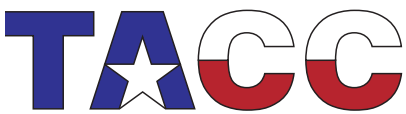
SC17: 7th Annual Lmod Booth Talk

Robert McLay

The Texas Advanced Computing Center

November 15, 2017



Introduction

- Welcome to the 7th annual TACC Booth Talk
- What is Lmod?
- Why you might to use it
- What is new?

Lmod's Big Ideas

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

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 and Fedora packages available
- Many more advantages when you're ready
- It is what we use every day!

Features

- Reads for TCL and Lua modulefiles
- One name rule.
- Support Software Hierarchy
- Spider Cache: fast `$ module avail`
- 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, ...

History of Support for Module Names

- Originally only *name/version*: gcc/4.8.1
- Lmod 5+ *cat/name/version*: compiler/gcc/4.8.1
- Lmod 7+ *name/version/version*: intel/impi/64/18.0.1

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`

New Features since SC 16

- New module function: `depends_on()`
- Reference counting on PATH like variables
- French, German, Spanish translations for Lmod messages.
- Admin list (AKA Nag List) supports Lua Regex for matching
- Improved Settarg (more on this later?)

depends_on()

- Modules X and Y depends on Module A
- ml purge; ml X; ml unload X; $\Rightarrow$ unload A
- ml purge; ml X Y; ml unload X; $\Rightarrow$ keep A
- ml purge; ml X Y; ml unload X Y ; $\Rightarrow$ unload A
- ml purge; ml A X Y; ml unload X Y ; $\Rightarrow$ keep A

Reference Counting for PATH like variables

- AKA: the /usr/local/bin problem
- Old:
  - Default path has /usr/local/bin
  - Module A also has /usr/local/bin
  - Unloading module A removes /usr/local/bin from path
- New: With Ref. Count the problem is fixed.

Future Work (I): Module Export

- Module Collections are for individuals.
- They are not meant to be shared between users
- To share I plan to add "module export"

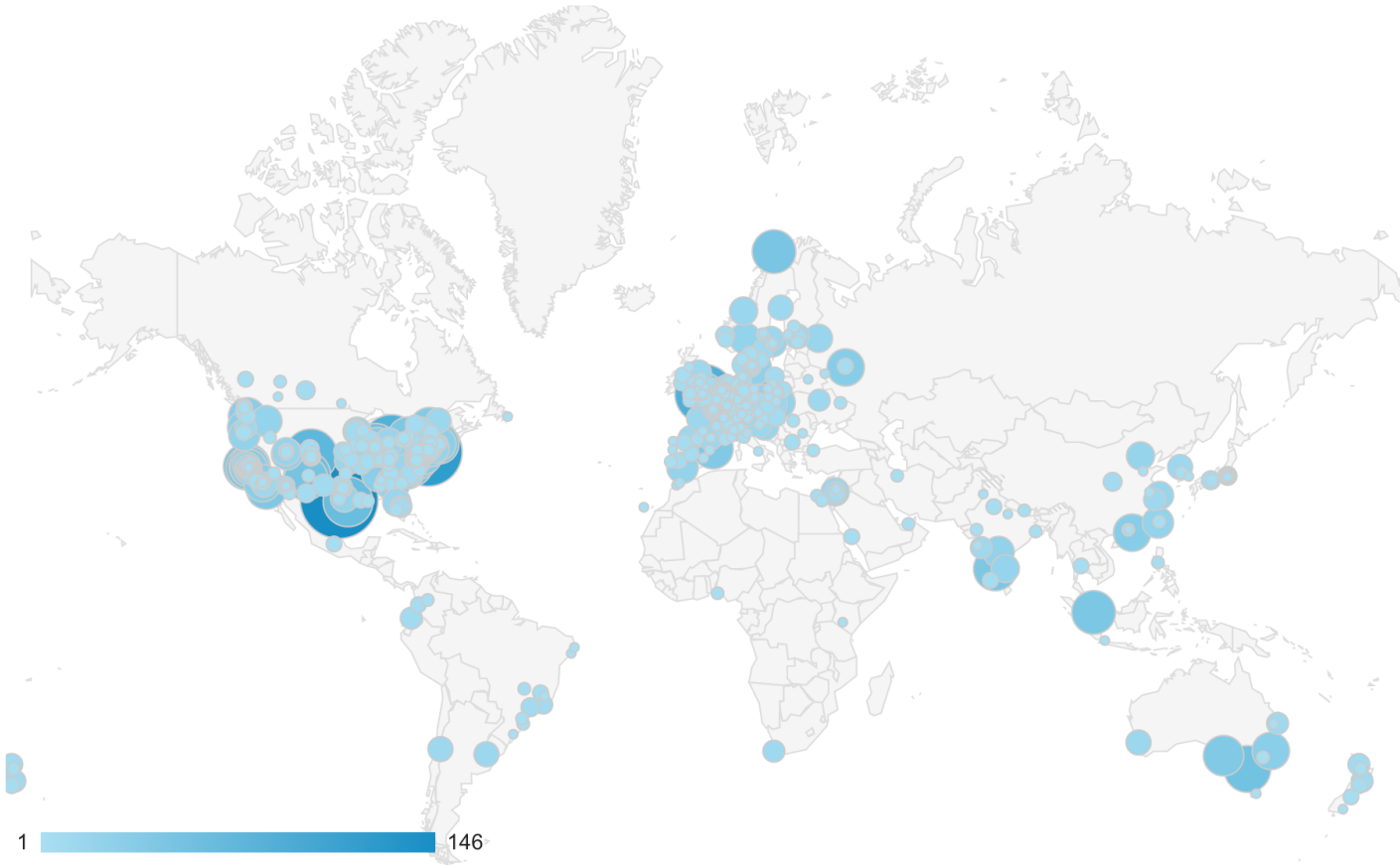
Module Export

```
$ module export <collection> 2> export.txt
$ cat export.txt

module purge
clearMT
export MODULEPATH=/path1:/path2

module collection_load X Y Z
module --ref_count 2 depend_on A
```

Future Work (II): MODULEPATH ref counting

- A user has requested the MODULEPATH have ref-counting
- `ml unuse /path/to/modules` would always remove directory from MODULEPATH

Lmod Doc usage



Conclusions: Lmod 7+

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