Introduction to Modules at CHPC

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

1 February 2018
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 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 log out and back in

• Useful when need to regularly use packages that 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.php)
- [https://www.chpc.utah.edu/documentation/software/modules-advanced.php](https://www.chpc.utah.edu/documentation/software/modules-advanced.php)
- Video -- [https://www.youtube.com/watch?v=Cu6C5INLDAY](https://www.youtube.com/watch?v=Cu6C5INLDAY)

We make use of TACC’s LMOD

- [https://www.tacc.utexas.edu/research-development/tacc-projects/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
Moving to use ONLY modules

- MAKE A COPY OF YOUR OLD ~/.bashrc AND ~/.tcshrc FIRST – especially if you have customizations!

- Copy the CHPC bashrc and tcshrc to your home directory
  
  ```
  cp /uufs/chpc.utah.edu/sys/modulefiles/templates/bashrc ~/.bashrc
  cp /uufs/chpc.utah.edu/sys/modulefiles/templates/tcshrc ~/.tcshrc
  ```

- Also copy
  
  ```
  cp /uufs/chpc.utah.edu/sys/modulefiles/templates/custom.sh ~/.custom.sh
  cp /uufs/chpc.utah.edu/sys/modulefiles/templates/custom.csh ~/.custom.csh
  ```

These allow you to customize your shell environment so that you do not have to load in modules that you always use every time
Recommendations & Helpful Hints

• Keep both the cshell and bash versions in your home directory
• DO NOT make changes in the .tcshrc and .bashrc
• Use the .custom.csh/.custom.sh to load modules for programs you want access to 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
• The software database mentions which installations have modules – if there is one you would like us to create, let us know!
Basic Module commands

- **module** - shows the list of module commands
- **module load <name>** - loads a module (shortcut: ml <name>)
- **module unload <name>** - unloads a module (or ml -<name>)
- **module avail** - shows a list of "available" modules
- **module list** - shows a list of loaded modules (also ml)
- **module help <name>** - prints help for a module
- **module show <name>** - prints the module file
- **module purge** - unload all modules
- **module swap <name1> <name2>** - swaps between two modules
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
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/17` loads the default 2017 intel
  - `ml intel/17.0` loads the 2017.0.098 version

- With move to CentOS7 we have depreciated older installations and their modules so some have been hidden
  - `module --show_hidden avail`
Module avail command

• "module avail" shows all modules available based on already loaded module
  – This also marks default (D), already loaded (L), 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 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
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
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 (9-5 M-F)

• We use chpc-hpc-users@lists.utah.edu for sending messages to users; also have Twitter accounts for announcements -- @CHPCOutages & @CHPCUpdates