Dockerize Sample Application

Overview

Dockerizing is the process of packing, deploying, and running applications using Docker containers. Docker is an open source tool that ships your application with all the necessary functionalities as one package.

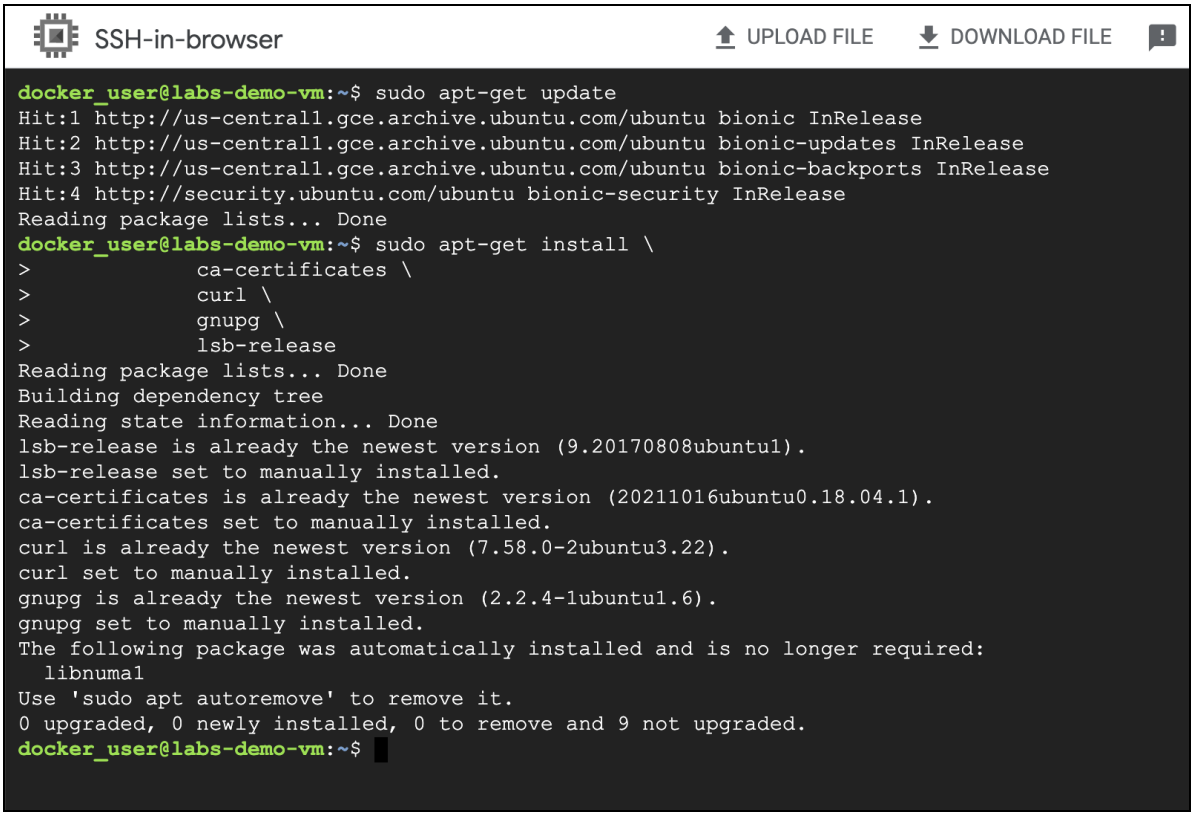
In this section, sample C++ applications can be deployed on Ubuntu 18.04 virtual machines (Created in Lab 1).

Setup

Sample Hello World

- SSH into VM created previously in Lab 1. Clone the following repository on VM using command:
    - git clone https://github.com/ketanjoshi10/synopsys-hackathon-workshop.git
- Folder named synopsys-hackathon-workshop will be created in the present working directory(PWD).
- The repository consists of C++ application code and Dockerfile to containerize the application.
- To run dockerize applications on VM follow the following steps till docker version.
- Install Docker on a virtual machine. Follow the below steps
    - Docker Installation on ubuntu 18.04 (Reference)
    - Update the apt package index and install packages to allow apt to use a repository over HTTPS:

```
$ sudo apt-get update
$ sudo apt-get install \
        ca-certificates \
        curl \
        gnupg \
        lsb-release
```



- Add Docker's official GPG key

```
sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

- Use the following command to set up the repository

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
docker_user@labs-demo-vm:~$ echo \
>   "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download
.docker.com/linux/ubuntu \
>   $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
docker_user@labs-demo-vm:~$
```

- Install Docker Engine

```
sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io
docker-compose-plugin
```

```
docker_user@labs-demo-vm:~$ sudo apt-get update
Hit:1 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:3 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic-backports InRelease
Get:4 https://download.docker.com/linux/ubuntu bionic InRelease [64.4 kB]
Hit:5 http://security.ubuntu.com/ubuntu bionic-security InRelease
Get:6 https://download.docker.com/linux/ubuntu bionic/stable amd64 Packages [31.3 kB]
Fetched 95.7 kB in 1s (169 kB/s)
Reading package lists... Done
docker_user@labs-demo-vm:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compos
e-plugin
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libnuma1
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  docker-ce-rootless-extras docker-scan-plugin libltdl7 pigz
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
Recommended packages:
  slirp4netns
The following NEW packages will be installed:
  containerd.io docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin
  docker-scan-plugin libltdl7 pigz
0 upgraded, 8 newly installed, 0 to remove and 9 not upgraded.
Need to get 111 MB of archives.
After this operation, 428 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic/universe amd64 pigz amd64 2.4-1 [57.4
kB]
Get:2 http://us-central1.gce.archive.ubuntu.com/ubuntu bionic/main amd64 libltdl7 amd64 2.4.6-2 [38.
8 kB]
```

- Check the docker version and service status.

```
#To check docker version
docker --version

#To check docker service running
systemctl status docker
```

```
docker_user@labs-demo-vm:~$ docker --version
Docker version 20.10.22, build 3a2c30b
docker_user@labs-demo-vm:~$ systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2023-01-19 10:57:17 UTC; 47s ago
     Docs: https://docs.docker.com
 Main PID: 4493 (dockerd)
    Tasks: 8
   CGroup: /system.slice/docker.service
           └─4493 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
docker_user@labs-demo-vm:~$
```

- Add your user to the **Docker** group to run all docker commands effortlessly.
- Use the below command to get the current user. Replace your username in the $USER command.

```
#Display current user logged in
whoami

#Add your user to Docker group (Replace with output of above command)
sudo usermod -aG docker <user>

#Validate user added to Docker group
getent group docker

#reboot the system
sudo reboot
```

```
docker_user@labs-demo-vm:~$ whoami
docker_user
docker_user@labs-demo-vm:~$ sudo usermod -aG docker docker_user
docker_user@labs-demo-vm:~$ getent group docker
docker:x:999:docker_user
docker_user@labs-demo-vm:~$
```

- Post reboot, your user is added to the docker group. Now all docker commands can be executed with ease. Run below command to verify.

```
docker info
```

```
docker_user@labs-demo-vm:~$ docker info
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  app: Docker App (Docker Inc., v0.9.1-beta3)
  buildx: Docker Buildx (Docker Inc., v0.9.1-docker)
  compose: Docker Compose (Docker Inc., v2.14.1)
  scan: Docker Scan (Docker Inc., v0.23.0)

Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 20.10.22
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
 Cgroup Version: 1
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 io.containerd.runtime.v1.linux runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: 5b842e528e99d4d4c1686467debf2bd4b88ecd86
```

- Build Docker image for HelloWorldProject0

cd synopsys-hackathon-workshop/DockerHelloWorldProject0/

docker build -t hello0 .

docker images

```
docker_user@labs-demo-vm:~$ cd labs-sample-app/DockerHelloWorldProject0/
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject0$ docker build -t hello0 .
Sending build context to Docker daemon  4.608kB
Step 1/8 : FROM amytabb/docker_ubuntu16_essentials
latest: Pulling from amytabb/docker_ubuntu16_essentials
8ee29e426c26: Pull complete
6e83b260b73b: Pull complete
e26b65fd1143: Pull complete
40dca07f8222: Pull complete
b420ae9e10b3: Pull complete
dfaf13193fe3: Pull complete
Digest: sha256:aa1aee49d49b6d641a81afeedb252befe55fbbce6a6badc50e61cf649dc9e8e6
Status: Downloaded newer image for amytabb/docker_ubuntu16_essentials:latest
 ---> 13f6d277d91b
Step 2/8 : COPY HelloWorld /HelloWorld
 ---> 1029acae5704
Step 3/8 : WORKDIR /HelloWorld/
 ---> Running in b39d2f0327a5
Removing intermediate container b39d2f0327a5
 ---> 951820d1e1ad
Step 4/8 : RUN g++ -o HelloWorld helloworld.cpp
 ---> Running in 355eaf2c02c5
Removing intermediate container 355eaf2c02c5
 ---> 6a96819ee5b7
Step 5/8 : COPY script.sh /HelloWorld/script.sh
 ---> c3adc03ae48c
Step 6/8 : EXPOSE 80
 ---> Running in 9552334e76ce
Removing intermediate container 9552334e76ce
 ---> fe6f495533d5
Step 7/8 : RUN ["chmod", "+x", "/HelloWorld/script.sh"]
 ---> Running in 9abe887d0538
Removing intermediate container 9abe887d0538
 ---> 711d1ea9d7bf
Step 8/8 : ENTRYPOINT ["/HelloWorld/script.sh"]
 ---> Running in 95ae53bb1923
Removing intermediate container 95ae53bb1923
 ---> 8dd521481839
Successfully built 8dd521481839
Successfully tagged hello0:latest
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject0$ docker images
REPOSITORY                          TAG       IMAGE ID       CREATED         SIZE
hello0                              latest    8dd521481839   9 seconds ago   807MB
amytabb/docker_ubuntu16_essentials  latest    13f6d277d91b   4 years ago     807MB
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject0$
```

- Create a docker container using the created image and check the output.
  $ docker run -it hello0

```
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject0$ docker run -it hello0
Hello world 0!
```

- The container will run for 900 sec and automatically exit after that, else Press Ctrl + C to exit the container before 900 sec.
- To conclude, sample Hello World C++ application containerized locally and validate the result.

Sample Hello World with Arguments

- Build Docker image for HelloWorldProject0

```
cd synopsys-hackathon-workshop/DockerHelloWorldProject1/

docker build -t hello1 .

docker images
```

```
docker_user@labs-demo-vm:~$ cd labs-sample-app/DockerHelloWorldProject1/
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject1$ docker build -t hello1 .
Sending build context to Docker daemon  4.608kB
Step 1/10 : FROM amytabb/docker_ubuntu16_essentials
 ---> 13f6d277d91b
Step 2/10 : ENV NAME VAR1
 ---> Running in d75d5b426ad2
Removing intermediate container d75d5b426ad2
 ---> 8806bb070226
Step 3/10 : ENV NAME VAR2
 ---> Running in b00dc695f44c
Removing intermediate container b00dc695f44c
 ---> 35db7ca7de41
Step 4/10 : ENV NAME VAR3
 ---> Running in 0073e0708317
Removing intermediate container 0073e0708317
 ---> 7c638ed96b81
Step 5/10 : COPY run_hello1.sh /run_hello1.sh
 ---> 27459045615e
Step 6/10 : COPY HelloWorld /HelloWorld
 ---> 33c93c797c27
Step 7/10 : WORKDIR /HelloWorld/
 ---> Running in 93317dd9862a
Removing intermediate container 93317dd9862a
 ---> 739e031492ff
Step 8/10 : RUN g++ -o HelloWorld1 helloworld1.cpp
 ---> Running in 3de2a2dd0eff
Removing intermediate container 3de2a2dd0eff
 ---> c34e84bb0dd6
Step 9/10 : WORKDIR /
 ---> Running in c1242674cd1d
Removing intermediate container c1242674cd1d
 ---> fd02c1a6220c
Step 10/10 : CMD ["/bin/sh", "/run_hello1.sh"]
 ---> Running in 9fe8a3e4926c
Removing intermediate container 9fe8a3e4926c
 ---> 63bcba4a8eb7
Successfully built 63bcba4a8eb7
Successfully tagged hello1:latest
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject1$ docker images
REPOSITORY                          TAG       IMAGE ID       CREATED         SIZE
hello1                              latest    63bcba4a8eb7   6 seconds ago   807MB
hello0                              latest    8dd521481839   3 minutes ago   807MB
amytabb/docker_ubuntu16_essentials  latest    13f6d277d91b   4 years ago     807MB
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject1$
```

- Create a docker container using the created image and check the output.

  $ docker run -it -e VAR1='23' hello1

```
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject1$ docker run -it -e VAR1='23' hello1
Hello world 1, with arguments!
Argument 1 23
docker_user@labs-demo-vm:~/labs-sample-app/DockerHelloWorldProject1$
```

- To conclude, sample Hello World C++ application with argument containerized locally and validate the result.

References

- Docker Installation - https://docs.docker.com/engine/install/ubuntu/