Containers

Vienna workshop on simulations 2024
22nd April

Carlo Mancini Terracciano
carlo.mancini-terracciano@uniroma1.it

INFN

SAPIENZA
UNIVERSITÀ DI ROMA

Why?

- Imagine you develop an application in C++ with some external dependency
- E.g.: the Geant4 example extended/medical/DICOM
    - It uses DCMTK to load DICOM files
    - And it needs that DCMTK has been compiled with the flag CMAKE_POSITION_INDEPENDENT_CODE=ON

Virtualisation!

- If you create a virtual machine with Geant4 and DCMTK compiled in the way needed by the DICOM example it would work everywhere
- It’s heavy and slow!
- Containers overcome all the shortcomings of Virtual Machines

Virtualisation!

- Containers don’t require the installation of a separate guest operating system.
- They directly run and use the host operating system
- Containers only need the dependent file system and binaries for their functioning
- lightweight than Virtual Machines

What is a container?

- Containers (such as Docker) are
    - a standard for cloud computing and clusters
    - a good way to run an application in the same environment on different machines
    - a fast way to distribute code for multiple architectures
    - integrated in all the CI/CD platforms
    - light and efficient

Run a Docker container

- docker pull hello-world
- docker run hello-world
- The image is pulled from https://hub.docker.com/

```
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (arm64v8)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Develop a Docker container

- Write a “Dockerfile” file
- To build the image:
  docker build -t <tag> \
  -f <dockerfile> <path>
- To run it:
  docker run <tag> <command>
- Have a look of:
  https://github.com/carlomt/docker-dicom-g4example

```
FROM almalinux:9.3


RUN dnf install -y epel-release
RUN dnf --enablerepo=crb install -y \
    gcc \
    g++ \
    cmake \
    xerces-c xerces-c-devel \
    expat expat-devel  \
    ninja-build && \
    dnf clean all
```

Volumes

- Containers are ephemeral, to have an output you have to bind a volume
- -v (or —volume)
- <host path>:<container path>:<options>

GUI

- To forward X11
- Linux (xhost local:root)
  -e DISPLAY=$DISPLAY --volume /tmp/.X11-unix:/tmp/.X11-unix
- Mac (once XQuartz is installed and xhost +localhost)
  -e DISPLAY=docker.for.mac.host.internal:0
- Windows (once XMing is installed)
  -e DISPLAY=docker.host.internal:0

Geant4 Docker container

- Getting inspiration from the work done by A. Dotti and W. Takase
- I developed a Geant4 container for x86 and ARM
- https://hub.docker.com/r/carlomt/geant4
  https://github.com/carlomt/docker-geant4
- Once Docker is installed, you can run with:
  `docker run carlomt/geant4:<G4-VERSION>`
- To keep the size of the Docker images limited, datasets are not installed. It's possible to map a folder in the host with the option:
  `–volume="<GEANT4_DATASETS_PATH>:/opt/geant4/data:ro"`
- The version `carlomt/geant4:<G4-VERSION>-dcmtk` includes the library to read DICOM
- You can build a Docker container for your application on top of these images, example: https://github.com/carlomt/docker-dicom-g4example

Docker Compose

- A tool for defining and running multi-container Docker applications
- A YAML file to configure your application
  You can see it as a makefile for Docker
- Have a look of
  https://github.com/carlomt/docker-geant4course/blob/main/docker-compose.yml

Security issue and Apptainer

- Depending on how the Docker demon is installed, you could be root of the container (and if the host is Linux on the volumes mounted)

- Apptainer (formerly Singularity)
is a container system (compatible
with Docker images) which doesn't have this security issue

- https://apptainer.org/

- Largely available on scientific computing clusters
eg: https://confluence.infn.it/display/TD/Singularity+in+batch+jobs