



E E S S I

EUROPEAN ENVIRONMENT FOR
SCIENTIFIC SOFTWARE INSTALLATIONS

ISC'26 tutorial - 22 June 2026

<https://eessi.io>

<https://eessi.io/isc26-tutorial>

Agenda



[14:00-14:05] Practical information for this tutorial

[14:05-14:30] Introduction to EESSI

[14:30-15:00] **Using EESSI (hands-on)**

[15:00-15:15] Installing software on top of EESSI

[15:15-15:45] **Installing software on top of EESSI (hands-on)**

[15:45-16:00] Using EESSI in Continuous Integration (CI) environments

[16:00-16:30] *(coffee break)*

[16:30-17:15] **Using EESSI for CI in GitHub Actions CI (hands-on)**

[17:15-17:45] Advanced topics (incl. MPI, GPU, using Spack on top of EESSI)

[17:45-18:00] Q&A + closing remarks

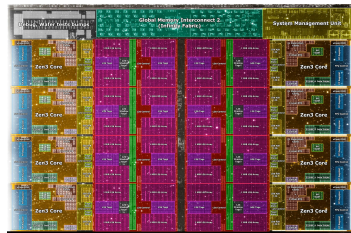
Practical information



- Tutorial website: <https://eessi.io/isc26-tutorial>
 - **Use this during the hands-on sessions!**
 - Includes code snippets or shell commands that you can copy-paste
- Prepared environment
 - Get access to a **temporary virtual machine to use during hands-on**
 - See instructions at <https://eessi.io/isc26-tutorial/prepared-environment>
 - Alternatively: use a system where EESSI is already available, see <https://eessi.io/docs/systems>

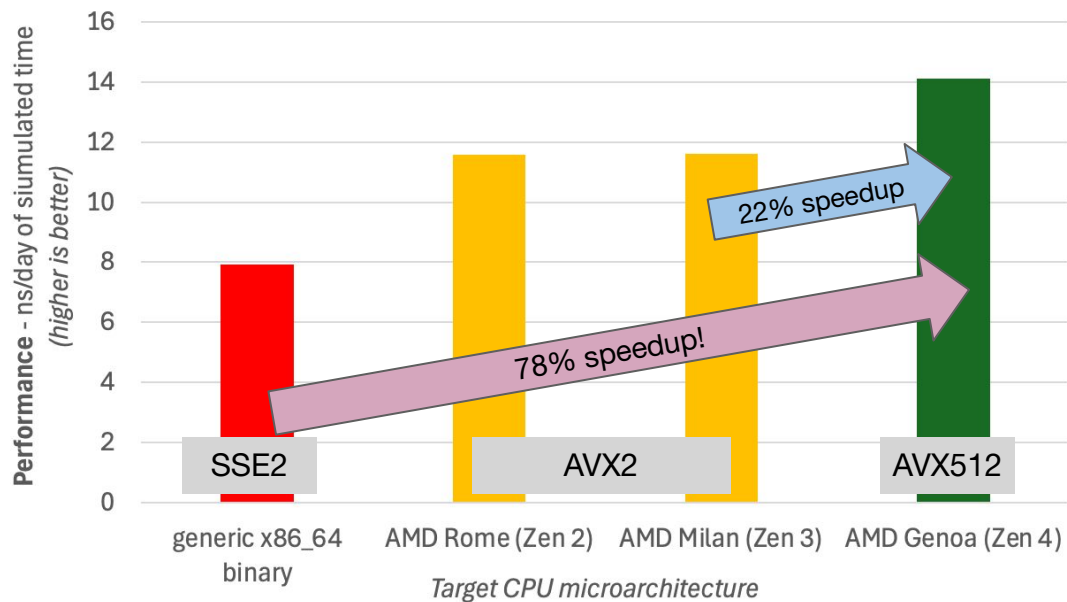
Software performance through vector instructions

- Modern microprocessors support **vector instructions**, a.k.a. SIMD (Single Instruction Multiple Data)
- Examples: AVX2, AVX-512, AVX10 (AMD, Intel), or SVE (Arm), ...
- Parts of the microprocessor are dedicated to running these vector instructions
- If you run binaries that are not using these instructions, you're not using a significant part of what the CPU supports!
- ... and **performance suffers** (potentially a lot)
- Downside: binaries that use vector instructions are less “portable”...
`Illegal instruction` errors will occur if an unsupported (vector) instruction is run



Keeping the P in HPC

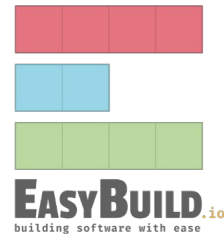
- Software should be optimized for (the CPU/GPU of) the system it will be run on
- This implies building software from source code, using specific compiler options, ...
- Impact on software performance is often significant for scientific software!



- Barplot shows performance of different GROMACS binaries, **same source code & workload, same hardware & OS**
- GROMACS version 2025.2, Test Case B from PRACE UEABS v2.2
- 1 node, dual-socket, 2x 96-core AMD Zen4
- Hybrid run on 192 cores in total, 48 MPI ranks with 4 OpenMP threads each

EasyBuild in a nutshell

<https://easybuild.io>

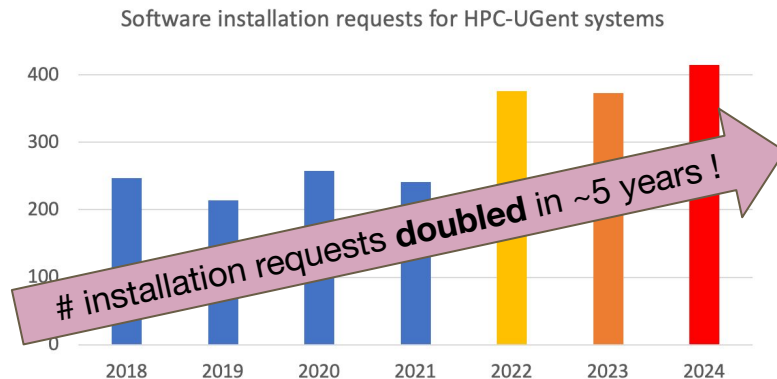


- Tool to install (scientific) software on HPC systems, usually from source code
- Created by HPC team at Ghent University (Belgium) in summer 2009
- Open source software (GPLv2) since Nov 2012 (during SC'12)
- Easy installation (`pip install easybuild`), extensive documentation (docs.easybuild.io)
- Used to manage central software stack on VSC systems + LUMI, JUPITER, Deucalion, ...
- Worldwide community grew around it, used all over Europe, Canada, US, Australia, ...



The landscape of scientific computing has changed

- **Explosion of available scientific software applications** (bioinformatics, AI boom, ...)
- Increasing interest in **cloud** for scientific computing (flexibility!)
- Increasing **variety in processor (micro)architectures** beyond Intel & AMD:
Arm is ~~coming~~ here (see Fugaku, JUPITER, ...), RISC-V is coming (soon?)
- In stark contrast: available (wo)manpower in HPC support teams is (still) limited
- **Software installation tools like EasyBuild and Spack are no longer sufficient...**



*What if you no longer have to install
a **broad range of scientific software**
from scratch on every laptop, HPC cluster,
or cloud instance you use or maintain,
without compromising on performance?*



European Environment for Scientific Software Installations

- **Shared set of ready-to-use (optimized!) scientific software installations**
- Goal: **avoid duplicate work** across by collaborating on a shared software stack
- Uniform way of providing software to users, regardless of the system they use
- **Should work on any Linux OS and system architecture**
 - From laptops and personal workstations to HPC clusters and cloud
 - Support for different CPUs, interconnects, GPUs, etc.
- Focus on **performance, automation, testing, collaboration**
- Development effort funded through **MultiXscale EuroHPC Centre-of-Excellence**



E E S S I
EUROPEAN ENVIRONMENT FOR
SCIENTIFIC SOFTWARE INSTALLATIONS

<https://eessi.io>

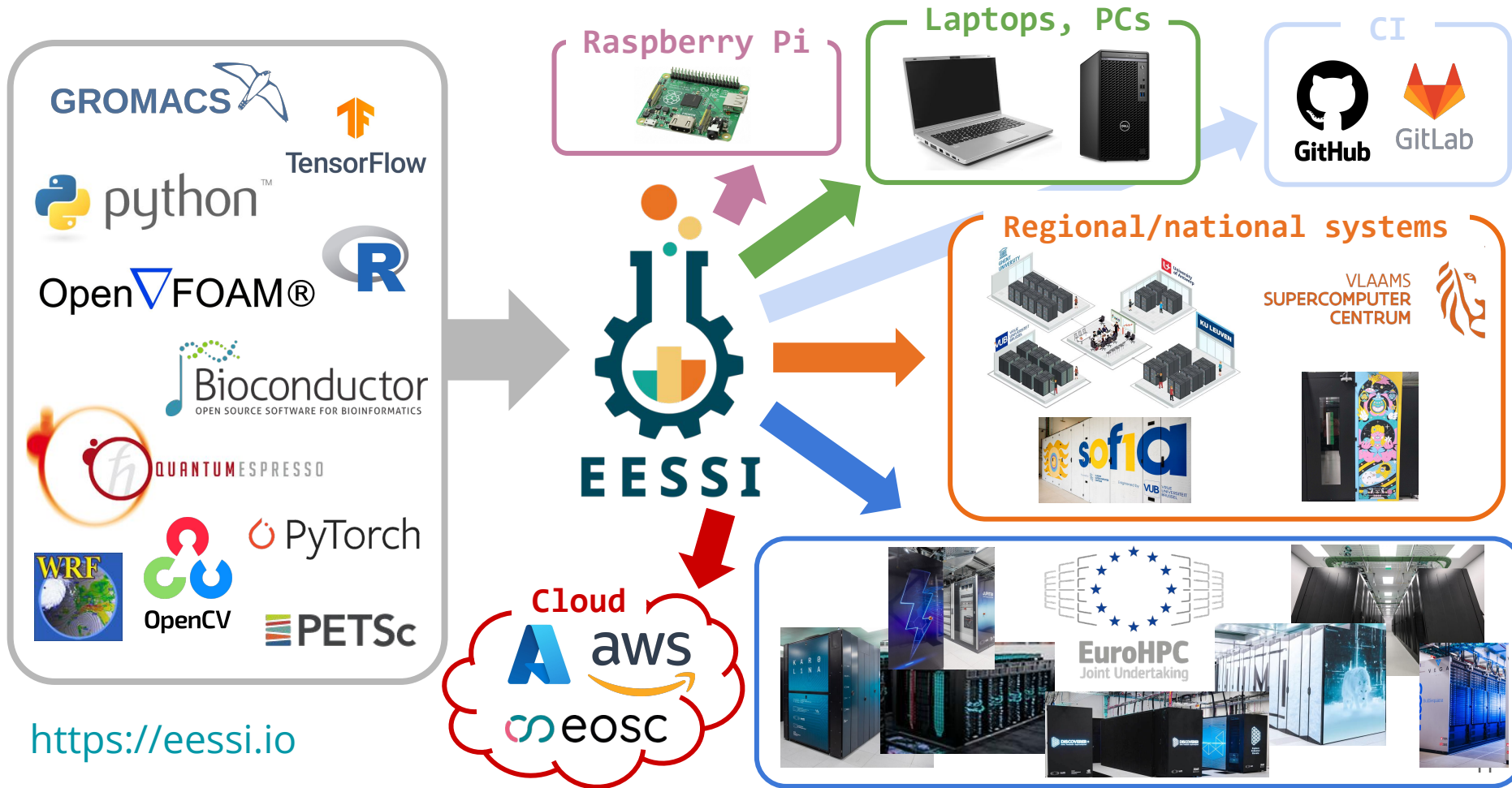
*inspired by the
ComputeCanada software stack*

Major goals of EESSI

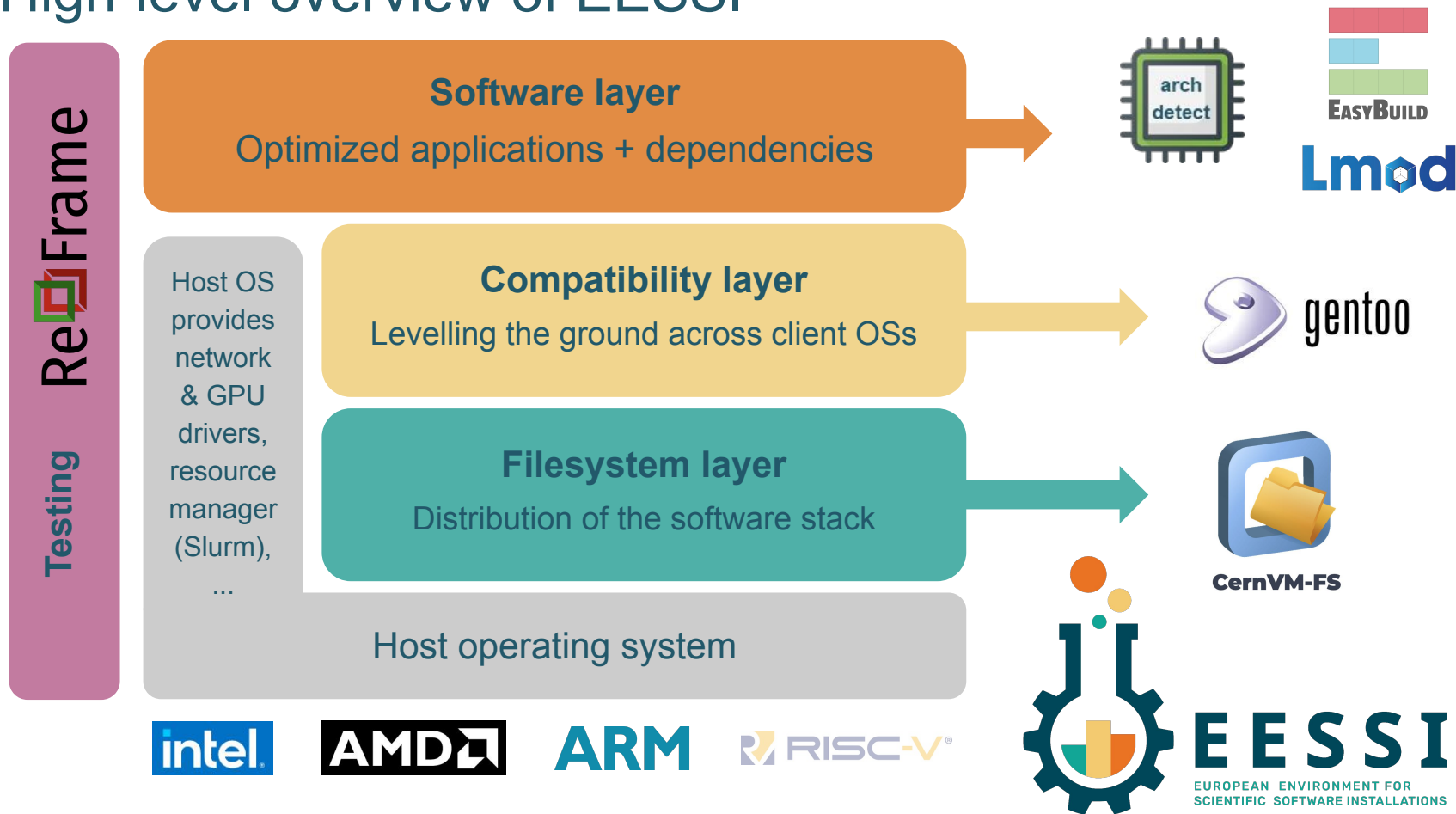


- Providing a truly **uniform software stack**
 - Have access to the (exact) same software environment everywhere
 - Across HPC, cloud, laptop, Continuous Integration environments, ...
 - **Without sacrificing performance** for “mobility of compute”
(as is typically done with containers/conda)
- **Avoid duplicate work** for researchers, HPC support teams, sysadmins, ...
 - Tools that automate software installation process
(EasyBuild, Spack) are not sufficient anymore
 - Go beyond sharing build recipes => work towards a shared software stack
- **Facilitate** HPC training, development of (scientific) software, ...

EESSI as shared software stack for HPC, and beyond



High-level overview of EESSI



EESSI is powered by open source software



gentoo linux™

Compatibility layer

Abstraction from the
host operating system



CernVM-FS

Filesystem Layer

Global distribution of
software installations
(on-demand streaming)

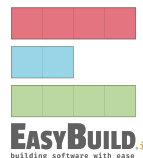
ReFrame

Regression
testing
of software



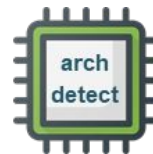
E E S S I

EUROPEAN ENVIRONMENT FOR
SCIENTIFIC SOFTWARE INSTALLATIONS



Optimized software
installations for specific
CPU microarchitectures

Intuitive user interface:
module avail,
module load, ...

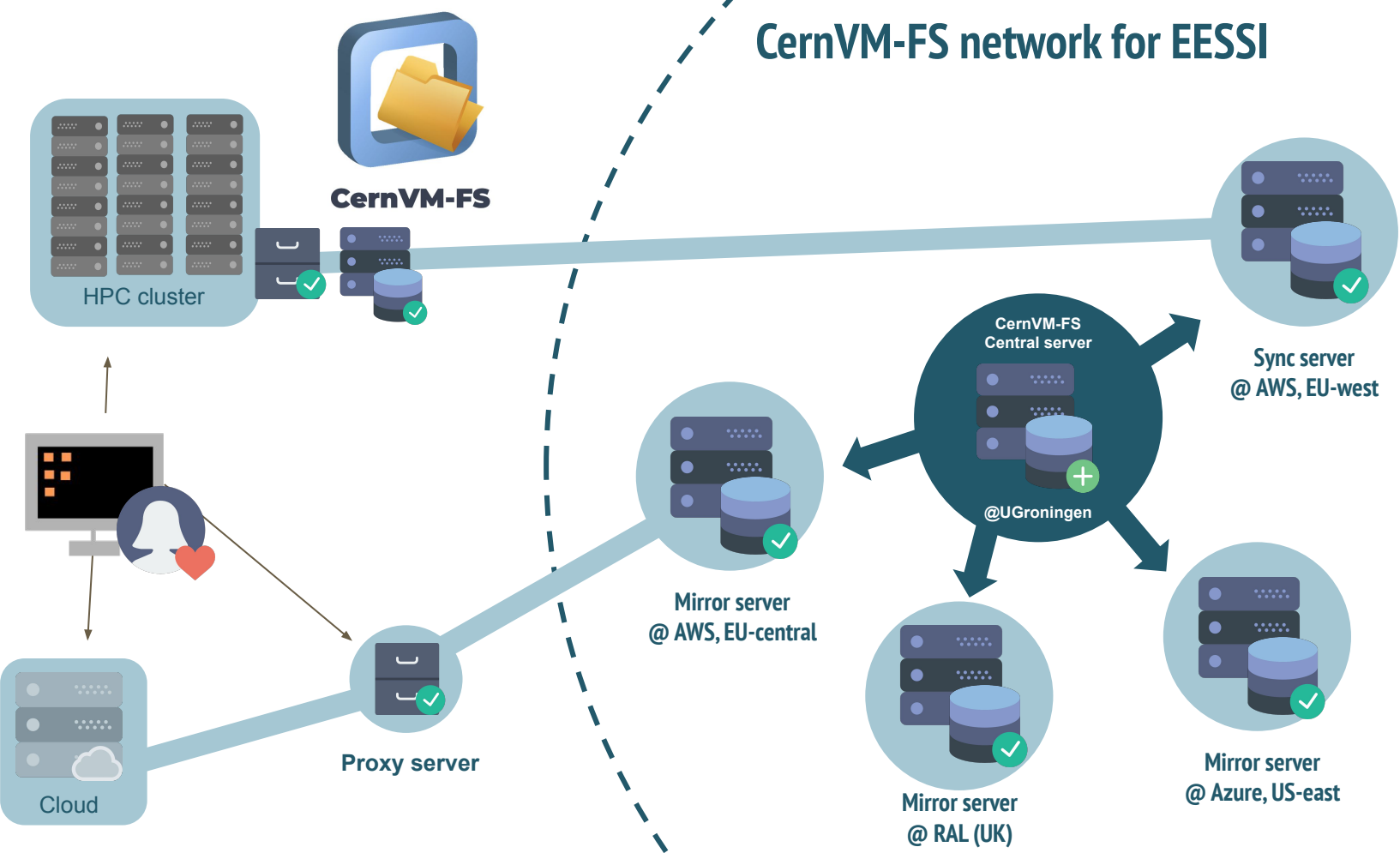


Automatic selection of
best suited part of
software stack for
CPU microarchitectures



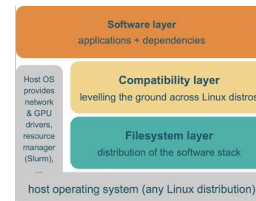
Magic Castle
to create (ephemeral)
clusters in the cloud

CernVM-FS network for EESSI



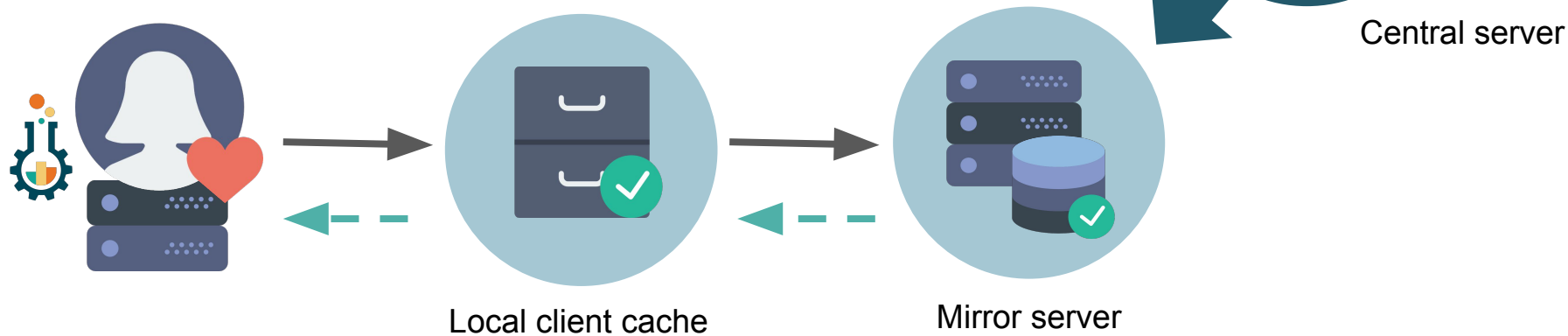
How does EESSI work?

- Software installations included in EESSI are:
 - Automatically “**streamed in**” on demand (via CernVM-FS)
 - Built to be **independent of the host operating system**
“Containers without the containing”
 - **Optimized** for specific CPU + GPU generations
 - **Ready-to-use**: no explicit installation or updating required
- Initialization script **auto-detects CPU + GPU** of the host system



The EESSI User Experience

```
source /cvmfs/software.eessi.io/versions/2025.06/init/lmod/bash  
module load GROMACS/2025.4-foss-2025b  
gmx ...
```



EESSI provides **on-demand streaming**
of (scientific) software (like music, TV-series, ...)

Overview of available software

An up-to-date overview of available software

is available in the EESSI documentation:

https://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

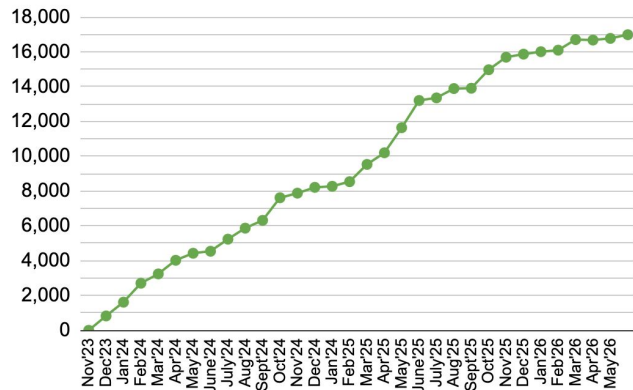


- EESSI provides **more than 2,000 software installations per supported CPU target** via `software.eessi.io` CernVM-FS repository, increasing every week
- **15 supported CPU targets** (x86_64 + Arm), support for RISC-V actively being explored
- **~750 different software packages** + another **~3,000 extensions** (Python packages, R libraries, ...)
- ~30,000 software installations in total, see https://eessi.io/docs/available_software
- Including GROMACS, LAMMPS, OpenFOAM, PyTorch, R, QuantumESPRESSO, 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

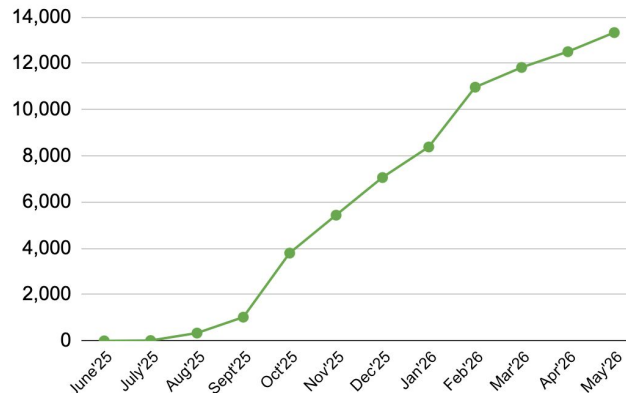
Set of available software is growing steadily...



software installations in EESSI 2023.06 (total)

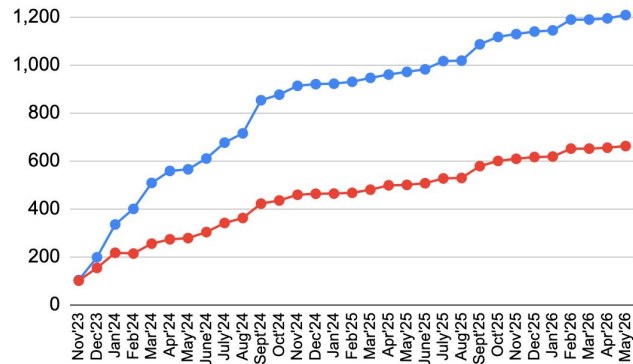


software installations in EESSI 2025.06 (total)



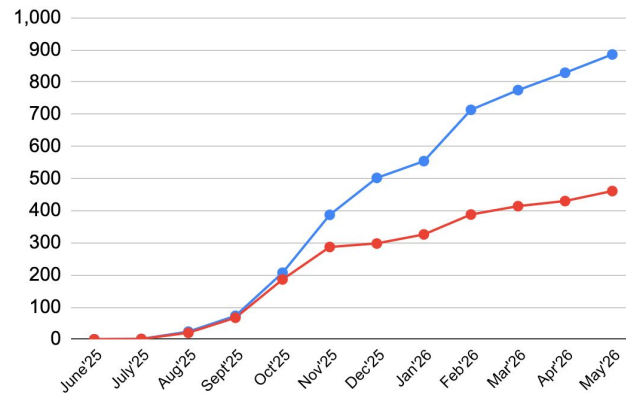
software installations in EESSI 2023.06 (per CPU target)

● # modules ● # unique software



software installations in EESSI 2025.06 (per CPU target)

● # modules ● # unique software



Supported CPU targets

eessi.io/docs/software_layer/cpu_targets



- AMD, Intel CPUs (`x86_64` CPU family)
 - 5 generations of Intel CPUs: Haswell, Skylake, Cascade Lake, Ice Lake, Sapphire Rapids
 - 4 generations of AMD CPUs: Zen2 (Rome), Zen 3 (Milan), Zen4 (Genoa), Zen5 (Turin)
- Arm CPUs (`aarch64` CPU family)
 - 4 different microarchitectures: Fujitsu A64FX, NVIDIA Grace, Neoverse N1 + V1
- Plus a generic fallback for both `x86_64` and `aarch64`
- Host CPU is automatically detected, automatic fallback to best suited CPU target
- Support for RISC-V CPU family is a work-in-progress, see <https://eessi.io/docs/repositories/dev.eessi.io-riscv>

Supported GPU targets

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://eessi.io/docs/site_specific_config/gpu
- AMD (ROCm)
 - Work-in-progress, initial installations for ROCm 6.4.1 have been deployed in EESSI 2025.06
 - Support for 8 different “generations” of AMD GPUs: `gfx90a` (MI250X, LUMI), ...
- See also NVIDIA + AMD GPU support in EESSI talk @ EasyBuild User Meeting ([slides](#), [recording](#))

On which systems is EESSI available today?



- On various systems throughout Europe (and beyond):
 - **EuroHPC JU supercomputers** incl. Vega, Karolina, MareNostrum 5, Deucalion, Discoverer, MeluXina, LUMI, Leonardo, JUPITER (soon), ...
See also <https://docs.my-eurohpc.eu/software-catalog/system-specific/about>
 - **National HPC systems:** VSC Tier-2 & Tier-1 (Belgium), Snellius @ SURF (NL), EMBL + Univ. of Stuttgart (Germany), Sigma2 & Olivia in Norway, etc.
 - Overview of (known) systems that have EESSI available at <https://eessi.io/docs/systems>
- EESSI can be used in virtual machine in European Open Science Cloud (EOSC), see also <https://eessi.io/docs/blog/2025/10/22/eosc>

Getting access to EESSI: via CernVM-FS



```
# Native installation via CernVM-Fs | see also https://eessi.io/docs/getting_access/native_installation
# Installation commands for RHEL-based distros like CentOS, Rocky Linux, Almalinux, Fedora, ...

# Install CernVM-FS
sudo yum install -y
https://ecsft.cern.ch/dist/cvmfs/cvmfs-release/cvmfs-release-latest.noarch.rpm
sudo yum install -y cvmfs

# Create client configuration file for CernVM-FS
# (no proxy, 10GB local CernVM-FS client cache)
sudo bash -c "echo 'CVMFS_CLIENT_PROFILE='single'' > /etc/cvmfs/default.local"
sudo bash -c "echo 'CVMFS_QUOTA_LIMIT=10000' >> /etc/cvmfs/default.local"

# Make sure that EESSI CernVM-FS repository is accessible
sudo cvmfs_config setup
```

see also <https://eessi.io/isc26-tutorial/getting-access>

Getting access to EESSI: alternative mechanisms



- Via Singularity/Apptainer, using a container image that includes CernVM-FS
 - Using wrapper script: https://eessi.io/docs/getting_access/eessi_container
 - See also <https://eessi.io/isc26-tutorial/eessi-getting-access/#eessi-via-a-container>
- Via `cvmfsexec` (<https://github.com/cvmfs/cvmfsexec>)
 - Without admin privileges, requires support in Linux kernel of host system (user namespaces, ...)
 - See <https://eessi.io/isc26-tutorial/eessi-getting-access/#eessi-via-cvmfsexec>
- By exporting the contents of EESSI to a SquashFS image, and mounting it in the right location

Live demo: Basic usage of EESSI



```
/cvmfs/software.eessi.io/versions/2023.06/software
```

```
-- linux
  |-- aarch64
    |-- generic
    |-- neoverse_n1
    |-- neoverse_v1
    |-- a64fx
    |-- nvidia
    |-- grace
  |-- x86_64
    |-- amd
      |-- zen2
        |-- modules
        |-- software
      |-- zen3
      |-- zen4
      |-- zen5
    |-- generic
  |-- intel
    |-- haswell
    |-- cascadelake
    |-- icelake
    |-- sapphirerapids
    |-- skylake_avx512
```

```
software.eessi.io/versions/2025.06/init/lmod/bash
```

before initialising EESSI

2025.06 loaded successfully

detected x86_64/amd/zen2 as the compatible CPU target

2025.06 Automatically detects CPU microarchitecture

to identify an accelerator on the system

For more information when loading the EESSI module, set the

environment variable EESSI_MODULE_DEBUG_INIT)

```
$ module load R/4.5.1-gfbbf-2025a
```

```
$ which R
```

```
/cvmfs/software.eessi.io/versions/2025.06/software/linux/x86_64/
```

```
bin/R/4.5.1-gfbbf-2023a/bin/R
```

```
$ R --version
```

Script for running GROMACS with EESSI

```
#!/bin/basheessi-demo.sh  
  
source /cvmfs/software.eessi.io/versions/2025.06/init/lmod/bash  
module load GROMACS/2025.4-foss-2025b  
  
# download input file + unpack (only if it's missing)  
if [ ! -f ion_channel.tpr ]; then  
    curl -OL https://repository.prace-ri.eu/ueabs/GROMACS/1.2/GROMACS_TestCaseA.tar.gz  
    tar xfz GROMACS_TestCaseA.tar.gz  
fi  
  
# cleanup, to force new run  
rm -f ener.edr logfile.log  
  
# run GROMACS (downscaled to 1k steps instead of full run with 10k steps)  
gmxd mdrun -s ion_channel.tpr -maxh 0.50 -nsteps 1000 -g logfile
```

Running GROMACS with EESSI

```
macbook-pro $ lima shell eessi
```

```
lima $ ls /cvmfs/software.eessi.io  
host_injections  init  
README.eessi  versions
```

```
lima $ ./eessi-demo.sh
```

```
...
```

	(ns/day)	(hour/ns)
Performance:	2.215	10.834



```
$ ssh slurm-cluster-aws
```



```
aws $ P=aarch64-neoverse-v1-node
```

```
aws $ sbatch -p $P -c 8 eessi-demo.sh  
113508
```

```
aws $ cat slurm-113508.out
```

```
...
```

	(ns/day)	(hour/ns)
Performance:	7.758	3.094

```
$ ssh tier1.hpc.ugent.be
```



```
vsc $ qsub -l nodes=1:ppn=16 eessi-demo.sh  
13001403
```

```
vsc $ cat eessi-demo.sh.*13001403
```

```
...
```

	(ns/day)	(hour/ns)
Performance:	20.802	1.154

```
$ ssh login.deucalion.macc.fccn.pt
```

```
deucalion $ sbatch -p normal-arm eessi-demo.sh  
Submitted batch job 7654321
```

```
deucalion $ cat slurm-7654321.out
```

```
...
```

	(ns/day)	(hour/ns)
Performance:	8.960	2.678



Hands-on: Using EESSI



- Make yourself familiar with using EESSI:
 1. Get access to EESSI, or use a system where it readily is available, see <https://eessi.io/isc26-tutorial/getting-access>
 2. Set up your shell environment by initializing EESSI, see <https://eessi.io/isc26-tutorial/usage>
 3. Run one (or more) of the EESSI demos, see <https://github.com/EESSI/eessi-demo>
- **We will continue with the next topic (Installing software on top of EESSI) at 15:00**
- **Do not hesitate to ask for help! (*raise your hand*)**

Installing software on top of EESSI



- You can use the software available in EESSI to **resolve dependencies** of the software you need
- To “manually” install software on top of EESSI (by running gcc, make, etc.), first load the `buildenv` module
- To install software **with EasyBuild** on top of EESSI, first load the `EESSI-extend` module to correctly configure EasyBuild
- See also https://eessi.io/docs/using_eessi/building_on_eessi
- Using Spack to install software on top of EESSI is also possible (more on this later)

Hands-on: Installing software on top of EESSI



- Live demo of:
 - Manual software installation of top of EESSI, with the help of `buildenv`
 - Installing software on top of EESSI with EasyBuild, facilitated by `EESSI-extend`
- For step-by-step instructions to try this yourself, see <https://eessi.io/isc26-tutorial/installing-on-top>
- **We will continue with the next topic (Using EESSI in CI) at 15:45**
- **Do not hesitate to ask for help! (*raise your hand*)**

Using EESSI for Continuous Integration (CI)



- EESSI can be used in CI environments like:
 - GitHub: github.com/marketplace/actions/eessi
 - GitLab: gitlab.com/explore/catalog/eessi/gitlab-eessi
- EESSI can provide:
 - Different compilers to test your software with
 - Required dependencies for your software
 - Additional tools like ReFrame, performance analysis tools, ...
- Other than CernVM-FS to get access to EESSI, no software installations required!
 - Everything that is actually needed is pulled in on-demand by CernVM-FS
- Significantly facilitates also running CI tests in other contexts (laptop, HPC, ...)

Using the EESSI GitHub Action



We have an EESSI GitHub Action that provides EESSI+di renv:

See it in action in the `github-eessi-action` repository:

github.com/EESSI/github-action-eessi

github.com/EESSI/github-action-eessi/blob/main/.github/workflows/tensorflow-usage.yml

```
name: ubuntu_tensorflow
on: [push, pull_request]
jobs:
```

```
  build:
```

```
    runs-on: ubuntu-latest
```

```
    steps:
```

```
      - uses: actions/checkout@v3
```

```
      - uses: eessi/github-action-eessi@v3
```

```
      with:
```

```
        eessi_stack_version: '2023.06'
```

```
      - name: Test EESSI
```

```
        shell: bash
```

```
        run: |
```

```
          module load TensorFlow
```

```
          python -c 'import tensorflow; print(tensorflow.__version__)'
```



Using the EESSI GitHub Action



build

succeeded 2 minutes ago in 1m 1s

Search logs



- > ☒ Set up job 2s
- > ☒ Run actions/checkout@v2 0s
- > ☒ Run eessi/github-action-eessi@main 52s
- ▼ ☒ Test EESSI 5s

```
1 ▼ Run module load GROMACS
2   module load GROMACS
3   gmx --version
4   shell: /usr/bin/bash --noprofile --norc -e -o pipefail {0}
5   env:
6     EESSI_SILENT: 1
7     BASH_ENV: /cvmfs/pilot.eessi-hpc.org/versions/2021.06/init/bash
8
9     :- ) GROMACS - gmx, 2020.4-MODIFIED (-:
10
11      GROMACS is written by:
12      Emile Apol      Rossen Apostolov      Paul Bauer      Herman J.C. Berendsen
13      Par Bjelkmar    Christian Blau    Viacheslav Bolnykh    Kevin Boyd
14      Aldert van Buuren    Rudi van Drunen    Anton Feenstra    Alan Gray
15      Gerrit Groenhof    Anca Hamuraru    Vincent Hindriksen    M. Eric Irrgang
16      Aleksei Iupinov    Christoph Junghans    Joe Jordan    Dimitrios Karkoulis
17      Peter Kasson      Jiri Kraus      Carsten Kutzner      Per Larsson
18      Justin A. Lemkul    Viveca Lindahl    Magnus Lundborg      Erik Marklund
19      Pascal Merz      Pieter Meulenhoff    Teemu Murtola      Szilard Pall
20      Sander Pronk      Roland Schulz      Michael Shirts      Alexey Shvetsov
21      Alfons Sijbers    Peter Tieleman      Jon Vincent      Teemu Virolainen
22      Christian Wennberg    Maarten Wolf      Artem Zhmurov
23      and the project leaders:
```

Coffee break

- Time to take a break...



- **We will resume at 16:30 with hands-on for using EESSI in CI**
- **Do not hesitate to ask us questions during the break!**

Hands-on: EESSI in CI



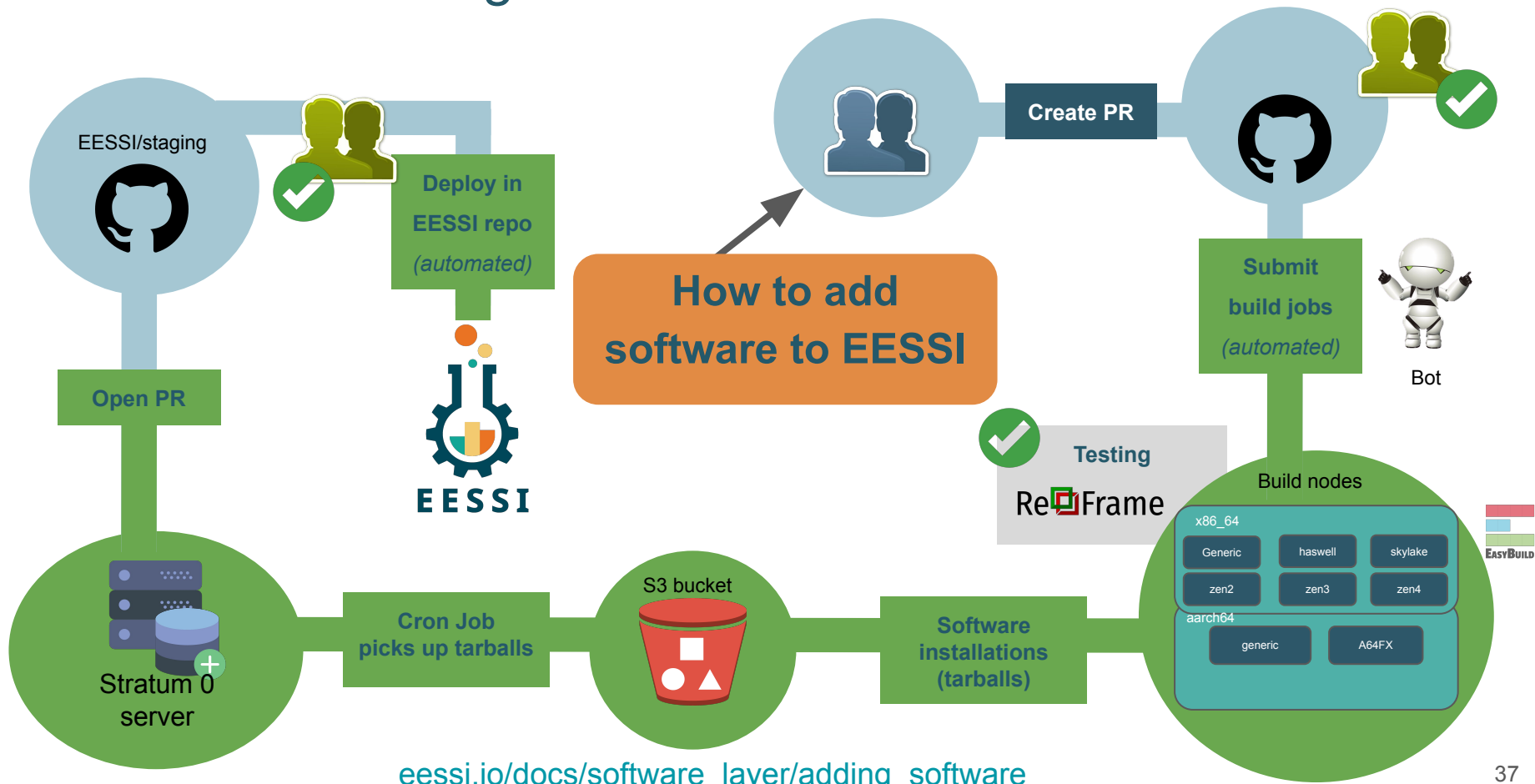
- Try creating a GitHub Actions workflow that uses EESSI...
- For step-by-step instructions, see <https://eessi.io/isc26-tutorial/ci>
- We will continue at 17:15 with the next topic (Advanced topics: MPI, GPU, Spack)
- Do not hesitate to ask for help! (*raise your hand*)

Advanced topics



- Adding software to EESSI
- Software testing in EESSI
- Running GPU-accelerated software through EESSI
- Building CUDA software on top of EESSI
- Using a tuned MPI library for improved performance
- Using Spack to install software on top of EESSI
- EESSI in the EuroHPC Federation Platform

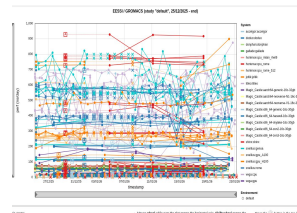
Workflow for adding software to EESSI



Software testing in EESSI



- Smoke tests: sanity check commands are run by EasyBuild to check that installed software is not horribly broken while using EasyBuild
- Regression testing via EESSI test suite: <https://eessi.io/docs/test-suite>
 - Collection of **portable tests** for software available in EESSI
 - Running on selected (single node) tests when building new software (before deployment)
 - Periodically (daily/weekly) on about multiple different systems
 - Can also be used for other software stacks (that are built with EasyBuild)
 - Periodic runs of EESSI test suite are done to help to catch performance regressions
- Public dashboard with test results: <https://dashboard.eessi.io>



Running GPU-accelerated software through EESSI



- To *run* software on NVIDIA GPUs, you need:
 - **CUDA runtime libraries** (shipped through EESSI)
 - **GPU drivers** (not included in EESSI)
- **Host GPU driver libraries must be exposed to EESSI** in order to *run* GPU software
- `link_nvidia_host_libraries.sh` script creates symbolic links in a known path:
`/cvmfs/software.eessi.io/host_injections/nvidia/<arch>/host/...`
- Must be run by an admin, write access to target of `host_injections` symlink is needed
- See https://eessi.io/docs/site_specific_config/gpu/#nvidia_drivers

Building CUDA software on top of EESSI



- To *build* CUDA software, you need a CUDA compiler
 - `nvcc` & co, part of full CUDA SDK
 - This **can not be provided by EESSI** (as specified in CUDA EULA)
- To build CUDA software on of EESSI, a **local installation of full CUDA SDK is needed**
 - `install_cuda_and_libraries.sh` script installs full CUDA SDK (with EasyBuild) under `/cvmfs/software.eessi.io/host_injections/nvidia/...`
 - This unbreaks symlinks in (partial) CUDA installations that are included in EESSI
 - Must be done by admin, write access to target of `host_injections` symlink is needed
- See https://eessi.io/docs/site_specific_config/gpu/#cuda_sdk

Using a tuned MPI library for improved performance



- EESSI (currently) only includes OpenMPI as MPI library
 - Also including MPICH-based MPI libraries are planned
- EESSI provides “fat” installations of OpenMPI: depends on UCX + libfabric
 - Supports different types of interconnect: Infiniband, Slingshot, Omni-Path, ...
 - Can be “tuned” through environment variables (OpenMPI, UCX, libfabric)
- **On some systems, using a custom configured MPI library may be desirable/required**
 - A custom MPI library can be injected into EESSI, but must be ABI-compatible!
- Broader adoption of common ABI (see MPI 5.0 standard) will make things easier
- See performance results for this approach on Cray Slingshot-11:
<https://eessi.io/docs/blog/2026/05/11/EESSI-on-Cray-Slingshot-part2>

Using Spack top of EESSI



Spack



- Spack can be made aware of software installations provided by EESSI, via a `packages.yaml` file that specifies them as “external packages”; see also https://spack.readthedocs.io/en/latest/packages_yaml.html#external-packages
- As a result, Spack can resolve dependencies via EESSI, rather than installing them (again)
- A proof-of-concept was implemented using QuantumESPRESSO, see <https://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 top of EESSI



Spack



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

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

```
git clone https://github.com/lorisercole/spack
cd spack

git checkout eessi

cd -

source spack/share/spack/setup-env.sh

spack --version
```

Using Spack top of EESSI



Spack



Hands-on demo - Part 2: Initialize EESSI (2023.06), get spood, run demo

```
git clone https://github.com/EESSI/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/EESSI/spood/issues/2>

Key part is packages.yaml, which tells Spack about software installed in EESSI:

https://github.com/EESSI/spood/blob/develop/examples/ext_install/packages.yaml

Using Spack on top of EESSI



```
⇒ 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-zxky252cemgoipfhhqoolnlow2s7qfrd
```

```
linux-vdso.so.1 (0x00007ffc0293e000)
libfftw3.so.3 ⇒ /cvmfs/software.eessi.io/versions/2023.06/software/linux/x86_64/intel/haswell/software/FFTW/3.3.10-60
libfftw3_omp.so.3 ⇒ /cvmfs/software.eessi.io/versions/2023.06/software/linux/x86_64/intel/haswell/software/FFTW/3.3.10-60
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-vvcl5hwifb2atnvu
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/0.3.21
libgcc_s.so.1 ⇒ /tmp/vsc40023/EESSI-webinar/spood/demo_spood/opt/linux-cascadelake/gcc-runtime-13.2.0-vvcl5hwifb2atnvu
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 =====
```

Using Spack on top of EESSI (in CI)



Using Spack on top of EESSI can also be done in GitHub Actions (for example)

```
steps:
- uses: actions/checkout@v6
- uses: eessi/github-action-eessi@v3
  with:
    eessi_stack_version: '2023.06'
- name: Build and test our code
  run: |
    # from a Python virtual environment
    git clone -b releases/v1.1 https://github.com/spack/spack.git

    # apply patch
    cd spack
    curl -L "https://github.com/spack/spack/compare/develop...lorisercole:spack:eessi.diff" | git apply -3

    # activate spack
    source ./share/spack/setup-env.sh
    # now go back
    cd ..
    ./quick_start.sh demo_dir
  shell: bash
```

See for example https://github.com/EESSI/spood/blob/develop/.github/workflows/test_build.yml

Using Spack on top of EESSI: next steps



- Make sure that Spack doesn't need to be patched anymore, cfr. <https://github.com/spack/spack/pull/51873> and <https://github.com/spack/spack/compare/develop...lorisercole:spack:eessi>
- More testing: other `spack install` commands, functional & performance tests, ...
- Auto-generate the `packages.yaml`, and ship it in EESSI repository; see also <https://github.com/EESSI/spood/pull/6>
- Come up with a maintainable way of mapping software names from EESSI to Spack
- Figure out how to deal with variants concept of Spack packages
- EESSI-`extend` module for Spack?

EESSI in the EuroHPC Federation Platform



- Ghent University was invited into the consortium led by CSC.fi that is currently developing the **EuroHPC Federation Platform (EFP)**
- EFP will be the “one-stop shop” for people to access & use EuroHPC systems
- For entire EuroHPC ecosystem: supercomputers, AI factories, quantum computers
- **EESSI is being integrated into EFP as base for the Federated Software Stack**
- First version of EFP became operational in April 2026
- All new EuroHPC projects are onboarded through EFP
- **More info via <https://my-eurohpc.eu> , see also BoF session at ISC’26 (Wed 10:45-11:45)**
- Recordings of first EFP webinar series available via <https://my-eurohpc.eu/training>

EESSI won an HPCWire Reader's Choice award!



eessi.io/docs/blog/2024/11/18/hpcwire-readers-choice-awards-2024-for-eessi

EESSI webinars (spring 2026)



- Topics:
 - Introduction to EESSI
 - Building software on top of + contributing to EESSI
 - Using EESSI for Continuous Integration
 - Introduction to CernVM-FS
 - Using EESSI as a base for a central software stack
- Slides + recordings available via <https://eessi.io/docs/training-events/2026/webinar-series-2026Q2>

EESSI Happy Hours sessions

Come join us



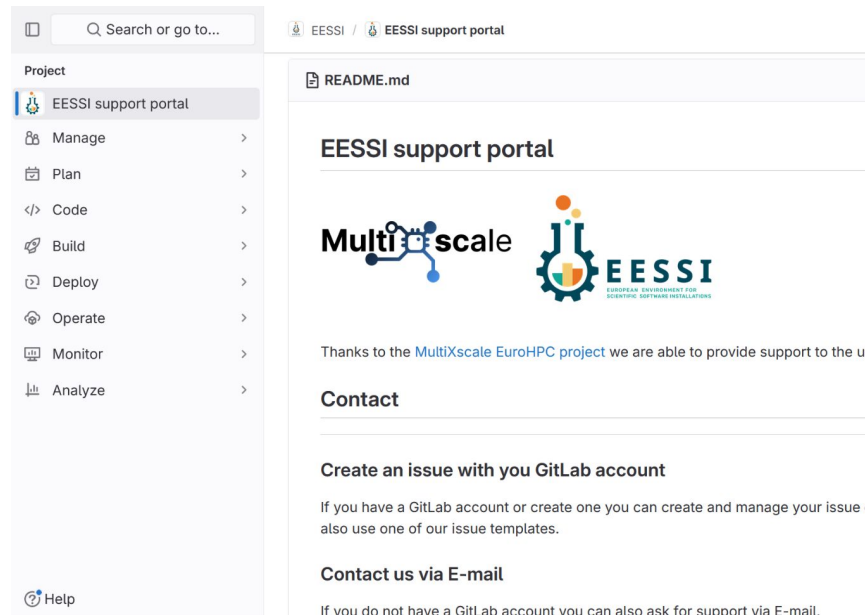
Every Monday at 14:00 CEST, we organise an EESSI Happy Hour

- Online, via Zoom
- Free to join for anyone interested in EESSI
- Features a short presentation or demo on a particular topic (10-15min)
- Open floor for discussions, questions, ...
- Recorded, but also more informal (off the record) towards the end
- See also <https://eessi.io/docs/training-events/happy-hours-sessions>

Getting support for EESSI

eessi.io/docs/support

- 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, ...)



The screenshot displays the EESSI support portal interface. On the left is a sidebar with a search bar and a 'Project' menu containing options like 'Manage', 'Plan', 'Code', 'Build', 'Deploy', 'Operate', 'Monitor', 'Analyze', and 'Help'. The main content area shows the 'README.md' for the 'EESSI support portal'. It features the 'MultiXscale' and 'EESSI' logos, a thank-you message to the MultiXscale EuroHPC project, and sections for 'Contact' and 'Create an issue with you GitLab account'. The 'Contact' section includes instructions on how to use GitLab for issue tracking and provides an email contact option for users without a GitLab account.

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



Upcoming events



- **EESSI Birds-of-a-Feather session at ISC'26**

- Thursday 24 June'26, 09:00-10:00 CEST
- Quick status update of EESSI
- Interactive poll, discussions, Q&A: join us!



EESSI BoF session @ ISC'24

- **EESSI hackathon after EuroHPC User Days 2026 in Dublin**

- Most likely on Fri 25 Sept 2026
- Topic: **Adding (your) software to EESSI**
- Details will be announced soon...



EESSI hackathon @ EuroHPC User days 2025

MultiXscale

Website: multixscale.eu

Facebook: [MultiXscale](https://www.facebook.com/MultiXscale)

Twitter: [@MultiXscale](https://twitter.com/MultiXscale)

LinkedIn: [MultiXscale](https://www.linkedin.com/company/multixscale)

BlueSky: [MultiXscale](https://bsky.app/profile/multixscale)



Co-funded by
the European Union



EuroHPC
Joint Undertaking



UNIVERSITAT DE
BARCELONA



Universität
Stuttgart



SORBONNE
UNIVERSITÉ



Université
de Toulouse



Consiglio Nazionale
delle Ricerche



MAX-PLANCK-GESELLSCHAFT





Website: eessi.io

GitHub: github.com/EESSI

Documentation: eessi.io/docs

Blog: eessi.io/docs/blog

Getting support: eessi.io/docs/support

[Join](#) the EESSI Slack (see link on EESSI website)

YouTube channel: youtube.com/@eessi_community

Paper (open access): doi.org/10.1002/spe.3075

EESSI support portal: gitlab.com/eessi/support

[Bi-monthly online meetings](#) (1st Thu, odd months, 2pm CE(S)T)

Free EESSI coffee mug for every tutorial attendee!



sponsored by:



Please rate the tutorial!

