Step-by-step guide to Shifter

A note about SLURM commands

Piz Daint uses the SLURM Workload Manager to assign jobs to its compute nodes. In case you are not familiar with basic usage of SLURM, here we provide brief explanations to the commands used throughout this guide:

`salloc` is used to allocate resources for a job in real time. Typically this is used to allocate resources and spawn a shell. The shell is then used to execute `srun` commands to launch parallel tasks.

`srun` is used to submit a job for execution or initiate job steps in real time.

Both these commands support the following options:

`-C` indicates a list of contraints for the nodes where to make an allocation or run a job. In this document we will be using `-C gpu` to indicate we want to run on Piz Daint's hybrid partition, with nodes featuring Intel Haswell CPUs and NVIDIA Pascal GPUs.

`--reservation` allocates resources on a specific reservation (please note that the reservation for the hands-on will expire shortly after the session itself has ended).

`-n` indicates the total number of tasks to run

`-N` indicates the number of compute nodes to use

---

1. Query Shifter images

   You can list the Shifter images available on a system with the `shifterimg images` command.

   EXAMPLE:

   ```
   $ shifterimg images

   daint      docker     READY    b7ccb5a012   2017-05-10T02:27:09 debian:jessie
   daint      docker     READY    80e46367f8   2017-05-09T15:00:41 centos:6.8
   daint      docker     READY    4a9dd14f74   2017-05-09T15:03:22
   my_custom_image:latest
   ```

2. Pull a new image from Docker Hub

   You can pull images from Docker Hub into the HPC system with the `shifterimg pull` command.

   EXAMPLE:

   ```
   $ shifterimg pull debian:jessie

   2017-05-19T16:11:50 Pulling Image: docker:debian:jessie, status: READY
   ```

3. Run a container with Shifter

   You can run containers with the `shifter` command. The image is selected with the `--image` command line option. Everything that follows `shifter` and its options will be executed as a command inside the container. You can check that you are actually running in a container by inspecting `/etc/os-release`.

   EXAMPLE:

```
$ salloc -n 1 -C gpu --reservation=shifter-cscs
$ srun shifter --image=debian:jessie cat /etc/os-release

PRETTY_NAME="Debian GNU/Linux 8 (jessie)"
NAME="Debian GNU/Linux"
VERSION_ID="8"
VERSION="8 (jessie)"
ID=debian
HOME_URL="http://www.debian.org/"
SUPPORT_URL="http://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"

$ srun cat /etc/os-release

NAME="SLES"
VERSION="12"
VERSION_ID="12"
PRETTY_NAME="SUSE Linux Enterprise Server 12"
ID="sles"
ANSI_COLOR="0;32"
CPE_NAME="cpe:/o:suse:sles:12"
```

4. Run a container with an interactive shell

   As with Docker, you can access Shifter containers through an interactive shell. As opposed to Docker, no specific command line options are required by Shifter to start an interactive shell. The `--pty` flag to `srun` is optional, but it makes the experience much more user-friendly.

   EXAMPLE:

```
$ salloc -n 1 -C gpu --reservation=shifter-cscs
$ srun --pty shifter --image=debian:jessie bash
$ cat /etc/os-release

PRETTY_NAME="Debian GNU/Linux 8 (jessie)"
NAME="Debian GNU/Linux"
VERSION_ID="8"
VERSION="8 (jessie)"
ID=debian
HOME_URL="http://www.debian.org/"
SUPPORT_URL="http://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
```

5. Access personal files from the container

   Your home and scratch directories are automatically mounted inside the container so all your personal files are available within the container environment.

   EXAMPLE (exact path of the scratch is dependent on your system):

```
$ cd $HOME
$ touch test_file
$ salloc -n 1 -C gpu --reservation=shifter-cscs
$ srun shifter --image=debian:jessie ls

test_file

$ cd $SCRATCH
$ srun shifter --image=debian:jessie pwd

/scratch/snx3000/<user name>
```

6. Detect GPUs available in the container

   Enabling native GPU support in Shifter does not require any direct user action, besides using an image with the CUDA Toolkit installed.

   Also, Shifter does not rely on specific labels in the image, as opposed to `nvidia-docker`.

   To list the GPU devices available in the container, you can run the CUDA sample `deviceQuery` which is provided with the CUDA Toolkit SDK. We have already built an image with compiled CUDA samples, and you can retrieve it from Docker Hub using the identifier `ethcscs/dockerfiles:cudasamples8.0`.

   EXAMPLE:

```
$ shifterimg pull ethcscs/dockerfiles:cudasamples8.0
$ salloc -n 1 -C gpu --reservation=shifter-cscs
$ srun shifter --image=ethcscs/dockerfiles:cudasamples8.0
/usr/local/cuda/samples/bin/x86_64/linux/release/deviceQuery

/usr/local/cuda/samples/bin/x86_64/linux/release/deviceQuery Starting...

 CUDA Device Query (Runtime API) version (CUDART static linking)

Detected 1 CUDA Capable device(s)

Device 0: "Tesla P100-PCIE-16GB"
  CUDA Driver Version / Runtime Version          8.0 / 8.0
  CUDA Capability Major/Minor version number:    6.0
  Total amount of global memory:                 16276 MBytes (17066885120
bytes)
  (56) Multiprocessors, ( 64) CUDA Cores/MP:     3584 CUDA Cores
  GPU Max Clock rate:                            1329 MHz (1.33 GHz)
  Memory Clock rate:                             715 Mhz
  Memory Bus Width:                              4096-bit
  L2 Cache Size:                                 4194304 bytes
  Maximum Texture Dimension Size (x,y,z)         1D=(131072), 2D=(131072,
65536), 3D=(16384, 16384, 16384)
  Maximum Layered 1D Texture Size, (num) layers  1D=(32768), 2048 layers
  Maximum Layered 2D Texture Size, (num) layers  2D=(32768, 32768), 2048
layers
  Total amount of constant memory:               65536 bytes
  Total amount of shared memory per block:       49152 bytes
  Total number of registers available per block: 65536
```

```
  Warp size:                                     32
  Maximum number of threads per multiprocessor:  2048
  Maximum number of threads per block:           1024
  Max dimension size of a thread block (x,y,z): (1024, 1024, 64)
  Max dimension size of a grid size    (x,y,z): (2147483647, 65535, 65535)
  Maximum memory pitch:                          2147483647 bytes
  Texture alignment:                             512 bytes
  Concurrent copy and kernel execution:          Yes with 2 copy engine(s)
  Run time limit on kernels:                     No
  Integrated GPU sharing Host Memory:            No
  Support host page-locked memory mapping:       Yes
  Alignment requirement for Surfaces:            Yes
  Device has ECC support:                        Enabled
  Device supports Unified Addressing (UVA):      Yes
  Device PCI Domain ID / Bus ID / location ID:   0 / 2 / 0
  Compute Mode:
     < Exclusive Process (many threads in one process is able to use
::cudaSetDevice() with this device) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 8.0, CUDA Runtime
Version = 8.0, NumDevs = 1, Device0 = Tesla P100-PCIE-16GB
Result = PASS
```

7. Run GPU application in the container

   You can run the `nbody` CUDA sample which is provided with the CUDA Toolkit SDK.

   EXAMPLE:

```
$ salloc -n 1 -C gpu --reservation=shifter-cscs
$ srun shifter --image=ethcscs/dockerfiles:cudasamples8.0
/usr/local/cuda/samples/bin/x86_64/linux/release/nbody -benchmark -fp64 -
numbodies=200000

Run "nbody -benchmark [-numbodies=<numBodies>]" to measure performance.
    -fullscreen       (run n-body simulation in fullscreen mode)
    -fp64             (use double precision floating point values for
simulation)
    -hostmem          (stores simulation data in host memory)
    -benchmark        (run benchmark to measure performance)
    -numbodies=<N>    (number of bodies (>= 1) to run in simulation)
    -device=<d>       (where d=0,1,2.... for the CUDA device to use)
    -numdevices=<i>   (where i=(number of CUDA devices > 0) to use for
simulation)
    -compare          (compares simulation results running once on the default
GPU and once on the CPU)
    -cpu              (run n-body simulation on the CPU)
    -tipsy=<file.bin> (load a tipsy model file for simulation)

NOTE: The CUDA Samples are not meant for performance measurements. Results may
vary when GPU Boost is enabled.

> Windowed mode
> Simulation data stored in video memory
> Double precision floating point simulation
> 1 Devices used for simulation
GPU Device 0: "Tesla P100-PCIE-16GB" with compute capability 6.0

> Compute 6.0 CUDA device: [Tesla P100-PCIE-16GB]
Warning: "number of bodies" specified 200000 is not a multiple of 256.
Rounding up to the nearest multiple: 200192.
200192 bodies, total time for 10 iterations: 4400.005 ms
= 91.084 billion interactions per second
= 2732.509 double-precision GFLOP/s at 30 flops per interaction
```

To see the effect of GPU acceleration, try to run the sample benchmark on the CPU using the `-cpu` option. We advise to reduce the number of bodies specified with the `-numbodies` option to avoid waiting too long.

8. Run MPI application in the container

   To enable native MPI support, supply the `--mpi` command line option to Shifter.

   As an example, you can run the MPI latency test which is part of the OSU benchmarks. We have already built a container image with the OSU benchmarks, and you can retrieve it from Docker Hub using the identifier `ethcscs/dockerfiles:osu5.3.2-mpich3.1.4`.

   EXAMPLE:

```
$ salloc -C gpu -N 2 --reservation=shifter-cscs
$ srun -n 2 shifter --mpi --image=ethcscs/dockerfiles:osu5.3.2-mpich3.1.4
/usr/local/libexec/osu-micro-benchmarks/mpi/pt2pt/osu_latency

# OSU MPI Latency Test v5.3.2
# Size          Latency (us)
0                       1.14
1                       1.13
2                       1.11
4                       1.10
8                       1.10
16                      1.11
32                      1.08
64                      1.08
128                     1.10
256                     1.11
512                     1.14
1024                    1.38
2048                    1.65
4096                    2.25
8192                    4.36
16384                   5.20
32768                   6.86
65536                  10.21
131072                 16.84
262144                 30.15
524288                 56.87
1048576               110.00
2097152               216.69
4194304               433.59
```