Riviera Singularity Guide

1 Running a Basic Image

To get started with using singularity on Riviera to run docker images we will start with a very basic docker image that only executes a python file and builds on the latest ubuntu image. We can start by making a directory called BasicImage and adding these files to it.

Dockerfile

```
# A base image to build off of, in this case Ubuntu
FROM ubuntu:latest

# Sets the working directory inside of the container
WORKDIR /usr/src/app

# Copies application files from host to container
COPY . .

# Installs dependencies in the container
RUN apt-get update && apt-get install -y \
    python3 \
    python3-pip

# Set the default command to execute when the container starts
CMD ["python3", "test.py"]
```

test.py

```
def main():
    print("Hello, Docker!")

if __name__ == "__main__":
    main()
```

Now that we have a basic Dockerfile set up we can build the docker image with docker and then save it in a form that Singularity will be able to work with. To do that we need docker running on our local machines, Docker Desktop is the easiest way to have docker running locally.

```
$ cd BasicImage/
$ docker build -t basicimage .
$ docker save basicimage -o basicimage.tar
```

We now have everything we need to run this docker image with singularity on Riviera, so now all we need to do is transfer our image archive and python script over.

```
$ scp basicimage.tar test.py your_username@riviera.colostate.edu:~/your_directory
```

Now that it is transferred over we need to ssh into Riviera and we can run the container with singularity. When running with singularity we first build the image into a .sif file targeting the docker archive .tar file we just transferred over. We can then run the container making sure that the python file is in the current directory.

```
[your_username@login001]$ cd your_directory/
[your_username@login001]$ module load slurm
[your_username@login001]$ module load singularity
[your_username@login001]$ srun --pty bash
bash-4.4$ singularity build basicimage.sif docker-archive://basicimage.tar
WARNING: 'nodev' mount option set on /tmp, it could be a source of failure during build
↪  process
INFO:    Starting build...
INFO:    Fetching OCI image...
INFO:    Extracting OCI image...
INFO:    Inserting Singularity configuration...
INFO:    Creating SIF file...
INFO:    Build complete: basicimage.sif
bash-4.4$ singularity run basicimage.sif
Hello, Docker!
bash-4.4$ exit
[your_username@login001]$
```

2 PyTorch image

Here is an example of building a more advanced container for use with PyTorch. This image is notable because we do not specify a a command but instead an entrypoint. This allows us to change what python file we execute with the container.

Dockerfile

```
FROM nvidia/cuda:12.2.2-devel-ubuntu22.04

# Prevents interactive prompts
ENV DEBIAN_FRONTEND=noninteractive
# Install system dependencies
RUN apt-get update && \
    apt-get install -y \
```

```
        git \
        python3-pip \
        python3-dev \
        python3-opencv \
        libglib2.0-0

COPY . .

RUN python3 -m pip install -r requirements.txt
RUN python3 -m pip install --upgrade pip
RUN pip3 install torch torchvision torchaudio -f
↪  https://download.pytorch.org/whl/cu118/torch_stable.html

# Set the working directory
WORKDIR /app

# Set the entrypoint
ENTRYPOINT [ "python3" ]
```

Now that we have our dockerfile we can build our pytorch container and transfer it over to Riviera. Due to the size of the base nvidia container and the libraries being installed this will take some time. This can also all be sped up by carefully selecting the files that need to be installed and using a lighter weight image like the rocky linux one from nvidia for cuda.

```
$ cd PyTorch/
$ docker build -t pytorchimage .
$ docker save pytorchimage -o pytorchimage.tar
$ scp pytorchimage.tar your_username@riviera.colostate.edu:~/your_directory
```

Now that our container archive is transferred over we can build it with singularity. Due to the size of the container building it with singularity will overrun the default tmp directory on Riviera, so we will need to specify a new temp directory to use before building such as `~/tmp`.

```
[your_username@login001]$ mkdir your_temp_directory
[your_username@login001]$ export TMPDIR=~/your_temp_directory
[your_username@login001]$ cd your_directory/
[your_username@login001]$ module load slurm
[your_username@login001]$ module load singularity
[your_username@login001]$ srun --pty bash
bash-4.4$ singularity build pytorchimage.sif docker-archive://pytorchimage.tar
bash-4.4$ exit
[your_username@login001]$
```

We have now built our pytorch container with singularity. To run it we need to load our cuda modules and make sure that we tell singularity to use nvidia GPUs with the `--nv` flag. We also need to make sure to be in a partition with GPUs when we run this container with pytorch. We will also need to pass the container a python file to run.

```
[your_username@login001]$ module load cuda12.2/blas
[your_username@login001]$ module load cuda12.2/fft
[your_username@login001]$ module load cuda12.2/toolkit
[your_username@login001]$ srun --pty --partition=short-gpu bash
bash-4.4$ singularity run --nv pytorchimage.sif your_python_file.py
bash-4.4$ exit
[your_username@login001]$
```