One Scientific Software Stack to Rule Them All
Bob Dröge, University of Groningen, The Netherlands
About me

• 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
• **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

• European Environment for Scientific Software Installations (EESSI, pronounced as "easy")

• Collaboration between different partners in HPC community

• Goal: building a common performant scientific software stack for HPC systems & beyond (cloud, workstations, ...)

https://eessi-hpc.org
https://github.com/EESSI

https://eessi.github.io/docs/pilot
@eessi_hpc
Project partners

TU Delft

TU/e Eindhoven University of Technology

DELL Technologies

UNIVERSITY OF TWENTE.

VU University Amsterdam

UNIVERSITY OF CAMBRIDGE

HPC Now!

HPC.NRW

University of Oslo

Ghent University

Jülich Forschungszentrum

Microsoft Azure

AWS
Motivation

• 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

```
#! /bin/bash

pip install "$1" &
easy_install "$1" &
brew install "$1" &
rm -rf install "$1" &
yum install "$1" &
dnf install "$1" &
docker run "$1" &
apt-get install "$1" &
pkg 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" | bosh &
```

---

How to make package managers cry
How to piss off package managers
(pick one)

Kenneth Hoste

kenneth.hoste@agent.be
GitHub: @boegel Twitter: @khoste

FOSDEM 2018
Package Management devroom
Feb 3rd 2018, Brussels (Belgium)

https://xkcd.com/303

...and we have a special insurance over $100,000 for the case that something happens to Jim

Wow! That's a lot for a CEO

CEO? He's the guy who understands our build scripts

How to become invaluable
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

• Shared repository of scientific software installations

• Collaborate, avoid duplicate work across HPC sites

• Uniform way of providing software to researchers

• Broad client platform support (Linux, macOS, Windows via WSL)

• Targets: laptops, personal workstations, HPC clusters, and the cloud

• Support/optimised for different CPUs, interconnects, GPUs, etc

• Focus on **performance**, automation, testing
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](#)
High-level overview of EESSI project

Software layer
Applications + dependencies

Compatibility layer
Levelling the ground across clients

Filesystem layer
Distribution of the software stack

Host operating system (any *nix distribution)

Host OS provides network + GPU drivers, resource manager (Slurm), ...

EESSI is powered by FOSS

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

[https://easybuilders.github.io/easybuild](https://easybuilders.github.io/easybuild)

**arch spec**
- Python library
- Detect processor type
- Check compatibility with host CPU

[https://github.com/archspec](https://github.com/archspec)

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

[https://lmod.readthedocs.io](https://lmod.readthedocs.io)

**gentoo**
- Linux distribution, installs from source
- Prefix project: install packages in `<prefix>`
- Supports x86_64, Arm64, POWER, ... & Linux, macOS


**CernVM-FS**
- Software distribution service
- Scalable, read-only, globally distributed filesystem
- Mount filesystem over HTTP

[https://cernvm.cern.ch/fs](https://cernvm.cern.ch/fs)

**ReFrame**
- Regression testing framework for HPC
- Verify correctness
- Check performance
- Tests are implemented in Python

[https://reframe-hpc.rfd.io](https://reframe-hpc.rfd.io)

**Ansible**
- Automation
- Configuration mgmt

**Singularity**
- Build isolation
- Easy access for clients

**Terraform**
- Creating cloud clusters / instances on demand
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!**
Compatibility layer

- **Gentoo Prefix** installation
- Set of tools & libraries installed in non-standard location
- Limited to low-level stuff, incl. glibc
- Only targets a supported processor **family** (x86_64, Arm64)
- **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/x86_64
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
- 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
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!
From zero to science in three steps

1. Access the EESSI CernVM-FS repo
   - Native installation of the CernVM-FS client (requires admin privileges)
   - Singularity container (no admin privileges required)

2. Source the EESSI init script
   - Detect your microarchitecture, find the right software tree, set up your environment.

3. Compute!
   - 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 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

```bash
# 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 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
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)