

LUMI

A white wolf is the central focus, standing in a futuristic, blue-toned digital environment. The background is filled with vertical light beams, floating particles, and a grid-like structure, creating a high-tech, data-driven atmosphere. The wolf is looking slightly to the right of the camera.

Environment modules on LUMI

Kurt Lust
LUMI User Support Team (LUST)
University of Antwerp

April 2026

Environment modules implementations

L U M I

- Environment modules are used on HPC systems to enable users to create custom environments and select between multiple versions of applications
 - Applications on HPC systems are installed in non-standard places
 - Can set environment variables for the application also
- 3 systems in use
 - Original module tool written in C with modules in Tcl, development halted
 - New implementation in Tcl with many new features, developed at CEA
 - Not supported by HPE Cray
 - Lmod, an implementation in Lua with native module files in Lua but support for most Tcl module files
- We chose Lmod for LUMI

Exploring modules with Lmod

- Contrary to some other module systems, not all modules are immediately available for loading
 - **Installed modules**: All modules on the system that can be loaded one way or another
 - **Available modules**: Can be loaded without first loading another module
- Examples in the HPE Cray PE:
 - `cray-mpich` requires a compiler module and network target module first
 - Many of the performance monitoring tools require `perftools-base` first
 - `cray-fftw` only becomes available when a processor target module is loaded
- Tools
 - `module avail` searches in the available modules
 - `module spider` and `module keyword` search in the installed modules
 - But with some restrictions on LUMI

Benefits of a hierarchy

- When well designed, you get some protection from loading modules that do not work together well
 - Only partially implemented on LUMI
- When “swapping” a module that makes other modules available with a different one, Lmod will try to look for equivalent modules in the new hierarchy
 - Example: Try `module load PrgEnv-gnu` in the default login environment and see what happens

kulust@uan03.lumi.csc - ~

~#2

kulust@uan03.lumi.csc - ~ (ssh) #1

Did you know?

With software built with EasyBuild (as is most software in the LUMI stack), it is very easy to find the location of the software binaries. After loading the module, an environment variable based on the name of the module will be defined. E.g., after loading one of the Boost modules, the location of Boost will be given by the EBR00TBOOST environment variable. Try

```
env | egrep ^EBR00T
```

in a shell to see all EBR00T variables that are defined after loading some modules.

```
[lumi][kulust@uan03-1001 ~]$ module load PrgEnv-gnu
```

Lmod is automatically replacing "cce/19.0.0" with "gcc-native/14.2".

Lmod is automatically replacing "PrgEnv-cray/8.6.0" with "PrgEnv-gnu/8.6.0".

Due to MODULEPATH changes, the following have been reloaded:

- 1) cray-libsci/25.03.0
- 2) cray-mpich/8.1.32

```
[lumi][kulust@uan03-1002 ~]$
```

Module names and families

- In Lmod you cannot have two modules with the same name loaded together
 - On LUMI, when loading a new module the other one with the same name will be automatically unloaded
 - Automatic protection from conflicts
- Extension: **family concept**: No two modules of the same family can be loaded together
 - E.g., make compilers member of the family “compiler”
 - On LUMI, the conflicting module of the same family will be unloaded automatically

Extensions

- It would not make sense to have a separate module for each of the hundreds of R packages or tens of Python packages a software stack may contain.
 - Would actually also create a performance problem due to excess metadata access and long PATH variables
 - Bundle related packages in a single module
- Lmod solution: A module can define a list of [extensions](#), basically other packages provided by the module.
 - And the regular commands can be used to search for these
 - Unfortunately not used in the HPE Cray PE `cray-python` and `cray-R` modules

module spider

- `module spider` : Long list of all installed software with short description
 - Will also look into modules for “extensions” and show those also, marked with an “E”
 - By default only in the main software stacks on LUMI

module spider (command) (1)

```
kulust@uan01.lumi.csc --
kulust@uan01.lumi.csc -- (ssh) #1
| works. Spack is offered as-is. We do not do development or
*| bug fixing in Spack but do offer a configuration compatible
**| with the Cray PE.
** `-----*****-----***-----*****-----`

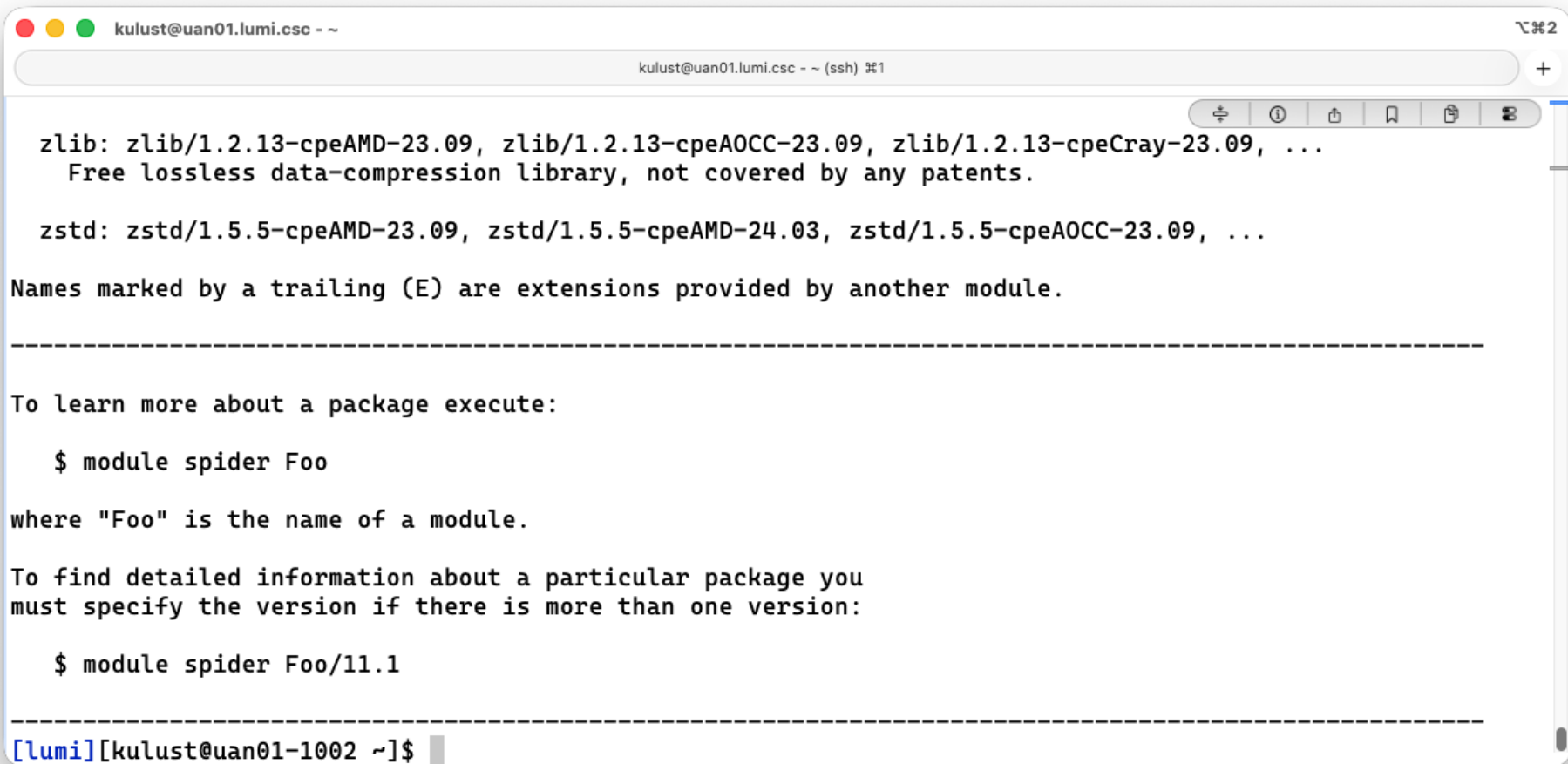
Did you know?
*****
The behaviour of many modules of the Cray Programming Environment depends
on the target modules that are loaded. All compilers on LUMI except for the
gcc implementation provided by the operating system support proper
optimizations for the zen2 and zen3 processors. The zen3 architectures is
the core architecture of the AMD Epyc Milan CPUs in the LUMI-C compute nodes
and the AMD EPYC Trento CPUs in the LUMI-G compute nodes. You may get get a
little extra performance by optimising for zen3 specifically. The CrayEnv
environment will automatically load the best target CPU, network and a
ccelerator target modules for each node. You can always restore them by
reloading the CrayEnv module (and there is no need to unload first).

In a job script, the environment is copied from the login nodes, so you'll
still have the Rome target modules. Here also reloading the CrayEnv module
will set the ones for the compute node you're running on.

[lumi][kulust@uan01-1001 ~]$ module spider
```

```
kulust@uan01.lumi.csc --  
kulust@uan01.lumi.csc -- (ssh) #1  
-----  
The following is a list of the modules and extensions currently available:  
-----  
ARMForge: ARMForge/22.0.1  
  Arm Forge debugging and profiling tools  
  
Autoconf: Autoconf/2.71 (E), Autoconf/2.72 (E)  
  
Autoconf-archive: Autoconf-archive/2023.02.20 (E), ...  
  
Automake: Automake/1.16.5 (E), Automake/1.17 (E)  
  
Bison: Bison/3.8.2 (E)  
  
Blosc: Blosc/1.21.5-cpeAMD-23.09, Blosc/1.21.5-cpeAMD-24.03, Blosc/1.21.5-cpeAOCC-23.09, ...  
  Blosc is an extremely fast, multi-threaded, meta-compressor library  
  
Boost: Boost/1.82.0-cpeAMD-23.09, Boost/1.82.0-cpeAOCC-23.09, Boost/1.82.0-cpeCray-23.09, ...  
  Boost provides free peer-reviewed portable C++ source libraries.  
  
Brotli: Brotli/1.0.9-cpeAMD-23.09, Brotli/1.0.9-cpeAOCC-23.09, Brotli/1.0.9-cpeCray-23.09, ...  
lines 1-22
```

module spider (command) (3)



```
kulust@uan01.lumi.csc --
kulust@uan01.lumi.csc -- (ssh) #1

zlib: zlib/1.2.13-cpeAMD-23.09, zlib/1.2.13-cpeAOCC-23.09, zlib/1.2.13-cpeCray-23.09, ...
Free lossless data-compression library, not covered by any patents.

zstd: zstd/1.5.5-cpeAMD-23.09, zstd/1.5.5-cpeAMD-24.03, zstd/1.5.5-cpeAOCC-23.09, ...

Names marked by a trailing (E) are extensions provided by another module.

-----

To learn more about a package execute:

$ module spider Foo

where "Foo" is the name of a module.

To find detailed information about a particular package you
must specify the version if there is more than one version:

$ module spider Foo/11.1

-----

[lumi][kulust@uan01-1002 ~]$
```

module spider

- `module spider` : Long list of all installed software with short description
 - Will also look into modules for “extensions” and show those also, marked with an “E”
 - By default only in the main software stacks on LUMI
- `module spider FFTW` : Look for the FFTW libraries on the system

```
kulust@uan01.lumi.csc --
kulust@uan01.lumi.csc -- (ssh) #1
[lumi][kulust@uan01-1003 ~]$ module spider FFTW
-----
cray-fftw:
-----
Versions:
  cray-fftw/3.3.10.5
  cray-fftw/3.3.10.7
  cray-fftw/3.3.10.10
  cray-fftw/3.3.10.11
-----
For detailed information about a specific "cray-fftw" package (including how to load the modules) use the
module's full name.
Note that names that have a trailing (E) are extensions provided by other modules.
For example:

  $ module spider cray-fftw/3.3.10.11
-----
[lumi][kulust@uan01-1004 ~]$
```

module spider

- `module spider` : Long list of all installed software with short description
 - Will also look into modules for “extensions” and show those also, marked with an “E”
 - By default only in the main software stacks on LUMI
- `module spider FFTW` : Look for the FFTW libraries on the system
- `module spider cray-fftw/3.3.10.11` : Look for this specific version
 - But this immediately shows the problems with the HPE Cray PE
 - Some of the lines don’t make much sense (see later)
 - Some options are missing also

```
kulust@uan01.lumi.csc --  
kulust@uan01.lumi.csc -- (ssh) #1  
-----  
cray-fftw: cray-fftw/3.3.10.11  
-----  
  
You will need to load all module(s) on any one of the lines below before the "cray-fftw/3.3.10.11" mod  
ule is available to load.  
  
LUMI/23.09 partition/C craype-x86-genoa  
LUMI/23.09 partition/C craype-x86-milan  
LUMI/23.09 partition/C craype-x86-milan-x  
LUMI/23.09 partition/C craype-x86-rome  
LUMI/23.09 partition/C craype-x86-spr  
LUMI/23.09 partition/C craype-x86-trento  
LUMI/23.09 partition/C craype-x86-turin  
LUMI/23.09 partition/D craype-x86-genoa  
LUMI/23.09 partition/D craype-x86-milan  
LUMI/23.09 partition/D craype-x86-milan-x  
LUMI/23.09 partition/D craype-x86-rome  
LUMI/23.09 partition/D craype-x86-spr  
LUMI/23.09 partition/D craype-x86-trento  
LUMI/23.09 partition/D craype-x86-turin  
lines 1-21
```

```
kulust@uan01.lumi.csc --  
kulust@uan01.lumi.csc -- (ssh) #1  
  
LUMI/23.09 partition/D craype-x86-rome  
LUMI/23.09 partition/D craype-x86-spr  
LUMI/23.09 partition/D craype-x86-trento  
LUMI/23.09 partition/D craype-x86-turin  
LUMI/23.09 partition/G craype-x86-genoa  
LUMI/23.09 partition/G craype-x86-milan  
LUMI/23.09 partition/G craype-x86-milan-x  
LUMI/23.09 partition/G craype-x86-rome  
LUMI/23.09 partition/G craype-x86-spr  
LUMI/23.09 partition/G craype-x86-trento  
LUMI/23.09 partition/G craype-x86-turin  
LUMI/23.09 partition/L craype-x86-genoa  
LUMI/23.09 partition/L craype-x86-milan  
LUMI/23.09 partition/L craype-x86-milan-x  
LUMI/23.09 partition/L craype-x86-rome  
LUMI/23.09 partition/L craype-x86-spr  
LUMI/23.09 partition/L craype-x86-trento  
LUMI/23.09 partition/L craype-x86-turin  
  
Help:  
Documentation: `man intro_fftw3`  
  
[lumi][kulust@uan01-1006 ~]$
```

module spider for a regular package

L U M I

- `module spider gnuplot` : Shows all versions of gnuplot on the system
- `module spider gnuplot/6.0.3-cpeGNU-25.09` : Shows help information for the specific module, including what should be done to make the module available

```
kulust@uan01.lumi.csc --  
kulust@uan01.lumi.csc -- (ssh) #1  
-----  
gnuplot:  
-----  
Description:  
  Gnuplot is a portable command-line driven graphing utility  
  
Versions:  
  gnuplot/5.4.8-cpeAOCC-23.09  
  gnuplot/5.4.8-cpeGNU-23.09  
  gnuplot/5.4.10-cpeAMD-24.03  
  gnuplot/5.4.10-cpeAOCC-24.03  
  gnuplot/5.4.10-cpeCray-24.03  
  gnuplot/5.4.10-cpeGNU-24.03  
  gnuplot/6.0.3-cpeAMD-25.03  
  gnuplot/6.0.3-cpeAOCC-25.03  
  gnuplot/6.0.3-cpeAOCC-25.09  
  gnuplot/6.0.3-cpeCray-25.03  
  gnuplot/6.0.3-cpeCray-25.09  
  gnuplot/6.0.3-cpeGNU-25.03  
  gnuplot/6.0.3-cpeGNU-25.09  
lines 1-22
```

module spider gnuplot (2)

```
kulust@uan01.lumi.csc --  
kulust@uan01.lumi.csc -- (ssh) #1  
  
gnuplot/5.4.8-cpeAOCC-23.09  
gnuplot/5.4.8-cpeGNU-23.09  
gnuplot/5.4.10-cpeAMD-24.03  
gnuplot/5.4.10-cpeAOCC-24.03  
gnuplot/5.4.10-cpeCray-24.03  
gnuplot/5.4.10-cpeGNU-24.03  
gnuplot/6.0.3-cpeAMD-25.03  
gnuplot/6.0.3-cpeAOCC-25.03  
gnuplot/6.0.3-cpeAOCC-25.09  
gnuplot/6.0.3-cpeCray-25.03  
gnuplot/6.0.3-cpeCray-25.09  
gnuplot/6.0.3-cpeGNU-25.03  
gnuplot/6.0.3-cpeGNU-25.09  
  
-----  
For detailed information about a specific "gnuplot" package (including how to load the modules) use the  
module's full name.  
Note that names that have a trailing (E) are extensions provided by other modules.  
For example:  
  
$ module spider gnuplot/6.0.3-cpeGNU-25.09  
  
-----  
[lumi][kulust@uan01-1007 ~]$
```

```
kulust@uan01.lumi.csc --
kulust@uan01.lumi.csc -- (ssh) #1

-----
gnuplot: gnuplot/6.0.3-cpeGNU-25.09
-----

Description:
  Gnuplot is a portable command-line driven graphing utility

You will need to load all module(s) on any one of the lines below before the "gnuplot/6.0.3-cpeGNU-25.09" module is available to load.

  LUMI/25.09 partition/C
  LUMI/25.09 partition/G
  LUMI/25.09 partition/L

Help:
  Description
  =====
  Gnuplot is a portable command-line driven graphing utility available for many platforms. The source code is copyrighted but freely distributed (i.e., you don't have to pay for it). It was originally created to allow scientists and students to visualize mathematical functions and data interactively, but has
```

lines 1-21

```
kulust@uan01.lumi.csc --
kulust@uan01.lumi.csc -- (ssh) #1

=====
Gnuplot is a portable command-line driven graphing utility available for many
platforms. The source code is copyrighted but freely distributed (i.e., you
don't have to pay for it). It was originally created to allow scientists and
students to visualize mathematical functions and data interactively, but has
grown to support many non-interactive uses such as web scripting. It is also
used as a plotting engine by third-party applications like Octave. Gnuplot has
been supported and under active development since 1986.

This version of GNUplot does not use Qt5 for its GUI, so the GUI is rather
primitive.

More information
=====
- Homepage: http://gnuplot.sourceforge.net/
- Documentation:
  - Web-based documentation: http://gnuplot.sourceforge.net/documentation.html
  - Manual page for gnuplot
- Site contact: LUMI User Support @ https://lumi-supercomputer.eu/user-support/need-help/

[lumi][kulust@uan01-1008 ~]$
```

module spider for extensions

- No example in the default Cray modules, so examples come from the LUMI software stacks
- `module spider CMake`
- `module spider CMake/4.2.3` : Will tell you which module contains this version of CMake and how to load it

```
kulust@uan01.lumi.csc --  
kulust@uan01.lumi.csc -- (ssh) #1  
-----  
CMake:  
-----  
Versions:  
  CMake/3.27.7 (E)  
  CMake/3.29.3 (E)  
  CMake/3.31.7 (E)  
  CMake/3.31.11 (E)  
  CMake/4.2.3 (E)  
  
Names marked by a trailing (E) are extensions provided by another module.  
  
-----  
For detailed information about a specific "CMake" package (including how to load the modules) use the module's full name.  
Note that names that have a trailing (E) are extensions provided by other modules.  
For example:  
  
$ module spider CMake/4.2.3  
-----  
lines 1-21
```

```
kulust@uan01.lumi.csc --  
kulust@uan01.lumi.csc -- (ssh) #1  
-----  
CMake: CMake/4.2.3 (E)  
-----  
This extension is provided by the following modules. To access the extension you must load one of the  
following modules. Note that any module names in parentheses show the module location in the software hier  
archy.  
  
buildtools/25.09 (LUMI/25.09 partition/L)  
buildtools/25.09 (LUMI/25.09 partition/G)  
buildtools/25.09 (LUMI/25.09 partition/D)  
buildtools/25.09 (LUMI/25.09 partition/C)  
buildtools/25.09 (CrayEnv)  
buildtools/25.09-bootstrap (LUMI/25.09 partition/L)  
buildtools/25.09-bootstrap (LUMI/25.09 partition/G)  
buildtools/25.09-bootstrap (LUMI/25.09 partition/D)  
buildtools/25.09-bootstrap (LUMI/25.09 partition/C)  
buildtools/25.09-bootstrap (CrayEnv)  
  
Names marked by a trailing (E) are extensions provided by another module.  
lines 1-20
```

module keyword

- Searches in the module short description and help message for the keyword.
 - E.g., try
`module keyword editor`
- We do try to put enough information in the modules to make this a suitable additional way to discover software that is already installed on the system

```
kulust@uan01.lumi.csc --
kulust@uan01.lumi.csc -- (ssh) #1
-----
The following modules match your search criteria: "editor"
-----

Vim: Vim/9.0.0016, Vim/9.0.1392, Vim/9.0.2059, Vim/9.1.0447
  Vim is an advanced text editor that seeks to provide the power of the de-facto Unix editor 'Vi',
  with a more complete feature setThe module provides a text-only version of vim, no GUI version.

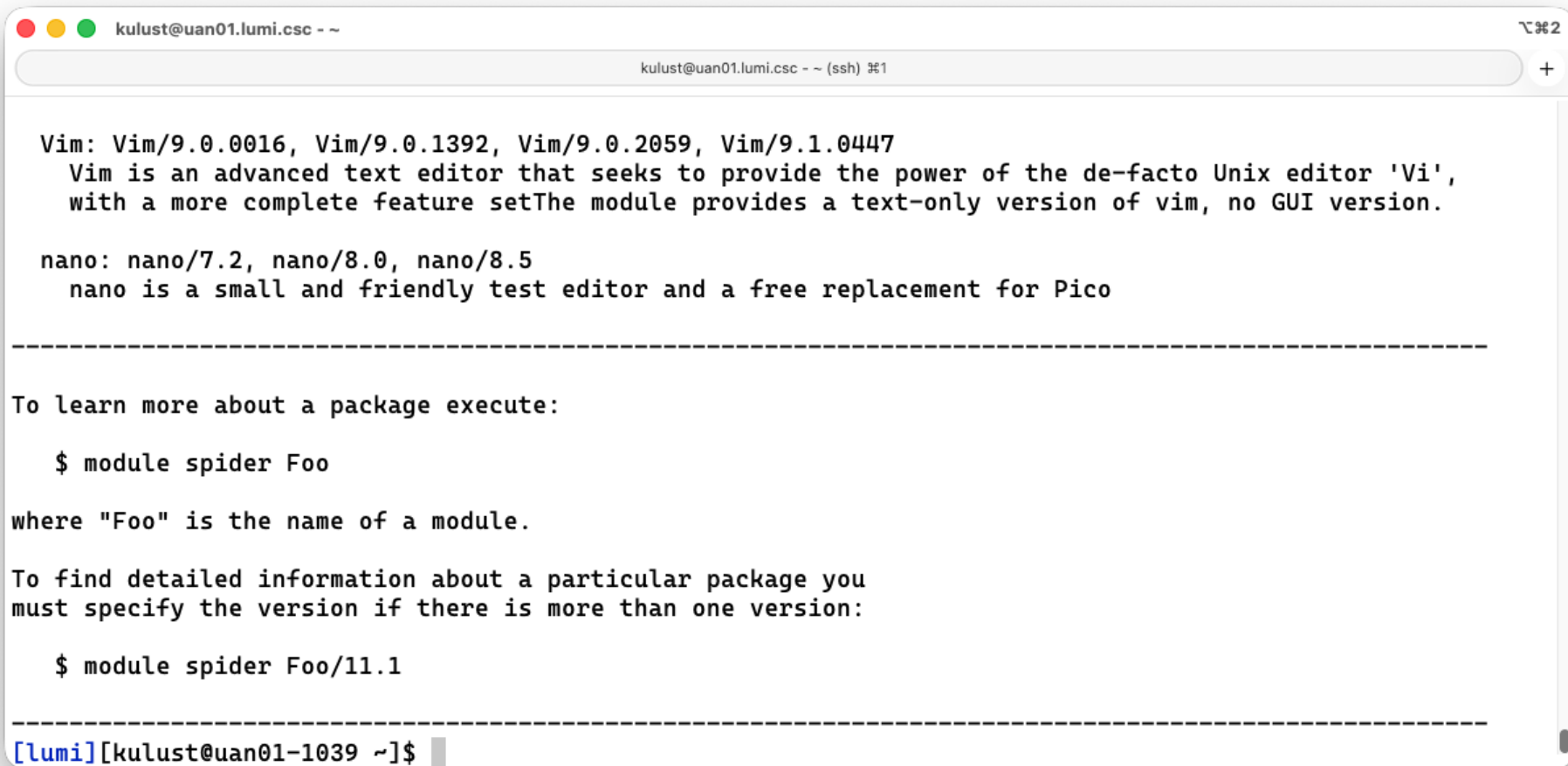
nano: nano/7.2, nano/8.0, nano/8.5
  nano is a small and friendly text editor and a free replacement for Pico
-----

To learn more about a package execute:

  $ module spider Foo

where "Foo" is the name of a module.

To find detailed information about a particular package you
must specify the version if there is more than one version:
lines 1-22
```



```
kulust@uan01.lumi.csc -- ~
kulust@uan01.lumi.csc -- (ssh) #1

Vim: Vim/9.0.0016, Vim/9.0.1392, Vim/9.0.2059, Vim/9.1.0447
Vim is an advanced text editor that seeks to provide the power of the de-facto Unix editor 'Vi',
with a more complete feature setThe module provides a text-only version of vim, no GUI version.

nano: nano/7.2, nano/8.0, nano/8.5
nano is a small and friendly text editor and a free replacement for Pico

-----

To learn more about a package execute:

$ module spider Foo

where "Foo" is the name of a module.

To find detailed information about a particular package you
must specify the version if there is more than one version:

$ module spider Foo/11.1

-----

[lumi][kulust@uan01-1039 ~]$
```

Sticky modules and module purge

- On some systems, you will be taught to avoid `module purge` (which unloads all modules)
- Sticky modules are modules that are not unloaded by `module purge`, but reloaded.
 - They can be force-unloaded with `module --force purge` and `module --force unload` but use at your own risk!
- Used on LUMI for the software stacks and modules that set the display style of the modules
 - But keep in mind that the modules are reloaded which can have side effects
- And also used for some system initialisation (`init-lumi` and `lumi-tools`)

```
kulust@uan04.lumi.csc - ~
kulust@uan04.lumi.csc - - (ssh) #1

----- EasyBuild managed systemwide software -----
ARMForge/22.0.1          lumi-tools/23.03 (S)      lumi-vnc/20230110        lumio/2.0.0 (D)
Vampir/10.0.0           lumi-tools/23.04 (S)      lumi-workspaces/0.1
Vampir/10.2.1           lumi-tools/23.11 (S)      lumio-ext-tools/1.0.0
Vampir/10.6.1 (D)      lumi-tools/24.05 (S,L,D)  lumio/1.0.0

----- HPE-Cray PE modules -----
PrgEnv-amd/8.4.0
PrgEnv-amd/8.5.0
PrgEnv-amd/8.6.0 (D)
PrgEnv-aocc/8.4.0
PrgEnv-aocc/8.5.0
PrgEnv-aocc/8.6.0 (D)
PrgEnv-cray-amd/8.4.0
PrgEnv-cray-amd/8.5.0
PrgEnv-cray-amd/8.6.0 (D)
PrgEnv-cray/8.4.0
PrgEnv-cray/8.5.0
PrgEnv-cray/8.6.0 (L,D)
PrgEnv-gnu-amd/8.4.0
PrgEnv-gnu-amd/8.5.0

lines 1-22
```

module av (2)

```
kulust@uan04.lumi.csc ~  
kulust@uan04.lumi.csc -- (ssh) #1  
PrgEnv-gnu-amd/8.6.0          (D)  
PrgEnv-gnu/8.4.0  
PrgEnv-gnu/8.5.0  
PrgEnv-gnu/8.6.0          (D)  
PrgEnv-nvhpc/8.5.0  
PrgEnv-nvidia/8.5.0  
amd-mixed/6.3.4  
amd/6.3.4                  (5.0.2:5.1.0:5.2.0:5.2.3:5.5.1:5.7.0:6.0.0:6.0.3:6.2.1:6.3.0)  
aocc-mixed/3.2.0_module  
aocc-mixed/3.2.0  
aocc-mixed/4.1.0  
aocc-mixed/5.0.0          (D)  
aocc/3.2.0_module  
aocc/3.2.0  
aocc/4.1.0  
aocc/5.0.0                (D)  
atp/3.15.1  
atp/3.15.3  
atp/3.15.6                (D)  
atp/3.15.7  
cce-mixed/16.0.1  
cce-mixed/17.0.1  
lines 23-44
```



A terminal window titled 'kulust@uan04.lumi.csc - ~' with a sub-window 'kulust@uan04.lumi.csc - - (ssh) #1'. The terminal displays the output of the 'module av' command, listing various software modules and their status. The output is as follows:

```
cce-mixed/19.0.0
cce-mixed/20.0.0          (D)
cce/16.0.1
cce/17.0.1
cce/19.0.0              (L,D)
cce/20.0.0
cpe-cuda/25.09
cpe/23.09
cpe/24.03
cpe/25.03              (D)
cpe/25.09
cray-R/4.2.1.2
cray-R/4.3.2
cray-R/4.4.0          (D)
cray-ccdb/5.0.1
cray-ccdb/5.0.3
cray-ccdb/5.0.6      (D)
cray-ccdb/5.0.7
cray-cti/2.18.1
cray-cti/2.18.3
cray-cti/2.19.1     (D)
cray-cti/2.20.0
```

At the bottom left of the terminal, a status bar indicates 'lines 45-66'.



A terminal window titled 'kulust@uan04.lumi.csc - ~' with a sub-window 'kulust@uan04.lumi.csc - - (ssh) #1'. The terminal displays the output of the 'module av' command, listing various software modules and their status. The status '(D)' indicates a default module, while '(L,D)' indicates a loaded module. The list includes modules from the 'cce' and 'cpe' families, as well as 'cray-R' and 'cray-cti' modules. The bottom of the terminal shows 'lines 45-66'.

```
kulust@uan04.lumi.csc - ~
kulust@uan04.lumi.csc - - (ssh) #1

cce-mixed/19.0.0
cce-mixed/20.0.0          (D)
cce/16.0.1
cce/17.0.1
cce/19.0.0              (L,D)
cce/20.0.0
cpe-cuda/25.09
cpe/23.09
cpe/24.03
cpe/25.03              (D)
cpe/25.09
cray-R/4.2.1.2
cray-R/4.3.2
cray-R/4.4.0          (D)
cray-ccdb/5.0.1
cray-ccdb/5.0.3
cray-ccdb/5.0.6      (D)
cray-ccdb/5.0.7
cray-cti/2.18.1
cray-cti/2.18.3
cray-cti/2.19.1     (D)
cray-cti/2.20.0

lines 45-66
```

```
kulust@uan04.lumi.csc ~  
kulust@uan04.lumi.csc -- (ssh) #1  
cray-libsci_acc/25.09.2  
cray-mpich-abi/8.1.29  
cray-mpich-abi/8.1.32 (D)  
cray-mpich-abi/9.0.0  
cray-mpich/8.1.29  
cray-mpich/8.1.32 (L,D)  
cray-mpich/9.0.0  
cray-mpixlate/1.0.4  
cray-mpixlate/1.0.7 (D)  
cray-mrnet/5.1.1  
cray-mrnet/5.1.2  
cray-mrnet/5.1.5 (D)  
cray-mrnet/5.1.6  
cray-openshmemx/11.6.1  
cray-openshmemx/11.7.1  
cray-openshmemx/11.7.4 (D)  
cray-openshmemx/11.8.0  
cray-pals/1.2.12  
cray-parallel-netcdf/1.12.3.11 (D)  
cray-parallel-netcdf/1.12.3.17  
cray-pmi/6.1.12  
cray-pmi/6.1.14  
lines 89-110
```



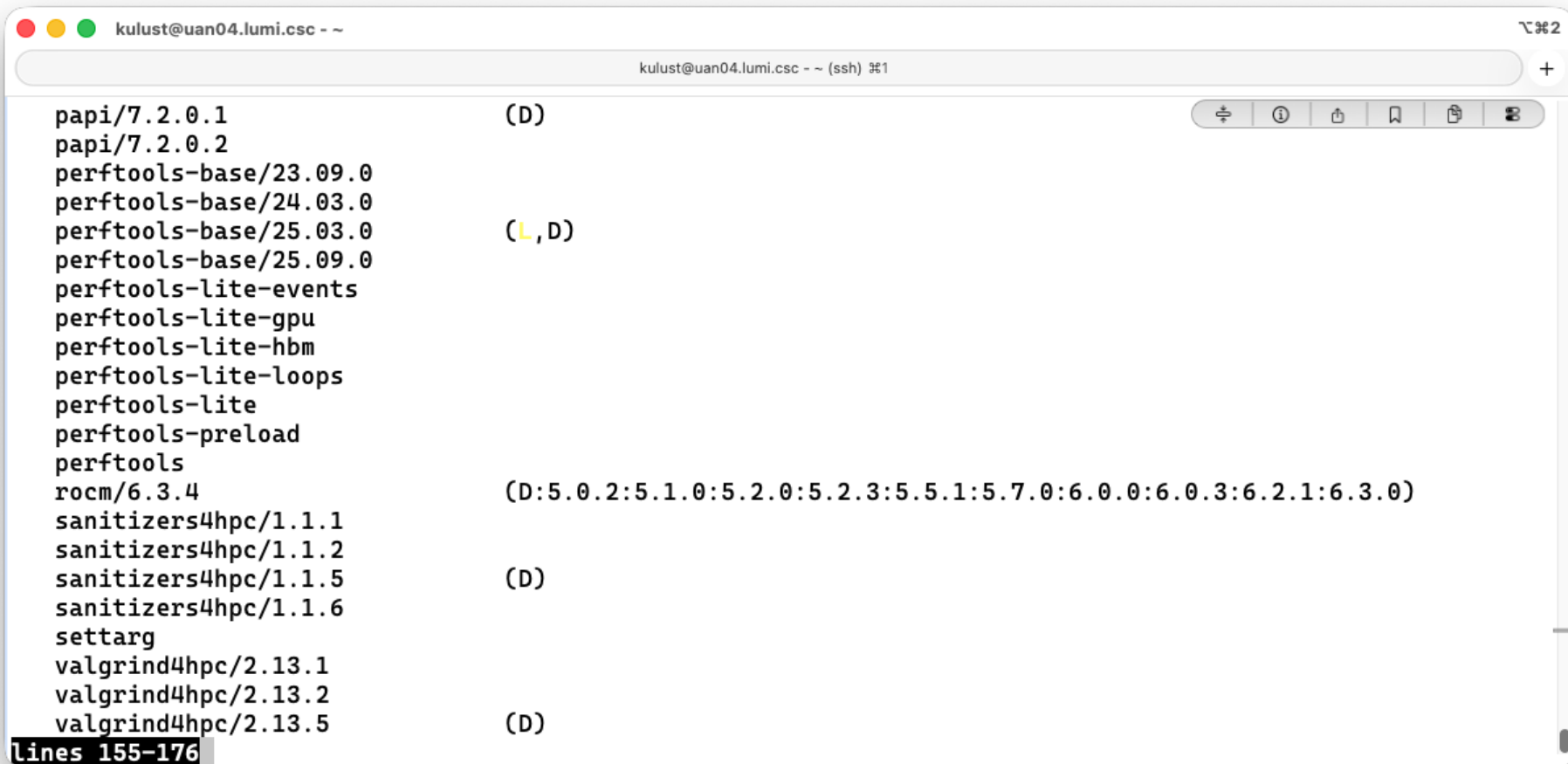
A terminal window titled 'kulust@uan04.lumi.csc - ~' with a sub-window 'kulust@uan04.lumi.csc - - (ssh) #1'. The terminal displays a list of modules and their status. The status '(L,D)' is highlighted in yellow. At the bottom left, a black box contains the text 'lines 111-132'.

```
cray-pmi/6.1.15          (D)
cray-pmi/6.1.16
cray-python/3.10.10
cray-python/3.11.7      (D)
cray-stat/4.12.1
cray-stat/4.12.2
cray-stat/4.12.5        (D)
cray-stat/4.12.6
cray-zmqnet/1.3.1       (D)
cray-zmqnet/1.3.2
craype/2.7.23
craype/2.7.31.11
craype/2.7.34           (L,D)
craype/2.7.35
craypkg-gen/1.3.28
craypkg-gen/1.3.30
craypkg-gen/1.3.32
craypkg-gen/1.3.35      (D)
craypkg-gen/1.3.36
gcc-mixed/11.2.0
gcc-mixed/12.2.0        (D)
gcc-native-mixed/12
```

lines 111-132

A terminal window with a title bar showing 'kulust@uan04.lumi.csc - ~' and a status bar showing 'kulust@uan04.lumi.csc - - (ssh) #1'. The terminal displays a list of modules with their versions and some are marked with '(D)'. The list includes gcc-native-mixed, gcc-native, gcc, gdb4hpc, iobuf, lmod, and papi. The bottom of the terminal shows 'lines 133-154' in a dark bar.

```
kulust@uan04.lumi.csc - ~  
kulust@uan04.lumi.csc - - (ssh) #1  
gcc-native-mixed/12.3  
gcc-native-mixed/13  
gcc-native-mixed/13.2  
gcc-native-mixed/14  
gcc-native-mixed/14.2 (D)  
gcc-native/12  
gcc-native/12.3  
gcc-native/13  
gcc-native/13.2  
gcc-native/14  
gcc-native/14.2 (D)  
gcc/10.3.0  
gcc/11.2.0  
gcc/12.2.0 (D)  
gdb4hpc/4.15.1  
gdb4hpc/4.16.1  
gdb4hpc/4.16.4 (D)  
gdb4hpc/4.16.5  
iobuf/2.0.10  
lmod  
papi/7.0.1.1  
papi/7.1.0.1  
lines 133-154
```



A terminal window titled 'kulust@uan04.lumi.csc - ~' showing the output of a 'module av' command. The window has a title bar with standard macOS window controls and a toolbar with icons for search, info, share, bookmark, print, and refresh. The terminal output lists various modules and their availability status in parentheses. The status '(L,D)' is highlighted in yellow. At the bottom left, a black box with white text indicates 'lines 155-176'.

```
kulust@uan04.lumi.csc - ~  
kulust@uan04.lumi.csc - ~ (ssh) #1  
papi/7.2.0.1 (D)  
papi/7.2.0.2  
perftools-base/23.09.0  
perftools-base/24.03.0  
perftools-base/25.03.0 (L,D)  
perftools-base/25.09.0  
perftools-lite-events  
perftools-lite-gpu  
perftools-lite-hbm  
perftools-lite-loops  
perftools-lite  
perftools-preload  
perftools  
rocm/6.3.4 (D:5.0.2:5.1.0:5.2.0:5.2.3:5.5.1:5.7.0:6.0.0:6.0.3:6.2.1:6.3.0)  
sanitizers4hpc/1.1.1  
sanitizers4hpc/1.1.2  
sanitizers4hpc/1.1.5 (D)  
sanitizers4hpc/1.1.6  
settarg  
valgrind4hpc/2.13.1  
valgrind4hpc/2.13.2  
valgrind4hpc/2.13.5 (D)  
lines 155-176
```

```

kulist@uan04.lumi.csc - ~
kulist@uan04.lumi.csc - - (ssh) #1

valgrind4hpc/2.13.6

----- HPE-Cray PE target modules -----
craype-accel-amd-gfx908      craype-arm-grace      craype-hugepages4M      craype-x86-milan-x
craype-accel-amd-gfx90a    craype-hugepages128M  craype-hugepages512M    craype-x86-milan
craype-accel-amd-gfx940    craype-hugepages16M   craype-hugepages64M     craype-x86-rome (L)
craype-accel-amd-gfx942    craype-hugepages1G    craype-hugepages8M      craype-x86-spr-hbm
craype-accel-host          craype-hugepages256M  craype-network-none     craype-x86-spr
craype-accel-nvidia70     craype-hugepages2G    craype-network-ofi (L)  craype-x86-trento
craype-accel-nvidia80     craype-hugepages2M    craype-network-ucx      craype-x86-turin
craype-accel-nvidia90     craype-hugepages32M   craype-x86-genoa

----- Software stacks -----
CrayEnv (S)    LUMI/25.03 (S,D)    Local-quantum/default (S)    spack/23.03
LUMI/23.09 (S)    LUMI/25.09 (S)    spack/22.08                  spack/23.03-2
LUMI/24.03 (S)    Local-CSC/default (S)    spack/22.08-2                spack/23.09 (D)

----- Modify the module display style -----
ModuleColour/off (S)    ModuleFullSpider/off (S)    ModuleLabel/system (S)
ModuleColour/on (S,D)  ModuleFullSpider/on (S,D)  ModulePowerUser/LUMI (S)
ModuleExtensions/hide (S)  ModuleLabel/label (S,L,D)  ModuleStyle/default
ModuleExtensions/show (S,D)  ModuleLabel/PEhierarchy (S)  ModuleStyle/reset (D)

lines 177-198

```

```
kulust@uan04.lumi.csc - ~
kulust@uan04.lumi.csc - - (ssh) #1

----- System initialisation -----
init-lumi/0.2 (S,L)

----- Non-PE HPE-Cray modules -----
libfabric/1.22.0          (L)
rocm/6.3.4                (5.0.2:5.1.0:5.2.0:5.2.3:5.5.1:5.7.0:6.0.0:6.0.3:6.2.1:6.3.0)
xpmem/2.11.5-1.3_g73ade43320bc (L)

----- This is a list of module extensions. Use "module --nx avail ..." to not show extensions. -----
  rclone (E)    restic (E)    s3cmd (E)

These extensions cannot be loaded directly, use "module spider extension_name" for more information.

Where:
L:      Module is loaded
S:      Module is Sticky, requires --force to unload or purge
Aliases: Aliases exist: foo/1.2.3 (1.2) means that "module load foo/1.2" will load foo/1.2.3
D:      Default Module
E:      Extension that is provided by another module

Additional ways to search for software:
http://ModuleFullSpider/on
```

```
kulust@uan04.lumi.csc - ~
kulust@uan04.lumi.csc - - (ssh) #1
xpmem/2.11.5-1.3_g73ade43320bc (L)
----- This is a list of module extensions. Use "module --nx avail ..." to not show extensions. -----
  rclone (E)    restic (E)    s3cmd (E)

These extensions cannot be loaded directly, use "module spider extension_name" for more information.

Where:
L:      Module is loaded
S:      Module is Sticky, requires --force to unload or purge
Aliases: Aliases exist: foo/1.2.3 (1.2) means that "module load foo/1.2" will load foo/1.2.3
D:      Default Module
E:      Extension that is provided by another module

Additional ways to search for software:
* Use "module spider" to find all possible modules and extensions.
* Use "module keyword key1 key2 ..." to search for all possible modules matching any of the "keys".
See the LUMI documentation at https://docs.lumi-supercomputer.eu/runjobs/lumi\_env/Lmod\_modules/ for more information on searching modules.
If then you still miss software, contact LUMI User Support via https://lumi-supercomputer.eu/user-support/need-help/.

[lumi][kulust@uan04-1002 ~]$
```

Changing how the module list is displayed

L U M I

- You may have noticed that you see descriptive texts in the module, not directories
- This can be changed by loading a module
 - `ModuleLabel/label` : The default view
 - `ModuleLabel/PEhierarchy` : Descriptive texts and unfolded PE hierarchy
 - `ModuleLabel/system` : Module directories
- Turn colour on or off using `ModuleColour/on` or `ModuleColour/off`
- Show or hide the module extensions with `ModuleExtensions/show` or `ModuleExtensions/hide`
- Index all modules with the spider command using `ModuleFullSpider/on`
- Show some hidden modules with `ModulePowerUser/LUMI`
 - This will also show undocumented/unsupported modules!
 - Can use `module --show_hidden avail` instead
- More customisation possible via LMOD environment variables

Getting help

- `module help` is the command to get help information for available modules
 - Without further arguments: help about the module command
 - We do try to add a bit more help information about what a module provides to the modules than default EasyBuild or Spack installations tend to do.
- Examples (require loading `CrayEnv`):
`module help cray-mpich`
`module help cray-python/3.11.7`
`module help buildtools/25.03`
- `module whatis` can produce a short description
`module whatis Subversion`
`module whatis Subversion/1.14.5`

A note on caching

- Large module system = lots of small module files = Lustre not very happy
 - But Lmod does use caches by default
 - Currently no system cache, only a user cache in `$HOME/.cache/lmod`
- Cache refreshed automatically every 24 hours
 - You'll notice when the `spider` or `available` commands are slow
 - But you may need to clean the cache after creating a new module as on LUMI Lmod does not always detect the change
- Also clear the cache if you notice very strange answers from `module spider`.
 - In the past, the HPE Cray PE has sometimes caused cache problems

A note on other commands

- `module load`, `module unload`, `module list` are fairly standard commands and the basic operation is the same in all module systems
 - Note that `module list` may also show inactive modules: Modules that were loaded at some point but got unloaded when a module closer to the root of the hierarchy got unloaded
- `module swap`:
 - Equivalent to an unload followed by a load
 - For two modules of the same family `module swap` is more efficient as Lmod does not first have to discover the family conflict
 - But it is not essential as LUMI has autoswap enabled

Questions?

