Lab 2 – Kubernets

Toka Abdelhakim

1- Deploy a pod named nginx-pod using the nginx:alpine image with the labels set to tier=backend.

Yaml file:

pod-n.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  labels:
    tier: backend
spec:
  containers:
  - name: alpine
    image: nginx:alpine
```

2- Deploy a test pod using the nginx:alpine image.

Yaml file:

pod-n.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: test
spec:
  containers:
    - name: test
      image: nginx:alpine
```

Output:

```
controlplane $ kubectl get po
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   1          22m
test        1/1     Running   1          92s
```

3- Create a service backend-service to expose the backend application within the cluster on port 80.

Yaml file:

pod-n.yaml × pod-test.yaml× ba

```yaml
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  type: ClusterIP
  ports:
    - targetPort: 80
      port: 80
  selector:
    tier: backend
```

4- try to curl the backend-service from the test pod. What is the response?

Command: Kubectl exec test – curl http://backend-service

Terminal +

```
controlplane $ kubectl exec test -- curl http://backend-service
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
```

5- Create a deployment named web-app using the image nginx with 2 replicas

Yaml file:

pod-n.yaml × pod-test.yaml× backend-svc.ya

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web-app
  labels:
    app: web-app
spec:
  replicas: 2
  selector:
      matchLabels:
        app: web-app
  # use replica set definition
  template:
    metadata:
      labels:
        app: web-app
    spec:
      containers:
        - image: nginx
          name: nginx-deploy

```

6- Expose the web-app as service web-app-service application on port 30082 on the nodes on the cluster

Yaml file:

pod-n.yaml × pod-test.yaml×

```yaml
apiVersion: v1
kind: Service
metadata:
  name: web-app-service
spec:
  type: NodePort
  ports:
    - targetPort: 80
      port: 80
      nodePort: 30082
  selector:
    app: web-app

```

Output:

```
controlplane $ kubectl get svc
NAME              TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
backend-service   ClusterIP   10.106.118.252   <none>        80/TCP         16m
kubernetes        ClusterIP   10.96.0.1        <none>        443/TCP        146m
web-app-service   NodePort    10.98.170.221    <none>        80:30082/TCP   62s
controlplane $ kubectl get no
```

7- access the web app from the node

8- How many Nodes exist on the system?

Two nodes

```
controlplane $ kubectl get no
NAME           STATUS   ROLES    AGE    VERSION
controlplane   Ready    master   146m   v1.14.0
node01         Ready    <none>   146m   v1.14.0
```

9- Do you see any taints on master?

Command: kubectl describe node controlplane | grep Taint

No

```
controlplane $ kubectl describe node controlplane | grep Taint
Taints:             node-role.kubernetes.io/master:NoSchedule
controlplane $
```

10- Apply a label color=blue to the master node

Command: kubectl label nodes controlplane color=blue

```
Terminal
controlplane $ kubectl describe no controlplan
Name:               controlplane
Roles:              master
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    color=blue
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=controlplane
                    kubernetes.io/os=linux
                    node-role.kubernetes.io/master=
Annotations:        kubeadm.alpha.kubernetes.io/cri-socket: /var/run/dockershim.sock
                    node.alpha.kubernetes.io/ttl: 0
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Sat, 18 Sep 2021 22:33:41 +0000
Taints:             node-role.kubernetes.io/master:NoSchedule
Unschedulable:      false
```

11- Create a new deployment named blue with the nginx image and 3 replicas
Set Node Affinity to the deployment to place the pods on master only
NodeAffinity: requiredDuringSchedulingIgnoredDuringExecution
Key: color
values: blue

Yaml file:

blue-deploy.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: blue
spec:
  selector:
    matchLabels:
      color: blue-d
  replicas: 3
  template:
    metadata:
      labels:
        color: blue-d
    spec:
      containers:
      - name: my-nginx
        image: nginx
      affinity:
        nodeAffinity:
            requiredDuringSchedulingIgnoredDuringExecution:
                nodeSelectorTerms:
                - matchExpressions:
```

12- How many DaemonSets are created in the cluster in all namespaces?

Command: kubectl get ds --all-namespaces

```
controlplane $ kubectl get ds --all-namespaces
NAMESPACE     NAME                 DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
kube-system   kube-keepalived-vip   1         1         1       1            1           <none>          52m
kube-system   kube-proxy            2         2         2       2            2           <none>          52m
kube-system   weave-net             2         2         2       2            2           <none>          52m
```


13- what DaemonSets exist on the kube-system namespace?

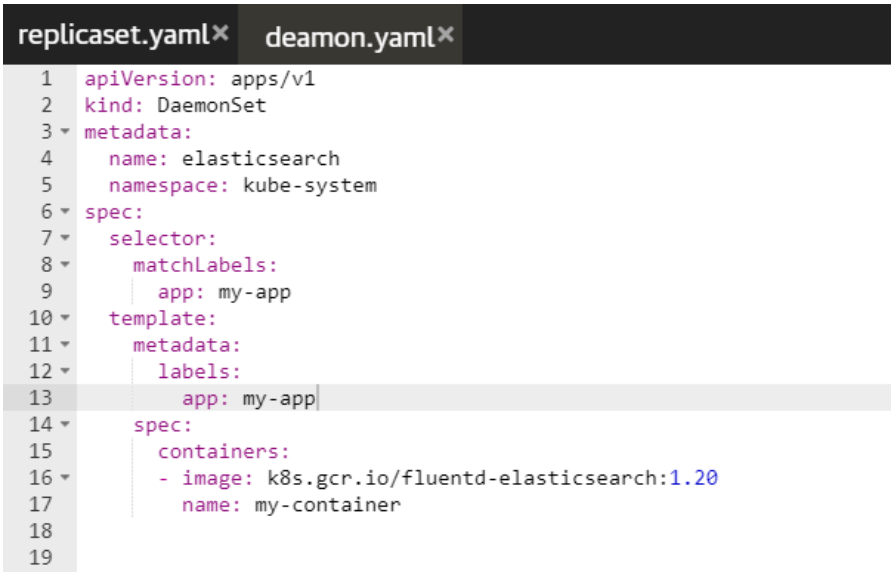
Command: kubectl get ds -n kube-system

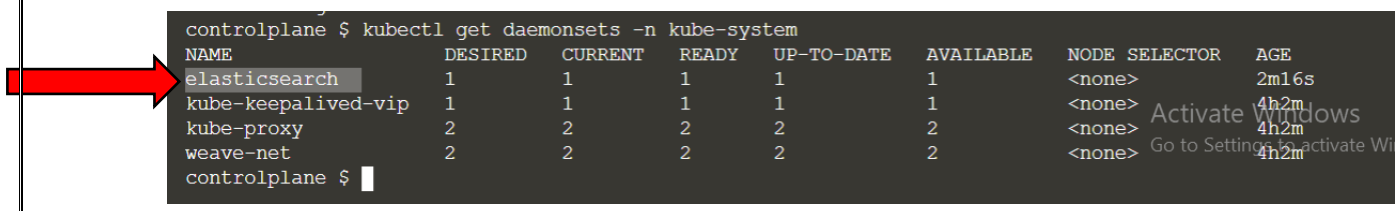
Three DaemonSets

```
controlplane $ kubectl get ds -n kube-system
NAME                  DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
kube-keepalived-vip   1         1         1       1            1           <none>          55m
kube-proxy            2         2         2       2            2           <none>          55m
weave-net             2         2         2       2            2           <none>          55m
controlplane $
```

14- What is the image used by the POD deployed by the kube-proxy DaemonSet

15- Deploy a DaemonSet for FluentD Logging. Use the given specifications.
Name: elasticsearch
Namespace: kube-system
Image: k8s.gcr.io/fluentd-elasticsearch:1.20





16- Create a taint on node01 with key of spray, value of mortein and effect of NoSchedule

Command: kubectl taint nodes node01 spray= mortein: NoSchedule

```
controlplane $ kubectl taint nodes node01 spray=mortein:NoSchedule
node/node01 tainted
controlplane $ kubectl describe node node01 | grep Taint
Taints:             spray=mortein:NoSchedule
controlplane $
```

17- Create a new pod with the NGINX image, and Pod name as Mosquito

18- What is the state of the mosquito POD?

Pending

Yaml file:

replicaset.yaml deamon.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mosquito
spec:
  containers:
  - image: nginx
    name: my-container-name

```

State: Pending

```
controlplane $ kubectl get po
NAME       READY   STATUS    RESTARTS   AGE
mosquito   0/1     Pending   0          9s
controlplane $
```

19- Create another pod named bee with the NGINX image, which has a toleration set to the taint Mortein Image name: nginx Key: spray Value: mortein Effect: NoSchedule Status: Running

replicaset.yaml × deamon.yaml × pc

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: bee
spec:
  containers:
  - image: nginx
    name: my-container-name
  tolerations:
    - key: "spray"
      operator: "Equal"
      value: "mortein"
      effect: "NoSchedule"
```

State: Running

```
controlplane $ kubectl get po -o wide
NAME       READY   STATUS    RESTARTS   AGE     IP            NODE     NOMINATED NODE   READINESS GATES
bee        1/1     Running   0          27m     10.32.0.194   node01   <none>           <none>
mosquito   0/1     Pending   0          5m58s   <none>        <none>   <none>           <none>
controlplane $
```


20- Remove the taint on master/controlplane, which currently has the taint effect of NoSchedule

Command: kubectl taint nodes controlplane node-role.kubernetes.io/master-

21- What is the state of the pod mosquito now and Which node is the POD mosquito on?

State: Running

```
controlplane $ kubectl get po -o wide
NAME       READY   STATUS    RESTARTS   AGE   IP            NODE          NOMINATED NODE   READINESS GATES
bee        1/1     Running   0          27m   10.32.0.194   node01        <none>           <none>
mosquito   1/1     Running   0          [illegible]   10.32.0.2     controlplane  <none>           <none>
controlplane $
```