Containers in HPC: Docker, Singularity, Apptainer

Grigory Shamov

HPC Workshop , Autumn 2023

Nov 7, 2023

University of Manitoba

What are containers and why they are popular

Containers are supposed to make software dependencies management easier.

Containers are Linux-specific tool of software isolation

- “Chroot” + “Linux namespaces” + runtime to run things + tools to manage things
- Shares kernel with the host Linux system: very little overhead
- Shares kernel with host, unlike Virtual Machines : bad security

The Earliest and most popular container environment   for long time was “**Docker**”.

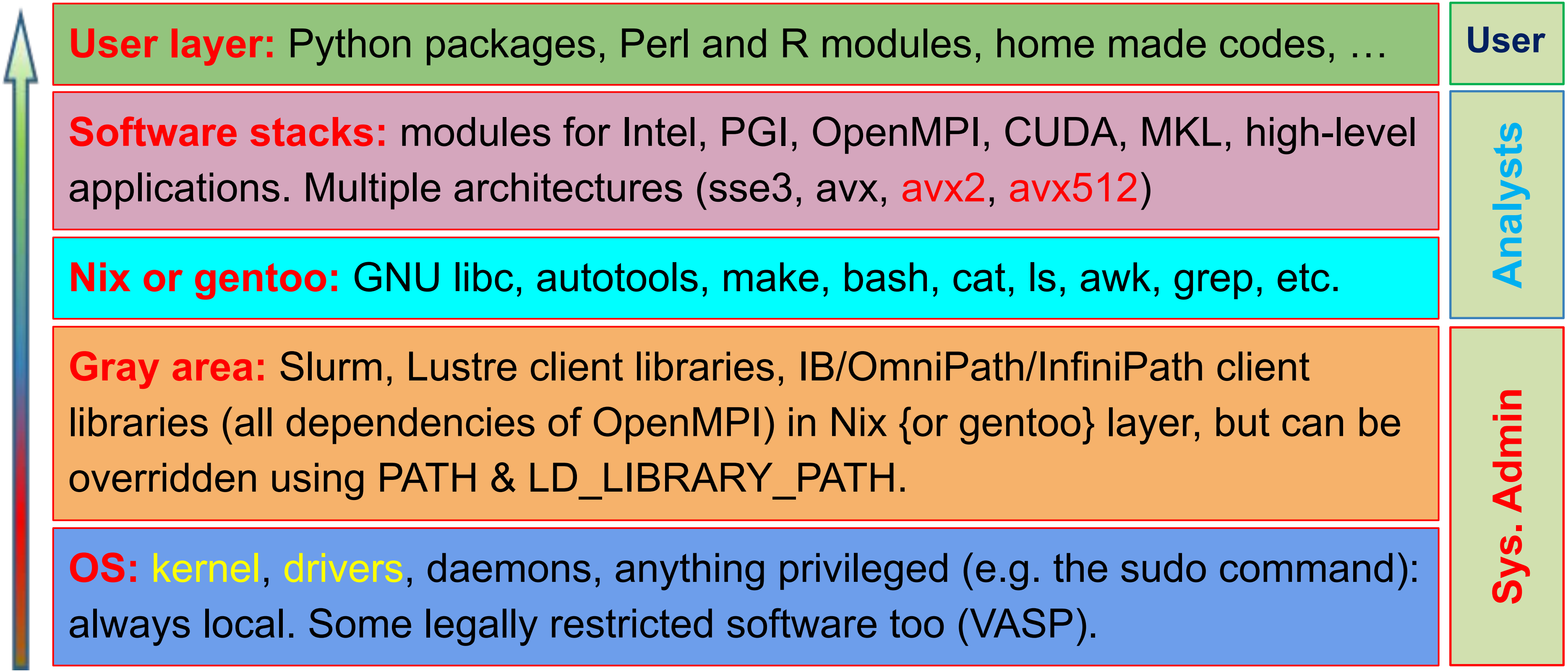
- Q: Can I have my Docker in your HPC environment?

Another popular environment developed for HPC environments is Singularity.

- The project since forked :
- **SingularityCE** by Sylabs and **Apptainer** by the Linux Foundation

University of Manitoba

Software layers *(slide by Dr. Ali Kerrache)*



User layer: Python packages, Perl and R modules, home made codes, …

Software stacks: modules for Intel, PGI, OpenMPI, CUDA, MKL, high-level applications. Multiple architectures (sse3, avx, avx2, avx512)

Nix or gentoo: GNU libc, autotools, make, bash, cat, ls, awk, grep, etc.

Gray area: Slurm, Lustre client libraries, IB/OmniPath/InfiniPath client libraries (all dependencies of OpenMPI) in Nix {or gentoo} layer, but can be overridden using PATH & LD_LIBRARY_PATH.

OS: kernel, drivers, daemons, anything privileged (e.g. the sudo command): always local. Some legally restricted software too (VASP).

User

Analysts

Sys. Admin

Popular container systems

#1 CE is Docker, which provides:

- Container tools and runtime that uses cgroups to manage resources
  - Assumers super-user access to the system
  - Runs as “root” inside container, may change privileges/users inse
- Container “recipes” to make new containers
- “Images” that are made of overlaid “layers”
  - Now standardized as the OCI format https://opencontainers.org/
  - Very convenient, economic images; each RUN makes a new layer.
- Container Registry that has ready images to download
  - Very successful DockerHub registry: https://hub.docker.com/
  - Recently got restricted, does limit bandwidth and number of downloads

- Podman, Flatpack, Snap, Bubblewrap, .. : many other Container engines around!

Popular use cases

#1 is Docker, which provides:

- Originally developed for running “services” in Docker. (a Web server, a DB)
- Several tools got developed to run many “microservices”, autoscale etc
  - Google Kubernetes ; eventually ceased to depend on Docker runtime
  - DockerSwarm, etc.

- In Research Computing, Docker become popular for software distribution.
  - For software development, building software in fixed environments
  - For reproducibility, sharing container images or container recipes
  - For dealing with bleeding edge software that has bad/changing dependencies
    - AI/ML, Genomics like containers a lot
  - For running software in container-native environments! K8s, etc.

So, can I just use Docker in HPC environment?

- The question comes to “do I need root access”.
  - On a shared system, it is not possible.
  - Also, HPC does resource management with SLURM while Docker does its own. These are hard to coordinate.
- Singularity was developed to run as a user, and as a regular process.
  - Mostly geared towards batch computing (a job starts and ends)
  - Can be used on shared filesystems
- Can create container images from Docker images!
  - However, not every image will work
  - Docker overlays are writable, Singularity images are immutable
  - Docker container may change users, starts as root;
  - https://apptainer.org/docs/user/latest/docker_and_oci.html#differences-and-limitations-vs-docker

Singularity or Apptainer?

- Singularity was developed since 2017 by a company called Sylabs.
  - https://sylabs.io/
- Due to personal conflicts, the development got forked to HPC-NG
- Then, HPC-NG was taken as a Linux Foundation project Apptainer.
  - https://apptainer.org/
- Sylabs develops Singularity-CE and an Enterprise edition.

- Wikipedia has an interesting discussion
  - https://en.wikipedia.org/wiki/Talk:Singularity_(software)
- The teams work in different directions, but so far products are compatible
  - The Container SIF format
  - OverlayFS support, rootless features
  - Support of OCI container format

So, do I need “root” to use Singularity In HPC?

- Yes, in some cases it is still needed.
  - When building new containers
  - Inspecting container images

- Containers have a working copy of an entire Linux distribution, some parts of which are owned by root.
  - Thus to build a new container,one has to be root
  - Unless a ready image from Docker is usable
  - Unless a system and Singularity/Apptainer installation support fakeroot and namespaces
  - Unless you delegate build of the image to a remote build service

Using Singularity or Apptainer

- You will need the (a?) Singularity engine installed.
  - https://github.com/sylabs/singularity (sources, RPMS)
  - https://github.com/apptainer/apptainer ; also in EPEL
  - Needs root privileges to install

- On the Alliance Federation systems, Apptainer is installed as a module

```
$>  module load apptainer
```

- On Grex, Singularity-CE is installed as a module

```
$> module load singularity
```

- Then, “apptainer” or “singularity” will be in the PATH Lets run a first container?

```
$>  singularity help                                    (or apptainer help)
$>  singularity exec library://lolcow cowsay "Mooo"
$>  singularity run docker://godlovedc/lolcow     (this will work with apptainer)
```

Running vs Executing , Inside vs Outside

- A container image typically has more than one executable
- There may be well defined “Entrypoints” (Docker) or “Runscripts” (Singularity)

```
$> singularity run {container_image}
```

- Any command can be executed inside a container with “exec”

```
$> singularity exec  {container_image} {a_command}
```

- How to find what is there, and what container is about to do?

```
$> singularity inspect –runscript {container_image}
$> singularity shell {container_image}
$> singularity exec {container_image} bash
```

- Let’s explore the lolcow container images.

Binding directories into the container

- Singularity containers are immutable ; how do we let them access our data?
  - (mostly, –*writable-tmpfs* and –*overlay* features may work)
  - Docker used to have “volume” containers for data

- Because ran as unpriviliged user, Singularity containers are safe to use on HPC’s cluster filesystems, like /home/ or /project or /global/scratch or /local
  - –*bind* or *-B* options to bind host directory into container
    - –*bind /scratch:/workdir* binds /workdir in the container to */scratch*
    - –*bind /opt* binds /opt on host to /opt in the container
  - /home/$USER , /tmp , /proc, /sys, /dev mounts by default
  - GPU drivers mounts by default with –*nv* or –*roce*
  - –*containall* prevents default mounts if needed

- Let’s try to bind and contain directories using a image..

Getting containers (that is, container images)

- Q: do I still need “root” to make my own images?

- “Pulling” containers from existing registries’ URI does not need root
    - *docker://*, SylabsCloud *library://* , Singularity Hub (defunct) *shub://* , etc.
    - Local registries, if present; https://singularity-hpc.readthedocs.io/

      ```
      $> apptainer pull docker://alpine
      $> apptainer pull docker://quay.io/biocontainers/pandas

      $> singularity pull --arch amd64 library://hpc/default/psi4:1.3
      ```

- “Building” containers from Recipes (Definition files)
    - Generally requires “root”

      ```
      $> sudo singularity  build {container_image}.sif  Singularity
      ```

Using Sandbox containers from CVMFS

- An image is a chrooted and compressed directory tree; SquashFS
  - There can be “sandbox”, uncompressed directory tree containers
- Some organizations distribute their software via CERN VM Filesystem

  https://cernvm.cern.ch/fs/
  - OpenScienceGrid https://osg-htc.org/
- OSG distributes Singularity / Apptainer containers in sandbox format
  - A recipe can be deposited in OSG registry by OSG members
- If a HPC machine (like Cedar or Grex) does provide the OSG software:

  *$> ls /**cvmfs/singularity.opensciencegrid.org**/lammps*
- Lets run an Intel HPL benchmark from OSG

```
$> singularity shell /cvmfs/singularity.opensciencegrid.org/intel/oneapi-hpckit:latest
$> cd /opt/intel/oneapi/mkl/latest/benchmarks/linpack
$> ./runme_xeon64
```

Building new containers from recipes

- “Building” containers from Recipes (Definition files)
  - Generally requires “root”

    $> *sudo singularity build {container_image}.sif Singularity*

- By default builds a compressed image. –sandbox can make a sandbox image
- Has to start a container from some base Linux OS distribution
  - From a Docker image , from Sylabs library
  - From scratch using a package manager from a Linux distribution
    - Debootstrap
    - Yum / DNF
  - From an existing container or a sandbox.

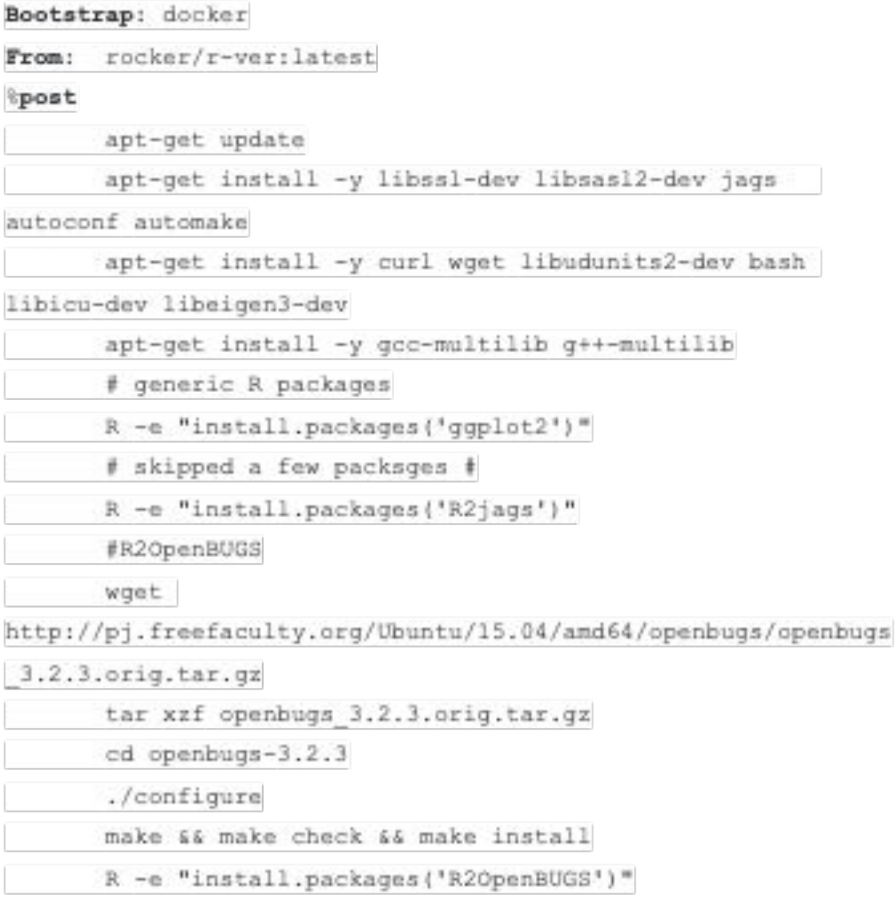
- Can run custom commands, installation scripts after the base Linux is installed
- Can set Environment variables , copy files, define entrypoints/runscripts

- Where to **Bootstrap** it **From** ?
- Modifies the container in **%post**

(can also:)

- copy **%files**
- Set the **%environment**
- Define entry point in **%runscript**
- etc.

https://apptainer.org/docs/user/latest/build_a_container.html



```
Bootstrap: docker
From:  rocker/r-ver:latest
%post
        apt-get update
        apt-get install -y libssl-dev libsasl2-dev jags
autoconf automake
        apt-get install -y curl wget libudunits2-dev bash
libicu-dev libeigen3-dev
        apt-get install -y gcc-multilib g++-multilib
        # generic R packages
        R -e "install.packages('ggplot2')"
        # skipped a few packsges #
        R -e "install.packages('R2jags')"
        #R2OpenBUGS
        wget
http://pj.freefaculty.org/Ubuntu/15.04/amd64/openbugs/openbugs
_3.2.3.orig.tar.gz
        tar xzf openbugs_3.2.3.orig.tar.gz
        cd openbugs-3.2.3
        ./configure
        make && make check && make install
        R -e "install.packages('R2OpenBUGS')"
```

Using remote builds in SIngularity

- Old SingularityHub by V. Sochat was very useful when it was
  - Would autobuild from recipes on a Github repository

- Sylabs Cloud provides “remote build” functionality
  - Works in SingularityCE, Apptainer has the functionality removed

- Needs an access key and a registration on Sylabs Cloud
  - Mind the I.P. rights there, if you share your recipe with the company!

```
$> singularity remote {command}   (list, login , etc.)
```

(need to initialize the remote build with the accesskey)

```
$> singularity build -r {container_image} Singularity.def
```

Demos and examples of use cases

- Using NVidia NGC container registry

```
$> salloc --partition=gpu --gpus=1 --cpus-per-gpu=6 --mem=12000

$> module load gcc/11.2 cuda/11.7 singularity

$> singularity pull docker://nvcr.io/hpc/lammps:patch_3Nov2022

$> wget https://lammps.sandia.gov/inputs/in.lj.txt

$> wget https://gitlab.com/NVHPC/ngc-examples/-/raw/master/lammps/single-node/run_lammps.sh

$> singularity run --nv -B $PWD:/host_pwd --pwd /host_pwd ./lammps_patch_3Nov2022.sif ./run_lammps.sh
```

- Using Singularity/Apptainer as part of larger workflow systems
  - Nextflow is one of them, for example this project:
  - https://github.com/Lcornet/GENERA/wiki/01.-Table-of-contents
  - https://github.com/Lcornet/GENERA/blob/main/Singularity/Genome-downloader.def

Demos and examples of use cases

- Using Singularity to encapsulate Conda (reduces number of files)
  - Conda is a chrooted environment that manages Python libraries
  - Also includes all the binary/OS dependencies, large number of small files

```
Bootstrap: docker
From: continuumio/miniconda:latest
%files
    # the file below must be present along the Singularity.def recipe
    environment.yml
%post
    ENV_NAME=mytest
    echo ". /opt/conda/etc/profile.d/conda.sh" >> $SINGULARITY_ENVIRONMENT
    echo "conda activate $ENV_NAME" >> $SINGULARITY_ENVIRONMENT
    . /opt/conda/etc/profile.d/conda.sh
    conda env create -f environment.yml -p /opt/conda/envs/$ENV_NAME
    conda clean --all
%runscript
    exec "$@"
```

environment.yml :

```
name:_my_env
channels:
  - defaults
dependencies:
  - numpy=1.18.1
  - pandas=1.0.1
  - scikit-learn=0.22.1
```

Is Apptainer/Singularity a silver bullet?

- Can “exec” software from well-built containers images
- Can convert suitably built Docker images
  - Making or finding a suitable container image is a bit of work
  - Bleeding-edge codes usually are poorly maintained and that includes their Docker images
- If software is already provided via Modules-based HPC software stack?
- Encapsulating software and sometimes data to reduce number of files
  - Conda is the prime example
  - OpenFOAM, certain GIS software could benefit from writable overlays

University of Manitoba