EESSI

EUROPEAN ENVIRONMENT FOR
SCIENTIFIC SOFTWARE INSTALLATIONS

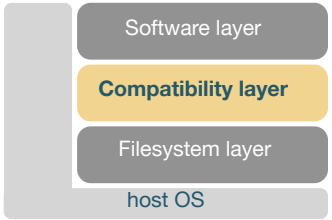
EESSI behind the scenes: compat layer

EESSI Community Meeting @ Amsterdam

15 Sept 2022

Kenneth Hoste (HPC-UGent) + Bob Dröge (Univ. of Groningen)

What is the compatibility layer?



- Operating system layer containing OS packages
  - No kernel, but everything from glibc to build tools, text editors and other libraries/tools
  - Makes the software independent of the client's OS, similar to what the OS in containers do
- Gentoo Prefix allows us to install this at a custom location (`/cvmfs/...`)
- Included in the EESSI CernVM-FS repository
  - One per CPU family + per EESSI version, e.g.:

    `/cvmfs/pilot.eessi-hpc.org/versions/2021.12/compat/linux/{aarch64,ppc64le,x86_64}`
  - Possibly a separate one for macOS in the future?

Which packages are installed in the compatibility layer?

- Can be a bit of a grey area, but in general it contains packages that:
  - Are not performance-critical, i.e. do not need to be optimized for specific CPUs
  - Don't need to have multiple versions installed
  - Are really required by Gentoo itself (e.g. Python, even though it is also in the software layer)
  - Are needed to actually offer the software stack in a user-friendly way, e.g. archspec and Lmod
- The fewer packages installed, the better (?)
- The layers should be as equal as possible for the different architectures
  - Not always possible due to packages that don't work on specific architectures, e.g. opa-psm2
- We define package sets that need to be installed on top of the default ones:
  - https://github.com/EESSI/gentoo-overlay/tree/main/etc/portage/sets

How do we build the compatibility layer?

- Ansible Playbook install.yml to automate the installation process
- Can be run from any computer that has Ansible installed
- Needs one build host (e.g. VM) for each architecture
- When executed, it will:
  - Log in to the build host, and do some checks (e.g. only EL8 is supported at the moment)
  - Install dependencies (git and Singularity)
  - Pull in a build container for bootstrapping Gentoo Prefix
    - This ensures that we always use the exact same and controlled environment
  - Add customizations that we need:
    - EESSI package overlay, package set, configuration, fixes, symlinks to host files, etc
  - Run a bunch of ReFrame tests to make sure everything is functioning properly
- Manually build a tarball of the installation, and ingest it to the repository

How do we build the compatibility layer?

- Ansible Playbook install.yml to automate the installation process
- Can be run from any computer that has Ansible installed
- Needs one build host (e.g. VM) for each architecture
- When e
  - Lo ment)
  - Ins
  - Pu
    -
  - Add customizations that we need:
    - EESSI package overlay, package set, configuration, fixes, symlinks to host files, etc
  - Run a bunch of ReFrame tests to make sure everything is functioning properly
- Manually build a tarball of the installation, and ingest it to the repository

Note that local sites or users never really need to do this, everything is included in the EESSI CernVM-FS repository

How do we update the compatibility layer?

- Why?
  - Install additional packages
  - Security vulnerabilities; these are being checked daily and reported to a private Slack channel
- Tests can be done with the build container and writable overlay: https://github.com/EESSI/software-layer/blob/main/build_container.sh
- Currently, we use a script that does the package updates, see for instance: https://github.com/EESSI/compatibility-layer/blob/main/scripts/update-pkgs-2021.12.sh
  - Could perhaps be converted to an Ansible playbook as well?
- A tarball is created of the updated compatibility layer, and ingested to the repo
  - The ingestion script first wipes the old layer and then untars the new one (faster than rsync)

Recent developments

- Atharva has worked on RISC-V support for Gentoo Prefix as a GSoC project
  - See presentation later today
- Bootstrapping Gentoo Prefix works now on a RISC-V system (native or via qemu)
- All packages that we need for EESSI have been "keyworded" for RISC-V
  - “Keywording” is Gentoo terminology for defining supported target infrastructures
- We can start building an EESSI compatibility layer for RISC-V…
  - Next pilot version?
  - We need a build container image for RISC-V first?

Future work

- Lack of interest for ppc64le, while it does usually take quite an effort to support it
  => drop support in favor of riscv64?
- Add some more packages (?)
  - p7zip for CUDA support script
  - htop
- Few additional fixes regarding broken/host symlinks
  - Take recommendations from Bart's presentation into account
- More active and automated follow-up of security updates
  - Not only Gentoo's GLSAs, which sometimes lag behind
  - Also consider other tools, like trivy ?
  - Automatically build and prepare updated compatibility layer, only approval to deploy by admin
  - Check impact on software layer (before deploying) by, for instance, running ReFrame test suite

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

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

YouTube channel (brand new!)

Monthly online meetings (first Thursday, 2pm CEST)