Introduction To Containers on HPC
2022
Contents
- What is a container
- Docker and Singularity
- Building / Using Containers
- GPUs
- MPI
- BioContainers
Why Use Containers?
- Verifiable reproducibility
- An immutable image
- Mobility of compute
- Limits user privileges
- Make use of GPU’s and high-speed networks
Docker vs Singularity

Docker containers are very popular particularly in bioinformatics
Docker runtime is not secure on shared clusters
Create the container using Docker and then containerize with Singularity
Note: Singularity has rebranded as Apptainer, but it is the same software
Docker vs Singularity

Singularity has better security and performance
Where it runs
How it works

Container Creation
- `sudo singularity create container.img`

Import and Bootstrap
- `sudo singularity import container.img docker://ubuntu`
- `sudo singularity bootstrap container.img ubuntu.def`

Interact and Modify
- `sudo singularity shell --writable container.img`

Container Execution
- `singularity run container.img`
- `singularity shell container.img`
- `singularity exec container.img ...`

USER ENDPOINT

SHARED COMPUTATIONAL RESOURCE
Singularity on HPC – Compute Node

laptop $ ssh netid@hpc.arizona.edu
netid@gatekeeper $ shell
(puma) netid@wentletrap $ elgato
(elgato) netid@wentletrap $ interactive –a PI
netid@cpu9 $ singularity help
netid@cpu9 $ singularity help build

https://ua-researchcomputing-hpc.github.io/Apptainer-Examples/
Singularity on HPC - Creating

$ singularity pull docker://godlovedc/lolcow
  pulls from docker registry
  less reproducible – image can change

$ singularity pull library://sylabsed/examples/lolcow
  pulls from a container library
  more reproducible

$ singularity build lolcow.sif docker://godlovedc/lolcow
  more options
  converts to latest format
  needs a name
Singularity on HPC - Creating

SingularityHub for building containers
- developed by Vanessa Sochat at Stanford’s Research Computing Group
- supported by Google

It’s gone now ..
Singularity on HPC - Creating

So now we use Sylabs – developer of Singularity

• Log into https://cloud.sylabs.io
• Generate an access token (API key)
• From an interactive session:
  • singularity remote login
  • singularity build --remote nersc.sif nersc.recipe
  • singularity run nersc.sif
Singularity on HPC - Running

$ singularity shell lolcow_latest.sif
Singularity>
Singularity> exit

$ singularity run lolcow_latest.sif
or
$ ./lolcow_latest.sif

$ singularity run library://sylabsed/examples/lolcow
INFO: Using cached image
#!/bin/bash
#SBATCH --job-name=lolcow
#SBATCH --ntasks=1
#SBATCH --nodes=1
#SBATCH --mem=1gb
#SBATCH --time=00:01:00
#SBATCH --partition=standard
#SBATCH --account=YOUR_GROUP

cd /path/to/container
singularity run lolcow_latest.sif
This example does not run cleanly, demonstrating that the container needs compatibility with the kernel on the compute node.

Copy the TFlow_example.py file from /contrib/singularity/nvidia

```
singularity exec docker://tensorflow/tensorflow python -m TFlow_example.py
INFO:  Converting OCI blobs to SIF format
INFO:  Starting build...
```

```
2022/02/25 15:32:10  info unpack layer: sha256:de37088510..
INFO:  Creating SIF file...
```

This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX2 FMA

step: 50, loss: 0.076930, W: 0.251121, b: 0.802448

... 

step: 1000, loss: 0.076929, W: 0.251473, b: 0.799952
Access to your files outside the container: Binding
You automatically get /home and /tmp and $PWD

$ echo "Hello from inside the container" > $HOME/hostfile.txt
$ singularity exec lolcow_latest.sif cat $HOME/hostfile.txt
Hello from inside the container

$ echo "Drink milk (and never eat hamburgers)." > data/cow_advice.txt
$ singularity exec --bind data:/mnt lolcow_latest.sif cat /mnt/cow_advice.txt
or implement these general descriptions:
$ singularity shell --bind /data my-container.sif
$ export SINGULARITY_BINDPATH=/data
GPUs and Containers

We show four ways to run tensorflow containers on HPC. We just use Tensorflow as an example:

1. Containers as modules: caffe, pytorch, rapids, tensorflow, and theano.
2. Tensorflow examples from our Github
3. Tensorflow directly from Nvidia
4. Tensorflow from Docker Hub
GPU and Containers

Singularity fully supports using GPUs with the –nv flag
A collection of containers with GPU support are available that were pulled from Nvidia’s service called NGC
“module whatis tensorflow/nvidia”

tensorflow/nvidia/2.0.0 : Name: Tensorflow
tensorflow/nvidia/2.0.0 : Version: nvidia-tensorflow_2.0.0-py3
tensorflow/nvidia/2.0.0 : Description: 'tensorflow' is an alias for 'singularity exec --nv /contrib/singularity/nvidia/nvidia-tensorflow_2.0.0-py3.sif python3'
tensorflow/nvidia/2.0.0 : Description: So typically 'tensorflow xxx.py'
GPUs and Containers

Singularity Using SLURM

```bash
#!/bin/bash
#SBATCH --job-name singularity-job
#SBATCH --account=your_pi
#SBATCH --partition=standard
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --gres=gpu:1
#SBATCH --time=01:00:00

cd /path/to/python/script
module load tensorflow/nvidia
tensorflow /contrib/singularity/nvidia/tensorflow_example.py
```
GPUs and Containers

Singularity Examples at our Github
https://ua-researchcomputing-hpc.github.io/Singularity-Examples/
Hello World  Docker  Remote Build  Tensorflow

Run this command from your test directory to get the examples:
wget https://ua-researchcomputing-hpc.github.io/Singularity-Examples/Tensorflow-Example/Tensorflow-Example.tar.gz
tar zxvf Tensorflow-Example.tar.gz
Register with Nvidia at https://ngc.nvidia.com/signin

From an interactive session on a compute node in a subdirectory of /HOME:
(literally $HOME refers to /home/uxx/netid which are directories starting u and your netid)
$ singularity build tensorflow-20.08-tf2-py3.simg docker://nvcr.io/nvidia/tensorflow:20.08-tf2-py3

From the same /HOME subdirectory
$ cp /contrib/singularity/nvidia/TFlow_example.py . # Note the “.” at the end
$ CONTAINER=$HOME/tensorflow-20.08-tf2-py3.simg #optional – set the path of container
$ singularity exec --nv $CONTAINER python TFlow_example.py
Go to Docker Hub and review Tags and copy Pull Command
https://hub.docker.com/r/nvidia/cuda
Create a recipe file, and modify Pull command:
Bootstrap: docker
FROM: nvidia/cuda:11.2.2-cudnn8-runtime-ubuntu18.04
Create container:
singularity build --remote tflow20.sif tflow20.recipe
Test:
$ singularity shell tflow20.sif
$ python3
>>> import tensorflow as tf
>>> tf.test.is_gpu_available()
... 
True
MPI and Containers

Singularity containers support MPI for multiple nodes
But the MPI type and version must match
The interconnect must match – infiniband or ethernet
Build the container with OpenMPI/3 or Intel/MPI
Using MPI reduces portability
sylabs.io/guides/ has a tutorial using MPI
BioContainers

A community driven project based on Docker focused on Proteomics and Genomics
A set of containers for the bioinformatics community
Specifications to build standardized containers
Guidelines for reproducible pipelines
Containers that can be pulled into Singularity for HPC use
### BioContainers on GitHub

#### Code

<table>
<thead>
<tr>
<th>Folder</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>abacas</td>
<td>Add debian package abacas/1.3.1-5-deb/Dockerfile</td>
<td>3 years ago</td>
</tr>
<tr>
<td>abyss</td>
<td>Add debian package abyss/2.1.5-7-deb/Dockerfile</td>
<td>3 years ago</td>
</tr>
<tr>
<td>acedb-other-belvu</td>
<td>Add debian package acedb-other-belvu/4.9.39dfsg.02-4-deb/Dockerfile</td>
<td>3 years ago</td>
</tr>
<tr>
<td>acedb-other-dotter</td>
<td>Add debian package acedb-other-dotter/4.9.39dfsg.02-4-deb/Dockerfile</td>
<td>3 years ago</td>
</tr>
<tr>
<td>adapterremoval</td>
<td>Add debian package adapterremoval/2.2.3-1-deb/Dockerfile</td>
<td>3 years ago</td>
</tr>
</tbody>
</table>
References

https://public.confluence.arizona.edu/display/UAHPC/Containers
https://ua-researchcomputing-hpc.github.io
https://hpc.nih.gov/apps/singularity.html
https://sylabs.io/guides/3.5/user-guide/introduction.html
https://github.com/sylabs/examples
https://www.tensorflow.org/tutorials/quickstart/beginner