**10.Installation and deploying a PhP application on a Docker Container**

**Description:**

Create a Machine Image of Ubuntu Bionic 18.04LTS or Xenial 16.04.

To install **Docker CE**, first, you need to remove older versions of **Docker** were called **docker**, **docker.io**, or **docker-engine** from the system using the following command.

$ sudo apt-get remove docker docker-engine docker.io containerd runc

Next, you need to set up the Docker repository to install and update Docker from the repository using following commands.

1. Update the apt package index

$ sudo apt-get update

1. Install packages to allow apt to use a repository over HTTPS

$ sudo apt-get install \

apt-transport-https \

ca-certificates \

curl \

gnupg-agent \

software-properties-common

1. Add Docker’s official GPG key

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

1. Verify that you now have the key with the fingerprint 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88, by searching for the last 8 characters of the fingerprint

sudo apt-key fingerprint 0EBFCD88

pub rsa4096 2017-02-22 [SCEA]

9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88

uid [ unknown] Docker Release (CE deb) <docker@docker.com>

sub rsa4096 2017-02-22 [S]

1. Use the following command to set up the stable repository

$ sudo add-apt-repository \

"deb [arch=amd64] https://download.docker.com/linux/ubuntu \

$(lsb\_release -cs) \

stable"

The lsb\_release -cs sub-command below returns the name of your Ubuntu distribution, such as xenial. Sometimes, in a distribution like Linux Mint, you might need to change $(lsb\_release -cs) to your parent Ubuntu distribution. For example, if you are using Linux Mint Tessa, you could use bionic. Docker does not offer any guarantees on untested and unsupported Ubuntu distributions.

1. Update the apt package index and install the latest version of **Docker CE** using following commands.

$ sudo apt-get update

1. Install the latest version of Docker Engine - Community and containerd, or go to the next step 8 to install a specific version

$ sudo apt-get install docker-ce docker-ce-cli containerd.io

1. To install a specific version of Docker Engine - Community, list the available versions in the repo, then select and install: List the versions available in your repo:

$ apt-cache madison docker-ce

1. Install a specific version using the version string from the second column, for example, 5:18.09.1~3-0~ubuntu-xenial

$ sudo apt-get install docker-ce=<VERSION\_STRING> docker-ce-cli=<VERSION\_STRING> containerd.io

1. After successfully installing the **Docker CE** package, the service should be auto-started and auto-enabled to start at system boot, you can check its status using the following command.

$ sudo systemctl status docker

1. Press CTRL C to exit
2. Verify that Docker Engine - Community is installed correctly by running the hello-world image

$ sudo docker run hello-world

1. This command downloads a test image and runs it in a container. When the container runs, it prints the below informational message

Unable to find image 'hello-world:latest' locally

latest: Pulling from library/hello-world

1b930d010525: Pull complete

Digest: sha256:c3b4ada4687bbaa170745b3e4dd8ac3f194ca95b2d0518b417fb47e5879d9b5f

Status: Downloaded newer image for hello-world:latest

Hello from Docker!

This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.

2. The Docker daemon pulled the "hello-world" image from the Docker Hub.

(amd64)

3. The Docker daemon created a new container from that image which runs the

executable that produces the output you are currently reading.

4. The Docker daemon streamed that output to the Docker client, which sent it

to your terminal.

To try something more ambitious, you can run an Ubuntu container with:

$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:

https://hub.docker.com/

For more examples and ideas, visit:

https://docs.docker.com/get-started/

**Dockerizing a Node.js web application**

1. Create a new folder namely nodejsapp
2. Make a package.json file as follows

{

"name": "docker\_web\_app",

"version": "1.0.0",

"description": "Node.js on Docker",

"author": "Sashi’s First Nodejs Application on Container <sashi.mamidanna@gmail.com>",

"main": "server.js",

"scripts": {

"start": "node server.js"

},

"dependencies": {

"express": "^4.16.1"

}

}

1. Then create a file server.js to create a program that runs on the node. The idea is to enable the server.js file to run on the container at port no 8081

'use strict';

const express = require('express');

// Constants

const PORT = 8081;

const HOST = '0.0.0.0';

// App

const app = express();

app.get('/', (req, res) => {

res.send('Hello world\n');

});

app.listen(PORT, HOST);

console.log(`Running on http://${HOST}:${PORT}`);

1. Create a dockerfile now namely dockerfile in the same directory

$sudo nano dockerfile

1. Copy the source code into the dockerfile

FROM node:10

# Create app directory

WORKDIR /app

COPY . /app

RUN npm install

COPY . .

EXPOSE 8082

CMD [ "node", "server.js" ]

1. Now build the docker image with the node application on it

$sudo docker build -t nodejsapp .

1. Run the application by executing run command on docker

$sudo docker run -p 8082:8081 nodejsapp

1. The container engine will run the command node server.js that was initialized through the dockerfile. Now the server.js is listening to incoming requests on <http://localhost:8081> on the host operating system. But the application is running on port number 8082 on the docker engine.
2. Open a new ssh connection on the same VM and run the command to send an outgoing request to the application running on docker

$sudo curl http://localhost:8082z

Hello World

1. This response is a result of the application running on node, devoted on the docker container, that's running on Docker engine available on the Ubuntu OS.
2. Run the bow command to check if the docker image is present in the list of images on Docker C

$sudo docker ps

1. To stop the docker container image

$sudo docker stop <docker image ID>

1. To remove the docker image
2. $sudo docker rmi <docker image ID>