EESSI hackathon - show and tell

Dec 2021

https://github.com/EESSI/hackathons/tree/main/2021-12

Agenda

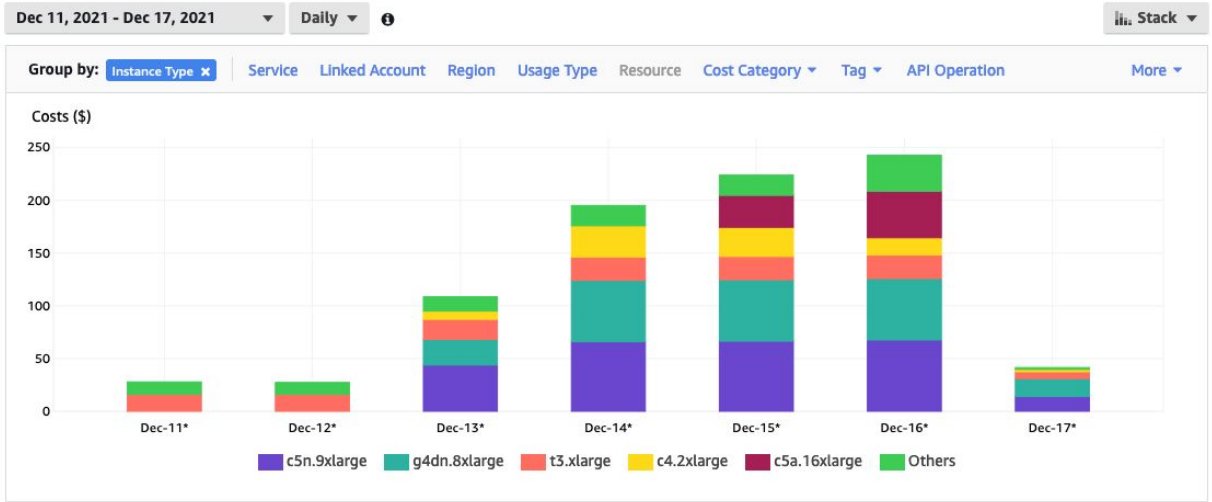
- General feedback
- Spent credits in AWS
- Task [02]: Installing software on top of EESSI
- Task [03]: Workflow to propose additions to EESSI software stack
- Task [05]: GPU support
- Task [06]: EESSI test suite
- Task [07]: Monitoring
- Task [08]: Setting up a (private) Stratum-1
- Task [16]: Export a version of the EESSI stack to a tarball and/or container image
- Task [XX] Azure support in CitC

General feedback

- What went well? What didn’t?
- What could be changed/improved for the next hackathon(s)?
- Organisation: HackMD notes, Zoom calls, Slack, GitHub hackathons repo, …
- Infrastructure: virtual clusters in AWS
- Allocating time for hackathon

Spent credits in AWS for Dec'21 hackathon

- (partial, credits consumed on Fri 17 Dec 2021 are incomplete)
- ~$700 worth of sponsored AWS credits spent on hackathon
  - Magic Castle: ~$215 on GPU node (g4dn), ~$250 on EFA nodes (c5n)
  - CitC: ~$80 on zen2 build node (c5a), ~$75 on other workernodes (c4, …)



Task [02]:
Installing software on top of EESSI

https://hackmd.io/EKBGawj_QM-TgG_BvPW1Ew

[02] Installing software on top of EESSI

1. Start user-facing docs to explain caveats with building on top of EESSI
    - WIP at https://hackmd.io/irkuPm4BSye6OL24wKmpgw
2. Building software on top of EESSI with EasyBuild works fine
    - Tested with WRF on top of EESSI 2021.06
    - Requires running in Prefix environment (startprefix)
    - Requires that EasyBuild is properly configured (RPATH, sysroot, filter-deps, …)
    - **Need to provide stand-alone script to correctly configure EasyBuild!**

[02] Installing software on top of EESSI

3. Manually building software on top of EESSI
    - Main caveat is RPATH linking to avoid using host libraries (only compat layer)
    - Can be made easier by including RPATH wrapper scripts with compilers in EESSI
    - Pull request for `GCCcore` easyblock:

      https://github.com/easybuilders/easybuild-easyblocks/pull/2638
        - Work-in-progress, but worked as expected in proof-of-concept setup
        - Some changes needed in EasyBuild framework for clean implementation: https://github.com/easybuilders/easybuild-framework/issues/3918
        - **Demo by Martin**

Task [03]:
Workflow to propose additions to EESSI software stack

https://hackmd.io/Z3OBwM3RQJKZVJdpaUaoEw

[03] Workflow to add software to EESSI

- All code for the Github App itself in in a separate repo:
  - https://github.com/EESSI/eessi-bot-software-layer
  - Basic structure available for an app that reacts to GitHub events
  - Currently only does something for “pull request opened” events
    - Pull in easystack file, prepare job directory, submit build script to Slurm
  - Logging method for dumping real events into json files, so they can be replayed
  - Authenticating as an App against the Github API
  - Run in different modes: web app, cron mode, input file

[03] Workflow to add software to EESSI

- Jörg focused on the software build Bash scripts
  - https://github.com/EESSI/hackathons/tree/03_workflow/2021-12/03_workflow
  - Submit job for a prepared job directory containing an easystack file
  - Build all the software for a (for now) hardcoded architecture
  - If the build succeeds, (re)run the sanity checks in a container with another distro
  - Provide log of build and test
  - Make tarball + checksum file

[03] Workflow to add software to EESSI

- Still needs quite a lot of work:
  - Use the EESSI build container and settings (paths, Easybuild config)
  - Monitor build/test jobs + handle failures
  - Reply results back to PR (comment on success vs fail, logs via gist, …)
  - Pick up logs and tarballs
  - More architectures
  - Event dependencies
  - Handle many more events

Task [05]:
GPU support

https://hackmd.io/BIYsQIrWRheSILpyt6m8Mg

[05] GPU support

- Hackathon impressions
  - Magic Castle environment worked pretty well
    - Would need to also test on other OSes/setups
  - Starting from scratch with varying schedules is not easy
    - Tweaking a working setup should be better
- Achievements
  - GPU support is working and initial script(s) made: https://github.com/EESSI/hackathons/tree/05_gpu/2021-12/05_gpu
- Issues
  - Final script will need a lot of tests (checks for drivers, space, location,...)
  - Need to be able to unpack `.dep` and `.rpm` with tools only from compat layer
    - Ok for deb (`ar` + `tar`), need `rpm.eclass` for RPMs

[05] GPU support

- **DEMO!**
- What still has to be done
  - Create EasBuild hook to add Lmod tag to CUDA and CUDA-enabled modules
  - Create Lmod hook to hide tagged modules unless some condition is met
    - Existence of `$EBROOTCUDA/bin`?
    - This will need to independently handle different EESSI versions
  - Need a symlink in CVMFS
    - Software installation path of CUDA with `versions` -> `host_injections`
    - Module should fail to load if `$EBROOTCUDA` does not exist (and perhaps another check for `nvidia-smi`?)

Task [06]:
EESSI test suite

https://hackmd.io/M6Dbslw8QvWFA0QEF9sIeA

[06] EESSI test suite

- Hackathon impressions
  - Availability of test cluster well organized - kudo's!
  - Small teams is good
  - Writing tests still pretty individual task - not a ton of interaction
- Achievements
  - Created list of required/desired tests for compat & SW layers
  - Agreed on test labels for test selection
  - GROMACS: test on top of CSCS GROMACS testlib (software-layer PR #156)
  - WRF: initial version in *06_test_suite* branch of hackathon repo
  - Task members gained experience with ReFrame

[06] EESSI test suite

- Issues
  - Issues with GROMACS test due to SELinux on Magic Castle
  - TensorFlow CSCS uses default Horovod example
    (no proper support for CPUs, can't run pure TensorFlow)
  - TensorFlow EESSI *should* support CPUs, but fails to use multiple threads on MC
- TODO
  - Actually integrate in CI pipeline
  - Decide on resourcedir setup to handle large inputs (WRF)
  - Finish WRF and create PR
  - Document how to run test suite for CI/monitoring
  - @Other tasks: Implemented new feature? Should come with test :) (e.g. GPU support)
  - Many more tests…

[06] EESSI test suite

- **Demo!**

Task [07]: Monitoring

https://hackmd.io/CBX5QM2dStmeQqRvCz35CQ

[07] Monitoring

- wtb time, pst.
- Got monitoring up and running decently well
  - Every stratum server can now get a local stack: Grafana, Prometheus, node_exporter, and with preliminary cvmfs support
  - Not integrated CVMFS on the node due to logistics (see issues)
- Hackathon model worked really well for setting aside time

[07] Monitoring : Issues

- The ansible set up from EESSI is currently repo based and not role based
  - Integration with existing setups is hard (shared inventory? Host classes?)
  - Fetching upstream fixes is annoying (git merge, not just update role version)
  - Solution? https://github.com/terjekv/ansible-eessi-roles (collection?)
- Managed to find a broken fork…
  - `objc[17187]: +[__NSCFConstantString initialize]`
  - “Solved” via `export OBJC_DISABLE_INITIALIZE_FORK_SAFETY=YES`
- What do we do with private stratum1s that don’t have internet access?
- The CVMFS monitoring dashboard probably needs a friendly paw

[07] Monitoring : Lessons

- Reuse code. Cloud Alchemy provides a ton of ansible roles, I used their setup as-is for grafana, prometheus, and node_exporter
- Actually works with SElinux and port management, and more!
- Created a role for cvmfs monitoring based on the same skeleton
- Works really well, local usage becomes very clean.
- **Consume, don’t clone!**

[07] Monitoring : Demo

- A brief walk through the Ansible setup
- No-one (sane) wants to see Ansible run?

[07] Monitoring : TODO

- Finalize CVMFS monitoring
  - Should involve role-based install of filesystem-layer for any node
- Decide on naming scheme for Ansible repos and the roles shipped
- Should probably migrate to the Ansible role repo to a collection
- Move repos to EESSI

Task [08]:
Setting up a (private) Stratum-1

https://hackmd.io/IJVKEGAOS7aolbgDZ-D_1Q

[08] Setting up a private Stratum 1

- Went through the current instructions on how to set up a private Stratum 1
  - Fixed some outdated information/instructions
  - Added instructions on how to put the snapshot in non-default location
- Tested how to connect to a private Stratum1 from a client
- Learned a few commands for how to test both client and Stratum 1

  (see hackathon task notes @ https://hackmd.io/lJVKEGAOS7aolbgDZ-D_1Q)
- Updated documentation: https://eessi.github.io/docs/filesystem_layer/stratum1
- Follow-up PR: https://github.com/EESSI/docs/pull/85

Task [16]:
Export a version of the EESSI stack to a tarball and/or container image

https://hackmd.io/EKBGawj_QM-TgG_BvPW1Ew

[16] Exporting EESSI to a tarball/container

- Available I/O performance caused long trial/error cycles
- Variant symlinks not yet in place, unable to test our primary objective
- We narrowed our ambitions and came up with a script to containerize just a subset of modules (demo would still run for about ~1 hour)
- Would help to get some `sudo` and `singularity` available on CitC setup
- Current result shows signs of life (`gcc --version` works, gromacs `gmx` prints out help, R can run a simple benchmark)
- Tarball sizes: foss 1.5GB, Gromacs 1.8GB, R-bioconductor 8.2GB
- Would be nice to integrate whatever ReFrame team has come up with
- One idea that popped up at our place is assigning DOI to resulting images

Task [XX]:
Azure support in CitC

(work-in-progress by Hugo & Matt)

EESSI hackathon (Jan’22)

17-21 Jan 2022

https://github.com/EESSI/meetings/wiki/EESSI-hackathon-Jan'22

https://github.com/EESSI/hackathons