Bioinformatics Skills for Microbial Genomics Workshop

2nd February 2022

Dr. Anna Price

Research Software Engineer, Cardiff University

The value and use of containers

CLIMB BIG DATA

Supporting material for this presentation

A copy of this presentation and supporting material can be found on the GitHub organization for this workshop:

https://github.com/Bioinfo-skills-2022-CLIMB-VM/The-value-and-use-of-containers

Overview of Container Engines

Container engines like Docker and Singularity can be used to deploy software packages in a lightweight, standalone, and reproducible environment

Software is packaged with its dependencies and is isolated from the host machine

Software runs uniformly regardless of infrastructure

OS-level virtualisation

Why should I use containers?

- Containerisation is an approach to software development in which the software is packaged with its dependencies, with an OS level of virtualisation
- It is also a way of isolating software packages from each other
- Biocontainers provides wide range of bioinformatics software ready to run
- Containers can be used with Nextflow and Snakemake to create fully reproducible workflows

Recipe → Image → Container

Useful Definitions

Dockerfile

- A Dockerfile is text file which provides instructions to Docker on how to build a Docker image.

Singularity recipe

- A Singularity recipe is text file which provides instructions to Singularity on how to build a Singularity image.

Images

- Images act as a snapshot of a container that Docker and Singularity can build upon. They contain the software and its dependencies.

Containers

- At runtime images become containers.

More Useful Definitions

Volumes

- Volumes are used to mount data by Docker so it can be accessed by a container.

Container Registries

- Cloud based services that are used to distribute images, e.g. Docker Hub, Singularity Hub, quay.io

Container Engine

- Creates and runs the container, e.g. Docker Engine

Docker Overview

- Works for all major Linux distributions
- Docker can run on Windows and Mac using Docker Desktop
- Docker daemon manages images, containers and volumes
- The Docker CLI communicates with this daemon
- Requires sudo privileges

Singularity Overview

- Singularity was developed with HPC systems in mind (Linux only)
- More effective security than Docker
- Easier to make use of GPUs & MPI
- Singularity mounts the $HOME directory
- SIF container image files were created for easier distribution

Differences between Docker and Singularity

- Docker can only work with Docker images. Whereas Singularity can create a Singularity image from a Docker image
- Docker requires the user to mount a volume when the container needs to access data on the local machine. Singularity mounts the home directory for you
- There are different user permissions between Docker and Singularity
- Only Docker has support for Mac and Windows

Which should I be using?

Both are valid choices. But Singularity does have advantages over Docker.

Singularity makes things easier for beginners by automatically mounting the home directory

Singularity is also more likely to be present on traditional HPC clusters

Snakemake provides support for Singularity but does not provide support for Docker. Nextflow can use either Docker or Singularity.

However, if you're interested in building and distributing your own images, then building Docker images does provide more flexibility as they can be used by both Docker and Singularity

Downloading Docker Images

- Search Docker Hub for images

```
docker search $searchterm
```

- Download an image from Docker Hub

```
docker pull $repository:tag
```

Downloading Singularity Images

Singularity images take the form of .sif files

Singularity can interact with many container registries:

- Singularity Container Library (Sylabs)
- Docker Hub
- Singularity Hub (no longer being updated)
- Quay.io

- Pulling an image from the Singularity Container Library

```
singularity pull library://$repository:tag
```

- Pulling an image from Docker Hub

```
singularity pull docker://$repository:tag
```

What sort of images are available?

- Biocontainers is an excellent resource for bioinformatics software
- There are container images for programming languages such as Python and R
- There are also container images for different Linux OS’s, e.g. Ubuntu, Debian, Alpine
- Conda images are also available

Docker: Working with images and containers

List all downloaded images

```
docker images
```

Run an image, while attaching a volume

```
docker run -v /absolute/path/to/files:/path/in/container --rm $imagename
```

Run an image with an interactive terminal

```
docker run -it $imagename bash
```

Docker: Working with images and containers

List all containers and their status

```
docker ps -a
```

Stop and delete all containers

```
docker stop $(docker ps -aq)
docker container prune
```

Singularity: Working with Images and Containers

To run commands within the container from the host

```
singularity exec $image.sif $command
```

To execute the runscript of a container

```
singularity run $image.sif
```

Docker: Working with Volumes

We can attach a host volume to any directory in the container using the -v flag. Note that when attaching a volume you must provide Docker with the absolute (full) path

```
docker run -v /path/to/volume/on/host:/path/to/directory/in/container $imagename
```

Dockerfiles and Singularity Recipes

Dockerfiles and Singularity Recipes are the instructions used to build container images

```
FROM debian:buster

ENV kraken2_version=2.0.8

ENV PACKAGES="gcc g++ wget parallel build-essential rsync
unzip ncbi-blast+"

RUN apt-get update && apt-get install -y $PACKAGES \
&& wget
https://github.com/DerrickWood/kraken2/archive/v${kraken2_
version}-beta.tar.gz \
&& tar -xzf v${kraken2_version}-beta.tar.gz \
&& rm v${kraken2_version}-beta.tar.gz \
&& cd kraken2-${kraken2_version}-beta \
&& ./install_kraken2.sh /usr/local/bin

CMD ["/bin/bash"]
```

Dockerfile

```
Bootstrap: docker
From: debian:buster

%environment
export kraken2_version=2.0.8
export PACKAGES="gcc g++ wget parallel build-essential rsync
unzip ncbi-blast+"

apt-get update && apt-get install -y $PACKAGES \
&& wget
https://github.com/DerrickWood/kraken2/archive/v${kraken2_
version}-beta.tar.gz \
&& tar -xzf v${kraken2_version}-beta.tar.gz \
&& rm v${kraken2_version}-beta.tar.gz \
&& cd kraken2-${kraken2_version}-beta \
&& ./install_kraken2.sh /usr/local/bin

%runscript
exec /bin/bash "$@"
```

Singularity Recipe

Best Practice when writing Container Recipes

- Consider what you want to use as a base image; base images such as Alpine can keep the image size down, but may not contain all the tools you need
- Use ENV variables to set the software version, this will make updating the image for new versions of software much easier!
- Each Docker/Singularity command creates a layer; chain together commands using \ &&

Useful Tools: Singularity Python (spython)

- Singularity python is Python API that works with Singularity
- It contains a useful feature that allows you to convert between Dockerfiles and Singularity recipes

Install

```
pip3 install  spython
```

Convert a Dockerfile to a Singularity recipe

```
spython recipe Dockerfile Singularity
```

Nextflow and Containers

- Nextflow can use both Docker and Singularity
- No need for the user to mount volumes. Just need to specify the container image to use and Nextflow does the rest
- Can define one container for entire workflow **or** define containers at a process level
- Can build profiles for docker, singularity (and conda) in the nextflow.config file

Snakemake and Containers

Snakemake can use Singularity

Different ways of managing Singularity containerisation with Snakemake:

- One container for entire workflow **or** define containers at a rule level
- Use existing container images **or** use ad-hoc combination of Conda package management with containers