Python Best Practices on Blue Waters

Colin MacLean (NCSA)
Email: cmaclean@illinois.edu
Python on Blue Waters

Python support on Blue Waters is provided by the BWPY module

$ module load bwpy

Other installations, such as Anaconda in your home directory, are not supported
BWPY versioning

• BWPY uses major.minor.patch versioning.
  - Major versions are for major changes
    • Different default python version (including minor)
    • Possibly a self-contained glibc, requiring a complete rebuild
  - Minor is for package updates
  - Patch fixes problems, mostly keeping package versions the same, unless specific package versions are broken. New packages may be added.
• The requested level of API/ABI stability can be selected via latest-stable, major.minor.x, major.y.x, and specific versions
Python on Blue Waters

Python Interpreters

CPython 2.7 (alias: python2)
CPython 3.5 (aliases: python, python3)
CPython 3.6
Pypy
Pypy3

Now with much improved CPython compatibility!
module load

- bwpy-mpi: MPI support enabled (should only be used on compute nodes!)
- bwpy-libsci_mp: BWPY built with OpenMP Cray BLAS libraries (libsci_mp)
- bwpy-libsci_acc: BWPY built with auto CUDA BLAS libraries (libsci_acc)
- bwpy-visit: BWPY's VisIt (requires older vtk, so is a separate module)
- bwpy-visit-mpi: BWPY's VisIt with MPI (only supported on compute nodes!)

Default BLAS: MKL
Running Python

- Use `#!/usr/bin/env python{2,3}` shebang at start of script. Only use “python” if compatible with both 2 and 3!
- Moving to Python 3! **NOW!!!**
  - Python 2 will no longer be updated past 2020
    - 1 year, 6 months, 24 days, 10 hours from now is EoL
  - https://pythonclock.org/
  - Many big packages such as Numpy and Scipy will drop all Python 2 support
  - Do not assume Python 2 will be on LCCF
Building software against BWPY

- Export these variables, so these dirs come after -l/-L
  ```bash
  module switch PrgEnv-cray PrgEnv-gnu
  $ export CPATH="${BWPY_INCLUDE_PATH}"
  $ export LIBRARY_PATH="${BWPY_LIBRARY_PATH}"
  $ export LDFLAGS="${LDLIBRARY_PATH} -Wl,--rpath=${BWPY_LIBRARY_PATH}"
  ```

- Use the CMake from BWPY. It is configured to use /mnt/bwpy/single as its prefix for finding “system” libraries/cmake files

- This must be done in a bwpy-environ execution context
bwpy-environ

- bwpy-environ is a program execution context wrapper necessary to run current versions of BWPY
  - bwpy-environ runs child processes within its context
  - No child processes can modify their parents
    - Each execution of bwpy-environ will reset the environment to that of the parent of bwpy-environ
- bwpy-environ should be run on the outermost layer of work
- Makes changes local to each node
What is program execution context?

- Program execution context is the associated data needed by the program for execution: PIDs, mount points, security capabilities, UID/GID, etc.
- This is important for `bwpy-environ` because it modifies the mount point information in a context private to `bwpy-environ` and its child processes
  - Private mount namespace
- This technique allows for simultaneous mounts of different disk images at the same mountpoint
So what does bwpy-environ do?

- Sets up kernel modules, if necessary
- Creates a private mount namespace
- Mounts the requested bwpy image
- Drops privileges
- Runs the requested program, defaulting to $SHELL
Things to keep in mind

- Environment variables reset to original values between bwpy-environ executions
- Private mountpoints automatically unmount after the last process using them terminates. Running bwpy-environ multiple times causes unnecessary overhead (remounts)
- The Linux kernel will aggressively cache bwpy files while mounted
Running bwpy-environ

- Must be run on every node requiring bwpy

```bash
#!/sw/bw/bwpy/mnt/bin/bash-wrapped
#PBS ...
.
/opt/modules/default/init/bash
module load bwpy
python -c "Hello from MOM node"
module load bwpy-mpi
aprun -n 1 -b bwpy-environ python -c "hello \n from compute node"
aprun -n 1 python -c "hello using wrapper"
```

```bash
#!/bin/bash
#PBS ...
.
/opt/modules/default/init/bash
module load bwpy
bwpy-environ python -c "Hello from MOM node"
python -c "Using wrapper on MOM node"
module load bwpy-mpi
aprun -n 1 -b bwpy-environ python -c "hello \n from compute node"
aprun -n 1 python -c "hello using wrapper"
```
Doing multiple things on compute nodes

- Wrap everything with bwpy-environ

```bash
#!/bin/bash
#PBS ...
.
#opt/modules/default/init/bash
module load bwpy
module load bwpy-mpi
aprun -n 1 -b bwpy-environ ./script.sh
aprun -n 1 ./script2.sh
```

```bash
#!/bin/bash
# script.sh
python -c "print('Hello 1')"
python -c "print('Hello 2')"
```

```bash
#!/sw/bw/bwpy/mnt/bin/bash-wrapped
# script2.sh
python -c "print('Hello 1')"
python -c "print('Hello 2')"
```
Virtualenv

- Virtualenv has been patched to automatically use bwpy-environ
- The wrapping uses the value of $BWPY_VERSION at creation
- An outer bwpy-environ should only be used if executing multiple python commands

```
$ cat venv/bin/python
#!/bin/sh
exec /sw/bw/bwpy/mnt/bin/bwpy-environ -v 2.0.0 -a /mnt/a/u/staff/cmaclean/venv/bin/python3.5 -- \\
/mnt/a/u/staff/cmaclean/venv/bin/python3.5.venv "$@
```
Questions?