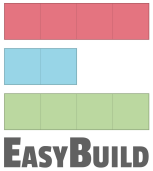
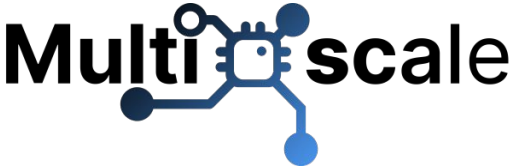
Building Software on top of & Contributing to EESSI

EUROPEAN ENVIRONMENT FOR SCIENTIFIC SOFTWARE INSTALLATIONS





Mon 4 May 2026

Lara Peeters & Kenneth Hoste (Ghent University)

Webinar series: Different aspects of EESSI

5 Mondays in a row April-May-June 2026

https://www.eessi.io/docs/training-events/2026/webinar-series-2026Q2

- 27 April: Introduction to EESSI - **recording available**
- 4 May: **Building software on top of EESSI + contributing to EESSI**
- 11 May: Using EESSI for Continuous Integration
- 18 May: Introduction to CernVM-FS
- 1 June: Using EESSI as a base for a central software stack

More info and registration →

Q&A via EESSI Slack

Please join the `#webinar-series-2026q2` channel in the EESSI Slack for questions and discussion

Step 1) **Join the EESSI Slack,** see “Slack channel” link at https://eessi.io

Step 2) **Join `#webinar-series-2026q2` channel** in EESSI Slack

(direct link: https://eessi-hpc.slack.com/archives/C0AP84QKU05)

Agenda

- A detailed look at the EESSI software layer
- Building software on top of EESSI
  - Using EasyBuild, via `EESSI-extend`
  - Manually, with the help of `buildenv`
  - Using Spack
- Contributing software to EESSI

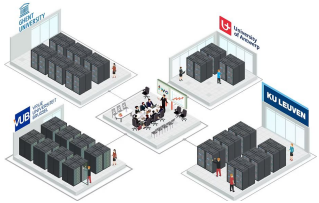
EESSI as shared software stack for HPC, cloud, and beyond

GROMACS
TensorFlow
python
OpenFOAM®
R
Bioconductor
OPEN SOURCE SOFTWARE FOR BIOINFORMATICS
QUANTUMESPRESSO
PyTorch
WRF
OpenCV
PETSc
and much more software...

EESSI

Raspberry Pi

Laptops, PCs



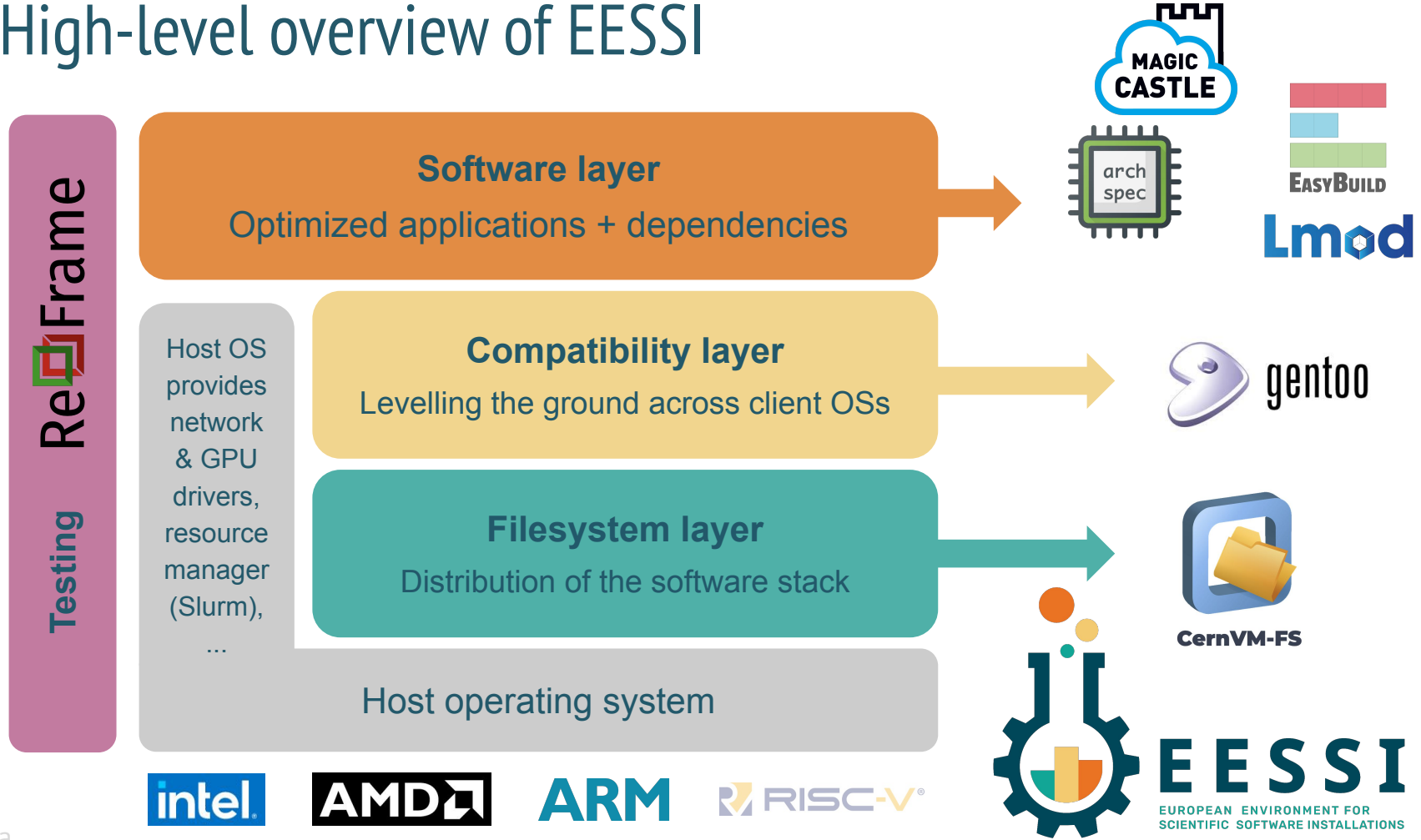
VLAAMS SUPERCOMPUTER CENTRUM

Regional/national systems

Cloud
aws
eosc

EuroHPC Joint Undertaking

High-level overview of EESSI



High-level overview of EESSI

Software layer

Optimized applications + dependencies

In this webinar we will focus on the software layer,

More specifically on *how* the software installations

that are provided by it there were built,

and how you can build software on top.

CernVM-FS

EESSI

EUROPEAN ENVIRONMENT FOR SCIENTIFIC SOFTWARE INSTALLATIONS

Overview of available software

Currently more than 2,000 software installations available per supported CPU target via `software.eessi.io` CernVM-FS repository, increasing every week

- 15 supported CPU targets (x86_64 + Arm) + RISC-V actively being explored
- **~750 different software packages** + another ~3,000 extensions (Python packages, R libraries, ...)
- ~30,000 software installations in total
- Including GROMACS, LAMMPS, OpenFOAM, PyTorch, R, QuantumESPRESSO, TensorFlow, WRF, ...
- EESSI 2023.06 provides software built with `foss/2023a` and `foss/2023b` toolchains
- EESSI 2025.06 provides software built with `foss/2024a`, `foss/2025a`, `foss/2025b` toolchains

Overview of available software

An up-to-date overview of available software is available in the EESSI documentation: https://www.eessi.io/docs/available_software

Software available in EESSI

Overview of software available in EESSI's production repository `software.eessi.io`.

752 unique software projects (+ 3007 unique extensions)

name:GROMACS

GROMACS

(more details)

https://www.gromacs.org

GROMACS is a versatile package to perform molecular dynamics, i.e. simulate the Newtonian equations of motion for systems with hundreds to millions of particles. This is a CPU only build, containing both MPI and threadMPI binaries for both single and double precision. It also contains the gmxapi extension for the single precision MPI build.

Available in EESSI versions: 2023.06 2025.06

Supported CPU families: AMD Intel Arm

Supported GPU families: NVIDIA

GROMACS

GROMACS is a versatile package to perform molecular dynamics, i.e. simulate the Newtonian equations of motion for systems with hundreds to millions of particles.

This is a GPU enabled build, containing both MPI and threadMPI binaries.

It also contains the gmxapi extension for the single precision MPI build.

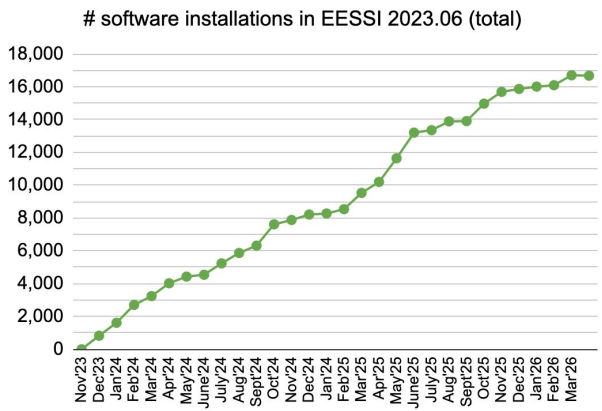
homepage: https://www.gromacs.org

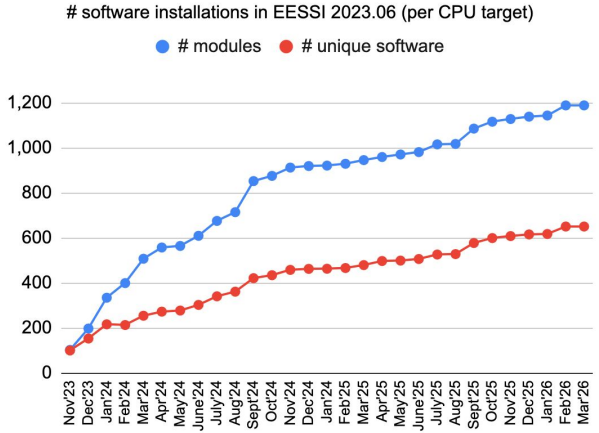
Available installations

GROMACS version	Supported CPU targets	Supported GPU targets	EESSI version	Module
2025.4	generic: aarch64, x86_64 Arm: a64fx, neoverse_n1, neoverse_v1, nvidia/grace AMD: zen2, zen3, zen4 Intel: haswell, skylake_avx512, sapphirerapids, icelake, cascadelake	*(none)*	2025.06	GROMACS/2025.4-foss-2025b
2025.2	generic: aarch64, x86_64 Arm: a64fx, neoverse_n1, neoverse_v1, nvidia/grace AMD: zen2, zen3, zen4 Intel: haswell, skylake_avx512, sapphirerapids, icelake, cascadelake	*(none)*	2025.06	GROMACS/2025.2-foss-2025a
2024.4	generic: aarch64, x86_64 Arm: neoverse_n1,	NVIDIA: cc70, cc80, cc90	2023.06	GROMACS/2024.4-foss-2023b-

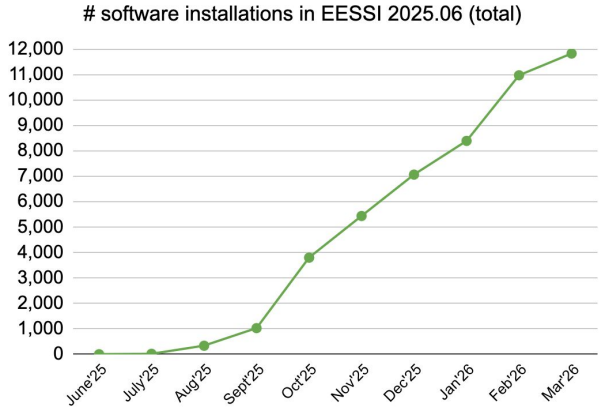
Overview of available software

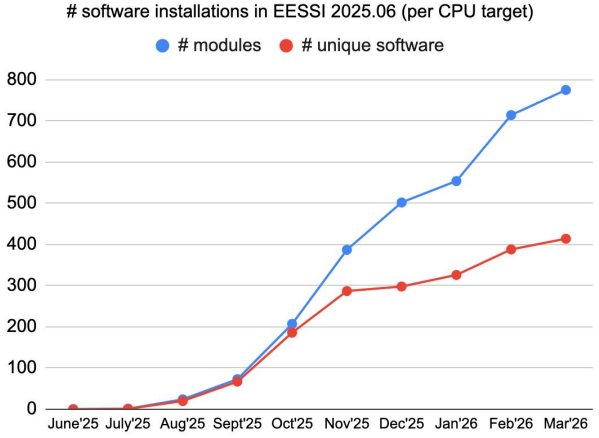
2023.06





2025.06





Supported CPU targets

https://eessi.io/docs/software_layer/cpu_targets

- AMD, Intel (`x86_64`)
  - 5 generations of Intel CPUs: Haswell, Skylake, Cascade Lake, Ice Lake, Sappire Rapids
  - 4 generations of AMD CPUs: Zen2 (Rome), Zen 3 (Milan), Zen4 (Genoa), Zen5 (Turin)
- Arm (`aarch64`)
  - 4 different microarchitectures: Fujitsu A64FX, NVIDIA Grace, Neoverse N1 + V1
- Plus a generic fallback for both `x86_64` and `aarch64`
- RISC-V CPUs are work-in-progress
  - Exploration ongoing via RISC-V-specific subdirectory of `dev.eessi.io` repository

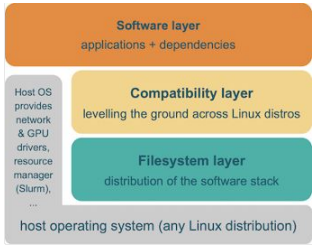
Supported GPU targets

https://eessi.io/docs/software_layer/gpu_targets

- NVIDIA (CUDA)
  - Different generations of NVIDIA GPUs, based on CUDA Compute Capability (CC), see also https://developer.nvidia.com/cuda/gpus
  - In EESSI 2023.06: CC 7.0 (V100, T4), 8.0 (A100 & co, incl. L40), 9.0 (H100/H200)
  - In EESSI 2025.06: CC 7.0, 8.0, 9.0, 10.0 (B100/B200/B300), 12.0 (B10, RTX PRO Blackwell)
  - GPU driver libraries (include `libcuda.so`) must be exposed to EESSI
  - See https://www.eessi.io/docs/site_specific_config/gpu
- AMD (ROCm)
  - Work-in-progress, initial support included in EasyBuild 5.3.0 (mid April 2026)
  - PR for easyconfigs for ROCm 6.4.1 merged, will be included in next EasyBuild release

See also *NVIDIA + AMD GPU support in EESSI* talk @ EUM'26 (slides, recording)

The trick behind the EESSI software layer



Software installations only link to **libraries from compatibility layer or other installations in software layer**, *not* from host (Linux) operating system.

- Because of this, these software installations work on any Linux distribution
- This is ensured via *RPATH linking*
- Build tools (like CMake) need to be made aware of alternative system root (sysroot)
- Some exceptions need to be made: GPU driver libraries must be provided by host OS
- *"Containers without the containing"*

Automated procedure via EESSI build-deploy bots

- Software installations deployed in EESSI are built by the EESSI build-deploy bots (using EasyBuild), see also https://eessi.io/docs/bot
- Semi-automatic installation procedure (human in the loop)
- Via pull request to `EESSI/software-layer` repository
- Contributors do not need access to all CPU targets supported by EESSI
- More on this in the last part of this webinar…

Example: `gmx` binary for GROMACS in EESSI

- Let's inspect the `gmx` binary in a GROMACS installation up close…
- We'll use standard tools like `readelf` and `ldd` to show that:
  - RPATH section of binary specifies hardcoded list of paths for libraries
  - `$LD_LIBRARY_PATH` is irrelevant (actually, only used as a fallback)
  - All libraries used by `gmx` binary come from EESSI compatibility layer or dependencies provided via EESSI software layer
- Everything shown is standard on Linux, no tricks up our sleeve

Preparing environment to inspect `gmx` binary for GROMACS

- Initialize shell for using EESSI version 2023.06:

```
source /cvmfs/software.eessi.io/versions/2023.06/init/lmod/bash
```

- Find and load a GROMACS module:

```
module avail GROMACS

module load GROMACS/2024.4-foss-2023b
```

- Check path to `gmx` binary:

```
command -v gmx
```

(or `which gmx`)

Inspect RPATH section of `gmx` binary with `readelf`

`readelf -d` command reveals details for a specific binary:

```
readelf -d $(command -v gmx)
```

- `(RPATH)` section specifies which paths should be considered first when loader looks for libraries that are required by `gmx` binary
- `(NEEDED)` entries specify which libraries are required, which themselves may require additional libraries in turn

Inspect RPATH section of `gmx` binary with `readelf`

There are different types of paths in (RPATH) section:

- Paths to **subdirectories of `host_injections`** (`/rpath_overrides/`)

  which allow to inject specific libraries (via a path controlled by system administrators)
- Paths for **libraries in software installation directory** itself (like `/lib`)
- Paths **relative to path of binary** itself (`$ORIGIN/...`)
- Paths to libraries provided by **dependencies in EESSI software layer** (`/software/`)
- Paths to libraries provided by **EESSI compatibility layer** (`/compat/`)

Inspect libraries used by `gmx` binary with `ldd`

`ldd` command reveals libraries that are used when launching `gmx` binary:

```
ldd $(command -v gmx)
```

All libraries are provided either by EESSI compatibility layer or software layer.

Some noteworthy ones:

GROMACS' own library (`libgromacs.so`), provided by the GROMACS software installation directory:

```
libgromacs.so.10 => /cvmfs/software.eessi.io/.../software/GROMACS/.../lib/libgromacs.so.10 (0x000014f646e00000)
```

MPI library (`libmpi.so`), provided by OpenMPI dependency in EESSI software layer:

```
libmpi.so.40 => /cvmfs/software.eessi.io/.../software/OpenMPI/.../lib64/libmpi.so.40 (0x000014f646a00000)
```

GNU C library (glibc, `libc.so`), provided by EESSI compat layer (*not* by host OS):

```
libc.so.6 => /cvmfs/software.eessi.io/.../compat/.../lib/../lib64/libc.so.6 (0x000014f646425000)
```

Building software on top of EESSI

To build software on top of EESSI, we have to make sure that:

- No libraries from host OS are relied on, so that binaries are “portable” across Linux systems
- Path to required libraries is “hardcoded” into binaries and libraries, using RPATH linking.

This is a technical challenge, and is different from how people usually build software:

- At build time, specify library paths via `-L` option in compiler command or via `$LIBRARY_PATH`
- At run time, rely on standard system paths (`/usr/lib`) or `$LD_LIBRARY_PATH`

This can’t be relied on when building software to be used in or on top of EESSI (too brittle).

Building software on top of EESSI: options

There are different options to build software on top of EESSI:

- Using **EasyBuild**
  - Should be configured correctly for building on top of EESSI
- Using a "**manual**" build procedure
  - Need to make sure that compatibility layer is used and that RPATH linking is done
- Using **Spack**
  - Can use use software provided by EESSI as "external packages",
    for compiler, dependencies

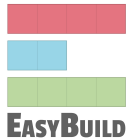
Building software on top of EESSI with EasyBuild

Over the last couple of years, significant enhancements have been made in EasyBuild to make it easier to build software such that it can be included in EESSI, including:

- Improve support for RPATH linking, and make it the default in EasyBuild v5.0
- `--sysroot` configuration option, and making various easyblocks aware of it
- Support for easystack files
- `--sanity-check-only` option
- `--from-commit` option, which is more reproducible than `--from-pr`
- CUDA sanity check

For a recent introduction to EasyBuild, see *Introduction to EasyBuild* webinar (spring 2025)

Building software on top of EESSI with `EESSI-extend`



- An `EESSI-extend` module is provided by EESSI which loads + pre-configures EasyBuild
- Very useful when doing local installations on top of EESSI with EasyBuild, or testing
- Basic usage:
  - Initialize EESSI
  - (optional) Set `$EESSI_*_INSTALL` environment variable to control installation path
  - Load `EESSI-extend` module
  - Check EasyBuild configuration (`eb --show-config`)
  - Provide one or more easyconfig files to EasyBuild to install software on top of EESSI
- The EESSI build-deploy bots also use `EESSI-extend`, so this mechanism is very well tested!
- Detailed documentation available at https://eessi.io/docs/using_eessi/building_on_eessi

Building software on top of EESSI with `EESSI-extend`

Configuring EESSI-extend before loading via `$EESSI_*_INSTALL` env. variables

- `$EESSI_USER_INSTALL` to specify that **user-specific** installation prefix that should be used
    - Permissions are set such that directories are only readable by user (umask 077)
- `$EESSI_PROJECT_INSTALL` to specify that **project** installation prefix that should be used
    - Permissions set accordingly: GID bit, group writable installation directories, umask 002, …
- `$EESSI_SITE_INSTALL` can be set to help with **site-wide** installations on top of EESSI
- `$EESSI_CVMFS_INSTALL` can be set to specify that installation should be done in **CernVM-FS repo**

See also https://eessi.io/docs/using_eessi/building_on_eessi/#using-the-eessi-extend-module

Building software on top of EESSI with `EESSI-extend`

Hands-on demo (EESSI 2025.06): installing scikit-bio with `foss/2025a`

`scikit-bio-0.7.1.post1-foss-2025a.eb` is already available in EasyBuild 5.3.0, so:

```
source /cvmfs/software.eessi.io/versions/2025.06/init/lmod/bash
export EESSI_USER_INSTALL=$HOME/EESSI-2025.06
mkdir -p $EESSI_USER_INSTALL
module load EESSI-extend
eb --show-config
eb scikit-bio-0.7.1.post1-foss-2025a.eb
```

Building software on top of EESSI with `EESSI-extend`

Hands-on demo (EESSI 2025.06): installing latest scikit-bio (0.7.2) with `foss/2025b`

- No easyconfig file for scikit-bio 0.7.2 with foss/2025b available in EasyBuild yet, so had to create one first...
- Open pull request to EasyBuild: https://github.com/easybuilders/easybuild-easyconfigs/pull/25908
- Can test using `EESSI-extend` and `eb --from-pr`:

```
source /cvmfs/software.eessi.io/versions/2025.06/init/lmod/bash
export EESSI_USER_INSTALL=$HOME/EESSI-2025.06
module load EESSI-extend
eb --from-pr 25908 --robot
```

Building software on top of EESSI manually, using `buildenv`

Using EasyBuild to build/install software of top is not required, there are other options.

However, you should make sure that:

- RPATH linking is used, or you need to set `$LD_LIBRARY_PATH`
- Compilers and build tools are aware of the alternate sysroot (compatibility layer)

To facilitate this, EESSI provides a `buildenv` module for each (`foss`) toolchain which:

- Sets up the build environment similar to what EasyBuild uses (`$CC`, `$CFLAGS`, etc.)
- Injects RPATH wrappers for the compiler and linker commands (`gcc`, `g++`, `gfortran`, `ld`, ...)

Detailed documentation available at https://eessi.io/docs/using_eessi/building_on_eessi

Building software on top of EESSI manually

Hands-on demo: Manual build, with the help of `buildenv`

```
source /cvmfs/software.eessi.io/versions/2025.06/init/lmod/bash
module load buildenv/default-foss-2025b
module load libpng/1.6.50-GCCcore-14.3.0
git clone https://github.com/R-Gerard/libpng-demo
cd libpng-demo
make
readelf -d target/pngdemo
ldd target/pngdemo
```

Using Spack to install software on top of EESSI

- Spack can be made aware of software installations provided by EESSI,

  via a `packages.yaml` file that specifies them as “external packages”

- As a result, Spack can resolve dependencies via EESSI, rather than installing them again

- Proof-of-concept was implemented by Loris Ercole (CECAM) using QuantumESPRESSO,

  see https://www.eessi.io/docs/blog/2026/02/05/Spack-on-top-of-EESSI-best-of-both-worlds

- Scripts and detailed setup instructions available via https://github.com/lorisercole/spood

Using Spack to install software on top of EESSI

Hands-on demo - Part 1: Set up Spack *(note: patched version!)*

See also https://github.com/lorisercole/spood#installation--use

```
git clone https://github.com/lorisercole/spack
cd spack
git checkout eessi
cd -
. spack/share/spack/setup-env.sh
spack --version
```

Using Spack to install software on top of EESSI

Hands-on demo - Part 2: Initialize EESSI, get `spood`, run demo

```
git clone https://github.com/lorisercole/spood

cd spood

source /cvmfs/software.eessi.io/versions/2023.06/init/lmod/bash

./quick_start.sh
```

Note: Here be dragons, see also https://github.com/lorisercole/spood/issues/2

Using Spack to install software on top of EESSI

Hands-on demo - Example output of successful run

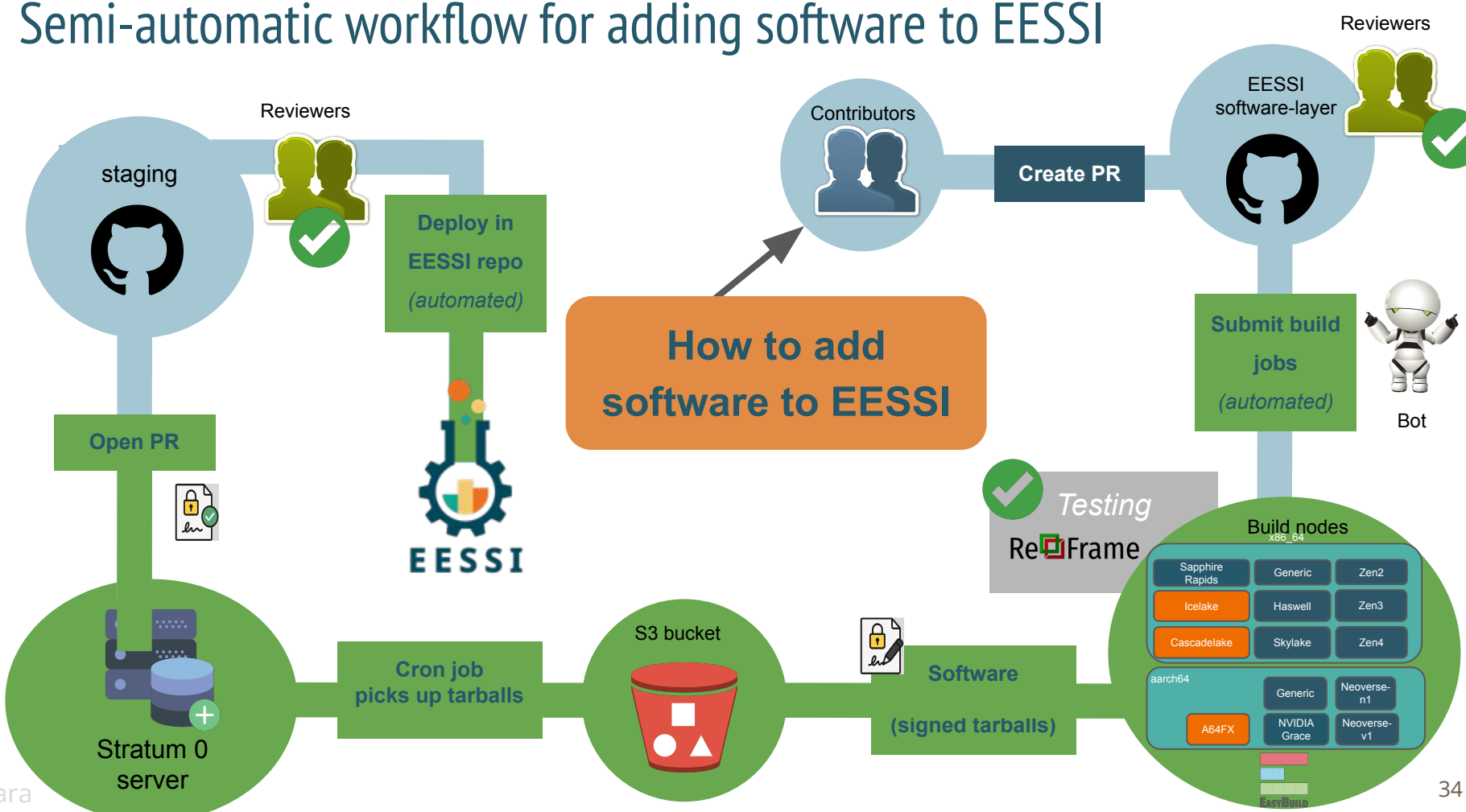
```
==> Installing quantum-espresso-7.5-zxky252cemgoipfhhqoolnlow2s7qfrd [17/17]
    [100%]   72.48 MB @   21.0 MB/s
==> No patches needed for quantum-espresso
==> quantum-espresso: Executing phase: 'cmake'
==> quantum-espresso: Executing phase: 'build'
==> quantum-espresso: Executing phase: 'install'
==> quantum-espresso: Successfully installed quantum-espresso-7.5-zxky252cemgoipfhhqoolnlow2s7qfrd
  Stage: 5.73s.  Cmake: 11.61s.  Build: 3m 4.04s.  Install: 1.35s.  Post-install: 1.04s.  Total: 3m 23.88s
[+] /tmp/vsc40023/EESSI-webinar/spood/demo_spood/opt/linux-cascadelake/quantum-espresso-7.5-zxky252cemgoipfhhqoolnlow2s7qfrd

 5. Verify the installation:
   $ ldd /tmp/vsc40023/EESSI-webinar/spood/demo_spood/opt/linux-cascadelake/quantum-espresso-7.5-zxky252cemgoipfhhqoolnlow2s7q

        linux-vdso.so.1 (0x00007ffc0293e000)
        libfftw3.so.3 => /cvmfs/software.eessi.io/versions/2023.06/software/linux/x86_64/intel/haswell/software/FFTW/3.3.10-GC
        libfftw3_omp.so.3 => /cvmfs/software.eessi.io/versions/2023.06/software/linux/x86_64/intel/haswell/software/FFTW/3.3.1
        libgomp.so.1 => /tmp/vsc40023/EESSI-webinar/spood/demo_spood/opt/linux-cascadelake/gcc-runtime-13.2.0-vvcl5hwifb2atnvu
        libgfortran.so.5 => /tmp/vsc40023/EESSI-webinar/spood/demo_spood/opt/linux-cascadelake/gcc-runtime-13.2.0-vvcl5hwifb2a
        libm.so.6 => /cvmfs/software.eessi.io/versions/2023.06/compat/linux/x86_64/lib64/libm.so.6 (0x000014efe6520000)
        libmvec.so.1 => /cvmfs/software.eessi.io/versions/2023.06/compat/linux/x86_64/lib64/libmvec.so.1 (0x000014efe6108000)
        libopenblas.so.0 => /cvmfs/software.eessi.io/versions/2023.06/software/linux/x86_64/intel/haswell/software/OpenBLAS/0.
        libgcc_s.so.1 => /tmp/vsc40023/EESSI-webinar/spood/demo_spood/opt/linux-cascadelake/gcc-runtime-13.2.0-vvcl5hwifb2atnv
        libc.so.6 => /cvmfs/software.eessi.io/versions/2023.06/compat/linux/x86_64/lib64/libc.so.6 (0x000014efe502f000)
        /cvmfs/software.eessi.io/versions/2023.06/compat/linux/x86_64/lib64/ld-linux-x86-64.so.2 (0x000014efe6912000)

============  End of Demo  ============
```

Contributing software to EESSI

To contribute additional software to EESSI, the following must be taken into account:

- An easyconfig file must be available in EasyBuild to install the software
  - Either in the latest EasyBuild release, or in a (merged) pull request to EasyBuild
- A pull request must be opened to the `EESSI/software-layer` repository
- The EESSI contribution policy must be followed
  - See https://eessi.io/docs/adding_software/contribution_policy
  - Must be redistributable software (like open source), must work on all CPU targets, must be built by EESSI bot, should be recent software version and use recent toolchain, ...

Semi-automatic workflow for adding software to EESSI



Contributing software to EESSI, step-by-step

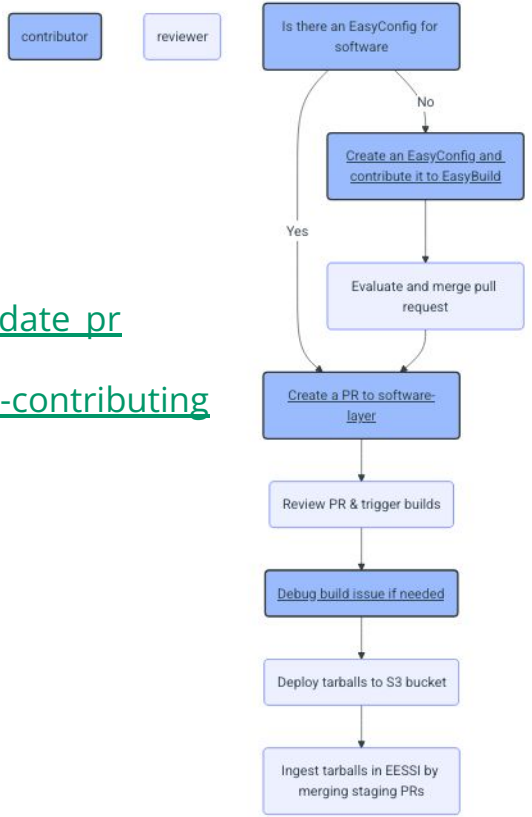
1. Find or create a (working) **easyconfig file** to install the software
2. **Test it in the EESSI build environment** (using `EESSI-extend`)
3. **Contribute it to EasyBuild** (if it hasn't been yet)
4. Open a **pull request** to the `EESSI/software-layer` repository
5. See if installation works on **all supported CPU targets**
    - Somebody from the EESSI "builders" team will trigger bot to do test builds
6. When that all checks out, the software installations will be **deployed to EESSI**

Recommendations for contributing software to EESSI

- **Test locally first**, using `EESSI-extend`
  - Can be on multiple CPU targets or families, but not strictly required
- Always **use latest EasyBuild release** (see `module avail EasyBuild` in EESSI)
- Pull request to EESSI can be opened while easyconfig PR to EasyBuild is still open, but *deployment* to EESSI can only be done when EasyBuild PR has been *merged*
- Work with EESSI “builders” team to resolve any problems that may pop up
  - Via comments in pull requests
  - Via EESSI Slack (see `#software-layer` or `#support` channels), see link @ eessi.io

Detailed guidelines for contributing software to EESSI

- See https://eessi.io/docs/adding_software/overview
- Documentation for contributing to EasyBuild:
  - https://docs.easybuild.io/contributing/#contributing_easyconfigs
  - https://docs.easybuild.io/integration-with-github/#github_new_update_pr
  - https://tutorial.easybuild.io/2023-eb-eessi-uk-workshop/easybuild-contributing
- Step-by-step instructions for opening PR to EESSI: https://www.eessi.io/docs/adding_software/opening_pr



Contributing software to EESSI, hands-on

Hands-on demo (EESSI 2025.06): contributing scikit-bio with `foss/2025a`

If test installation of `scikit-bio-0.7.1.post1-foss-2025a.eb` is OK,

we can open PR to `EESSI/software-layer` repository:

1. Add entry to `easystacks/software.eessi.io/2025.06/eessi-2025.06-eb-`**`5.3.0-2025a`**`.yml`
2. Open pull request to `EESSI/software-layer` repository
3. Check if installation works on all CPU targets + get it deployed...

Contributing software to EESSI, hands-on

Hands-on demo (EESSI 2025.06): contributing latest scikit-bio (0.7.2) with `foss/2025b`

If test installation of `scikit-bio-0.7.2.post1-foss-2025b.eb` from easyconfig PR is OK,

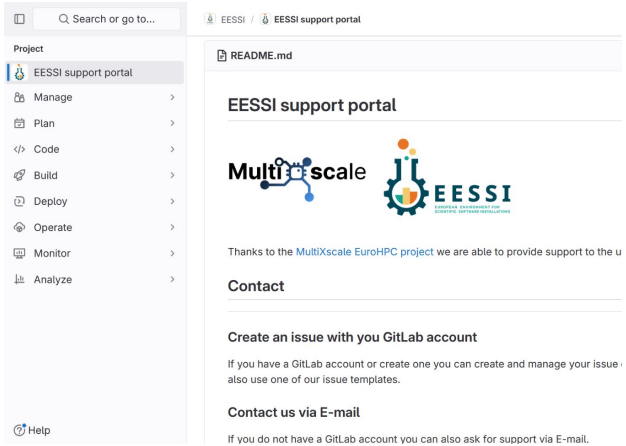
we can open PR to `EESSI/software-layer` repository:

1. Add entry to `easystacks/software.eessi.io/2025.06/eessi-2025.06-eb-5.3.0-`**`2025b`**`.yml`
2. **Use `from-commit` option** to let EasyBuild use easyconfig file(s) from easyconfigs pull request
3. Open pull request to `EESSI/software-layer` repository
4. Check if installation works on all CPU targets
5. Make sure that easyconfig PR gets merged, and then deployment in EESSI can be done

Support for installing, using, contributing to EESSI

- Via GitLab, or via email: `support@eessi.io`
- Report problems
- Ask questions
- Request additional software
- Get help with contributing to EESSI
- Suggest enhancements, additional features, …
- Confidential tickets possible (security issues, …)

eessi.io/docs/support



Dedicated support team, thanks to EuroHPC Centre-of-Excellence

Webinar series: Different aspects of EESSI

5 Mondays in a row April-May-June 2026

https://www.eessi.io/docs/training-events/2026/webinar-series-2026Q2

- 27 April: Introduction to EESSI - **recording available**
- 4 May: **Building software on top of EESSI + contributing to EESSI**
- 11 May: Using EESSI for Continuous Integration
- 18 May: Introduction to CernVM-FS
- 1 June: Using EESSI as a base for a central software stack

More info and registration →