Using the Frontier Programming Environment

Matt Belhorn

HPC Engineer
Oak Ridge Leadership Computing Facility (OLCF)
Oak Ridge National Laboratory (ORNL)

February 15, 2023

LMOD Environment Modules

- Often differing software and libraries available cannot coexist simultaneously in your environment
  (`$PATH`, `$LD_LIBRARY_PATH`, etc).
- Build and runtime environment software and libraries
  managed with Lua-based LMOD (https://lmod.readthedocs.io)
- Usage:

```
$ module -t list # list loaded modules
$ module avail   # Show modules that can be loaded given current env
$ module help <package>  # Help info for package (if provided)
$ module show <package>  # Show contents of module
$ module load <package> <package>…   # Add package(s) to environment
$ module unload <package> <package>… # Remove package(s) from environment
$ module reset                       # Restore system defaults
$ module restore <collection>        # Load a saved collection
$ module spider <package>            # Deep search for modules
$ module purge                       # Clear all modules from env.
```

Module Avail

- The `module avail` command shows only what *can* be loaded.
- Accepts full or partial package names to limit output to matches.
- (D)efault and (L)oaded packages are indicated in output with labels.

```
$ module avail
--------------------------- /sw/frontier/modulefiles ---------------------------
   DefApps/default    (L)    forge/22.1.0rc        rocm/5.1.0
   afar/14.0.0_5.0.0         forge/22.1.0          rocm/5.2.0
   afar/15.0.0_5.2.0 (D)     forge/22.1.1    (D)   rocm/5.3.0 (D)
   codee/1.6.0               rocm/4.5.2            rocm/5.4.0

  Where:
   L:  Module is loaded
   D:  Default Module

Use "module spider" to find all possible modules.
Use "module keyword key1 key2 ..." to search for all possible modules matching
any of the "keys".
```

Directory in MODULEPATH where block of modulefiles exists. Printed in order of priority.

Any new future labels will be explained in the legend at the bottom of non-terse output.

Modulefile Priority

- First modulefile among duplicate package/version names in MODULEPATH will be selected:

```
$ module avail conduit
 /sw/frontier/spack-envs/base/modules/spack/cray-sles15-x86_64/cray-mpich/8.1.23-d2badeh/gcc/10.3.0
   conduit/0.7.2    conduit/0.8.2    conduit/0.8.3 (D)
 /sw/frontier/spack-envs/base/modules/spack/cray-sles15-x86_64/cray-mpich/8.1.17-f42wy5g/gcc/10.3.0
   conduit/0.7.2    conduit/0.8.2    conduit/0.8.3
 /sw/frontier/spack-envs/base/modules/spack/cray-sles15-x86_64/cray-mpich/8.1.12-eg2x4ag/gcc/10.3.0
   conduit/0.7.2    conduit/0.8.2
```

- To override behavior, alter the `$MODULEPATH`:
  - `$ module use /path/to/module/file/tree`,
    - given path is *prepended* to `$MODULEPATH` with higher priority.
  - Can also provide path to your own custom modulefiles.

Searching for Modules with Spider

- The `module avail` command shows what *can* be loaded *given the currently loaded modulefiles*.
- Use `module spider` for deep searching what modules are *potentially* available to load

```
$ module -t spider kokkos/3.6.00

----------------------------------------------------------------------------------------
  kokkos: kokkos/3.6.00
----------------------------------------------------------------------------------------

    You will need to load all module(s) on any one of the lines below before the "kokkos/3.6.00" module
    is available to load.

      DefApps/default

    Help:
      Kokkos implements a programming model in C++ for writing performance
      portable applications targeting all major HPC platforms.
```

**The DefApps module interferes with how deep outside of currently-loaded modules `module spider` can search

Spider (cont'd)

- Complete listing of possible modules is *only reported when searching for a specific version:* `module spider <package>/<version>`
- Can search with using limited regular expressions:
  - All modules with 's' in their name: `module -t spider 's'`
  - All modules starting with the letter 's': `module -t -r spider '^su'`

```
$ module -t spider 's'
adios2/2.8.1
adios2/2.8.3
ascent/0.7.1
ascent/0.8.0
bison/3.7.6
bison/3.8.2
boost/1.79.0-cxx17
boost/1.79.0
….
```

```
$ module -t  -r spider '^su'
subversion/1.14.0
subversion/1.14.1
sundials/5.8.0-cpu
sundials/5.8.0
sundials/6.1.1-cpu
sundials/6.1.1
sundials/6.2.0-cpu
superlu-dist/7.1.1-cpu
superlu-dist/7.1.1
superlu-dist/7.2.0-cpu
superlu-dist/7.2.0
….
```

Module Dependency Management

- Conflicting modulefiles are automatically reloaded or inactivated.

```
$ module load PrgEnv-cray

Lmod is automatically replacing "gcc/10.3.0" with "cce/15.0.0".

Lmod is automatically replacing "PrgEnv-gnu/8.3.3" with "PrgEnv-cray/8.3.3".

Due to MODULEPATH changes, the following have been reloaded:
  1) cray-mpich/8.1.23
```

- Generally eliminates needs for `module swap <p1> <p2>`
- Check stderr output for messages about deprecated modules.
- Modules generally only available when all dependencies are currently loaded.

User Collections

- Save collection of modules for easy re-use:

```
$ module save my_favorite_modules
Saved current collection of modules to: "my_favorite_modules", for system: "frontier"

$ module reset
Resetting modules to system default

$ module restore my_favorite_modules
Restoring modules from user's my_favorite_modules, for system: "frontier"

$ module savelist                # Show what collections you've saved
$ module describe <collection>   # Show modules in a collection
$ module disable <collection>    # Make a collection unrestorable (does not delete)
```

- Modulefile updates may break saved collections.
  To fix: manually load desired modules, save to same name to update.
- Delete a collection: `rm ~/.lmod.d/<collection>.<system>`

Compilers and Programming Environments

CPE Programming Environment Module		PrgEnv-cray	PrgEnv-amd	PrgEnv-gnu	–
Host Toolchain Module		cce/15.0.0	amd/5.3.0	gcc/12.2.0 gcc/11.2.0 gcc/10.3.0	User Custom Toolchains
		cce/14.0.2 cce/14.0.1	amd/5.2.0 amd/5.1.0		
			amd/4.5.2		
CPE Compiler Drivers	C: cc	craycc	amdclang	gcc	–
	C++: CC	crayCC	amdclang++	g++	–
	Fortran: ftn	crayftn	amdflang	gfortran	–
ROCm Provider Module		amd-mixed/*	–	amd-mixed/*	rocm

- CPE provides modules `amd` and `amd-mixed` which expose the ROCm toolchain to the environment.
- OLCF provides a `rocm` modulefile for preview versions of ROCm not officially supported by current CPE

ROCm, Host Toolchain and Cray MPICH Compatibility

- cray-mpich GTL (GPU Transport Layer) depends on specific version-matched ROCm runtime libraries
- ROCm runtime libraries depend on specific LLVM runtime libraries
  - Recommended to match LLVM ABI version of LLVM-based host toolchains and ROCm lib runtimes

		ROCm Release				
		5.4.0	**5.3.0**	**5.2.0**	**5.1.0**	**4.5.2**
LLVM ABI		LLVM 15		LLVM 14		LLVM 13
cray-mpich Release Compatibility	8.1.23	✓	✓	✓	✓	
	8.1.17	✓	✓	✓	✓	
	≤ 8.1.14					✓
Host Toolchain+ROCM LLVM ABI Compatibility	cce	cce/15.0.0 +rocm/5.4.0	cce/15.0.0 +amd-mixed/5.3.0	cce/14.0.2 cce/14.0.1 +amd-mixed/5.2.0	cce/14.0.2 cce/14.0.1 +amd-mixed/5.1.0	cce/13.x +amd-mixed/4.5.2
	amd		amd/5.3.0	amd/5.2.0	amd/5.1.0	amd/4.5.2

DefApps Module

- Facility-installed software are available through the module system
  - (such as ECP E4S packages and user software requests)
- Dependent on user’s loaded CPE modules (PrgEnv-* and compiler)
- The DefApps module refreshes which facility-installed packages are available when CPE modules change
- Should be loaded by default, no user action is typically needed
  - Could be removed from your environment if using CPE-provided `cpe/*` modules.
  - Call `module load DefApps` to reset

Building your own Software

- Where to install?
  - NFS filesystem is preferred given it is not purged.
  - Paths in `/ccs/proj/<project>`
- Recommend rebuilding whenever key CPE modules are replaced with new CPE releases:
  - ROCm (`amd` or `amd-mixed` modules), `cray-mpich`, `libfabric`, `cray-pmi`
- Spack (https://spack.readthedocs.io/en/latest/)
  - Spack support for CPE is currently undergoing major changes and feature additions

Thanks For Listening

Should you have questions or comments regarding the Summit programming environment, please let us know by contacting us at `help@olcf.ornl.gov`.

We're happy to help and incorporate your feedback.