Exercises lecture 2

# Docker

## The basics – Get to know the Docker commands

1. Verify once more that you are able to run docker containers by running the **hello-world Docker image**. What is the command and do you need to be root/run it with sudo?
2. Have a look at the “/etc/group” file. What new group has been created during the installation of docker?
3. If this wasn’t done in the previous exercises lab. Look online (Google), in the man-pages, or take a trip down memory lane and find out to add yourself (the regular user) to this group.
4. Log out and back in to make the changes effective.
5. Now create another instance of the hello-world image and see if you still need sudo?
6. Use docker ps. Is the hello-world container still running?
7. Find the flag that shows all containers (since apparently by default it only shows running containers).
8. In the output of the hello-world container there is a line that says: “To try something more ambitious, you can run an Ubuntu container with:”. What is the command?
9. Execute the command, which will drop you in a shell on that container. Find out the options you used in the docker run command. Execute a cat /etc/os-release to find out the details of the Linux distro you’re in (in the container). What is the codename of this Ubuntu release?
10. Exit out of the container, is it still running?

## Run a webserver

1. Now let’s run an Apache2 web server on your Debian-docker without installing the apache2 software (in other words use a Docker container). You can use the ‘httpd’ image. Run it like this: `docker run -d httpd`. You already know what -it stands for with docker run, what does the -d do?
2. Now surf with a web browser on your laptop (host) to the ip (or hostname) of debiandocker. Can you see the apache start page or not?
3. Is it possible to do a curl to localhost **from your debiandocker**?
4. Containers have their own internal network within the debiandocker machine. (Tip: take a look at the ip configuration (ip a) on your debiandocker. Use `docker ps` to find out the container id of your running (webserver) container followed by the `docker inspect` command to retrieve more information about the container including the ip address it received.
5. Is it possible (**from your debiandocker)** to perform a curl to the ip address of the container showing output that the apache2 software is running?
6. Now, we want to make a website available on our debiandocker (so not only on the separate internal docker network within debiandocker) and we want to host our own files on the web server of this container. Create a folder in the home folder of your user called testweb. In that folder create a php-file “info.php” with the phpinfo() function in it:

<?php

Phpinfo();

?>

1. Run the following command: `docker run -dit --name my-first-apache-app -p 8080:80 -v ~/testweb/:/usr/local/apache2/htdocs/ httpd`
2. Do you understand everything in this docker command (hint: man docker run)? Try to browse to the IP (and the correct port) of your debiandocker with a web browser on your laptop.
3. Afterwards surf to info.php. Did the phpinfo() do his job? Look at <https://hub.docker.com/_/httpd> to find out why it failed and to find another image to work with. Stop your container with ‘docker stop <containerid>’. What is the new docker command with the php image (without needing a Dockerfile)? Does the info.php now display things correctly?
4. You might have realized that container names need to be unique. So in other words we need to do some cleanup from time to time. Try to find out what the following different commands do: `docker stop, docker rm, docker system prune, docker image prune`

## LAMP stack

This exercise gives you freedom as there are multiple ways to create this. Make use of the docker hub, the web or other resources to create a proper docker-compose file that configures a LAMP-stack.



Don’t worry about the Linux part as we are running Linux containers on a Debian. If you prefer nginx over apache or postgresql over mysql go ahead and make the adjustments. You could even try to implement a phpMyAdmin or Adminer container.