



## Why I like Lmod and why you should too!

Kenneth Hoste  
HPC-UGent, Ghent University, Belgium  
[kenneth.hoste@ugent.be](mailto:kenneth.hoste@ugent.be)

[http://users.ugent.be/~kehoste/Why\\_I\\_like\\_Lmod\\_20160128.pdf](http://users.ugent.be/~kehoste/Why_I_like_Lmod_20160128.pdf)

January 28th 2016 – 1st EasyBuild User Meeting – Ghent (Belgium)

<https://github.com/hpcugent/easybuild/wiki/1st-EasyBuild-User-Meeting>

# How I learned about Lmod



<http://hpcugent.github.io/easybuild/>

- makes it very easy to install **a lot** of software/modules quickly
- we started wondering how we could organise our modules tree better
- Lmod and the module hierarchy idea allow to deal with this

And then we discovered a whole bunch of other interesting features...

# Lmod: a modern modules tool

<https://tacc.utexas.edu/research-development/tacc-projects/lmod>

- developed by Dr. Robert McLay (TACC, UT Austin)
- created to properly support module hierarchies
- available since Oct'08, *actively developed*, frequent stable releases
- well documented: <http://lmod.readthedocs.org>
- drop-in alternative for Tcl-based module tools (a few edge cases)
- written in Lua, consumes module files in both Tcl and Lua syntax
- (vastly) improves user experience, without hindering experts
- highly community-driven development

# Lmod: feature highlights

- module hierarchy-aware design and functionality
  - searching across entire module tree with 'module spider'
  - automatic reloading of dependent modules on 'module swap'
  - marking missing dependent modules as inactive after 'module swap'
- caching of module files, for responsive subcommands (e.g., avail)
- site-customizable behavior via provided hooks
- ml command ('ml' is 'module list', 'ml GCC' is 'module load GCC', ...)
- load/unload shortcuts via + and -
- various other useful/advanced features, including:
  - case-insensitive 'avail' subcommand
  - can send subcommand output to stdout (rather than to stderr)
  - defining module families (e.g., 'compiler', 'mpi')
  - assigning properties to modules (e.g., 'Phi-aware')
  - stack-based definition of environment variables (using pushenv)
  - user-definable module collections (module save/module restore)
  - and a lot more ...

# Example

Behold: the power of the Lmod command line using 'ml' command:

- see which modules are loaded ('module list')
- change to different part of module hierarchy by swapping compilers
- recheck which modules are loaded

```
$ ml
```

```
Currently loaded modules:
```

```
1) GCC/4.8.2    2) MPICH/3.1.1    3) FFTW/3.3.2
```

```
$ ml -GCC Clang
```

```
The following have been reloaded:
```

```
1) FFTW/3.3.2    2) MPICH/3.1.1
```

```
$ ml
```

```
Currently loaded modules:
```

```
1) Clang/3.4     2) MPICH/3.1.1    3) FFTW/3.3.2
```

# HUST-14 paper

## Modern Scientific Software Management Using EasyBuild and Lmod

*Markus Geimer (JSC)*

*Kenneth Hoste (HPC-UGent)*

*Robert McLay (TACC)*

[http://hpcugent.github.io/easybuild/files/hust14\\_paper.pdf](http://hpcugent.github.io/easybuild/files/hust14_paper.pdf)

- paper at HUST-14 workshop (during SC14)
- explains basics of module tools, EasyBuild and Lmod
- highlights issues with current approaches in software installation
- advocates use of a hierarchical module naming scheme
- presents EasyBuild and Lmod as adequate tools for software management



## Why I like Lmod and why you should too!

Kenneth Hoste  
HPC-UGent, Ghent University, Belgium  
kenneth.hoste@ugent.be

[http://users.ugent.be/~kehoste/Why\\_I\\_like\\_Lmod\\_20160128.pdf](http://users.ugent.be/~kehoste/Why_I_like_Lmod_20160128.pdf)

January 28th 2016 – 1st EasyBuild User Meeting – Ghent (Belgium)

<https://github.com/hpcugent/easybuild/wiki/1st-EasyBuild-User-Meeting>