LUMI

LUMI Software Stacks

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

What this talk is about...

- Software stacks on LUMI

- Some remarks about Lmod

- Creating your customised environment with EasyBuild

- Containers

Design considerations

- Very leading edge and inhomogeneous machine (new interconnect, new GPU architecture with an immature software ecosystem, some NVIDIA GPUs for visualisation, a mix of zen2 and zen3)
  - Need to remain agile
- Users that come to LUMI from 11 different channels (not counting subchannels), with different expectations
- Small central support team considering the expected number of projects and users and the tasks the support team has
  - But contributions from local support teams
- Cray Programming Environment is a key part of our system
- Need for customised setups
  - Everybody wants a central stack as long as their software is in there but not much more
  - Look at the success of conda, Python virtual environments, containers, …

The LUMI solution

- Software organised in extensible software stacks based on a particular release of the PE
  - Many base libraries and some packages already pre-installed
  - Easy way to install additional packages in project space
- Modules managed by Lmod
  - More powerful than the (old) Modules Environment which is also supported by HPE Cray
  - Powerful features to search for modules
- EasyBuild is our primary tool for software installations
  - But uses HPE Cray specific toolchains
  - Offer a library of installation recipes
  - User installations integrate seamlessly with the central stack
  - We do have a Spack setup but don't do development in Spack ourselves

Policies

- Bring-your-own-license except for a selection of tools that are useful to a larger community
  - One downside of the distributed user management is that we do not even have the information needed to determine if a particular userid can use a particular software license
  - Even for software on the system, users remain responsible for checking the license!
- LUST tries to help with installations of recent software, but porting or bug fixing is not our work
  - Not all Linux or even supercomputer software will work on LUMI
  - We're too small a team to do all software installations, so don't count on us to do all the work
- Conda, (large) Python installations need to go in containers
  - We offer lumi-container-wrapper and cotainr to do that

Organisation: Software stacks

L U M I

- **CrayEnv:** Cray environment with some additional tools pushed in through EasyBuild

- **LUMI** stacks, each one corresponding to a particular release of the PE
  - Work with the Cray PE modules, but accessed through a replacement for the PrgEnv-* modules
  - Tuned versions for the 3 ~~4~~ types of hardware: zen2 (login, large memory nodes), zen3 (LUMI-C compute nodes), ~~zen2 + NVIDIA GPU (visualisation partition)~~, zen3 + MI250X (LUMI-G GPU partition)

- **spack:** Install software with Spack using compilers from the PE
  - Offered as-is for users who know Spack, but we do not do development in Spack

- Far future: Stack based on common EB toolchains as-is for LUMI-C

Accessing the Cray PE on LUMI
3 different ways

- Very bare environment available directly after login
    - What you can expect on a typical Cray system
    - Few tools as only the base OS image is available
    - User fully responsible for managing the target modules
- **CrayEnv**
    - "Enriched" Cray PE environment
    - Takes care of managing the target modules: (re)loading CrayEnv will reload an optimal set for the node you're on
    - Some additional tools, e.g., newer build tools (offered here and not in the bare environment as we need to avoid conflicts with other software stacks)
    - Otherwise used in the way discussed in this course

Accessing the Cray PE on LUMI
3 different ways

L U M I

- **LUMI** software stack
  - Each stack based on a particular release of the HPE Cray PE
    - Other modules are accessible but hidden from the default view
  - Better not to use the PrgEnv modules but the EasyBuild LUMI toolchains

HPE Cray PE	LUMI toolchain	
PrgEnv-cray	cpeCray	Cray Compiling Environment
PrgEnv-gnu	cpeGNU	GNU C/C++ and Fortran
PrgEnv-aocc	cpeAOCC	AMD CPU compilers
PrgEnv-amd	cpeAMD	AMD ROCm GPU compilers (LUMI-G only)

  - Environment in which we install most software (mostly with EasyBuild)

Accessing the Cray PE on LUMI
The LUMI software stack

- The LUMI software stack uses two levels of modules
    - LUMI/22.08, LUMI/22.12, LUMI/23.03: Versions of the LUMI stack
    - partition/L, partition/C, partition/G (~~and future partition/D~~): To select software optimised for the respective LUMI partition
        - partition/L is for both the login nodes and the large memory nodes (4TB)
    - Hidden partition/common for software that is available everywhere, but be careful using it for your own installs
    - When (re)loaded, the LUMI module will load the best matching partition module.
    - So be careful in job scripts: When your job starts, the environment will be that of the login nodes, but if you trigger a reload of the LUMI module it will be that of the compute node!

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

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"

- `module spider gnuplot` : Shows all versions of gnuplot on the system
  `module spider CMake`

- `module spider gnuplot/5.4.6-cpeGNU-22.12` : Shows help information for the specific module, including what should be done to make the module available
  - But this does not completely work with the Cray PE modules

- `module spider CMake/3.25.2` : Will tell you which module contains CMake and how to load it

module spider (command) (1)

module spider (command) (2)

LUMI

```
                                           kulust@uan04.lumi.csc - ~                                    ⌥⌘2
                                         kulust@uan04.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-archive: Autoconf-archive/2021.02.19 (E), ...

    Automake: Automake/1.16.5 (E)

    Bison: Bison/3.8.2 (E)

    Blosc: Blosc/1.21.1-cpeAMD-22.08, Blosc/1.21.1-cpeAOCC-21.12, Blosc/1.21.1-cpeAOCC-22.08, ...
      Blosc is an extremely fast, multi-threaded, meta-compressor library

    Boost: Boost/1.77.0-cpeAOCC-21.12, Boost/1.77.0-cpeCray-21.12, Boost/1.77.0-cpeGNU-21.12, ...
      Boost provides free peer-reviewed portable C++ source libraries.

    Brotli: Brotli/1.0.9-cpeAMD-22.08, Brotli/1.0.9-cpeAMD-22.12, Brotli/1.0.9-cpeAOCC-21.12, ...
  lines 1-22
```

`module spider (command) (3)`

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

  zlib: zlib/1.2.11-cpeAOCC-21.12, zlib/1.2.11-cpeCray-21.12, zlib/1.2.11-cpeGNU-21.12, ...
    Free lossless data-compression library, not covered by any patents.

  zstd: zstd/1.5.0-cpeAOCC-21.12, zstd/1.5.0-cpeCray-21.12, zstd/1.5.0-cpeGNU-21.12, ...

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@uan04-1002 ~]$
```

`module spider gnuplot`

```
------------------------------------------------------------------------------

  gnuplot:
------------------------------------------------------------------------------

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

    Versions:
        gnuplot/5.4.3-cpeAMD-22.08
        gnuplot/5.4.3-cpeAOCC-21.12
        gnuplot/5.4.3-cpeAOCC-22.08
        gnuplot/5.4.3-cpeCray-21.12
        gnuplot/5.4.3-cpeCray-22.06
        gnuplot/5.4.3-cpeCray-22.08
        gnuplot/5.4.3-cpeGNU-21.12
        gnuplot/5.4.3-cpeGNU-22.06
        gnuplot/5.4.3-cpeGNU-22.08
        gnuplot/5.4.6-cpeAMD-22.12
        gnuplot/5.4.6-cpeAOCC-22.12
        gnuplot/5.4.6-cpeCray-22.12
        gnuplot/5.4.6-cpeCray-23.03
        gnuplot/5.4.6-cpeGNU-22.12
lines 1-22
```

LUMI

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

        gnuplot/5.4.3-cpeAOCC-21.12
        gnuplot/5.4.3-cpeAOCC-22.08
        gnuplot/5.4.3-cpeCray-21.12
        gnuplot/5.4.3-cpeCray-22.06
        gnuplot/5.4.3-cpeCray-22.08
        gnuplot/5.4.3-cpeGNU-21.12
        gnuplot/5.4.3-cpeGNU-22.06
        gnuplot/5.4.3-cpeGNU-22.08
        gnuplot/5.4.6-cpeAMD-22.12
        gnuplot/5.4.6-cpeAOCC-22.12
        gnuplot/5.4.6-cpeCray-22.12
        gnuplot/5.4.6-cpeCray-23.03
        gnuplot/5.4.6-cpeGNU-22.12


------------------------------------------------------------------------------
  For detailed information about a specific "gnuplot" package (including how to load the modu
les) 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/5.4.6-cpeGNU-22.12
------------------------------------------------------------------------------
[lumi][kulust@uan01-1003 ~]$
```

module spider CMake

L U M I

```
[lumi][kulust@uan01-1004 ~]$ module spider CMake


--------------------------------------------------------------------------------
   CMake:
--------------------------------------------------------------------------------
     Versions:
        CMake/3.22.2 (E)
        CMake/3.23.2 (E)
        CMake/3.24.0 (E)
        CMake/3.25.2 (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 module
s) 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/3.25.2
--------------------------------------------------------------------------------
[lumi][kulust@uan01-1005 ~]$
```

`module spider gnuplot/5.4.6-cpeGNU-22.12`


```
-------------------------------------------------------------------------------
  gnuplot: gnuplot/5.4.6-cpeGNU-22.12
-------------------------------------------------------------------------------
    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/5.4
.6-cpeGNU-22.12" module is available to load.

      LUMI/22.12   partition/C
      LUMI/22.12   partition/G
      LUMI/22.12   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
lines 1-21
```

module spider gnuplot/5.4.6-cpeGNU-22.12 (2)

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

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-he
lp/


[lumi][kulust@uan01-1004 ~]$
```

module spider CMake/3.25.2

```
----------------------------------------------------------------------------
  CMake: CMake/3.25.2 (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 locatio
n in the software hierarchy.


      buildtools/23.03 (LUMI/23.03 partition/L)
      buildtools/23.03 (LUMI/23.03 partition/G)
      buildtools/23.03 (LUMI/23.03 partition/C)
      buildtools/23.03-bootstrap (LUMI/23.03 partition/L)
      buildtools/23.03-bootstrap (LUMI/23.03 partition/G)
      buildtools/23.03-bootstrap (LUMI/23.03 partition/C)
      buildtools/22.12 (LUMI/22.12 partition/L)
      buildtools/22.12 (LUMI/22.12 partition/G)
      buildtools/22.12 (LUMI/22.12 partition/C)
      buildtools/22.12 (CrayEnv)
      buildtools/22.12-bootstrap (LUMI/22.12 partition/L)
      buildtools/22.12-bootstrap (LUMI/22.12 partition/G)
      buildtools/22.12-bootstrap (LUMI/22.12 partition/C)
lines 1-20
```

module spider CMake/3.25.2

```
n in the software hierarchy.


        buildtools/23.03 (LUMI/23.03 partition/L)
        buildtools/23.03 (LUMI/23.03 partition/G)
        buildtools/23.03 (LUMI/23.03 partition/C)
        buildtools/23.03-bootstrap (LUMI/23.03 partition/L)
        buildtools/23.03-bootstrap (LUMI/23.03 partition/G)
        buildtools/23.03-bootstrap (LUMI/23.03 partition/C)
        buildtools/22.12 (LUMI/22.12 partition/L)
        buildtools/22.12 (LUMI/22.12 partition/G)
        buildtools/22.12 (LUMI/22.12 partition/C)
        buildtools/22.12 (CrayEnv)
        buildtools/22.12-bootstrap (LUMI/22.12 partition/L)
        buildtools/22.12-bootstrap (LUMI/22.12 partition/G)
        buildtools/22.12-bootstrap (LUMI/22.12 partition/C)
        buildtools/22.12-bootstrap (CrayEnv)


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


[lumi][kulust@uan01-1005 ~]$
```
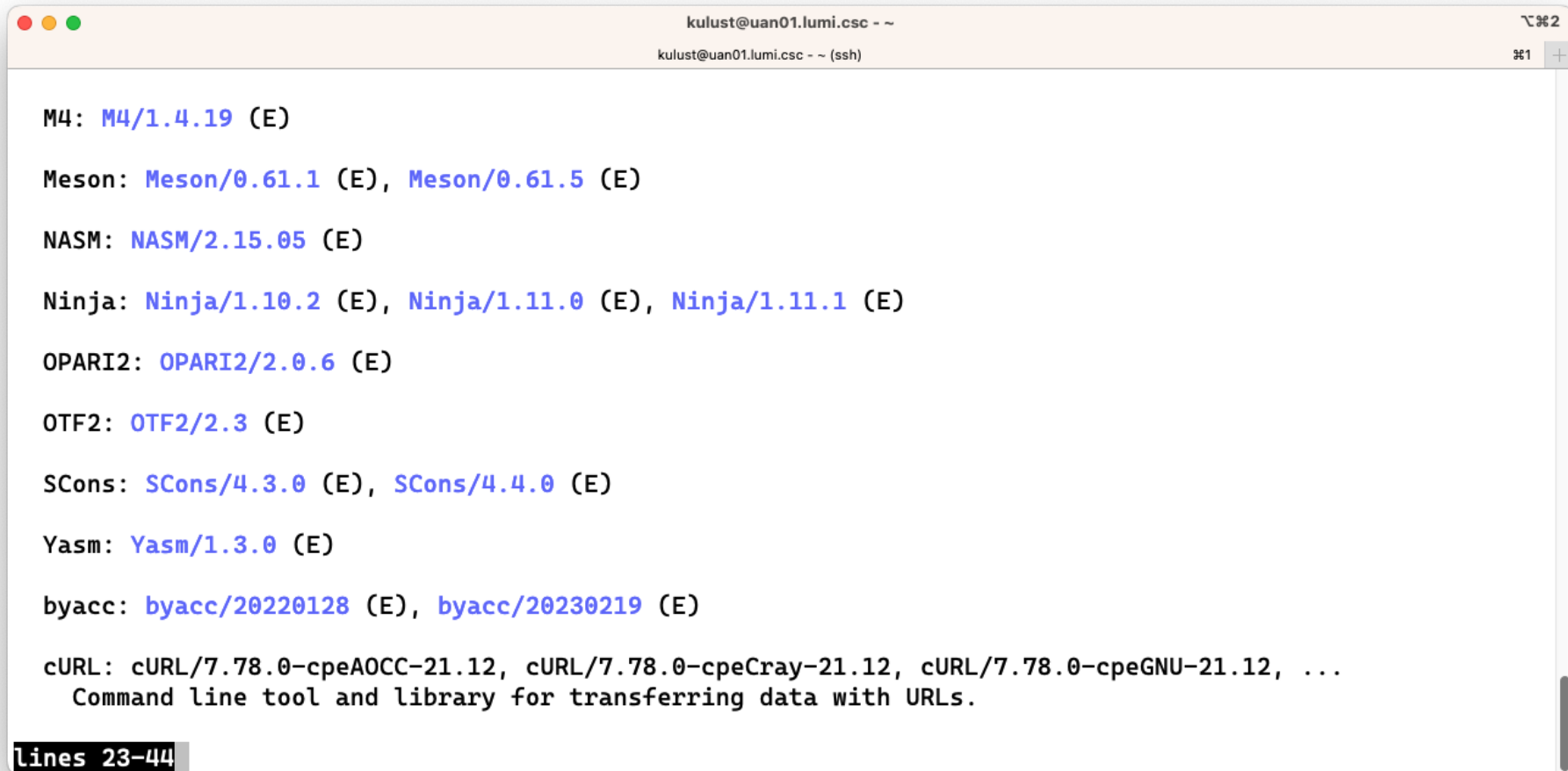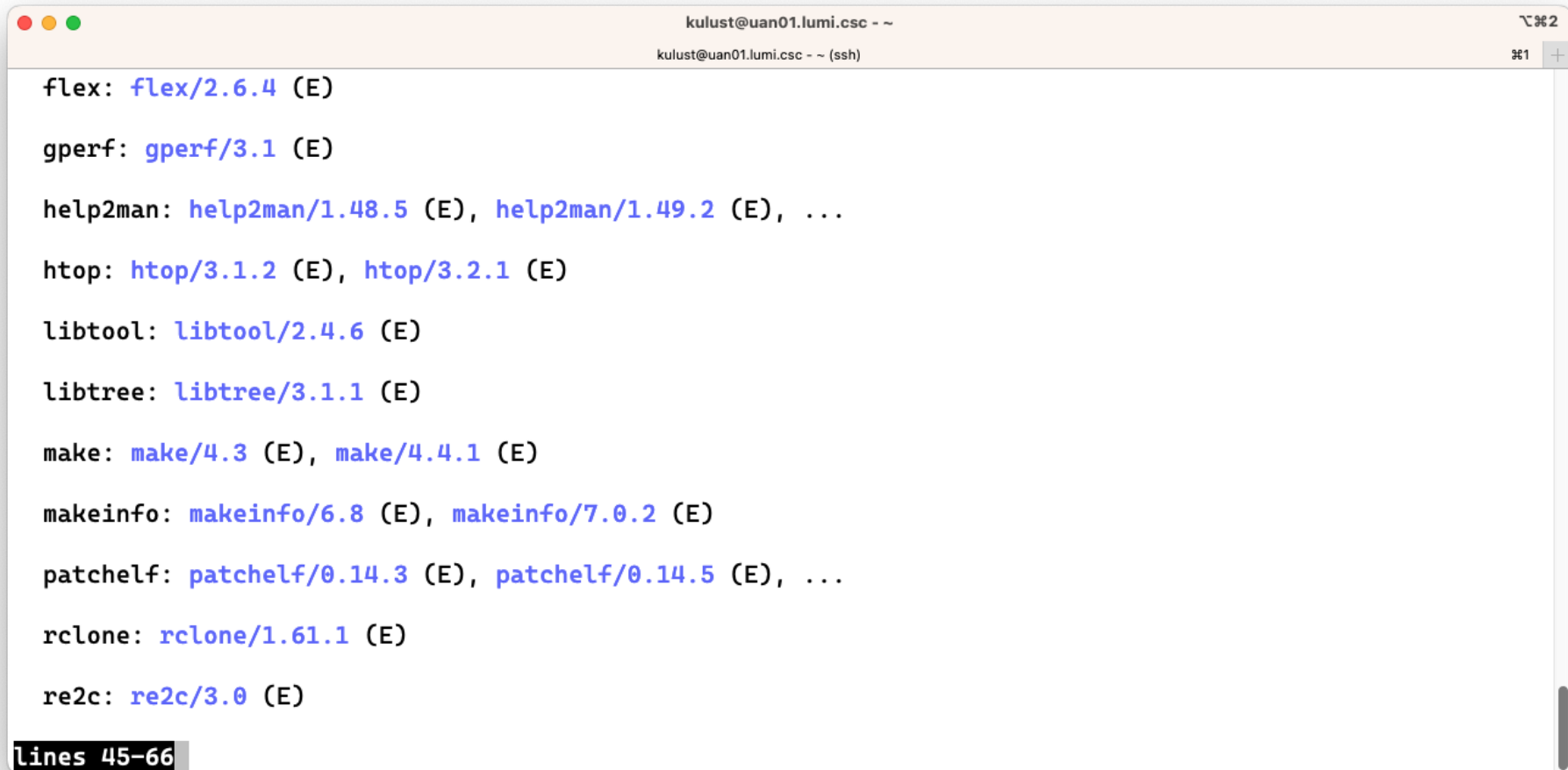
module keyword

- Currently not yet very useful due to a bug in Cray Lmod

- It searches in the module short description and help for the keyword.
  - E.g., try
    ```
    module keyword https
    ```

- 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

module keyword https

LUMI

```
--------------------------------------------------------------------------------

The following modules match your search criteria: "https"
--------------------------------------------------------------------------------

  Autoconf: Autoconf/2.71 (E)

  Autoconf-archive: Autoconf-archive/2021.02.19 (E), ...

  Automake: Automake/1.16.5 (E)

  Bison: Bison/3.8.2 (E)

  CMake: CMake/3.22.2 (E), CMake/3.23.2 (E), CMake/3.24.0 (E), ...

  CubeLib: CubeLib/4.6 (E)

  CubeWriter: CubeWriter/4.6 (E)

  Doxygen: Doxygen/1.9.3 (E), Doxygen/1.9.4 (E), Doxygen/1.9.6 (E)

  GPP: GPP/2.27 (E)
lines 1-22
```

module keyword https (2)

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

M4: M4/1.4.19 (E)

Meson: Meson/0.61.1 (E), Meson/0.61.5 (E)

NASM: NASM/2.15.05 (E)

Ninja: Ninja/1.10.2 (E), Ninja/1.11.0 (E), Ninja/1.11.1 (E)

OPARI2: OPARI2/2.0.6 (E)

OTF2: OTF2/2.3 (E)

SCons: SCons/4.3.0 (E), SCons/4.4.0 (E)

Yasm: Yasm/1.3.0 (E)

byacc: byacc/20220128 (E), byacc/20230219 (E)

cURL: cURL/7.78.0-cpeAOCC-21.12, cURL/7.78.0-cpeCray-21.12, cURL/7.78.0-cpeGNU-21.12, ...
   Command line tool and library for transferring data with URLs.
lines 23-44
```

module keyword https (3)

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

flex: flex/2.6.4 (E)

gperf: gperf/3.1 (E)

help2man: help2man/1.48.5 (E), help2man/1.49.2 (E), ...

htop: htop/3.1.2 (E), htop/3.2.1 (E)

libtool: libtool/2.4.6 (E)

libtree: libtree/3.1.1 (E)

make: make/4.3 (E), make/4.4.1 (E)

makeinfo: makeinfo/6.8 (E), makeinfo/7.0.2 (E)

patchelf: patchelf/0.14.3 (E), patchelf/0.14.5 (E), ...

rclone: rclone/1.61.1 (E)

re2c: re2c/3.0 (E)

lines 45-66
```

module keyword https (4)

```
    restic: restic/0.15.1 (E)

    s3cmd: s3cmd/2.3.0 (E)

    sec: sec/4.8 (E), sec/4.9 (E)

    tree: tree/2.0.2 (E)

    wget: wget/1.21.2-cpeAOCC-21.12, wget/1.21.2-cpeCray-21.12, wget/1.21.2-cpeGNU-21.12, ...
        wget - GNU wget, a free software package for retrieving files using HTTP, HTTPS and FTP

    xxd: xxd/8.2.4293 (E), xxd/8.2.5172 (E), xxd/9.0.0193 (E), ...

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.
lines 67-88
```

module keyword https (5)


```
  wget: wget/1.21.2-cpeAOCC-21.12, wget/1.21.2-cpeCray-21.12, wget/1.21.2-cpeGNU-21.12, ...
    wget - GNU wget, a free software package for retrieving files using HTTP, HTTPS and FTP

  xxd: xxd/8.2.4293 (E), xxd/8.2.5172 (E), xxd/9.0.0193 (E), ...

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@uan01-1005 ~]$
```

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`

- 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, so any change to modules that are loaded by these modules will be wiped out.

module av

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


--------------- EasyBuild managed software for software stack unknown on LUMI-X ---------------
    ARMForge/22.0.1          lumi-tools/23.03  (S)         lumi-workspaces/0.1
    Vampir/10.0.0            lumi-tools/23.04  (S,L,D)     lumio-ext-tools/1.0.0
    Vampir/10.2.1    (D)     lumi-vnc/20230110            lumio/1.0.0


------------------------------------- HPE-Cray PE modules -------------------------------------
    PrgEnv-amd/8.3.3                         cray-mpixlate/1.0.0.6
    PrgEnv-aocc/8.3.3                        cray-mpixlate/1.0.1.10          (D)
    PrgEnv-cray-amd/8.3.3                    cray-mrnet/5.0.4
    PrgEnv-cray/8.3.3            (L)         cray-openshmemx/11.5.6
    PrgEnv-gnu-amd/8.3.3                     cray-openshmemx/11.5.7          (D)
    PrgEnv-gnu/8.3.3                         cray-openshmemx/11.5.8
    amd-mixed/5.2.3                          cray-pals/1.2.0
    amd/5.2.3                   (5.0.2:5.2.0)  cray-pals/1.2.5                (D)
    aocc-mixed/3.2.0                         cray-pals/1.2.11
    aocc/3.2.0                               cray-parallel-netcdf/1.12.2.5
    atp/3.14.13                              cray-parallel-netcdf/1.12.3.1  (D)
    atp/3.14.16                 (D)          cray-parallel-netcdf/1.12.3.3
    atp/3.14.18                              cray-pmi-lib/6.0.17
    cce-mixed/14.0.2                         cray-pmi/6.0.17
    cce-mixed/15.0.0                         cray-pmi/6.1.3
lines 1-22
```

```
                                     kulust@uan01.lumi.csc - ~                                    ⌥⌘2
                                    kulust@uan01.lumi.csc - ~ (ssh)                                ⌘1

    cce-mixed/15.0.1            (D)        cray-pmi/6.1.8                (D)
    cce/14.0.2                             cray-pmi/6.1.10
    cce/15.0.0                 (L,D)       cray-python/3.9.12.1
    cce/15.0.1                             cray-python/3.9.13.1          (D)
    cpe/22.08                              cray-stat/4.11.12
    cpe/22.12                  (D)         cray-stat/4.11.13            (D)
    cpe/23.03                              craype/2.7.17
    cray-R/4.1.3.1                         craype/2.7.19                (L,D)
    cray-R/4.2.1.1             (D)         craype/2.7.20
    cray-ccdb/4.12.13                      craypkg-gen/1.3.25
    cray-cti/2.15.13                       craypkg-gen/1.3.28           (D)
    cray-cti/2.15.14                       gcc-mixed/11.2.0
    cray-cti/2.16.0                        gcc-mixed/12.2.0             (D)
    cray-cti/2.17.1           (D)          gcc/10.3.0
    cray-cti/2.17.2                        gcc/11.2.0
    cray-dsmml/0.2.2         (L)           gcc/12.2.0                   (D)
    cray-dyninst/12.1.1                    gdb4hpc/4.14.2
    cray-fftw/3.3.8.13                     gdb4hpc/4.14.6               (D)
    cray-fftw/3.3.10.1                     gdb4hpc/4.14.7
    cray-fftw/3.3.10.3       (D)           iobuf/2.0.10
    cray-hdf5-parallel/1.12.1.5            lmod
    cray-hdf5-parallel/1.12.2.1 (D)        papi/6.0.0.15
lines 23-44
```

module av (3)


```
cray-hdf5-parallel/1.12.2.3              papi/6.0.0.17                (D)
cray-hdf5/1.12.1.5                        papi/7.0.0.1
cray-hdf5/1.12.2.1          (D)           perftools
cray-hdf5/1.12.2.3                        perftools-base/22.06.0
cray-libpals/1.2.0                        perftools-base/22.12.0      (L,D)
cray-libpals/1.2.5         (D)            perftools-base/23.03.0
cray-libpals/1.2.11                       perftools-lite
cray-libsci/21.08.1.2                     perftools-lite-events
cray-libsci/22.08.1.1                     perftools-lite-gpu
cray-libsci/22.12.1.1      (L,D)          perftools-lite-hbm
cray-libsci/23.02.1.1                     perftools-lite-loops
cray-libsci_acc/22.08.1.1                 perftools-preload
cray-libsci_acc/22.12.1.1  (D)            rocm/5.2.3                   (D:5.0.2:5.2.0)
cray-mpich-abi/8.1.18                     sanitizers4hpc/1.0.1
cray-mpich-abi/8.1.23      (D)            sanitizers4hpc/1.0.4         (D)
cray-mpich-abi/8.1.25                     settarg
cray-mpich/8.1.18                         valgrind4hpc/2.12.10         (D)
cray-mpich/8.1.23          (L,D)          valgrind4hpc/2.12.11
cray-mpich/8.1.25


------------------------------ HPE-Cray PE target modules ------------------------------
  craype-accel-amd-gfx908     craype-hugepages256M     craype-network-ofi (L)
lines 45-66
```

LUMI

```
  craype-accel-amd-gfx90a     craype-hugepages2G     craype-network-ucx
  craype-accel-host           craype-hugepages2M     craype-x86-genoa
  craype-accel-nvidia70       craype-hugepages32M    craype-x86-milan
  craype-accel-nvidia80       craype-hugepages4M     craype-x86-milan-x
  craype-arm-grace            craype-hugepages512M   craype-x86-rome    (L)
  craype-hugepages128M        craype-hugepages64M    craype-x86-spr
  craype-hugepages16M         craype-hugepages8M     craype-x86-spr-hbm
  craype-hugepages1G          craype-network-none    craype-x86-trento


----------------------------------- Software stacks ------------------------------------
  CrayEnv     (S)      LUMI/22.12 (S)    spack/22.08       spack/23.03 (D)
  LUMI/22.08 (S,D)     LUMI/23.03 (S)    spack/22.08-2


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


--------------------------------- System initialisation --------------------------------
  init-lumi/0.1 (S)    init-lumi/0.2 (S,L,D)


-------------------------------- Non-PE HPE-Cray modules -------------------------------
lines 67-88
```

LUMI

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

    chapel/1.28.0
    cray-lustre-client-ofed/2.15.0.4_rc2_cray_156_g28e43d4-2.4_10.15__g28e43d47cb.shasta
    dvs/2.15_4.4.129-2.4_21.37__gfcb083ac
    libfabric/1.15.2.0                                              (L)
    rocm/5.2.3                                                      (5.0.2:5.2
.0)
    xpmem/2.5.2-2.4_3.20__gd0f7936.shasta                           (L)


----------------------------- This is a list of module extensions -----------------------------
    Autoconf          (E)    Doxygen  (E)    OTF2      (E)    htop      (E)    restic   (E)
    Autoconf-archive  (E)    GPP      (E)    SCons     (E)    libtool   (E)    s3cmd    (E)
    Automake          (E)    M4       (E)    Yasm      (E)    make      (E)    sec      (E)
    Bison             (E)    Meson    (E)    byacc     (E)    makeinfo  (E)    tree     (E)
    CMake             (E)    NASM     (E)    flex      (E)    patchelf  (E)    xxd      (E)
    CubeLib           (E)    Ninja    (E)    gperf     (E)    rclone    (E)
    CubeWriter        (E)    OPARI2   (E)    help2man  (E)    re2c      (E)

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

    Where:
     L:          Module is loaded
lines 89-108
```

LUMI

```
    CubeLib        (E)     Ninja   (E)     gperf    (E)     rclone   (E)
    CubeWriter     (E)     OPARI2  (E)     help2man (E)     re2c     (E)

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

  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/ fo
r 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@uan01-1008 ~]$
```

Changing how the module list is displayed

- You may have noticed that you don't see directories in the module view but descriptive texts

- 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 some hidden modules with `ModulePowerUser/LUMI`
  - This will also show undocumented/unsupported modules!

- More customisation possible via LMOD environment variables

Installing software on HPC systems

- Software on an HPC system is rarely installed from RPM
  - Generic RPMs not optimised for the specific CPU
  - Generic RPMs may not work with the specific LUMI environment (SlingShot interconnect, kernel modules, resource manager)
  - Multi-user system so usually no "one version fits all"
  - Need a small system image as nodes are diskless
- Spack and EasyBuild are the two most popular HPC-specific software build and installation frameworks
  - Usually install from sources to adapt the software to the underlying hardware and OS
  - Installation instructions in a way that can be communicated and executed easily
  - Make software available via modules
  - Dependency handling compatible with modules

Extending the LUMI stack with EasyBuild

- Fully integrated in the LUMI software stack
    - Load the LUMI module and modules should appear in your module view
    - EasyBuild-user module to install packages in your user space
    - Will use existing modules for dependencies if those are already on the system or in your personal/project stack
- EasyBuild built-in easyconfigs do not work on LUMI, not even on LUMI-C
    - GNU-based toolchains: Would give problems with MPI
    - Intel-based toolchains: Intel compilers and AMD CPUs are a problematic cocktail
- Library of recipes that we made in the [LUMI-EasyBuild-contrib GitHub repository](#)
    - EasyBuild-user will find a copy on the system or in your install
    - List of recipes in [lumi-supercomputer.github.io/LUMI-EasyBuild-docs](#)

EasyBuild recipes - easyconfigs

- Build recipe for an individual package = module
  - Relies on either a generic or a specific installation process provided by an easyblock
- Steps
  - Downloading sources and patches
  - Typical configure – build – (test) – install process
  - Extensions mechanism for perl/python/R packages
  - Some simple checks
  - Creation of the module
- All have several parameters in the easyconfig file

The toolchain concept

- A set of compiler, MPI implementation and basic math libraries
  - Simplified concept on LUMI as there is no hierarchy as on some other EasyBuild systems
- These are the cpeCray, cpeGNU, cpeAOCC and cpeAMD modules mentioned before!

HPE Cray PE	LUMI toolchain	
PrgEnv-cray	cpeCray	Cray Compiling Environment
PrgEnv-gnu	cpeGNU	GNU C/C++ and Fortran
PrgEnv-aocc	cpeAOCC	AMD CPU compilers
PrgEnv-amd	cpeAMD	AMD ROCm GPU compilers (LUMI-G only)

The toolchain concept (2)

- Special toolchain: SYSTEM to use the system compiler
  - Does not fully function in the same way as the other toolchains when it comes to dependency handling
  - Used on LUMI for CrayEnv and some packages with few dependencies
- It is not possible to load packages from different cpe toolchains at the same time
  - EasyBuild restriction, because mixing libraries compiled with different compilers does not always work
- Packages compiled with one cpe toolchain can be loaded together with packages compiled with the SYSTEM toolchain
  - But we do avoid mixing them when linking

easyconfig names and module names

GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb

Additional information

Toolchain name and version (missing for SYSTEM)

Version of the package

Name of the package

Module: GROMACS/2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU

Installing
Step 1: Where to install

- Default location is `$HOME/EasyBuild`

- But better is to install in your project directory for the whole project
  - `export EBU_USER_PREFIX=/project/project_465000000/EasyBuild`
  - Set this *before* loading the `LUMI` module
  - All users of the software tree have to set this environment variable to use the software tree
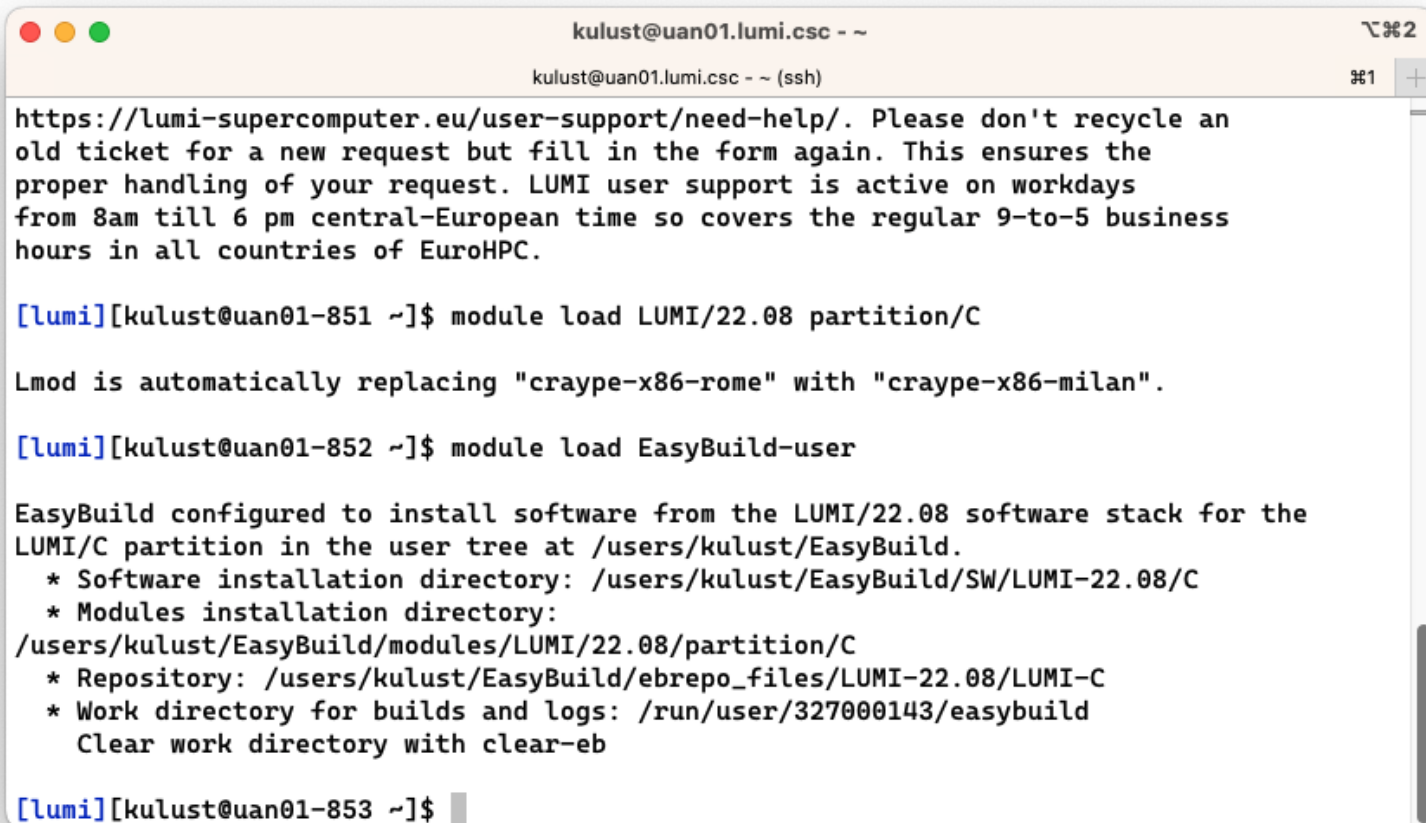
Installing
Step 2: Configure the environment

- Load the modules for the LUMI software stack and partition that you want to use. E.g.,
  ```
  module load LUMI/22.08 partition/C
  ```

- Load the EasyBuild-user module to make EasyBuild available and to configure it for installing software in the chosen stack and partition:
  ```
  module load EasyBuild-user
  ```

- In many cases, cross-compilation is possible by loading a different partition module than the one auto-loaded by LUMI
  - Though cross-compilation is currently problematic for GPU code

```
module load LUMI/22.08 partition/C
module load EasyBuild-user
```


LUMI

```
https://lumi-supercomputer.eu/user-support/need-help/. Please don't recycle an
old ticket for a new request but fill in the form again. This ensures the
proper handling of your request. LUMI user support is active on workdays
from 8am till 6 pm central-European time so covers the regular 9-to-5 business
hours in all countries of EuroHPC.

[lumi][kulust@uan01-851 ~]$ module load LUMI/22.08 partition/C

Lmod is automatically replacing "craype-x86-rome" with "craype-x86-milan".

[lumi][kulust@uan01-852 ~]$ module load EasyBuild-user

EasyBuild configured to install software from the LUMI/22.08 software stack for the
LUMI/C partition in the user tree at /users/kulust/EasyBuild.
  * Software installation directory: /users/kulust/EasyBuild/SW/LUMI-22.08/C
  * Modules installation directory:
/users/kulust/EasyBuild/modules/LUMI/22.08/partition/C
  * Repository: /users/kulust/EasyBuild/ebrepo_files/LUMI-22.08/LUMI-C
  * Work directory for builds and logs: /run/user/327000143/easybuild
    Clear work directory with clear-eb

[lumi][kulust@uan01-853 ~]$
```

Installing
Step 3: Install the software

- Let's, e.g., install GROMACS

  - Search if GROMACS build recipes are available:

    - Check the [LUMI Software Library](#) that lists all available software through EasyBuild.

    - Or on the command line:
      ```
      eb --search GROMACS
      eb –S GROMACS
      ```

  - Let's take GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb:
    ```
    eb GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb -D
    eb GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb -r
    ```

- Now the module should be available
  ```
  module avail GROMACS
  ```

```
●  ●  ●                    kulust@uan01.lumi.csc - ~                    ⌥⌘2

                    kulust@uan01.lumi.csc - ~ (ssh)                     ⌘1  +

 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2020.4-cpeCr
ay-21.08-PLUMED-2.6.4-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2020.4-cpeGN
U-21.08-PLUMED-2.6.4-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2020.6-cpeCr
ay-21.08-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2020.6-cpeCr
ay-21.08-PLUMED-2.7.2-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2020.6-cpeGN
U-21.08-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2020.6-cpeGN
U-21.08-PLUMED-2.7.2-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2021-cpeCray
-21.08-PLUMED-2.7.2-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2021-cpeGNU-
21.08-PLUMED-2.7.2-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2021.3-cpeCr
ay-21.08-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2021.3-cpeGN
U-21.08-CPU.eb
 * /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2021.4-cpeAO
lines 1-11
```

```
eb -S GROMACS | less
```


LUMI

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

CFGS1=/appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS
 * $CFGS1/GROMACS-2020.4-cpeCray-21.08-PLUMED-2.6.4-CPU.eb
 * $CFGS1/GROMACS-2020.4-cpeGNU-21.08-PLUMED-2.6.4-CPU.eb
 * $CFGS1/GROMACS-2020.6-cpeCray-21.08-CPU.eb
 * $CFGS1/GROMACS-2020.6-cpeCray-21.08-PLUMED-2.7.2-CPU.eb
 * $CFGS1/GROMACS-2020.6-cpeGNU-21.08-CPU.eb
 * $CFGS1/GROMACS-2020.6-cpeGNU-21.08-PLUMED-2.7.2-CPU.eb
 * $CFGS1/GROMACS-2021-cpeCray-21.08-PLUMED-2.7.2-CPU.eb
 * $CFGS1/GROMACS-2021-cpeGNU-21.08-PLUMED-2.7.2-CPU.eb
 * $CFGS1/GROMACS-2021.3-cpeCray-21.08-CPU.eb
 * $CFGS1/GROMACS-2021.3-cpeGNU-21.08-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeAOCC-21.12-PLUMED-2.7.4-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeAOCC-21.12-PLUMED-2.8.0-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeCray-21.12-PLUMED-2.7.4-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeCray-21.12-PLUMED-2.8.0-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeCray-22.06-PLUMED-2.7.4-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeCray-22.06-PLUMED-2.8.0-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeCray-22.08-PLUMED-2.7.4-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeGNU-21.12-PLUMED-2.7.4-CPU.eb
 * $CFGS1/GROMACS-2021.4-cpeGNU-21.12-PLUMED-2.8.0-CPU.eb
lines 1-21
```
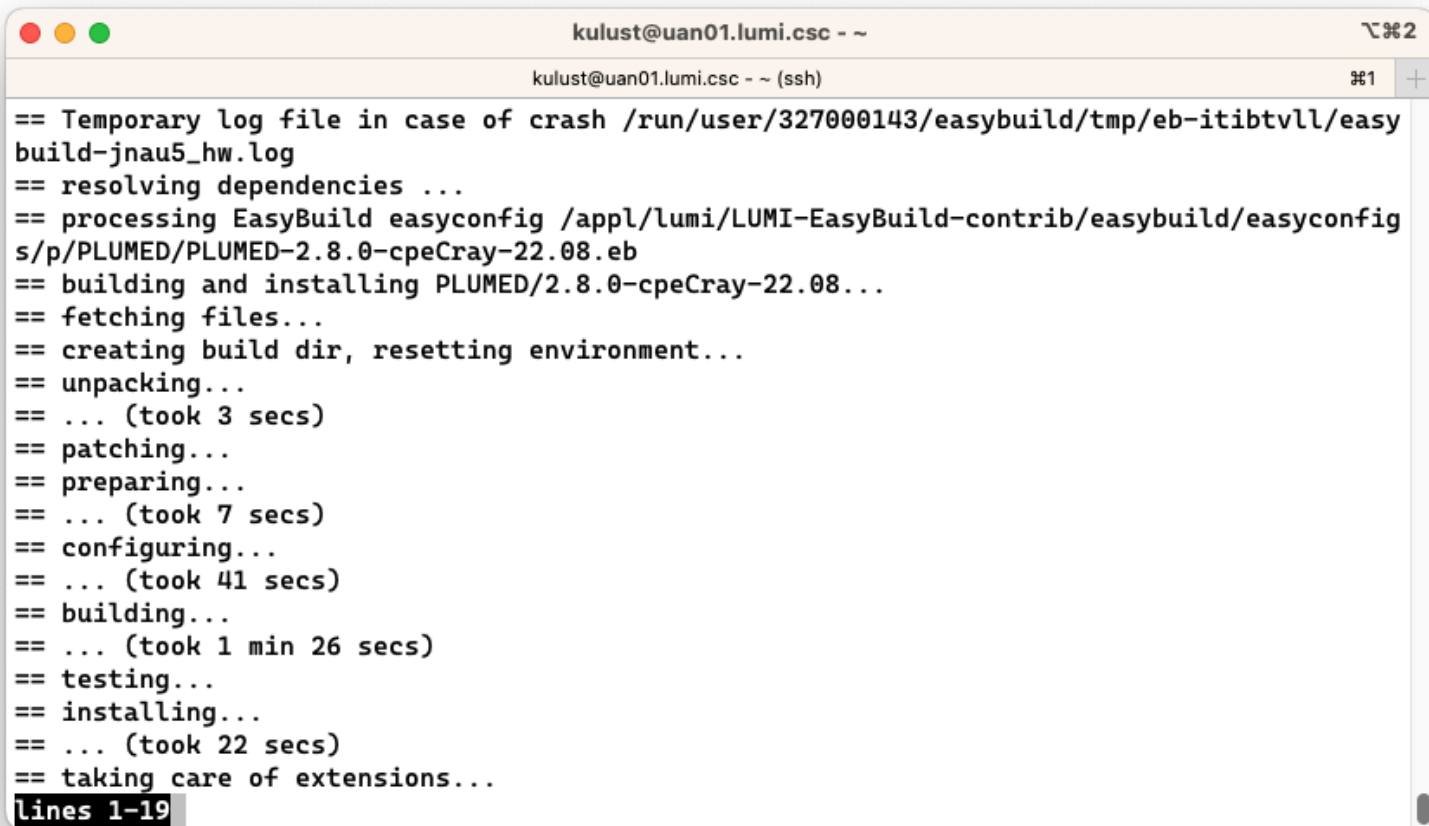
`eb GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb -D`


```
== Temporary log file in case of crash /run/user/327000143/easybuild/tmp/eb-1ckgk1_7/easy
build-tmiw_ajq.log
Dry run: printing build status of easyconfigs and dependencies
CFGS=/appl/lumi
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-common/buildtools/buildtools-22.08-minimal
.eb (module: buildtools/22.08-minimal)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/cpeCray/cpeCray-22.08.eb (module: cpeCra
y/22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-common/syslibs/syslibs-22.08-static.eb (mo
dule: syslibs/22.08-static)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-common/buildtools/buildtools-22.08.eb (mod
ule: buildtools/22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/zlib/zlib-1.2.12-cpeCray-22.08.eb (modul
e: zlib/1.2.12-cpeCray-22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/bzip2/bzip2-1.0.8-cpeCray-22.08.eb (modu
le: bzip2/1.0.8-cpeCray-22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/GSL/GSL-2.7.1-cpeCray-22.08-OpenMP.eb (m
odule: GSL/2.7.1-cpeCray-22.08-OpenMP)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/ICU/ICU-71.1-cpeCray-22.08.eb (module: I
CU/71.1-cpeCray-22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/gzip/gzip-1.12-cpeCray-22.08.eb (module:
lines 1-12
```

LUMI

```
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/gzip/gzip-1.12-cpeCray-22.08.eb (module:
 gzip/1.12-cpeCray-22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/lz4/lz4-1.9.3-cpeCray-22.08.eb (module:
lz4/1.9.3-cpeCray-22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/ncurses/ncurses-6.2-cpeCray-22.08.eb (mo
dule: ncurses/6.2-cpeCray-22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/gettext/gettext-0.21-cpeCray-22.08-minim
al.eb (module: gettext/0.21-cpeCray-22.08-minimal)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/XZ/XZ-5.2.5-cpeCray-22.08.eb (module: XZ
/5.2.5-cpeCray-22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/zstd/zstd-1.5.2-cpeCray-22.08.eb (module
: zstd/1.5.2-cpeCray-22.08)
 * [x] $CFGS/mgmt/ebrepo_files/LUMI-22.08/LUMI-C/Boost/Boost-1.79.0-cpeCray-22.08.eb (mod
ule: Boost/1.79.0-cpeCray-22.08)
 * [ ] $CFGS/LUMI-EasyBuild-contrib/easybuild/easyconfigs/p/PLUMED/PLUMED-2.8.0-cpeCray-2
2.08.eb (module: PLUMED/2.8.0-cpeCray-22.08)
 * [ ] $CFGS/LUMI-EasyBuild-contrib/easybuild/easyconfigs/g/GROMACS/GROMACS-2021.4-cpeCra
y-22.08-PLUMED-2.8.0-CPU.eb (module: GROMACS/2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU)
== Temporary log file(s) /run/user/327000143/easybuild/tmp/eb-1ckgk1_7/easybuild-tmiw_ajq
.log* have been removed.
== Temporary directory /run/user/327000143/easybuild/tmp/eb-1ckgk1_7 has been removed.
lines 12-22/22 (END)
```

`eb GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb -r`
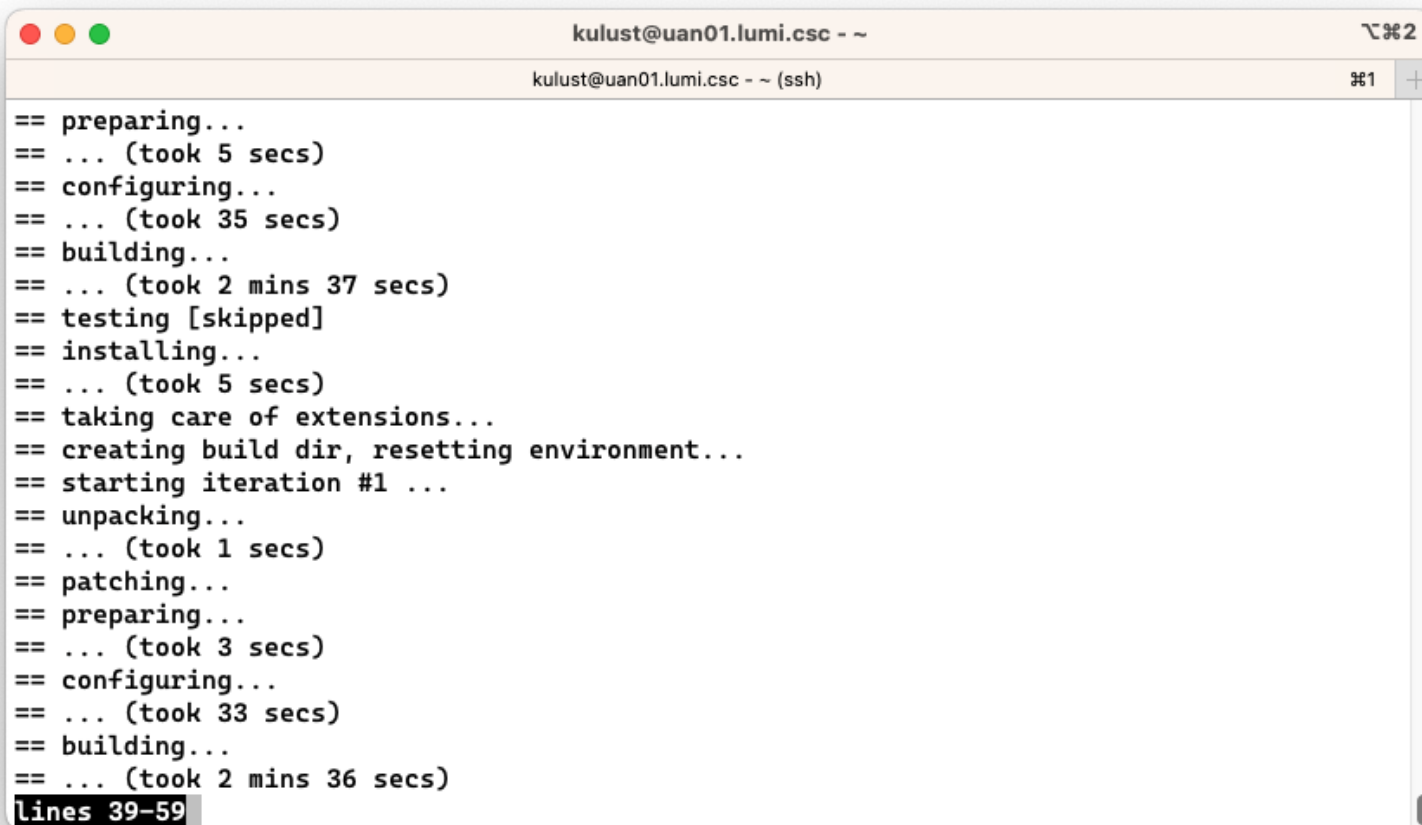

```
== Temporary log file in case of crash /run/user/327000143/easybuild/tmp/eb-itibtvll/easy
build-jnau5_hw.log
== resolving dependencies ...
== processing EasyBuild easyconfig /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfig
s/p/PLUMED/PLUMED-2.8.0-cpeCray-22.08.eb
== building and installing PLUMED/2.8.0-cpeCray-22.08...
== fetching files...
== creating build dir, resetting environment...
== unpacking...
== ... (took 3 secs)
== patching...
== preparing...
== ... (took 7 secs)
== configuring...
== ... (took 41 secs)
== building...
== ... (took 1 min 26 secs)
== testing...
== installing...
== ... (took 22 secs)
== taking care of extensions...
lines 1-19
```

```
== restore after iterating...
== postprocessing...
== sanity checking...
== ... (took 3 secs)
== cleaning up...
== creating module...
== ... (took 2 secs)
== permissions...
== packaging...
== COMPLETED: Installation ended successfully (took 2 mins 49 secs)
== Results of the build can be found in the log file(s) /users/kulust/EasyBuild/SW/LUMI-2
2.08/C/PLUMED/2.8.0-cpeCray-22.08/easybuild/easybuild-PLUMED-2.8.0-20221117.190106.log
== processing EasyBuild easyconfig /appl/lumi/LUMI-EasyBuild-contrib/easybuild/easyconfig
s/g/GROMACS/GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb
== building and installing GROMACS/2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU...
== fetching files...
== creating build dir, resetting environment...
== starting iteration #0 ...
== unpacking...
== ... (took 1 secs)
== patching...
lines 20-38
```

eb GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb -r (3)


```
== preparing...
== ... (took 5 secs)
== configuring...
== ... (took 35 secs)
== building...
== ... (took 2 mins 37 secs)
== testing [skipped]
== installing...
== ... (took 5 secs)
== taking care of extensions...
== creating build dir, resetting environment...
== starting iteration #1 ...
== unpacking...
== ... (took 1 secs)
== patching...
== preparing...
== ... (took 3 secs)
== configuring...
== ... (took 33 secs)
== building...
== ... (took 2 mins 36 secs)
lines 39-59
```
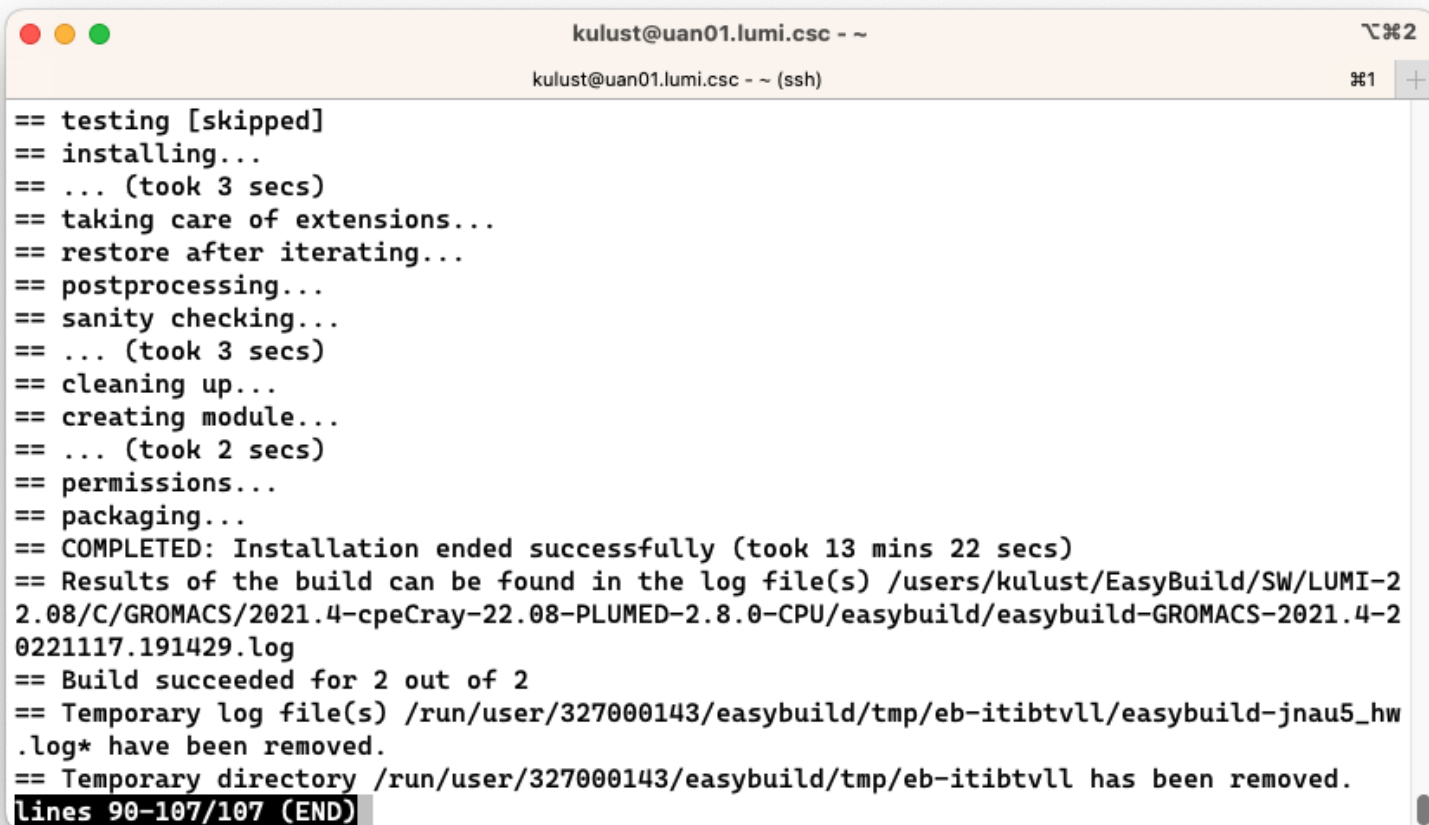
`eb GROMACS-2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU.eb -r (4)`

LUMI

```
== unpacking...
== ... (took 1 secs)
== patching...
== preparing...
== ... (took 3 secs)
== configuring...
== ... (took 33 secs)
== building...
== ... (took 2 mins 33 secs)
== testing [skipped]
== installing...
== ... (took 3 secs)
== taking care of extensions...
== restore after iterating...
== postprocessing...
== sanity checking...
== ... (took 3 secs)
== cleaning up...
== creating module...
== ... (took 2 secs)
== permissions...
lines 81-101
```

L U M I

```
== testing [skipped]
== installing...
== ... (took 3 secs)
== taking care of extensions...
== restore after iterating...
== postprocessing...
== sanity checking...
== ... (took 3 secs)
== cleaning up...
== creating module...
== ... (took 2 secs)
== permissions...
== packaging...
== COMPLETED: Installation ended successfully (took 13 mins 22 secs)
== Results of the build can be found in the log file(s) /users/kulust/EasyBuild/SW/LUMI-2
2.08/C/GROMACS/2021.4-cpeCray-22.08-PLUMED-2.8.0-CPU/easybuild/easybuild-GROMACS-2021.4-2
0221117.191429.log
== Build succeeded for 2 out of 2
== Temporary log file(s) /run/user/327000143/easybuild/tmp/eb-itibtvll/easybuild-jnau5_hw
.log* have been removed.
== Temporary directory /run/user/327000143/easybuild/tmp/eb-itibtvll has been removed.
lines 90-107/107 (END)
```

Step 3: Install the software - Note

- Installing this way is 100% equivalent to an installation in the central software tree. The application is compiled in exactly the same way as we would do and served from the same file systems.

- Note: EasyBuild clears the Lmod user cache so in principle newly installed modules should show up without problems after installation.

  - We've seen rare cases where internal Lmod data structures where corrupt and logging out and in again was needed.

- To manually remove the cache: Remove `$HOME/.lmod.d/.cache`
  ```
  rm -rf $HOME/.lmod.d/.cache
  ```

More advanced work

- You can also install some EasyBuild recipes that you got from support and are in the current directory (preferably one without subdirectories):
  `eb my_recipe.eb -r .`
  - Note the dot after the `-r` to tell EasyBuild to also look for dependencies in the current directory (and its subdirectories)

- In some cases you will have to download the sources by hand, e.g., for VASP, which is then at the same time a way for us to ensure that you have a license for VASP. E.g.,
  - `eb --search VASP`
  - Then from the directory with the VASP sources:
    `eb VASP-6.3.2-cpeGNU-22.08.eb -r .`

More advanced work (2): Repositories

- It is possible to have your own clone of the LUMI-EasyBuild-contrib repo in your `$EBU_USER_PREFIX` subdirectory if you want the latest and greatest before it is in the centrally maintained repository
  - `cd $EBU_USER_PREFIX`
    `git clone https://github.com/Lumi-supercomputer/LUMI-EasyBuild-contrib.git`

- It is also possible to maintain your own repo
  - The directory should be `$EBU_USER_PREFIX/UserRepo` (but of course on GitHub the repository can have a different name)
  - Structure should be compatible with EasyBuild: easyconfig files go in `$EBU_USER_PREFIX/easybuild/easyconfigs`

More advanced work (3): Reproducibility

- EasyBuild will keep a copy of the sources in `$EBU_USER_PREFIX/sources`

- EasyBuild also keeps copies of all installed easyconfig files in two locations:
  - In `$EBU_USER_PREFIX/ebrepo_files`
    - And note that EasyBuild will use this version if you try to reinstall and did not delete this version first!
    - This ensures that the information that EasyBuild has about the installed application is compatible with what's in the module files
  - With the installed software (in `$EBU_USER_PREFIX/SW`) in a subdirectory called easybuild
    This is meant to have all information about how EasyBuild installed the application and to help in reproducing

EasyBuild tips&tricks

- Updating version: Often some trivial changes in the EasyConfig (`.eb`) file
  - Checksums may be annoying: Use `--ignore-checksums` with the `eb` command
- Updating to a new toolchain:
  - Be careful, it is more than changing one number
  - Versions of preinstalled dependencies should be changed and EasyConfig files of other dependencies also checked
- [LUMI Software Library](lumi-supercomputer.github.io/LUMI-EasyBuild-docs) at lumi-supercomputer.github.io/LUMI-EasyBuild-docs
  - For most packages, pointers to the license
  - User documentation gives info about the use of the package, or restrictions
  - Technical documentation aimed at users who want more information about how we build the package

EasyBuild training for advanced users and developers

- EasyBuild web site: easybuild.io

- Generic EasyBuild training materials on easybuilders.github.io/easybuild-tutorial.

- Training for CSC and local support organisations: Most up-to-date version of the training materials on lumi-supercomputer.github.io/easybuild-tutorial.

Containers

This is about containers on LUMI-C and LUMI-G!

- What can they do and what can't they do?

- Getting containers onto LUMI

- Running containers on LUMI

- Enhancements to the LUMI environment to help you

- But remember: LUMI is an HPC infrastructure, not a container cloud!

What do containers not provide?
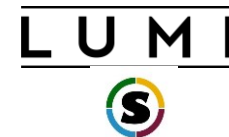
LUMI

- **Full reproducibility of results** is a myth

- **Full portability**: Not every container prepared on your Ubuntu or CentOS cluster or workstation will work on LUMI.

  - Containers that rely on certain hardware, kernel modules and/or kernel versions may fail.

  - Problem cases: High-performance networking (MPI) and GPU (driver version)

- **Performance portability**:

  - A container built from sources on one CPU will not be optimal for another one.

  - Containers built from downloaded binaries may not exploit all architectural features of the CPU.

  - No support for the LUMI interconnect may lead to fall-down to slower protocol that works

But what can they then do on LUMI?
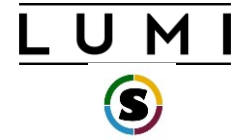
- **Storage manageability:** Lower pressure on the filesystems (for software frameworks that access hundreds of thousands of small files) for better I/O performance and management of your disk file quota.
  - E.g., conda installations are not appreciated straight on the Lustre file system
- **Software installation:** Can be a way to install software with an installation process that is not aware of multi-user HPC systems and is too complicated to recompile.
  - E.g., experiment with software that needs a newer version or ROCm, though with limitations
- **But note:** You're the system administrator of your container, not LUST!

Managing containers

LUMI

- Supported runtimes
  - Docker is **NOT** directly available from user environment
  - Singularity is natively available (as a system command) on the login and compute nodes
- Pulling containers
  - DockerHub and other registries (example: Julia container)
    `singularity pull docker://julia`
  - Singularity uses flat (single) sif file for storing container and pull command makes the conversion
  - Be carefull: cache in `.singularity` dir can easily exhaust your storage quota for larger images
    - May want to set `SINGULARITY_CACHEDIR`

singularity pull docker://julia

```
[lumi][kulust@uan01-1012 container-demo]$ singularity pull docker://julia
INFO:    Converting OCI blobs to SIF format
WARNING: 'nodev' mount option set on /tmp, it could be a source of failure during build process
INFO:    Starting build...
Getting image source signatures
Copying blob 34f65707cdc9 done
Copying blob 517972d95169 done
Copying blob 9e3ea8720c6d done
Copying blob bf4da5f2ad94 done
Copying config 4839902eb6 done
Writing manifest to image destination
Storing signatures
2023/05/12 17:18:31  info unpack layer: sha256:9e3ea8720c6de96cc9ad544dddc695a3ab73f5581c5d954e0504cc4f80
fb5e5c
2023/05/12 17:18:31  warn xattr{etc/gshadow} ignoring ENOTSUP on setxattr "user.rootlesscontainers"
2023/05/12 17:18:31  warn xattr{/tmp/build-temp-2626795503/rootfs/etc/gshadow} destination filesystem doe
s not support xattrs, further warnings will be suppressed
2023/05/12 17:18:33  info unpack layer: sha256:bf4da5f2ad94273f80352cb6898e2347ef78a3570c60ee03d652a6123a
571f70
2023/05/12 17:18:33  warn xattr{var/cache/apt/archives/partial} ignoring ENOTSUP on setxattr "user.rootle
sscontainers"
2023/05/12 17:18:33  warn xattr{/tmp/build-temp-2626795503/rootfs/var/cache/apt/archives/partial} destina
tion filesystem does not support xattrs, further warnings will be suppressed
```

singularity pull docker://julia

singularity pull docker://julia

```
kulust@uan01.lumi.csc - ~/.singularity
kulust@uan01.lumi.csc - ~/.singularity (ssh)

2023/05/12 17:18:38  warn rootless{usr/local/julia/lib/libjulia.so.1} ignoring (usually) harmless EPERM o
n setxattr "user.rootlesscontainers"
2023/05/12 17:18:39  info unpack layer: sha256:517972d951693e767dcac01bd8871495974d4bdab2446521630a3bb1c8
97d0fc
INFO:    Creating SIF file...
[lumi][kulust@uan01-1013 container-demo]$ cd ~/.singularity
[lumi][kulust@uan01-1014 .singularity]$ ls -la
total 12
drwx------  3 kulust pepr_kulust 4096 May 12 17:18 .
drwx------ 28 kulust pepr_kulust 4096 May  9 16:20 ..
drwx------  8 kulust pepr_kulust 4096 May 12 17:18 cache
[lumi][kulust@uan01-1015 .singularity]$ du -h
4.0K    ./cache/shub
175M    ./cache/blob/blobs/sha256
175M    ./cache/blob/blobs
175M    ./cache/blob
4.0K    ./cache/net
4.0K    ./cache/oras
4.0K    ./cache/library
171M    ./cache/oci-tmp
346M    ./cache
346M    .
[lumi][kulust@uan01-1016 .singularity]$
```

Managing containers (2)
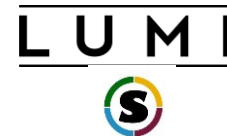
- Building containers
  - Support for building containers is very limited on LUMI: no elevated privileges but also no fakeroot and no user namespaces.
  - You should either pull or copy containers from outside
  - Singularity can build from existing (base) container
  - We plan to provide a set of base LUMI images

Interacting with containers
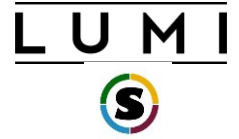
- Accessing a container with the shell command
  `singularity shell container.sif`

- Executing a command in the container with exec
  `singularity exec container.sif uname -a`

- "Running" a container
  `singularity run container.sif`

- Inspecting run definition script
  `singularity inspect --runscript container.sif`

- Accessing host filesystem with bind mounts
  - Singularity will mount `$HOME`, `/tmp`, `/proc`, `/sys`, `/dev` into container by default
  - Use `--bind src1:dest1,src2:dest2` or the `SINGULARITY_BIND` environment variable to mount other host directories (like `/project` or `/appl`)

singularity shell julia_latest.sif

L U M I

```
[lumi][kulust@uan01-1019 container-demo]$ ls /opt
admin-pe  AMD  cray  esmi  modulefiles  rocm  rocm-5.2.3  slingshot
[lumi][kulust@uan01-1020 container-demo]$ singularity shell julia_latest.sif
Singularity> ls /opt
Singularity> cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 11 (bullseye)"
NAME="Debian GNU/Linux"
VERSION_ID="11"
VERSION="11 (bullseye)"
VERSION_CODENAME=bullseye
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
Singularity> exit
exit
[lumi][kulust@uan01-1021 container-demo]$
```

singularity exec julia_latest.sif uname -a

```
singularity run julia_latest.sif
singularity inspect -runscript julia_latest.sif
```

```
[lumi][kulust@uan01-1026 container-demo]$ singularity run julia_latest.sif
               _
       _       _ _(_)_            |  Documentation: https://docs.julialang.org
      (_)     | (_) (_)           |
       _ _   _| |_  __ _          |  Type "?" for help, "]?" for Pkg help.
      | | | | | | |/ _` |         |
      | | |_| | | | (_| |         |  Version 1.9.0 (2023-05-07)
     _/ |\__'_|_|_|\__'_|         |  Official https://julialang.org/ release
    |__/                          |

julia>
[lumi][kulust@uan01-1027 container-demo]$ singularity inspect --runscript julia_latest.sif
#!/bin/sh
OCI_ENTRYPOINT='"docker-entrypoint.sh"'
OCI_CMD='"julia"'

# When SINGULARITY_NO_EVAL set, use OCI compatible behavior that does
# not evaluate resolved CMD / ENTRYPOINT / ARGS through the shell, and
# does not modify expected quoting behavior of args.
if [ -n "$SINGULARITY_NO_EVAL" ]; then
        # ENTRYPOINT only - run entrypoint plus args
        if [ -z "$OCI_CMD" ] && [ -n "$OCI_ENTRYPOINT" ]; then
                set -- 'docker-entrypoint.sh' "$@"
```

Running containers on LUMI

- Use SLURM to run containers on compute nodes

- Use srun to execute MPI containers
  ```
  srun singularity exec --bind ${BIND_ARGS} \
  ${CONTAINER_PATH} my_mpi_binary ${APP_PARAMS}
  ```

- **Be aware your container must be compatible with Cray MPI** (MPICH ABI compatible)
  - Configure suggestion: see next slide

- Open MPI based containers need workarounds and are not well supported on LUMI at the moment (and even more problematic for the GPU)

Environment enhancements
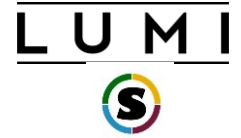
- LUMI specific tools for container interaction provided as modules
- **singularity-bindings/system** (available via easyconfig)
  - Sets the environment to use Cray MPICH provided outside the container
  - Requires a LUMI software stack
  - Use EasyBuild-user module and eb --search singularity-bindings to find the easyconfig or copy from our LUMI Software Library web site
  - Provides basic mount points for using host MPI in the container setting `SINGULARITY_BIND` and `SINGULARITY_LD_LIBRARY_PATH`
- **lumi-vnc** (LUMI and CrayEnv software stacks)
  - Provides basic VNC virtual desktop for interacting with graphical interfaces via a web browser or VNC client

Environment enhancements (2)

- **lumi-container-wrapper** (LUMI and CrayEnv software stacks)
  - Provides wrappers to encapsulate your custom environment in a container
  - Supports conda and pip environments
  - Helps with quota on the number of files in your project and I/O performance
  - With pip: Python provided by the cray-python module (so there is an optimised NumPy etc.)
- **cotainr** (LUMI and CrayEnv software stacks)
  - Another tool to pack conda installations in a singularity container, but more container-like interface than lumi-container-wrapper
  - So you'd use the singularity commands as shown on earlier slides to run

lumi-container-wrapper (1)


```
                    *  ,/  (              *
            *                   ,/   \   |  /
         *                     /      \  * || |
                 *        *   ,-   (    )| || |
   *    *   *  The Supercomputer of the North  * | `\_|  __ /_| || |
       **              *          * *    _____)\__)__)
    .********----------******------******----------****************.
    |User guide and support _____|
    |          https://docs.lumi-supercomputer.eu              |
 *  |          https://lumi-supercomputer.eu/user-support      |
** `--*****------------------***--------------------*****-------'
[lumi][kulust@uan04-1001 ~]$ cd Tykky-demo/
[lumi][kulust@uan04-1002 Tykky-demo]$ ls
conda-cont-1   env.yml
[lumi][kulust@uan04-1003 Tykky-demo]$ cat env.yml
channels:
  - conda-forge
dependencies:
  - python=3.8.8
  - scipy
  - nglview
[lumi][kulust@uan04-1004 Tykky-demo]$ module load LUMI/22.12 lumi-container-wrapper
[lumi][kulust@uan04-1005 Tykky-demo]$
```

lumi-container-wrapper (2)

L U M I

```
kulust@uan04.lumi.csc - ~/Tykky-demo

kulust@uan04.lumi.csc - ~/Tykky-demo (ssh)

[lumi][kulust@uan04-1004 Tykky-demo]$ module load LUMI/22.12 lumi-container-wrapper
[lumi][kulust@uan04-1005 Tykky-demo]$ conda-containerize new --prefix ./conda-cont-1 env.yml
[ INFO ] Constructing configuration
[ INFO ] Using /tmp/kulust/cw-YSVL4M as temporary directory
[ INFO ] Fetching container docker://opensuse/leap:15.4
[ INFO ] Running installation script
[ INFO ] Using miniconda version Miniconda3-latest-Linux-x86_64
[ INFO ] Installing miniconda
=========================
PREFIX=/LUMI_TYKKY_4EJoer8/miniconda                                          P
REFIX=/LUMI_TYKKY_4EJoer8/miniconda                                           P

Preparing transaction: ...working... done
Executing transaction: ...working... done
installation finished.
WARNING:
    You currently have a PYTHONPATH environment variable set. This may cause
    unexpected behavior when running the Python interpreter in Miniconda3.
    For best results, please verify that your PYTHONPATH only points to
    directories of packages that are compatible with the Python interpreter
    in Miniconda3: /LUMI_TYKKY_4EJoer8/miniconda
=========================
[ INFO ] Creating env, full log in /tmp/kulust/cw-YSVL4M/build.log
```

lumi-container-wrapper (3)


```
==========================
[ INFO ] Running user supplied commands
[ INFO ] Creating sqfs image
Parallel mksquashfs: Using 8 processors
Creating 4.0 filesystem on _deploy/img.sqfs, block size 131072.
[=====================================================================\] 37447/37447 100%

Exportable Squashfs 4.0 filesystem, gzip compressed, data block size 131072
        compressed data, compressed metadata, compressed fragments,
scipy-1.compressed xattrs, compressed ids3    |   63%
        duplicates are removed
Filesystem size 521302.16 Kbytes (509.08 Mbytes)
executin33.62% of uncompressed filesystem size (1550478.89 Kbytes)
Inode table size 408231 bytes (398.66 Kbytes) |   1%
        23.20% of uncompressed inode table size (1759978 bytes)
Directory table size 578457 bytes (564.90 Kbytes)
        41.52% of uncompressed directory table size (1393078 bytes)
Number of duplicate files found 5545
Number of inodes 37685
Number of files 27719
Number of fragments 1698
Number of symbolic links  4872
Number of device nodes 0
```

lumi-container-wrapper (4)

```
                    kulust@uan04.lumi.csc - ~/Tykky-demo                          ⌥⌘2
                  kulust@uan04.lumi.csc - ~/Tykky-demo (ssh)                      ⌘1
Directory table size 578457 bytes (564.90 Kbytes)
         41.52% of uncompressed directory table size (1393078 bytes)
Number of duplicate files found 5545
Number of inodes 37685
Number of files 27719
Number of fragments 1698
Number of symbolic links  4872
Number of device nodes 0
Number of fifo nodes 0
Number of socket nodes 0
Number of directories 5094
Number of ids (unique uids + gids) 1
Number of uids 1
         kulust (327000143)
Number of gids 1
         pepr_kulust (327000143)
[ INFO ] Creating wrappers
[ INFO ] Installing to ./conda-cont-1
[ INFO ] Done, duration: 263s
[ INFO ] Program has been installed to ./conda-cont-1
              To use add the bin folder to your path e.g:
              export PATH="/users/kulust/Tykky-demo/conda-cont-1/bin:$PATH"
[lumi][kulust@uan04-1006 Tykky-demo]$
```

lumi-container-wrapper (5)


```
[ INFO ] Program has been installed to ./conda-cont-1
         To use add the bin folder to your path e.g:
         export PATH="/users/kulust/Tykky-demo/conda-cont-1/bin:$PATH"
[lumi][kulust@uan04-1006 Tykky-demo]$ ls conda-cont-1/
_bin  bin  common.sh  container.sif  img.sqfs  share
[lumi][kulust@uan04-1007 Tykky-demo]$ ls conda-cont-1/bin
2to3                 ipython3             lzmadec              python3.8            wish
2to3-3.8             jupyter              lzmainfo             python3.8-config     wish8.6
captoinfo            jupyter-kernel       lzmore               python3-config       x86_64-conda_cos6-linux-gnu-ld
clear                jupyter-kernelspec   ncurses6-config      reset                x86_64-conda-linux-gnu-ld
c_rehash             jupyter-migrate      ncursesw6-config     sqlite3              xz
curve_keygen         jupyter-run          nglview              sqlite3_analyzer     xzcat
_debug_exec          jupyter-troubleshoot normalizer           tabs                 xzcmp
_debug_shell         list-packages        openssl              tclsh                xzdec
f2py                 lzcat                pip                  tclsh8.6             xzdiff
f2py3                lzcmp                pip3                 tic                  xzegrep
f2py3.8              lzdiff               pydoc                toe                  xzfgrep
idle3                lzegrep              pydoc3               tput                 xzgrep
idle3.8              lzfgrep              pydoc3.8             tset                 xzless
infocmp              lzgrep               pygmentize           unlzma               xzmore
infotocap            lzless               python               unxz
ipython              lzma                 python3              wheel
[lumi][kulust@uan04-1008 Tykky-demo]$
```

lumi-container-wrapper (6)

```
2to3              ipython3           lzmadec            python3.8           wish
2to3-3.8          jupyter            lzmainfo           python3.8-config    wish8.6
captoinfo         jupyter-kernel     lzmore             python3-config      x86_64-conda_cos6-linux-gnu-ld
clear             jupyter-kernelspec ncurses6-config    reset               x86_64-conda-linux-gnu-ld
c_rehash          jupyter-migrate    ncursesw6-config   sqlite3             xz
curve_keygen      jupyter-run        nglview            sqlite3_analyzer    xzcat
_debug_exec       jupyter-troubleshoot normalizer       tabs                xzcmp
_debug_shell      list-packages      openssl            tclsh               xzdec
f2py              lzcat              pip                tclsh8.6            xzdiff
f2py3             lzcmp              pip3               tic                 xzegrep
f2py3.8           lzdiff             pydoc              toe                 xzfgrep
idle3             lzegrep            pydoc3             tput                xzgrep
idle3.8           lzfgrep            pydoc3.8           tset                xzless
infocmp           lzgrep             pygmentize         unlzma              xzmore
infotocap         lzless             python             unxz
ipython           lzma               python3            wheel
[lumi][kulust@uan04-1008 Tykky-demo]$ cd conda-cont-1/bin
[lumi][kulust@uan04-1009 bin]$ ./python3
Python 3.8.8 | packaged by conda-forge | (default, Feb 20 2021, 16:22:27)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy
>>>
```

Container limitations on LUMI
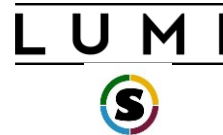
- Containers use the host's operating system kernel which may be different from your system. Containers do not abstract hardware.

- A generic container may not offer sufficiently good support for the SlingShot 11 interconnect on LUMI and fall back to TCP sockets resulting in poor performance, or not work at all.
  - Solution by injecting Cray MPICH, but only for containers with ABI compatibility with MPICH.
  - Distributed AI: Need to inject the proper RCCL plugin.

- Only very limited support to build containers on LUMI due to security concerns.