

Lab 3 – Kubernetes

Toka Abdelhakim

1- How many static pods exist in this cluster in all namespaces?

Terminal	+				
etcd-controlplane		1/1	Running	0	42m
katacoda-cloud-provider-5d779547bb-dv8jw		0/1	CrashLoopBackOff	13	43m
kube-apiserver-controlplane		1/1	Running	0	42m
kube-controller-manager-controlplane		1/1	Running	0	43m
kube-keepalived-vip-d9gpx		1/1	Running	0	43m
kube-proxy-5rfqh		1/1	Running	0	43m
kube-proxy-hh8nt		1/1	Running	0	43m
kube-scheduler-controlplane		1/1	Running	0	42m
weave-net-d2tq2		2/2	Running	1	43m
weave-net-rg9mq		2/2	Running	1	43m

2- On which nodes are the static pods created currently?

Master node -controlplane-

3- What is the path of the directory holding the static pod definition files?

/etc/Kubernetes/manifests

4- Create a static pod named static-busybox that uses the busybox image and the command sleep 1000

Yaml file:

```
Terminal +
apiVersion: v1
kind: Pod
metadata:
  name: static-busybox
  labels:
    app: nginx
spec:
  containers:
  - name: nginx-demo
    image: busybox
    command: ["sleep"]
    args: ["1000"]
```

Output:

```
controlplane $ kubectl get po
NAME                                READY   STATUS    RESTARTS   AGE
static-busybox-controlplane         1/1     Running   0           21s
controlplane $
```

5- Edit the image on the static pod to use busybox:1.28.4

```
Terminal +
apiVersion: v1
kind: Pod
metadata:
  name: static-busybox
  labels:
    app: nginx
spec:
  containers:
  - name: nginx-demo
    image: busybox:1.28.4
    command: ["sleep"]
    args: ["1000"]
```

6- How many ConfigMaps exist in the environment?

```
controlplane $ kubectl get configmaps
No resources found.
controlplane $
```

```
No resources found.
controlplane $ kubectl get configmaps -n kube-system
NAME                                DATA  AGE
coredns                             1      68m
extension-apiserver-authentication  6      68m
kube-proxy                           2      68m
kubeadm-config                       2      68m
kubelet-config-1.14                  1      68m
vip-configmap                        0      68m
weave-net                            0      68m
controlplane $
```

7- Create a new ConfigMap Use the spec given below ConfigName Name: webapp-config-map Data: APP_COLOR=darkblue

```
controlplane $ kubectl create configmap webapp-config-map --from-literal=APP_COLOR=darkblue
configmap/webapp-config-map created
controlplane $ kubectl get configmap
NAME            DATA  AGE
webapp-config-map 1      42s
controlplane $ kubectl describe configmap
Name:           webapp-config-map
Namespace:      default
Labels:         <none>
Annotations:    <none>

Data
====
APP_COLOR:
----
darkblue
Events:      <none>
controlplane $
```

Yaml file:

```
Terminal +
Port: <none>
Host Port: <none>
State: Running
  Started: Mon, 20 Sep 2021 16:26:25 +0000
Ready: True
Restart Count: 0
Environment Variables from:
  webapp-config-map ConfigMap Optional: false
Environment: <none>
Mounts:
  /var/run/secrets/kubernetes.io/serviceaccount from default-token-hhmnt (ro)
Conditions:
  Type          Status
  Initialized    True
  Ready          True
  ContainersReady True
  PodScheduled   True
Volumes:
```

One

Three

[illegible]

11- create a POD called db-pod with the image mysql:5.7 then check the POD status

```
web-pod.yaml x db-pod.yaml x
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: db-pod
5  spec:
6    containers:
7    - name: web
8      image: mysql:5.7
9
```

```
controlplane $ kubectl get po
NAME      READY   STATUS             RESTARTS   AGE
db-pod    0/1     CrashLoopBackOff   0           10s
web-app   1/1     Running            0           16m
controlplane $
```

12- why the db-pod status not ready?

Required secrets not found

13- Create a new secret named db-secret with the data given below.

Secret Name: db-secret

Secret 1: MYSQL_DATABASE=sqli01

Secret 2: MYSQL_USER=user1

Secret3: MYSQL_PASSWORD=password

Secret 4: MYSQL_ROOT_PASSWORD=password123

14- Configure db-pod to load environment variables from the newly created secret.

Delete and recreate the pod if required.

```
web-pod.yaml x db-pod.yaml x
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: db-pod
5  spec:
6    containers:
7    - name: web
8      image: mysql:5.7
9      envFrom:
10     - secretRef:
11       name: db-secret
12
```

```
controlplane $ kubectl get po
NAME      READY   STATUS    RESTARTS   AGE
db-pod    1/1     Running   0           6s
web-app   1/1     Running   0           19m
controlplane $
```

15- Create a multi-container pod with 2 containers.

Name: yellow

Container 1 Name: lemon

Container 1 Image: busybox

Container 2 Name: gold

Container 2 Image: redis

```
multic.yaml x
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: yellow
5    labels:
6      role: myrole
7  spec:
8    containers:
9      - name: lemon
10        image: busybox
11
12      - name: gold
13        image: redis
14
```

16- Create a pod red with redis image and use an initContainer that uses the busybox image and sleeps for 20 seconds

```
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: red
5    labels:
6      role: myrole
7  spec:
8    containers:
9      - name: lemon
10        image: redis
11
12    initContainers:
13      - name: mycont
14        image: busybox
15        command: ["sleep 20"]
16
```

17- Create a Persistent Volume with the given specification. Volume Name: pv-log
Storage: 100Mi Access Modes: ReadWriteMany Host Path: /pv/log

```
pod-with-configmap.y... x pod
1  apiVersion: v1
2  kind: PersistentVolume
3  metadata:
4    name: pv-log
5  spec:
6    accessModes:
7      - ReadWriteMany
8    capacity:
9      storage: 100Mi
10
11  hostPath:
12    path: "/pv/log"
13
```

```
controlplane $ kubectl get pv
NAME      CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS   CLAIM   STORAGECLASS  REASON   AGE
pv-log    100Mi     RWX           Retain          Available             100s
controlplane $
```

18- Create a Persistent Volume Claim with the given specification. Volume Name: claim-log1
Storage Request: 50Mi Access Modes: ReadWriteMany

```
pod-with-configmap.y... x pod.ya
1  apiVersion: v1
2  kind: PersistentVolumeClaim
3  metadata:
4    name: claim-log-1
5  spec:
6    accessModes:
7      - ReadWriteMany
8    resources:
9      requests:
10       storage: 50Mi
11
```

```
controlplane $ kubectl get pvc
NAME          STATUS  VOLUME  CAPACITY  ACCESS MODES  STORAGECLASS  AGE
claim-log-1   Bound   pv-log   100Mi     RWX           100Mi         5s
```

19- Create a webapp pod to use the persistent volume claim as its storage.

Name: webapp

Image Name: nginx

Volume: PersistentVolumeClaim=claim-log-1

Volume Mount: /var/log/nginx

```
pod.yaml  x  pv.yaml  x  pvc.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: webapp
5  spec:
6    containers:
7    - image: nginx
8      name: cont1
9      volumeMounts:
10     - mountPath: /var/log/nginx
11       name: data-volume
12  volumes:
13  - name: data-volume
14    persistentVolumeClaim:
15      claimName: claim-log-1
```

```
Terminal  +
Ready      True
ContainersReady  True
PodScheduled  True
Volumes:
data-volume:
  Type:      PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the same namespace)
  ClaimName: claim-log-1
  ReadOnly:  false
default-token-g8v9f:
  Type:      Secret (a volume populated by a Secret)
  SecretName: default-token-g8v9f
  Optional:  false
QoS Class:   BestEffort
Node-Selectors:  <none>
Tolerations:  node.kubernetes.io/not-ready:NoExecute for 300s
               node.kubernetes.io/unreachable:NoExecute for 300s
Activate Windows
Go to Settings to activate
```

```
controlplane $ kubectl get pvc
NAME          STATUS  VOLUME  CAPACITY  ACCESS MODES  STORAGECLASS  AGE
claim-log-1   Bound   pv-log   100Mi     RWX           default       12m
controlplane $ kubectl get pv
NAME          CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS  CLAIM
pv-log        100Mi     RWX           Retain          Bound   default/claim-log-1
controlplane $
```