TEXAS ADVANCED COMPUTING CENTER

WWW.TACC.UTEXAS.EDU

TEXAS
The University of Texas at Austin

Using Personal Modules and Inherit() w/ the Software Hierarchy

Robert McLay

March 7, 2023

Outline

- How to correctly use personal modules so that Lmod will find and use them.
- How to setup a personal library/application in the software hierarchy
- Why this is a PITA!

Creating Personal modules

- What is the big deal?
- Easy: Create a directory (say $HOME/my_modules)
- Create a directory ($HOME/my_modules/acme)
- Create a create modulefile: $HOME/my_modules/acme/3.2.lua)
- `$ module use $HOME/my_modules`
- `$ module load acme/3.2`
- Easy right!?

Testing a personal copy of a system module

- Suppose that acme/3.2 is already on your system
- And acme/3.2 is a marked default
- The command `module load acme/3.2`
- Will load the system one and not yours
- Even though $HOME/my_modules is listed first in $MODULEPATH
- Why?

Why?

- While Lmod does look in $MODULEPATH order
- So the first module found is usually picked.
- *However*, Marked defaults ALWAYS win in N/V module layouts (Best found)
- Note not in N/V/V layouts. (First found)
- A marked default is where there is either:
  1. A default symlink
  2. .modulerc.lua
  3. .modulerc
  4. .version
- I recently updated the documentation https://lmod.readthedocs.io/en/latest/060_locating.html to explain this

Getting Around a System Marked Defaults

- Make your own marked default.
- Easiest way is to make a default symlink

```
$ cd $HOME/my_modules/acme
$ ln -s 3.2.lua default
```

Checking with module avail

```
---------- /home/user/my_modules -----------
acme/3.2 (D)

---------- /opt/apps/modulefiles ------------
StdEnv    acme/3.2
```

- Make sure that the (D) is next to your acme module
- And not the system one.

Bigger issue: Testing a compiler dependent `boost/1.85.0`

- And you want it part of the software hierarchy
- How can you do this `without` modifying the system modulefiles?
- In particular you only want the correct version of boost available when you load the correct compiler.

The short answer: inherit()

- You can use the inherit() function to simplify this a little
- This is discussed in detail in https://lmod.readthedocs.io/en/latest/340_inherit.html

Overview

- We want to test/use boost install from our own account.
- And have it load when the “right” compiler is loaded
- This assumes that your site is using the software hierarchy
- How can we get the system compiler to load our directory into $MODULEPATH?
- Suppose we want to test a boost version with the intel 19.1 and gcc 12.2 compilers

- Build boost 1.85.0 with gcc 12.2 $\Rightarrow$ ~/pkg/gcc-12/boost/1.85.0
- Build boost 1.85.0 with intel 19.1 $\Rightarrow$ ~/pkg/intel-19/boost/1.85.0

Choice 1: Copy compiler modules into your account

- Easy to do
- Add your directory into $MODULEPATH
- Problem: you are now responsible to keep your copy up-to-date

Choice 2: Use inherit()

- Create your own compiler module and inherit from the system one.
- The inherit() function take NO arguments
- Lmod looks for the exact same name in $MODULEPATH
- This way it includes the system one with your changes.

Conclusions

- Two ways to check for modulefile syntax errors.
- One for building modulefiles
- Another for checking site module tree.

Future Topics

- Unknown at the moment.
- Next Meeting will be March 7th at 9:30 Central (15:30 UTC)