A simplistic view of a scientific application
Received an allocation on Blue Waters!
A more realistic view of a scientific application

My Application

LIBRARY X

EXECUTABLE Y

HARDWARE Z

DATA

RESULTS

Your Computer
A more realistic view of a scientific application
Software Containers

My Application

LIBRARY X

EXECUTABLE Y

HARDWARE C

DATA

RESULTS
**Shifter**

- Designed for HPC
- [GitHub](https://github.com/NERSC/shifter)

**Docker**

- Not suitable for HPC
- [GitHub](https://github.com/docker)

**Singularity**

- Can be used in HPC environment
- [GitHub](https://github.com/singularityware/singularity)
FROM centos:7
RUN yum update && ... wget ...
COPY file.ext inside/container/
RUN ./configure && make && ...
WORKDIR /home/user

1) Build: docker build -t repo/name:tag .
2) Upload: docker push repo/name:tag
3) Download: docker pull repo/name:tag
4) Run: docker run repo/name:tag
4a) Execute: docker exec <container> command
-l gres=shifter16
-l gres=shifter16
-v UDI=repo/name:tag
export CRAY_ROOTFS=SHIFTER
aprun -b ... --app-in-container --options

UDI: User-Defined Image
```
-l gres=shifter16
module load shifter
aprun -b ...
shifter --image=repo/name:tag --app-in-container --options
```
# Shifter module

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>shifter</td>
<td>launches an app from UDI on compute nodes</td>
</tr>
<tr>
<td>shifterimg</td>
<td></td>
</tr>
<tr>
<td>images</td>
<td>returns info about available UDIs</td>
</tr>
<tr>
<td>lookup</td>
<td>returns an ID of an image (if available)</td>
</tr>
<tr>
<td>pull</td>
<td>downloads Docker image and converts it to UDI</td>
</tr>
<tr>
<td>login</td>
<td>authenticates to DockerHub (for private images)</td>
</tr>
</tbody>
</table>
Examples

```
shifterimg images
shifterimg lookup ubuntu:zesty # image on Blue Waters
shifterimg pull centos:latest # image on DockerHub
shifterimg login
shifterimg pull --user bw_user prvimg1:latest
shifterimg pull --group bw_grp prvimg2:latest
```
MPI in Shifter: requirements

In a Container:

1. glibc 2.17+ (CentOS 7, Ubuntu 14)

2. Use one of the following MPIs (see MPICH-ABI compatibility)
   - MPICH v3.1+
   - Intel® MPI Library v5.0+
   - Cray MPT v7.0.0+
   - MVAPICH2 2.0+
   - Parastation MPI 5.1.7-1+
   - IBM MPI v2.1+

3. Do not use package manager to install MPI*
MPI in Shifter: requirements

On Blue Waters:

1. Use **Intel** or **GNU** Programming Environment
2. Load **MPICH ABI-compatible** MPI (on BW: **cray-mpich-abi**)
3. Set **LD_LIBRARY_PATH** to the location of MPICH-ABI libraries
FROM centos:7
RUN yum -y install file gcc make gcc-gfortran gcc-c++ wget curl
RUN cd /usr/local/src/ && \
    wget http://www.mpich.org/static/downloads/3.2/mpich-3.2.tar.gz && \
    tar xf mpich-3.2.tar.gz && \
    rm mpich-3.2.tar.gz && \
    cd mpich-3.2 && \
    ./configure && \
    make && make install && \
    cd /usr/local/src && \
    rm -rf mpich-3.2*
module unload PrgEnv-cray
module unload cce
module load PrgEnv-gnu
module unload cray-mpich
module load cray-mpich-abi

LIBS=\$CRAY_LD_LIBRARY_PATH:/opt/cray/wlm_detect/default/lib64
CACHE=\$PWD/cache.\$PBS_JOBID
mkdir -p $CACHE
for dir in $( echo $LIBS | tr ":" " " ); do cp -L -r $dir $CACHE; done
export LD_LIBRARY_PATH=$CACHE/lib:$CACHE/lib64
MPI in Shifter

1

qsub -l gres=shifter16 -l nodes=2:ppn=32:xe
module load shifter
aprun -b -n 64 -N 32 -- shifter --image=repo/image:tag -- app
aprun -b -n 64 -N 32 -- shifter --image=repo/image2:tag -- app2

2

qsub -l gres=shifter16 -v UDI=repo/image:tag -l nodes=2:ppn=32:xe
export CRAY_ROOTFS=SHIFTER
aprun -b -n 64 -N 32 -- app
MPI in Shifter

\[ T(N, \text{ppn}=1) \]

\[ T(N=80, \text{ppn}) \]

Hon Wai Leong
Performance of MPI applications in Shifter

**Figure a:**
- Time (µs) vs. Message size (bytes) for MPI_Alltoall and MPI_Alltoallv in Shifter CLE.
- Two plots showing the performance of different MPI operations.

**Figure b:**
- Time (ms) vs. Message size (kilobytes) for MPI_Bcast and MPI_Reduce in Shifter CLE.
- Two plots showing the performance of different MPI operations.

**Graphs:**
- Shifter CLE marked with different symbols for different operations.
- Comparison with Cray Linux Environment.

**Benchmarks:**
- Benchmark number on the x-axis.
- IOR Read and Write (GB/s) on the y-axis.

**Authors:**
- Galen Arnold

**DOI:** 10.1145/3219104.3219145
Using GPUs in Shifter

1. Use Shifter CLI:

   `shifter --image ...`

2. Set `CUDA_VISIBLE_DEVICES`:

   `export CUDA_VISIBLE_DEVICES=0`

`aprun -b ... -- shifter --image=centos:latest -- nvidia-smi`
Using MPI + GPUs in Shifter

Caveat: **shifter** command removes `LD_LIBRARY_PATH`

**Solution:** use a script to pass `LD_LIBRARY_PATH`

```
aprun -b -n ... -N 1 -e VAR="$LD_LIBRARY_PATH" ... \n  shifter --image=repo/name:tag ... -- script.sh
```

**script.sh:**

```
export LD_LIBRARY_PATH="$VAR"
application
```
Directory mapping between BW and Shifter UDI

shifter ... --volume=/path/on/BW:/path/in/UDI

Important: /path/in/UDI must exist in UDI

qsub -v UDI="centos:latest -v /u/sciteam/$USER:/home"

/projects, /scratch, and “Home” folders are mapped between Blue Waters and UDI
https://github.com/NERSC/shifter

https://github.com/singularityware/singularity
-l gres=singularity

User

ssh

Login nodes

qsub

MOM nodes

aprun

Compute nodes
-l gres=singularity

module load singularity

singularity build IMAGE.simg docker://centos:latest

singularity exec -H /u:/home IMAGE.simg -- app
Directory mapping between BW and Singularity

`singularity exec -B /path/on/BW:/path/in/singularity`

And... `/path/in/singularity` does not have to exist.

Note: `/projects` on BW is `/mnt/abc/projects` in Singularity
`/scratch` on BW is `/mnt/abc/scratch` in Singularity
GPUs in Singularity

```
singularity exec --nv ...
```

If you’re using BW directories, make sure they are mapped into Singularity image

**Example:**

```
aprun -n 16 -N 8 -b -- singularity exec -B /dsl/opt/cray/nvidia/default/bin:/home/staff/mbelkin
/home/staff/mbelkin/nvidia-smi
```
Thank you!