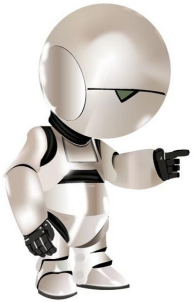


Semi-automated workflow for adding software to EESSI

Jörg Saßmannshausen (NIHR Biomedical Research Centre, UK)
Bob Dröge (Univ. of Groningen)
Kenneth Hoste (HPC-UGent)

7th EasyBuild User Meeting (virtual) - 28 Jan 2022

https://easybuild.io/eum22 https://eessi.github.io/docs

Agenda

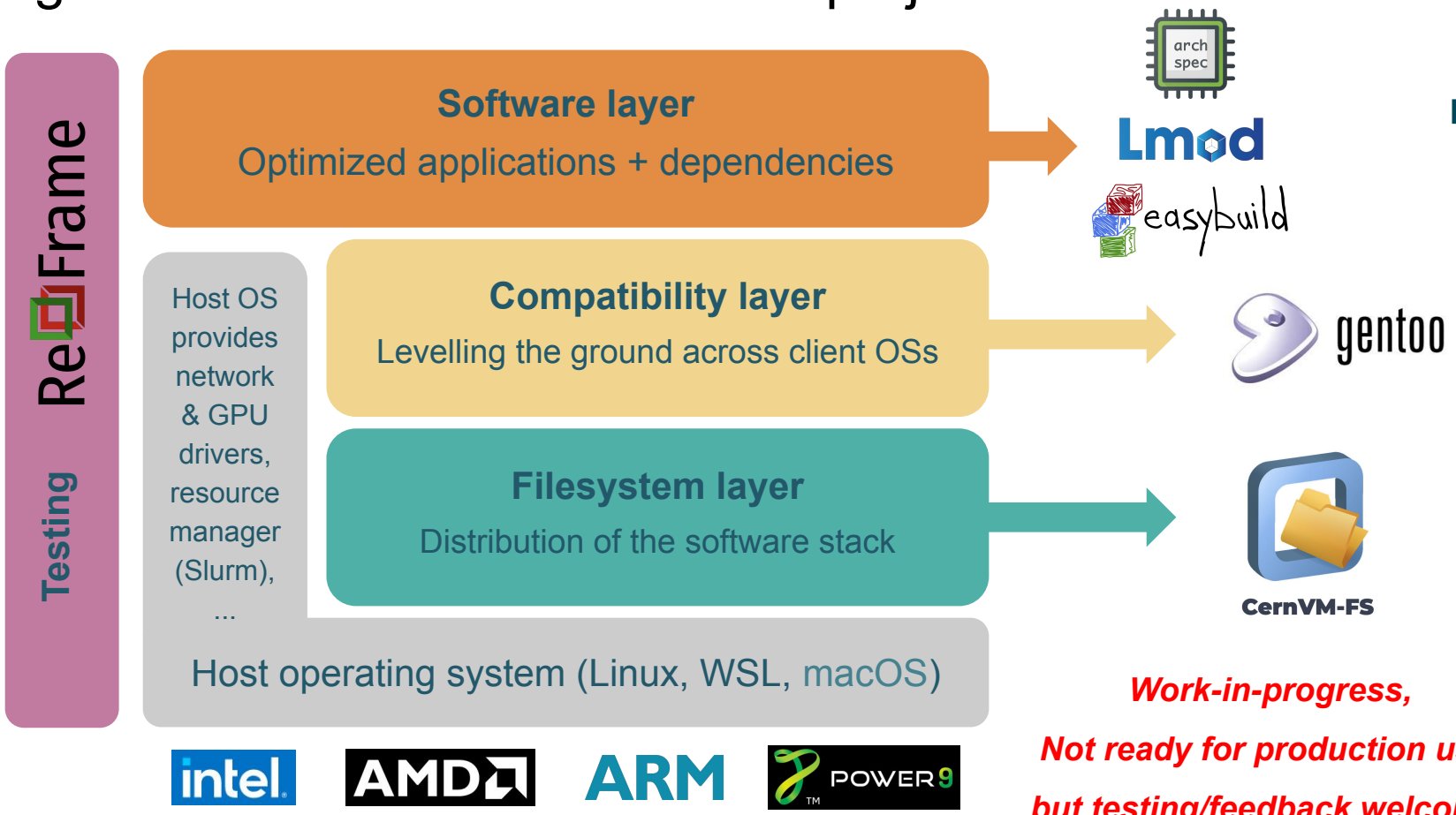
- EESSI in a nutshell
- Adding software to EESSI
- Goals of this project
- High-level overview
- Current status
- Live demo?
- Use cases beyond EESSI

What is EESSI?

https://www.eessi-hpc.org https://eessi.github.io/docs

- **European Environment for Scientific Software Installations** (EESSI)
- Collaboration between various HPC sites worldwide
- Goal: build a *common* stack of scientific software installations for HPC systems and beyond
  - Supported on HPC systems, cloud, personal workstations, etc.
  - Should work regardless of operating system (versions) on clients, CPU architecture, etc.
  - Focus on **performance**, automation, testing, etc.
- Kickstarted early 2020 by Dutch universities + Dell as collaborative project
  - HPC-UGent got involved early through EasyBuild
  - Original plan was to have a good excuse to drink beer together…
- EESSI community is now much broader: incl. Nordic HPC sites, CECAM, AWS, Azure, …
- **Work-in-progress: not ready (yet) for production usage!**

High-level overview of the EESSI project



EESSI

Work-in-progress,

Not ready for production use,

but testing/feedback welcome!

More information about EESSI

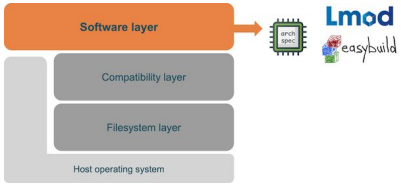
- **Introduction to EESSI** (preliminary) by Kenneth Hoste (HPC-UGent) at HPCKP'20;
  see https://github.com/EESSI/docs/tree/main/talks/20200618_HPCKP20

- **EESSI behind the scenes** by Kenneth Hoste (HPC-UGent) & Bob Dröge (Univ. of Groningen)
  https://github.com/EESSI/docs/tree/main/talks/20210119_EESSI_behind_the_scenes

- **Introduction to EESSI** by Bob Dröge (Univ. of Groningen) at EUM'21;
  see https://easybuild.io/eum21/#eessi

- **Getting Started with EESSI** by Thomas Röblitz (Univ. of Bergen) at EUM'22;
  see https://easybuild.io/eum22/#eessi-getting-started

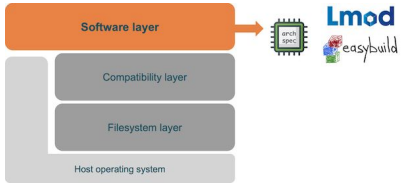
Software layer in EESSI



EESSI

- Includes scientific software applications + required dependencies
- **Software is auto-downloaded on-demand, in background, no installation!** (thanks to CernVM-FS)
- **Optimized for specific CPU microarchitectures** (Intel Haswell, AMD Rome, Arm Graviton2, etc.)
- **Best possible match for CPU microarchitectures of client is auto-detected** (thanks to archspec)
- Only links to OS libraries from compatibility layer (glibc, …)
- Software is installed with **EasyBuild**, exposed via environment module files (**Lmod**)
- Important aspects: RPATH linking, different ISAs (`x86_64`, `aarch64`, …), long installation prefix
- Extensive testing (functional, correctness, performance) with **ReFrame** *[PLANNED, work-in-progress]*

Software layer in EESSI



EESSI

Applications currently included in EESSI pilot repository (v2021.12):

GROMACS, OpenFOAM, QuantumESPRESSO, TensorFlow + Horovod, R + Bioconductor, WRF

(see easystack file at https://github.com/EESSI/software-layer/blob/main/eessi-2021.12.yml)

```
$ source /cvmfs/pilot.eessi-hpc.org/versions/2021.12/init/bash

Found EESSI pilot repo @ /cvmfs/pilot.eessi-hpc.org/versions/2021.12!

Using x86_64/intel/haswell as software subdirectory.
…
Environment set up to use EESSI pilot software stack, have fun!

$ module avail TensorFlow

--- /cvmfs/pilot.eessi-hpc.org/versions/2021.12/software/linux/x86_64/intel/haswell/modules/all ---

   TensorFlow/2.3.1-foss-2020a-Python-3.8.2

$ git clone https://github.com/EESSI/eessi-demo; cd eessi-demo/TensorFlow; ./run.sh

Epoch 1/5

1875/1875 [==============================] - 3s 1ms/step - loss: 0.2925 - accuracy: 0.9147
```

Adding software to EESSI (1/2)

- Current workflow:
  - Human runs software installation script in EESSI build container (on each target CPU arch.)
  - Human runs script to create tarball with added software installations + upload it to AWS S3 bucket
  - Cron script on CernVM-FS central server picks up new uploaded tarballs
  - Creates PR to (private) EESSI/staging repository on GitHub
  - Tarball is automatically ingested into EESSI pilot CernVM-FS repository when PR is merged
- Scripts available in https://github.com/EESSI/software-layer + https://github.com/EESSI/infrastructure
  - `install_software_layer.sh` to install EESSI software layer on top of compat layer
  - `build_container.sh` to easily run software installation script in EESSI build container
  - `create_tarball.sh` to create tarball for added installations (based on fuse-overlayfs upper dir)
  - `eessi-upload-to-staging` to upload into dedicated AWS S3 bucket (requires permissions)

Adding software to EESSI (2/2)

- Problems with current workflow:
  - Still way too manual and time-consuming: human babysitting + taking action
  - Doesn't allow (low-effort) contributions to EESSI software layer from people not familiar with workflow
  - Requires access to (growing) set of target CPUs
    - Different Intel/AMD CPU generations, Arm @ AWS, POWER9, soon also RISC-V?
    - In EESSI pilot v2021.12: `aarch64/generic`, `aarch64/graviton2`, `ppc64le/generic`, `ppc64le/power9`, `x86_64/generic`, `x86_64/amd/zen2`, `x86_64/amd/zen3`, `x86_64/intel/haswell`, `x86_64/intel/skylake_avx512`
  - Requires permissions to upload tarball into AWS S3 bucket for ingestion (who can we trust?)
  - How do we know that provided software builds are not tampered with in any way (knowingly or not)?

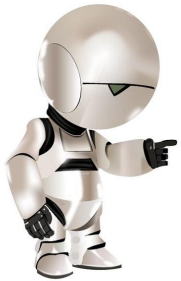
Towards a semi-automated workflow (1/2)

- Goal:
  - Allow contributors to propose additional software to include in EESSI
  - Ideally via a low effort interface: pull requests to GitHub
  - Automatic feedback on whether proposed integration into EESSI works
- Attention points: automation, performance, security, (minimal) human oversight, …
- Conditions for accepting contribution:
  - Software should work correctly in EESSI environment (compat layer, RPATH, long prefix, etc.)
  - Tests should be included to test end user applications (with ReFrame)
  - Software should build + tests should pass on all target CPUs (ideally)

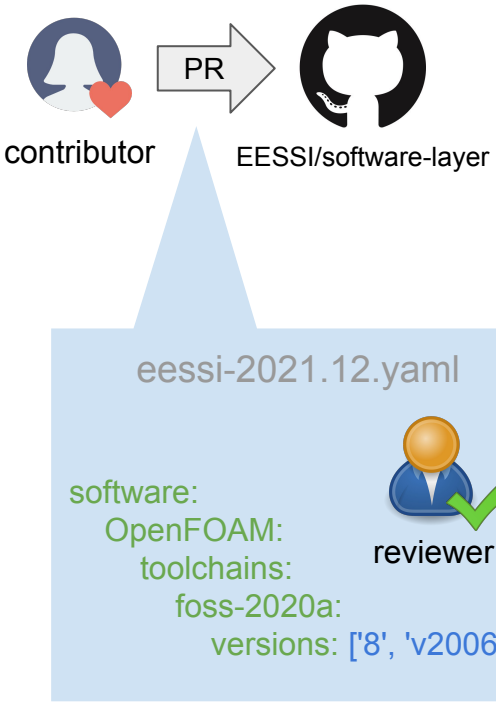
Towards a semi-automated workflow (2/2)

Implement a bot as a GitHub App:

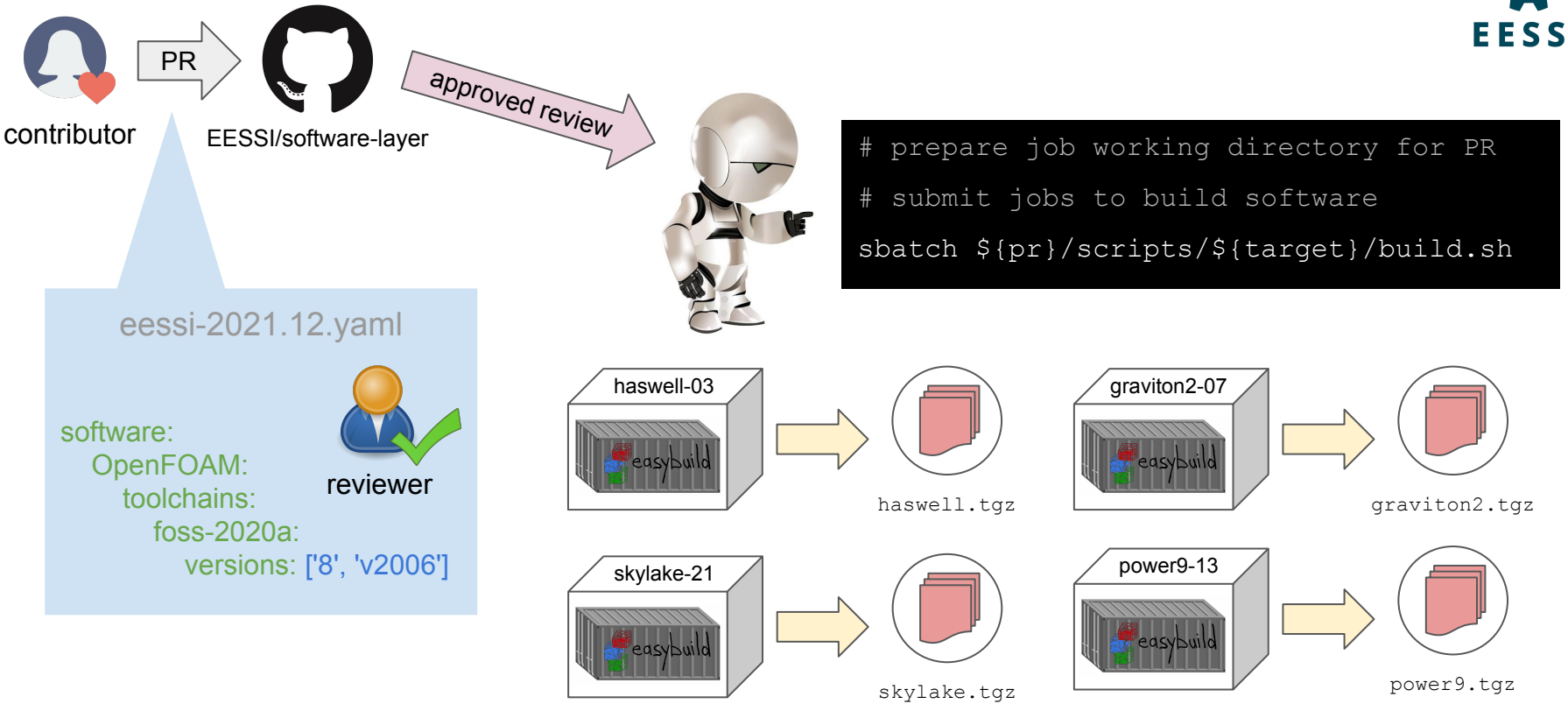
- In Python 3, using Flask (web app framework) + PyGithub (talk to GitHub API)
- Event-based bot that reacts to pull requests (PRs) to `EESSI/software-layer` repository
  - Events include: opening a PR, posting a comment, adding/removing a label, …
- Tasks:
  - Automatically build & install software for different target CPUs (no human intervention)
  - Using EESSI build container, on top of compat layer
  - Run tests to verify that software installation works (in different environment: OS, system, etc.)
  - Get software installations ingested into EESSI repository (after PR is merged?)



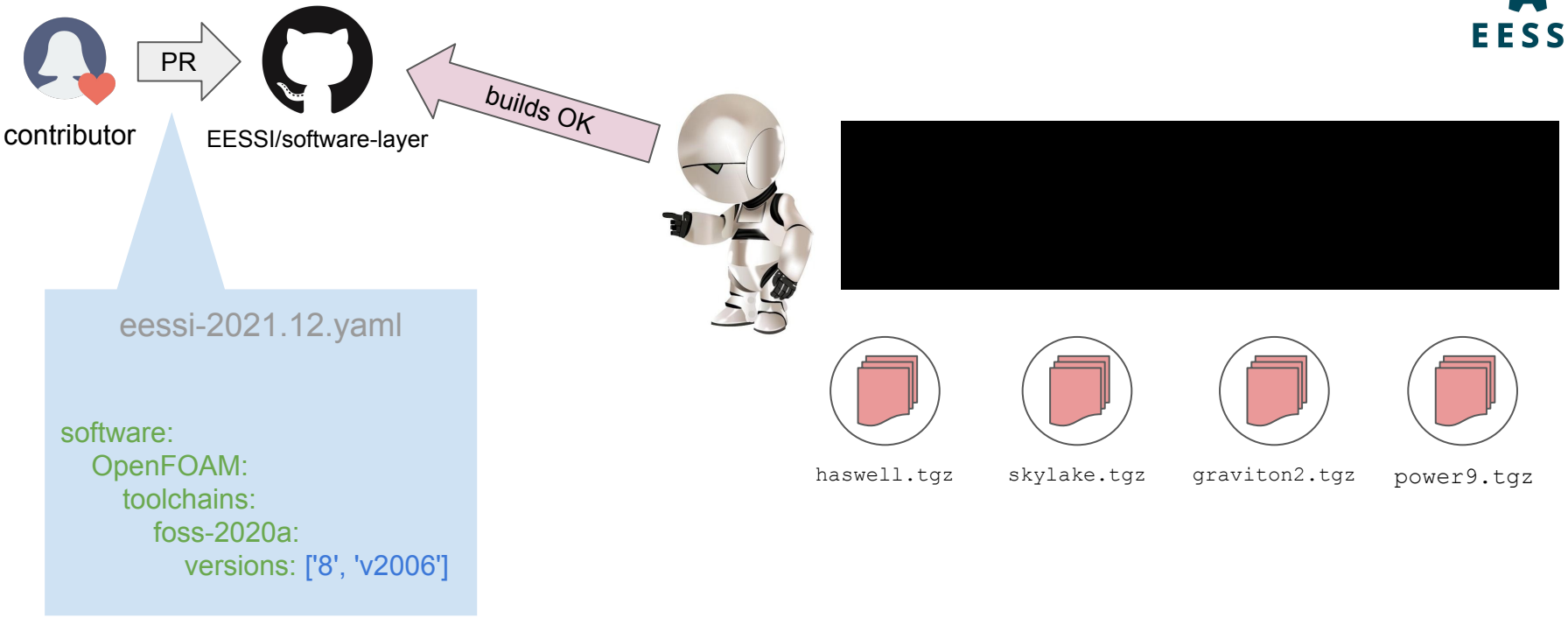
High-level overview of EESSI software bot



High-level overview of EESSI software bot

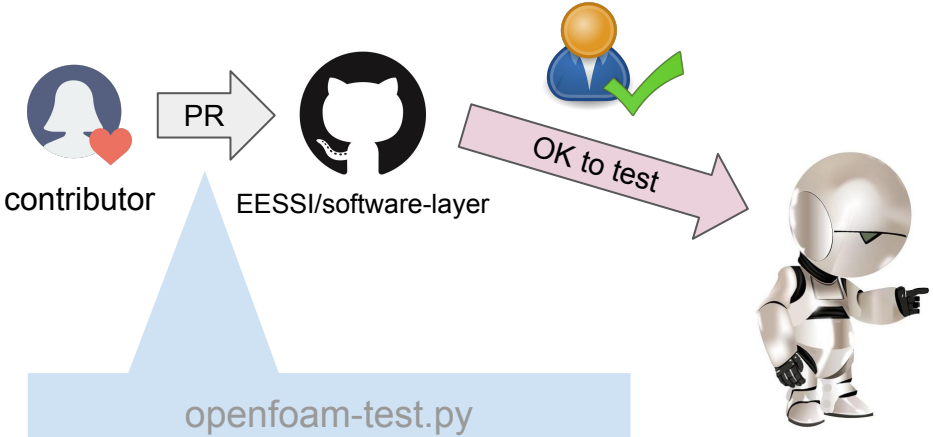


High-level overview of EESSI software bot



High-level overview of EESSI software bot

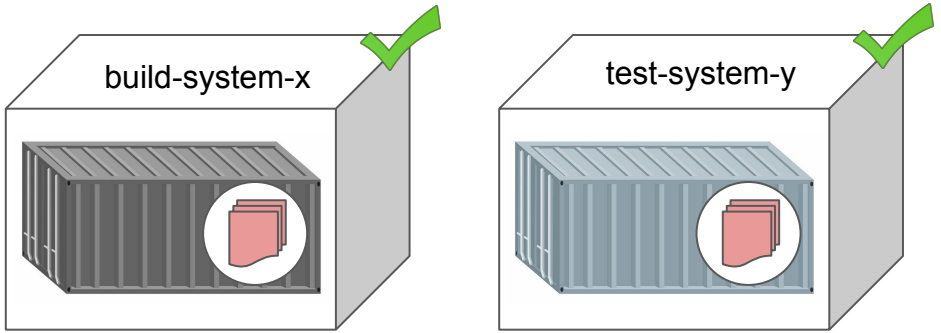
(simplified view)



openfoam-test.py

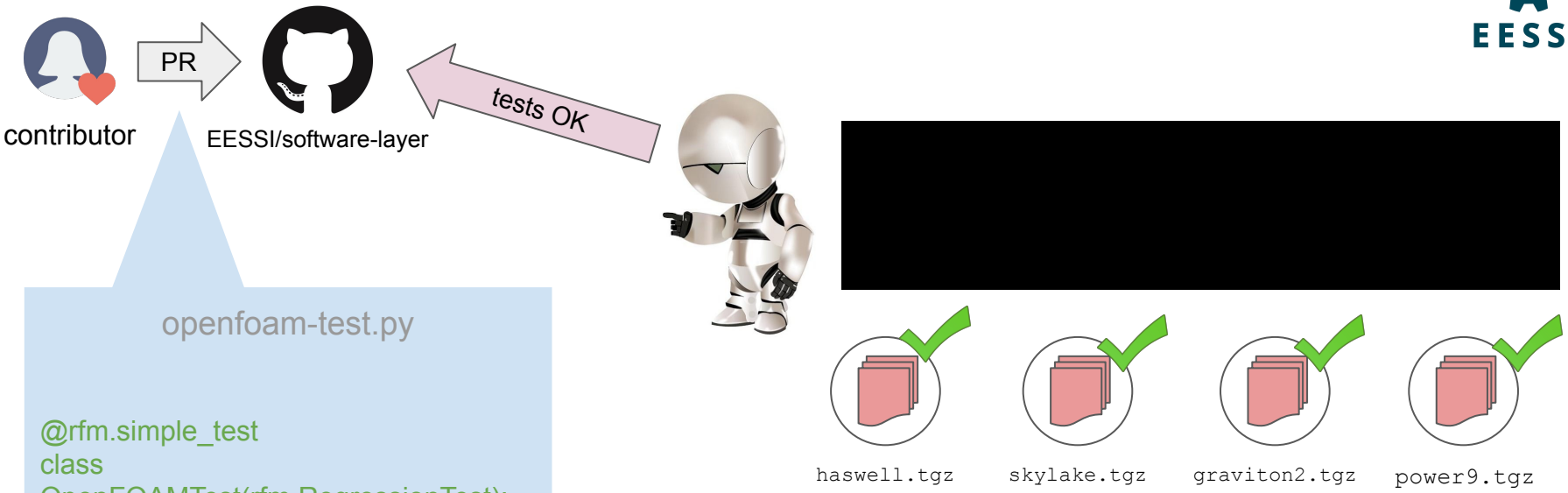
```
@rfm.simple_test
class
OpenFOAMTest(rfm.RegressionTest):
```

```
# submit jobs to test built software
sbatch ${pr}/scripts/${target}/test.sh
```

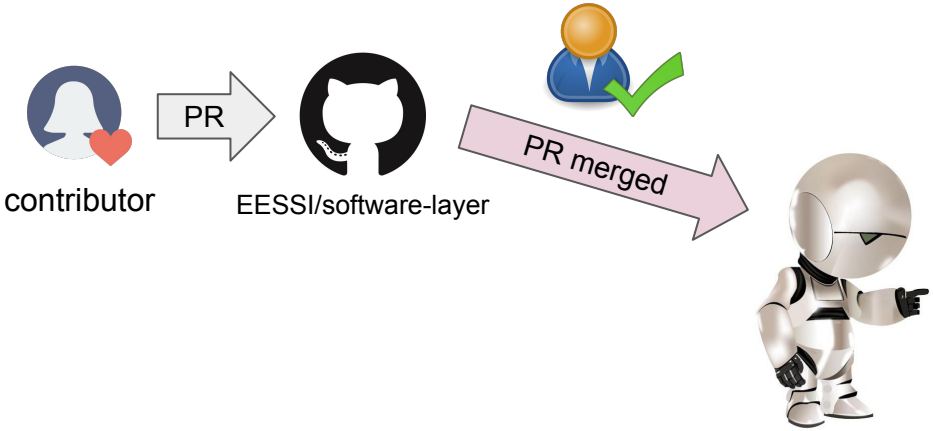


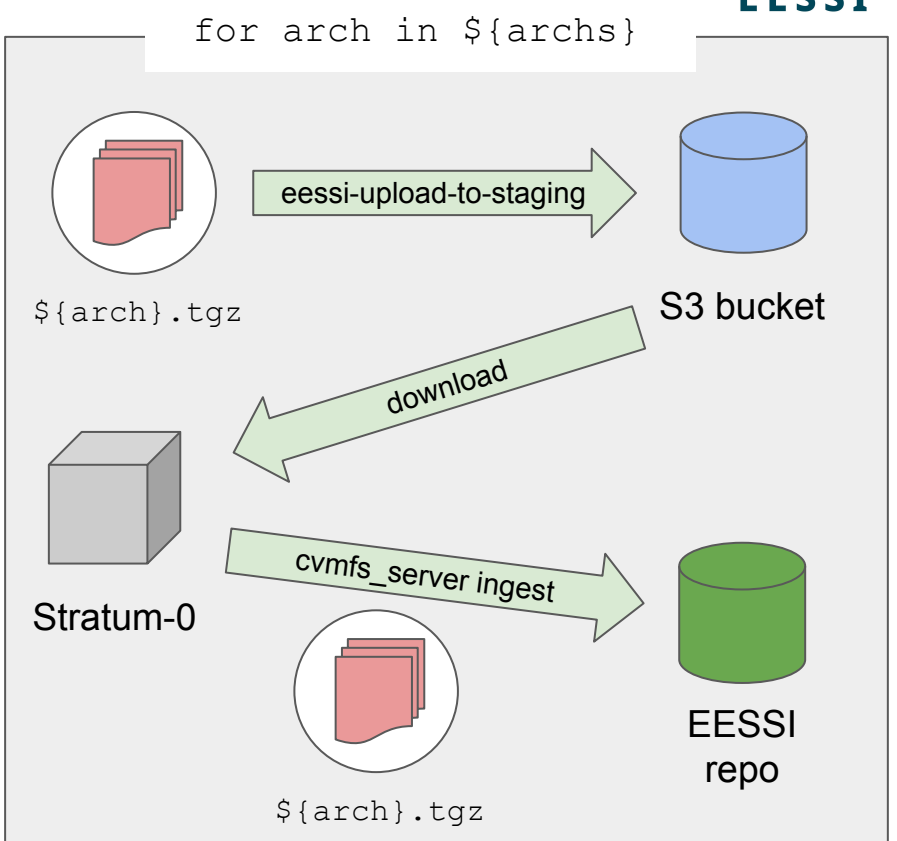
High-level overview of EESSI software bot

(simplified view)

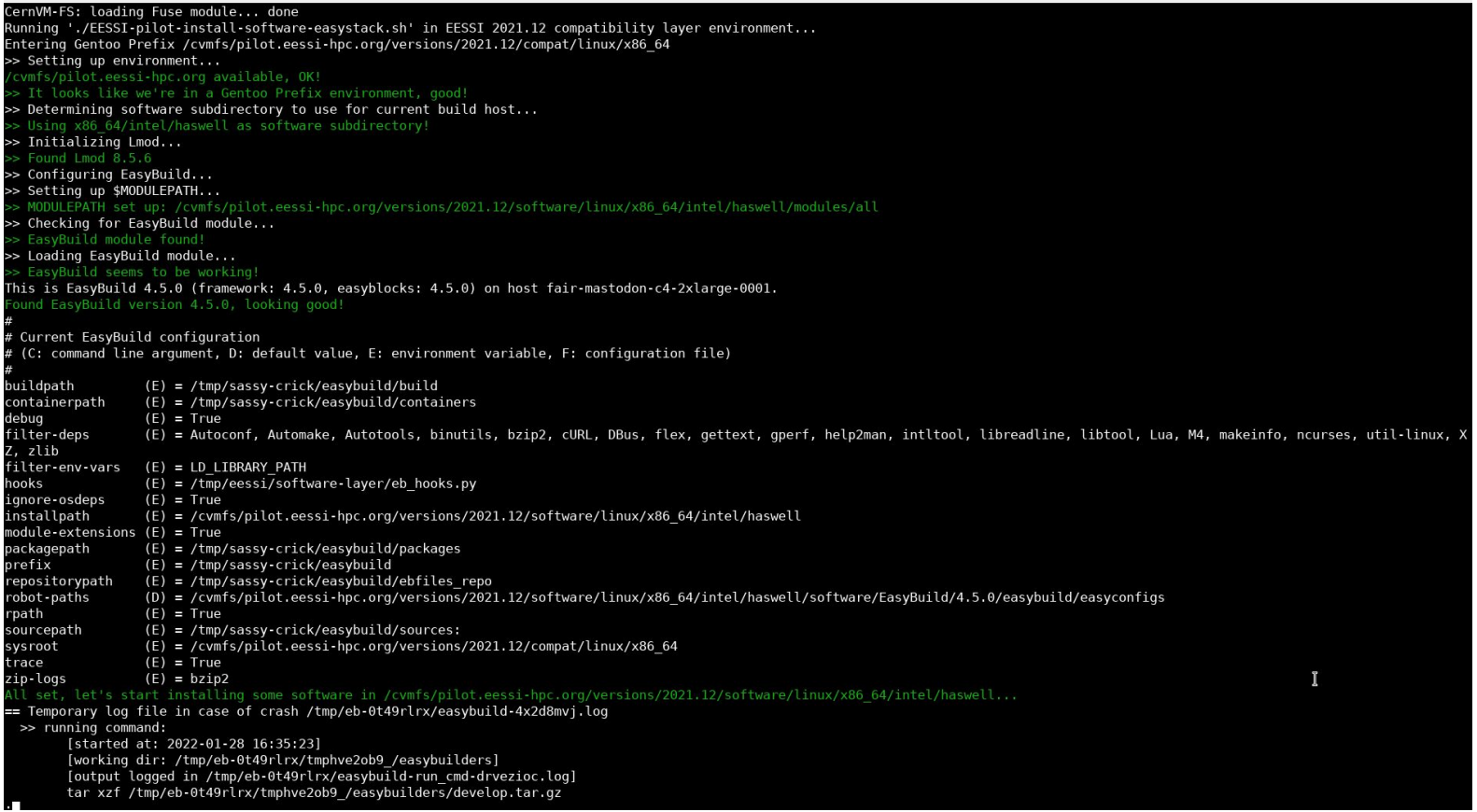


High-level overview of EESSI software bot





Real time demo of part of the work!



Current status

https://github.com/EESSI/eessi-bot-software-layer

- Initial proof-of-concept implementation of GitHub App working
- Developed recently during EESSI hackathons (Dec'21 + Jan'22)
- Limitations:
  - Only build phase
  - No feedback in PR yet
  - Only opening of PR is a trigger event
- Raw notes on what's left to do: https://hackmd.io/6V91CHRWRtuutANPaZRVPw
- **Contributions welcome!**

Use cases beyond EESSI

- Same concept can be used outside of EESSI project too!
- For HPC sites using EasyBuild
  - Manage software stack via PRs to easystack file in GitHub repository
  - Define tests to run to verify installations using ReFrame
  - Let bot automatically build + test installation before deploying
  - Only deploy if everything works out as expected
  - Avoid manually running EasyBuild!
- For EasyBuild maintainers
  - Testing contributions before merging pull requests
  - Only deploy corresponding installations on test cluster(s) when PR is merged