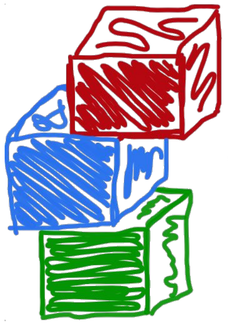


easybuild

6th EasyBuild User Meeting

Jan 25-29 2021, online

EESSI

EUROPEAN ENVIRONMENT FOR
SCIENTIFIC SOFTWARE INSTALLATIONS

European Environment for Scientific Software Installations

Bob Dröge, University of Groningen, The Netherlands

About me

EESSI

- Team High Performance Computing
- Center for Information Technology
- University of Groningen

- HPC user support and training, installing software
- HPC system administration
- Currently involved in two large projects: Euclid (ESA project/mission) and EESSI

university of groningen

EESSI

EUROPEAN ENVIRONMENT FOR
SCIENTIFIC SOFTWARE INSTALLATIONS

- **What** is the project about?
- **Who** is involved in EESSI?
- **Why** did we start it?
- **How** are we tackling the problem?
- Which **FOSS projects** do we use?
- What is the **current status**?
- **Live demo!**
- **Future work**

EESSI in a nutshell

EESSI

- **European Environment for Scientific Software Installations**
  (EESSI, pronounced as "easy")
- Collaboration between different partners in HPC community
- Goal:
  building a common scientific software stack for HPC & beyond

https://eessi-hpc.org

https://github.com/EESSI

https://eessi.github.io/docs/pilot

@eessi_hpc

EESSI partners & interested parties

Founding partners:

TU Delft

VU University Amsterdam

university of groningen

TU/e Eindhoven University of Technology

SURF

University of Twente.

University of Cambridge

Ghent University

DELL Technologies

Extensive interest from (HPC) community:

HPC Now!

Microsoft Azure

CSCS

CERN

aws

FRED HUTCH

JÜLICH Forschungszentrum

UiO : University of Oslo

HPC.NRW

Motivation

EESSI

- More scientists need to run large computations
- Explosion of open source scientific software in recent years
- Increasing variety in CPUs: Intel, AMD, Arm, POWER, RISC-V
- Various types of accelerators: NVIDIA & AMD GPUs, Intel Xe, ...
- Rise of the cloud: Amazon EC2, Microsoft Azure, Google, Oracle, ...
- In stark contrast: available manpower in HPC support teams...

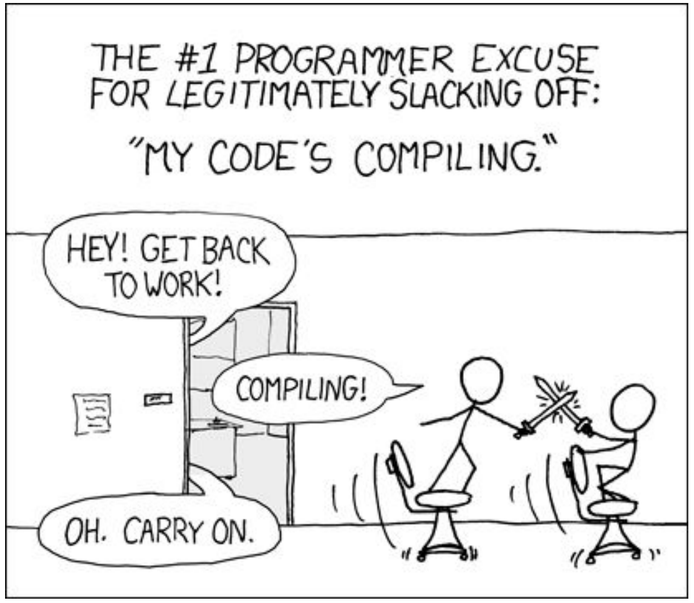
Getting Scientific Software Installed

https://xkcd.com/1654

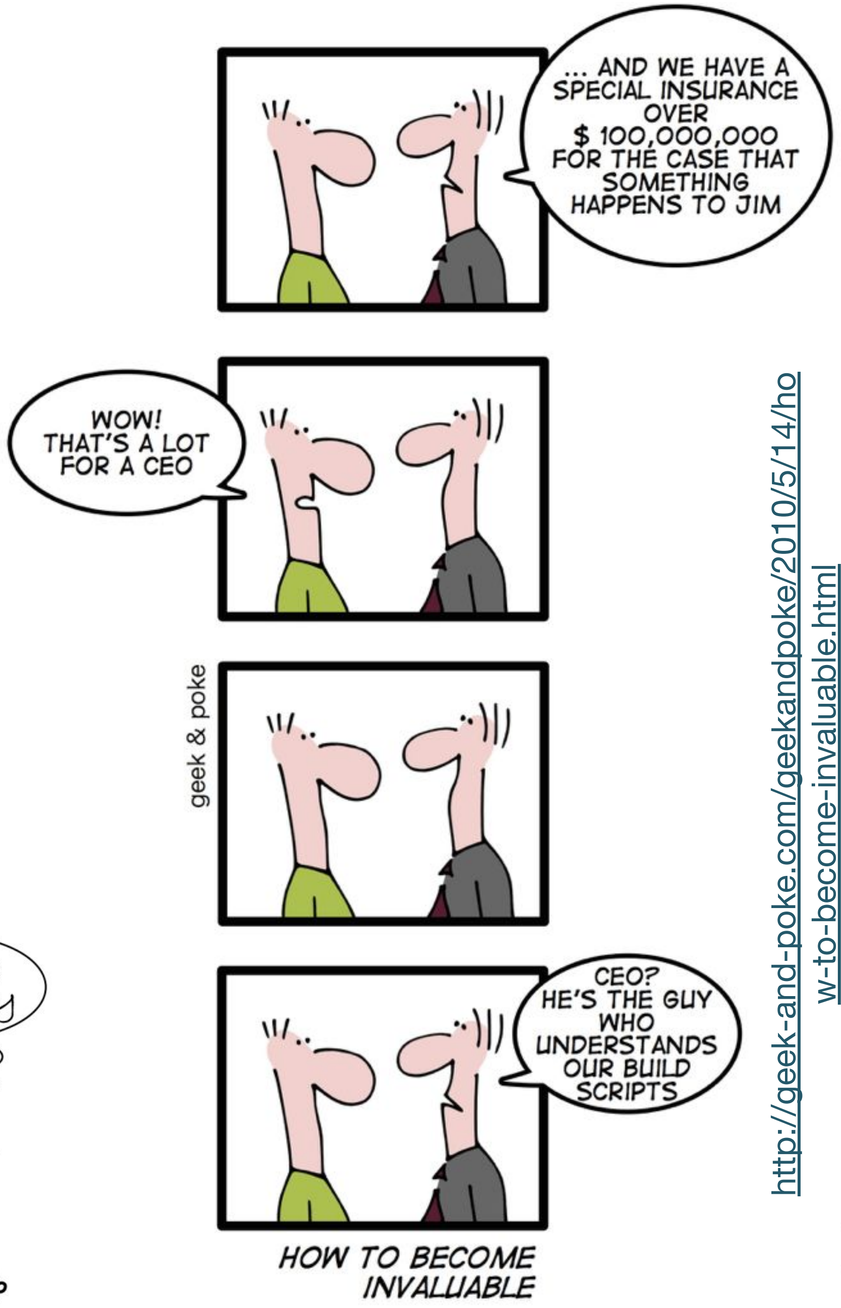
INSTALL.SH

```
#!/bin/bash

pip install "$1" &
easy_install "$1" &
brew install "$1" &
npm install "$1" &
yum install "$1" & dnf install "$1" &
docker run "$1" &
pkg install "$1" &
apt-get install "$1" &
sudo apt-get install "$1" &
steamcmd +app_update "$1" validate &
git clone https://github.com/"$1"/"$1" &
cd "$1";./configure;make;make install &
curl "$1" | bash &
```



https://xkcd.com/303



http://geek-and-poke.com/geekandpoke/2010/5/14/how-to-become-invaluable.html

How to make package managers cry

(or)

How to piss off package managers

(pick one)

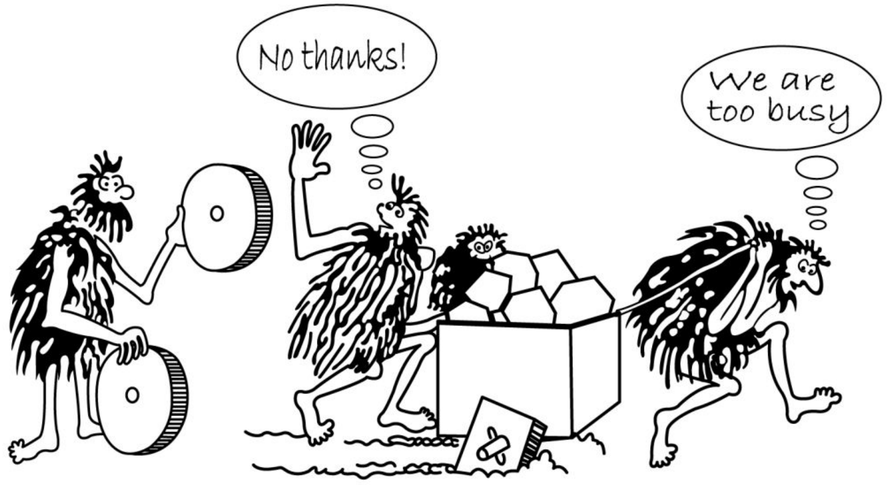
Kenneth Hoste

kenneth.hoste@ugent.be GitHub: @boegel Twitter: @kehoste

FOSDEM 2018

Package Management devroom

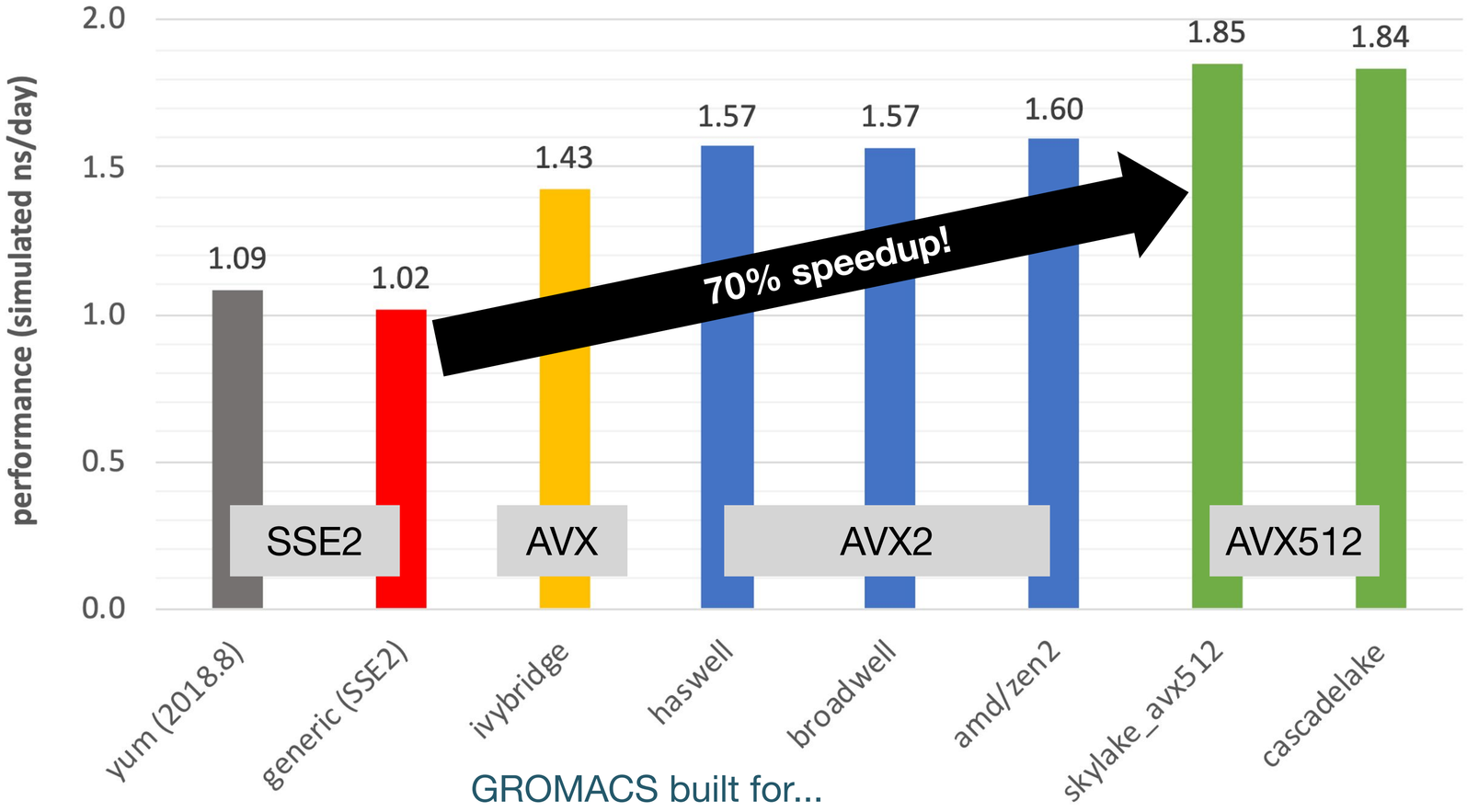
Feb 3rd 2018, Brussels (Belgium)



Keeping the P in HPC

- **Software should be optimised for the system it will run on**
- Impact on performance is often significant for scientific software

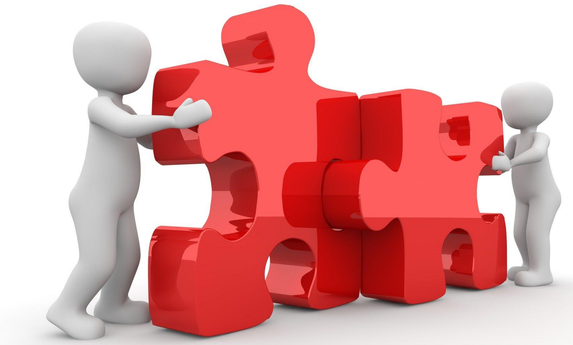
- Example: GROMACS 2020.1
  (PRACE benchmark, Test Case B)
- Metric: (simulated) ns/day,
  higher is better
- Test system: dual-socket
  Intel Xeon Gold 6420
  (Cascade Lake, 2x18 cores)



Scope & goals

EESSI

- **Shared repository of scientific software installations**
- **Collaborate**, avoid duplicate work across HPC sites
- Uniform way of providing software to researchers
- Should work on a **variety of systems**:
  - Any Linux distribution, macOS, Windows (via WSL)
  - From laptops and personal workstations to HPC clusters and the cloud
  - Support for different CPUs, interconnects, GPUs, etc
- Focus on **performance, automation, testing, collaboration**

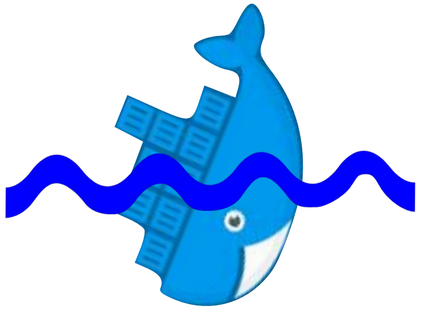


Inspiration for EESSI

- EESSI concept is **heavily** inspired by Compute Canada software stack
- Shared across 5 major national systems in Canada + a bunch of smaller ones
- **3 layers**: CernVM-FS / ~~Nix~~ Gentoo Prefix / EasyBuild + Lmod
- See paper by Maxime Boissonneault & co at PEARC'19 (PDF available here)
  "Providing a Unified Software Environment for Canada's National Advanced Computing Centers"
- See also Maxime's talk at 5th EasyBuild User Meeting (slides - recorded talk)
  and the Compute Canada documentation


alternative

Possible alternatives: containers

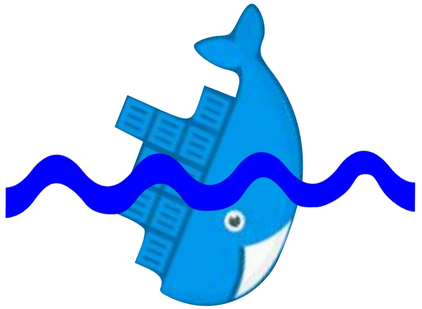


- “Native performance”

- “Mobility of compute”

SARUS

Possible alternatives: containers

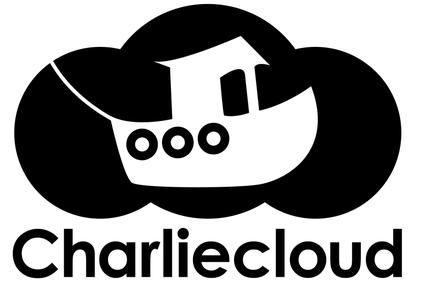


- “Native performance”

  ➔ "*very little overhead for the container engine*"

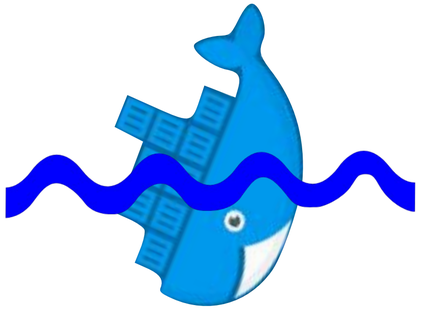
- “Mobility of compute”

  ➔ *“Large image with non-optimised software”*



SARUS

Possible alternatives: containers



- “Native performance”

  ➔ "*very little overhead for the container engine*"

- “Mobility of compute”

  ➔ *“Large image with non-optimised software”*

SARUS

Regarding scientific software installations for HPC clusters:

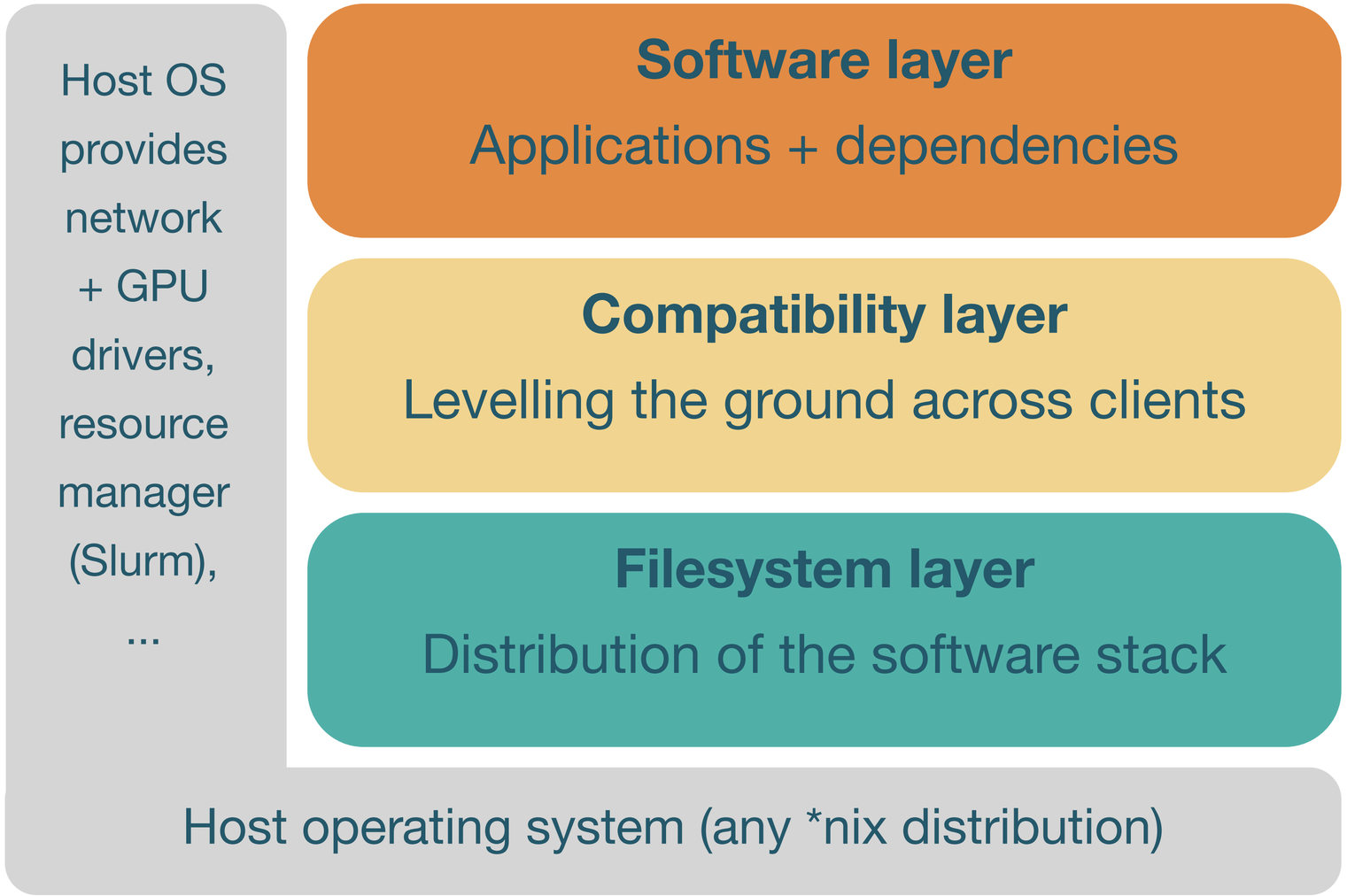
- How to properly optimise for different hardware?
- Who is going to build and maintain all these images?
- How to make it easily and quickly available to users?
- How to easily/natively support schedulers, MPI implementations, accelerators, ...?

Possible alternatives: E4S

- Extreme-scale Scientific Software Stack: https://e4s-project.github.io/
- Predefined set(s) of applications that are known to work well together
  - Spack environment YAML files
- Additionally, prebuilt containers and packages (Spack build cache) are offered
  - The (scientific) applications are not optimised!
  - The container images are large, e.g. the latest image with GPU support:


EESSI
EUROPEAN ENVIRONMENT FOR
SCIENTIFIC SOFTWARE INSTALLATIONS

High-level overview of EESSI project



EESSI is powered by FOSS (1/2)

- **Installation tool for scientific software**
- Optimises for build host (by default)
- Supports over 2,000 different pkgs

https://easybuild.io/eum

- Python library
- **Detects processor type**
- Check compatibility with host CPU

https://github.com/archspec

- **Environment modules tool** (in Lua)
- Intuitive access to software installations
- Multiple versions side-by-side

https://lmod.readthedocs.io

https://wiki.gentoo.org/wiki/Project:Prefix

- Gentoo: Linux distribution, installs from source
- Prefix subproject: **install packages in <prefix>**
- Supports x86_64, Arm64, POWER, …
- Supports both Linux and macOS

- Regression testing framework for HPC
- Tests are implemented as Python classes
- **Verify correctness**
- **Evaluate performance**

https://reframe-hpc.rtfd.io

CernVM-FS

https://cernvm.cern.ch/fs

- **Software distribution service** (software *installations*, not packages!)
- Scalable, read-only, globally distributed filesystem
- Served by web servers (HTTP only), no firewall issues
- Originally build for Large Hadron Collider (LHC) project at CERN

EESSI is powered by FOSS (2/2)

- Tool for automation and configuration management
- Using “playbooks” (YAML)
- **Used to automate deployment of filesystem and compatibility layer**

https://www.ansible.com

- Singularity: popular container runtime for HPC
- Own container image format
- Also consumes Docker containers
- **Used to fully control build environment for compatibility and software layer + (optionally) to let clients access EESSI**

https://sylabs.io/singularity

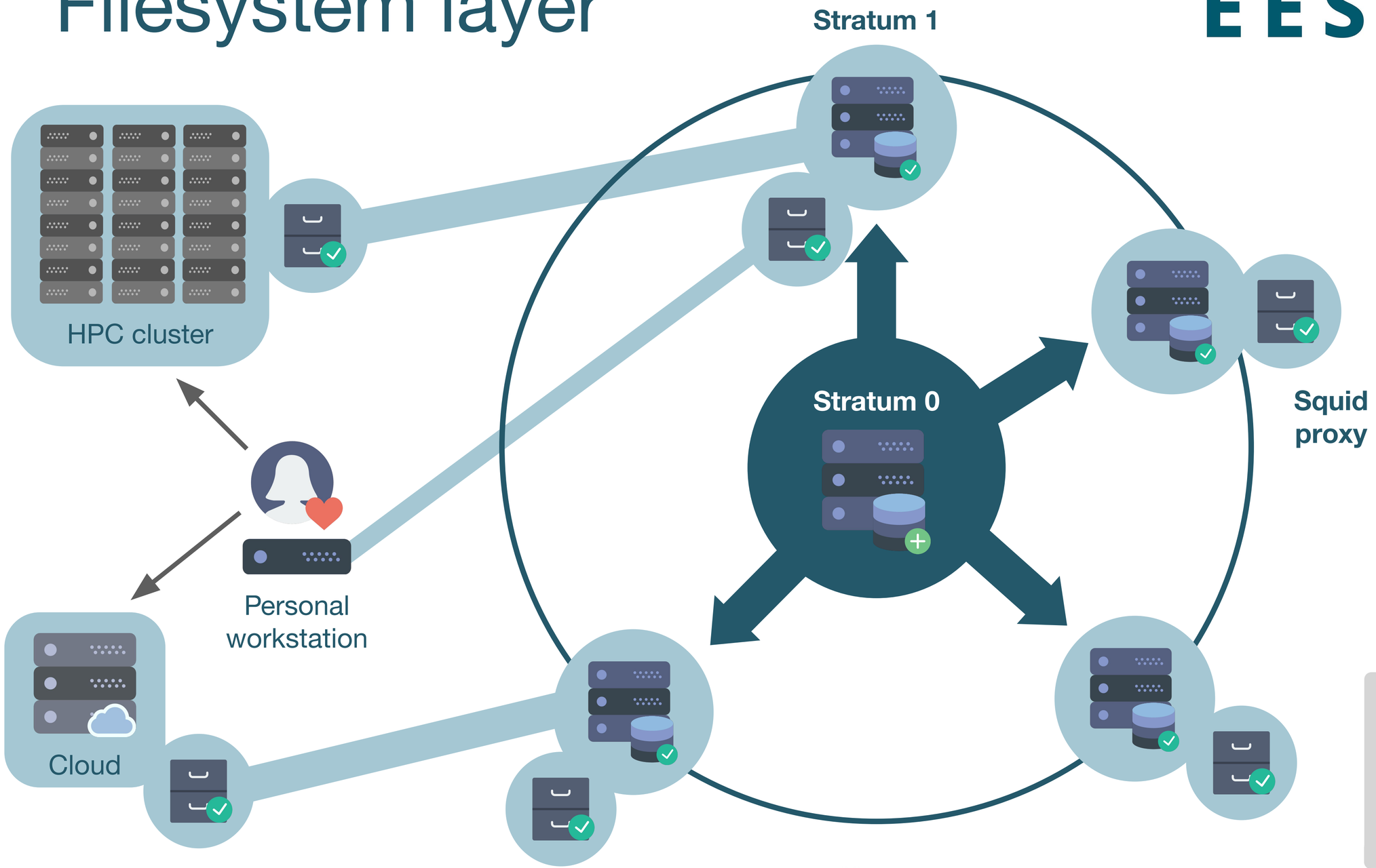
- “Infrastructure as code” tool
- Creating & managing cloud instances
- Declarative configuration files in custom DSL (HashiCorp Configuration Language - HCL)
- **Planning to use this for creating on-demand build/test nodes in AWS/Azure/...**

https://www.terraform.io

- CLI tool to easily create disposable Slurm clusters in the cloud
- Supports AWS, Oracle, Google cloud (Azure not yet supported)
- Leverages Ansible and Terraform in the background
- **Planning to use this to set up (heterogenous) Slurm clusters for building and testing software**

https://github.com/clusterinthecloud

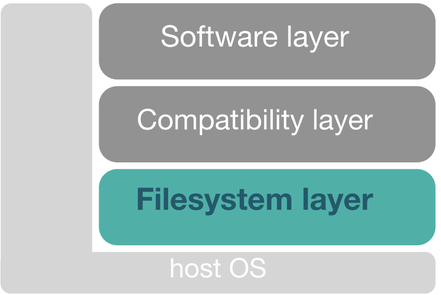
Filesystem layer



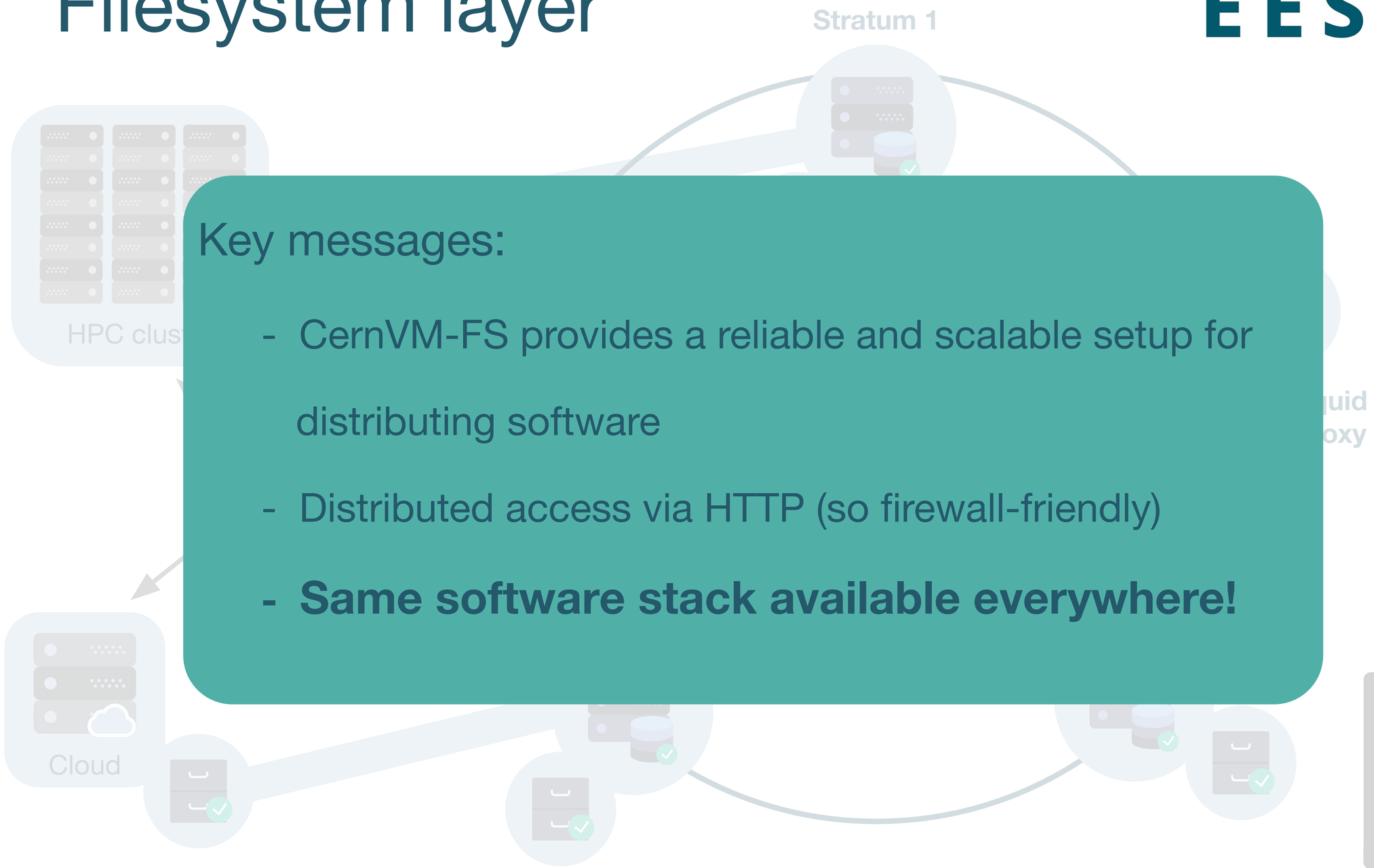
EESSI

powered by

CernVM-FS



Filesystem layer

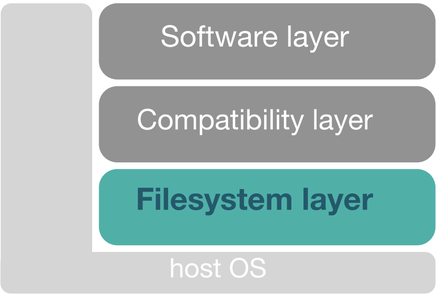


Key messages:

- CernVM-FS provides a reliable and scalable setup for distributing software
- Distributed access via HTTP (so firewall-friendly)
- **Same software stack available everywhere!**

powered by

CernVM-FS

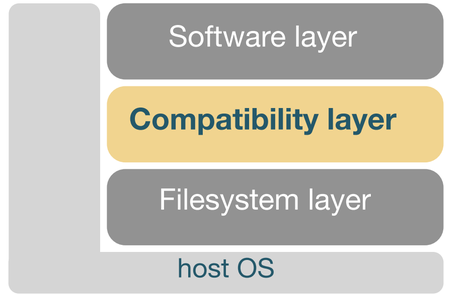


Compatibility layer

- **Gentoo Prefix** installation
- Set of tools & libraries installed in non-standard location
- Limited to low-level stuff, incl. glibc. No kernel or drivers.
- Only targets a supported processor **family** (x86_64, Arm64, ppc64le)
- **Levels the ground for different client operating systems** (Linux distros, macOS)
- Currently in pilot repository:

```
/cvmfs/pilot.eessi-hpc/2020.12/compat/linux/aarch64
/cvmfs/pilot.eessi-hpc/2020.12/compat/linux/ppc64le
/cvmfs/pilot.eessi-hpc/2020.12/compat/linux/x86_64
```

powered by



Software layer

- Provides scientific software applications, libraries, and dependencies
- **Optimised for specific CPU microarchitectures** (Intel Haswell, ...)
- **Leverages libraries from compatibility layer** (not from host OS)
- Installed with EasyBuild, incl. environment module files
  - Intention to start using Easystack files soon
- Lmod environment modules tool is used to access installations
- Different subdirectories/trees: one per CPU microarchitecture
- **Best subdirectory for host is picked automatically** via archspec

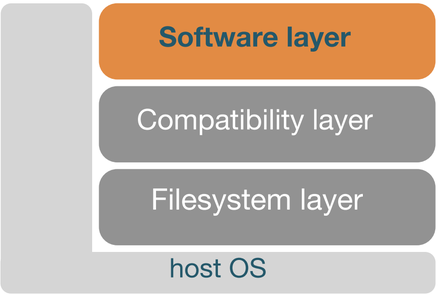
EESSI

powered by

easybuild

Lmod

arch spec



Current status: pilot repository

- Ansible playbooks, scripts, docs at `https://github.com/eessi`
- CernVM-FS: Stratum 0 @ Univ. of Groningen + two Stratum 1 servers
- Compatibility layer for both `x86_64` and `aarch64` (only Linux clients, for now)
- Software (CPU-only): Bioconductor, GROMACS, OpenFOAM, TensorFlow
- Hardware targets:
  - `x86_64/generic`, `intel/haswell`, `intel/skylake_avx512`, `amd/zen2`
  - `aarch64/generic`, `aarch64/graviton2`, `aarch64/thunderx2`

NOT FOR PRODUCTION USE!

Try it yourself: https://eessi.github.io/docs/pilot

From zero to science in three steps

1. Access the EESSI CernVM-FS repo
2. Source the EESSI init script
3. Compute!

➔ Native installation of the CernVM-FS client (requires admin privileges)

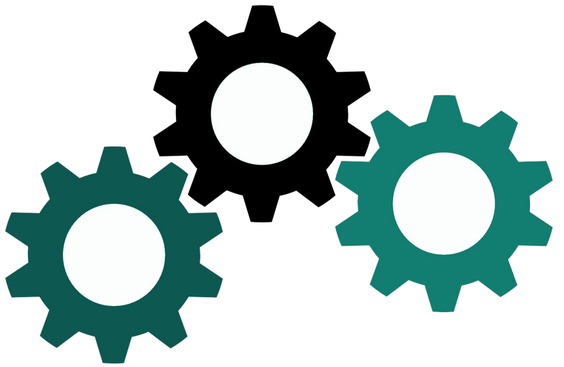
➔ Singularity container (no admin privileges required)

Detect your microarchitecture,

find the right software tree,

set up your environment.

Load the module(s) that you need, and start running!



Step 1: Access the EESSI repository

Option 1 (example):

native CernVM-FS installation on fresh (x86_64) RHEL 8.2 system

```
# install CernVM-FS client (see https://cernvm.cern.ch/fs/)
sudo yum install -y https://ecsft.cern.ch/dist/cvmfs/cvmfs-release/cvmfs-release-latest.noarch.rpm
sudo yum install -y cvmfs

# install CernVM-FS configuration files for EESSI repositories (see https://github.com/EESSI/filesystem-layer)
wget https://github.com/EESSI/filesystem-layer/releases/download/v0.2.3/cvmfs-config-eessi-0.2.3-1.noarch.rpm
sudo yum install -y cvmfs-config-eessi-0.2.3-1.noarch.rpm

# create local CernVM-FS configuration file (direct access, no proxy; 10GB for CernVM-FS cache)
sudo bash -c "echo 'CVMFS_HTTP_PROXY=DIRECT' > /etc/cvmfs/default.local"
sudo bash -c "echo 'CVMFS_QUOTA_LIMIT=10000' >> /etc/cvmfs/default.local"

# set up CernVM-FS
sudo cvmfs_config setup

# access EESSI pilot repository
ls /cvmfs/pilot.eessi-hpc.org/2020.12
```

Step 1: Access the EESSI repository

Option 2 (example, see https://eessi.github.io/docs/pilot):

use Singularity to run Docker container to access EESSI

```
# configure Singularity (bind mounts + home directory)
mkdir -p /tmp/$USER/{var-lib-cvmfs,var-run-cvmfs,home}
export SINGULARITY_BIND="/tmp/$USER/var-run-cvmfs:/var/run/cvmfs,/tmp/$USER/var-lib-cvmfs:/var/lib/cvmfs"
export SINGULARITY_HOME="/tmp/$USER/home:/home/$USER"

# values to pass to --fusemount (EESSI config + pilot repositories)
export EESSI_CONFIG="container:cvmfs2 cvmfs-config.eessi-hpc.org /cvmfs/cvmfs-config.eessi-hpc.org"
export EESSI_PILOT="container:cvmfs2 pilot.eessi-hpc.org /cvmfs/pilot.eessi-hpc.org"

# minimal Docker container from Docker Hub (includes CernVM-FS + EESSI configuration files)
export DOCKER_IMAGE="docker://eessi/client-pilot:centos7-$(uname -m)"

# start shell in Singularity container (ignore the scary looking 'setxattr' warnings, they're harmless)
singularity shell --fusemount "$EESSI_CONFIG" --fusemount "$EESSI_PILOT" $DOCKER_IMAGE

# access EESSI pilot repository
ls /cvmfs/pilot.eessi-hpc.org/2020.12
```

Step 1: Access the EESSI repository

Option 2 (example, see https://eessi.github.io/docs/pilot):

use Singularity to run Docker container to access EESSI

```
# configure Singularity (bind mounts + home directory)
mkdir -p /tmp/$USER/{var-lib-cvmfs,var-run-cvmfs,home}
export SINGULARITY_BIND="/tmp/$USER/var-run-cvmfs:/var/run/cvmfs,/tmp/$USER/var-lib-cvmfs:/var/lib/cvmfs"
export SINGULARITY_HOME="/tmp/$USER/home:/home/$USER"

# values to pass to --fusemount (EESSI config + pilot repositories)
export EESSI_CONFIG="container:cvmfs2 cvmfs-config.eessi-hpc.org /cvmfs/cvmfs-config.eessi-hpc.org"
export EESSI_PILOT="container:cvmfs2 pilot.eessi-hpc.org /cvmfs/pilot.eessi-hpc.org"

# minimal Docker container from Docker Hub (includes CernVM-FS + EESSI configuration files)
export DOCKER_IMAGE="docker://eessi/client-pilot:centos7-$(uname -m)"

# start shell in Singularity container (ignore the scary looking 'setxattr' warnings, they're
singularity shell --fusemount "$EESSI_CONFIG" --fusemount "$EESSI_PILOT" $DOCKER_IMAGE

# access EESSI pilot repository
ls /cvmfs/pilot.eessi-hpc.org/2020.12
```

Step 2: source the EESSI init script


```
# source the EESSI init script to set up your environment
$ source /cvmfs/pilot.eessi-hpc.org/2020.12/init/bash
Found EESSI pilot repo @ /cvmfs/pilot.eessi-hpc.org/2020.12!
Found Lmod configuration file at /cvmfs/pilot.eessi-hpc.org/2020.12/software/x86_64/intel/haswell/.lmod/lmodrc.lua
Initializing Lmod...
Prepending /cvmfs/pilot.eessi-hpc.org/2020.12/software/x86_64/intel/haswell/modules/all to $MODULEPATH...
Environment set up to use EESSI pilot software stack, have fun!

[EESSI pilot 2020.12] $ echo $EESSI_PREFIX
/cvmfs/pilot.eessi-hpc.org/2020.12

[EESSI pilot 2020.12] $ echo $EESSI_SOFTWARE_SUBDIR
x86_64/intel/haswell
```

Step 3: load your modules, and go!


```
# check which modules are available
[EESSI pilot 2020.12] $ module avail gromacs

---------- /cvmfs/pilot.eessi-hpc.org/2020.12/software/x86_64/intel/haswell/modules/all ----------
   GROMACS/2020.1-foss-2020a-Python-3.8.2

# load the module(s) for the software you want to use
[EESSI pilot 2020.12] $ module load GROMACS

# ready to compute!
[EESSI pilot 2020.12] $ gmx mdrun -s ion_channel.tpr -maxh 0.50 -resethway -noconfout -nsteps 1000
```

Demo time!

https://github.com/EESSI/eessi-demo

EESSI

EUROPEAN ENVIRONMENT FOR SCIENTIFIC SOFTWARE INSTALLATIONS

Future work

- Further improve pilot EESSI repository (monthly revisions)
- Identify problems, and fix them...
- More **automation** (Ansible, Terraform, …) and **testing** (ReFrame + GitHub Actions)
- Also support macOS / POWER / GPUs, add more software
- Let developers of scientific software validate the installation of *their* software
- Solicit more manpower, get project funded to make it sustainable
- Set up a consortium, and change the “European” in our name
- Work towards **production** setup…

Website: https://www.eessi-hpc.org

Join our mailing list & Slack channel

https://www.eessi-hpc.org/join

Documentation: https://eessi.github.io/docs

GitHub: https://github.com/eessi

Twitter: @eessi_hpc

Monthly online meetings (first Thursday, 2pm CET)