Introduction to Modules at CHPC

Anita Orendt
Assistant Director
Research Consulting & Faculty Engagement
anita.orendt@utah.edu

Overview of Talk

- Why Modules
- Where to find information
- How to setup to use modules
- Module basics
- Advanced Modules
- Demonstration

What modules do

- Modules are a way of managing the user's shell environment in an interactive session or a batch job

Why Modules

- Modules lets users dynamically change the environment – including easily adding and removing directories needed for a given task from $PATH etc – without needing to log out and back in
- Easy to switch between version of a package or application – again without having to start a new session
- Useful when packages have conflicts in their environment settings

Module Documentation at CHPC

- https://www.chpc.utah.edu/documentation/software/modules.php
- https://www.chpc.utah.edu/documentation/software/modules-advanced.php
- Video -- https://youtu.be/Cu6C5lNLDAY

We make use of TACC's LMOD

- https://www.tacc.utexas.edu/research-development/tacc-projects/lmod
- LUA based

All accounts automatically use modules –

- This is done via the login scripts CHPC provides all accounts, even if you have older dot files
- CHPC uses modules to set up environments upon login: chpc/1.0

Recommendations & Helpful Hints

- Keep both the cshell and bash versions of provided login scripts in your home directory
  - Standard CHPC provided scripts found in /uufs/chpc.utah.edu/sys/modulefiles/templates
- **DO NOT** make changes in the .tcshrc and .bashrc
- Use the .custom.csh/.custom.sh to load modules for programs regularly used in ssh sessions
- Use .aliases file to create aliases – but do not set other environment variables in this file; if this file exists it will be sourced during login
- Module spider (more later) is easiest way to search for modules

Basic Module commands

- **module** - shows the list of module commands
- **module load <name>** - loads module name (shortcut: **ml <name>**)
- **module unload <name>** - unloads module name (**ml -<name>**)
- **module avail** - shows a list of "available" modules (**ml av**)
- **module list** - shows a list of loaded modules (**ml**)
- **module help** - prints help for the module command
- **module help <name>** - prints help for module
- **module show <name>** - prints the module file
- **module purge** - unload all modules
- **module reset system** – resets to system default (only chpc module loaded)
- **module swap <name1> <name2>** - swaps between two modules
- **module spider <string> -** shows all modules that have string in name

CHPC Module Organization

- Core
  - Contains modules for applications independent of both the compiler and MPI implementation
- Compiler
  - Contains modules for applications dependent on a compiler (& version) but not on a MPI implementation
- MPI
  - Contains modules for applications dependent on both a compiler and a MPI implementation

Modules themselves are named by application name/version

Other Information

- We also define module “families”; can only have one module in a family loaded at one time
  - Used for python, compilers, mpi, R
  - For example, if you have intel loaded, and load any gcc it will unload intel
- Parallel versions of boost, HDF5 have separate modules
  - hdf5 for module for serial build, phdf5 for module for parallel build
  - boost for serial, pboost for parallel

Default, aliases, and hidden modules

- For some applications have a default module – one that is installed if you do not provide a specific version
  - typically the latest version is specified to be the default
- For some modules, especially those with long version names, there is also an alias defined
  - **ml intel** loads the default intel-oneapi-compilers/2021.4.0
  - **ml intel/18.0** loads the 2018.1.163 version
- We have depreciated older installations and their modules so some modules have been hidden
  - **module --show_hidden avail**

Rocky8 changes

- For intel compilers
  - intel-oneapi-compliers
- For intel mpi
  - intel-oneapi-mpi
- For intel mkl
  - intel-oneapi-mkl
- For netcdf
  - netcdf-c, netcdf-cxx, netcdf-fortran
- For pgi – now nvhpc
- For gcc – default version is gcc/8.5.0

Rocky8 changes

- For python
  - No /usr/bin/python, instead there is /usr/bin/python2 (2.7.18), /usr/bin/python3 (3.6.8)
    - For these we have created modules python/2.7.18 and python/3.6.8 so that these can be used with 'python'
  - CHPC installed (3.10.3, use ml python)
- For R
  - 4.1.3
  - Can still use the containerized builds of R
    - R/4.1.2-basic
    - R/4.1.2-bioconductor
    - R/4.1.2-geospatial

Module avail command

- **module avail** shows all modules available based on already loaded module
  - This also marks default (D), already loaded (L), gpu specific (g) and aliases
- Some modules are dependent on other modules based on organization
  - these modules are not listed unless the modules they depend on are loaded

Module show command

- Format **module show modulename/version**
- Shows you the content of the module file
- This is useful if there is information on running the program included in the module
- Only works if module is available, i.e., you have modules that it depends on loaded

Module spider command

- **module spider** shows all modules, including modules that aren't available
- Use **module spider <string>** to see a subset of modules with **string** in name, and how to either load the module or how to get more detailed information on how to load

Advanced Modules

- Users can create “save lists” for commonly needed environments
- Users can write and use their own modules, creating modules for their own installations
- Contact CHPC if you need assistance doing this

Getting Help

- CHPC website
  - www.chpc.utah.edu
    - Getting started guide, cluster usage guides, software manual pages, CHPC policies
- Service-Now issue/incident tracking system
  - Email: helpdesk@chpc.utah.edu
- Help Desk: 405 INSCC, 581-6440 (9-6 M-F)
- We use chpc-hpc-users@lists.utah.edu for sending messages to users