Docker Workshop
Solutions for the exercises

Ignacio Vidal - Abbas Sarvmeily

April 18, 2016

Exercise 1 Running a custom container

- `docker run -t -i --name computation feeg6003/scicomp:v1` creates and run a containerized `bash` session of the `feeg6003/scicomp:v1` docker image with the name *computation*.
- You can clearly see that `gcc` and `python` are not available in the host machine, whereas they are accessible within the contanerized bash session.
- The kernel version can be checked with `uname -r`. You can see that both kernel versions inside and outside the container are the same
- `stress -c 4` will indeed show that all the cores are being used, regardless if it is called outside the container or within the container.
- `docker stop computation` will stop the container.
- `docker start computation`, followed by `docker attach computation` will allow you to access again to the containerized `bash`.

Exercise 2 Data persistence

Proceed as above, but using two separate containers:

- `docker run -t -i --name container1 feeg6003/scicomp:v1`
- In the container, run `echo "Your best joke" > joke.txt`
- Install `cowsay` with `apt-get install cowsay` and execute `cat joke.txt|cowsay`.
- In another terminal window, run `docker run -t -i --name container2 feeg6003/scicomp:v1`
- Using `ls`, check that there is no `joke.txt` file, because you are in another different container. Also, there is no `cowsay` command.

Exercise 3 Acessing host data from a container

`docker run --name data1 -t -i -v /home/feeg6003-docker/data:/data feeg6003/scicomp:v1` will run the first container.

In another terminal window, `docker run --name data2 -t -i -v /home/feeg6003-docker/data:/data feeg6003/scicomp:v1` will run another different container sharing the same `/data` directory.

From the second container, edit the `README.md` file. Check that the changes made in one container are reflected in the other.

Exercise 4 Recreate the `feeg6003/scicomp` image using commits

Run the `ubuntu:14.04` image and install as many software packages as you want. Once finished, stop the container and commit the changes using

```
docker commit -m "Commit message" -a "Author name" containerID user/newimagename:tag
```

After that, check the images with `docker images` and you will see your newly created image. If you want to commit it to the docker hub, you need to have an account, login with `docker login` and push the image with `docker push imagename:tag`.

Exercise 5 Recreate the `feeg6003/scicomp` image using a Dockerfile

Download the dockerfile example from the blog and fill it with `RUN apt-get install -y packagename` statements.

Exercise 6 Running a daemonized container

The following command line

`docker run -d -p 80:80 owncloud` will download the latest *owncloud* image, map the port `80` from the container to the port `80` in the host and run the container in the background.

After that, you only have to open `http://localhost` in your web browser and enjoy your very own (local) cloud.

Exercise 7 Managing containers and images

`docker ps -a` will display every container, running or not. You can remove them one by one with the command `docker rm containername`.

`docker images` will display the locally installed images. You can remove them with `docker rmi`.