Containers for CORSIKA 8 – for testing and distribution – … and related challenges

Lukas Nellen
ICN-UNAM
lukas@nucleares.unam.mx

Types and uses of containers

- Technologies
  - **docker**: *isolate application*
  - podman, buildah, skopeo: *isolate application*
  - linux containers (LXD, LXC)
  - apptainer (singularity): *High Performance Computing*

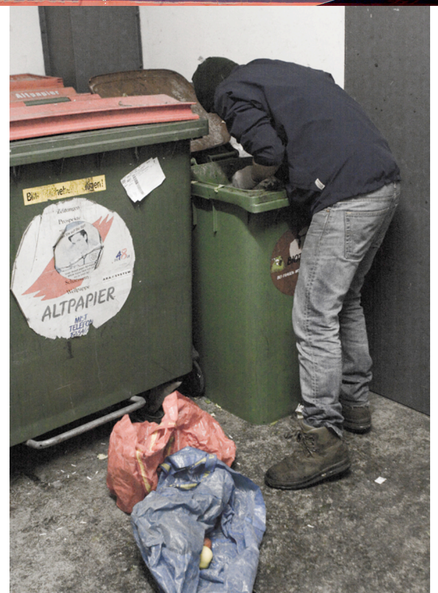
- Use cases
  - (Integration) testing: *light-weight VM*
  - Software distribution:
    new version of statically linked application

- Don’t forget Virtual Machines

Types and uses of containers

de-facto reference
Also provides desktop application

- Technologies
  - **docker**: *isolate application*
  - podman, buildah, skopeo: *isolate application*
  - linux containers (LXD, LXC)
  - apptainer (singularity): *High Performance Computing*
- Use cases
  - (Integration) testing: *light-weight VM*
  - Software distribution:
    new version of statically linked application
- Don’t forget Virtual Machines

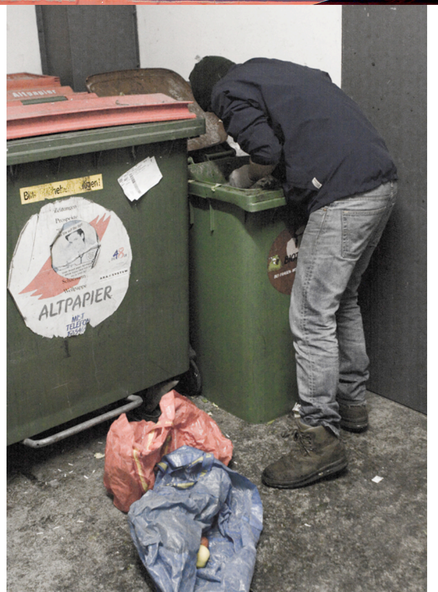


Types and uses of containers

- Technologies
  - **docker**: *isolate application*
  - podman, buildah, skopeo: *isolate application*
  - linux containers (LXD, LXC)
  - apptainer (singularity): *High Performance Computing*

By default not full isolation,
more access to host file system

- Use cases
  - (Integration) testing: *light-weight VM*
  - Software distribution:
    new version of statically linked application



- Don’t forget Virtual Machines

Common tasks

- Have to be done setting up
  - Containers
  - Virtual machines
  - Bare iron (physical) machines
- Have to be done for all use cases

- Distribution specific: install packages
  - Build tools
  - Libraries and headers (some)

- Generic
  - Build support packages from source
  - Conan cache and package build

Common task solutions

- Use universal solutions
  - Shell scripts
  - Configuration management tools
    - ansible and others
  - Can be used for different types of containers and for (virtual) machines

- A low-level Dockerfile is not portable
  - Can copy and run scripts

- My solution:
  - OCI images
    - Can be used, converted, or imported
  - Create using **buildah** or singularity
    - Can handle scripts or ansible ← Bulk of building
    - Can access containers as sandboxes ← Access to host credentials, e.g. git access

Container tools

- Have corsika-docker repository
  - Needs updating
  - includes scripts for on-instruction build of containers
- Evolve tools from Dockerfiles to buildah
  - same tooling used for auger, HAWC, SWGO
  - Evaluate: scripts or ansible
- Expand range of images produced and uploaded
- Rework to include pre/post conan images
  - Simplifies cache updates

Dockerfile risks

- It is possible to leak confidential information
- Image contains
  - Layer per instruction
  - Metadata: commands from Dockerfile
  - Environment

- Erased files still in lower layers
- All information (could be a password!) in commands can be read

- Requires care and understanding to, e.g., import private information
  - git repo
  - Files from protected web site

- Easier in buildah: mount file system and act from build host

rootfull containers

- default setup of docker: daemon **running as root**
  - Access to docker equivalent to **access to root account**
  - The container itself remains isolated
    The user can mount restricted files for access from within container

- Acceptable for single user machines
- Docker desktop for MAC, Windows: runs in VM

- Scheduler (e.g. kubernetes) might restrict access to system
  - Requires configuration by admin

- Use case: **test environments for CI**
  - Trust startup (user!), runtime, and container image

rootless containers

- Rootless containers avoid problems with root access
  - singularity designed to run applications on supercomputers
  - podman/buildah suite designed to run rootless out of the box
  - Can be used on shared equipment
- Normally: networking limited, less efficient
- Convenient for development
- Use cases:
  - pre-packaged build environment
  - software distribution and installation

Containers for CI

- Gitlab runner uses docker images
  - In the process of moving to buildah for setup
- Need more distributions besides ubuntu
  - CentOS / y RHEL testing needed: production clusters
  - Especially CentOS 7: current grid production platform for many
- Need automated re-build when conan specifications change
  - Cache speeds up builds, especially when building from source
  - Need control of the number of threads used by conan
    - Uses all hardware cores by default, even when resource restrictions in place
    - Unreasonably high load on server
- Related to validation / verification (after lunch)

Distribution / deployment

- Source code
- Container images
- python
  - pip, conda, mamba
  - fermitools and other gamma ray astronomy tools provided this way
  - Many standard packages available
- Conan
  - Can provide executables, not just libraries
- Distribution native
  - distribution formats: rpm, deb, …
  - would need to provide at least two or three formats
  - learning curve

The problem of old distributions

- RHEL/CentOS 7: 10 years
  - 5 years from RHEL/CentOS 8 on
- New: 5 + 5 year LTS for ubuntu
- Old kernel: can limit versions of libc
  - New libc might depend on new kernel calls
  - Containers based on new distribution can fail
- Old distributions provide old compilers
  - Bad for optimization and use of new C++ features
  - New compilers sometimes packaged (RHEL software collection)
- Production sites (GRID!) delay upgrades
  - CORSIKA 8 has to run on RHEL and Ubuntu LTS

- No good, universal solution available
  - Need to validate on production platforms

Distribution channels

- Use gitlab for sources

- Open registries for containers
  - docker hub, gcr, quay?
  - No need to use gitlab or other, private registries
  - Do we need special apptainer (singularity) images?

Definitely, most work already needed for CI

- Python repositories
  - pypi, conda forge
  - Evaluate as means for distribution
- Conan

- VM image(s)?

- We should not do all, decide before public beta

Production sites: CVMFS

- Cern VM File System
- Global, unique paths
- Available on GRID production sites
  - and some computing centers
- Installation easy, from packages

- Read only: cannot generate tables at startup, do at install

- CVMFS includes optimization for layered container images

- Probably not relevante before first, full release

Using CVMFS

- Should we request corsika namespace?
  - Alternative: leave installation to user communities
  - E.g., auger manges CORSIKA 7 in auger namespace
- Could provide standard CORSIKA application(s) in container
- Also possible: Python installation and virtual environment
  - Need to make sure libraries are found
- Could probably come up with a solution for conan
  - Not really a priority, though

Containers as build enviroments

- Already done for CI
- Could make one or a few available for developers
  - but nor really needed for CORSIKA 8, not complicated dependencies

- Not a priority

Conclusion

- Rework image creating tooling
- Expand number of images for CI
- Preferences
  - OCI images
  - Use buildah, not Dockerfiles
  - Also for distribution of CORSIKA 8
    - Might also ship apptainer images
- We should evaluate other packaging tools
  - Use of python distribution infrastructures is popular
  - Versioning of all packages not always simple