Lmod

Robert McLay

The Texas Advanced Computing Center

Nov. 11, 2015

TACC

THE UNIVERSITY OF TEXAS AT AUSTIN

Lmod

- Introduction: Remind everyone about Lmod
- Lmod: Our Env. Module System

Why invent Y.A. Module System?

- Many thing right with the Original.
- But it was designed before multi-compilers/MPI
- Sites make it work but with herculean efforts
- Lmod can make this work easily.

Why You Might Want To Switch

- Active Development; Frequent Releases; Bug fixes.
- Vibrant Community
- It is used from Norway to Isreal to New Zealand from Stanford to MIT to NASA
- Enjoy many capabilities w/o changing a single module file
- Debian and Fedora packages available
- Many more advantages when you're ready

Features requiring no changes to modulefiles

- Reads TCL modulefiles directly (Cray modules supported)
- User default and named collections of modules
- Module cache system: Faster avail, spider, etc
- Tracking module usage
- A few edge cases where Env. Modules and Lmod differ

Features of Lmod with small changes to modulefiles

- Family function: Prevent users from loading two compilers at the same time (experts can override)
- Properties: (MIC-aware, Beta, etc)
- Sticky modules
- ...

Lmod supports a software hierarchy

- Lmod supports flat layout of modules
- Some really cool features if you have a software hierarchy
  - Protecting users from mismatched modules
  - Auto Swapping of Compiler and MPI dependent modules
- When you are ready, it will be there

A Transition Strategy

- Install Lua and Lmod in your account
- Staff & Friendly Opt-in Testing
- Deploy to your users with an Opt-out choice
- Some users can run TCL/C modules (a.k.a. Tmod)
- Others can run Lmod
- No single user can run both at the same time!
- Transistion doc: lmod.readthedocs.org

Lmod Features

- Support for a Hierarchical Module layout
- Module spider: find all modules
- Caching system for rapid avail and spider
- Support for Properties
- Module collections, output to stdout, proper version sorting
- Reads TCL modulefiles directly (Cray modules supported)
- And so much more...

Lmod Documentation

- lmod.readthedocs.org
- Beginning Topics: User, FAQ
- Intermediate Topics: Transitioning to Lmod, Installing Lmod, Software Hierarchy
- Advanced Topics: Generic Modules, Deprecating Modules,
- How to install Lmod on a Shared File system.
- And much more.

Lmod handling of Cray modules

- NO DUPS: Lmod now correctly handles no duplicated in a PATH-like variable.
- TMOD_RULE: if an entry is in a path do not replace it.
- GNU4.8_LIB: Prevent Lmod from generating variables with a '.' in it.

Injecting the Software Hierarchy into Cray Modules

- Goal: Create the Software Hierarchy where there is none.
- Create the following module /opt/apps/modulefiles/PrgEnv-intel/5.2.40.lua

```
local name = myModuleName():gsub("PrgEnv%-","")
local mpath = pathJoin("/opt/apps",name,
              myModuleVersion(),"modulefiles")
inherit()
prepend_path("MODULEPATH",mpath)
family("MPI_COMPILER")
```

- The `inherit()` function will load /opt/cray/PrgEnv-intel/5.2.40
- The directory /opt/apps/intel/5.2.40/modulefiles is prepend to MODULEPATH

Lmod and the Pager

- Cray and Mac OS default the pager to less (not more)
- By default you have to type "q" to exit less
- Lmod now uses `LMOD_PAGER` as `less`
- `LMOD_PAGER_OPTS` as `-XqMREF`
- This will give consistant behavior across systems.

Possible change to Lmod: Do not reload

- Tmod doesn't reload a module if it is already loaded.
- Lmod always reloads
- Reloads the right thing to do in hierarchies.
- Not reloading might help sites transitioning from Tmod.
- This might help with Kenneth's problem. (;->)

Invisible Modules

- Tmod and Lmod have always supported hidden modules.
- Hidden modules are module names that start with a leading "." in the version.
- Module names with dot are ignored completely.
- They do not show up with avail or spider but can be loaded and will be listed.
- This is a way to provide a module for testing without making it publicly available
- Invisible would be new and this hackathon would be a place to discuss what this means.

Invisible Modules: Motivation

- Cray have architecture modules that aren't needed: craype-barcelona, ...
- Sites may wish install real module names and hide them for testing
- But what does this mean?

What are the rules?

- Can users load an invisible module? Yes
- Can you make all versions of a module invisible? Yes
- Can you make a single version invisible? Yes
- Can an invisible module be the default? No
- How will sites mark modules as invisible?
- Yet another Lua table? Combine with lmodrc.lua?

Default Handling (I)

- The current default handling works well but ...
- It doesn't work with ~/.modulerc when users set their own default.
- It won't work with invisible modules.

Default handling (II)

- Selection of what the default module is already complicated.
- Especially when there are multiple directories in MODULEPATH.
- Lmod Rules for picking a default:
  - The first marked default is chosen in MODULEPATH order
  - The Highest version is chosen in MODULEPATH order.

Default handling (III)

- Lmod current thinks it knows what the defaults are by walking the directory trees.
- The spider cache will have to be built based on walking the tree.
- But there will have to enough information to compute the default in light of ~/.modulerc and invisible modules.

Conclusions

- Do not reload option.
- Invisible modules and what does this mean
- Default Handling