Docker Containers Configuration and Deployment

System Benchmarking and Deployment

2020/2021

The main goal of this guide is to understand how to configure and deploy docker containers running different services/applications.

Useful Docker documentation is available at:

- Docker Documentation - `https://docs.docker.com`
- Step by Step Example - `https://docs.docker.com/get-started/`

Setup

1. Use the VagrantFile, written in the *Virtual Machine Deployment and Configuration* lab guide, to install the needed Docker utilities in one VM. Docker and Docker-Compose utilities can be installed with the following commands:

```
sudo apt-get install -y apt-transport-https ca-
    certificates curl software-properties-common
curl -fsSL https://download.docker.com/linux/
    ubuntu/gpg | sudo apt-key add -
sudo apt-key fingerprint 0EBFCD88
sudo add-apt-repository "deb [arch=amd64] https
    ://download.docker.com/linux/ubuntu $(
    lsb_release -cs) stable"
sudo apt-get -y update
sudo apt-get -y install docker-ce
sudo curl -L https://github.com/docker/compose/
    releases/download/1.16.1/docker-compose-‘
    uname -s‘-‘uname -m‘ -o /usr/local/bin/docker
    -compose
sudo chmod +x /usr/local/bin/docker-compose
```

2. Deploy the VM using Vagrant.
3. At the host machine, create the folder *docker_env* which will be used as the base directory for the next steps.

4. Download the sample.war file at `https://tomcat.apache.org/tomcat-6.0-doc/appdev/sample/` to a folder called *myapps*, while this folder should be placed inside the *docker_env* folder. This war file will be used to test the tomcat service that will be deployed during this lab guide.

Steps

Deploying a single Docker Container

1. Setup a Docker configuration file (*Dockerfile*) that will deploy apache tomcat in a Ubuntu 18.04 image. The commands for installing and starting tomcat in ubuntu are:

```
apt-get update && apt-get -y upgrade
apt-get -y install software-properties-common curl
apt-get -y install openjdk-8-jre-headless
curl -O http://archive.apache.org/dist/tomcat/
    tomcat-7/v7.0.55/bin/apache-tomcat-7.0.55.tar.
    gz
tar xzf apache-tomcat-7.0.55.tar.gz
apache-tomcat-7.0.55/bin/startup.sh && tail -f
    apache-tomcat-7.0.55/logs/catalina.out
```

2. The Dockerfile should specify a command to copy the *myapps* folder to the path */apache-tomcat-7.0.55/webapps/* at the Docker container.

3. Port 8080 should be exposed to the host.

4. For the previous configurations see the documentation of FROM, RUN, EXPOSE, CMD and COPY Dockerfile commands.

5. Use the command *scp* to copy the *docker_env* folder to the VM with the ip *10.0.0.101*.

6. At the VM, build the Docker image using the *docker build* command. The image should be named *my/tomcat*.

7. Deploy the container with the command *docker run*. In order to be able to reach tomcat from the host machine use the *-p 8080:8080* flag. To run the container in background use the *-d* flag.

8. Understand the usage of the commands *docker ps, exec, stop, kill. E.g.*, understand and explore the *docker exec -ti "container_id" /bin/bash* command.

9. Access the tomcat service from the host machine browser (*10.0.0.101:8080/sample*).

Docker Compose

1. At the VM, create a docker compose YAML file that specifies a service called *web* that will use one replica and will export port 8080 (`https://docs.docker.com/compose/gettingstarted/`).
2. Launch the service with the command *docker-compose up -d web*.
3. Check that the docker container is running with the *docker ps* command.
4. Access the tomcat service from the host machine browser (*10.0.0.101:8080/sample*).

Learning outcomes Experiment linux Docker containers configuration and deployment. Revise Docker configuration parameters and deployment/management commands.