methods for launching processes:

- **Interactive:** `singularity shell`
- **Batch processing:** `singularity exec`
- **Container-as-executable:** `singularity run`

# Singularity Run Exercise

Singularity run will execute the default runscript, if one was defined. You may also execute the container directly.

```
singularity pull docker://hello-world  
singularity run hello-world_latest.sif  
Hello from Docker!  
./hello-world_latest.sif  
Hello from Docker!
```

Docker hello-world is a minimal image. This is all it can do.

# Singularity Exec Exercise

Singularity Exec lets you access executables and other commands in a container. This is appropriate for batch jobs.

ACES nodes have Python 3.

```
python3 --version  
Python 3.6.8
```

Our singularity image has a different Python 3.

```
singularity exec scipy-notebook_latest.sif python3 --version  
Python 3.11.6
```

# Working with Files

- Filesystem inside a container is isolated from the real, physical filesystem.
- To access your files, ensure the directory is *mounted*.
- By default, Singularity will mount `$HOME` and `$PWD` if it can.
- To specify additional directories, use the `SINGULARITY_BINDPATH` environment variable or the `--bind` command line option.

# Working with Files Exercise

Recommended that you mount `/scratch` to get access to your data storage, and `/tmp` to get access to the local disk on the node.

```
singularity shell --bind "/scratch,/tmp" <image>  
mkdir $TMPDIR/my_dir; exit  
ls $TMPDIR
```

Notice that your variables like `$TMPDIR` get passed into the container by default.

*(singularity on ACES already binds these directories by default)*

# Singularity Batch Example

```
#!/bin/bash

## JOB SPECIFICATIONS
#SBATCH --job-name=sing_test          #Set the job name to "sing_test"
#SBATCH --time=00:10:00               #Set the wall clock limit to 1hr and 30min
#SBATCH --ntasks=4                   #Request 4 task
#SBATCH --mem=2560M                   #Request 2560MB (2.5GB) per node
#SBATCH --output=sing_test.%j         #Send stdout/err to "sing_test.[jobID]"
```

```
export SINGULARITY_BINDPATH="/scratch,/tmp"
```

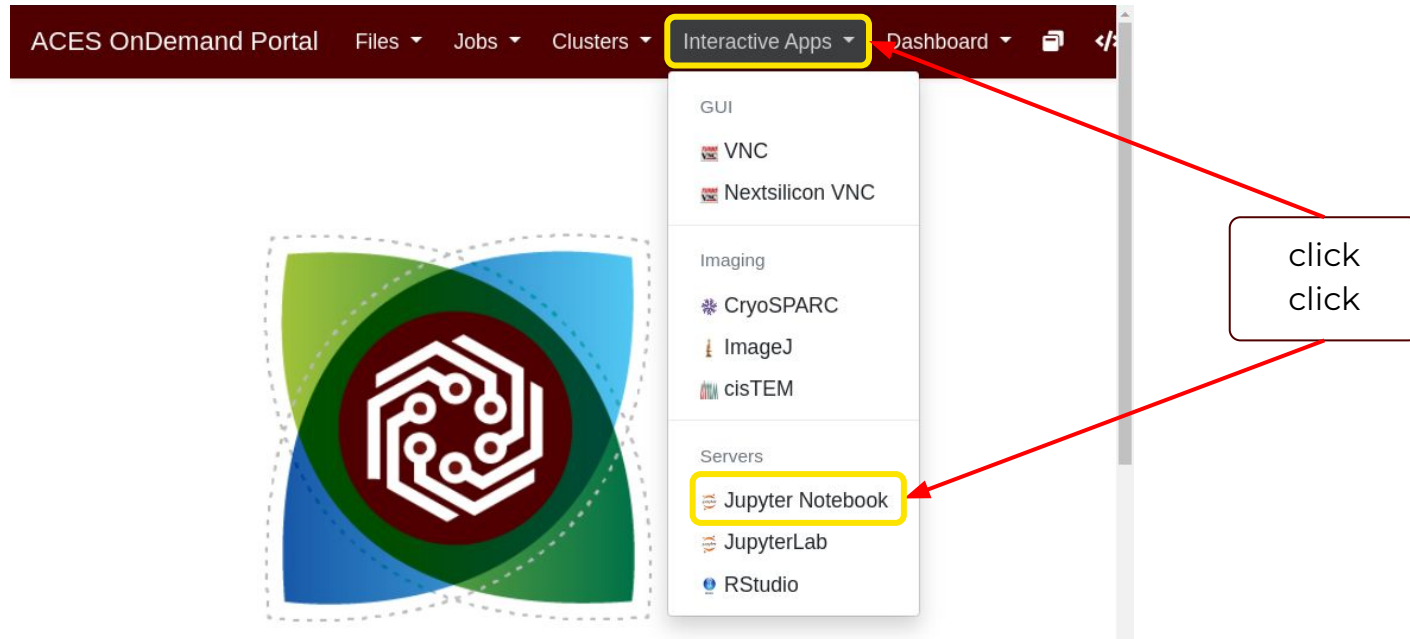
```
# execute the default runscrip defined in the container
singularity run hello-world_latest.sif
```

```
# execute a command within container
# specify the full path if the command is not in PATH
singularity exec scipy-notebook_latest.sif python3 hello.py
```

ONE VARIABLE

2 CONTAINERS

# Interactive Graphical Computing





# Containerized Jupyter Notebook

Choose *Containers*

Enter

`$SCRATCH/s_tutorial/scipy-notebook_latest.sif`  
or wherever your file actually is (see [Slide 27](#))

Backup copy at  
`/scratch/training/singularity/scipy-notebook_2023.sif`

The screenshot shows a web interface for launching a Jupyter Notebook. The breadcrumb navigation at the top reads 'Home / My Interactive Sessions / Jupyter Notebook'. On the left, a sidebar titled 'Interactive Apps' lists various options: GUI, VNC, Nextsilicon VNC, Imaging, CryoSPARC, ImageJ, cisTEM, Servers, and Jupyter Notebook. Two red arrows point from this sidebar to the main content area: one from 'Nextsilicon VNC' to the 'Type of Environment' dropdown, and another from 'ImageJ' to the 'Path to container image file' input field. The 'Type of Environment' dropdown is set to 'Containers (Singularity/Charliecloud)'. The 'Path to container image file' input field contains the path '\$SCRATCH/s\_tutorial/scipy-notebook\_latest.sif'. The main content area on the right is titled 'Jupyter Notebook' and contains instructions: 'This app will launch a Jupyter Notebook server on the ACES cluster.', 'Type of Environment' (with the selected dropdown), 'Select the type of environment in which Jupyter is installed. Help me choose', 'Path to container image file' (with the entered path), 'Enter the full path to an image file. Recommended that this be located in your \$SCRATCH directory.', and 'Singularity images and Charliecloud images are supported. Images should containing the Jupyter app.'

# ...Continued

The image illustrates the process of launching and using a Jupyter Notebook. It features several key elements:

- Launch Button:** A blue button labeled "Launch" at the top left.
- Notebook List:** A table showing two Jupyter Notebooks:

Notebook Name	Nodes	Cores	Status
Jupyter Notebook (5488)	1 node	1 core	Starting
Jupyter Notebook (5489)	1 node	1 core	Running
- Host Information:** Host: >\_ac110, Created at: 2023-09-21 15:39:52 CDT, Time Remaining: 56 minutes, Session ID: a5f41dfd-7c0d-44e3-aea7-7331c66a4d24.
- Connect to Jupyter Button:** A blue button labeled "Connect to Jupyter" at the bottom left.
- New Menu:** A dropdown menu with options: Notebook, Terminal, Console, New File, New Folder.
- Jupyter Notebook Interface:** A screenshot of the Jupyter Notebook interface showing the menu bar (File, Edit, View, Run, Kernel, Settings, Help) and the code editor with the following code:

```
[1]:  
import numpy  
print(numpy)
```

The output shows the module path for numpy: <module 'numpy' from '/opt/conda/lib/python3.11/site-packages/numpy/\_\_init\_\_.py'>
- WOW Starburst:** A red starburst graphic with the word "WOW" inside, indicating a successful or impressive result.

Red arrows indicate the workflow: from the "Launch" button to the "Connect to Jupyter" button, and from the "Connect to Jupyter" button to the "New" menu.

click  
...wait  
click  
...wait  
click

# Containerized Scientific Applications

# Singularity with GPU

- Containers should be built with CUDA version compatible with local GPUs (CUDA  $\geq$  11)
- Just add the `--nv` flag to your singularity command

Many repositories on Docker Hub have GPU-ready images. Search for images with “gpu” in tags

The nvidia cloud also provides GPU-ready images. See:  
[https://hprc.tamu.edu/wiki/SW:Singularity:Examples#NVIDIA\\_GPU\\_Cloud](https://hprc.tamu.edu/wiki/SW:Singularity:Examples#NVIDIA_GPU_Cloud)

# NVIDIA Container Registry Example

The screenshot displays the NVIDIA NGC Catalog interface for the PyTorch container. The breadcrumb trail is 'Catalog > Containers > PyTorch'. The page title is 'PyTorch'. On the left, there's a section with the PyTorch logo and 'Accelerated with NVIDIA'. The main content area has tabs for 'Overview', 'Tags', and 'Layers'. The 'Overview' tab is active, showing a description of PyTorch as an optimized tensor library. A yellow box highlights the '23.09-py3' tag, the instruction 'Copy the latest tag's image path below:', and a text box containing the image path 'nvcr.io/nvidia/pytorch:23.09-py3'. To the right of this box are buttons for 'Get Container' and 'Deploy to Vertex AI'. A red box with the text 'warning: do not attempt' is overlaid on the left side. At the bottom, a black box contains the command 'singularity pull docker://nvcr.io/nvidia/pytorch:23.09-py3'.

**warning: do not attempt**

```
singularity pull docker://nvcr.io/nvidia/pytorch:23.09-py3
```

# PyTorch GPU Exercise

Image file: `pytorch_23.09-py3.sif`

*from* `docker://nvcr.io/nvidia/pytorch:23.09-py3`

*Located at* `/scratch/training/singularity/`

From the login node: (all on one line)

```
srun --mem=4G --time=60
```

```
--gres=gpu:1 --partition=gpu --pty bash -i
```

following along live? add:  
`--reservation=containers`

From the compute node: (all on one line)

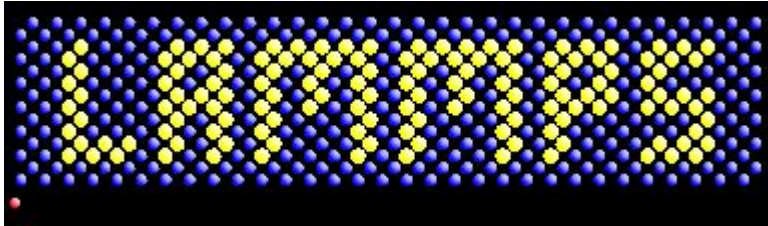
```
singularity exec --nv pytorch_23.09-py3.sif
```

```
python3 -c "import torch;
```

```
print(torch.cuda.device_count())"
```

# LAMMPS Molecular Dynamics on GPUs

- LAMMPS is a classical MD code
- <https://www.lammps.org/> has a cool animated logo.
- NVIDIA provides GPU-ready container images for lammps.  
<https://catalog.ngc.nvidia.com/orgs/hpc/containers/lammps>



# LAMMPS on H100 GPUs

- *This specific build works with H100 GPUs*

The screenshot shows the NVIDIA NGC Catalog interface for the LAMMPS container. The browser address bar displays `catalog.ngc.nvidia.com/orgs/hpc/containers/lammps/tags`. The page header includes the NVIDIA NGC | CATALOG logo and a 'Welcome Guest' message. The left sidebar shows the navigation path: Catalog > Containers > LAMMPS. The main content area has tabs for Overview, Tags (selected), Layers, Security Scanning, and Related Collections. A search bar for tags is present. A table of tags is displayed, with the 'patch\_15Jun2023' tag highlighted by a yellow border. The tag details include a cube icon, the tag name, a timestamp of 08/09/2023 11:34 AM, a size of 561.38 MB, and 2 Architectures. The tag's full name, `nvcr.io/hpc/lammps:patch_15Jun2023`, is shown in a dropdown menu.

Tag Name	Created	Size	Architectures	Full Name
patch_15Jun2023	08/09/2023 11:34 AM	561.38 MB	2 Architectures	nvcr.io/hpc/lammps:patch_15Jun2023



# LAMMPS on GPUs

Image file: `lammops-nv-patch_15Jun2023.sif`

*from* `docker://nvcr.io/hpc/lammops:patch_15Jun2023`

*Located at* `/scratch/training/singularity/`

From the login node: (all on one line)

`srun --mem=4G --time=60`

`--gres=gpu:1 --partition=gpu --pty bash -i`

following along live? add:  
`--reservation=containers`

From the compute node:

`cd /scratch/training/singularity`

(all on one line):

`singularity run --nv lammops-nv-patch_15Jun2023.sif  
bash benchmark.sh`

# Acknowledgements

This work was supported by

- the National Science Foundation (NSF), award numbers:
  - 2112356 - ACES - Accelerating Computing for Emerging Sciences
  - 1925764 - SWEETER - SouthWest Expertise in Expanding, Training, Education and Research
  - 2019129 - FASTER - Fostering Accelerated Scientific Transformations, Education, and Research
- Staff and students at Texas A&M High-Performance Research Computing.
- ACCESS CCEP pilot program, Tier-II



**HIGH PERFORMANCE  
RESEARCH COMPUTING**  
TEXAS A&M UNIVERSITY

<https://hprc.tamu.edu>

HPRC Helpdesk:

[help@hprc.tamu.edu](mailto:help@hprc.tamu.edu)

Phone: 979-845-0219

*Take our short course survey!*



HPRC Survey

[https://u.tamu.edu/hprc\\_shortcourse\\_survey](https://u.tamu.edu/hprc_shortcourse_survey)

Help us help you. Please include details in your request for support, such as, Cluster (ACES, FASTER, Grace, Launch), NetID (UserID), Job information (JobID(s), Location of your jobfile, input/output files, Application, Module(s) loaded, Error messages, etc), and Steps you have taken, so we can reproduce the problem.



High Performance Research Computing | [hprc.tamu.edu](https://hprc.tamu.edu)