Streaming Optimised Scientific Software, an Introduction to EESSI

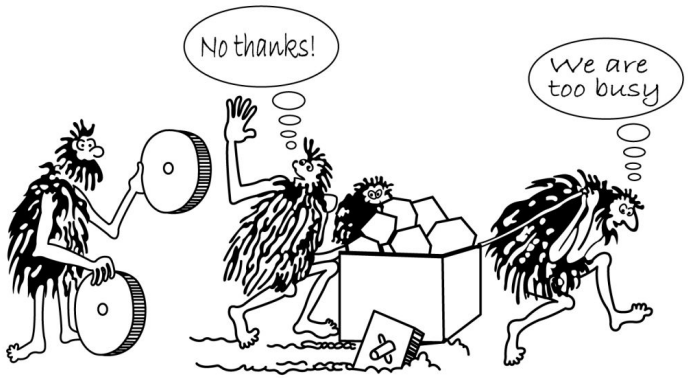
EUROPEAN ENVIRONMENT FOR SCIENTIFIC SOFTWARE INSTALLATIONS

MultiXscale

Thu 27 Mar 2025

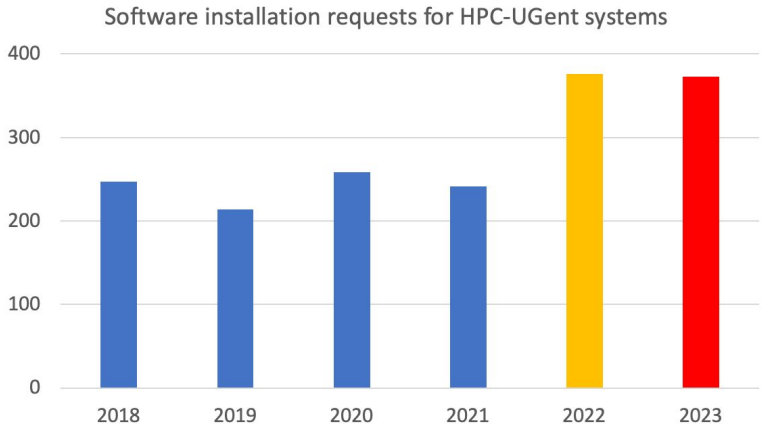
Lara Peeters - Ghent University (BE)

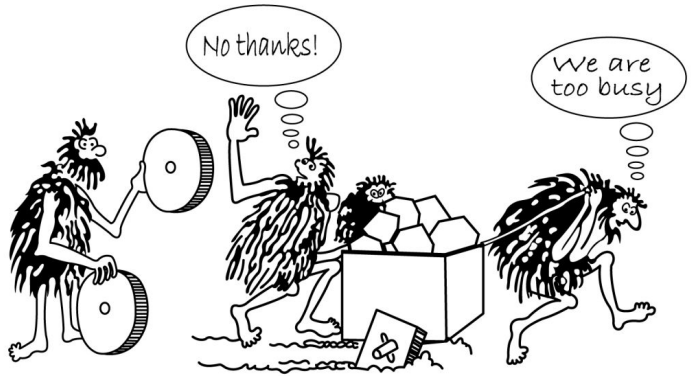
lara.peeters@ugent.be



The changing landscape of scientific computing

- **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~~ already here (see Fugaku, JUPITER, …), RISC-V is coming (soon?)
- In strong contrast: available (wo)manpower **in HPC support teams is (still) limited**…

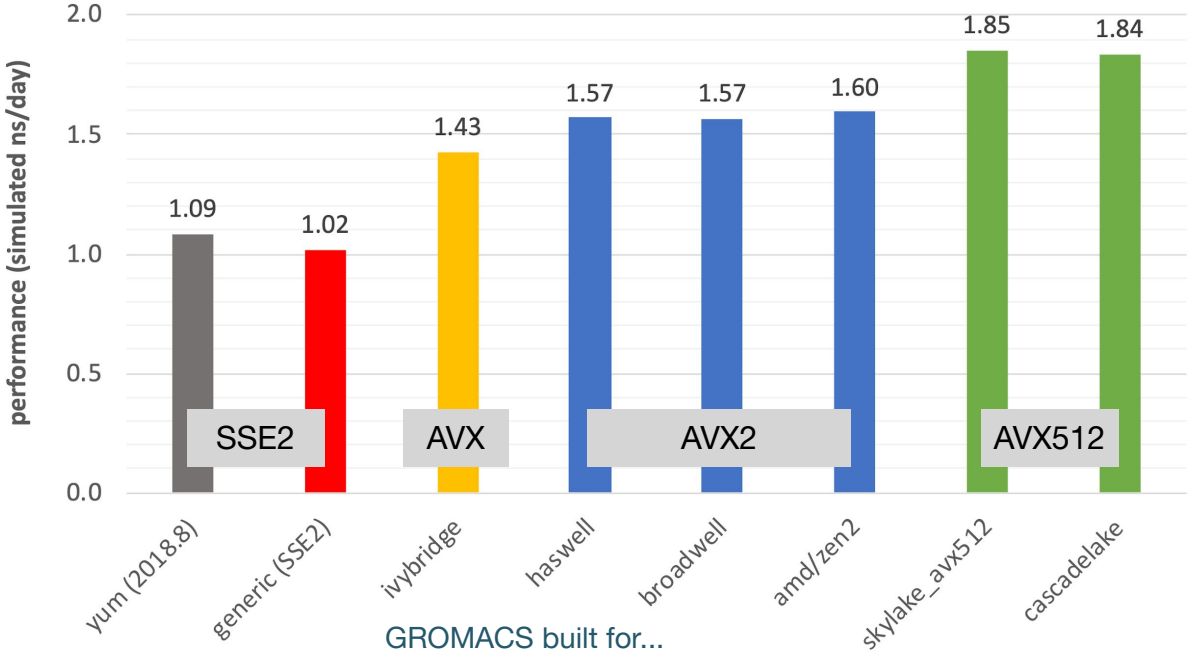




Optimized scientific software installations

- Software should be optimized for the system it will run on (keep the P in HPC!)
- Impact on performance is often significant for scientific software!

- Example: GROMACS 2020.1 (PRACE benchmark, Test Case B)
- Metric: (simulated) ns/day, higher is better
- Test system: dual-socket Intel Xeon Gold 6420 (Cascade Lake, 2x18 cores)
- **Performance of different GROMACS binaries, on exact same hardware/OS**



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?

Alan

EESSI in a nutshell

- European Environment for Scientific Software Installations (EESSI)
- **Shared repository of (optimized!) scientific software installations**
- Uniform way of providing software to users, regardless of the system they use!
- Should work on any Linux OS (+ WSL, macOS via Lima) and system architecture
- From laptops and personal workstations to HPC clusters and cloud
- Support for different CPU (micro)architectures, interconnects, GPUs, etc.
- **Focus on performance, automation, testing, collaboration**

https://eessi.io

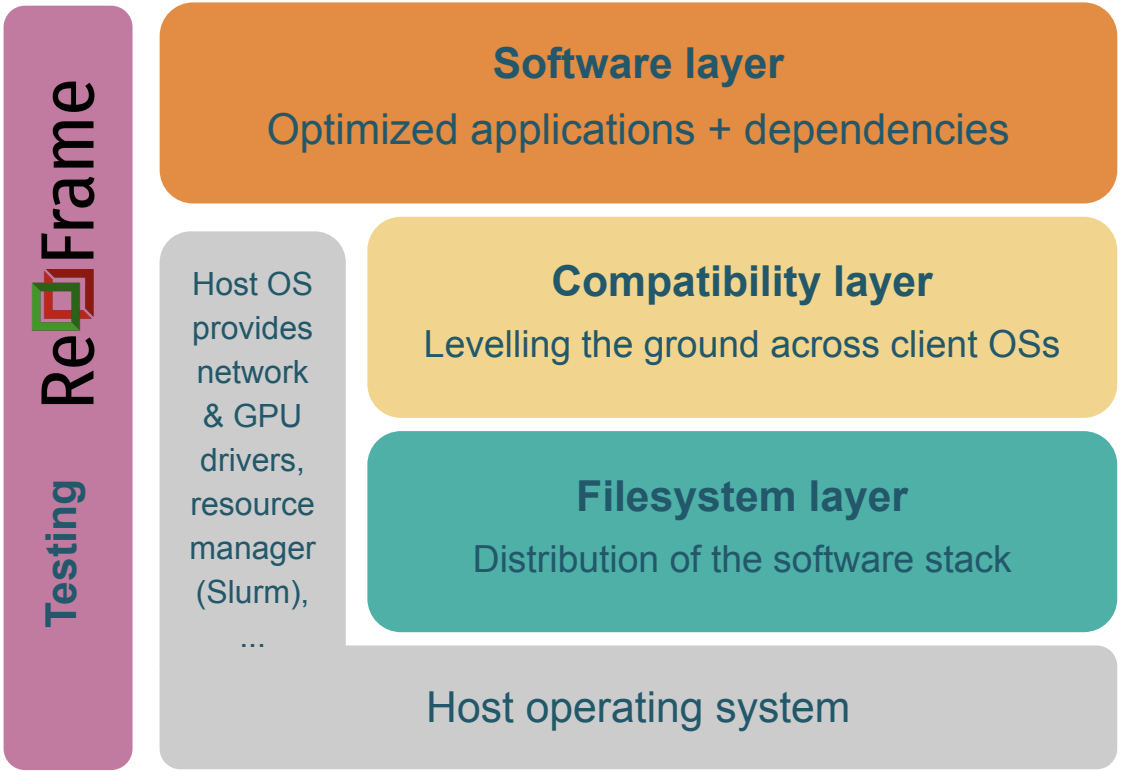
https://eessi.io/docs

Alan

Major goals of EESSI

- **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
- Providing a truly **uniform software stack**
  - Use the (exact) same software environment everywhere
  - **Without sacrificing performance** for “mobility of compute” (like is typically done with containers/conda)
- Facilitate HPC training, development of (scientific) software, …

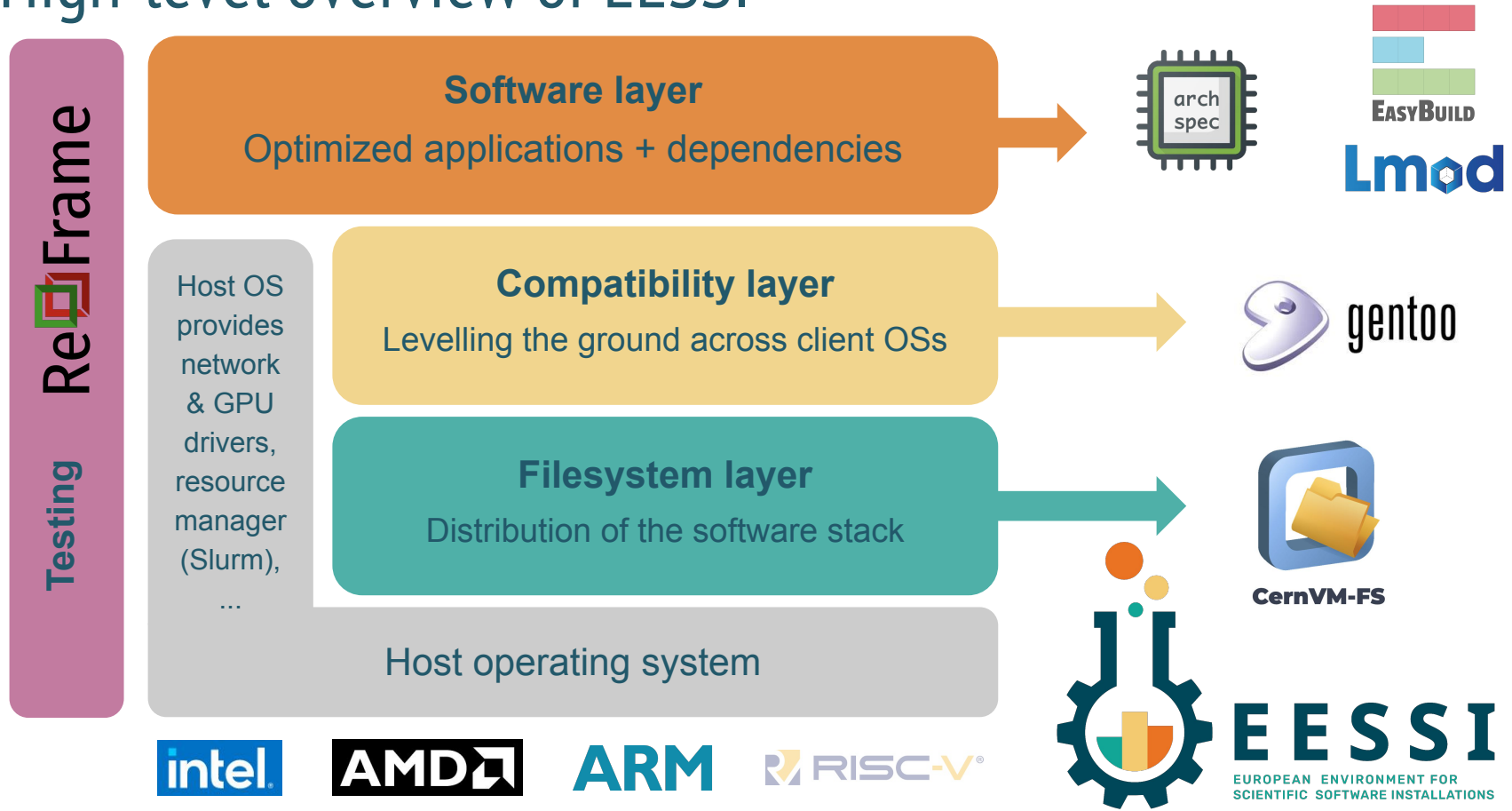
High-level overview of EESSI



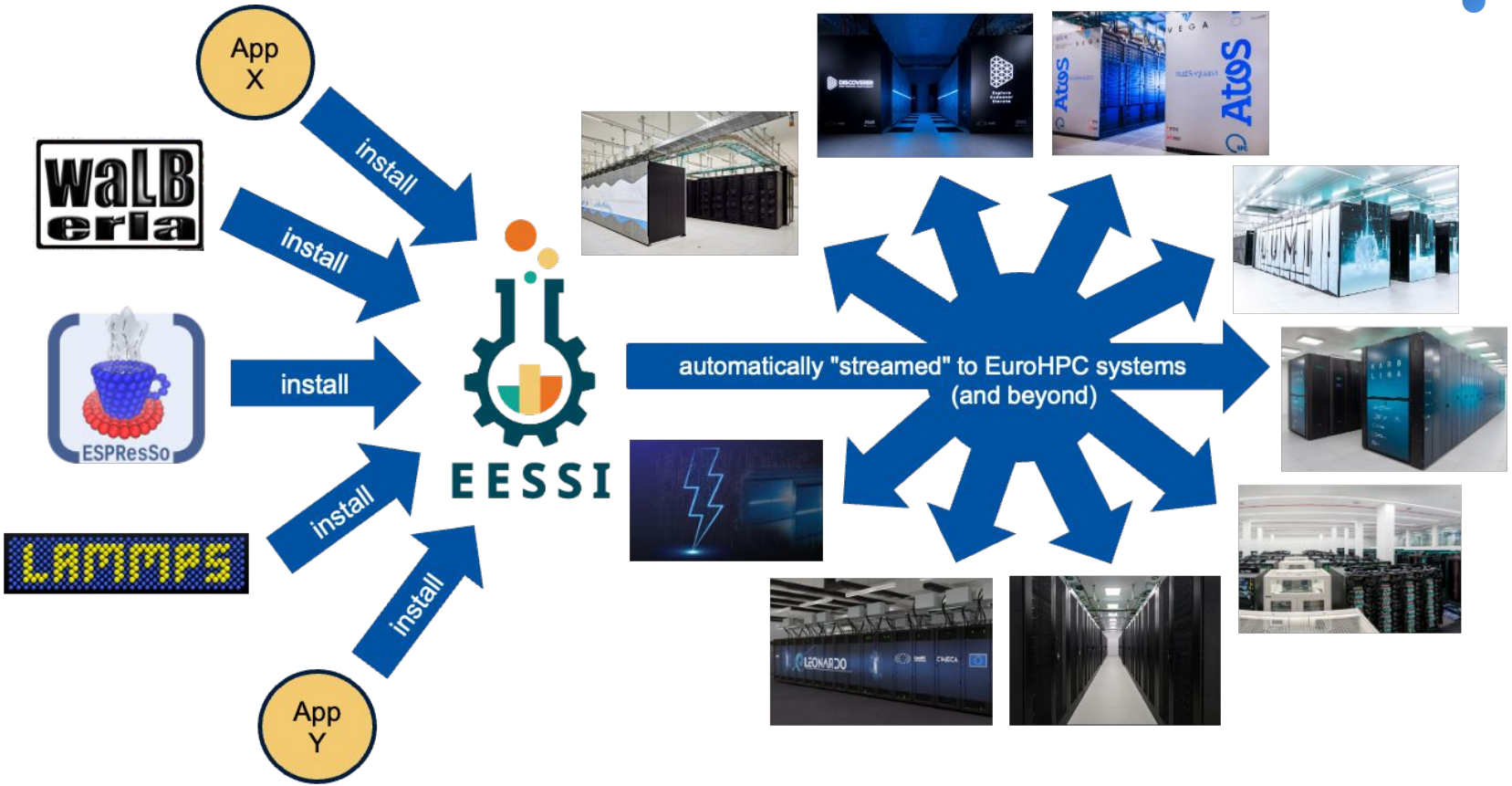
EESSI
EUROPEAN ENVIRONMENT FOR SCIENTIFIC SOFTWARE INSTALLATIONS

Alan

High-level overview of EESSI



EESSI as a shared software stack



Hands-on live demo

Getting access to EESSI

Using EESSI

Kenneth

Native installation of CernVM-FS

```
# 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 docs for alternative ways of installing CernVM-FS natively, via a VM on a personal computer
eessi.io/docs/getting_access/eessi_wsl/ - eessi.io/docs/getting_access/eessi_limactl/

Native installation of CernVM-FS on an HPC cluster

- For a single system, it's sufficient to install and configure CernVM-FS client
- For an HPC cluster, a bit more work is needed to:
  - Enhance the reliability of the access to EESSI
  - Improve startup performance of software
- It is recommended to:
  - Have a full copy of the EESSI repositories in your local network, by setting up a private CernVM-FS Stratum-1 "mirror" server
  - Have one or more proxy servers, to offload the Stratum-1 server(s)
- See also https://multixscale.github.io/cvmfs-tutorial-hpc-best-practices

Demo: Using EESSI

eessi.io/docs/using_eessi/eessi_demos

s

```
/cvmfs/software.eessi.io/versions/2023.06/software
`-- linux
    |-- aarch64
    |   |-- generic
    |   |-- neoverse_n1
    |   `-- neoverse_v1
    `-- x86_64
        |-- amd
        |   |-- zen2
        |   `-- zen3
        |-- generic
        `-- intel
            |-- haswell
            `-- skylake_avx512
                |-- modules
                `-- software
```

```
$ source /cvmfs/software.eessi.io/versions/2023.06/init/bash
Found EESSI pilot repo @
/cvmfs/software.eessi.io/versions/2023.06!
archdetect says x86_64/amd/zen3
Using x86_64/amd/zen3 as software subdirectory
...
Environment set up to use EESSI pilot software stack, have fun!

{EESSI 2023.06} $ module load R/4.3.2-gfbf-2023a

{EESSI 2023.06} $ which R
/cvmfs/software.eessi.io/versions/2023.06/software/linux/x86_64/
amd/zen3/software/R/4.3.2-gfbf-2023a/bin/R

{EESSI 2023.06} $ R --version
R version 4.3.2
```

Demo: Running LAMMPS in a Slurm job script

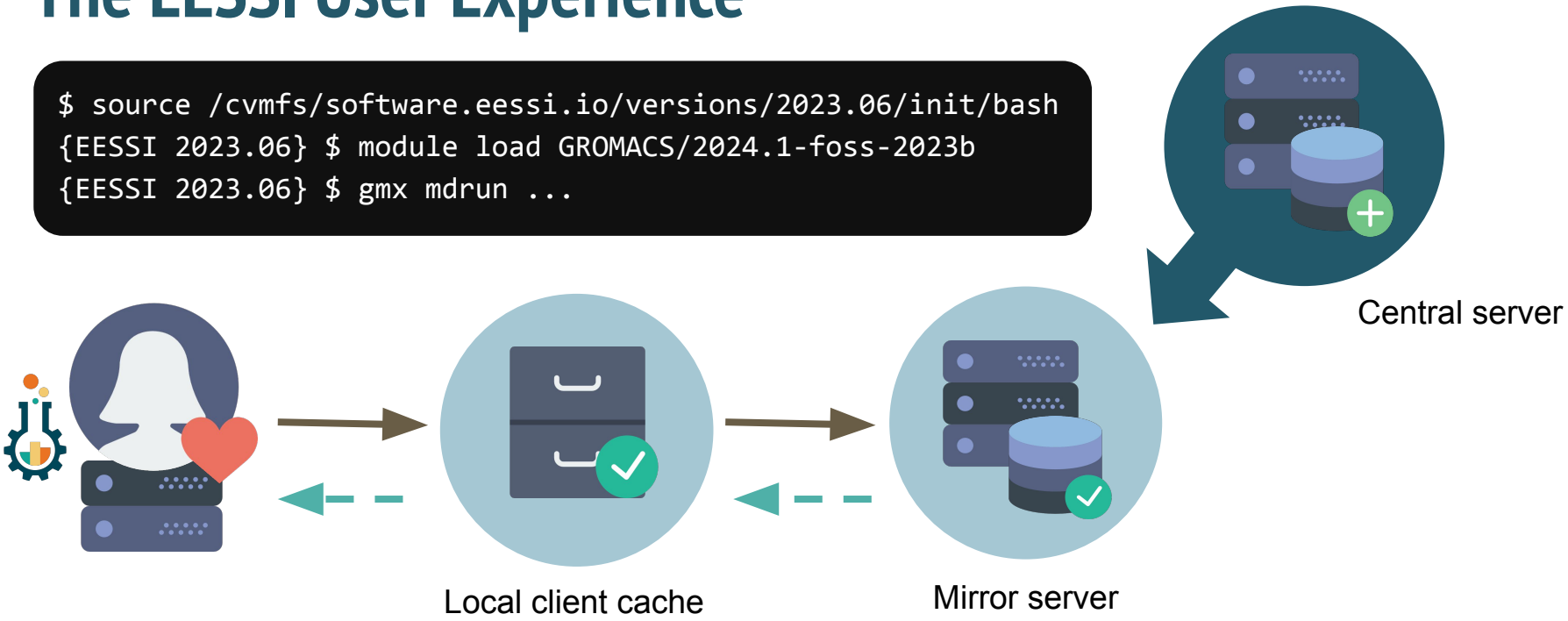

```
#!/bin/bash
# EESSI_demo.sh script
#SBATCH --job-name="EESSI_Demo_LAMMPS_lj"
#SBATCH --ntasks=4
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=1
#SBATCH --output=EESSI_demo.out
#SBATCH --error=EESSI_demo.err
#SBATCH --time=0:30:0
#SBATCH --partition=cpu_rome

source /cvmfs/software.eessi.io/versions/2023.06/init/bash
module load LAMMPS/29Aug2024-foss-2023b-kokkos
mkdir /tmp/$USER && cd /tmp/$USER
curl -o in.lj https://raw.githubusercontent.com/lammps/lammps/refs/heads/develop/bench/in.lj
export OMP_NUM_THREADS=1
mpirun -np 4 lmp -in in.lj
rm -r /tmp/$USER
```

EESSI

CernVM-FS

The EESSI User Experience

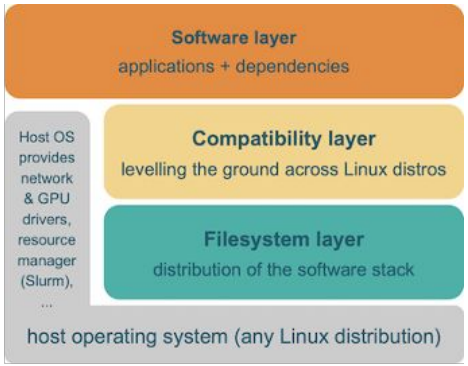
```
$ source /cvmfs/software.eessi.io/versions/2023.06/init/bash
{EESSI 2023.06} $ module load GROMACS/2024.1-foss-2023b
{EESSI 2023.06} $ gmx mdrun ...
```



EESSI provides **on-demand streaming**

of (scientific) software (like music, TV-series, …)

How does EESSI work?



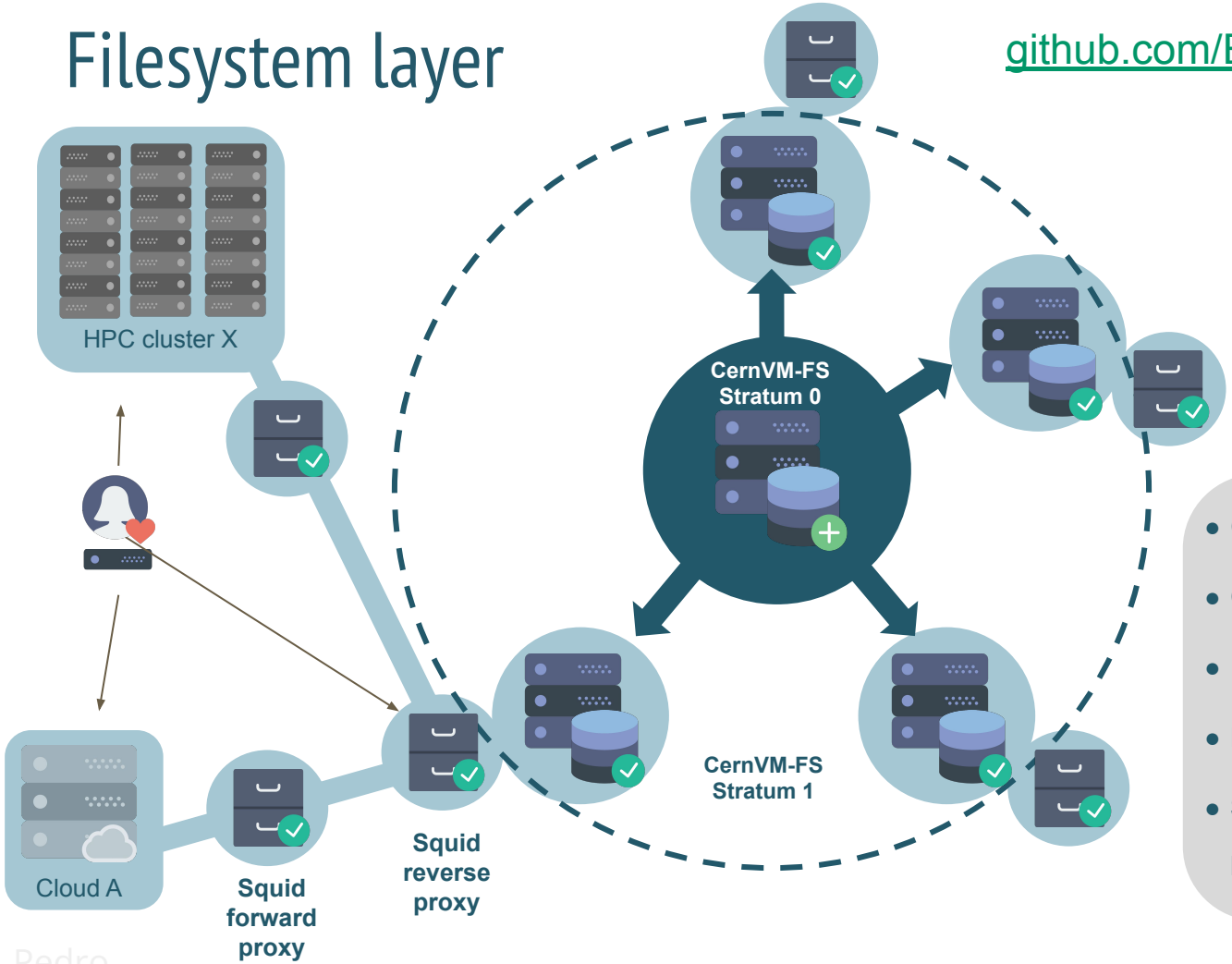
- Software installations included in EESSI are:
  - Automatically **“streamed in” on demand** (via CernVM-FS)
  - Built to be **independent of the host operating system**

    “Containers without the containing”
  - **Optimized** for specific CPU generations + specific GPU types
- Initialization script **auto-detects** CPU + GPU of the system

Pedro

Filesystem layer

github.com/EESSI/filesystem-layer



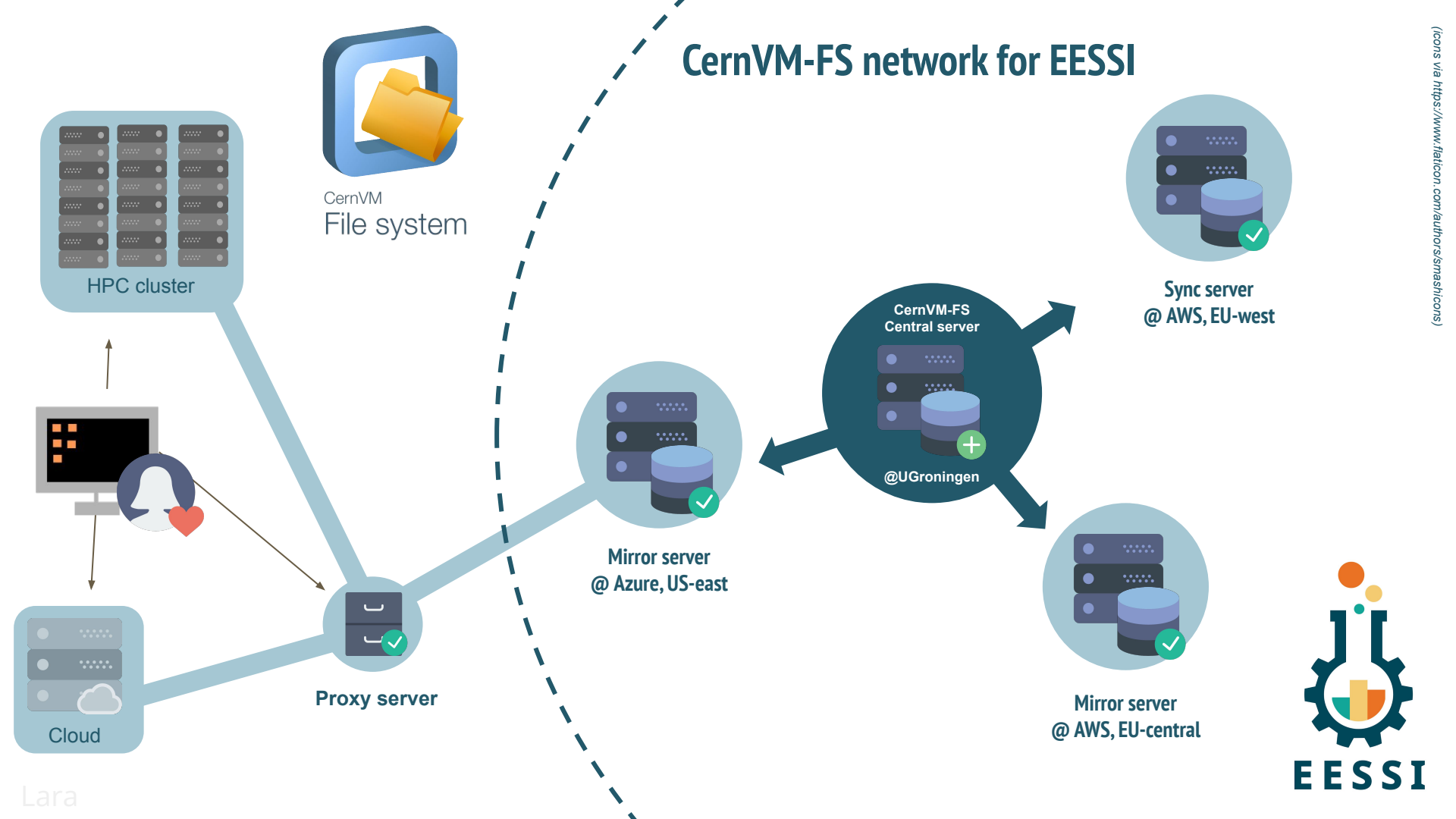
CernVM-FS

cvmfs.readthedocs.io

- Global distribution of software installations
- Centrally managed software stack
- Redundant network of “mirrors”
- Multiple levels of caching
- **Same software stack everywhere**:
  laptops, HPC clusters, cloud VMs, …

(icons via https://www.flaticon.com/authors/smashicons)

CernVM-FS network for EESSI



(icons via https://www.flaticon.com/authors/smashicons)

Lara

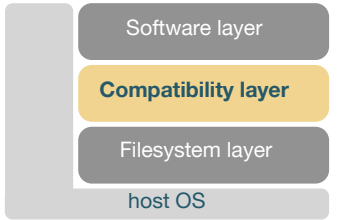
Compatibility layer

github.com/EESSI/compatibility-layer

EESSI

- Gentoo Prefix installation (in `/cvmfs/.../compat/<os>/<arch>/`)
- **Set of Linux tools & libraries installed in non-standard location**
- Limited to low-level stuff, incl. glibc (no Linux kernel or drivers)
- Similar to the OS layer in container images
- Only targets a supported **processor family** (`aarch64`, `x86_64`, `riscv64`)
- **Levels the ground for different client operating systems** (Linux distributions)
- Currently in production repository:
  `/cvmfs/software.eessi.io/versions/2023.06/compat/linux/aarch64`
  `/cvmfs/software.eessi.io/versions/2023.06/compat/linux/x86_64`

powered by gentoo



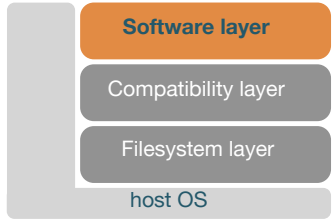
Pedro

Software layer

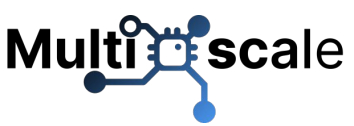
github.com/EESSI/software-layer

- Provides **installations of scientific software** applications & libraries (incl. deps)
- Optimized for specific CPU microarchitectures (AMD Zen3, ...)
  - Separate subdirectory/tree for each (in `/cvmfs/.../software/...`)
- Support for specific generation of **(NVIDIA) GPUs** via `/accel/` subdirectories
- **Leverages libraries** (like glibc) **from compatibility layer** (*not* from host OS)
- Installed with EasyBuild, incl. environment module files
- Lmod environment modules tool is used to access installations
- **Best subdirectory for host is selected automatically** via archdetect

powered by



Current status of EESSI



- Production CernVM-FS repository **`software.eessi.io`** available since Nov'23
- Ansible playbooks, scripts, docs available at https://github.com/eessi
- CernVM-FS: Stratum 0 @ Univ. of Groningen + two Stratum 1 servers in AWS + Azure
- Target CPU microarchitecturs (see also https://eessi.io/docs/software_layer/cpu_targets):

  `{aarch64,x86_64}/generic`

  `intel/{haswell, skylake_avx512}, amd/{zen2,zen3,zen4},`

  `aarch64/{neoverse_n1,neoverse_v1,a64fx}`

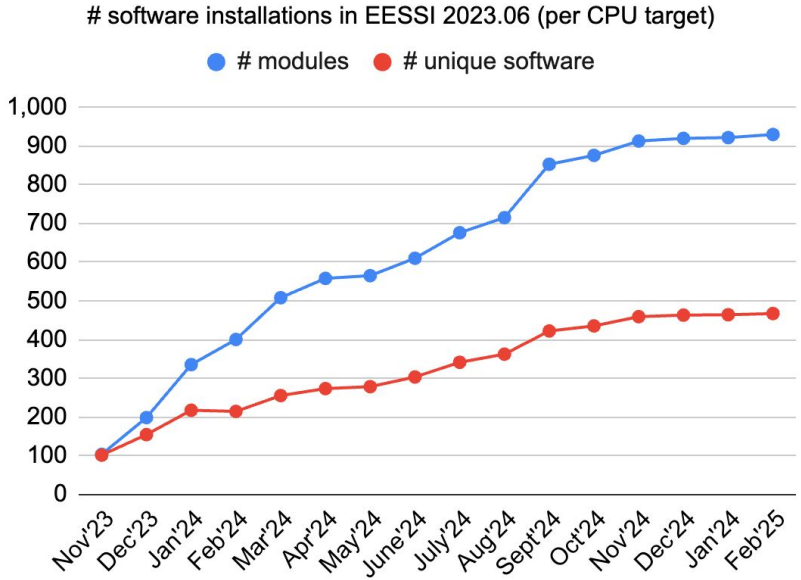
- **NVIDIA GPU support in place**, limited set of GPU software installed

- **Supported by Azure and AWS**: sponsored credits to develop necessary infrastructure

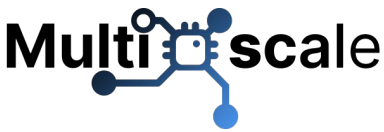
Overview of available software

Currently ~940 software installations available per CPU target via software.eessi.io CernVM-FS repository; increasing every day

- Over 467 different software packages
- Excl. extensions: Python packages, R libraries
- Including ESPResSo, GROMACS, LAMMPS, OpenFOAM, PyTorch, R, QuantumESPRESSO, TensorFlow, waLBerla, WRF, …
- eessi.io/docs/available_software/overview
- Using recent compiler toolchains: currently focusing on `foss/2023a` and `foss/2023b`

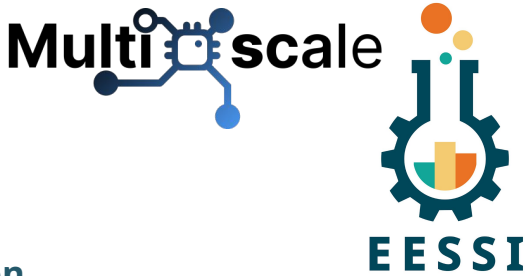


Supported system architectures



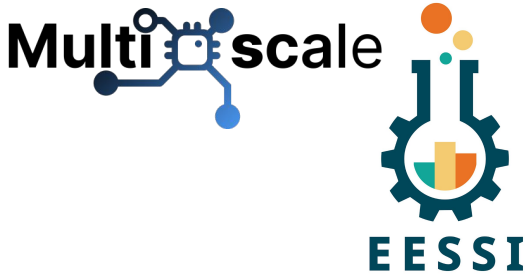
- Different generations of `x86_64` (Intel, AMD) and Arm 64-bit CPUs; RISC-V is WIP
  - Including A64FX (Deucalion, WIP) & NVIDIA Grace (JUPITER, WIP)
  - Also works on laptops, in virtual machines in the cloud, on Raspberry Pi boards, etc.
- Different accelerators: **NVIDIA GPUs** (today) + **AMD GPUs** (soon)
  - Available combination: AMD Rome (Zen2) + NVIDIA A100 (cc80), AMD Milan (zen3) + NVIDIA A100 (cc80) and AMD Genoa (zen4) + NVIDIA H100 (cc90) only software installations for AMD Rome (Zen2) + NVIDIA A100 are available
- **Various interconnects** like Infiniband, via “fat” MPI libraries
  - Support for injecting a vendor-provided MPI library is available
- Goal is to support system architecture of **all** (current & future) **EuroHPC systems**

On which systems is EESSI already available?



- EuroHPC JU systems:
  - Native installation (via CernVM-FS) on **Vega** + **Karolina + Deucalion**
  - Semi native installation (via rsync) on **MareNostrum5**
  - EESSI can be used via `cvmfsexec` tool on Deucalion, Discoverer, MeluXina (see blog post)
  - Native installation on **MeluXina**, **Lumi** is a work-in-progress
  - JSC has expressed significant interest to make EESSI available on **JUPITER**
- EESSI is already available on various other European systems (and beyond)
  - Snellius @ SURF, EMBL, Univ. of Stuttgart, VSC sites in Belgium, Sigma2 in Norway, etc.
- **Overview of (known) systems that have EESSI available at eessi.io/docs/systems**

NVIDIA GPU support in EESSI



- Initial support for CUDA software is in place in EESSI version 2023.06
- Detailed documentation available at eessi.io/docs/gpu
- Problems we had to deal with:

    1) We don't know where the **NVIDIA GPU driver libraries** are in host OS…

    2) We **can not redistribute the full CUDA installation** due to EULA, only runtime libraries…

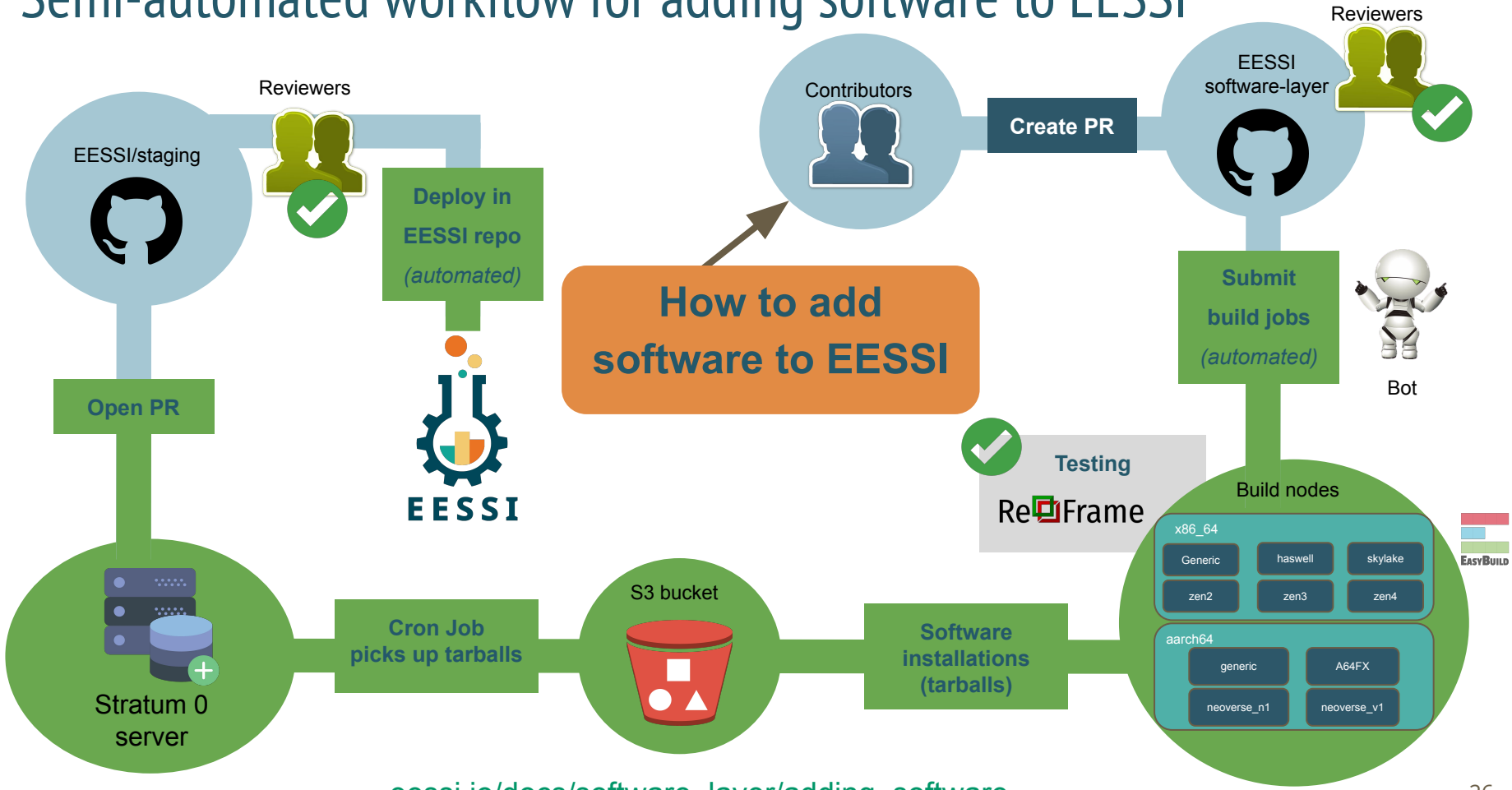
- In EESSI, we provide scripts to deal with both these problems:

    1) `link_nvidia_host_libraries.sh` script to link GPU driver libraries provided by OS "into" EESSI; (requires write access to (target of) `/cvmfs/software.eessi.io/host_injections`)

    2) `install_cuda_host_injections.sh` script to **install full CUDA installation** to subdirectory of (target of) `/cvmfs/software.eessi.io/host_injections` (and unbreak symlinks in CUDA in EESSI)

- **Available CUDA software in EESSI:** CUDA-Samples, ESPResSo, LAMMPS, NCCL, OSU Micro-Benchmarks
- More CPU/GPU combos and software (GROMACS, PyTorch, TensorFlow, AlphaFold) coming soon…

Semi-automated workflow for adding software to EESSI



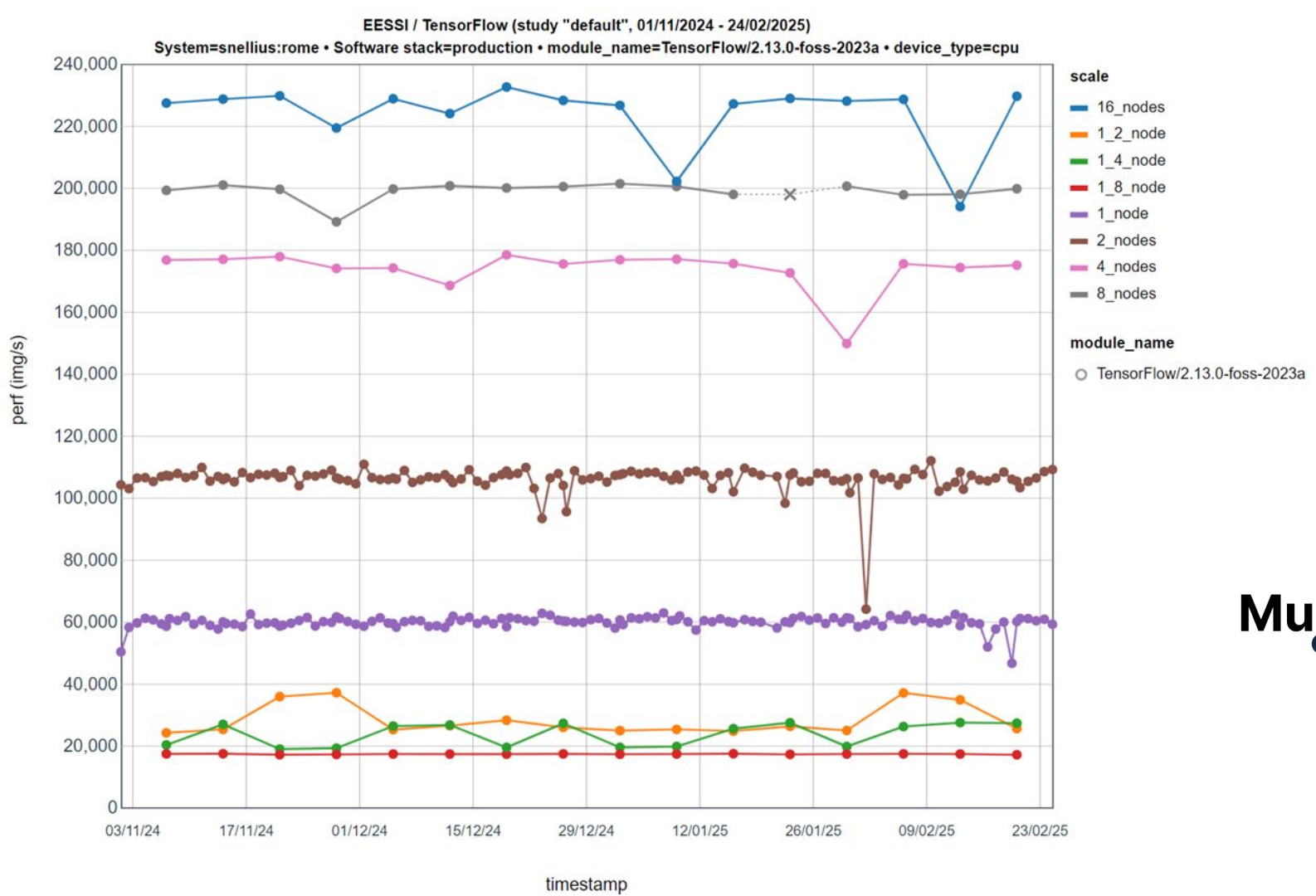
eessi.io/docs/software_layer/adding_software

Software testing is an important part of EESSI

- EESSI test suite: eessi.io/docs/test-suite
  - Collection of *portable* tests for software available in EESSI
  - Periodically (daily/weekly) on about 6 different systems
  - Running of selected (single node) tests when building new Software for EESSI (before deployment)
  - Can also be used for other software stacks than the EESSI software stack

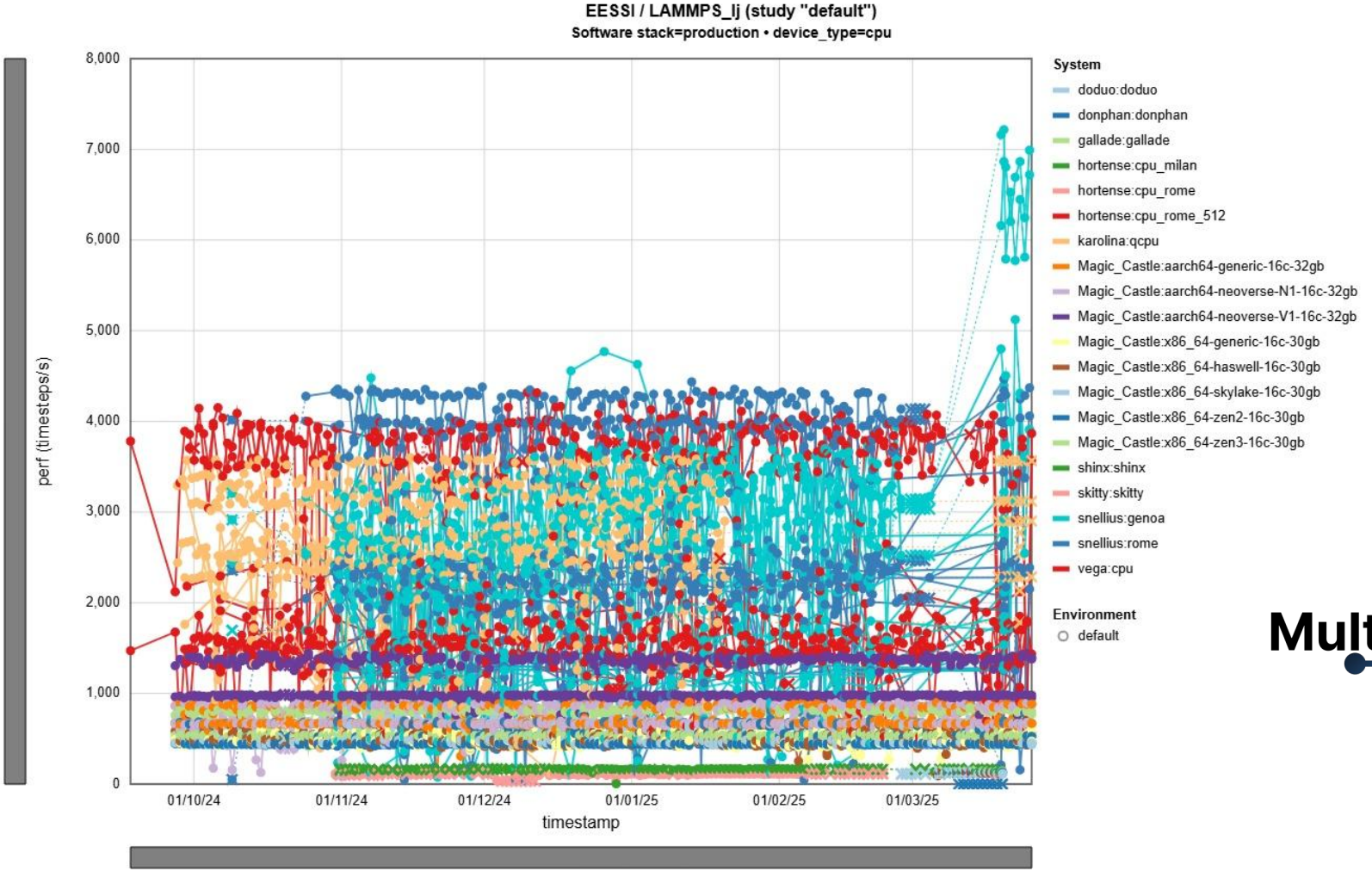
Software testing is an important part of EESSI

- Example: failing tests in GROMACS test suite when installing it in EESSI
  - See https://gitlab.com/eessi/support/-/issues/47
  - Filesystem race in GROMACS test suite when running tests concurrently
  - **Bug in Arm SVE support**, leading to (very) wrong results for several tests
    - See https://gitlab.com/gromacs/gromacs/-/issues/5057
    - Works fine on A64FX (512-bit SVE), but problem on Graviton 3 + NVIDIA Grace!



EESSI

Multi scale



EESSI

Multi scale

EESSI test suite

- 13:50-14:10 Hands-on: run the test suite yourself
- ideally: be ready to login to your own HPC
    - If your system is very full, maybe try to reserve one (ideally CPU) node
- Alternative: local laptop with module env
- You'll need one of the following modules
    - Torchvision, Pytorch-bundle, TensorFlow, LAMMPS, EspresSo, QuantumEspresso, CP2K, Gromacs, Metalwalls, OSU
    - If you don't have any: install OSU with EasyBuild - it's the fastest install

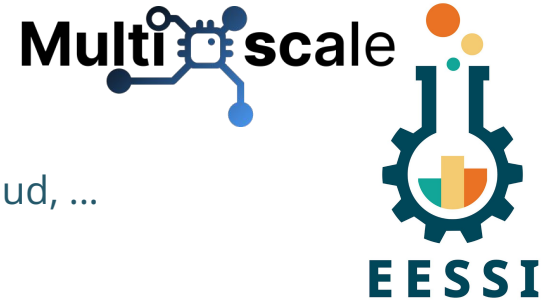
EESSI use cases

Use cases enabled by EESSI

- A **uniform software stack** across HPC clusters, clouds, laptops
- Enable **portable workflows**
- Can be leveraged in **continuous integration (CI)** environments
- Significantly facilitates setting up infrastructure for **HPC training**
- Enhanced **collaboration with software developers** and application experts

Also discussed in our open-access paper, available via doi.org/10.1002/spe.3075

EESSI as a uniform software stack



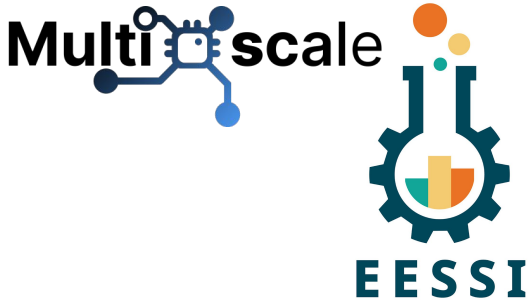
- Main goal: **same software everywhere**: laptop, server, HPC, cloud, …
- Wide variety of systems supported
  - CPUs: `x86_64` (Intel, AMD), `aarch64` (Arm), `riscv64` (work-in-progress…)
  - OS: any Linux distribution, on Windows via WSL, on macOS via Lima
  - High-speed interconnects (Infiniband), GPUs, etc.
- **Without compromising on software performance**
  - Optimized software installations for specific CPU microarchitectures + auto-detection
  - Large contrast with generic binaries often used in containers
- **Facilitates migrating** from laptop to HPC, cloud bursting, …

EESSI enables portable workflows

- Portable workflows are significantly easier when relying on EESSI
- They often involve running a broad set of tools, which all need to be available
- Workflows definitions (Snakemake, Nextflow,...) can leverage (or be included in) EESSI
- You can ship your execution environment inside your git repository using `direnv`
- If your users have EESSI and `direnv`, then can start running your workflow after cloning!

Alan

Leveraging EESSI in CI environments



- 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, …)

Leveraging EESSI in CI environments

We have an EESSI GitHub Action that provides EESSI+`direnv`:

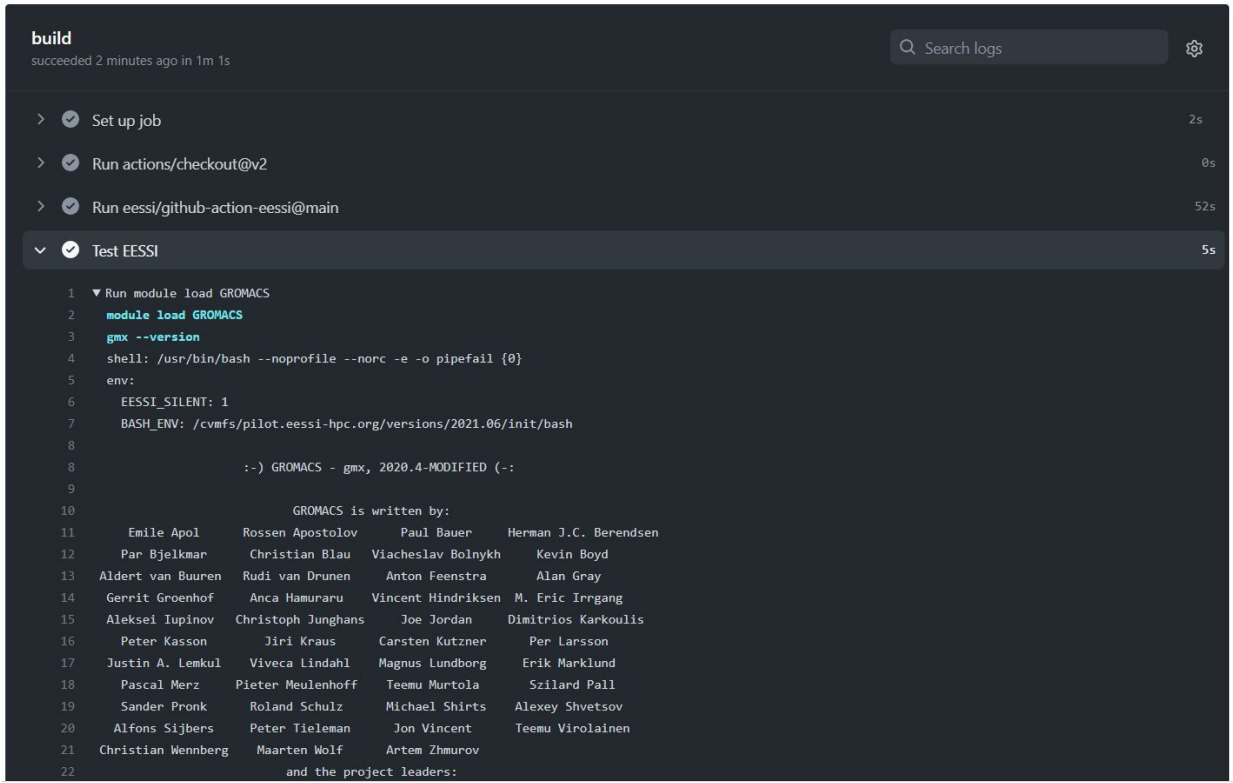
See it in action in the `github-eessi-action` repository:

github.com/EESSI/github-action-eessi

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

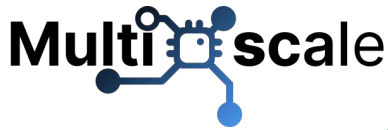
```yaml
name: ubuntu_tensorflow
on: [push, pull_request]
jobs:
 build:
   runs-on: ubuntu-latest
   steps:
   - uses: actions/checkout@v3
   - uses: eessi/github-action-eessi@v3
     with:
       eessi_stack_version: '2023.06'
   - name: Test EESSI
     shell: bash
     run: |
       module load TensorFlow
       python -c 'import tensorflow; print(tensorflow.__version__)'
```

Leveraging EESSI GitHub Action



https://github.com/EESSI/github-action-eessi/actions/runs/11183032689/job/31090668500

Facilitate HPC training



- EESSI can significantly reduce effort required to set up infrastructure for HPC training sessions (introductory, software-specific, …)
- Setting up a throwaway Slurm cluster in the cloud is easy via **Magic Castle**
  - Simple process once you have cloud credits:
    - Create cluster by editing a single file
    - Automatically configured within 20 minutes
    - Includes support for GPU and fast interconnects (Infiniband, EFA)
  - EESSI project uses Magic Castle for some of the build-and-deploy “bots”
  - There are also commercial alternatives that can/do support EESSI (Azure/AWS)
- **EESSI can provide (scientific) software that is required for the training**
- Attendees can easily set up the same software environment later on their own system(s) by leveraging EESSI

Alan

Collaboration with software developers + experts

- A central software stack by/for the community opens new doors…
- We can **work with software developers/experts** to verify the installation
  - Check how installation is configured and built
  - Help to verify whether software is functional for different use cases
  - Show us how to do extensive testing of their software
  - Evaluate performance of the software, enable performance monitoring
  - *"Approved by developers"* stamp for major applications included in EESSI
- Relieve software developers from burden of getting their software installed
  - Remove need to provide pre-built binary packages?
- Developers can also leverage EESSI themselves: dependencies, CI, …

Deployment of test-release of Scientific Software with EESSI

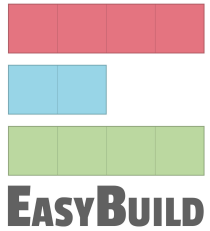
- **Dev.eessi.io**
  - Available on Vega
  - https://eessi.io/docs/repositories/dev.eessi.io
  - More information Coming soon

```
/cvmfs/dev.eessi.io/versions/2023.06/software
`-- linux
    `-- x86_64
        |-- amd
        |   |-- zen2
                |-- modules
                    |-- all
                        |-- Espresso
                        |   |-- 4.2.2-foss-2023a-2ba17de6096933275abec0550981d9122e4e5f28.lua
                        |-- Gromacs
                        |   |-- 2024.3-foss-2023b-d0f934abfd1394621c40858a2c2dd9123451df4e.lua
                        |-- LAMMPS
                        |   |-- 570c9d190fee556c62e5bd0a9c6797c4dffcc271-foss-2023a-kokkos-dev_OBMD.lua
                        |-- ...
```

EESSI won an HPCWire Reader's Choice award!

EESSI

Webinar series: Different aspects of EESSI



5 Mondays in a row May-June 2025 (subject to change)

- **CernVM-FS for HPC sites**
- **Introduction to EESSI webinar/tutorial**
- **Using EESSI as the base for a system stack webinar**
- **Introduction to EasyBuild**
- **EESSI for CI/CD webinar/tutorial**

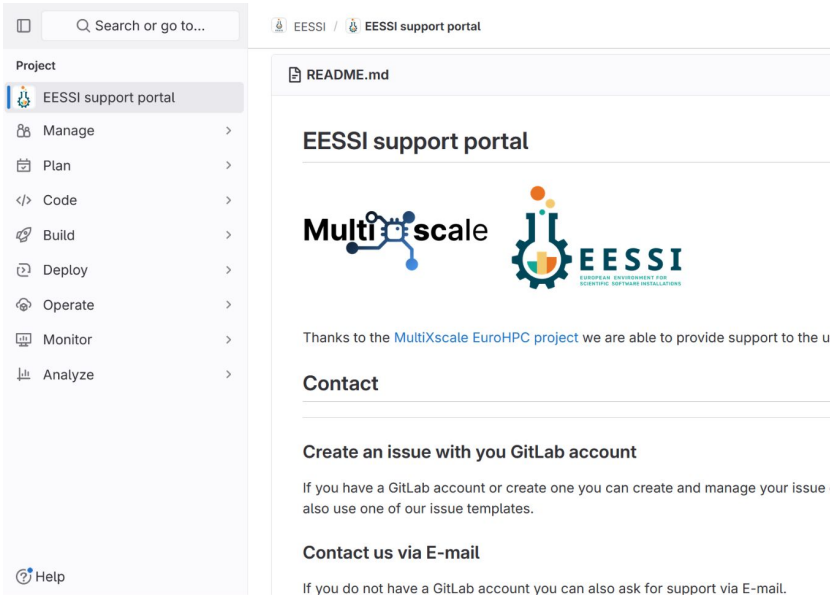
More details will be announced soon

Support for installing, using, contributing to EESSI

GitLab

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, ...)



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

Tutorial *"Best Practices for CernVM-FS in HPC"*

- multixscale.github.io/cvmfs-tutorial-hpc-best-practices
- Held online on 4 Dec 2023 (~3 hours), recorded & available on YouTube
- Over 200 registrations, ~125 attending the meeting
- Lecture + hands-on demos
- Topics:
  - Introduction to CernVM-FS + EESSI
  - Configuring CernVM-FS: client, Stratum 1 mirror server, proxy server
  - Troubleshooting problems
  - Benchmarking of start-up performance w/ TensorFlow
- **We are planning to organise this again soon…**

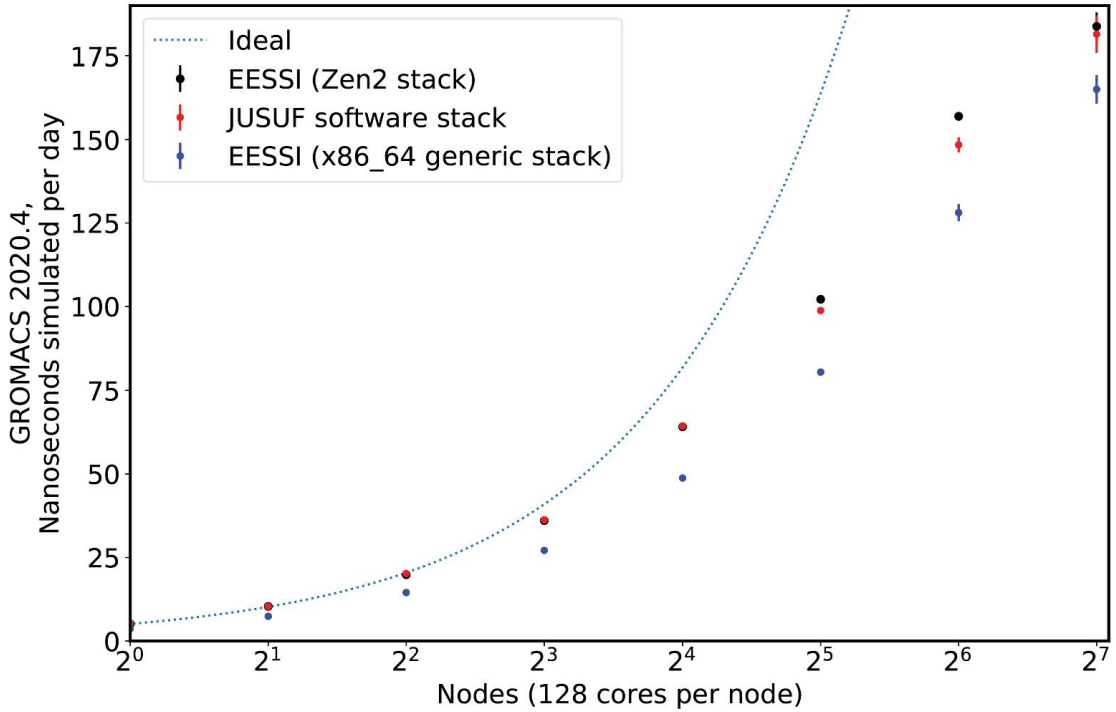
CernVM-FS

EESSI

Kenneth

EESSI paper (open access, Feb'22)

doi.org/10.1002/spe.3075



Paper includes proof-of-concept performance evaluation compared to system software stack, performed at JUSUF @ JSC using GROMACS 2020.4, up to 16,384 cores (CPU-only)

Kenneth

Website: eessi.io

GitHub: github.com/eessi

Documentation: eessi.io/docs

Blog: eessi.io/docs/blog

Join the EESSI Slack

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)

Kenneth

MultiXscale

Co-funded by the European Union

EuroHPC Joint Undertaking

Web page: multixscale.eu

Facebook: MultiXscale

Twitter: @MultiXscale

LinkedIn: MultiXscale

Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and countries participating in the project under grant agreement No 101093169.