Singularity Tutorial

> Carlos del-Castillo-Negrete
>
> May 21, 2024
>
> Kindly adapted for FASTER from ACES Container Workshop

Setting up Tutorial Environment

On Faster, you're going to want to create a folder on scratch to store everything

```
cd $SCRATCH
mkdir container_workshop
cd container_workshop
pwd
```

I've stored pre-built singularity files for this workshop at `/scratch/user/u.cd80202/singularity/`, in case you aren't able to pull containers on your own.

Setting Up Your Singularity Environment

Return to your tutorial directory (if necessary) and set your singularity cache directory for temporary files

```
cd $SCRATCH/s_tutorial
export SINGULARITY_CACHEDIR=$TMPDIR
```

Connect to the internet for fetching images

```
module load WebProxy
```

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 `--writable` flag?

```
singularity shell --writable 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-world_latest.sif
```

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
```

Our singularity image has a different Python 3.

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

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.

Docker vs. Singularity Commands

Action	Docker Command	Popular Docker Flags	Singularity Command	Popular Singularity Flags
Build Image	`docker build -t <image_name> .`	`-t` (tag), `-f` (Dockerfile)	`singularity build <image.sif> <recipe>`	`--sandbox`, `--writable`
Tag Image	`docker tag <image_id> <repository:tag>`	N/A	N/A	N/A

Notes:

- **Docker**: Primarily used for building, tagging, pushing, running, and managing containers.
- **Singularity**: Often used in HPC environments, supports running Docker images directly, and focuses on reproducibility and security.