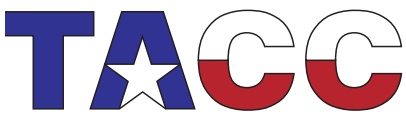
Lmod BoF at SC'13

Robert McLay

The Texas Advanced Computing Center

November 20, 2012



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."

Modern Replacement

- Active Development.
- Vibrant Community.
- Robust Regression Testing.
- Great control over defaults, versions and dependencies.
- Work great w/ or w/o a software hierarchy.

Making life easier

- You want to save an empty collection, Lmod will let you but with warning.
- You want multiple compilers, you can if you are an expert.

New Features

- Meta-modules can be saved even when they include `setenv` and `prepend_path` commands (Thx to Maxime Boissonneault)
- Module properties.
- Modulefiles are now read through a "sandbox".
- Sticky modules: unloaded or purged with "–force" flag.
- Improvements when searching for default modules.
- Allows duplicate paths.
- New function "`pushenv`"
- hooks: load, msgHook, etc.
- tab completion for Bash and Zsh for both "`module`" and "`ml`".

Category/name/version

- Lmod now supports Category/name/version.
- Module can be named "`bio/bowtie/1.7`"
- Nesting can be as complex as you like.
- The one "name" rule works here.
- Meta-module support generalized:
  - Any module file in a directory that contains a sub-directory is a meta-module (no version).
  - Version files are in a directory with no sub-directories.

Load and prereq modify functions

- `load(atleast("A","1.4"))`
- `load(between("B","1.4","2.7"))`
- `load(latest("C"))`
- `prereq(atleast("D","1.4"))`
- `prereq(between("E","1.4","2.7"))`
- `prereq(latest("F"))`
- Typically marked default then latest.

settarg

- This module provides safety, flexibility and repeatability in a dynamic environment.
- It encapsulates the state of the modules loaded and dynamically updates the state when modules change:

```
$  env | grep '^TARG'
TARG_BUILD_SCENARIO=dbg
TARG=OBJ/_x86_64_dbg_gcc-4.6_mpich-3.0
TARG_MPI_FAMILY=mpich
TARG_MPI=mpich-3.0
$ module swap mpich openmpi; opt; env | grep '^TARG'
TARG_BUILD_SCENARIO=opt
TARG=OBJ/_x86_64_opt_gcc-4.6_openmpi-1.6
TARG_MPI=openmpi-1.6
TARG_MPI_FAMILY=openmpi
```

settarg (II)

- Typically TARG is OBJ/$ARCH_$SCENARIO_$CMPLR_$MPI
- `TARG=OBJ/_x86_64_dbg_gcc-4.6_mpich-3.0`
- User can extend this with user level or directory level specialization.
- `OBJ/_x86_64_dbg_intel-14.0_mpich-3.0_petsc-3.4`
- A makefile can modified to write generated file into $TARG.
- Never need to "`make clobber`" when switching scenario, compiler, etc.

Questions for users of Lmod

- How big are your systems?
- Any of you use a Lustre based home file system?
- Biggest headaches with module systems or Lmod:
  - Trouble dealing with staff?
  - Trouble converting users?
- Software hierarchy vs prereq/conflict?
- Best story/worst story in dealing with modules/Lmod?

Topics to Discuss

- Getting Bash to work right.
- Leveraging Lmod to know what software your users are using.
- Using modulefiles and a package manager (e.g. rpm)
- Using SitePackage.
- Feature requests.

For more information

- Introduction to Lmod talk at lmod.sf.net.
- Download source from: lmod.sourceforge.net (lmod.sf.net)
- lmod-users@lists.sourceforge.net
- Documentation: www.tacc.utexas.edu/tacc-projects/lmod
- Best Practices Paper from SC'11

Future Work

- Improve and update documentation.
- Move new features from README into web documentation.
- Theory and Practice of Modules and Site Management.

Previous Features

- Case-insensitive searching with "module spider"
- spider, keyword, avail and help now use `$PAGER`
- A standard way to support site packages.
- `getdefault/setdefault` becoming `save/restore`
- Module files can have properties: MIC, Apps/Lib, ...
- module wrapper command for the typing challenged: "ml"
- Version Parsing: 5.6 is now older than 5.10
- Mailing list: lmod-users@lists.sourceforge.net

Save / Restore

- `getdefault/setdefault` becoming `save/restore`
- `module restore` load user's default or system default when user's default doesn't exist
- `reset` and `getdefault/setdefault` will be deprecated.

Module Properties

- TACC is deploying Stampede with MIC accelerators
- Some modules will be "MIC" aware: mkl, fftw3, phdf5, ...
- Lmod will decorate these modules:

```
1) unix/unix     3) ddt/ddt       5) mpich2/1.5    7) phdf5/1.8.9 (m)
2) intel/13.0    4) mkl/mkl (*)   6) petsc/3.2     8) PrgEnv

Where:
 (m):  module is build natively for MIC
 (*):  module is build natively for MIC and offload to the MIC

 ------
 add_property("arch","mic")              -- > phdf5
 add_property("arch","mic:offload")      -- > mkl
```

- What properties would you like to support?

For those who can't type: "ml"

- `ml` is a wrapper:
  - With no argument: `ml` means `module list`
  - With a module name: `ml foo` means `module load foo`.
  - With a module command: `ml spider` means `module spider`.
- See `ml --help` for more documentation.

Module version sorting

- Old way lexicographically sort: 5.6 is newer that 5.10
- New way 5.10 is newer than 5.6
- Old to new: 2.4dev1, 2.4a1, 2.4rc2, 2.4, 2.4-1, 2.4.1
- Same: 2.4-1, 2.4p1, 2.4-p1
- Old to new: 3.2-shared, 3.2

Projects

- Fast directory tree walker for lustre. (similar functionality to luafilesystem) (Anyone interested?)
- Allow users with personal modules use system caches (R. McLay)
- Produce Json output of entire module system to populate system software web-pages (R. McLay - just completed)
- Support for options after commands: `module keyword --prop mic`
- Other support for properties: searching?,
- What happens when a site wants two or more sets of properties?

.lmodrc.lua

```
propT = {
   arch = {
      validT = { mic = 1, offload = 1, gpu = 1, },
      displayT = {
         ["mic:offload"]     = { short = "(*)",  color = "red",     doc = "...",},
         ["mic"]             = { short = "(m)",  color = "blue",    doc = "...",},
         ["offload"]         = { short = "(o)",  color = "blue",    doc = "...",},
         ["gpu"]             = { short = "(g)",  color = "magenta", doc = "...",},
         ["gpu:mic"]         = { short = "(gm)", color = "magenta", doc = "...",},
         ["gpu:mic:offload"] = { short = "(@)",  color = "magenta", doc = "...",},
      },
   },
}
```

Getting Bash to work right

- At TACC we rebuild bash so that it reads /etc/bashrc on interactive shells.
- It also reads `/etc/bash_logout` on logout.
- We patch config-top.h to change bash behavior.
- Ubuntu does the same, Red Hat does not.

SitePackage

- Lmod now has a standard way to include site-specific functions
- See Contrib/SitePackage for details

Track software usage

- Track software usage via syslog or logout data.
- Lmod can build a reverse map: directories to modules