Containers on the FASRC Clusters

Overview

- Software difficulties on HPC systems
- Why use containers?
- Containers overview
- Podman/Singularity containers
  - How to build your own containers
  - How to run containers on Cannon/FASSE
  - Bind mounts

Software difficulties on HPC systems

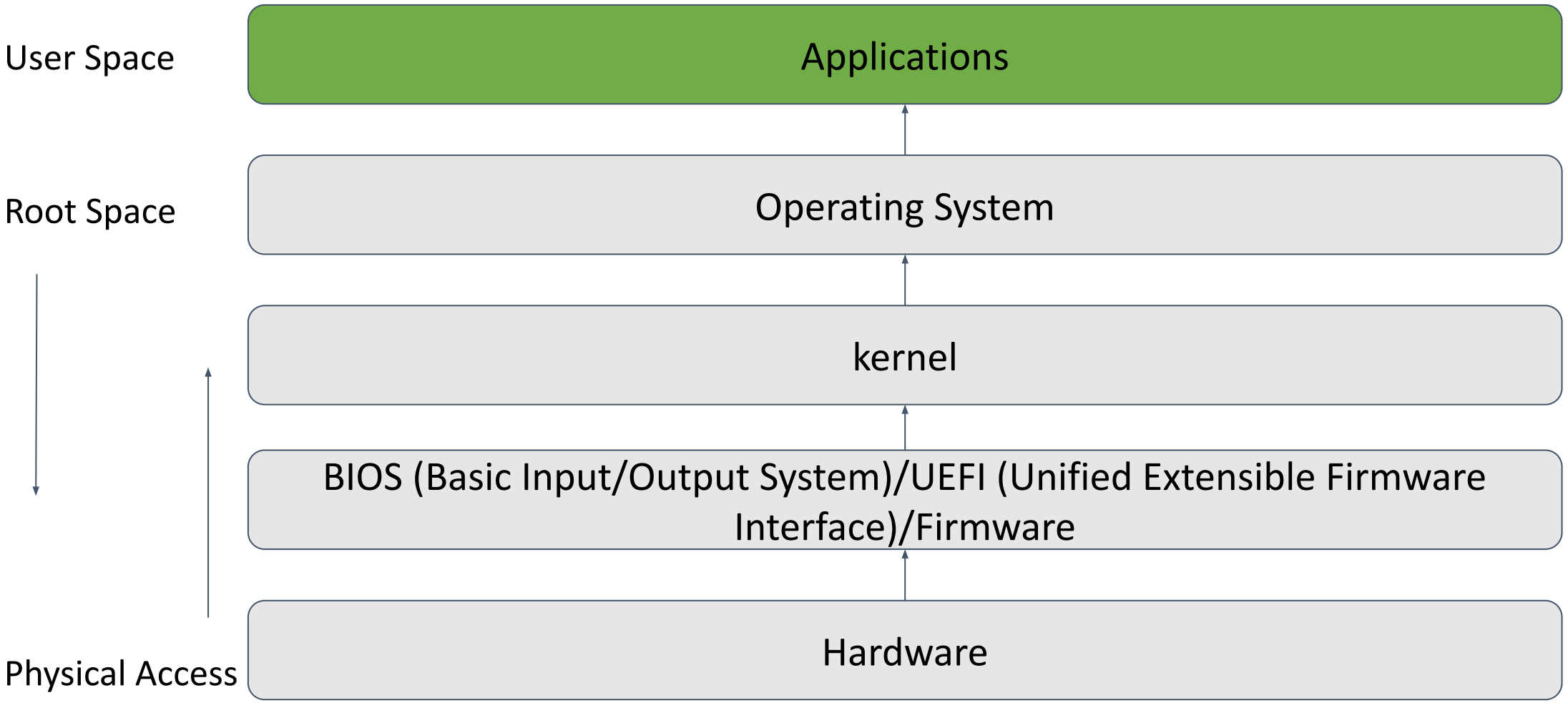
- Building software is often complicated, particularly on a shared and multi-tenant system
- Some applications might need dependencies that are not readily available and/or complex to build from source on a shared system
- Applications requiring software compatible with a different OS than what is offered on the cluster, e.g., Rocky Linux vs Ubuntu
- Reproducibility:
  - Different researchers may install different versions of an application and/or dependencies
- Portability & system-agnostic
  - Hard to share workflows & pipelines, with external collaborators, on another HPC system

Why use containers?

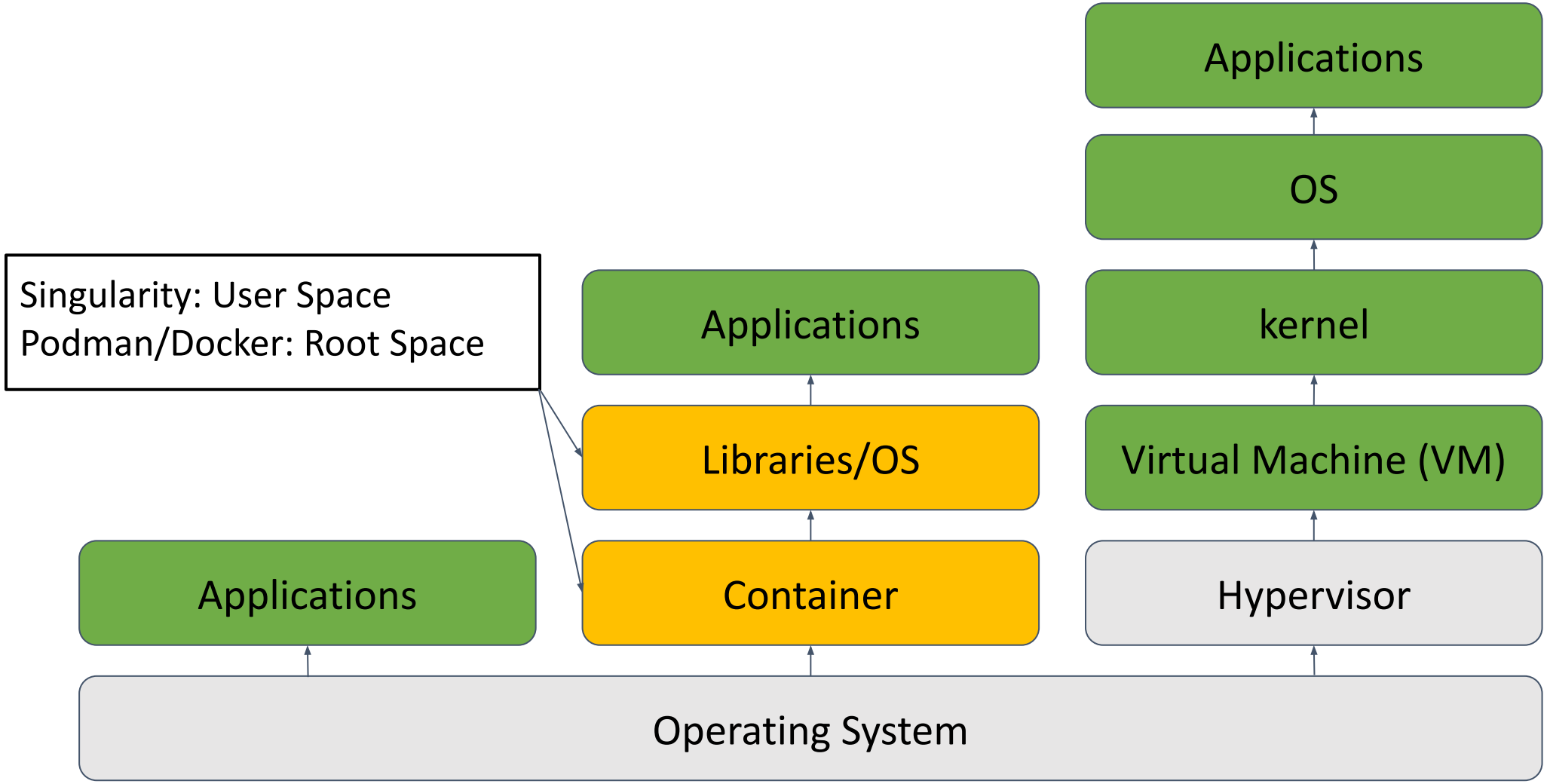
To overcome software stack, reproducibility and portability difficulties

- Create a virtual environment that contains the entire software stack needed
- Package in one place all necessary dependencies
- Choose a Linux operating system that is different than host (e.g. Ubuntu)
- Easy to publish
- Portable
- Reproducible

Software Stack



Virtualization Software Stack



Virtual machines (VMs) vs. Containers vs. Bare Metal

	Virtual Machines (VM)	Containers	Bare Metal
Software Flexibility	Very flexible -- for example, run Windows on MacOS	Less flexible Only Linux systems	Tied to OS libraries
Overhead	Heavyweight -- need to install all files of virtual environment	Very lightweight -- uses the kernel of host OS	Uses host OS
Portability	Can run on any hypervisor that supports VM type	Can run on any system that supports container type	Tied to specific OS

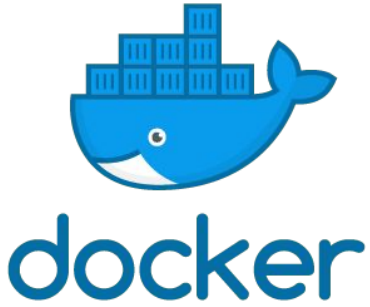
Adapted from LSU Singularity training slides:
http://www.hpc.lsu.edu/training/weekly-materials/2022-Fall/HPC_Singularity_Fall2022.pdf

Container nomenclature

- SingularityCE, Apptainer, Podman, Docker – the software that creates the container
  - As in “SingularityCE” or “Apptainer” or “Podman” or “Docker”
- Image
  - a compressed, usually read-only file that contains an OS and specific software stack
  - provides a template for a container
  - Examples: “Build a Matlab2021a image”, “Build an Alphafold image”, “Build an OpenFOAM image”
- Container
  - The technology: “containers vs. virtual machines”; is a running application
  - An instance of an image
    - Example: “process my data in a Singularity container of Matlab”; build an image & run a container using that
- Host – computer/supercomputer/laptop where the container is run

Containers

Cloud Native:



podman

HPC Native:

SingularityCE

Apptainer

Podman vs. SingularityCE

podman

- Assumes user has root (admin) privileges on the host system
- Not designed for HPC and multi-tenant systems

- Assumes user **does not** have root (admin) privileges on the host system
- Designed for HPC and multi-tenant systems

SingularityCE (Community Edition)

- Open-source container software: [SingularityCE | Sylabs]
- Specifically designed for HPC systems (i.e. multi-tenant systems)
  - No root (admin) privileges
- Package applications with their dependencies and workflow into a single file
- Singularity, SingularityCE, Apptainer
  - Singularity: deprecated since 2021
  - SingularityCE and Apptainer: branches/children of Singularity since 2021
  - SingularityCE: maintained by Sylabs since May 2021
  - Apptainer: Singularity open source project & maintained by the Linux Foundation since Nov 2021

Podman

podman

- Open Container Initiative (OCI) Toolchain developed by RedHat
- Daemonless
- Compatible with Docker or OCI compliant containers
  - `docker` commands run as `podman` and work in most cases
- Can be run in rootless mode using subuid

Subordinate User ID (subuid)

- subuid subordinates a uid range to a user
- permits spoofing another user in the container
- Filesystems
  - Does not work with Lustre
  - Need to permit world access

In the Container	Outside the Container
root (0)	jharvard (20000)
apache (48)	1020048
ubuntu (1000)	1021000

How to build Singularity/Podman images

- Start an interactive session on a compute node using `salloc`
- Select storage appropriate for the build method

Singularity

1. Download existing container
2. Download OCI container image from registry
3. Build from Singularity Definition File
4. Build from Dockerfile

Podman

1. Download OCI container from registry
2. Build from Containerfile/Dockerfile

Podman Image Storage

- Default path for image storage is `/tmp/containers-user-<uid>/storage`
  - `podman info` to learn more
  - Note that this storage is local to the node and transient

- Since image storage is on the local host we recommend that users either:
  - Use `podman save` to dump the image to disk (see later in this slide deck)
  - Upload any custom images they wish to keep to external registries like Docker Hub (see documentation for the registry)

Pull image from SingularityCE Container Library

SingularityCE library (https://cloud.sylabs.io/library)

```
# request interactive job
[jharvard@boslogin06 ~]$ salloc --partition test --time 01:30:00 -c 4 --mem 16G
[jharvard@holy8a24302 ~]$ mkdir -p ~/singularity
[jharvard@holy8a24302 ~]$ cd singularity/

# pull ubuntu container
[jharvard@holy8a24302 singularity]$ singularity pull library://library/default/ubuntu
INFO:    Downloading library image
28.4MiB / 28.4MiB [=============================================] 100 % 3.5 MiB/s 0s

[jharvard@holy8a24302 singularity]$ ls
ubuntu_latest.sif
```

Pull image from DockerHub

1. Go to DockerHub (https://hub.docker.com/) and search for a container. For example, alphafold
2. Click on tacc/alphafold
3. Click on the “Tags” tab
4. Select the version that you need. You will see something like

   ```
   docker pull tacc/alphafold:2.3.2
   ```
5. Singularity syntax to pull the image:

   ```
   singularity pull docker://<organization>/<repository>:<version>
   ```

   For tacc/alphafold, this becomes

   ```
   singularity pull docker://tacc/alphafold:2.3.2
   ```
6. Podman syntax to pull the image is the same as docker just replace it with podman (or just use the docker command directly)

Pull image from DockerHub (Singularity)

```
[jharvard@holy8a24302 singularity]$ singularity pull docker://tacc/alphafold:2.3.2
INFO:    Converting OCI blobs to SIF format
INFO:    Starting build...
INFO:    Fetching OCI image...
25.5MiB / 25.5MiB [=============================================] 100 % 28.6 MiB/s 0s
838.4MiB / 838.4MiB [===========================================] 100 % 28.6 MiB/s 0s
6.9MiB / 6.9MiB [===============================================] 100 % 28.6 MiB/s 0s
10.3MiB / 10.3MiB [=============================================] 100 % 28.6 MiB/s 0s
1.4GiB / 1.4GiB [===============================================] 100 % 28.6 MiB/s 0s
430.3KiB / 430.3KiB [===========================================] 100 % 28.6 MiB/s 0s
1.4GiB / 1.4GiB [===============================================] 100 % 28.6 MiB/s 0s
210.8MiB / 210.8MiB [===========================================] 100 % 28.6 MiB/s 0s
33.0MiB / 33.0MiB [=============================================] 100 % 28.6 MiB/s 0s
31.9MiB / 31.9MiB [=============================================] 100 % 28.6 MiB/s 0s
31.9MiB / 31.9MiB [=============================================] 100 % 28.6 MiB/s 0s
930.1MiB / 930.1MiB [===========================================] 100 % 28.6 MiB/s 0s
221.5MiB / 221.5MiB [===========================================] 100 % 28.6 MiB/s 0s
INFO:    Extracting OCI image...
INFO:    Inserting Singularity configuration...
INFO:    Creating SIF file...
```

Pull image from DockerHub (Podman)

```
[jharvard@holy8a24302]$ podman pull docker://tacc/alphafold:2.3.2
Emulate Docker CLI using podman. Create /etc/containers/nodocker to quiet msg.
✔ docker.io/tacc/alphafold:2.3.2
Trying to pull docker.io/tacc/alphafold:2.3.2...
Getting image source signatures
Copying blob e2250e4b9e64 done   |
Copying blob a404e5416296 done   |
Copying blob 391d07dd0009 done   |
Copying blob e479e09bf09a done   |
Copying blob 1ed7c42029c3 done   |
Copying blob 48c7a22e1966 done   |
Copying config 93d2c9a5a4 done   |
Writing manifest to image destination
93d2c9a5a4622fa4f3852c2fa2c88a28e1da49a47ff25630a1b3409d0b926b75
```

Build from Dockerfile

```
FROM ubuntu:22.04

RUN apt-get -y update \
 && apt-get -y install cowsay lolcat

ENV LC_ALL=C PATH=/usr/games:$PATH

ENTRYPOINT ["/bin/sh", "-c", "date | cowsay | lolcat"]
```

Build from Dockerfile (Singularity)

```
[jharvard@holy2c02302 ~]$ XDG_RUNTIME_DIR=$(mktemp -d) singularity build --oci lolcow.oci.sif Dockerfile
INFO:    singularity-buildkitd: running server on
/tmp/tmp.fCjzW2QnfV/singularity-buildkitd/singularity-buildkitd-3709445.sock
... omitted ouptput ...
INFO:    Terminating singularity-buildkitd (PID 3709477)
WARNING: removing singularity-buildkitd temporary directory /tmp/singularity-buildkitd-2716062861
INFO:    Build complete: lolcow.oci.sif
```

- Note that Singularity containers built with --oci need to be run with --oci

Build from Dockerfile (Podman)

Assuming the Dockerfile is in the same directory:

```
[jharvard@holy8a26602 ~]$ podman build -t localhost/lolcow
STEP 1/4: FROM ubuntu:22.04
Resolved "ubuntu" as an alias (/etc/containers/registries.conf.d/000-shortnames.conf)
Trying to pull docker.io/library/ubuntu:22.04...
Getting image source signatures
Copying blob 6414378b6477 done |
Copying config 97271d29cb done |
Writing manifest to image destination
STEP 2/4: RUN apt-get -y update && apt-get -y install cowsay lolcat

... omitted output ...

Running hooks in /etc/ca-certificates/update.d...
done.
--> a41765f5337a
STEP 3/4: ENV LC_ALL=C PATH=/usr/games:$PATH
--> e9eead916e20
STEP 4/4: ENTRYPOINT ["/bin/sh", "-c", "date | cowsay | lolcat"]
COMMIT
--> 51e919dd571f
51e919dd571f1c8a760ef54c746dcb190659bdd353cbdaa1d261ba8d50694d24
```

Save Image Podman

```
[jharvard@holy8a26602 ~]$ podman save --format oci-archive -o lolcow.tar
localhost/lolcow
[jharvard@holy8a26602 ~]$ ls -lh lolcow.tar
-rw-r--r--. 1 jharvard jharvard_lab 57M Jan 27 11:37 lolcow.tar
```

- OCI archive is recommended as it uses less storage.

Load Image Podman

```
[jharvard@holy8a26603 ~]$ podman images
REPOSITORY  TAG         IMAGE ID    CREATED     SIZE
[jharvard@holy8a26603 ~]$ podman load -i lolcow.tar
Getting image source signatures
Copying blob 163070f105c3 done   |
Copying blob f88085971e43 done   |
Copying config e9749e43bc done   |
Writing manifest to image destination
Loaded image: localhost/lolcow:latest
[jharvard@holy8a26603 ~]$ podman images
REPOSITORY        TAG         IMAGE ID      CREATED         SIZE
localhost/lolcow  latest      e9749e43bc74  6 minutes ago  172 MB
```

How to run images

Singularity syntax

```
singularity run [options: --oci] <container_image.sif>
```

Podman syntax

```
podman run [options] <image name>
```

How to get shell in the image

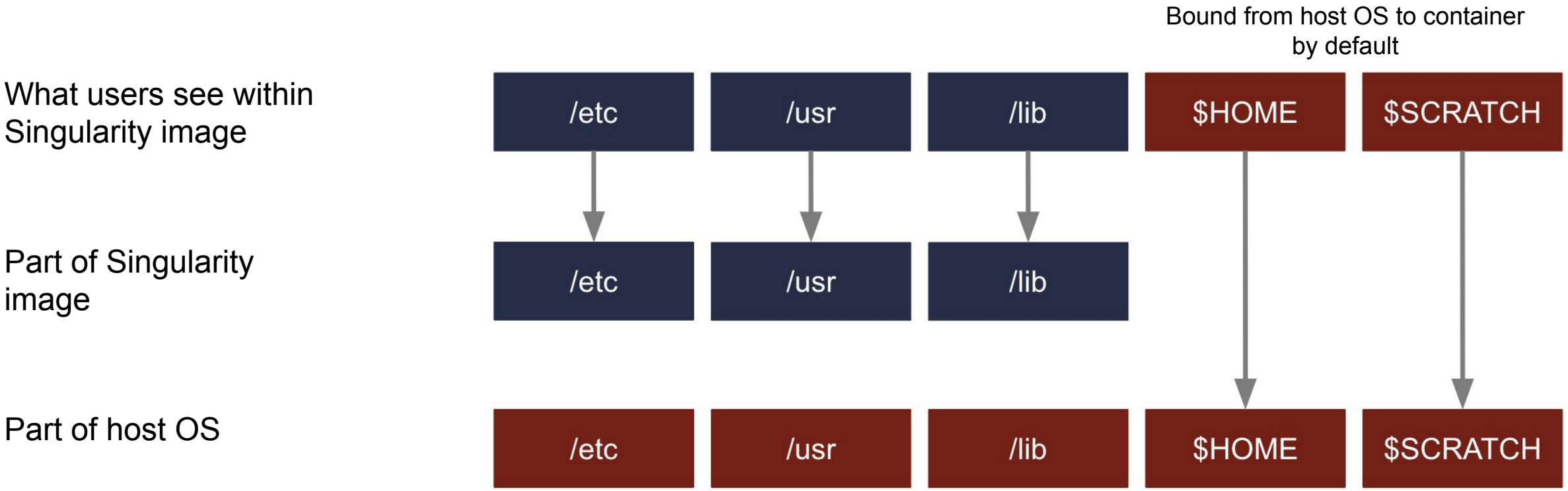
Singularity syntax

```
singularity shell [options: --oci] <container_image.sif>
```

Podman syntax

```
podman run –rm -it –entrypoint bash <image name>
```

Singularity and the host filesystem



To allow other filesystems to be accessible from container, use `--bind` option: `--bind src:dest`

- Note that Singularity by default binds in /scratch, /tmp, and /n

Podman and the host filesystem

- Podman only sees its own internal file structure by default.
- To bind in additional filesystems do:

  `--volume src:dest`

- Note that access to the filesystem paths will be limited by subuid
  - `--userns=keep-id` can be used to continue working in userspace similar to Singularity (i.e. the same uid/gid on the host is used in the container)

GPUs

Singularity

```
singularity run --nv <image name>
```

Podman

```
podman run --rm --device nvidia/gpu=all <image name>
```

Resources and help

- Documentation
  - https://docs.rc.fas.harvard.edu/
  - Containers: https://docs.rc.fas.harvard.edu/kb/containers/
  - Singularity: https://docs.rc.fas.harvard.edu/kb/singularity-on-the-cluster/
  - Podman: https://docs.rc.fas.harvard.edu/kb/podman/
- Email
  - rchelp@rc.fas.harvard.edu
- Office Hours
  - Wednesday noon-3pm https://harvard.zoom.us/j/255102481
- Training
  - https://www.rc.fas.harvard.edu/upcoming-training/

Training Session Evaluation

Please, fill out our training session evaluation. Your feedback is essential for us to improve our trainings!!

https://tinyurl.com/FASRC-training

Thank you!