SOFE4630 Cloud Computing (Winter 2022 - Dr. M. El-darieby)

Lab 1: Project Milestone-- IaaS: Virtualization and Containerization

Mitul Patel

Learning Objectives:

- Understand containers.
- Build containers from images.
- Build customized images.
- Build and run multi-container Docker applications.
- Get familiar with Google Cloud Platform (GCP).
- Run docker images on GCP.
- Get familiar with how to use Kubernetes on GCP.

Procedure:

Video Activity 1

1. Watch the following introductory video that describes what is the difference between containerization and virtualization

Virtualization:

- Hosts Multiple Operating system
- Uses up more resources
- Requires a hypervisor
- Separates available resources (computing power, memory, storage.. etc)'

Containerization

- Enables deploying multiple application on same OS
- One server or VM can run multiple container
- Uses up less resources compared to Virtualization

- Requires a Runtime Engine for the containers

Description: Virtualization is designed to run server side applications on a VM each with its own OS, memory and computing resources. It takes up a long time to boot up due to heavy OS load. Virtualization allows for teams to centralized workload and run several OS. Containerization is a way to package any type of software along all the versions, libraries, and dependencies into a container which runs with the help of a runtime engine. This removes the need for installing new OS for each VM on a server.

2. Install Docker on your local machine

   //file does not have audio

   Docker install link: https://drive.google.com/file/d/1INfQReRaMsW9mnRhLO_B-jA4FXKYuC9-/view?usp=sharing

```
C:\Users\patel>docker --version
Docker version 20.10.12, build e91ed57

C:\Users\patel>sn
```

3.

```
C:\Users\patel\IdeaProjects\HelloWorldDocker>docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   1.0       bfccf64b4723   6 hours ago    471MB
hello-world   2.0       e54295f84d34   6 hours ago    471MB
<none>        <none>    84920d82c4b8   6 hours ago    471MB
<none>        <none>    558ff10ceb67   6 hours ago    471MB

C:\Users\patel\IdeaProjects\HelloWorldDocker>
```

4. Answer the following questions.
5. What are docker images, container, and registry?
   a. Images
      - Docker image is a read-only template which contains a set of instructions

- Instructions for creating a container which runs on docker platform
- Images are like a java class

b. Containers
   i. Running instance of a docker image
   ii. Each container can be considered an object in the class/object analogy
   iii. Each container contains the application which it is running

c. Registry
   i. Docker registry is used for versioning
   ii. It is used for storing and distributing docker images
   iii. If you want to reliably deploy containers for an enterprise application, it is better to use a docker registry.

Video 1 Commands

6. List the Docker commands used in the video with a brief description for each command and option.

- *Docker images* => lists all the docker images
- *Docker build -t hello-world:1.0* => builds a docker image with the name hello-world and tag 1.0
- *Docker run hello-world:1.0* => runs the docker image on a container
- *Docker ps* => shows all running containers
- *Docker ps -a* => shows all running and non-running containers
- *Docker run -d hello-world:2.0* => runs the application in the background
- *Docker logs [id]* => shows the container output of based on the id
- *Docker stop [container id*] => stops the running container
- *Docker pause [id]* => pauses the container

7. At the end of the video, there are two running containers, what commands can be used to stop and delete those two containers?
   - *Docker stop [container_id]* => The command above stops the two containers
   - *Docker image rm [container_id]* => deletes the image

8. The video 1 completion is available at the link: //file does not have audio

   https://drive.google.com/file/d/1BJb8PePG1ox-wgXzG4FS2Ya22fj2LZ9b/view?usp=sharing

Video 2 Activity

Running mysql container

```
C:\Users\patel\IdeaProjects\MyWebApp>docker run --name app-db -d -e MYSQL_ROOT_PASSWORD=password -e MYSQL_DATABASE=myDB mysql
8b8672884768e2218b7f972e2186f0b55e4d1ed883e053b47af09ab5aa192e86
```

```
C:\Users\patel\OneDrive\Documents\Year IV\Semester 2\Cloud Computing\Labs\Lab1\SOFE4630U-tut1\v2>docker ps
CONTAINER ID   IMAGE            COMMAND                  CREATED              STATUS              PORTS                 NAMES
2d42f2f2bedb   my-web-app:1.0   "catalina.sh run"        About a minute ago   Up About a minute   8080/tcp              app
8b8672884768   mysql            "docker-entrypoint.s…"   17 hours ago         Up 17 hours         3306/tcp, 33060/tcp   app-db

C:\Users\patel\OneDrive\Documents\Year IV\Semester 2\Cloud Computing\Labs\Lab1\SOFE4630U-tut1\v2>
```

What's a multi-container Docker application?

- A multi -container docker application is an application that utilizes different types of services to separate and modularize the application.
- An example would be MySQL would exist on one container
- Apache web server would exists on an another container

Justification:

The main reason for deploying multiple containers for different services is due to scalability, deployment and modularity. This helps separate the concerns for an application to scale each component separately. This also provides more flexibility.

How do these containers communicate together?

- Each container communicate if they are part of the same network
- Docker creates a virtual network called a bridge
- Used to connect your container to other containers
- Ports can be exposed to the container and binded on runtime.
- In the network, each container is assigned an IP address.
- Containers communicate with the help of an IP address.

What commands are used to stop the docker application and delete its image

- *Docker stop [containerId]* => stop the running container
- *Docker image rm [containerId]* => remove the container instance.

List of new Docker commands:

//pulls mysql liberty from the docker.io library

Docker pull mysql

// Runs a Mysql container with the root password and db name along with the mysql take to use the library which was pulled earlier

Docker run --name app-db -d -e MYSQL_ROOT_PASSWORD=password -e MYSQL_DATABASE=myDB mysql

//Runts the docker container with port 8080, with a -p option which means publish and binds port 8080 with the external port for 8080 and d flags allows for it to run in the background.

docker run --name app -d -p 8080:8080 my-web-app:1.0

//creates a network bridge which allows for containers to communicate with each other

Docker network create

//displays Network Id, name, driver and scope

Docker network ls

//Connects network to the app-db

docker network connect app-network app-db

//Runs the application container and publishes the port on 8080 and exposes the endpoint while connecting the web-app to the network.

Docker run --name app -d -p 8080:8080 --network my-web-app:1.0

//using docker compose, you can manage the build commands in a yml file called docker compose. The command bellow runs the docker-compose file

Docker-compose up - d

Video Link to video activity 2:

https://drive.google.com/file/d/1cZZRlUFCR-UX0x2KcvVZJyslRnlOdXZ2/view?usp=sharing

Video Activity 3:

List all used GCP shell commands and their description

Video Link to video activity 3 deploy ngnix server and webapp server with audio:

https://drive.google.com/file/d/132OUEGLxkURml4TrB4UHfmDSh1_RiluT/view?usp=sharing

Commands	Explanation
Part 1	V3 index.html file
docker run -p 8080:80 nginx:latest	The following commands are explained in part 2 bellow
docker cp index.html [container-id]:/usr/share/nginx/html/	The following commands are explained in part 2 bellow
docker commit [container-id] cad/web:version1	The following commands are explained in part 2 bellow
docker tag cad/web:version1 us.gcr.io/youtube-demo-255723/cad-site:version1	The following commands are explained in part 2 bellow
docker push us.gcr.io/youtube-demo-255723/cad-site:version1	The following commands are explained in part 2 bellow
Part 2	Deploying mywebapp to google kubernetes using yml file
docker build -t my-web-app:1.0	Builds the docker image with the name my-web-app with version 1.0
docker tag my-web-app:1.0 us.gcr.io/<project_id>/webapp:version1	Tags the image with the name tag us.gcr.io/<project_id>/webapp:version 1. This tells google cloud about the

	deployment tag
docker push my-web-app:1.0 us.gcr.io/<project_id>/webapp:version1	Commands pushs the image app on to google container registry
gcloud config set project <project_id>	This sets the gcloud configuration to the project id and it is used to deploy container in GKE cluster
gcloud config set compute/zone us-central1-a	The following command is used to set the compute/zon on the GKE cluster
gcloud container clusters create gk-cluster --num-nodes=1	This commands is used to create a GKE cluster with the number of nodes
gcloud container clusters get-credentials gk-cluster	This command is used configures the kubectl to ue the cluster I created
kubectl apply -f webApp.yml	This command applies the yml file on the application to build the containers
kubectl create deployment web-server --image=us.gcr.io/<project_id>/cad-site:version1	This command is used to deploy an application to the cluster
kubectl expose deployment web-server --type LoadBalancer --port 80 --target-port 8080	This command is used to expose the deployment cluster to a port 80 and a target port of 80
kubectl get pods	Lists the running pods to view the application
kubectl get service	Inspects the webapp to get the external ip address.

What is Kubernetes' pod, service, node, and deployment?

A pod is created and contains the application instance. Pod is an abstraction which represents a group of one or more containers. It has resources such as:

- Shared storage
- Networking with cluster Ip address
- Info on how different containers images versions and port run

A Node contains 1 or more pods. A node is a worker machine in kubernetes which may be a physical or virtual machine. Nodes can be managed by a control plane. A Node typically runs at least the following:

- Kublete, which is a process that manages communication between the kubernetes control plane and Node.
- Docker or also a container runtime which is needed for pulling the container images from a registry.

A Kubernetes deployment is a process of deploying your containerized application which is typically run by a yml file to a cluster of nodes which contain their own pods.

What is meant by replicas?

- Replicas is essentially hosting web application over several geographical location and maintaining multiple data across many servers.
- This comes with its own challenges to maintain data consistency and data availability
- Kubernetes allows this by deploying the application on several nodes across different areas for replication.

What are the types of Kubernetes services? What is the purpose of each?

Kubernetes has several services such as the following;

- ClusterIP
- NodePort
- LoadBalancer
- ExternalName

ClusterIp:

- Responsible for assigning a cluster-internal IP address to ClusterIP service.
- This makes the services reachable within the cluster

NodePort:

- Responsible for exposing the service outside of the cluster through adding a cluster-wide port on top of ClusterIP

LoadBalancer

- Responsible for integrating nodeport with cloud-based load balancers

ExternalName

- It is a service which provides an external name map service to a DNS name.

The image below shows a Node overview.

Node overview