Yusuf Shaik
100 655 451

All lab videos:
https://drive.google.com/drive/folders/1B1aHww6WhHIFygCM23zaV5RFgzoJnPE_?usp=sharing

What are docker image, container, and registry?

Container: standard component that allows users to package their applications and dependencies in an easy to share way. It is portable, isolated, and lightweight

Image: template containing instructions that is used to create and run a docker container

Registry: contains multiple docker images for commonly used applications. Popular images can be pulled to integrate into a new container

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

Docker build -t <image_name>:<image_tag> <path>

- This builds the docker image. *-t* allows users to specify the name of the image, and the image tag. <path> is the path to the dockerfile. E.g:
  docker build -t hello-world:1.0 .
- "." Is used here since the dockerfile is in the current directory.

Docker images

- Lists the current images along with the image id, time it was created, and the image size.

Docker run <image_name>:<image_tag>

- Instantiate an image and run container based on image
- Running with *-d* flag allows container to run in the background

Docker ps

- Lists running containers along with their info

Docker ps -a

- Lists all containers including running/stopped along with their info

Docker logs <container id>

- Shows the output of what is occurring in the docker container specified

Yusuf Shaik
100 655 451

At the end of the video, there are two running containers, what commands can be used to stop and delete those two containers?

Stop container: docker stop <container id>

Remove container: docker rm <container id>

What's a multi-container Docker application?

A multi container docker application is 2 or more containers that are used together to complete an application. E.g: web app and database. These two containers need to communicate over a network in order for the application to work. Hence the name, multi container docker application.

How these containers are communicated together?

Bridge networks are used for secure communication between two docker containers. We use "docker network create <network name>" to create it, then connect both containers to this network. "docker network connect <network name> <app name>" is used to connect a single container to the network. You can also do the following while running the container: docker run –name app -d -p 8080:8080 –network=app-network my-web-app:1.0

What command can be used to stop the Docker application and delete its images?

Stop docker container: Docker stop <container id>

Remove docker container: docker rm <container id>

Remove docker image: docker image rm <image id>

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

Docker pull mysql: pull mysql image from docker repository

Docker run –name app-db -d -e MYSQL_ROOT_PASSWORD=password -e MYSQL_DATABASE=myDB mysql: run app with environment variables (password for mysql)

docker network create <network name>: create network

docker network connect <network name> <app name>: connect app to network

Yusuf Shaik
100 655 451

docker run –name app -d -p 8080:8080 –network=app-network my-web-app:1.0: run app on network

List all used GCP shell commands and their description in your report:

**these commands were copied directly from my gcp terminal

docker run -d -p 8080:80 nginx:latest → run the web server container

docker ps → list available containers

git clone https://github.com/goergedaoud/SOFE4630U-tut1.git --> clone assignment repo

docker cp index.html 338ee85cc2e8:/usr/share/nginx/html/

docker commit 338ee85cc2e8 cad/web:version 1 → create local repo of container

docker images → list docker images on device

docker tag cad/web:version1 us.gcr.io/cloud-lab-1-340101/cad-site:version1 → name image before pushing to container registry

docker push us.gcr.io/cloud-lab-1-340101/cad-site:version1 → push to container registry

gcloud config set compute/zone us-central1-a → set google compute engine location

gcloud container clusters create gk-cluster --num-nodes=1 → create cluster with given number of nodes

gcloud container clusters get-credentials gk-cluster → configure kubectl to use cluster that was just created

docker images → list docker images

kubectl create deployment web-server --image=us.gcr.io/cloud-lab-1-340101/cad-site:version1 → deploy application to cluster

kubectl expose deployment web-server --type LoadBalancer --port 80 --target-port 80 → expose deployed application on ports

kubectl get pods → inspect running pods

kubectl get service web-server → inspect kubernetes service that is running

Yusuf Shaik
100 655 451

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

Pod: the most atomic deployable object in kubernetes. Pods are deployed when running your kubernetes cluster. Each pod contains one or more containers that are either alone, or replicas of other containers.

Service: a set of pods with some sort of access policy. Eg: database requests will always be forwarded to a set of pods

Node: nodes make up a kubernetes cluster. When a service is run its placed in a pod, and each node will contain 1 or more pods. A node is either a VM or an actual machine/device. Each node contains kubelet which enables communication between the node and the kubernetes cluster management.

Deployment: deploy a cluster of nodes with a given configuration, either ran manually or defined in a yaml configuration file.

What's meant by replicas?

A replica is a copy of another container in kubernetes. For example, if we want to ensure high availability of our database, we will create 3+ replicas of this database when deploying. These replicas are exact copies of eachother, and will ensure the functionality remains in the chance that one container fails.

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

Clusterip: exposes a service with an ip address within the cluster so that only other processes within the cluster can access a service. (default)

Nodeport: clusterip is created, but also the service is exposed on each nodes static ip address, and a defined "nodeport". External communication can access a node, and the request will be forwarded to the required service.

Loadbalancer: builds upon clusterip and nodeport. It uses the gcp load balancer and exposes the services on that load balancer. Requests are forwarded to the services within the cluster.

Externalname: this is used to map services from one name to an actual DNS name rather than simply an ip:port combination.