Lmod: A Modern Replacement for Environment Modules

Robert McLay

The Texas Advanced Computing Center

February 8, 2017

Introduction

- What is Lmod?
- Why you might to use it?
- What's the difference between Tmod and Lmod
- What is new with Lmod?
- How could Lmod possibly work?
- Conclusions

What is Lmod?

- A modern replacement for a tried and true concept.
- The guiding principal: "Make life easier w/o getting in the way."
- Sites use Modules to communicate w/ Users.

Why You Might Want To Use Lmod?

- Same `module` command as in Tmod
- 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, Fedora and Brew packages available
- Many more advantages when you're ready
- It is what we and many sites around the world use every day!

Features

- Reads for TCL and Lua modulefiles
- One name rule.
- Support a Software Hierarchy
- Fast `module avail` via optional spider cache
- Properties (gpu, mic)
- Semantic Versioning: 5.6 is older than 5.10
- family("compiler") family("mpi") support
- Optional Tracking: What modules are used?
- Many other features: ml, collections, hooks, nag, ...

Tmod vs. Lmod

- Tmod is in maintenance mode, Lmod active
- Lmod has many more features
- Tmod: `module load gcc/5.3 gcc/6.0` works
- Lmod has the "One Name Rule"
- Lmod close to Tmod, but not the same.

What is new with Lmod 7?

- Support for Name-Version-Version
- Support for Hidden Modules
- Support for Translations

History of Support for Module Names

- Originally only *name/version*: gcc/4.8.1
- Lmod 5+ *cat/name/version*: compiler/gcc/4.8.1

New with Lmod 7: NVV

- Support for *name/v1/v2*: fftw/64/3.3.4
- MODULERC Support:
  - Set Defaults under Site and/or User
  - Hide any installed module
- Major refactoring of Lmod
  - support NVV
  - Code Cleanup
  - Better Spider Cache handling

Setting Defaults

- System MODULERC file: `/path/to/lmod/etc/rc`
- `$MODULERC` points to a file.
- User `~/.modulerc`
- Can set defaults User, System, Files
- Examples: account for web services

Hiding Modules

- System MODULERC file: `/path/to/lmod/etc/rc`
- User `~/.modulerc`
- `hide-version foo/1.2.3`
- Hidden from avail, spider and keyword
- Hidden modules can be loaded
- Sites: deprecation, experimental
- show hidden: `module --show-hidden avail`
- Hidden modules are marked and displayed dim

Why does Lmod work at all?

- The Environment is inherited from the parent process
- Changes in the child's enviroment DOES NOT affect the parent.
- So how could Lmod work at all?

The trick is

- The `lmod` program generates text.
- The module command eval's that text.

Why is this important?

- It's a useful trick to know
- Debugging Modulefiles:
- `$LMOD_CMD bash load` *module* `2> /dev/null > stdout.txt`

Debugging Lmod

- `module --config` : reports Lmod configuration
- `module -D load foo > load.log`

Conclusions: Lmod 7+

- Latest version: https://github.com:TACC/Lmod.git
- Stable version: http://lmod.sf.net
- Documentation: http://lmod.readthedocs.org