docker & HEP:
containerization of applications for development, distribution and preservation

Sébastien Binet

LAL/IN2P3

2015-04-13

Docker: what is it ?

- http://www.docker.io/
- an open source project to pack, ship and run any application as a lightweight container

High level description

- kind of like a lightweight VM
- runs in its own process space
- has its own network interface
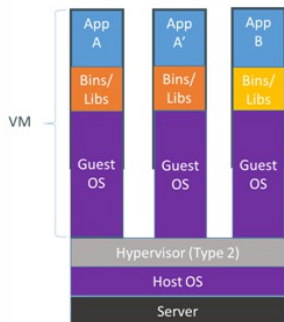- can run stuff as root

Low level description

- chroot on steroids
- container = isolated process(es)
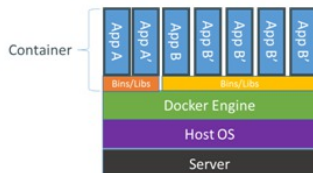- share kernel with host
- no device emulation

Docker: why ?

- same use cases than for VMs
- speed: boots in (milli)seconds
- footprint: 100-1000 containers on a single machine/laptop. small disk requirements

Docker: why ?

Efficiency: *almost* no overhead
- processes are isolated but run straight on the host
- `CPU` performance = native performance
- memory performance = a few % shaved off for (optional) accounting
- network performace = small overhead

Efficiency: storage friendly
- unioning filesystems
- snapshotting filesystems
- copy-on-write

- provisionning takes a few milliseconds
- . . . and a few kilobytes
- creating a new container/base-image takes a few seconds

Docker: introduction

Hello World

- get a base container (ubuntu, centos, ...)
  ```
  $ docker pull ubuntu
  ```
- list images already pulled in:
  ```
  $ docker images
  ```

  ```
  ubuntu        12.04        8dbd9e392a96      5 months ago      131.5 MB (virtual 131.5 MB)
  ubuntu        latest       8dbd9e392a96      5 months ago      131.5 MB (virtual 131.5 MB)
  ubuntu        precise      8dbd9e392a96      5 months ago      131.5 MB (virtual 131.5 MB)
  ubuntu        12.10        b750fe79269d      6 months ago      24.65 kB (virtual 180.1 MB)
  ubuntu        quantal      b750fe79269d      6 months ago      24.65 kB (virtual 180.1 MB)
  ```

- run an executable inside a container
  ```
  $ docker run ubuntu:12.10 echo ``hello world''
  ```

Docker: introduction - II

Detached mode

- run a container in detached mode:
  ```
  $ docker run -d ubuntu sh -c \
    while true; do echo ''hello''; sleep 1; done;
  ```
- get the container id:
  ```
  $ docker ps
  ```

ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
78c88e279f26	ubuntu:12.04	/bin/sh -c while tru	14 seconds ago	Up 11 seconds	

- attach to the container
  ```
  $ docker attach 78c88e279f26
  ```
- start/stop/restart a container
  ```
  $ docker stop 78c88e279f26
  ```

Docker: public index

Public index

- pull an `apache` container from the index:
  ```
  $ docker search apache
  $ docker pull creack/apache2
  ```
- run the image and check the ports
  ```
  $ docker run -d creack/apache2
  $ docker ps
  ```

ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
369602483ae9	creack/apache2:latest	/usr/sbin/apache2ctl	4 seconds ago	Up 1 seconds	49153->80, 49154->443

Also available from the browser:
https://index.docker.io/

Docker: creating a customized container

- run `docker` interactively:
  ```
  $ docker run -i -t ubuntu bash
  root@bf72b1a06e6c:/# apt-get update
  Reading package lists... Done

  root@bf72b1a06e6c:/# apt-get install memcached
  [...]
  root@bf72b1a06e6c:/# exit
  ```
- commit the resulting container
  ```
  $ docker commit `docker ps -q -l ` binet/memcached
  ab59e4b14266
  ```
- run the image
  ```
  $ docker run -d -p 11211 -u daemon binet/memcached memcached
  ab59e4b14266
  ```

Docker: creating a customized container - Dockerfile

```
## install gaudi from RPMs
FROM hepsw/slc-base
MAINTAINER binet@cern.ch

ENV MYSITEROOT /opt/lhcb-sw
ENV CMTCONFIG x86_64-slc6-gcc48-opt

RUN mkdir -p $MYSITEROOT

## install some system dependencies
RUN yum install -y bzip2 freetype glibc-headers tar which

## retrieve install
RUN curl -O -L  http://cern.ch/lhcbproject/dist/rpm/lbpkr && \
    chmod +x ./lbpkr

## install (source+binaries)
RUN ./lbpkr install-project GAUDI v26r1
```

Docker: Dockerfile - II

- build the container
  ```
  $ docker build --tag=hepsw/lhcb-gaudi:v26r1 .
  $ docker tag hepsw/lhcb-gaudi:v26r1 hepsw/lhcb-gaudi:latest
  ```

- run the container (and test the build)
  ```
  $ docker run -i -t hepsw/lhcb-gaudi /bin/bash
  [hepsw/lhcb-gaudi] $ cd /scratch
  [hepsw/lhcb-gaudi] $ gaudirun.py \
      $GAUDIEXAMPLESROOT/options/TupleEx.py
  ```
- bind mounts
  ```
  $ docker run -i -t hepsw/lhcb-gaudi \
    -v /host/build/results:/scratch
    /bin/bash
  ```
- copy files from container to host
  ```
  $ docker cp hepsw/lhcb-gaudi:/scratch /host/build/results
  ```

- `kvm-and-docker-lxc-benchmarking`
  - executive summary: `docker` delivers very close to the bare-metal performances (consistantly better than `KVM` save for some `mysql` tests)
- tests `docker` containers creation, guest CPU/Mem/IO performances, ...

Benchmarks - II

Disk sizes of containers:

```
REPOSITORY              TAG                 VIRTUAL SIZE
hepsw/lhcb-base         20150331            336.6 MB
hepsw/lhcb-gaudi        v26r1               3.911 GB
hepsw/lhcb-davinci      v36r5               7.790 GB

lhcb-base   (slimmed)   latest              322.3 MB
lhcb-gaudi  (slimmed)   latest              3.893 GB
lhcb-davinci (slimmed)  latest              7.771 GB

hepsw/cvmfs-base        20150331            629.4 MB
hepsw/cvmfs-lhcb        20150331            629.4 MB
```

Disk sizes of $MYSITEROOT:

```
hepsw/lhcb-base:    /opt/lhcb-sw    67M
hepsw/lhcb-gaudi:   /opt/lhcb-sw 3.600G
hepsw/lhcb-davinci: /opt/lhcb-sw 7.300G
```

slimmed: use docker export+import to shrink image size.

Benchmarks - III

Running `gaudirun.py GaudiExamples/TupleEx.py`

- AFS

  ```
  56.87s user 14.26s system 66% cpu 1:46.50 total # kick AFS
  57.62s user 13.07s system 99% cpu 1:11.17 total
  57.69s user 13.46s system 99% cpu 1:11.58 total
  57.93s user 13.26s system 99% cpu 1:11.66 total
  ```

- Docker-RPMs

  ```
  55.93s user 12.34s system 98% cpu 1:09.54 total
  55.43s user 12.88s system 98% cpu 1:09.12 total
  55.54s user 12.16s system 98% cpu 1:08.83 total
  55.39s user 11.60s system 98% cpu 1:07.81 total
  ```

- Docker-CVMFs (a docker container where CVMFs is configured and running)

  ```
  55.53s user 14.01s system 88% cpu 1:18.75 total # kick CVMFs
  54.95s user 12.83s system 97% cpu 1:09.36 total
  55.42s user 12.86s system 98% cpu 1:09.35 total
  55.42s user 13.01s system 98% cpu 1:09.63 total
  ```

- no container backend for MacOSX (yet?)
- it is foreseen that at some point a `jail`-based backend will appear
- in the meantime: `boot2docker`
  - launches a very thin Linux-VM where the `docker` daemon is installed
  - installs the `docker` client on the host
  - talks via HTTP/REST to the daemon

- `boot2docker` works also for `Windows (TM)`

Conclusions & Prospects

- easily distribute dev-environments
- easily provision build and dev-environments
- easily relocate binaries (remember: `chroot` on steroids!)
- provision efficient performance-wise environments (production)

- run a `HEP`-dedicated `docker` images repository ?
  - *ACL*s
  - *O*(*GB*) images ...
- put a `Frontier` server in front ?

Docker: HEP examples

```
hepsw/docks (github)
hepsw/containers (docker public registry)
```