1. Create a new pod called admin-po with image busybox. Allow the pod to be able to set system\_time. Then container should sleep for 3200 seconds.

Solution:

Kubectl run admin-pod –image=busybox –command sleep 3200 –dry-run=client -o yaml

apiVersion: v1

kind: Pod

metadata:

creationTimestamp: null

labels:

run: admin-pod

name: admin-pod

spec:

containers:

- command:

- sleep

- "3200"

image: busybox

name: admin-pod

securityContext:

capabilities:

add: ["SYS\_TIME"]

kubectl apply -f <file\_name>

1. A kubeconfig file called test.kubeconfig has been created in /root/TEST. There is something wrong with the configuration. Troubleshoot and fix it.

Solution:

Cd TEST

Ls

Cat test.kubeconfig

Kubectl config view

-----check the port from this output and replace the value in the kubeconfig file in the given path.

1. Create a new deployment called web-proj-268, with image nginx:1.16 and 1 replica. Next upgrade the deployment to version 1.17 using rolling update. Make sure that the version upgrade is record in the resource annotation.

Solution:

vi deploy-1.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: web-proj-268

labels:

app: nginx

spec:

replicas: 1

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:1.16

ports:

- containerPort: 80

kubectl apply -f deploy-1.yaml

kubectl get deploy

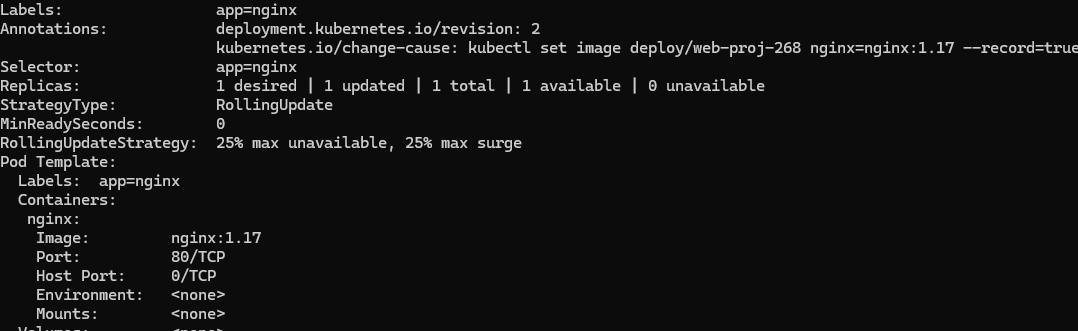
kubectl describe deploy/web-proj-268 --- here the nginx version will be 1.16

kubectl set image deploy/web-proj-268 nginx=nginx:1.17 --record

kubectl rollout status deployment/web-proj-268

kubectl get rs

kubectl describe deployments ---- this time the version of nginx is changed



kubectl rollout history deployment web-proj-268

1. Create a new deployment called web-003. Scale the deployment to 3 replicas. Make sure desired number of pod always running.

Solution:

This could be simple but trickier as well as the weightage is 7%.

Kubectl create deployment web-003 –image=nginx –replicas=3

Kubectl get pods

NOTE: control manager error could happen in the kube-system namespace.

Kubectl -n kube-system get pod

* CrashLoopBackOff error we will be encountering at this point

Kubectl -n kube-system describe pod <pod>

Cd /etc/Kubernetes/manifests

Ls

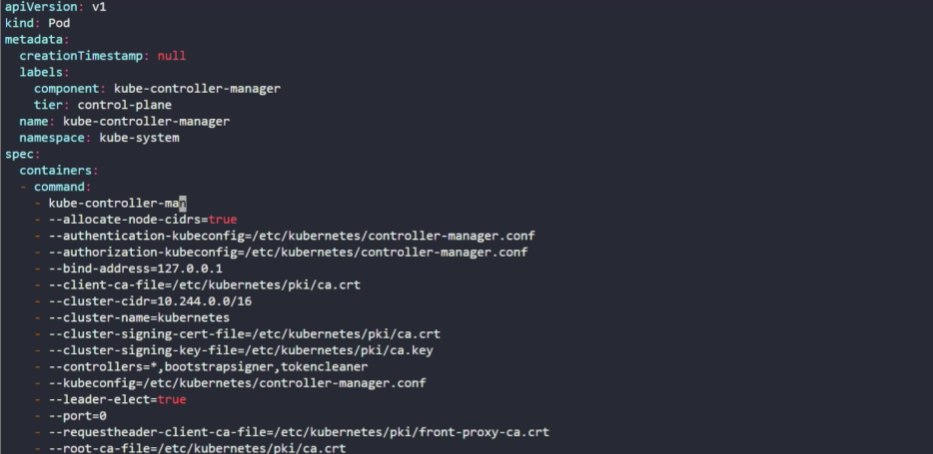
Vi kube-controller.yaml

Check for any misspelled words

Save and kubetl get deploy

Kubectl get pod

Watch Kubectl get deploy



1. Upgrade the cluster(Master and worker node) from 1.18.0 to 1.19.0. Make sure to first drain both the nodes and make it available after upgrade.

Solution:

Upgrading the controlplane:

Kubectl get node

Kubectl drain controlplane –ignore-daemonsets

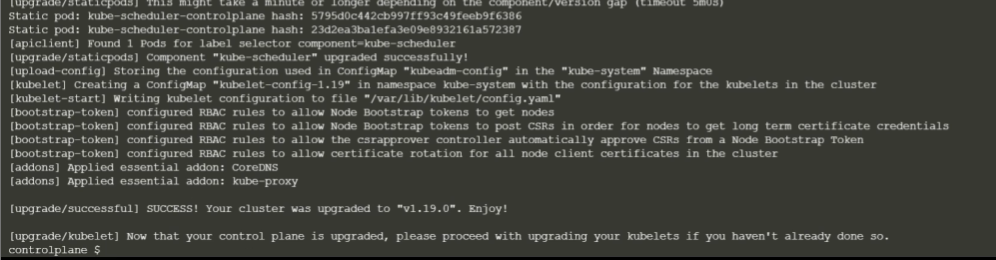
Kubectl get node

Kubectl get pod

Apt update

Apt install kubeadm=1.19.0-00

Kubeadm upgrade apply v1.19.0



Apt install kubelet=1.19.0-00

Systemctl restart kubelet

Kubectl get node

Kubectl uncordon controlplane

Kubectl get node

Upgrading the worker nodes:

Kubectl drain node01 –ignore-daemonsets

Kubectl get pod -o wide

Ssh node01

Apt update

Apt install kubeadm=1.19.0-00

Kubeadm upgrade node

Apt install kubelet=1.19.0-00

Systemctl restart kubelet

Go back to controlpane:

Kubectl get node

Kubectl uncordon node01

1. Deploy a web-load-5461 pod using nginx:1.17 image with the labels set to tier=web

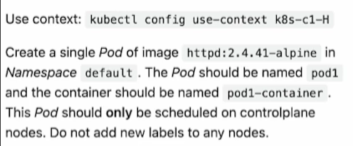
Solution:

vi deploy-3.yaml

kubectl run web-load-5461 –image=nginx:1.17 –labels tier=web

kubectl get pod -o wide

kubectl get pod –show-labels

**Q. **

**Ans:**

controlplane $ cat pod1.yaml

apiVersion: v1

kind: Pod

metadata:

creationTimestamp: null

labels:

run: pod1

name: pod1

spec:

nodeSelector:

node-role.kubernetes.io/control-plane: ""

containers:

- image: httpd:2.4.41-alpine

name: pod1-container

resources: {}

dnsPolicy: ClusterFirst

restartPolicy: Always

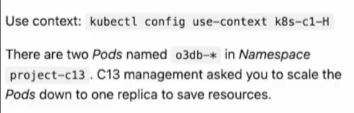
tolerations:

- key: "node-role.kubernetes.io/control-plane"

effect: "NoSchedule"

status: {}

controlplane $

Q. 

Ans:

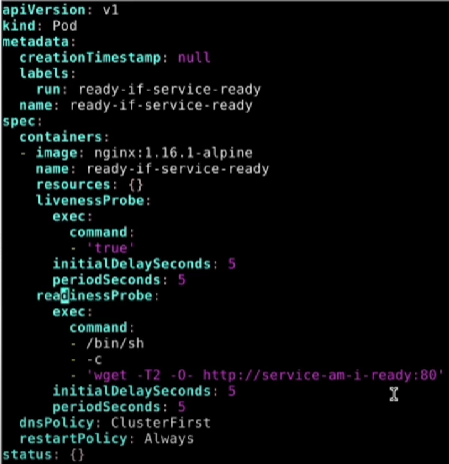
K scale –replicas=1 sts o3db -n project-c13

Q. 

Ans:

Step1: kubectl config use-context k8s-c1-H

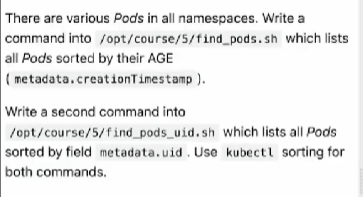
Step2: k run ready-if-service-ready –image=nginx:1.16.1-alpine –dry-run=client -o yaml



K create -f q4.yaml

k run am-i-ready --image nginx:1.16.1-alpine -l id=cross-server-ready

k get pods

Q. 

Solution:

Vi /opt/course/5/find\_pods.sh

#!/bin/bash

Kubectl get pods -A --sort-by=.metadata.creationTimestamp

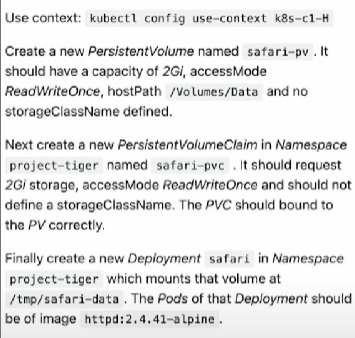
Sh /opt/course/5/find\_pods.sh

Vi /opt/course/5/find\_pods\_uid.sh

#!/bin/bash

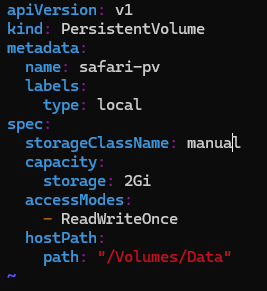
Kubectl get pods -A --sort-by=.metadata.uid

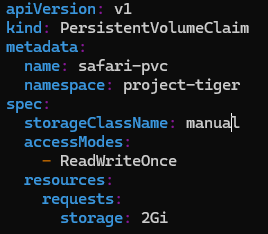
Sh /opt/course/5/find\_pods\_uid.sh

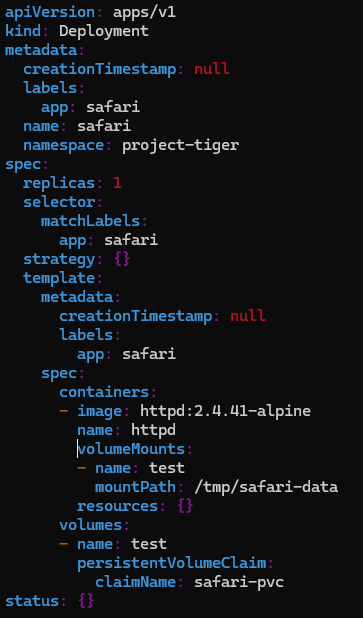
Q. 

Solution:

Kubectl config use-context k8s-c1-H







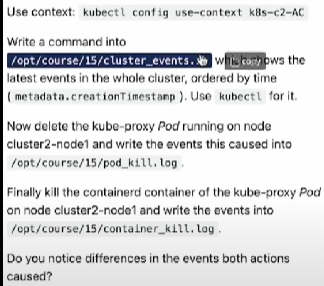
Mkdir -p /Volume/Data

K create ns project-tiger

k apply -f pv1.yaml

k apply -f pvclaim.yaml

k apply -f q6deploy.yaml

Q. 

Solution:

Vi /opt/course/15/cluster\_events.sh

#!/bin/bash

Kubectl get events -A –sort-by=.metadata.creationTimestamp

Sh /opt/course/15/cluster\_events.sh

K get pods -n kube-system -o wide

K delete pods kube-procy-<> -n kube-system

K get events -A > /opt/course/15/pod\_kill.log

K get nodes

Ssh node01

Crictl ps -a | grep kube-proxy

Crictl stop <>

Crictl rm <>

Logout

Kubectl get events -A > /opt/course/15/container\_kill.log

KUBERNETES SERVICE:

An abstract way to expose an application running on a set of pods as a network service.