Docker @ BioHPC Lab

BioHPC Lab Open House

Jaroslaw Pillardy

Bioinformatics Facility
Institute of Biotechnology
Cornell University

http://cbsu.tc.cornell.edu/lab/lab.aspx

What is Docker?

Docker is a Linux subsystem allowing users to run applications in a way that is isolated from the host operating system.

Docker is an OS-level virtualization engine – it shares lots of components with host operating system and has direct access to host machine resources via host OS kernel.

There are no I/O or CPU penalties typically associated with regular Virtual Machines.

What is Docker?

- Docker applications are instantiated from templates called **images**.
- Running Docker instances created from images are called **containers**.
- PRO: Docker container is an OS-level virtualization, which is much faster and not imposing additional performance penalties compared to regular virtual machines.
- PRO: Many software packages are already available as Docker images – no installation necessary
- CON: Docker cannot run non-Linux operating system (like Windows).

Why is Docker important?

Users can run applications native to other Linux distributions without porting or modifying them – for example Ubuntu pipelines on CentOS/RedHat.

Users can install applications that require administrator (“root”) access inside containers.

Images and containers can have complicated sets of software dependencies installed without affecting host machine at all. Users or admins can start with a fresh version of Linux and install applications from scratch.

Docker @ BioHPC Lab

Regular Docker cannot be run by users without compromising system security – full Docker access is same as “root” access.

Docker @ BioHPC Lab is a modified version with some minor restrictions imposed making it safe in a multiuser environment.

Docker @ BioHPC Lab

Regular Docker interface command **docker** has been replaced with **docker1**

```
$ docker ps -a
Cannot connect to the Docker daemon. Is the docker daemon running on this host?
```

Local directory **/workdir/labid** is mounted inside Docker container as **/workdir** allowing for easy interfacing with storage and data exchange.

Running Docker Applications

Single command

After the command is done the container stops and cannot be rerun.

```
$ docker1 run biohpc/cowsay cowsay 'Hi there!'
```

```
 ____________
< Hi there!>
------------
     \   ^__^
      \  (oo)\_______
         (__)\       )\/\
             ||----w |
             ||     ||
```

Running Docker Applications

We have prepared a set of images that can be used as a starting point. Default images from Docker repository are usually "light", i.e. without most of the software and libraries installed. Our images are configured for easy development with compilers, libraries and typical programs installed.

Description	Repository image	File
CentOS 7 cowsay Basic image for testing with two extra commands installed: fortune and cowsay.	biohpc/cowsay	/programs/docker/images/cowsay.tar
CentOS 7 development CentOS 7 image with development tools and libraries installed (compilers, Java, Perl, Python etc)	biohpc/centos7dev	/programs/docker/images/centos7dev.tar
Ubuntu development Ubuntu image with development tools and libraries installed (compilers, Java, Perl, Python etc).	biohpc/ubuntudev	/programs/docker/images/ubuntudev.tar
CentOS 7 development with GUI and sshd CentOS 7 image with a standard set of development tools and libraries built on centos7dev image. Includes X11 and GUI tools and libraries. Automatically starts sshd, it must be run in background and connected to with ssh.	biohpc/centos7devgui	/programs/docker/images/centos7devgui.tar

Running Docker Applications

Interactive mode

This command allows user to enter container and run commands interactively inside, once shell is exited the container permanently stops.

```
$ docker1 run -it biohpc_jarekp/cowsay /bin/bash

[root@a605b04a7ca5 workdir]#
```

Running Docker Applications

Background mode

Start the container in the background, commands may be executed in the container using container ID, including shell. Exiting shell does not stop the container.

```
$ docker1 run -d -t biohpc/cowsay /bin/bash
5ab4520a337fbb01b2b3f45c14688e09544623793065a7238c91c8864
$ docker1 ps -a
CONTAINER ID  IMAGE          COMMAND      CREATED        STATUS
5ab4520a337f  biohpc/cowsay  "/bin/bash"  6 seconds ago  Up 4 seconds
$ docker1 exec 5ab4520a337f /bin/bash -c "fortune | cowsay"
               ___________________________________
               < Beware of low-flying butterflies. >
               -----------------------------------
                       \   ^__^
                        \  (oo)\_______
                           (__)\       )\/\
                               ||----w |
                               ||     ||
```

Running Docker Applications

Ports and servers

Sometimes software needs to run as a service, e.g. in a website or as an ssh server.

You will need to find which port it is using and map it with

```
-p 127.0.0.1:NNN:MMM
```

NNN is a port number on your machine (e.g. 5000)
MMM is port your software uses (80 or 8080 often used for websites, 22 for ssh).

Running Docker Applications

Example: running an ssh server and GUI applications

```
$ docker1 pull biohpc/centos7devgui
```

```
$ docker1 run -d -p 127.0.0.1:5000:22 -P -t   \
    biohpc/centos7devgui /usr/sbin/sshd -D
```

Now you can connect to the container with ssh: port 22 in the container (ssh) is mapped to port 5000 on local machine (localhost, 127.0.0.1)

```
$ ssh -X root@localhost -p 5000
```

```
$ firefox http://localhost:8080/mysapp
```

Example of Docker Workflow

Many programs are already available as Docker images

```
$ docker1 pull biodckrdev/bowtie:1.1.2
Trying to pull repository dtr.cucloud.net/biodckrdev/bowtie ...
Trying to pull repository docker.io/biodckrdev/bowtie ...
sha256:0a070c1eb568f663ecbd5dc7585: Pulling from docker.io/biodckrdev/bowtie
952132ac251a: Pull complete
Digest: sha256:0a070c1e399bd927c7585
Status: Downloaded

$ docker1 run -it biodckrdev/bowtie:1.1.2 /bin/bash

biodocker@466562cf2d84:/workdir$ bowtie
No index, query, or output file specified!
Usage:
bowtie [options]* <ebwt> {-1 <m1> -2 <m2> | --12 <r> | <s>} […]
```

Typical Docker Workflow: New Software

First, pull appropriate Docker image from repository or import from a file

```
docker1 pull biohpc/ubuntudev
```

Start Docker container in the background.

```
docker1 run -d -t biohpc/ubuntudev /bin/bash
```

Find ID of the container

```
docker1 ps -a
```

Start an interactive shell inside a container.

```
docker1 exec -it d3b7d7463857 /bin/bash
[root@ d3b7d7463857 workdir]#
```

Typical Docker Workflow: New Software

Install software you need. Follow instructions from software website.

```
[root@ d3b7d7463857 workdir]# apt-get myapplication
```

Export the container to a file in a case you will need it later. You won't need to install software again.

```
[root@ d3b7d7463857 workdir]# exit
docker1 export -o /home/labid/myfile.tar d3b7d7463857
```

Run your software in Docker. Remember that local /workdir/labid is mapped to /workdir inside the container.

```
cp /home/labid/mydata.vcf /workdir/labid
docker1 exec -it d3b7d7463857 /bin/bash
[root@ d3b7d7463857 workdir]# myapplicaton -i /workdir/mydata.vcf
```

(Optional) Save the container again if you made any changes to software installed.

Stop and remove the container

```
docker1 stop d3b7d7463857
docker1 rm d3b7d7463857
```

Tips on Docker @ BioHPC

- Containers and images will be removed from the Lab machine after your reservation ends. Make sure you have exported your container for later reuse if you want to keep it!
- Do a search for Dockerized version of the software you want to use. There is a good chance it is already out there and all you need to do is pull.
- While working with Docker a list of executed and stopped containers grows very fast, you can use our add-on command "`docker1 clean`" to remove stopped containers (or "`docker1 clean all`" to remove all and start over).
- Files produced in the Docker container usually belong to "root", even if they are in /workdir, which may make it difficult to access or remove them. You can reclaim these files with "`docker1 claim`" on the local machine