Workshop: Building and using containers on the HPC clusters

Yasasvy Nanyam
Robert Grandin

14 October 2021
Container Images and Samples

- All samples, test codes, container images and these presentation slides are available on hpc-class at

/ptmp/containers-workshop-october-2021
Outline

• Introduction to Containers
• Introduction to Singularity
• Singularity and HPC clusters
• Important Singularity commands
• Singularity and MPI
• Singularity recipes
• Demonstrate possible use cases
• Q&A, hands-on session
Introduction to Containers

Source: https://www.docker.com/resources/what-container
# Introduction to Containers

<table>
<thead>
<tr>
<th></th>
<th>DOCKER</th>
<th>SINGULARITY</th>
<th>PODMAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daemon</td>
<td>Daemon-based</td>
<td>Daemonless</td>
<td>Daemonless</td>
</tr>
<tr>
<td>Requires admin privileges</td>
<td>Requires admin privileges</td>
<td>No special privileges</td>
<td>No special privileges</td>
</tr>
<tr>
<td>Persistent services (web services, database)</td>
<td>Persistent services (web services, database)</td>
<td>User-space applications</td>
<td>Will likely replace docker when HPC transitions to RHEL8</td>
</tr>
</tbody>
</table>
Introduction to Singularity

• Compatible with most stand-alone OCI images (includes Docker)
• Can build containers on local machine and copy to cluster
• Devices and directories are also visible inside the container
  • accelerator cards, networks, work directories, etc.
• User outside = user inside (No contextual changes)
• Maintain your existing workflow
  • works with SLURM, MPI
• Little to no overhead
Calculate the number of prime numbers until $1024^2$

<table>
<thead>
<tr>
<th>Host</th>
<th>Container</th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of processes is 16</td>
<td>The number of processes is 16</td>
</tr>
<tr>
<td>N</td>
<td>Pi</td>
</tr>
<tr>
<td>1048576</td>
<td>82025</td>
</tr>
</tbody>
</table>

Host
[ynanyam@hpc-class05 openmpi-4.0.2]$ mpirun -np $SLURM_NTASKS ./prime_mpi

Container
[ynanyam@hpc-class05 openmpi-4.0.2]$ mpirun -np $SLURM_NTASKS \    singularity exec openmpi_4.0.2.sif ./prime_mpi
Singularity use cases

• **BYOE** – Bring Your Own Environment

• **Reproducibility** – version control, custom scripts and workflows

• Legacy programs that require specific environment
Limitations

- **Architecture** – limited by CPU architecture. The containers used in this workshop only work with x86_64 arch.
- **Portability** – requires glibc and kernel compatibility between host and container.
- **User space** – Not all containers available in docker hub are usable.
Important Singularity Commands

- **pull**  Get container images from repositories
- **exec**  Run command in the container
- **shell**  “Login to” the container for debugging
- **build**  Create container from recipe
Importantly, SINGULARITY_VARIABLES are crucial:

- **SINGULARITY_CACHEDIR**
- **SINGULARITY_TMPDIR**

Limited space in home directories. Set to `$TMPDIR` to avoid quota limits.

```bash
export SINGULARITY_CACHEDIR=$TMPDIR
export SINGULARITY_TMPDIR=$TMPDIR
```
Singularity pull

• Pull (download) container images from “hubs”
  • Docker - https://hub.docker.com/
  • Singularity - https://cloud.sylabs.io/library
  • Quay (Bioinformatics) - https://quay.io/search
  • Nvidia NGC - https://ngc.nvidia.com/catalog/containers

singularity pull <library>://<image>[::<tag>]

singularity pull docker://gcc:8.3.0

Note: Only use the compute nodes to pull containers.
Singularity pull

[ynanyam@hpc-class05 container-workshop]$ singularity pull docker://gcc:8.3.0
INFO:    Converting OCI blobs to SIF format
INFO:    Starting build...
Getting image source signatures
Copying blob 50e431f79093 done
Copying blob dd8c6d374ea5 done
----------------------------
Writing manifest to image destination
Storing signatures
2021/10/13 16:49:01  info unpack layer:
sha256:50e431f790939a2f924af65084cc9d39c3d3fb9ad2d57d183b7eadf86ea46992
INFO:    Creating SIF file...

[ynanyam@hpc-class05 container-workshop]$ ls
gcc_8.3.0.sif
Singularity exec

- Spawn a command within a container image
- Recommended way to use containers in HPC as it facilitates batch submissions and can be included as a part of your SLURM script.

`singularity exec [options] image.simg command [command-args]`
Singularity exec

• Useful options
  • --nv: Leverage GPUs
  • --bind: Bind mount directories to the containers
    • Note: /home is mounted by default. /work /ptmp and $TMPDIR need to mounted manually.
    • Note: SINGULARITY_BIND can also be used
  • --contain/--containall: Better isolate the container runtime from the host
  • --cleanenv: Clean the environment
  • --pwd: Initial working directory within the container

• Issue `singularity exec --help` to see all options
Singularity exec

```
[ynanyam@hpc-class05 ~]$ which gcc; gcc --version
/usr/bin/gcc

gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36)
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

[ynanyam@hpc-class05 ~]$ singularity exec gcc_8.3.0.sif gcc --version

gcc (GCC) 8.3.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```
Singularity exec

By default, only /home/$USER and /tmp is bind mounted within the container

```
[ynanyam@hpc-class05 container-workshop]$ singularity exec gcc_8.3.0.sif df -hT
Filesystem                                      Type      Size  Used  Avail  Use% Mounted on
overlay                                         overlay   16M    20K  16M   1% /
devtmpfs                                       devtmpfs   63G     0   63G    0% /dev
tmpfs                                           tmpfs    63G     0   63G    0% /dev/shm
/dev/mapper/rhel-rootvol                        xfs       20G    3.9G  16G  20% /tmp
hpc-class-stor02:/hpc-class-stor02/home/ynanyam nfs4   10T    1.2T  8.8T  12% /home/ynanyam
tmpfs                                           tmpfs    16M    20K  16M   1% /etc/group
```
Singularity exec

```bash
[ynanyam@hpc-class05 container-workshop]$ export SINGULARITY_BIND=/tmp,$TMPDIR

[ynanyam@hpc-class05 container-workshop]$ singularity exec gcc_8.3.0.sif df -hT

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Type</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>overlay</td>
<td>overlay</td>
<td>16M</td>
<td>20K</td>
<td>16M</td>
<td>1%</td>
<td>/</td>
</tr>
<tr>
<td>devtmpfs</td>
<td>devtmpfs</td>
<td>63G</td>
<td>0</td>
<td>63G</td>
<td>0%</td>
<td>/dev</td>
</tr>
<tr>
<td>tmpfs</td>
<td>tmpfs</td>
<td>63G</td>
<td>0</td>
<td>63G</td>
<td>0%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/mapper/rhel-rootvol</td>
<td>xfs</td>
<td>20G</td>
<td>3.9G</td>
<td>16G</td>
<td>20%</td>
<td>/tmp</td>
</tr>
<tr>
<td>hpc-class-stor02:/hpc-class-stor02/home/ynanyam</td>
<td>nfs4</td>
<td>10T</td>
<td>1.2T</td>
<td>8.8T</td>
<td>12%</td>
<td>/home/ynanyam</td>
</tr>
<tr>
<td>tmpfs</td>
<td>tmpfs</td>
<td>16M</td>
<td>20K</td>
<td>16M</td>
<td>1%</td>
<td>/etc/group</td>
</tr>
<tr>
<td>hpc-class-stor02:/hpc-class-stor02/ptmp</td>
<td>nfs4</td>
<td>47T</td>
<td>2.2T</td>
<td>45T</td>
<td>5%</td>
<td>/ptmp</td>
</tr>
<tr>
<td>/dev/mapper/rhel-local</td>
<td>xfs</td>
<td>2.5T</td>
<td>779M</td>
<td>2.5T</td>
<td>1%</td>
<td>/scratch/ynanyam/101051</td>
</tr>
</tbody>
</table>
```

Binding $TMPDIR is required for MPI jobs

```bash
[ynanyam@hpc-class05 container-workshop]$ singularity exec -B /tmp -B $TMPDIR gcc_8.3.0.sif df -hT
```
Singularity shell

• Interactively access the container image
• Similar to logging-in to a machine via SSH
• Useful for debugging during interactive sessions (e.g., salloc), not suitable for batch submissions
Singularity + MPI

- OPENMPI available both inside and on the host

```bash
[ynanyam@hpc-class05 openmpi-4.0.2]$ export SINGULARITY_BIND=/ptmp,$TMPDIR
[ynanyam@hpc-class05 openmpi-4.0.2]$ module load gcc/7.3.0-xegsmw4 openmpi

[ynanyam@hpc-class05 openmpi-4.0.2]$ mpirun -np $SLURM_NTASKS singularity exec openmpi_4.0.2.sif ./hello_mpi
P0: HELLO_MPI - Master process:
P0: C/MPI version
P0: An MPI example program.
P0: The number of processes is 16.
P0: 'Hello, world!'
P0: Elapsed wall clock time = 0.000029 seconds.
P1: 'Hello, world!'  
----------------------
P11: 'Hello, world!'
P12: 'Hello, world!'
P13: 'Hello, world!'
P14: 'Hello, world!'
P15: 'Hello, world!'
P0: HELLO_MPI - Master process:
P0: Normal end of execution: 'Goodbye, world!'
```
Singularity build

• Build on Singularity Cloud library. Requires a GitHub account.
  • Relatively slow, resource limits can require splitting container into “layers” and building piece-by-piece.
  • Great for publishing/distributing the final container

• Build locally. Requires administrator privileges on the build machine. (not possible on ISU HPC systems)
  • Often faster to iterate and debug the container-build process
  • If you don’t have admin privileges, ask for a VM to use

• Once added to Singularity Cloud, containers can be pulled by any machine where singularity is installed
Singularity recipe

- Builds upon other containers
- Utilize package managers to install software into container
  - apt, yum
  - spack

Bootstrap: docker
From: centos:7
%post

```bash
  echo "Installing Development Tools YUM group"
yum -y groupinstall "Development Tools"

  echo "Installing OpenMPI into container..."
  # Here we are at the base, /, of the container
  git clone https://github.com/open-mpi/ompi.git
  cd ompi

  # Now at /ompi
  git checkout cb5f4e7 #4.0.2

  ./autogen.pl
  ./configure --prefix=/usr/local

  make
  make install

  /usr/local/bin/mpicc examples/ring_c.c -o /usr/bin/mpi_ring
```

CentOS-based container with locally-built OpenMPI
**Singularity recipe**

- Builds upon other containers
- Utilize package managers to install software into container
  - `apt`, `yum`
  - `spack`

```
Bootstrap:shub
From:ResearchIT/spack-singularity:spack

%labels
MAINTAINER ynanyam@iastate.edu
APPLICATION trinity

%help
This container provides trinity

%environment
source /etc/profile.d/modules.sh
module load trinity

%post
export SPACK_ROOT=/opt/spack
export PATH=$SPACK_ROOT/bin:$PATH

yum -y install bc paste
yum clean all

export FORCE_UNSAFE_CONFIGURE=1
source $SPACK_ROOT/share/spack/setup-env.sh
spack install trinity

%runscript
exec Trinity "$@"
```

Containerized Trinity, built via spack
For more information...

- https://www.hpc.iastate.edu/guides/containers
- https://opencontainers.org/about/overview/
- https://sylabs.io/guides/latest/user-guide/
- https://spack.readthedocs.io
- https://cloud.sylabs.io
- https://hub.docker.com
- https://quay.io/search
- https://ngc.nvidia.com/catalog
- As always: hpc-help@iastate.edu
Hands-On

• Demonstrations
  • Getting started with Singularity Cloud
  • Using Singularity Cloud to build a container from a recipe
  • Building locally from a recipe
  • Using containers
    • Compiling with GCC 8.3
    • Running TensorFlow on a GPU
    • Running Hisat2

• Workshop, Q&A
Using Singularity Cloud to Build

- Create a GitHub account
- Go to [https://cloud.sylabs.io/home](https://cloud.sylabs.io/home) and click sign-in
Using Singularity Remote Builder

- Creates OpenMPI 4.0.2 container image
- ~15min to build on remote builder

```
Bootstrap: docker
From: centos:7

%post

echo "Installing Development Tools YUM group"
yum -y groupinstall "Development Tools"

echo "Installing OpenMPI into container..."
# Here we are at the base, /, of the container
git clone https://github.com/open-mpi/ompi.git
cd ompi

# Now at /ompi
git checkout cb5f4e7 #4.0.2

./autogen.pl
./configure --prefix=/usr/local

make
make install

/usr/local/bin/mpicc examples/ring_c.c -o /usr/bin/mpi_ring
```
Building Locally from a Recipe

Bootstrap: docker
From: makaho/hisat2-zstd

%labels
MAINTAINER rgrandin@iastate.edu
APPLICATION hisat2

%help
This container provides hisat2

%runscript
exec hisat2 "@"

{root@d5q4v2g2} # singularity build hisat2.sif hisat2.def

1.75 minutes

{root@d5q4v2g2} # ls -alh
total 319M
drwxr-xr-x.  2 root root  4 Mar 27 09:06 .
drwxr-xr-x. 10 rgrandin root 10 Mar 27 09:06 ..
-rwxr-xr-x.  1 root root 319M Mar 27 09:01 hisat2.sif
-rw-r--r--.  1 root root 170 Mar 27 08:59 hisat2.def
Demo: Compiling with GCC 4.8.5

```c
1  //#include <stdio.h>
2 3  int main()
4  { 
5      printf("Hello, world!\n");
6      return 0;
7  }
```

```bash
{rgrandin@hpc-class06} $ gcc hello.c -o hello
hello.c: In function ‘main’:  
hello.c:5:5: warning: incompatible implicit declaration of built-in function ‘printf’ [enabled by default]
   printf("Hello, world!\n");
  ^
```

Compilation using system gcc (v4.8.5)
Demo: Compiling with GCC 8.3.0

{rgrandin@hpc-class06} > singularity exec -B /tmp gcc-8.3.0.sif gcc hello.c -o hello
hello.c: In function 'main':
hello.c:5:5: warning: implicit declaration of function 'printf' [-Wimplicit-function-declaration]
   printf("Hello, world!\n");
^~~~~~~
hello.c:5:5: warning: incompatible implicit declaration of built-in function 'printf'
hello.c:5:5: note: include '<stdio.h>' or provide a declaration of 'printf'
hello.c:1:1:
+#include <stdio.h>
//#include <stdio.h>
hello.c:5:5:
   printf("Hello, world!\n");

Compilation using containerized gcc (v8.3.0)
import tensorflow as tf

# Create some tensors
a = tf.constant([[1.0, 2.0, 3.0], [4.0, 5.0, 6.0]])
b = tf.constant([[1.0, 2.0], [3.0, 4.0], [5.0, 6.0]])
c = tf.matmul(a, b)

print(c)
Running TensorFlow

[ynanyam@hpc-class05 tensorflow]$ singularity exec tensorflow_2.1.0-gpu.sif python tf-test.py
2021-10-13 18:44:04.580698: I tensorflow/stream_executor/platform/default/dso_loader.cc:53
Successfully opened dynamic library libcudart.so.11.0
2021-10-13 18:44:07.056091: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'libcuda.so.1'; dlerror: libcuda.so.1: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/lib64/.singularity.d/libs
2021-10-13 18:44:07.056175: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running on this host (hpc-class05): /proc/driver/nvidia/version does not exist
2021-10-13 18:44:07.056537: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: SSE3 SSE4.1 SSE4.2 AVX AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
[5 12 21 32]

Run the test script inside the container – NO GPU

Print() statement outputs

Print() statement outputs

Error that CUDA device is unavailable (container built with GPU expectation)
Running TensorFlow on GPU

$ singularity exec --nv /ptmp tensorflow_2.1.0-gpu.sif python tf-test.py

2021-10-13 21:09:01.089951: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1555] Found device 0 with properties:
pciBusID: 0000:82:00.0 name: Tesla K20m computeCapability: 3.5
coreClock: 0.7055GHz coreCount: 13 deviceMemorySize: 4.63GiB deviceMemoryBandwidth: 193.71GiB/s
2021-10-13 21:09:01.414724: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 200000500 Hz
2021-10-13 21:09:03.460912: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1096] Device interconnect
2021-10-13 21:09:03.461007: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1102]      0
2021-10-13 21:09:03.461026: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1115] 0:   N
2021-10-13 21:09:03.465466: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcublas.so.10
tf.Tensor([[22. 28.]
[49. 64.]], shape=(2, 2), dtype=)
hisat2

[ynanyam@hpc-class05 ]$ cp -r /ptmp/containers-workshop-october-2021/hisat2 .
[ynanyam@hpc-class05 ]$ cd hisat2
[ynanyam@hpc-class05 hisat2]$ mkdir HS_out
[ynanyam@hpc-class05 hisat2]$ module load parallel
[ynanyam@hpc-class05 hisat2]$ singularity exec -B /ptmp hisat2_2.2.1.sif hisat2-build Arabidopsis_thaliana.TAIR10.dna.chromosome.1.fa At_chr1
[ynanyam@hpc-class05 hisat2]$ parallel -j 4 "singularity exec -B /ptmp
hisat2_2.2.1.sif hisat2 -p 4 -x At_chr1 -1 {1} -2 {2} -S HS_out/{1/.}.sam >&
HS_out/{1/.}.log" ::: samples/*_1.* :::+ samples/*_2.*
[ynanyam@hpc-class05 hisat2]$ ls -altr HS_out/
total 16831
  drwxr-xr-x. 4 ynanyam domain users 14 Oct 13 20:12 ..
  -rw-r--r--. 1 ynanyam domain users 5406097 Oct 13 20:12 SRR4420295_1.fastq.sam
  -rw-r--r--. 1 ynanyam domain users 5521927 Oct 13 20:12 SRR4420296_1.fastq.sam
  -rw-r--r--. 1 ynanyam domain users 587 Oct 13 20:12 SRR4420295_1.fastq.log

SNIP
Q&A – Hands-on Session

• Questions?

• Try to run these examples yourself
  • Compute nodes: `salloc -N 1 -n 4 -t 15:00`
  • GPU nodes: `salloc -N 1 -n 4 -t 15:00 --gres gpu:1`

• Be considerate with resource requests. We have to share the cluster.