Introduction to Containers

On KAUST Supercomputing Core Lab platforms

Dr. Samuel Kortas
samuel.kortas@kaust.edu.sa

Dr. Mohsin Ahmed Shaikh
mohsin.shaikh@kaust.edu.sa

Agenda

What makes a container	0930
Introduction to Singularity	1000
Working with pre-existing images	1010
Creating and building images	1025
Brake	1040
Launching Containers	1050
Writable images	1120
Demonstration of use cases	1125
Best Practices	1145
Q/A	1150

HPC containers

on KSL platforms

Singularity

- Created first at LBNL, now developed by a company (SyLabs) https://sylabs.io/
- Mobility of compute, reproducible research with Containers.
- Developed for HPC use case: runs as a regular user, can access shared filesystems.
- Interoperable with Docker; can run services as well.
- Supported on KSL HPC machines.

Accesing Singularity

- Singularity 3.5 is installed on Shaheen and Ibex

```
> module load singularity
```

Singularity CLI

```
> singularity --help
Options:
  -d, --debug     print debugging information (highest verbosity)
  -h, --help      help for singularity
      --nocolor   print without color output (default False)
  -q, --quiet     suppress normal output
  -s, --silent    only print errors
  -v, --verbose   print additional information
      --version   version for singularity
Available Commands:
  build       Build a Singularity image
  cache       Manage the local cache
  capability  Manage Linux capabilities for users and groups
  config      Manage various singularity configuration (root user only)
  delete      Deletes requested image from the library
  exec        Run a command within a container
  help        Help about any command
  inspect     Show metadata for an image
  instance    Manage containers running as services
  key         Manage OpenPGP keys
```

Singularity CLI

```
oci         Manage OCI containers
plugin      Manage Singularity plugins
pull        Pull an image from a URI
push        Upload image to the provided URI
remote      Manage singularity remote endpoints
run         Run the user-defined default command within a container
run-help    Show the user-defined help for an image
search      Search a Container Library for images
shell       Run a shell within a container
sif         siftool is a program for Singularity Image Format (SIF) file manipulation
sign        Attach a cryptographic signature to an image
test        Run the user-defined tests within a container
verify      Verify cryptographic signatures attached to an image
version     Show the version for Singularity
```

- All scripts and Dockerfiles mentioned during this training can be found in

https://github.com/kaust-rccl/singularity_workshop2020.git

You may clone the repository:

```
> git clone https://github.com/kaust-rccl/singularity_workshop2020.git
```

Working with container images

What is a container image

- A **standalone & immutable** (unchangeable) static file that contains
  - The source
  - All library dependencies required during runtime
  - Tools (e.g. code, runtime, system tools, system libs, user space libs)
  - Settings (e.g. environment variables, config files etc)
- Effectively, it’s a packaging format, with all required files as a single file, and some manifest(metadata) about its creation, content and execution instrcutions
- Different image formats are converging towards Open Container Initiative (OCI) Format Specification

What is a container image …

- Images become containers once launched with a Container runtime (e.g. Docker Engine, Singularity, Podman, Sarus etc.)
- Multiple containers from the same image can be lauched concurrently
- As naming convention goes, images have two parts:
  - Image name – name it anything you want, just like a file
  - Image tag – allows provenance, i.e. think of it as version control

Working with existing Docker images

- DockerHub – a public reposiroty of container images
  - Powerful way of sharing your good work with wider community
- Pull image from DockerHub using Singularity
  - Layers in the Docker image are all downloaded and converted in a *single* file called Singularity Image File or SIF
  - The reverse is not possible
- By convention pull command will name the SIF file as `IMAGENAME_TAG.sif`

```
> singularity pull docker://krccl/osu_openmpi403:563

INFO:    Converting OCI blobs to SIF format
INFO:    Starting build...
Getting image source signatures
Copying blob d7c3167c320d done
……
Copying blob 0b3d6ac94b60 skipped: already exists
Writing manifest to image destination
……
INFO:    Creating SIF file...
INFO:    Build complete: osu_openmpi403_563.sif
```

Working with existing Docker images

- Another way to get the images from DockerHub is to use singularity build command

```
> singularity build osu_openmpi403_563.sif docker://krccl/osu_openmpi403:563

INFO:    Starting build...
Getting image source signatures
Copying blob d7c3167c320d done
Copying blob 131f805ec7fd done
……
Copying blob 0b3d6ac94b60 skipped: already exists
Writing manifest to image destination
……
INFO:    Creating SIF file...
INFO:    Build complete: osu_openmpi403_563.sif
```

Pull images from Singularity cloud

```
> singularity pull library://shaima0d/ksl/osu_openmpi403:563

INFO:    Downloading library image
2.09 GiB / 2.09 GiB [============================================================] 100.00% 12.46 MiB/s 2m51s
WARNING: unable to verify container: osu_openmpi403_563.sif
WARNING: Skipping container verification
```

Inpsecting SIF files

- Singularity, similar to Docker, maintains informatio about the SIF file
  - Its more provenant if the image was built from Singularirty Definition File
- Useful for maintaining provenance
- Useful in debugging forward/backward compatibity of images

For an image built from Dockerfile using `docker build`

```
> singularity inspect osu_openmpi403_563.sif
WARNING: No SIF metadata partition, searching in container...
org.label-schema.build-date: Sunday_4_October_2020_15:59:16_+03
org.label-schema.schema-version: 1.0
org.label-schema.usage.singularity.deffile.bootstrap: docker
org.label-schema.usage.singularity.deffile.from: krccl/osu_openmpi403:563
org.label-schema.usage.singularity.version: 3.5.3
```

Inpsecting SIF files

For an image built from Singularity Definition file

```
> singularity inspect hpl.sif
WARNING: No SIF metadata partition, searching in container...
Author: mohsin.shaikh@kaust.edu.sa
Version: 2.3.0
org.label-schema.build-date: Monday_19_October_2020_19:16:13_+03
org.label-schema.schema-version: 1.0
org.label-schema.usage: /.singularity.d/runscript.help
org.label-schema.usage.singularity.deffile.bootstrap: docker
org.label-schema.usage.singularity.deffile.from: krccl/openmpi_base:403
org.label-schema.usage.singularity.runscript.help:
/.singularity.d/runscript.help
org.label-schema.usage.singularity.version: 3.5.3
```

Help on specific image

- In case of images built from Singualrity Definition File

```
> singularity run-help hpl.sif
```

```
This is a cross-platform HPL container built with OpenMPI. HPL or High
Performance Linpack is a known benchmark to test the capability of a
computer. For more information, please refer to
https://www.netlib.org/benchmark/hpl
```

Create own images

Two options to build images from scratch

- Docker route
  - Create a Docker image from a Dockerfile on your workstation
  - Upload on DockerHub in your private repository (free)
  - Pull using Singularity on Ibex/Shaheen

- Singularity route
  - Write a Singularity Definition file
  - Build a Singularity image on Ibex compute nodes using

```
singulairty --fakeroot
```

Building images – Docker route

Dockerfile

```
FROM krccl/openmpi_base:403


RUN apt-get install -y libatlas-base-dev gfortran

RUN mkdir /hpl

WORKDIR /hpl

RUN curl -o hpl-2.3.tar.gz -L http://www.netlib.org/benchmark/hpl/hpl-2.3.tar.gz

RUN tar xvf hpl-2.3.tar.gz -C /hpl --strip-component=1

ADD ./Make.ubuntu /hpl

RUN make arch=ubuntu

WORKDIR /hpl
```

Built on your local workstation/laptop/cloud/VM

```
> docker build -t hpl .
Step 1/9 : FROM krccl/openmpi_base:403
403: Pulling from krccl/openmpi_base
……
Step 2/9 : RUN apt-get install -y libatlas-base-dev gfortran
 ---> Running in 236d02d70687
……
Step 5/9 : RUN curl -o hpl-2.3.tar.gz -L http://www.netlib.org/benchmark/hpl/hpl-2.3.tar.gz
 ---> Running in a3b8f68fd704
……
Step 9/9 : WORKDIR /hpl
 ---> Running in fc6a978d1f79
Removing intermediate container fc6a978d1f79
 ---> 6603e1fcb7d2
Successfully built 6603e1fcb7d2
Successfully tagged hpl:latest
```

Building images – Docker route

- Next step is to push the image to DockerHub

```
> docker run -d -t --name hpl_cont hpl:latest
> docker commit -m "My first HPL container" hpl_cont mshaikh/hpl:230
> docker push mshaikh/hpl:230
```

Building images – Docker route

- This OCI image can now be pulled with Singularity platform on Shaheen/Ibex

```
> module load singularity
> cd $HOME/somewhere
> export SINGULARITY_TMPDIR=$HOME
> singularity pull docker://mshaikh/hpl:latest
```

Building images – Singularity route

- You can built an image in Singularity's native format from a ***Singualrity Definition File*** or `def` file
- Has two parts:
  - Header:

    Choices about base OS, e.g. Linux Distrinution, core packages networking etc & where to get them from i.e. repos,
  - Sections:

    A module description of step to perform while build an image, e.g. install dependeices, download source/data, build commands, set environment etc.

```
Bootstrap: docker
FROM: krccl/openmpi_base:403

%files
        ./Make.ubuntu /tmp

%post
        apt-get install -y libatlas-base-dev gfortran
        mkdir /hpl && cd /hpl
        curl -o hpl-2.3.tar.gz -L
        http://www.netlib.org/benchmark/hpl/hpl-2.3.tar.gz
        tar xvf hpl-2.3.tar.gz -C /hpl --strip-component=1
        cp /tmp/Make.ubuntu /hpl
        make arch=ubuntu VERBOSE=1
        cd /hpl
%environment
        export PATH=$PATH:/hpl/bin/ubuntu
        export HPL_HOME=/hpl
%labels
        Author mohsin.shaikh@kaust.edu.sa
        Version 1.0
%help
        This is a cross-platform HPL container …….
```

Building images – Singularity route

- Ordinarily `singularity build` requires priviliged user to run
  - Can’t do this on HPC cluster
- Singulairty allows `build` command invoccation in unprivileged mode
  - **Images can only be built on Ibex compute nodes**

```
> salloc --time=01:00:00 --ntasks=1
> export XDG_RUNTIME_DIR=$HOME/somewhere
> module load singularity
> srun singularity build --fakeroot ./hpl.sif ./hpl.def
```

Building images – running the image

- Resulting SIF file can be launched with Singularity runtime on compute nodes of Shaheen

Example Jobscript

```
#!/bin/bash
#SBATCH --ntasks=32
#SBATCH -t 01:00:00

module load openmpi/4.0.3
module load singularity

mpirun -n 32 singularity exec ./hpl_latest.sif /hpl/bin/ubuntu/xhpl
```

Singularity cache

- Singularity `pull` and `build` commands make use of cache in $HOME
  - keep the layers/blobs downloaded while building an image.
- The cached layers are reused whenever needed when building or pulling new images e.g.
  - ubuntu 18.04 may already exist because of a previously pulled image
  - Can check stats of cache

```
> singularity cache list
```

- Can run out of space, so clean cache

```
> singularity cache clean
```

Q/A and Interval

(10 minutes)

Launching containers

Using Singularity runtime to instantiate containers from images

Some facts ....

- A container runtime instantiates a container from an image
- Containers are emphemeral – stateless
- HPC filesystems are mounted by default
  - Shaheen -- $HOME,/project,/scratch,CWD
  - Ibex -- $HOME,/scratch,CWD
- **Containers is not writable by default**

Launching interactive container

- To discover container's content you can launch an interactive container
  - You may navigate in the filesystem
  - Investigate environment settings e.g. variable values, LD_LIBRARY_PATH, PATH etc

```
> singularity shell hpl.sif
Singularity> cat /etc/os-release
NAME="Ubuntu"
VERSION="18.04.4 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.4 LTS"
VERSION_ID="18.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=bionic
UBUNTU_CODENAME=bionic
```

`user` Management

- Within a singularity container, user is consistent as on the host:

```
> whoami
shaima0d
-bash-4.2$ id
uid=174988(shaima0d) gid=1174988(g-shaima0d) groups=1174988(g-shaima0d),1001(noor-
users),53001(ksl-k01),53033(ksl-k1028),53070(ksl-v1000),53071(ksl-s1001),53201(ksl-
k1191),53204(ksl-k1194),53280(ksl-ansys),53296(ksl-swtools),53414(ksl-
k1343),53499(ksl-k1411),53569(ksl-v1008),53588(ksl-k1488),56116(ibex-c2094)
```

`user` Management

- Within a singularity container, user is consistent as on the host:

```
> whoami
shaima0d
-bash-4.2$ id
uid=174988(shaima0d) gid=1174988(g-shaima0d) groups=1174988(g-shaima0d),1001(noor-
users),53001(ksl-k01),53033(ksl-k1028),53070(ksl-v1000),53071(ksl-s1001),53201(ksl-
k1191),53204(ksl-k1194),53280(ksl-ansys),53296(ksl-swtools),53414(ksl-
k1343),53499(ksl-k1411),53569(ksl-v1008),53588(ksl-k1488),56116(ibex-c2094)
```

```
> singularity shell hpl.sif
Singularity> whoami
shaima0d
Singularity> id
uid=174988(shaima0d) gid=1174988(g-shaima0d) groups=1174988(g-shaima0d),1001(noor-
users),53001(ksl-k01),53033(ksl-k1028),53070(ksl-v1000),53071(ksl-s1001),53201(ksl-
k1191),53204(ksl-k1194),53280(ksl-ansys),53296(ksl-swtools),53414(ksl-
k1343),53499(ksl-k1411),53569(ksl-v1008),53588(ksl-k1488),56116(ibex-c2094)
```

STDIN & STDOUT management

- Running container is just like another process
  - Can redirect input and output
  - Can pipe from/to a container
  - Can pipe from contianer to container

```
$ echo  “Hello” | xargs singularity exec ./alpine_latest.sif echo -
n > out.txt | singularity exec ./hpl_latest.sif echo " World" >>
out.txt
```

```
$ cat output.txt
Hello World
```

Execute container

- To execute a command inside a container

```
> singularity exec ./image.sif
./app -args
```

- To run a user-defined default command within a container

```
> singularity run IMAGE_NAME
```

Dockerfile

```
FROM alpine:latest

CMD ["echo","Hello World"]
```

Run command

```
$ singularity run ./helloworld_latest.sif

Hello World
```

Bind Mount

- Sometime required to map directories on host to make them available inside a container
- Can be achieved using
  - `--bind source:<dest>,source:<dest>`
  - setting `SINGULARITY_BIND=source:<dest>,source:<dest>`

- E.g. Filesystems on node local SSDs on Ibex are not mounted by default

```
> singularity exec --bind /local/reference ./tensorflow2.sif python train.py
```

Launching services (background processes)

- Some applications e.g. databases or proxy etc, require to run as daemon or background processes
- Singularity calls it an `instance`

```
> singularity instance start --bind /some/direcotries mysql.simg mysql

> singularity instance list
INSTANCE NAME    PID      IP              IMAGE
mysql            49322                    mysql.simg

> singularity run instance://mysql
```

Singularity instance (mysql)

- Launching a `mysql` server on compute node (in a jobscript):

```
> singularity instance start --bind
/some/direcotries mysql.simg mysql

> singularity run instance://mysql

> singularity exec instance://mysql
create_remote_admin_user.sh
INFO:    instance started successfully
/home/shaima0d/.my.cnf already exists.  Using
that version.
/home/shaima0d/.mysqlrootpw already
exists.  Using that version.
Start mysqld ……
Random password for remote user 'remote_usr':
fR3K6X6y
```

- On client side

```
> singularity shell ../mysql/mysql.simg
> mysql -h cn512-22-l \
        -u remote_usr -pfR3K6X6y
mysql: [Warning] Using a password on the command line
interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or
\g.
Your MySQL connection id is 6
Server version: 5.7.21 MySQL Community Server (GPL)
Copyright (c) 2000, 2018, Oracle and/or its
affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation
and/or its
affiliates. Other names may be trademarks of their
respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the
current input statement.
```

Writable containers

Unrolling containers in a directory

- At times you want to experiment with images and change them on the fly –
  - add or remove files from otherwise immutable rootfs of container

```
$ singularity build --sandbox hpl ../images/hpl_ompi_2.3.0.sif
INFO:    Starting build...
INFO:    Creating sandbox directory...
INFO:    Build complete: hpl
$ ls hpl/
bin   dev          etc   hpl  lib64  mnt  proc  run   singularity  sys  usr
boot  environment  home  lib  media  opt  root  sbin  srv          tmp  var
$ cat hpl/etc/os-release
NAME="Ubuntu"
VERSION="18.04.4 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.4 LTS"
VERSION_ID="18.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=bionic
UBUNTU_CODENAME=bionic
```

Run container from a sandbox

- Its possible to run the application from a sandbox

Jobscript

```
#!/bin/bash
#SBATCH --ntasks=16
#SBATCH --ntasks-per-node=8
#SBATCH -t 01:00:00
```

```
module load openmpi/4.0.3
module load singularity
```

```
export OMPI_MCA_btl=openib
export OMPI_MCA_btl_openib_allow_ib=1
mpirun -n 16 -N 8 singularity exec hpl/ hpl/hpl/bin/ubuntu/xhpl
```

Converting sandbox to SIF image

- When you have customized the sandbox, you would want to freeze it into an image

```
> ls hpl/opt/
mellanox
> touch hpl/opt/a_new_file.txt
> singularity build -f hpl_new.sif hpl/
INFO:    Starting build...
INFO:    Creating SIF file...
INFO:    Build complete: hpl_new.sif
$ singularity exec hpl_new.sif ls /opt
a_new_file.txt mellanox
```

- Note that these changes will not be reproducible when the image is rebuilt from a Singularity Definition File because they only apply to the sandbox directory

Use case Demos

Gromacs (CPU) -- Dockerfile

```
FROM krccl/openmpi_base:403

RUN apt-get install -y cmake libfftw3-dev WORKDIR /opt

RUN curl -o gromacs-2020.4.tar.gz -L http://ftp.gromacs.org/pub/gromacs/gromacs-
2020.4.tar.gz && \

tar zxvf gromacs-2020.4.tar.gz && \ rm gromacs-2020.4.tar.gz


WORKDIR gromacs-2020.4/build

RUN cmake -DCMAKE_C_COMPILER=mpicc -DCMAKE_CXX_COMPILER=mpicxx -DGMX_MPI=on -
DBUILD_SHARED_LIBS=on -DGMX_FFT_LIBRARY=fftw3 -DCMAKE_BUILD_TYPE=Release .. && \

make VERBOSE=1 && \

make install


ENV PATH /usr/local/gromacs/bin:$PATH ENV LD_LIBRARY_PATH
/usr/local/gromacs/lib:$LD_LIBRARY_PATH
```

Gromacs (CPU) – Jobscript (Ibex)

```
#!/bin/bash

#SBATCH --job-name="gromacs"
#SBATCH --ntasks=32
#SBATCH --tasks-per-node=32
#SBATCH --time=4:00:00
#-----------------------------------------------------------#

module load singularity openmpi/4.0.3
export OMPI_MCA_btl=openib
export OMPI_MCA_btl_openib_allow_ib=1
export SINGUALRITYENV_GROMACS_USE=""
export SINGULARITYENV_OMP_NUM_THREADS=1

export IMAGE=/ibex/scratch/shaima0d/scratch/singularity_mpi_testing/images/gromacs_cpu_latest.sif
export GROMACS_HOME=/usr/local/gromacs
.............
.............
# Step Six: Equilibration
mpirun -np 1  singularity exec ${IMAGE} ${GROMACS_HOME}/bin/gmx_mpi grompp -f nvt.mdp -c em.gro -r
                                                                    em.gro -p topol.top -o nvt.tpr
mpirun -np 32 singularity exec ${IMAGE} ${GROMACS_HOME}/bin/gmx_mpi mdrun -deffnm nvt
```

Gromacs (CPU) – Jobscript (Shaheen)

```
#!/bin/bash
#SBATCH --job-name="gromacs"
#SBATCH --ntasks=32
#SBATCH --time=4:00:00
#-----------------------------------------------------#
module swap PrgEnv-cray PrgEnv-gnu
module load openmpi/4.0.3
module load singularity
#-----------------------------------------------------#
export LD_LIBRARY_PATH=$CRAY_LD_LIBRARY_PATH:$LD_LIBRARY_PATH
export SINGULARITYENV_LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/cray/……:/usr/lib
export SINGULARITYENV_APPEND_PATH=$PATH
export BIND_MOUNT="-B /sw,/usr/lib64,/opt,/etc"
export SINGUALRITYENV_GROMACS_USE=""
export SINGULARITYENV_OMP_NUM_THREADS=1

export IMAGE=/project/k01/shaima0d/singularity_test/images/gromacs_cpu_latest.sif
export GROMACS_HOME=/usr/local/gromacs
.............
.............
# Step Six: Equilibration
mpirun -np 1  singularity exec ${BIND_MOUNT} ${IMAGE} ${GROMACS_HOME}/bin/gmx_mpi grompp -f nvt.mdp -c
em.gro -r em.gro -p topol.top -o nvt.tpr
mpirun -np 32 singularity exec ${BIND_MOUNT} ${IMAGE} ${GROMACS_HOME}/bin/gmx_mpi mdrun -deffnm nvt
```

Gromacs on Ibex (CPU)

Shaheen	**Ibex**
~ 0.3173 hour/ns	~ 0.267 hour/ns

OpenFOAM -- Dockerfile

```
FROM krccl/openmpi_base:403
RUN  apt-get install -y flex bison cmake zlib1g-dev libboost-system-dev libboost-thread-dev \
       gnuplot libreadline-dev libncurses-dev libxt-dev qt4-dev-tools libqt4-dev libqt4-opengl-dev freeglut3-dev libqtwebkit-dev
```

```
WORKDIR /app
RUN curl -o OpenFOAM-v2006.tgz -L https://sourceforge.net/projects/openfoam/files/v2006/OpenFOAM-v2006.tgz && \
    curl -o ThirdParty-v2006.tgz -L https://sourceforge.net/projects/openfoam/files/v2006/ThirdParty-v2006.tgz && \
    tar xvf OpenFOAM-v2006.tgz && tar xvf ThirdParty-v2006.tgz && \
    rm OpenFOAM-v2006.tgz ThirdParty-v2006.tgz
```

```
# Creating prefs.sh for changing forwarding preferences to bashrc
RUN echo 'export FOAM_INST_DIR="/app/OpenFOAM-v2006" \nexport WM_MPLIB=SYSTEMOPENMPI \nexport
WM_COMPILER_TYPE=system \nexport WM_COMPILER=Gcc \nexport WM_LABEL_SIZE=64 \nexport CC=mpicc CXX=mpicxx
FC=mpif90 \n' >> $OF_HOME/etc/prefs.sh
```

```
WORKDIR /app/
SHELL ["/bin/bash", "-c"]
RUN source /app/OpenFOAM-v2006/etc/bashrc && wclean -a && ./Allwmake -s -l -j1
```

```
ENV PATH=/app/OpenFOAM-v2006/site/2006/platforms/linux64GccDPInt64Opt/bin:……:/usr/bin:/sbin:/bin
ENV LD_LIBRARY_PATH=/app/ThirdParty-v2006/platforms/linux64Gcc/fftw-3.3.7/lib64:……:/usr/lib/x86_64-linux-gnu
ENV FOAM_APP=/app/OpenFOAM-v2006/applications
ENV FOAM_UTILITIES=/app/OpenFOAM-v2006/applications/utilities
ENV FOAM_API=2006
ENV FOAM_APPBIN=/app/OpenFOAM-v2006/platforms/linux64GccDPInt64Opt/bin
```

OpenFOAM -- Jobscript (Shaheen)

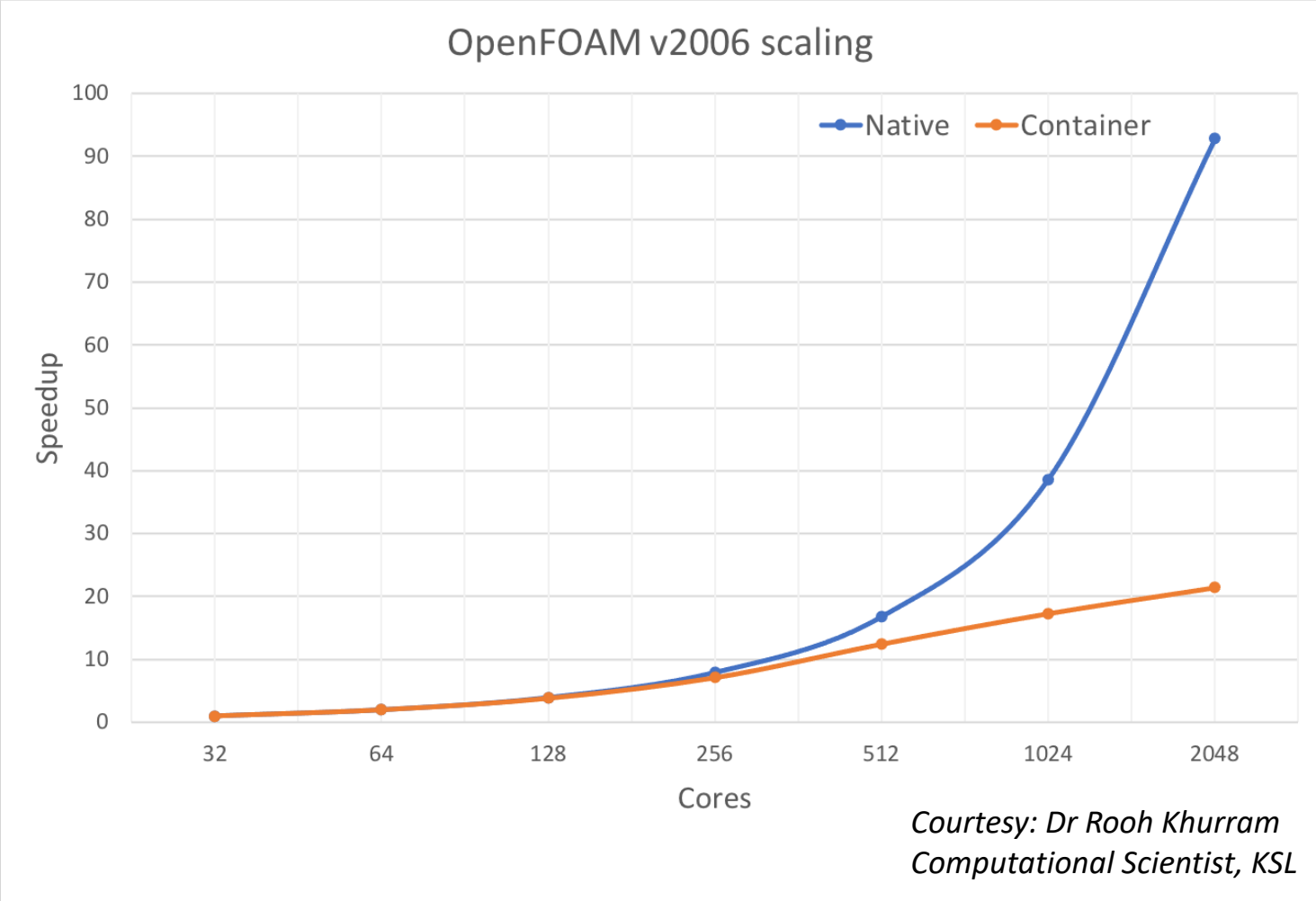
```
#!/bin/bash
#SBATCH --job-name=openfoam-container
#SBATCH --time=02:00:00
#SBATCH --nodes=1

module swap PrgEnv-cray PrgEnv-gnu
module use /project/k01/shaima0d/software/cle7up01/modulefiles
module load openmpi/4.0.3
module load singularity
export UCX_NET_DEVICES=ipogif0

export LD_LIBRARY_PATH=$CRAY_LD_LIBRARY_PATH:$LD_LIBRARY_PATH
export SINGULARITYENV_LD_LIBRARY_PATH=/app/ThirdParty-v2006/platforms/……:/usr/lib64:/usr/lib:$LD_LIBRARY_PATH
export SINGULARITYENV_APPEND_PATH=$PATH:/root/OpenFOAM-v2006/platforms/……:/sbin:/bin

export IMAGE=opefoam_openmpi_base403_2006.sif
export BIND_MOUNT="-B $UCX_DIR,$OPENMPI_DIR,/usr/lib64,/opt,/etc”
……
mpirun -np 32 singularity exec $BIND_MOUNT $IMAGE simpleFoam $decompDict -parallel >> simpleFoam.log
```

OpenFOAM Scaling: Native vs Container



Courtesy: Dr Rooh Khurram
Computational Scientist, KSL

DL training on Ibex GPUs – Dockerfile

```
FROM nvcr.io/nvidia/cuda:10.1-cudnn7-devel-ubuntu18.04
RUN apt-get update && apt-get install -y …….
RUN apt-get install -y curl make vim apt-utils libnuma-dev libnl-3-200
RUN apt-get install -y gcc-6 g++-6 gfortran-6
RUN update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-6 80 --slave
/usr/bin/g++ g++ /usr/bin/g++-6 --slave /usr/bin/gcov gcov /usr/bin/gcov-6
```

DL training on Ibex GPUs – Dockerfile

```
#Install MOFED OpenMPI
WORKDIR /opt
COPY MLNX_OFED_LINUX-5.0-2.1.8.0-ubuntu18.04-x86_64.tgz /opt
RUN tar xvf MLNX_OFED_LINUX-5.0-2.1.8.0-ubuntu18.04-x86_64.tgz
RUN echo "deb file:/opt/MLNX_OFED_LINUX-5.0-2.1.8.0-ubuntu18.04-
x86_64/DEBS/UPSTREAM_LIBS ./" > /etc/apt/sources.list.d/mlnx_ofed.list
RUN curl -L http://www.mellanox.com/downloads/ofed/RPM-GPG-KEY-Mellanox | apt-key
add - && rm -rf /opt/MLNX_OFED_LINUX*
RUN curl -o /opt/openmpi-4.0.3.tar https://download.open-mpi.org/release/open-
mpi/v4.0/openmpi-4.0.3.tar.gz
RUN tar xvf /opt/openmpi-4.0.3.tar
WORKDIR /opt/openmpi-4.0.3
RUN ./configure --prefix=/usr/local --with-verbs --with-cuda=/usr/local/cuda && make
-j 8 VERBOSE=1 && make install
WORKDIR /
RUN rm -rf /opt/openmpi-4.0.3.tar /opt/openmpi-4.0.3 && ldconfig
ENV LD_LIBRARY_PATH /usr/local/lib:/usr/lib/x86_64-linux-gnu:$LD_LIBRARY_PATH
ENV PATH /usr/local/bin:$PATH
```

DL training on Ibex GPUs – Dockerfile

```
# TensorFlow version is tightly coupled to CUDA and cuDNN so it should be selected carefully
ENV TENSORFLOW_VERSION=2.2.0
ENV PYTORCH_VERSION=1.6.0
ENV TORCHVISION_VERSION=0.7.0
ENV NCCL_VERSION=2.7.8-1+cuda10.1
ENV MXNET_VERSION=1.6.0.post0
# Python 3.7 is supported by Ubuntu Bionic out of the box
ARG python=3.7
ENV PYTHON_VERSION=${python}
```

```
# Set default shell to /bin/bash
SHELL ["/bin/bash", "-cu"]
RUN apt-get install -y --allow-change-held-packages --no-install-recommends \
        build-essential cmake git ca-certificates python${PYTHON_VERSION} \
        python${PYTHON_VERSION}-dev python${PYTHON_VERSION}-distutils \
        libnccl2=${NCCL_VERSION} libnccl-dev=${NCCL_VERSION} libjpeg-dev libpng-dev
RUN ln -s /usr/bin/python${PYTHON_VERSION} /usr/bin/python
RUN curl -O https://bootstrap.pypa.io/get-pip.py && python get-pip.py && rm get-pip.py
```

DL training on Ibex GPUs – Dockerfile

```
# Install TensorFlow, Keras, PyTorch and MXNet
RUN pip install future typing packaging
RUN pip install tensorflow==${TENSORFLOW_VERSION} keras h5py

#Install Pytorch
RUN pip install torch==1.6.0+cu101 torchvision==0.7.0+cu101 -f
https://download.pytorch.org/whl/torch_stable.html
#RUN PYTAGS=$(python -c "from packaging import tags; tag = list(tags.sys_tags())[0];
print(f'{tag.interpreter}-{tag.abi}')") && \


#Installing MXNet
RUN pip install mxnet-cu101==${MXNET_VERSION}
```

DL training on Ibex GPUs – Dockerfile

```
WORKDIR /opt
# Install Horovod, temporarily using CUDA stubs
RUN ldconfig /usr/local/cuda/targets/x86_64-linux/lib/stubs && \
    git clone --recursive https://github.com/horovod/horovod horovod-0.19.2 && \
    cd horovod-0.19.2 && git fetch --tags --all && \
    git checkout v0.19.2 && git submodule sync && \
    git submodule update --init --recursive
```

DL training on Ibex GPUs – Jobscript
Single GPU

```
#!/bin/bash
#SBATCH --gres=gpu:1
#SBATCH --constraint=v100
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=6
#SBATCH --mem=64G
#SBATCH --time=00:30:00

module load openmpi/4.0.3-cuda10.1
module load singularity

export IMAGE=horovod_gpu_0192.sif

echo "PyTorch with Horovod”
mpirun -np 1  singularity exec --nv $IMAGE python ./pytorch_synthetic_benchmark.py --model
resnet50 --batch-size 128 --num-warmup-batches 10 --num-batches-per-iter 10 --num-iters 10
```

DL training on Ibex GPUs – Results
Single GPU

```
Model: resnet50
Batch size: 128
Number of GPUs: 1
Running warmup...
Running benchmark...
Iter #0: 354.9 img/sec per GPU
……
Img/sec per GPU: 354.2 +-0.9
Total img/sec on 1 GPU(s): 354.2 +-0.9
```

DL training on Ibex GPUs – Jobscript
Multi-GPU on same node

```
#!/bin/bash
#SBATCH --gres=gpu:8
#SBATCH --constraint=v100
#SBATCH --ntasks=8
#SBATCH --tasks-per-node=8
#SBATCH --cpus-per-task=6
#SBATCH --mem=64G
#SBATCH --time=00:30:00

module load openmpi/4.0.3-cuda10.1
module load singularity

export IMAGE=horovod_gpu_0192.sif

echo "PyTorch with Horovod"
mpirun -np 8  singularity exec --nv $IMAGE python ./pytorch_synthetic_benchmark.py --model
resnet50 --batch-size 128 --num-warmup-batches 10 --num-batches-per-iter 10 --num-iters 10
```

DL training on Ibex GPUs – Results
Multi-GPU on same node

```
Model: resnet50
Batch size: 128
Number of GPUs: 8
Running warmup...
Running benchmark...
Iter #0: 344.3 img/sec per GPU
……
Img/sec per GPU: 343.8 +-1.2
Total img/sec on 8 GPU(s): 2750.5 +-9.6
```

DL training on Ibex GPUs – Jobscript
Multi-GPUs on multi node

```
#!/bin/bash
#SBATCH --gres=gpu:8
#SBATCH --constraint=v100
#SBATCH --ntasks=8
#SBATCH --tasks-per-node=4
#SBATCH --cpus-per-task=6
#SBATCH --mem=64G
#SBATCH --time=00:30:00

module load openmpi/4.0.3-cuda10.1
module load singularity

export IMAGE=horovod_gpu_0192.sif


echo "PyTorch with Horovod"
mpirun -np 8 -N 4 singularity exec --nv $IMAGE python ./pytorch_synthetic_benchmark.py --model
resnet50 --batch-size 128 --num-warmup-batches 10 --num-batches-per-iter 10 --num-iters 10
```

DL training on Ibex GPUs – Results
Multi-GPUs on multi node

Working with images from **NVIDIA NGC Registry**

- You will need to generate an API key, one off, to access NGC registry.
- For this you first need to create an account.
- Please follow the instruction on the link below or email ibex@hpc.kaust.edu.sa for assistance

https://docs.nvidia.com/ngc/ngc-getting-started-guide/index.html#generating-api-key

GROMACS GPU image from NGC registry

Once registered, you should be able to access NGC image registry and pull images

```
> singularity pull docker://nvcr.io/hpc/gromacs:2020.2
```

- Pull images in $HOME directory with `SINGULARITY_TMPDIR` set to somewhere in $HOME directory

GROMACS GPU -- Jobscript

```
#!/bin/bash
#SBATCH --job-name="gromacs"
#SBATCH --ntasks=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --gres=gpu:1
#SBATCH --constraint=v100
#SBATCH --time=1:00:00
#-----------------------------------------------------------#

module load singularity
module load openmpi/4.0.3-cuda11.0
export OMPI_MCA_btl=openib
export OMPI_MCA_btl_openib_allow_ib=1

…….
```

GROMACS GPU -- Jobscript

```
export
SINGULARITYENV_LD_LIBRARY_PATH=/usr/local/gromacs/sm70/lib:/usr/local/fftw/lib:/usr/local/cuda/li
b:/usr/local/cuda/lib64:/.singularity.d/libs
export SINGULARITYENV_GMXBIN=/usr/local/gromacs/sm70/bin
export SINGULARITYENV_CUDA_HOME=/sw/csgv/cuda/11.0.1
export SINGULARITYENV_GMXDATA=/usr/local/gromacs/sm70/share/gromacs
export SINGULARITYENV_GMXLDLIB=/usr/local/gromacs/sm70/lib
export SINGULARITYENV_GMXMAN=/usr/local/gromacs/sm70/share/man
export
SINGULARITYENV_PATH=/usr/local/gromacs/sm70/bin:/usr/local/nvidia/bin:/usr/local/cuda/bin:/usr/local/sbin:/usr/local/
bin:/usr/sbin:/usr/bin:/sbin:/bin
export SINGULARITYENV_GROMACS_DIR=/usr/local/gromacs/sm70


export IMAGE=/ibex/scratch/shaima0d/scratch/singularity_mpi_testing/images/gromacs_2020.2.sif
export GROMACS_HOME=/usr/local/gromacs/sm70

…….
```

GROMACS GPU -- Jobscript

……

```
# Step Six: Equilibration

mpirun -np 1 singularity exec --nv ${IMAGE} ${GROMACS_HOME}/bin/gmx grompp -f nvt.mdp -c em.gro -r
em.gro -p topol.top -o nvt.tpr

mpirun -np 1 -N 1  singularity exec --nv ${IMAGE} ${GROMACS_HOME}/bin/gmx mdrun -deffnm nvt
```

……

Gromacs on Ibex (CPU)

Shaheen	**Ibex**
CPU: ~ 0.3173 hours/ns	CPU: ~ 0.267 hours/ns GPU: ~0.567 hours/ns

Running Services – mysql

- Live Demo of running mysql server and connecting a client application (LIMS) running in a separate container.

Best practices

- Sourcing environment setting script is painful in Docker
  - Can do manually as done in case of OpenFOAM
  - If bind mounting directories, as on Shaheen, both PATH and LD_LIRBARY_PATH needs to be updated

OpenFOAM -- Jobscript (Shaheen)

```
#!/bin/bash
#SBATCH --job-name=openfoam-container
#SBATCH --time=02:00:00
#SBATCH --nodes=1

module swap PrgEnv-cray PrgEnv-gnu
module use /project/k01/shaima0d/software/cle7up01/modulefiles
module load openmpi/4.0.3
module load singularity
export UCX_NET_DEVICES=ipogif0

export LD_LIBRARY_PATH=$CRAY_LD_LIBRARY_PATH:$LD_LIBRARY_PATH
export SINGULARITYENV_LD_LIBRARY_PATH=/app/ThirdParty-
v2006/platforms/……:/usr/lib64:/usr/lib:$LD_LIBRARY_PATH
export SINGULARITYENV_APPEND_PATH=$PATH:/root/OpenFOAM-v2006/platforms/……:/sbin:/bin

export IMAGE=opefoam_openmpi_base403_2006.sif
export BIND_MOUNT="-B $UCX_DIR,$OPENMPI_DIR,/usr/lib64,/opt,/etc”
……
mpirun -np 32 singularity exec $BIND_MOUNT $IMAGE simpleFoam $decompDict -parallel >> simpleFoam.log
```

Best practices

- Libc version shouldn’t be too old inside the container compared to host kernel
  - On Ibex, Kernel release is:

```
> uname -r
  3.10.0-1062.12.1.el7.x86_64
```

- Singularity is Read Only by default
  - Except for the directories mounted by default on Shaheen/Ibex
    - /tmp, /home, /project (on Shaheen), /scratch
  - Make it writable as sandbox if needed

- For large images, pull may fail due to insufficient space in /tmp
  - Run `singularity pull` command in $HOME which is NFS
  - export SINGULARITY_TMPDIR=$HOME/some/path

Best practices

- While building images with `--fakeroot` on Ibex,
  - Always allocate a compute node on, (won't work on login nodes)
  - `export XDG_RUNTIME_DIR=$HOME/somewhere`, to allow temporary space for Singularity to write intermediate blobs/images

Best practices

By default Docker image blobs are cached in ~/.singularity/cache

. It can fill pretty quickly you pull different images frequently.

- To clean cache run:

```
> singularity cache clean
```

- Ports are published by default, mapped on same ports as host
- Bind mount $HOME can have some unintended implications:
  - e.g. Python maintains user packages in $HOME
  - To mitigate use --contain with singularity exec or run

Documentation and support

- Documentation : It work in progress
  https://www.hpc.kaust.edu.sa/ibex/containers

- Support
  - Email us
    - Shaheen : help@hpc.kaust.edu.sa
    - Ibex : ibex@hpc.kaust.edu.sa
  - Please use “Containers:” in subject line so your request lands with relevant support persons

Slides and recording will be shared in follow up email of this webinar

Q/A & wrapup

Thank you for attending

Happy containerinzing