ATMIC NETWORKS
YOUR GUIDE TO GLOBAL CERTIFIATION

**"Please note that these files may not be up to date. However, the questions will help you understand the exam format and typical question patterns."**

**www.atmicnetworks .com**

Warning: Keep connected with our support team for latest updates

Question: 1
SIMULATION

Monitor the logs of pod foo and:
Extract log lines corresponding to error unable-to-access-website
Write them to/opt/KULM00201/foo

Set configuration context:

Ktud^ric'fitode-ll * kube ctl config use-context k8s

Answer: See the solution below.

Explanation:

solution

SB Readme >_ Web Terminal

□ THE**LINUX** FOUNDATION

```
student 0node-1:~$ 3tudentgnode-l:~$ sudo -1 root@node-l:~# alias k=kubectl root@nodel:'i
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\1 B.JPG

```
root@node-l:-# k logs foo | grep unable-to-access-website
Thu Aug 27 05:25:28 UTC 2020 - ERROR - unableto-access-w«bsite
rootPnode-l:M k logs foo I grep unable-to-access-website > /opt/KULM00201/foo
root@node 1:'# IF:\Work\Data Entry Work\Data Entry\20200827\CKA\1 C.JPG
```

Step 0: Set the correct Kubernetes context

If you're given a specific context (k8s in this case), you must switch to it:

kubectl config use-context k8s

! Skipping this can cause you to work in the wrong cluster/namespace and cost you marks.

Step 1: Identify the namespace of the pod foo

First, check if foo is running in a specific namespace or in the default namespace.

kubectl get pods --all-namespaces | grep foo

Assume the pod is in the default namespace if no namespace is mentioned.

Step 2: Confirm pod foo exists and is running

kubectl get pod foo

You should get output similar to:

NAME READY STATUS RESTARTS AGE

foo 1/1 Running 0 1h

If the pod is not running, logs may not be available.

Step 3: View logs and filter specific error lines

We're looking for log lines that contain:

unable-to-access-website

Command:

kubectl logs foo | grep "unable-to-access-website"

Step 4: Write the filtered log lines to a file

Redirect the output to the required path:

kubectl logs foo | grep "unable-to-access-website" > /opt/KULM00201/foo

Q This creates or overwrites the file /opt/KULM00201/foo with the filtered logs.

You may need sudo if /opt requires elevated permissions. But in most exam environments, you're already the root or privileged user.

Step 5: Verify the output file (optional but smart)

Check that the file was created and has the correct content:

cat /opt/KULM00201/foo

Q Final Answer Summary:

kubectl config use-context k8s

kubectl logs foo | grep "unable-to-access-website" > /opt/KULM00201/foo

## Question: 2

SIMULATION

List all persistent volumes sorted by capacity, saving the full kubectl output to /opt/KUCC00102/volume_list. Use kubectl 's own functionality for sorting the output, and do not manipulate it any further.

Answer: See the solution below.

Explanation:

solution

• Readme >_ Web Terminal OTHELINUX FOUNDATION

```
77d

pv0007   7Gi        RWO            Recycle          Available          slow
    77d
pv0006   8Gi        RWO            Recycle          Available          slow
    77d
pv0003   10G1       RWO            Recycle          Available          slow
    77d
pv0002   HGi        RWO            Recycle          Available          slow
    77d
pvOOlO   13Gi       RWO            Recycle          Available          slow
    77d
pvOOl1   14Gi       RWO            Recycle          Available          slow
    77d
pvOOOl   16Gi       RWO            Recycle          Available          slow
    77d
pv0009   17G1       RWO            Recycle          Available          slow
    77d
pv0005   18Gi       RWO            Recycle           Available          slow
    77d
pv0008   19Gi       RWO            Recycle          Available          slow
    77d
pvOOOO   21Gi       RWO            Recycle          Available          slow
    77d

root@node-l:~# k get pv —sort-by=.spec.capacity.storage > /opt/KUCC00102/volume_list root8node-l:~#
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\2 C.JPG

## Question: 3

SIMULATION

Ensure a single instance of pod nginx is running on each node of the Kubernetes cluster where nginx also represents the Image name which has to be used. Do not override any taints currently in place. Use DaemonSet to complete this task and use ds-kusc00201 as DaemonSet name.

Answer: See the solution below.

Explanation:

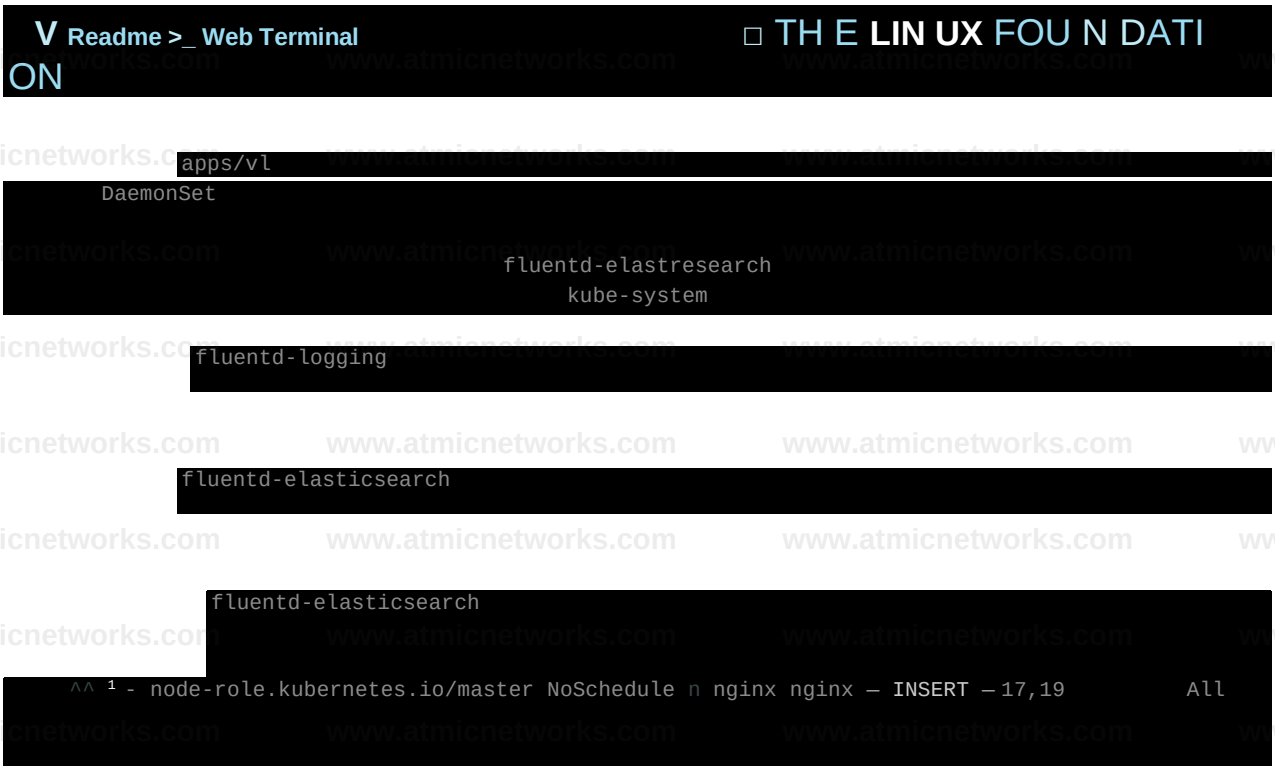
solution

• Readme >_ Web Terminal

□ THELINUX FOUNDATION

```
root@node-1:~# vim ds.yami
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\3 B.JPG




F:\Work\Data Entry Work\Data Entry\20200827\CKA\3 C.JPG

V Readme >_ Web Terminal □ THELINUX FOUNDATION

```
          apps/vl DaemonSei ram : ds-kusc00201 fluentd-elasticsearch fluentd-elasticsearch -
nginx i:      nginx
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\3 D.JPG

## Question: 4

SIMULATION

Perform the following tasks:

Add an init container to hungry-bear (which has been defined in spec file /opt/KUCC00108/pod-spec-KUCC00108.yaml)

The init container should create an empty file named/workdir/calm.txt

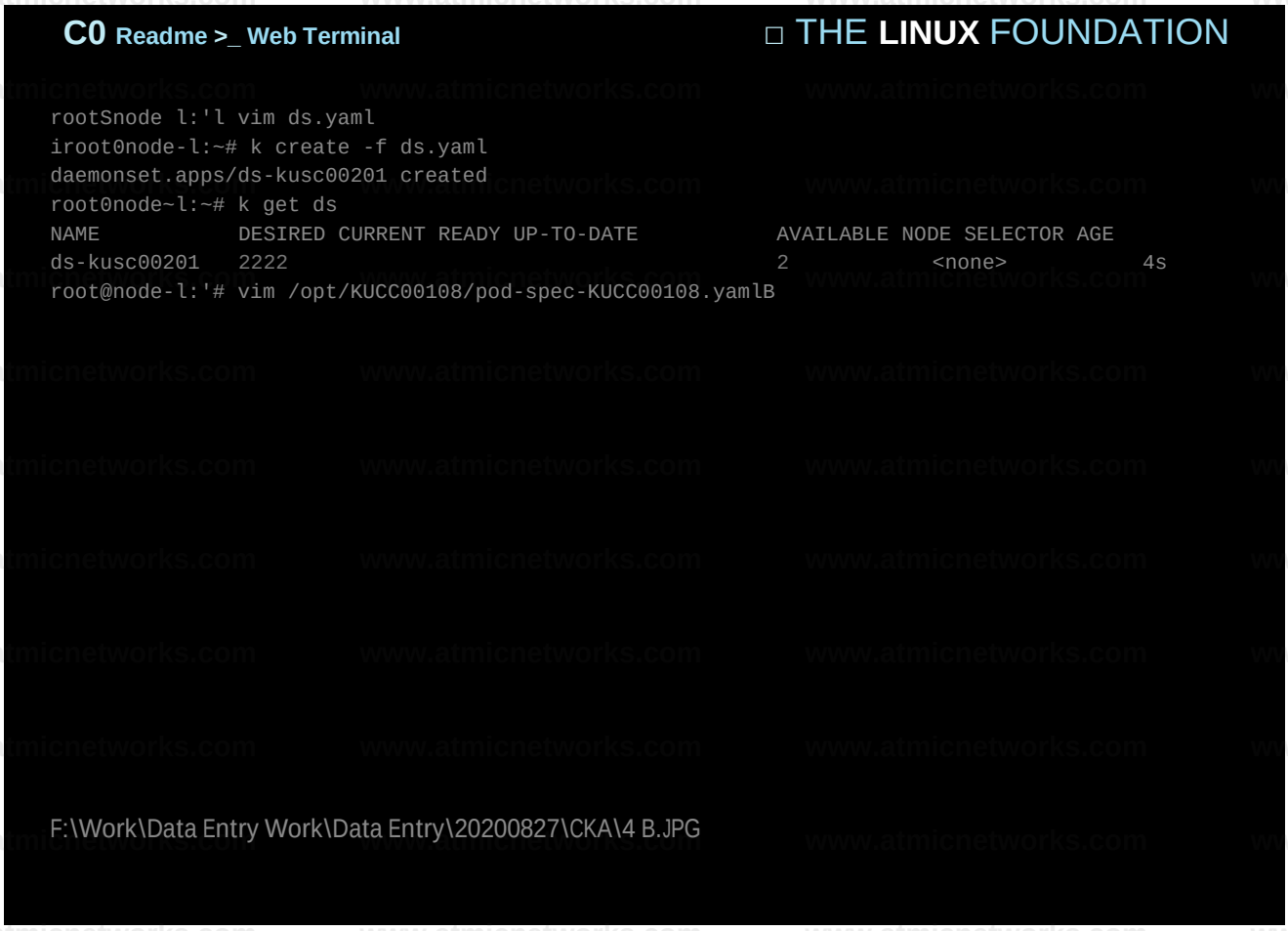
If /workdir/calm.txt is not detected, the pod should exit

Once the spec file has been updated with the init container definition, the pod should be created

Answer: See the solution below.

Explanation:

solution



• Readme >_ Web Terminal LITHE LINUX FOUNDATION

```
. i. vl
        Pod

hungry-bear

workdir

                                    ■ checker
                                      alpine

workdir
      /workdir

create
 alpine

workdir
      /workdir
:wq
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\4 C.JPG


F:\Work\Data Entry Work\Data Entry\20200827\CKA\4 D.JPG

## Question: 5

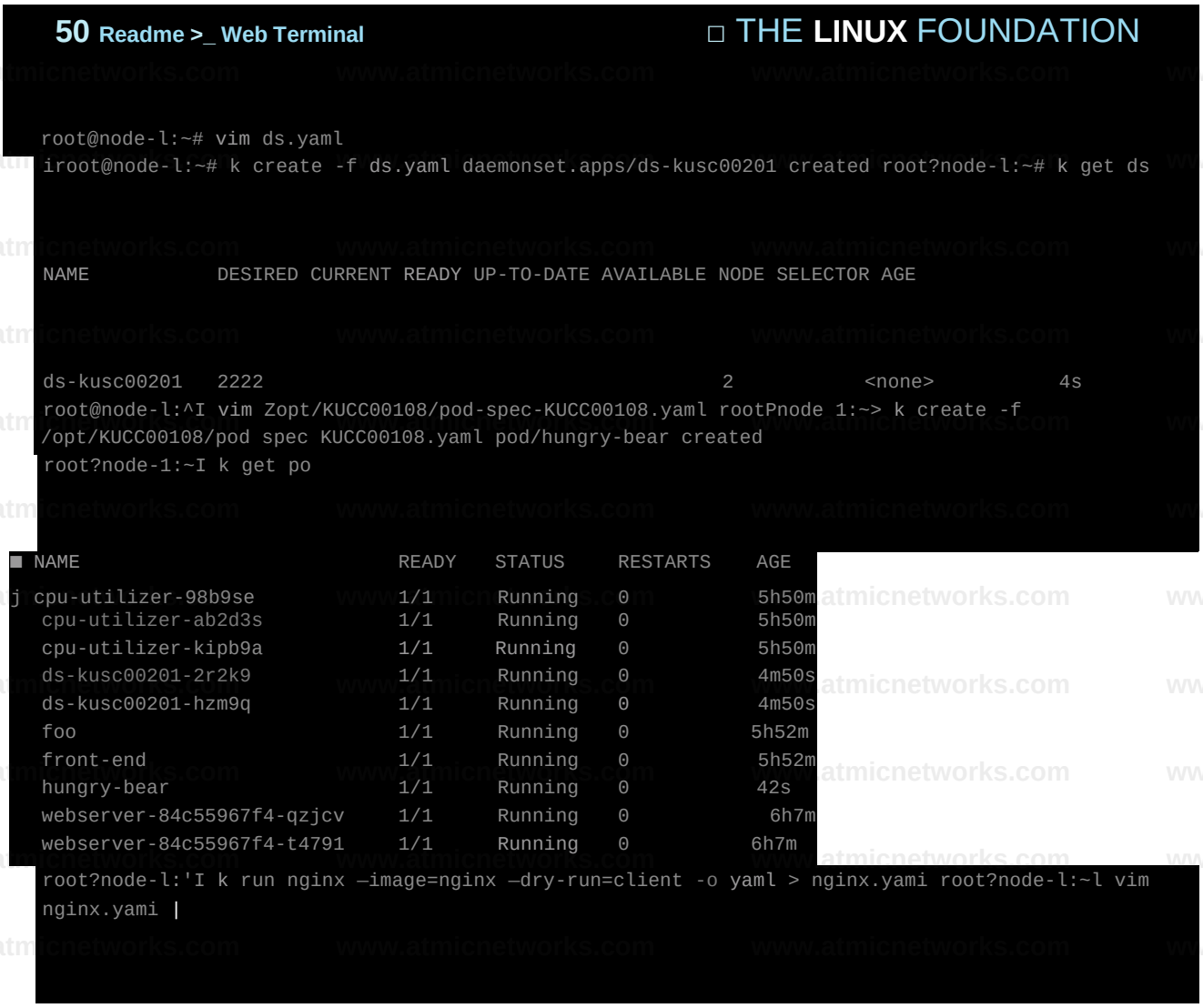
SIMULATION

Create a pod named kucc8 with a single app container for each of the following images running inside (there may be between 1 and 4 images specified): nginx + redis + memcached.

Answer: See the solution below.

Explanation:

solution




F:\Work\Data Entry Work\Data Entry\20200827\CKA\5 B.JPG

QB Readme >_ Web Terminal

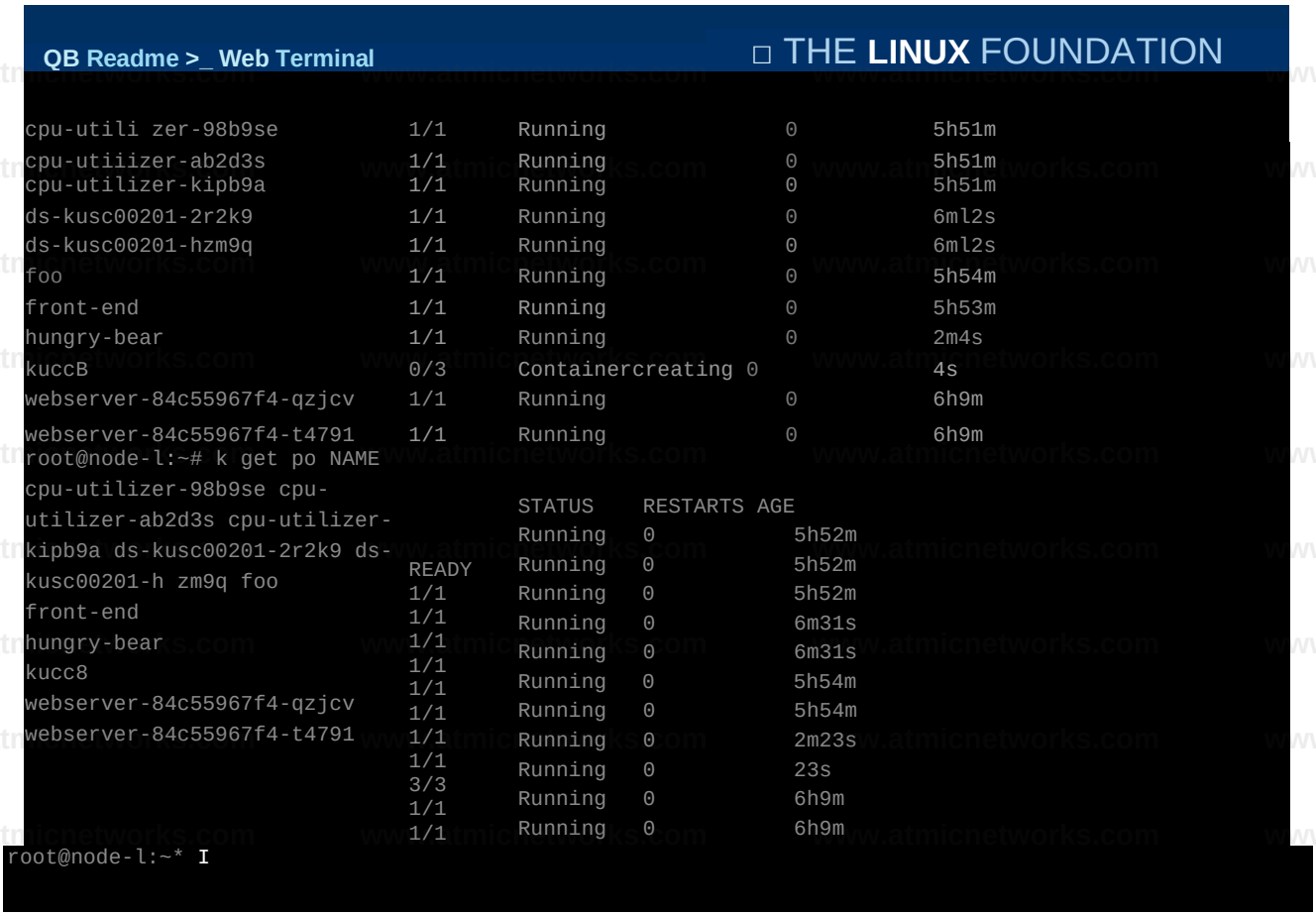
□ THE**LINUX** FOUNDATION

Pod

kucc8

nginx
nginx
redis
redis
memcached

1




F:\Work\Data Entry Work\Data Entry\20200827\CKA\5 D.JPG

## Question: 6

SIMULATION

Schedule a pod as follows: Name: nginx-kusc00101 Image: nginx
Node selector: disk=ssd

Answer: See the solution below.

Explanation:

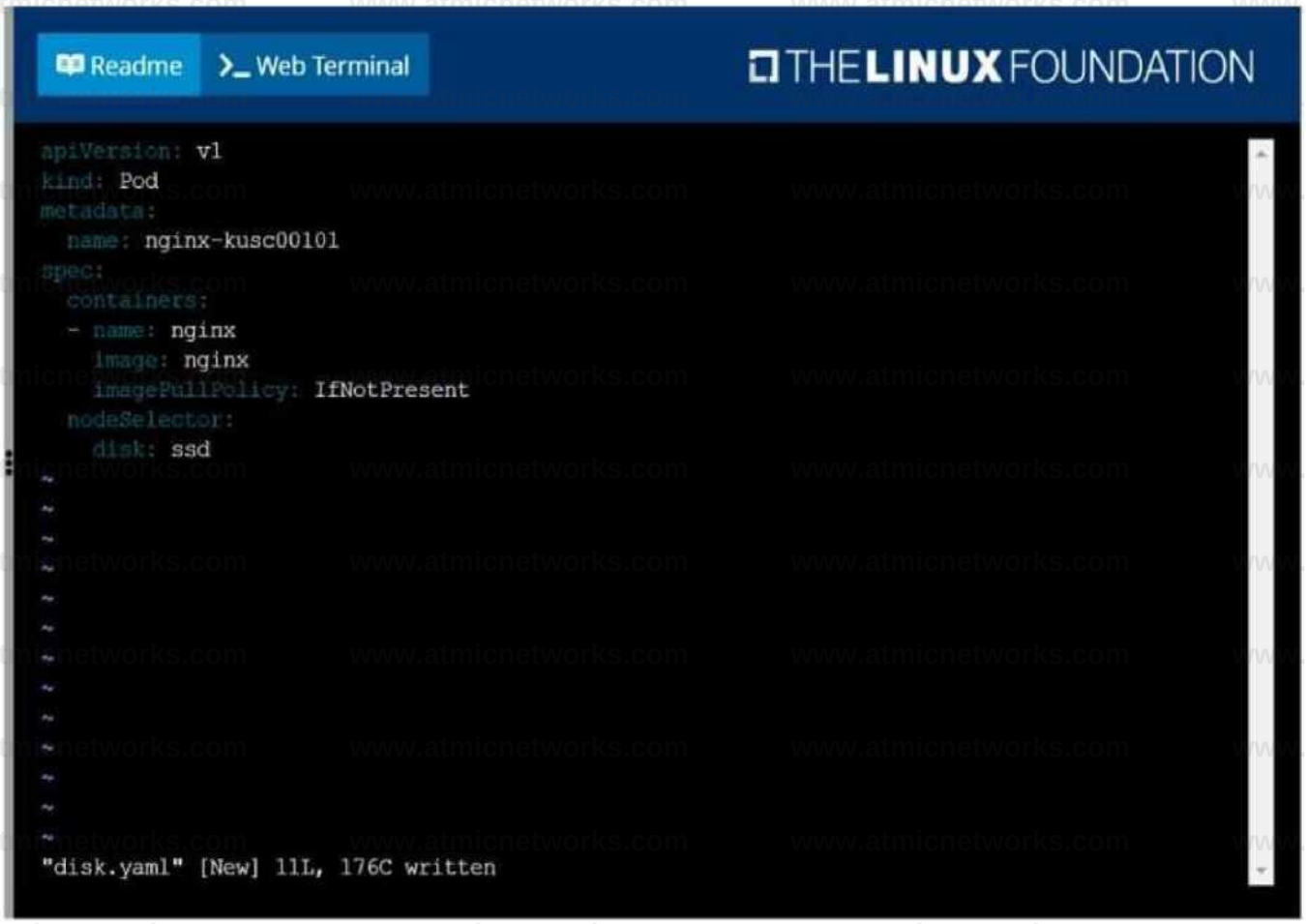
solution

99 Readme >_ Web Terminal

□ THELINUX FOUNDATION

```
root@node-1:--II vim
disk.yaml]
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\6 B.JPG




F:\Work\Data Entry Work\Data Entry\20200827\CKA\6 C.JPG

00 Readme >_ Web Terminal □ THE LINUX FOUNDATION

```
root@node-l:~# vim disk.yaml
root@node-l:~# k create -f disk.yaml
pod/nginx-kuscOOlOl created

root@node-l:~# k get po

NAME                          READY   STATUS    RESTARTS   AGE
cpu-utilizer-98b9se           1/1     Running   0          5h59m
cpu-utilizer-ab2d3s           1/1     Running   0          5h59m
cpu-utilizer-kipb9a           1/1     Running   0          5h59m
ds-kusc00201-2r2k9            1/1     Running   0          13m
ds-kusc00201-h zm9q           1/1     Running   0          13m
foo                           1/1     Running   0          6hlm
front-end                     1/1     Running   0          6hlm
hungry-bear                   1/1     Running   0          9m37s
kucc8                         3/3     Running   0          7m37s
nginx-kuscOOLOl               1/1     Running   0          9s
webserver-84c55967f4-qzjcv    1/1     Running   0          6hl6m
webserver-84c55967f4-t4791    1/1     Running   0          6hl6m

root@node-l:~f |
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\6 D.JPG

## Question: 7

SIMULATION

Create a deployment as follows:
Name: nginx-app
Using container nginx with version 1.11.10-alpine
The deployment should contain 3 replicas
Next, deploy the application with new version 1.11.13-alpine, by performing a rolling update.
Finally, rollback that update to the previous version 1.11.10-alpine.

Answer: See the solution below.

Explanation:

solution

00 Readme >_ Web Terminal □ THELINUX FOUNDATION

```
root@node-l:~# k create deploy nginx app —image=nginx:1.11.10-alpine -dry-run=client -o y> ami >
app.yami
root@node-l:~» vim app.yarn! |
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\7 B.JPG

00 Readme >_ Web Terminal

□ THELINUX FOUNDATION

```
apps/v1 Deployment
nginx-app
nginx-app
nginx-app
nginx:1.11.10-alpine nginx
'app.yaml
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\7 C.JPG

```
root@node-1:-I k create deploy nginx-app image=nginx:1.11.10-alpine        dry-run=cllent -o y B
ami > app.yaml
root@node-l:~l vim app.yaml
root@node 1:-# k create -f app.yaml
deployment.apps/nginx-app created
root@node-l:'#
root@node-1:~I
rootjnode-l:-# k set image deploy nginx-app nginx=nginx:1.11.13-alpine —record
deployment.apps/nginx-app image updated
root@node-l:~l k rollout undo deploy nginx-app
deployment.apps/nginx-app rolled back
root8node-l:-l
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\7 D.JPG

## Question: 8

SIMULATION

Create and configure the service front-end-service so it's accessible through NodePort and routes to the existing pod named front-end.

Answer: See the solution below.

Explanation:

solution

```
root@node-l:~# k expose po

error: resource(s) were provided, but no name, label selector, or –all flag specified

See 'kubectl expose -h' for help and examples
root@node-l:--# k expose po fron-end –name=trent-end-service –port–80 –target-port=80 –t I
ype=NodePort
Error from server (NotFound): pods "fron-end" not found
xoot@node-l:-I k expose po front-end –name-front-end-service –port=80 –target-port=80 – I type-
NodePort

service/front-end-service exposed
root?node-l;~# k get sve

NAME                TYPE        CLUSTER-IP          EXTERNAL-IP    PORT (S)        AGE
f ront-end-se rvice  NodePort    10.103.221.227      <none>         80:3182B/TCP      3s
kubernetes          ClusterIP   10.96.0.1           <none>         443/TCP         77d
root@node I:'P
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\8 B.JPG

## Question: 9

SIMULATION

Create a pod as follows:

Name: mongo

Using Image: mongo
In a new Kubernetes namespace named: my-website

Answer: See the solution below.

Explanation:


F:\Work\Data Entry Work\Data Entry\20200827\CKA\9 B.JPG

Question: 10

SIMULATION

Create a deployment spec file that will:

Launch 7 replicas of the nginx Image with the labelapp_runtime_stage=dev

deployment name: kual00201

Save a copy of this spec file to /opt/KUAL00201/spec_deployment.yaml

(or /opt/KUAL00201/spec_deployment.json).

When you are done, clean up (delete) any new Kubernetes API object that you produced during this task.

Explanation:

solution

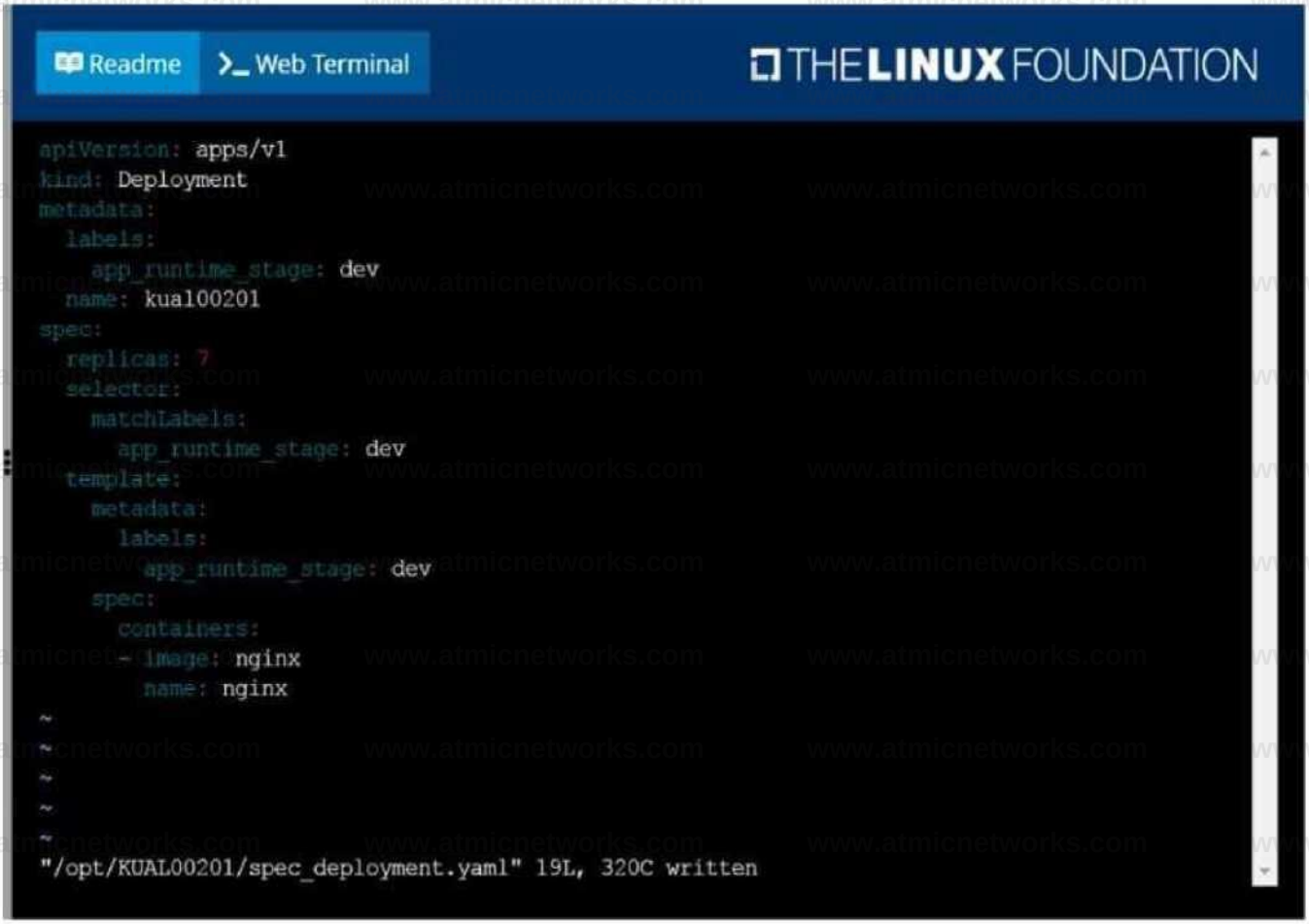
Answer: See the solution below.

09 Readme >_ Web Terminal

□ THELINUX FOUNDATION

```
rootSnode-k'l k create deploy kual00201 image=nginx dry run=cllent -o yaml > /opt/KUAL. B 00201/spec
deployment.yaml
root@node-l:~f vim /opt/KUAL00201/spec_deployment.yaml
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\10 B.JPG




F:\Work\Data Entry Work\Data Entry\20200827\CKA\10 C.JPG

## Question: 11

SIMULATION

Create a file:

/opt/KUCC00302/kucc00302.txt that lists all pods that implement service baz in namespace development.

The format of the file should be one pod name per line.

## Answer: See the solution below.

Explanation:

solution

```
root@node-1:~#
root@node-l:~S k  describe svc baz -n development
Name:              baz
Namespace:         development
Labels:            <none>
Annotations:       <none>
Selector:          name foo
Type:              ClusterIP
IP:                10.104.252.175
Port:              <unset> 80ZTCP
TargetPort:        9376ZTCP
Endpoints:         10.244.1.5:9376,10.244.2.3:9376,10.244.2.6:9376
Session Affinity:  None
Events:            <none>
root@node-l:*-l k get po -1 name=foo -n development
NAME                   READY STATUS       RESTARTS   AGE
pod-kucc00302-847878    1/1      Running   0          6h35m
pod-kucc00302-983457    1/1      Running   0          6h35m
pod-kucc00302-985953    1/1      Running   0          6h35m
root6node-l:~l k get   po -1  name=foo -ndevelopment  -o NAME
pod/pod-kucc00302-847878 pod/pod-kuccO0302-983457
```

**08 Readme >_ Web Terminal**

**LITHELINUX** FOUNDATION

```
  pod/pod-kucc00302-985953
  root@node-l:~# k get po -1 name-foo -n development -o NAME       > ZoptZKUCC00302Zkucc00302.txt
  root8node-l:~# vim /opt/KUCC00302Zkucc00302.txt]
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\11 B.JPG

QB Readme >_ Web Terminal

□THELINUX FOUNDATION

```
pod-kucc00302-847878
pod-kucc00302-983457
pod-kucc00302-985953
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\11 C.JPG


F:\Work\Data Entry Work\Data Entry\20200827\CKA\11 D.JPG

## Question: 12

SIMULATION

Create a Kubernetes secret as follows:

Name: super-secret

password: bob

Create a pod named pod-secrets-via-file, using the redis Image, which mounts a secret named supersecret at /secrets.

Create a second pod named pod-secrets-via-env, using the redis Image, which exports password as CONFIDENTIAL

Answer: See the solution below.

Explanation:

solution

```
root@node-1:-I
root6node-l:~l k create secret generic super-secret —from-literal=password=bob secret/super-secret
created
root@node 1:~# vim secret.yami]
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\12 B.JPG

- Readme >_ Web Terminal □THELINUX FOUNDATION

```
vl
Xi J: Pod

  pod-secrets-viatile

-  . redis redis

-  an: foo too

super-secret
```

:w F:\Work\Data Entry Work\Data Entry\20200827\CKA\12 C.JPG

SB Readme >_ Web Terminal LITHELINUX FOUNDATION

```
root@node-l:~# k create -f secret.yaml pod/pod-secrets-via-file created root@node-l:~8 vim
secretl.yaml root@node-l:~i k create -f secretl.yaml pod/pod-secrets-via-env created root@node-l:~#
k get po
NAME                          READY    STATUS     RESTARTS    AGE
cpu-utilizer-98b9se           1/1      Running    0           6h25m
cpu -utilizer-ab2d3s          1/1      Running    0           6h25m
cpu-utili zer-kipb9a          1/1      Running    0           6h25m
ds-kusc00201-2r2k9            1/1      Running    0           40m
ds-kusc00201-hzm9q            1/1      Running    0           40m
foo                           1/1      Running    0           6h28m
front-end                     1/1      Running    0           6h27m
hungry-bear                   1/1      Running    0           36m
kuccB                         3/3      Running    0           34m
nginx-app-S48cfc£495-9prjh    1/1      Running    0           19m
nglnx-app-848cfcf495-gl2kh    l/l      Running    0           19m
nglnx-app-848cfcf495-pg2c8    1/1      Running    0           19m
nginx-kuscOOlOl               1/1      Running    0           26m
pod-secrets-via env           1/1      Running    0           4s
pod-secrets-via-file          1/1      Running    0           106s
webserver-84c55967f4-qzjcv    1/1      Running    0           6h43m
webserver-84c55967f4-t4791    1/1      Running    0           6h43m
rootBnode-l:~l £
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\12 D.JPG

## Question: 13

SIMULATION

Create a pod as follows:
Name: non-persistent-redis
container Image: redis
Volume with name: cache-control
Mount path: /data/redis
The pod should launch in the staging namespace and the volume must not be persistent.

Answer: See the solution below.

Explanation:

solution

SB Readme >_ Web Terminal

□ THE LINUX FOUNDATION


F:\Work\Data Entry Work\Data Entry\20200827\CKA\13 B.JPG

SB Readme >_ Web Terminal □ THELINUX FOUNDATION

```
vl _ Pod non-persistent-redis i      staging
```

```
redis redis cache-control I : /data/redis cache -cont rol
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\13 C.JPG

• Readme >_ Web Terminal □ THELINUX FOUNDATION

```
root@node-1:'# root@node-l:'l root@node-l:~# vim volume.yaml root@node 1:-# k create -f volume.yarn!
pod/non-persistent-redis created root@node-l:-# k get po -n staging NAME READY STATUS   RESTARTS
                    AGE
non-persistent-redis 1/1        Running 0             6s


root@node-1:'# |
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\13 D.JPG

## Question: 14

SIMULATION

Scale the deployment webserver to 6 pods.

Answer: See the

Explanation:

solution


F:\Work\Data Entry Work\Data Entry\20200827\CKA\14 B.JPG

## Question: 15

SIMULATION

Check to see how many worker nodes are ready (not including nodes tainted NoSchedule) and write the number to /opt/KUCC00104/kucc00104.txt.

Answer: See the solution below.

Explanation:

```
root@node-l:~# k scale deploy webserver --replicas=6
deployment.apps/webserver scaled
root0node-l:~l k get deploy
NAME        READY   UP-TO-DATE   AVAILABLE   AGE
nginx-app 3/3       3            3           29m
webserver 6/6       6            6           6h50m
root@node-l:-I
root@node-l:~# k get nodes
NAME           STATUS   ROLES    AGE     VERSION
k8s-master-0   Ready    master   77d     Vl.18.2
k8s-node-0     Ready    <none>   77d     vl.18.2
k8s-node-l     Ready    <none>   77d     vl.18.2
root@node-l:~# vim /opt/KUCC00104/kucc00104.txtl
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\15 B.JPG


F:\Work\Data Entry Work\Data Entry\20200827\CKA\15 C.JPG

## Question: 16

SIMULATION

From the pod label name=cpu-utilizer, find pods running high CPU workloads and write the name of the pod consuming most CPU to the file /opt/KUTR00102/KUTR00102.txt (which already exists).

Answer: See the solution below.

Explanation:

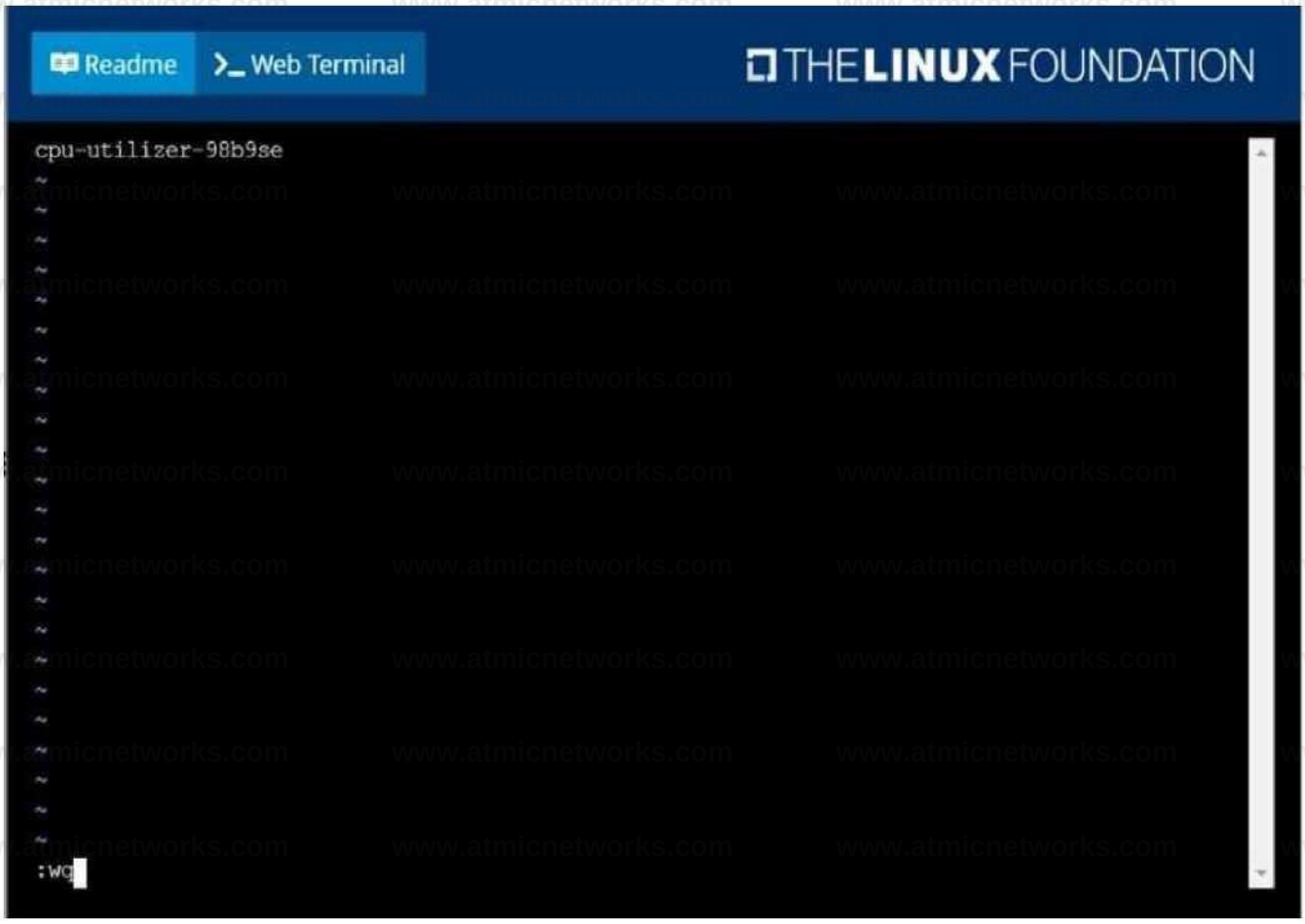
solution

```
root @ node 1:--(I k  po -1 name= cpu utilizer
NAME                    CPU(cores)    MEMORY(bytes)
cpu-ut11i zer-98b 9se   60m           7Mi
cpuutilizer ab2d3s      14m           7Mi
cpu-ut11i ze r-k        45m           7Mi
root0node-l:~# vim /opt/KUTROO102/KUTR00102.txt
```

• Readme >_ Web Terminal

□ **THELINUX** FOUNDATION

F:\Work\Data Entry Work\Data Entry\20200827\CKA\16 B.JPG




F:\Work\Data Entry Work\Data Entry\20200827\CKA\16 C.JPG

## Question: 17

SIMULATION

Create a deployment as follows:
Name: nginx-random
Exposed via a service nginx-random
Ensure that the service and pod are accessible via their respective DNS records
The container(s) within any pod(s) running as a part of this deployment should use the nginx Image Next, use the utility nslookup to look up the DNS records of the service and pod and write the output to /opt/KUNW00601/service.dns and /opt/KUNW00601/pod.dns respectively.

Answer: See the solution below.

Explanation:

Solution:

```
vl
iri: Pod
;   busybox1
busybox
busybox:1.28
- sleep
Busybox
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\17 D.JPG

M Readme >_ Web Terminal □ THELINUX FOUNDATION

```
root@node-1:-8 k create deploy nginx-random image=nginx

deployment.apps/nginx-random created
root@node-l:~8 k expose deploy nginx-random —name-nginx-random —port=80 —target-port=80
service/nginx-random exposed


root@node-1:~8 vim dns.yaml
root@node-1:-8 k create -f dns.yaml
pod/busyboxi created
root@node-1:-8 k get po -o wide | grep nginx-random
nginx-random-6d5766bbdc-ptzv2 1/1 Running 0                    103s    10.244.2.16 k8s-node-
1   <none>           <none>
root8node-l:~# k exec -it busyboxl — nslookup nginx-random
Server:    10.96.0.10
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local

Name:      nginx-random
Address 1: 10.111.37.132 nginx-random.default.svc.cluster.local
root@node-l:'8 k exec -it busyboxl — nslookup nginx-random > /opt/KUNW00601/service.dns
root0node-l:~8 k exec -it busyboxl — nslookup 10-244-2-16.default.pod
Server:    10.96.0.10              —
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local

Name:      10-244-2-16.default.pod
Address 1: 10.244.2.16 10-244-2-16.nginx-random.default.svc.cluster.local
root@node-l:-# k exec -it busyboxl — nslookup 10-244-2-16.default.pod > /opt/KUNW00601/pod
.dns|
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\17 E.JPG

## Question: 18

SIMULATION

Create a snapshot of the etcd instance running at https://127.0.0.1:2379, saving the snapshot to the file path /srv/data/etcd-snapshot.db.

The following TLS certificates/key are supplied for connecting to the server with etcdctl:

CA certificate: /opt/KUCM00302/ca.crt

Client certificate: /opt/KUCM00302/etcd-client.crt

Client key: Topt/KUCM00302/etcd-client.key

Answer: See the solution below.

Explanation:

solution



```
root@node 1:~# ETCDCTL API=3 etcdctl endpoints=https://127.0.0.1:2379 cacert=/opt/KUO40
0302/ca.crt —cert=/opt/KUCM00302/etcd-client.crt —key-/opt/KUCM00302/etcd-client.key an apshot save
/srv/data/etcd-snapshot.db
("level":"info","ts":1598530470.8313155,"caller":"snapshot/v3_snapshot.go:110","msg":"create


d temporary db file","path":"/srv/data/etcd-snapshot.db.part")
{"level":"warn","ts":"2020-08-27T12:14:30.838Z","caller":"clientv3/retry_interceptor.go:116"
,"msg":"retry stream intercept")
("level":"info","ts":1598530470.8388612,"caller":"snapshot/v3_snapshot.go:121","msg":"fetchi
ng snapshot","endpoint":"https://127.0.0.1:2379")
("level":"info","ts":1598530470.8570414,"caller":"snapshot/v3 snapshot.go:134","msg":"fetche
d snapshot","endpoint":"https://127.0.0.1:2379","took":0.025676157)
("level":"info","ts":1598530470.8571067,"caller":"snapshot/v3_snapshot.go:143","msg":"saved"
,"path":"/srv/data/etcd-snapshot.db")
Snapshot saved at /srv/data/etcd-snapshot.db
root6node-l:~l I
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\18 C.JPG

## Question: 19

SIMULATION

Set the node named ek8s-node-1 as unavailable and reschedule all the pods running on it.

Answer: See the solution below.

Explanation:

solution

```
root@node-l:~l kubectl config use-context ek8s Switched to context "ek8s".


rootGnode-1:~H k drain ek8s-node-l —ignore-daemonsets --delete-local-data —force node/ek8s-node-l
cordoned
WARNING: ignoring DaemonSet-managed Pods: kube-system/kube-flannel-ds-amd64-qj7w8, kube-syst I
em/kube-proxy x7xkv
evicting pod default/nginx-568f5649b8-c9zkj


evicting pod kube system/metrics-server 64b57fd654 cktk5
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\19 B.JPG

## Question: 20

SIMULATION

A Kubernetes worker node, named wk8s-node-0 is in state NotReady. Investigate why this is the case, and perform any appropriate steps to bring the node to a Ready state, ensuring that any changes are made permanent.

You can ssh to the failed node using:

[student@node-1] $ | ssh Wk8s-node-0

You can assume elevated privileges on the node with the following command: [student@w8ks-node-0] $ | sudo –i

Answer: See the solution below.

Explanation:

Solution

9 Readme >_ Web Terminal | Wk8s | □ THE LINUX FOUNDATION

root@node-1:~# kubectl config use-context Switched to context "wk8s".

```
root@node-l:-  k get nodes
NAME            STATUS     ROLES     AGE   VERSION
wk8s-master-0   Ready      master    77(   vl.18.2
wk8s-node-0     NotReady   <none>    77(   vl.18.2
wk8s-node-l     Ready      <none>    77(   vl.18.2
root@node-l:-  ssh wk8s-node-0
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\20 C.JPG


F:\Work\Data Entry Work\Data Entry\20200827\CKA\20 D.JPG

```
    https://microk8s.lo/ has docs and details.

4  packages can be updated.
1  update is a security update.

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

student.@wk8s-node-0:'$ sudo -i
root@wk8s-node-0:'I systemctl restart kubelet root?wk8s-node-0:*l systemctl enable kubelet Created
symlink from /etc/systemd/system/multi-user.target.wants/kubelet.service to /lib/sys
temd/system/kubelet.service. root@wk8s-node-0:~l exit logout student@wk8s-node-0:~$ exit logout
Connection to 10.250.5.34 closed.
root?node-1 :~l k get nodes

NAME            STATUS    ROLES     AGE  VERSION
wk8s-master-0   Ready     master    77d  vl.18.2
wk8s-node-0     Ready     <none>    77d  vl.18.2
wk8s-node-l     Ready     <none>    77d  vl.18.2
root£node-l:~l £
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\20 E.JPG

## Question: 21

SIMULATION

Configure the kubelet systemd- managed service, on the node labelled with name=wk8s-node-1, to launch a pod containing a single container of Image httpd named webtool automatically. Any spec files required should be placed in the /etc/kubernetes/manifests directory on the node.

You can ssh to the appropriate node using:

[student@node-1] $ ssh wk8s-node-1

You can assume elevated privileges on the node with the following command: [student@wk8s-node-1] $ | sudo –i

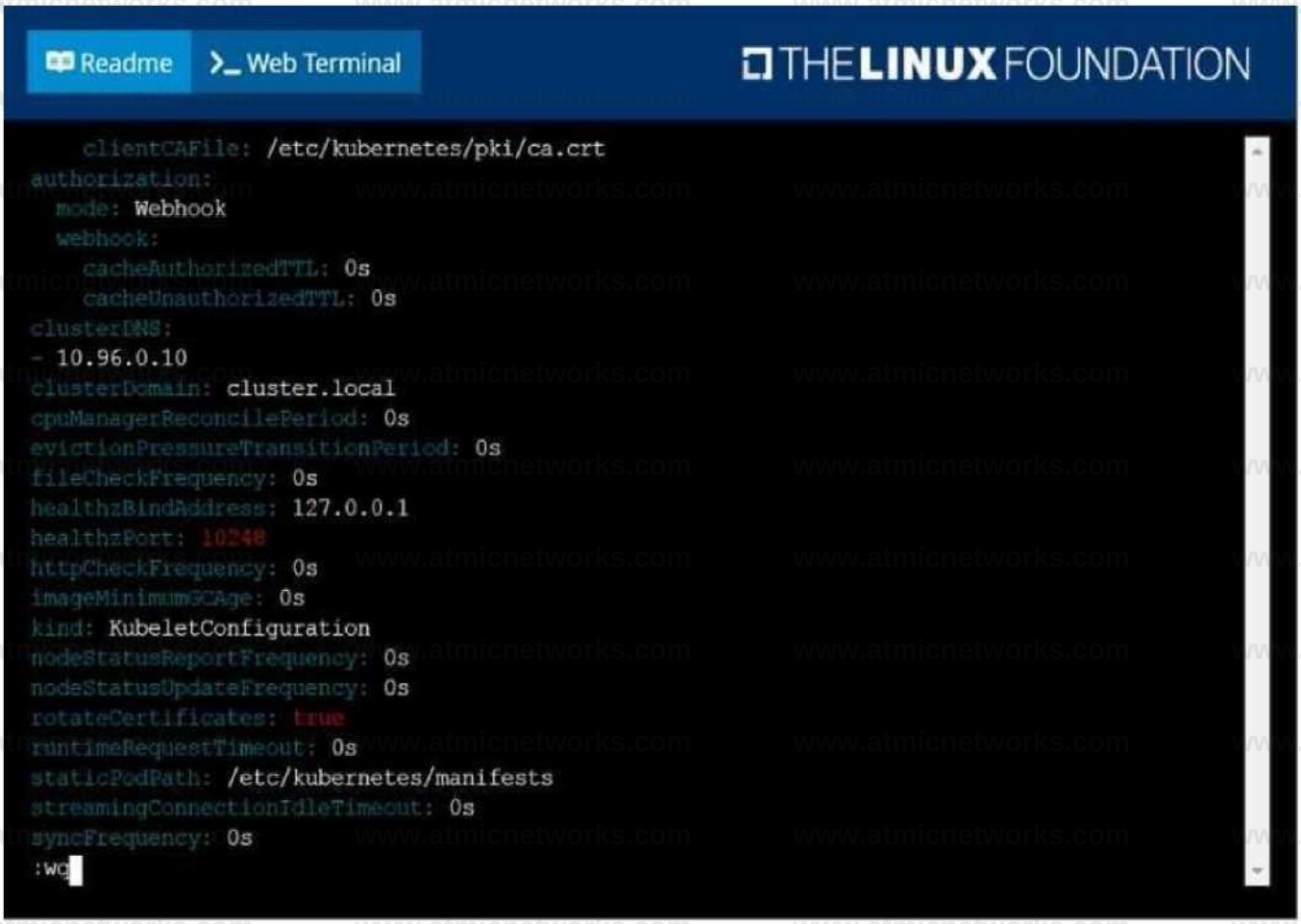
Answer: See the solution below.

Explanation:

solution

```
root8node-l:~#

rootfinode-l:*# kubectl config use-context wk8s Switched to context "wk8s".
rootSnode-l:*# ssh wkfls-node-l

Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1109-aws x86 64)

 * Documentation: https://help.ubuntu.com

 * Management:      https://landscape.canonical.com

 * Support:         https://ubuntu.com/advantage

* Are you ready for Kubernetes 1.19? It's nearly here! Try RC3 with sudo snap install microkSs —
  channel=1.19/candidate —classic

   https://microkfls.io/ has docs and details.

4  packages can be updated.
1  update is a security update.

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


student@wk8s-node-l:~$ sudo -i
root@wk8s-node-l:~# vim /var/lib/kubelet/config.yaml
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\21 C.JPG




F:\Work\Data Entry Work\Data Entry\20200827\CKA\21 D.JPG

M Readme >_ Web Terminal □ THELINUX FOUNDATION

```
root@node-l:~# ssh wk8s-node-l
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1109-aws x86 64)

 * Documentation: https://help.ubuntu.com
 * Management:      https://landscape.canon!cal.com

 * Support:         https://ubuntu.com/advantage

* Are you ready for Kubernetes 1.19? It's nearly here! Try RC3 with sudo snap install microk8s --
  channel-1.19/candidate —classic

   https://microk8s.io/ has docs and details.

4 packages can be updated.
1 update is a security update.

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


student@wk8s-node-l:~$ sudo -i
root?wk8s-node-l:~i vim /var/lib/kubelet/config.yaml
root@wk8s-node-l:~l cd /etc/kubernetes/manifests root@wk8s-node-l:/etc/kubernetes/manifests!
root@wk8s-node-l:/etc/kubernetes/manifestsi vim pod.yaml
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\21 E.JPG


F:\Work\Data Entry Work\Data Entry\20200827\CKA\21 F.JPG

```
    https://microk8s.io/ has docs and details.
4 packages can be updated.
1  update is a security update.

New release '18.04.5 LTS' available.
Run 'do-release-upgrade’ to upgrade to it.

student@wk8s-node-l:”$ sudo -i

root@wk8s-node-l:~# vim /var/lib/kubelet/config.yaml
root@wk8s-node-l:~l cd /etc/kubernetes/manifests r oot(?wk8s-node-1:/etc/kubernetes/mani fests t
root@wk8s-node-l:/etc/kubernetes/manifestsj vim pod.yaml root@wk8s-node-
l:/etc/kubernetes/manifestsl systemctl restart kubelet root@wk8s-node-l:/etc/kubernetes/manifestsl
systemctl enable kubelet root@wk8s-node-l:/etc/kubernetes/manifests# exit logout student@wk8s-node-
l:~$ exit logout
Connection to 10.250.5.39 closed.
root@node-l:~# k get po
NAME                READY    STATUS    RESTARTS AGE
webtool-wk8s-node-l    1/1      Running    0          Ils
root@node-l:~l |
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\21 G.JPG

## Question: 22

SIMULATION

For this item, you will have to ssh to the nodes ik8s-master-0 and ik8s-node-0 and complete all tasks on these nodes. Ensure that you return to the base node (hostname: node-1) when you have completed this item.

### Context

As an administrator of a small development team, you have been asked to set up a Kubernetes cluster to test the viability of a new application.

### Task

You must use kubeadm to perform this task. Any kubeadm invocations will require the use of the --ignore-preflight-errors=all option.

Configure the node ik8s-master-O as a master node. .

Join the node ik8s-node-o to the cluster.

Answer: See the solution below.

Explanation:

solution

You must use the kubeadm configuration file located at /etc/kubeadm.conf when initializing

your cluster.

You may use any CNI plugin to complete this task, but if you don't have your favourite CNI plugin's manifest URL at hand, Calico is one popular option: https://docs.projectcalico.org/v3.14/manifests/calico.yaml
Docker is already installed on both nodes and apt has been configured so that you can install the required tools.

## Question: 23

SIMULATION

Given a partially-functioning Kubernetes cluster, identify symptoms of failure on the cluster. Determine the node, the failing service, and take actions to bring up the failed service and restore the health of the cluster. Ensure that any changes are made permanently.

You can ssh to the relevant I nodes (bk8s-master-0 or bk8s-node-0) using: [student@node-1] $ ssh <nodename>
You can assume elevated privileges on any node in the cluster with the following command: [student@nodename] $ | sudo –i

Answer: See the solution below.

Explanation:

solution

```
root^node-1:~#
root@node-l:~S kubectl config use-context bk8s Switched to context "bk8s".
root@node-l:~# ssh bk8s-master-0


Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1109-aws x86_64)

 * Documentation: https://help.ubuntu.com
 ‘ Management:       https://landscape.canonical.com


 * Support:        https://ubuntu.com/advantage

 * Are you ready for Kubernetes 1.19? It's nearly here! Try RC3 with sudo snap install microkSs —
   channel=1.19/candidate —classic

    https://microk8s.io/ has docs and details.

4 packages can be updated.
1  update is a security update.

New release '18.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

student0bk8s-master-O:~$ sudo -i
rootgbk8s-master-0:~4 vim /var/lib/kubelet/config.yaml |
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\23 C.JPG


Readme
Web Terminal
THE LINUX FOUNDATION
authorization:
mode: Webhook
webhook:
cacheAuthorizedTTL: 0s
cacheUnauthorizedTTL: 0s
clusterDNS:
- 10.96.0.10
clusterDomain: cluster.local
cpuManagerReconcilePeriod: 0s
evictionPressureTransitionPeriod: 0s
fileCheckFrequency: 0s
healthzBindAddress: 127.0.0.1
healthzPort: 10248
httpCheckFrequency: 0s
imageMinimumGCAge: 0s
kind: KubeletConfiguration
nodeStatusReportFrequency: 0s
nodeStatusUpdateFrequency: 0s
rotateCertificates: true
runtimeRequestTimeout: 0s
staticPodPath: /etc/kubernetes/manifests
streamingConnectionIdleTimeout: 0s
syncFrequency: 0s
volumeStatsAggPeriod: 0s
:wq

F:\Work\Data Entry Work\Data Entry\20200827\CKA\23 D.JPG

00 Readme >_ Web Terminal □THE LINUX FOUNDATION

```
   https://microk8s.io/ has docs and details.

 4 packages can be updated.
 1  update is a security update.

 New release *18.04.5 LTS* available.
 Run 'do-release-upgrade' to upgrade to it.
```

```
student0bk8s-master-O:~$ sudo -i
root0bk8s-master-0:-♦ vim /var/lib/kubelet/config.yaml
root0bk8s-master-O:~i systemctl restart kubelet
root0bk8s-master-O:~i systemctl enable kubelet
root0bk8s-master-O:-i kubectl get nodes
```

| NAME | STATUS | ROLES | AGE | VERSION |
|---|---|---|---|---|
| bk8s-master-0 | Ready | master | 77d | vl.18.2 |
| bk8s-node-0 | Ready | <none> | 77d | vl .18.2 |

```
root@bk8smaster-0:~4
root@bk8s-master-0:~i exit
logout
student@bk8s-master-0:~$ exit logout
Connection to 10.250.4.77 closed.
root0node-1:'I I
```

F:\Work\Data Entry Work\Data Entry\20200827\CKA\23 E.JPG

## Question: 24

SIMULATION

Create a persistent volume with name app-data, of capacity 2Gi and access mode ReadWriteMany. The type of volume is hostPath and its location is /srv/app-data.

Answer: See the solution below.

Explanation:

solution

Persistent Volume

A persistent volume is a piece of storage in a Kubernetes cluster. PersistentVolumes are a clusterlevel resource like nodes, which don't belong to any namespace. It is provisioned by the administrator and has a particular file size. This way, a developer deploying their app on Kubernetes need not know the underlying infrastructure. When the developer needs a certain amount of persistent storage for their application, the system administrator configures the cluster so that they consume the PersistentVolume provisioned in an easy way.

Creating Persistent Volume

kind: PersistentVolume
apiVersion: v1
metadata:

name:app-data

spec:

capacity: # defines the capacity of PV we are creating

storage: 2Gi #the amount of storage we are tying to claim

accessModes: # defines the rights of the volume we are creating

- ReadWriteMany

hostPath:

path: "/srv/app-data" # path to which we are creating the volume

## Challenge

Create a Persistent Volume named app-data, with access mode ReadWriteMany, storage classname shared, 2Gi of storage capacity and the host path /srv/app-data.


2. Save the file and create the persistent volume.

```
njerry1910cloud^tiell:- (extreme-clone-265411)$ kubectl create -f pv.yaml persistentvolume/pv created
```

Image for post

3. View the persistent volume.

```
njcrryl91Pcloydshcll:~ (cxtreac-clone-265411)? kubcctl get pv
NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECIASS REASON AGE
app-data 2Gi RWX Retain Available shared 31s
```

Our persistent volume status is available meaning it is available and it has not been mounted yet. This status will change when we mount the persistentVolume to a persistentVolumeClaim. PersistentVolumeClaim

In a real ecosystem, a system admin will create the PersistentVolume then a developer will create a PersistentVolumeClaim which will be referenced in a pod. A PersistentVolumeClaim is created by specifying the minimum size and the access mode they require from the persistentVolume.

## Challenge

Create a Persistent Volume Claim that requests the Persistent Volume we had created above. The claim should request 2Gi. Ensure that the Persistent Volume Claim has the same storageClassName as the persistentVolume you had previously created.

kind: PersistentVolume apiVersion: v1 metadata: name:app-data spec:
accessModes:
- ReadWriteMany resources: requests:
storage: 2Gi storageClassName: shared

2. Save and create the pvc

njerry191@cloudshell:~ (extreme-clone-2654111)$ kubect1 create -f app-data.yaml persistentvolumeclaim/app-data created

3. View the pvc

```
njerryl91Qcloudshell:~ (extreme-clone-265411)$ kubectl get pvc NAME
       STATUS    VOLUME    CAPACITY    ACCESS   MODES STORAGECLASS
pv     Bound     pv        512m        RWX              shared
```

Image for post

4. Let's see what has changed in the pv we had initially created.

```
njerryl91@cloudshell:~ (extreme-clone-265411)$ kubectl get pv
NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM                    STORAGECLASS REASON AGE
pv 512m              RWX             Retain          Bound default/pv shared 16m
```

Image for post

Our status has now changed from available to bound.

5. Create a new pod named myapp with image nginx that will be used to Mount the Persistent Volume Claim with the path /var/app/config.

Mounting a Claim

apiVersion: v1
kind: Pod
metadata:
creationTimestamp: null
name: app-data
spec:
volumes:
- name:congigpvc
persistenVolumeClaim:
claimName: app-data containers:
- image: nginx
name: app
volumeMounts:
- mountPath: "/srv/app-data " name: configpvc

## Question: 25

SIMULATION

Create a namespace called 'development' and a pod with image nginx called nginx on this namespace.

Answer: See the solution below.

Explanation:

kubectl create namespace development
kubectl run nginx --image=nginx --restart=Never -n development

## Question: 26

SIMULATION

Create a nginx pod with label env=test in engineering namespace

Answer: See the solution below.

Explanation:

kubectl run nginx --image=nginx --restart=Never --labels=env=test --namespace=engineering --dryrun -o yaml > nginx-pod.yaml
kubectl run nginx --image=nginx --restart=Never --labels=env=test --namespace=engineering --dryrun -o yaml | kubectl create -n engineering -f –

YAML File:

```
apiVersion: v1
kind: Pod
metadata:
name: nginx
namespace: engineering
labels:
env: test
spec: containers:
- name: nginx

image: nginx
imagePullPolicy: IfNotPresent
restartPolicy: Never
```

kubectl create -f nginx-pod.yaml

## Question: 27

SIMULATION

Get list of all pods in all namespaces and write it to file "/opt/pods-list.yaml"

Answer: See the solution below.

Explanation:

kubectl get po –all-namespaces > /opt/pods-list.yaml

## Question: 28

SIMULATION

Create a pod with image nginx called nginx and allow traffic on port 80

Answer: See the solution below.

Explanation:

kubectl run nginx --image=nginx --restart=Never --port=80

## Question: 29

SIMULATION

Create a busybox pod that runs the command “env” and save the output to “envpod” file

Answer: See the solution below.

Explanation:

kubectl run busybox --image=busybox --restart=Never —rm -it -- env > envpod.yaml

## Question: 30

SIMULATION

List pod logs named “frontend” and search for the pattern “started” and write it to a file “/opt/error- logs”

Answer: See the solution below.

Explanation:

Kubectl logs frontend | grep -i “started” > /opt/error-logs

## Question: 31

SIMULATION

Create a pod that echo “hello world” and then exists. Have the pod deleted automatically when it’s completed

Answer: See the solution below.

Explanation:

kubectl run busybox --image=busybox -it --rm --restart=Never --
/bin/sh -c 'echo hello world'
kubectl get po # You shouldn't see pod with the name "busybox"

## Question: 32

SIMULATION

Create a pod with environment variables as var1=value1.Check the environment variable in pod

Answer: See the solution below.

Explanation:

```
kubectl run nginx --image=nginx --restart=Never --env=var1=value1
#  then
kubectl exec -it nginx -- env
#  or
kubectl exec -it nginx -- sh -c 'echo $var1'
#  or
kubectl describe po nginx | grep value1
```

## Question: 33

SIMULATION

Get list of all the pods showing name and namespace with a jsonpath expression.

Answer: See the solution below.

Explanation:

```
kubectl get pods -o=jsonpath="{.items[*]['metadata.name'
, 'metadata.namespace']}"
```

## Question: 34

SIMULATION

Check the image version in pod without the describe command

Answer: See the solution below.

Explanation:

```
kubectl get po nginx -o
jsonpath='{.spec.containers[].image}{"\n"}'
```

## Question: 35

SIMULATION

List the nginx pod with custom columns POD_NAME and POD_STATUS

Answer: See the solution below.

Explanation:

kubectl get po -o=custom-columns="POD_NAME:.metadata.name, POD_STATUS:.status.containerStatuses[].state"

## Question: 36

SIMULATION

List all the pods sorted by name

Answer: See the solution below.

Explanation:

kubect1 get pods --sort-by=.metadata.name

## Question: 37

SIMULATION

Create a pod that having 3 containers in it? (Multi-Container)

Answer: See the solution below.

Explanation:

image=nginx, image=redis, image=consul

Name nginx container as “nginx-container”

Name redis container as “redis-container”

Name consul container as “consul-container”

Create a pod manifest file for a container and append container section for rest of the images

kubectl run multi-container --generator=run-pod/v1 --image=nginx -dry-run -o yaml > multi-container.yaml

# then

vim multi-container.yaml

apiVersion: v1

kind: Pod metadata: labels:

run: multi-container name: multi-container spec: containers:

# image: nginx name: nginx-container - image: redis name: redis-container

# image: consul

name: consul-container restartPolicy: Always

## Question: 38

SIMULATION

Create 2 nginx image pods in which one of them is labelled with env=prod and another one labelled with env=dev and verify the same.

Answer: See the solution below.

Explanation:

kubectl run --generator=run-pod/v1 --image=nginx -- labels=env=prod nginx-prod --dry-run -o yaml > nginx-prodpod.yaml Now, edit nginx-prod-pod.yaml file and remove entries like “creationTimestamp: null” “dnsPolicy: ClusterFirst”

vim nginx-prod-pod.yaml

apiVersion: v1
kind: Pod
metadata:
labels:
env: prod
name: nginx-prod
spec:
containers:
# image: nginx
name: nginx-prod
restartPolicy: Always
# kubectl create -f nginx-prod-pod.yaml

kubectl run --generator=run-pod/v1 --image=nginx -- labels=env=dev nginx-dev --dry-run -o yaml > nginx-dev-pod.yaml

apiVersion: v1
kind: Pod
metadata:
labels:
env: dev
name: nginx-dev
spec: containers:
- image: nginx name: nginx-dev restartPolicy: Always
# kubectl create -f nginx-prod-dev.yaml

Verify :

kubectl get po --show-labels kubectl get po -l env=prod kubectl get po -l env=dev

## Question: 39

SIMULATION

Get IP address of the pod – “nginx-dev”

Answer: See the solution below.

Explanation:

Kubect1 get po -o wide

Using JsonPath

kubect1 get pods -o=jsonpath='{range items[*]}{.metadata.name}{"\t"}{.status.podIP}{"\n"}{end}'

## Question: 40

SIMULATION

Print pod name and start time to “/opt/pod-status” file

Answer: See the solution below.

Explanation:

kubect1 get pods -o=jsonpath='{range items[*]}{.metadata.name}{"\t"}{.status.podIP}{"\n"}{end}'

## Question: 41

SIMULATION

Check the Image version of nginx-dev pod using jsonpath

Answer: See the solution below.

Explanation:

kubect1 get po nginx-dev -o jsonpath='{.spec.containers[].image}{"\n"}'

## Question: 42

SIMULATION

Create a busybox pod and add “sleep 3600” command

Answer: See the solution below.

Explanation:

kubectl run busybox --image=busybox --restart=Never -- /bin/sh -c "sleep 3600"

## Question: 43

SIMULATION

Create an nginx pod and list the pod with different levels of verbosity

Answer: See the solution below.

Explanation:

// create a pod
kubectl run nginx --image=nginx --restart=Never --port=80

// List the pod with different verbosity
kubectl get po nginx --v=7
kubectl get po nginx --v=8
kubectl get po nginx --v=9

## Question: 44

SIMULATION
List the nginx pod with custom columns POD_NAME and POD_STATUS

Answer: See the solution below.

Explanation:

kubectl get po -o=custom-columns="POD_NAME:.metadata.name, POD_STATUS:.status.containerStatuses[].state"

## Question: 45

SIMULATION
List all the pods sorted by name

Answer: See the solution below.

Explanation:

kubectl get pods --sort-by=.metadata.name

## Question: 46

SIMULATION
List all the pods sorted by created timestamp

Answer: See the solution below.

Explanation:

kubect1 get pods--sort-by=.metadata.creationTimestamp

## Question: 47

SIMULATION
List all the pods showing name and namespace with a json path expression

Answer: See the solution below.

Explanation:

kubectl get pods -o=jsonpath="{.items[*]['metadata.name', 'metadata.namespace']}"

## Question: 48

SIMULATION

List “nginx-dev” and “nginx-prod” pod and delete those pods

Answer: See the solution below.

Explanation:

kubect1 get pods -o wide
kubectl delete po “nginx-dev”
kubectl delete po “nginx-prod”

## Question: 49

SIMULATION

Score: 4%

Set configuration context:

f ■■|ı|,--|l .;: , !-- 1] 5 kube ctl config use-context k

8s

Context

You have been asked to create a new ClusterRole for a deployment pipeline and bind it to a specific ServiceAccount scoped to a specific namespace.

Task

Create a new ClusterRole named deployment-clusterrole, which only allows to create the following resource types:

- Deployment
- StatefulSet
- DaemonSet

Create a new ServiceAccount named cicd-token in the existing namespace app-team1.

Bind the new ClusterRole deployment-clusterrole lo the new ServiceAccount cicd-token , limited to the namespace app-team1.

Answer: See the solution below.

Explanation:

Solution:

Task should be complete on node k8s -1 master, 2 worker for this connect use command [student@node-1] > ssh k8s kubectl create clusterrole deployment-clusterrole --verb=create --resource=deployments,statefulsets,daemonsets

kubectl create serviceaccount cicd-token --namespace=app-team1

kubectl create rolebinding deployment-clusterrole --clusterrole=deployment-clusterrole --serviceaccount=default:cicd-token --namespace=app-team1

## Question: 50

SIMULATION

Score: 4%

Ser configuration context?

is:j'S itimodc 11 5 kube ctl config use-context e k8s

Task

Set the node named ek8s-node-1 as unavailable and reschedule all the pods running on it.

Answer: See the solution below.

Explanation:

SOLUTION:

[student@node-1] > ssh ek8s

kubectl cordon ek8s-node-1

kubectl drain ek8s-node-1 --delete-local-data --ignore-daemonsets --force

## Question: 51

SIMULATION

Score: 7%

Set configuration context:

j: ; . kube

ctl corifig use-context m

kSs

Task

Given an existing Kubernetes cluster running version 1.20.0, upgrade all of the Kubernetes control plane and node components on the master node only to version 1.20.1.

Be sure to drain the master node before upgrading it and uncordon it after the upgrade.

You ran ssh to th^ master node using:

f n^lentfiK'xi—l] $ ssh mk8s-master-0

You can assume elevated privileges on the master node with the following command:


sudo -i

You are also expected to upgrade kubelet and kubectl on the master node.

Do not upgrade the worker nodes, etui the container manager, the CN1 plugin, the DNS service or any other addons.

Answer: See the solution below. Explanation:

SOLUTION: [student@node-1] > ssh ek8s kubectl cordon k8s-master

kubectl drain k8s-master --delete-local-data --ignore-daemonsets --force apt-get install kubeadm=1.20.1-00 kubelet=1.20.1-00 kubectl=1.20.1-00 -- disableexcludes=kubernetes

kubeadm upgrade apply 1.20.1 --etcd-upgrade=false systemctl daemon-reload systemctl restart kubelet kubectl uncordon k8s-master

## Question: 52

SIMULATION

Score: 7%

No configuration context change required for this task.

Ensure, however, that you have returned to the base node before starting to work on this task:

[ studenWnkSs nwster 0] $ exit

Task

First, create a snapshot of the existing etcd instance running at https://127.0.0.1:2379, saving the snapshot to /srv/data/etcd-snapshot.db.

Creating a snapshot of the given instance is expected to complete in seconds.

If the operation seems to hang, something's likely wrong with your command. Use CTRL -C to cancel the operation and try again.

Next, restore an existing, previous snapshot located at /var/lib/backup/etcd-snapshot-previo us.db

.

The following TLS certificates/key are supplied for connecting to the server wnh etcdctl:

- CA certificate: Zopi/KUIN00601/cicrt
- Client certificate: /opt/KUINOOWI/etcd^^ Cert
- Client key: /opt/KU IN 0 0601 /eted-dien t.key

Answer: See the solution below.

Explanation:

Solution:
#backup
ETCDCTL_API=3 etcdctl --endpoints="https://127.0.0.1:2379" --cacert=/opt/KUIN000601/ca.crt --cert=/opt/KUIN000601/etcd-client.crt --key=/opt/KUIN000601/etcd-client.key snapshot save /etc/data/etcd-snapshot.db

#restore
ETCDCTL_API=3 etcdctl --endpoints="https://127.0.0.1:2379" --cacert=/opt/KUIN000601/ca.crt --cert=/opt/KUIN000601/etcd-client.crt --key=/opt/KUIN000601/etcd-client.key snapshot restore /var/lib/backup/etcd-snapshot-previoys.db

## Question: 53

SIMULATION
Score: 7%

Set configuration context:

H tudentjm^dc li 'I kube ctl config use-context h k8s

Task

Create a new NetworkPolicy named allow-port-from-namespace in the existing namespace echo. Ensure that the new NetworkPolicy allows Pods in namespace my-app to connect to port 9000 of Pods in namespace echo.

Further ensure that the new NetworkPolicy:

- does not allow access to Pods, which don't listen on port 9000
- does not allow access from Pods, which are not in namespace my-app

Answer: See the solution below.

Explanation:

Solution:

#network.yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy metadata:
name: allow-port-from-namespace namespace: internal
spec:
podSelector:
matchLabels: {

}
policyTypes:
- Ingress ingress: - from:
- podSelector: {

}
ports:
- protocol: TCP
port: 8080
#spec.podSelector namespace pod kubectl create -f network.yaml

## Question: 54

SIMULATION

Score: 7%

Set configuration context:

[ studentiwiode-l] $ kube ctl config use-context k 8s

Task

Reconfigure the existing deployment front-end and add a port specification named http exposing port 80/tcp of the existing container nginx.

Create a new service named front-end-svc exposing the container port http.

Configure the new service to also expose the individual Pods via a NodePort on the nodes on which they are scheduled.

Answer: See the solution below.

Explanation:

Solution:

kubectl get deploy front-end
kubectl edit deploy front-end -o yaml
#port specification named http
#service.yaml
apiVersion: v1
kind: Service
metadata:
name: front-end-svc
labels:
app: nginx
spec:
ports:
- port: 80
protocol: tcp
name: http
selector:
app: nginx
type: NodePort
- kubectl create -f service.yaml
- kubectl get svc
- port specification named http

kubectl expose deployment front-end --name=front-end-svc --port=80 --tarport=80 --type=NodePort

## Question: 55

SIMULATION

Score: 7%

Set configuration context:

```
[ ■ u lent » 1] $ kube ctl contig use-context k 8s
```

Task

Create a new nginx Ingress resource as follows:

- Name: ping
- Namespace: ing-internal
- Exposing service hi on path /hi using service port 5678

The avaliability of service hi can be checked using the following command, which should return hi:

```
?; . ^-v- ■. Li ■ curl kL <INTERNAL_IP>/hi
```

Answer: See the solution below.

Explanation:

Solution:

vi ingress.yaml

#

apiVersion: networking.k8s.io/v1

kind: Ingress metadata: name: ping namespace: ing-internal spec: rules:

- http: paths:
- path: /hi pathType: Prefix backend: service: name: hi port: number: 5678 #

kubectl create -f ingress.yaml

## Question: 56

SIMULATION

Score: 4%

Set configuration context

[studuntwodc 1] J kube ctl config use-context k 8$

Task

Scale the deployment presentation to 6 pods.

Answer: See the solution below.

Explanation:

Solution:

kubectl get deployment
kubectl scale deployment.apps/presentation --replicas=6

## Question: 57

SIMULATION

Score: 4%

Set configuration context

[student^i. . 1] $ kube ctl config use-context k 8s

Task

Schedule a pod as follows:

- Name: nginx-kusc00401
- Image: nginx

- Node selector: disk=ssd

Answer: See the solution below.

Explanation:

Solution:

#yaml
apiVersion: v1 kind: Pod metadata: name: nginx-kusc00401
spec: containers:
- name: nginx image: nginx imagePullPolicy: IfNotPresent nodeSelector:
disk: spinning
#
kubectl create -f node-select.yaml

## Question: 58

SIMULATION
Score: 4%

Set configuration cantext

■ < . ■ - -i ' ? ku be ctl config use-context k 8s

Task
Check to see how many nodes are ready schedulable (not including nodes tainted NoSchedule ) and write the number to /opt/KUSC00402/kusc00402.txt.

Answer: See the solution below.

Explanation:

Solution:

kubectl describe nodes | grep ready|wc -l
kubectl describe nodes | grep -i taint | grep -i noschedule |wc -l
echo 3 > /opt/KUSC00402/kusc00402.txt
#
kubectl get node | grep -i ready |wc -l

- taints、noSchedule

kubectl describe nodes | grep -i taints | grep -i noschedule |wc -l
#
echo 2 > /opt/KUSC00402/kusc00402.txt

## Question: 59

SIMULATION
Score: 4%

Set configuration context:

iv. ■ < kube ttl config use-context k Us

Task
Create a pod named kucc8 with a single app container for each of the following images running inside (there may be between 1 and 4 images specified): nginx + redis + memcached .

Answer: See the solution below.

Explanation:

Solution:

kubectl run kucc8 --image=nginx --dry-run -o yaml > kucc8.yaml
- vi kucc8.yaml

apiVersion: v1 kind: Pod metadata:
creationTimestamp: null name: kucc8
spec:
containers:
- image: nginx name: nginx
- image: redis name: redis
- image: memcached name: memcached
- image: consul name: consul

#
kubectl create -f kucc8.yaml
#12.07

## Question: 60

SIMULATION

Score: 4%

Set configuration context;

[stu&ntjincKie 1] £ kube ctl config use-context h k8s

Task
Create a persistent volume with name app-data , of capacity 1Gi and access mode ReadOnlyMany.
The type of volume is hostPath and its location is /srv/app-data .

Answer: See the solution

Explanation:

Solution:

#vi pv.yaml apiVersion: v1 kind: PersistentVolume metadata: name: app-config spec: capacity: storage: 1Gi accessModes:
- ReadOnlyMany hostPath:
path: /srv/app-config #
kubectl create -f pv.yaml

## Question: 61
SIMULATION
Score:7%

Set configui-tion context;

| '.Ho ""ie L] ; kube ctl con-fig usc--context o k8s

Task
Create a new PersistentVolumeClaim
- Name: pv-volume
- Class: csi-hostpath-sc
- Capacity: 10Mi

Create a new Pod which mounts the PersistentVolumeClaim as a volume:

- Name: web-server
- Image: nginx
- Mount path: /usr/share/nginx/html

Configure the new Pod to have ReadWriteOnce access on the volume.

Finally, using kubectl edit or kubectl patch expand the PersistentVolumeClaim to a capacity of 70Mi and record that change.

Answer: See the solution below.

Explanation:

Solution:

vi pvc.yaml
storageclass pvc
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
name: pv-volume
spec:
accessModes:
- ReadWriteOnce
volumeMode: Filesystem
resources:
requests:
storage: 10Mi
storageClassName: csi-hostpath-sc

- vi pod-pvc.yaml

apiVersion: v1 kind: Pod metadata:
name: web-server
spec:
containers:
- name: web-server image: nginx volumeMounts:
- mountPath: "/usr/share/nginx/html" name: my-volume volumes:
- name: my-volume persistentVolumeClaim: claimName: pv-volume # craete
kubectl create -f pod-pvc.yaml
#edit
kubectl edit pvc pv-volume --record

Question: 62 SIMULATION Score: 5%

Set configuration context:

[ :d udc-nt i;rhi>k- 1] $ kube ctl config use-context k 8S

Task
Monitor the logs of pod bar and:

- Extract log lines corresponding to error file-not-found
- Write them to /opt/KUTR00101/bar

Answer: See the solution below.

Explanation:

Solution:

kubectl logs bar | grep 'unable-to-access-website' > /opt/KUTR00101/bar
cat /opt/KUTR00101/bar

## Question: 63

SIMULATION
Score:7%

Set configuration context

[stiifcit-. 1' ■ kube ctl config use-context k

Context
An existing Pod needs to be integrated into the Kubernetes built-in logging architecture (e. g. kubectl logs). Adding a streaming sidecar container is a good and common way to accomplish this requirement.

Task
Add a sidecar container named sidecar, using the busybox Image, to the existing Pod big-corp-app.
The new sidecar container has to run the following command:

/bin/sh -c tail -n+1 -f /va r/log/big-corp-app.log

Use a Volume, mounted at /var/log, to make the log file big-corp-app.log available to the sidecar container.

Don t modify the specification of the existing container other than adding the required volume mount.

Explanation:

Solution:

Answer: See the solution

# kubectl get pod big-corp-app -o yaml #
apiVersion: v1 kind: Pod metadata: name: big-corp-app spec:
containers:
- name: big-corp-app image: busybox args:
- /bin/sh
- -c
- > i=0; while true; do
echo "$(date) INFO $i" >> /var/log/big-corp-app.log; i=$((i+1));
sleep 1;
done
volumeMounts:
- name: logs mountPath: /var/log - name: count-log-1 image: busybox args: [/bin/sh, -c, 'tail -n+1 -f /var/log/big-corp-app.log'] volumeMounts:
- name: logs mountPath: /var/log
volumes:
- name: logs emptyDir: {
}
#
kubectl logs big-corp-app -c count-log-1

## Question: 64

SIMULATION
Score: 5%

Sei■: on figuration context:

[stuient^Oje 1] j kube ctl config use-context k 8s

Task

From the pod label name=cpu-utilizer, find pods running high CPU workloads and write the name of the pod consuming most CPU to the file /opt/KUTR00401/KUTR00401.txt (which already exists).

Answer: See the solution below.

Explanation:

Solution:

kubectl top -l name=cpu-user -A
echo 'pod name' >> /opt/KUT00401/KUT00401.txt

## Question: 65

SIMULATION

Score: 13%

Set configuration context

I studentmiJc 1) i ku be ctl c<3trfig use-context w kSs

Task

A Kubernetes worker node, named wk8s-node-0 is in state NotReady. Investigate why this is the case, and perform any appropriate steps to bring the node to a Ready state, ensuring that any changes are made permanent.

You can ssh to the failed

node using;

tu lent . ? i. ? ssh

wk8s'(iode-&

You can assume elevated privileges on the node witi' the foliowing command;

[ st udef i t^USks - node- 0 ] $ sudo -i

Answer: See the solution below.

Explanation:

Solution:

sudo -i

systemctl status kubelet systemctl start kubelet systemctl enable kubelet

## Question: 66

SIMULATION

Task Weight: 4%

Set con figuration context:

' 1 ■ : kurort] config

ui#»tanT4ict kas

Task

Scale the deployment webserver to 3 pods.

Answer: See the solution below.

Explanation:

Solution:

```
-it Erwnt Hmdr H'S kubertI 5CAH Ifploy wet server                    r^pticas=3
M-piayr^n: .Apps/w^Li^nn*!: scaled
studentGnodfi—l:^$ kubect1 scale deploy webserveE —replicas*!
```

## Question: 67

SIMULATION

Task Weight: 4%

Set configu ration context:

```
.■'.. Ir'lni^-. t kubettl CCnf ig use-context k8s
```

Task

Schedule a Pod as follows:

- Name: kucc1
- App Containers: 2
- Container Name/Images: o nginx

o consul

Answer: See the solution below.

Explanation:

Solution:

```
studentUnode It'S Irabecti config use context Ms
Switched to context •Ms".
studentfcnode-I:-$ kubectl run kuccl —iaage^nginx —dry-runnellent -o y<uil > M*y|
```

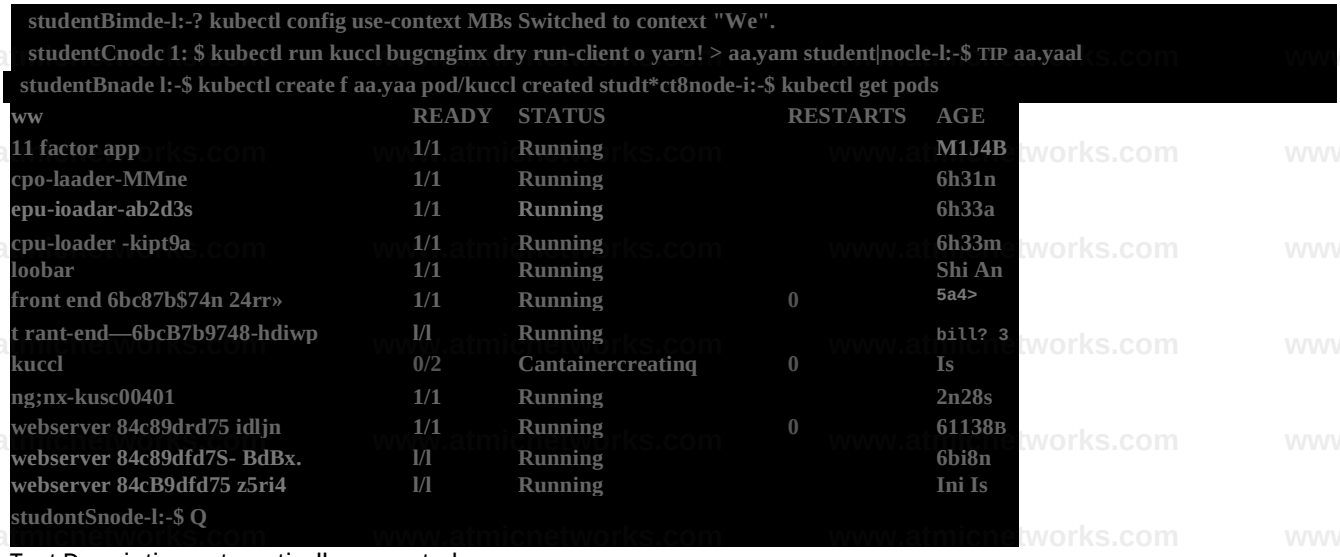
• Readme >_ web Terminal

□ THE LINUX FOUNDATION

. . ¥1
Pod

kuccl kuccl

ngwx nqinx
consul
COnSUll




Graphical user interface, text, application Description automatically generated

Text Description automatically generated

## Question: 68

SIMULATION

Schedule a Pod as follows:

. Name: kucc1

. App Containers : 2

. Container Name/Images : redis

Memcached

Answer: See

the solution below.

Explanation:

```
apiVersion: vl kind: Pod metadata:
  treationTimestamp: null labels:
    run: kuccl name: kuccl spec: containers:
- image: redis name: redis

  • image: memeached name: memcached
```

screen shot of a computer AI-generated content may be incorrect.

Question: 69

SIMULATION

Quick Reference

HorizontalPodAutoscaler Walkthrough

Documentation *

Horizontal Pod Autoscaling f Deployment

You must connect to the correct host. Failure to do so may result in a zero score.

■ ; - ba: - ssh ckaeeees©

Task

Create a new HorizontalPodAutoscaler (HPA ) named apache-server in the autoscale namespace. This HPA must target the existing Deployment called apache-server in the autoscale namespace.

Set the HPA to aim for 50% CPU usage per Pod . Configure it to have at least 1 Pod and no more than 4 Pods . Also, set the downscale stabilization window to 30 seconds.

Answer: See the solution below.

Explanation:

Task Summary

Create an HPA named apache-server in the autoscale namespace.

Target an existing deployment also named apache-server.

CPU target: 50%

Pod range: min 1, max 4

Downscale stabilization window: 30 seconds

Step-by-Step Answer

Step 1: Connect to the correct host

This is critical, as shown in the warning image. ssh cka000050

Skipping this may result in zero for this question!

Step 2: Verify the deployment exists

kubectl get deployment apache-server -n autoscale

Make sure it’s there before creating the HP

A. If it’s missing, the HPA won't bind correctly.

$ Step 3: Create the HPA

We will use the kubectl autoscale command for a quick setup, then patch it to add the stabilization window (since kubectl autoscale doesn't include it).

kubectl autoscale deployment apache-server \

- -namespace autoscale \
- -cpu-percent=50 \
- -min=1 \
- -max=4

Step 4: Add the downscale stabilization window

You’ll need to patch the HPA to include the stabilization window of 30s.

Create a patch file called hpa-patch.yaml:

spec:
behavior:
scaleDown:
stabilizationWindowSeconds: 30
Apply the patch: bash
CopyEdit
kubectl patch hpa apache-server \
- n autoscale \
- -patch "$(cat hpa-patch.yaml)"

Q Step 5: Confirm your work
bash
CopyEdit
kubectl describe hpa apache-server -n autoscale
Look for:
Min/Max Pods: 1/4
Target CPU utilization: 50%
Stabilization window: should appear under Behavior > ScaleDown

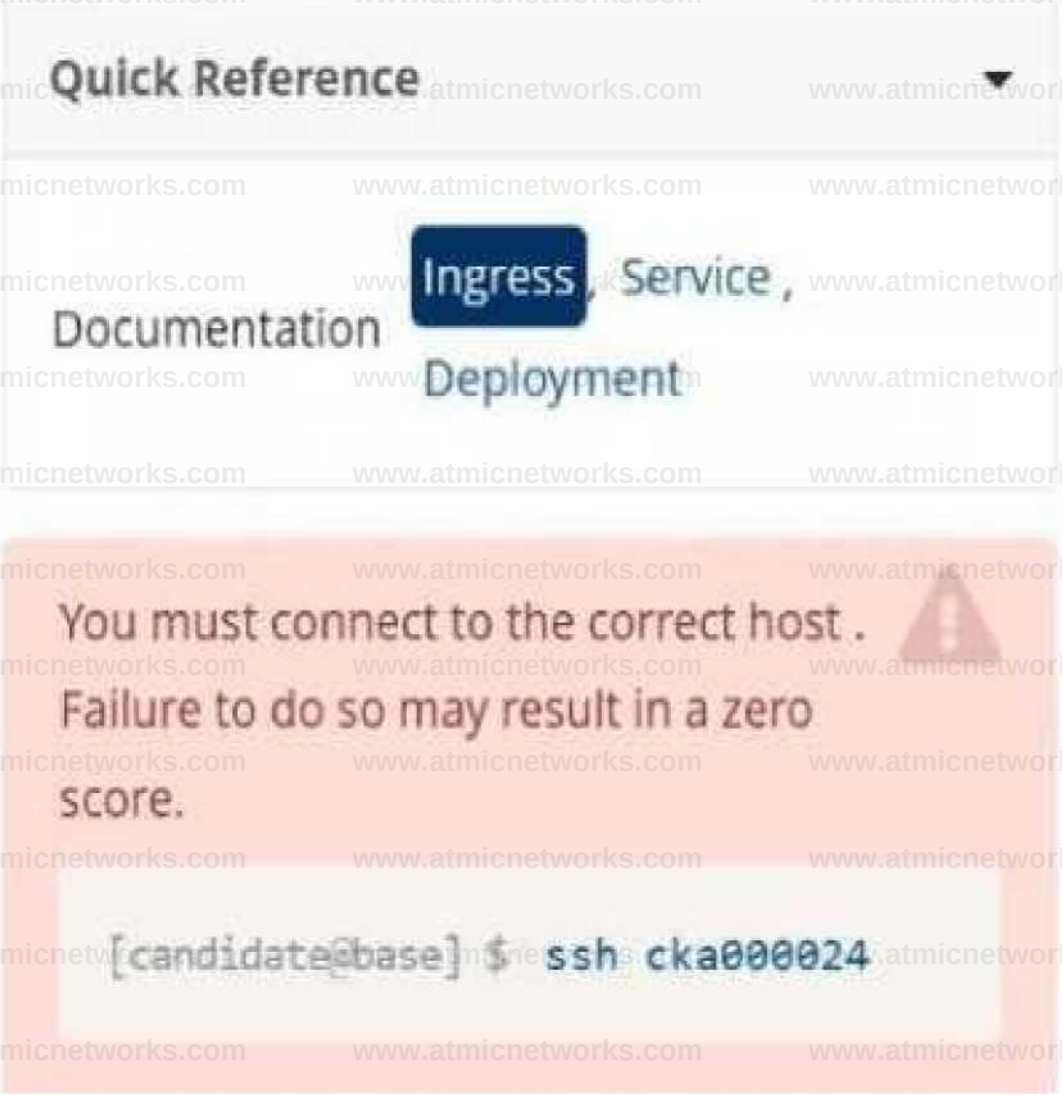
ssh cka000050

kubectl get deployment apache-server -n autoscale

kubectl autoscale deployment apache-server \
- -namespace autoscale \
- -cpu-percent=50 \
- -min=1 \
- -max=4

- Patch to add stabilization window
cat <<EOF > hpa-patch.yaml
spec:
behavior:
scaleDown:
stabilizationWindowSeconds: 30
EOF
kubectl patch hpa apache-server -n autoscale --patch "$(cat hpa-patch.yaml)"

## Question: 70
SIMULATION




Task

Create a new Ingress resource as follows:

. Name: echo

. Namespace : sound-repeater

. Exposing Service echoserver-service on

http://example.org/echo using Service port 8080

The availability of Service

echoserver-service can be checked

using the following command, which should return 200 :

[candidate@cka000024] $ curl -o /de v/null -s -w "%{http_code}\n" http://example.org/echo

Answer: See the solution below.

Explanation:

Task Summary
Create an Ingress named echo in the sound-repeater namespace that:
Routes requests to /echo on host example.org
Forwards traffic to service echoserver-service
Uses service port 8080
Verification should return HTTP 200 using curl

Step-by-Step Answer

1 QSH into the correct node
As shown in the image:
bash
CopyEdit
ssh cka000024
Skipping this will result in a ZERO score!

2 (2Verify the namespace and service
Ensure the sound-repeater namespace and echoserver-service exist:

kubectl get svc -n sound-repeater
Look for:

echoserver-service ClusterIP ... 8080/TCP

3 Qreate the Ingress manifest
Create a YAML file: echo-ingress.yaml

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
name: echo
namespace: sound-repeater
annotations:
nginx.ingress.kubernetes.io/rewrite-target: /$1
spec:
rules:
- host: example.org
http:
paths:
- path: /echo
pathType: Prefix
backend:
service:
name: echoserver-service
```

port:
number: 8080

4 (ZAPPly the Ingress resource

```
kubectl apply -f echo-ingress.yaml
```

5 Qest with curl as instructed
Use the exact verification command:

```
curl -o /dev/null -s -w "%{http_code}\n" http://example.org/echo
```

You should see:

200

Final Answer Summary

ssh cka000024

```
kubectl get svc -n sound-repeater
```

- Create the Ingress YAML

```
cat <<EOF > echo-ingress.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
name: echo
namespace: sound-repeater
annotations:
nginx.ingress.kubernetes.io/rewrite-target: /$1
spec:
rules:
- host: example.org
http:
paths:
- path: /echo
pathType: Prefix
backend:
service:
name: echoserver-service
port:
number: 8080
EOF
```

```
kubectl apply -f echo-ingress.yaml
curl -o /dev/null -s -w "%{http_code}\n" http://example.org/echo
```

## Question: 71

SIMULATION

You must connect to the correct host.

Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000051

Context

You manage a WordPress application. Some Pods are not starting because resource requests are too high. Your task Is to prepare a Linux system for Kubernetes . Docker is already installed, but you need to configure it for kubeadm .

Task

Complete these tasks to prepare the system for Kubernetes :

Set up cri-dockerd :

. Install the Debian package ~/cri-dockerd_0.3.9.3-0.ubuntu-jammy_amd64.deb

Debian packages are installed using dpkg .

. Enable and start the cri-docker service

Configure these system parameters:

. Set net.bridge.bridge-nf-call-iptables to 1

Answer: See the solution below.

Explanation:

Task Summary

You are given a host to prepare for Kubernetes:

Use dpkg to install cri-dockerd

Enable and start the cri-docker service

Set net.bridge.bridge-nf-call-iptables to 1 via sysctl

Step-by-Step Instructions

1 QSH into the correct node

bash

CopyEdit ssh cka000051

Required — failure to connect to the correct host = zero score.

2 Qnstall cri-dockerd

You are told the .deb file is already located at:

bash

CopyEdit

~/cri-dockerd_0.3.9.3-0.ubuntu-jammy_amd64.deb

Install it with dpkg:

bash

CopyEdit

sudo dpkg -i ~/cri-dockerd_0.3.9.3-0.ubuntu-jammy_amd64.deb

If any dependencies are missing (e.g., golang or containerd), you might need:

bash

CopyEdit

sudo apt-get install -f -y

But usually, the exam system provides a pre-validated .deb environment.

3 Qnable and start cri-docker service

Start and enable both services:

bash

CopyEdit

sudo systemctl enable cri-docker.service

sudo systemctl enable --now cri-docker.socket

sudo systemctl start cri-docker.service

Check status (optional but smart):

bash

CopyEdit

sudo systemctl status cri-docker.service

You should see it active (running).

4 Qonfigure the sysctl parameter
Set net.bridge.bridge-nf-call-iptables=1 immediately and persistently.
Step A: Apply immediately:

sudo sysctl net.bridge.bridge-nf-call-iptables=1
Step B: Persist it in /etc/sysctl.d:
Create or modify a file:

echo "net.bridge.bridge-nf-call-iptables = 1" | sudo tee /etc/sysctl.d/k8s.conf Reload sysctl:

sudo sysctl --system
Verify:

sysctl net.bridge.bridge-nf-call-iptables
Should return:

net.bridge.bridge-nf-call-iptables = 1
Now the system is ready for kubeadm with Docker (via cri-dockerd)!

ssh cka000051

sudo dpkg -i ~/cri-dockerd_0.3.9.3-0.ubuntu-jammy_amd64.deb

sudo systemctl enable cri-docker.service
sudo systemctl enable --now cri-docker.socket
sudo systemctl start cri-docker.service

sudo sysctl net.bridge.bridge-nf-call-iptables=1
echo "net.bridge.bridge-nf-call-iptables = 1" | sudo tee /etc/sysctl.d/k8s.conf sudo sysctl --system

# Question: 72
SIMULATION

You must connect to the correct host.
Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000058
Context

You manage a WordPress application. Some Pods

are not starting because resource requests are too high.

Task
A WordPress application in the relative-fawn namespace consists of:

5 A WordPress Deployment with 3 replicas.

Adjust all Pod resource requests as follows:

6 Divide node resources evenly across all 3 Pods.

7 Give each Pod a fair share of CPU and memory.

Answer: See the solution below.

Explanation:

Task Summary

You are managing a WordPress Deployment in namespace relative-fawn.

Deployment has 3 replicas.

Pods are not starting due to high resource requests.

Your job: Adjust CPU and memory requests so that all 3 pods evenly split the node's capacity.

Step-by-Step Solution

1 QSH into the correct host

bash

CopyEdit

ssh cka000058

Skipping this will result in a zero score.

2 Qheck node resource capacity

You need to know the node's CPU and memory resources.

bash

CopyEdit

kubectl describe node | grep -A5 "Capacity"

Example output:

yaml

CopyEdit

Capacity:

cpu: 3

memory: 3Gi

Let's assume the node has:

3 CPUs

3Gi memory

So for 3 pods, divide evenly:

CPU request per pod: 1

Memory request per pod: 1Gi

In the actual exam, check real values and divide accordingly. If the node has 4 CPUs and 8Gi, you'd allocate ~1.33 CPUs and ~2.66Gi RAM per pod (rounded reasonably).

4 Qdit the Deployment

Edit the WordPress deployment in the relative-fawn namespace:

kubectl edit deployment wordpress -n relative-fawn

Look for the resources section under spec.template.spec.containers like this:

resources:

requests:
cpu: "1"
memory: "1Gi"
If the section doesn't exist, add it manually.
Save and exit the editor (:wq if using vi).

5 3onfirm changes
Wait a few seconds, then check:

kubectl get pods -n relative-fawn
Ensure all 3 pods are in Running state.
You can also describe a pod to confirm resource requests are set:
kubectl describe pod <pod-name> -n relative-fawn | grep -A5 "Containers"

ssh cka000058

kubectl describe node | grep -A5 "Capacity"

kubectl edit deployment wordpress -n relative-fawn
# Set CPU: 1, Memory: 1Gi (or according to node capacity)

kubectl get pods -n relative-fawn

## Question: 73
SIMULATION

You must connect to the correct host.
Failure to do so may result in a zero score.
[candidate@base] $ ssh Cka000037

Context
A legacy app needs to be integrated into the Kubernetes built-in logging architecture (i.e. kubectl logs). Adding a streaming co-located container is a good and common way to accomplish this requirement.

Task

Update the existing Deployment synergy-leverager, adding a co-located container named sidecar using the busybox:stable image to the existing Pod . The new co-located container has to run the following command:

/bin/sh -c "tail -n+1 -f /var/log/syne
rgy-leverager.log"

Use a Volume mounted at /var/log to make the log file synergy-leverager.log available to the colocated container .

Do not modify the specification of the existing container other than adding the required volume mount .
Failure to do so may result in a reduced score.

Answer: See the solution below.

Explanation:

Task Summary
SSH into the correct node: cka000037
Modify existing deployment synergy-leverager
Add a sidecar container:
Name: sidecar
Image: busybox:stable
Command:

/bin/sh -c "tail -n+1 -f /var/log/synergy-leverager.log"
Use a shared volume mounted at /var/log
Don't touch existing container config except adding volume mount

Step-by-Step Solution

1 QSH into the correct node

ssh cka000037
Skipping this will result in a zero score.

2 Qdit the deployment

kubectl edit deployment synergy-leverager
This opens the deployment YAML in your default editor (vi or similar).

3 QModify the spec as follows
Inside the spec.template.spec, do these 3 things:
Q

A. Define a shared volume
Add under volumes: (at the same level as containers):
volumes:
- name: log-volume emptyDir: {}

Q B. Add volume mount to the existing container
Locate the existing container under containers: and add this:

volumeMounts:
- name: log-volume mountPath: /var/log Do not change any other configuration for this container.

Q C. Add the sidecar container
Still inside containers:, add the new container definition after the first one: - name: sidecar
image: busybox:stable command:
- /bin/sh
- -c

- "tail -n+1 -f /var/log/synergy-leverager.log" volumeMounts:
- name: log-volume mountPath: /var/log

spec: containers:
- name: main-container image: your-existing-image volumeMounts:
- name: log-volume mountPath: /var/log - name: sidecar image: busybox:stable command:
- /bin/sh
- -c

- "tail -n+1 -f /var/log/synergy-leverager.log" volumeMounts:
- name: log-volume mountPath: /var/log volumes:
- name: log-volume emptyDir: {}

Save and exit

If using vi or vim, type: bash

CopyEdit

wq

5 CVerify

Check the updated pods:

kubectl get pods -l app=synergy-leverager

Pick a pod name and describe it:

kubectl describe pod <pod-name>

Confirm:

2 containers running (main-container + sidecar)
Volume mounted at /var/log

ssh cka000037

kubectl edit deployment synergy-leverager
# Modify as explained above

kubectl get pods -l app=synergy-leverager
kubectl describe pod <pod-name>

## Question: 74

SIMULATION

You must connect to the correct host.

Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000054

Context:

Your cluster 's CNI has failed a security audit. It has been removed. You must install a new CNI that can enforce network policies.

Task

Install and set up a Container Network Interface (CNI ) that meets these requirements:

Pick and install one of these CNI options:

· Flannel version 0.26.1

Manifest:

https://github.com/flannel-io/flannel/releases/download/v0.26.1/kube-flannel.yml

· Calico version 3.28.2

Manifest:

https://raw.githubusercontent.com/project calico/calico/v3.28.2/manifests/tigera-operator.yaml

Answer: See the solution below.

Explanation:

Task Summary

SSH into cka000054

Install a CNI plugin that supports NetworkPolicies

Two CNI options provided:

Flannel v0.26.1 ( does NOT support NetworkPolicies)

Calico v3.28.2 (does support NetworkPolicies)

Decision Point: Which CNI to choose?

Choose Calico, because only Calico supports enforcing NetworkPolicies natively. Flannel does not.

Step-by-Step Solution

1 QSH into the correct node ssh cka000054

Required. Skipping this results in zero score.

2 Qnstall Calico CNI (v3.28.2)

Use the official manifest provided:

kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.28.2/manifests/tigera- operator.yaml

This installs the Calico Operator, which then deploys the full Calico CNI stack.

3 (2Wait for Calico components to come up

Check the pods in tigera-operator and calico-system namespaces:

kubectl get pods -n tigera-operator

kubectl get pods -n calico-system

You should see pods like:

calico-kube-controllers

calico-node

calico-typha

tigera-operator

Wait for all to be in Running state.

Q (Optional) 4 Qonfirm CNI is enforcing NetworkPolicies

You can check:

kubectl get crds | grep networkpolicy

You should see:

networkpolicies.crd.projectcalico.org

This confirms Calico's CRDs are installed for policy enforcement.

Final Command Summary ssh cka000054

kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.28.2/manifests/tigera- operator.yaml

kubectl get pods -n tigera-operator

kubectl get pods -n calico-system

kubectl get crds | grep networkpolicy

## Question: 75

SIMULATION

You must connect to the correct host.

Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000046

Task

First, create a new StorageClass named local-path for an existing provisioner named rancher.io/local- path .

Set the volume binding mode to WaitForFirstConsumer .

Not setting the volume binding mode or setting it to anything other than WaitForFirstConsumer may result in a reduced score.

Next, configure the StorageClass local-path as the default StorageClass .

Answer: See the solution below.

Explanation:

Task Summary

You need to:

SSH into cka000046

Create a StorageClass named local-path using the provisioner rancher.io/local-path

Set the volume binding mode to WaitForFirstConsumer

Make this StorageClass the default

Step-by-Step Solution

1 QSH into the correct host

ssh cka000046

Required. Skipping this = zero score

2 Qreate a StorageClass YAML file

Create a file named local-path-sc.yaml:

cat <<EOF > local-path-sc.yaml

apiVersion: storage.k8s.io/v1

kind: StorageClass

metadata:

name: local-path

annotations:

storageclass.kubernetes.io/is-default-class: "true"

provisioner: rancher.io/local-path

volumeBindingMode: WaitForFirstConsumer

EOF

This:

Sets WaitForFirstConsumer (as required)

Marks the class as default using the correct annotation

3 (ZApply the StorageClass

kubectl apply -f local-path-sc.yaml

4 ZVerify it's the default StorageClass

kubectl get storageclass

You should see local-path with a (default) marker:

NAME PROVISIONER RECLAIMPOLICY VOLUMEBINDINGMODE ALLOWVOLUMEEXPANSION AGE local-path rancher.io/local-path Delete WaitForFirstConsumer false 10s

Final Command Summary

ssh cka000046

cat <<EOF > local-path-sc.yaml

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
name: local-path
annotations:
storageclass.kubernetes.io/is-default-class: "true" provisioner: rancher.io/local-path volumeBindingMode: WaitForFirstConsumer EOF

kubectl apply -f local-path-sc.yaml kubectl get storageclass

## Question: 76

SIMULATION
You must connect to the correct host.
Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000022

Task
Reconfigure the existing Deployment front-end in namespace spline-reticulator to expose port 80/tcp of the existing container nginx .

Create a new Service named front-end-svc exposing the container port 80/tcp .

Configure the new Service to also expose the individual Pods via a NodePort .

Answer: See the solution below.

Explanation:

Task Summary
SSH into cka000022
Modify an existing Deployment:
Namespace: spline-reticulator
Deployment: front-end
Container: nginx
Expose: port 80/tcp
Create a Service:
Name: front-end-svc
Type: NodePort
Port: 80 → container port 80

Step-by-Step Solution

1 QSH into the correct node

ssh cka000022

Skipping this = zero score

2 Qdit the Deployment to expose port 80

kubectl edit deployment front-end -n spline-reticulator

Under containers: → nginx, add this if not present:

ports:
- containerPort: 80
protocol: TCP

This enables the container to accept traffic on port 80.

3 Qreate a NodePort Service

Create a file named front-end-svc.yaml:

cat <<EOF > front-end-svc.yaml
apiVersion: v1
kind: Service
metadata:
name: front-end-svc
namespace: spline-reticulator
spec:
type: NodePort
selector:
app: front-end
ports:
1 port: 80
targetPort: 80
protocol: TCP
EOF

Make sure the Deployment has a matching label selector like app: front-end. You can verify with:

kubectl get deployment front-end -n spline-reticulator -o yaml | grep labels -A 2

4 CApply the service

kubectl apply -f front-end-svc.yaml

5 [2Verify

Check if the service is created and has a NodePort assigned:

kubectl get svc front-end-svc -n spline-reticulator

You should see something like:

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
front-end-svc NodePort 10.96.0.123 <none> 80:3XXXX/TCP 10s
Where 3XXXX is your automatically assigned NodePort (between 30000–32767).

Final Command Summary

ssh cka000022

kubectl edit deployment front-end -n spline-reticulator
# Add:
# ports:
# - containerPort: 80

cat <<EOF > front-end-svc.yaml
apiVersion: v1
kind: Service
metadata:
name: front-end-svc
namespace: spline-reticulator
spec:
type: NodePort
selector:
app: front-end
ports:
# port: 80
targetPort: 80
protocol: TCP
EOF

kubectl apply -f front-end-svc.yaml
kubectl get svc front-end-svc -n spline-reticulator

## Question: 77

SIMULATION

You must connect to the correct host.
Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000049

Task
Perform the following tasks:

Create a new PriorityClass named high-priority for user-workloads with a value that is one less than the highest existing user-defined priority class value.

Patch the existing Deployment busybox-logger running in the priority namespace to use the high- priority priority class.

Answer: See the solution below.

Explanation:

Task Summary

SSH into the correct node: cka000049

Find the highest existing user-defined PriorityClass

Create a new PriorityClass high-priority with a value one less

Patch Deployment busybox-logger (in namespace priority) to use this new PriorityClass

## Step-by-Step Solution

1 QSH into the correct node bash

CopyEdit ssh cka000049

Skipping this = zero score

2 Qind the highest existing user-defined PriorityClass Run: bash

CopyEdit

kubectl get priorityclasses.scheduling.k8s.io

Example output: vbnet

CopyEdit

NAME VALUE GLOBALDEFAULT AGE default-low 1000 false 10d mid-tier 2000 false 7d critical-pods 1000000 true 30d

Exclude system-defined classes like system-* and the default global one (e.g., critical-pods). Let's assume the highest user-defined value is 2000.

So your new class should be: Value = 1999

3 Create the high-priority PriorityClass Create a file called high-priority.yaml:

cat <<EOF > high-priority.yaml apiVersion: scheduling.k8s.io/v1 kind: PriorityClass metadata: name: high-priority value: 1999 globalDefault: false description: "High priority class for user workloads" EOF

Apply it:

kubectl apply -f high-priority.yaml

4 QPatch the busybox-logger deployment

Now patch the existing Deployment in the priority namespace:

```
kubectl patch deployment busybox-logger -n priority \
--type='merge' \
-p '{"spec": {"template": {"spec": {"priorityClassName": "high-priority"}}}}'
```

5 (2Verify your work

Confirm the patch was applied:

```
kubectl get deployment busybox-logger -n priority -o
jsonpath='{.spec.template.spec.priorityClassName}'
```

You should see:

high-priority

Also, confirm the class exists:

```
kubectl get priorityclass high-priority
```

Final Command Summary ssh cka000049

kubectl get priorityclass

# Create the new PriorityClass cat <<EOF > high-priority.yaml apiVersion: scheduling.k8s.io/v1 kind: PriorityClass metadata:
name: high-priority value: 1999
globalDefault: false
description: "High priority class for user workloads" EOF
kubectl apply -f high-priority.yaml

# Patch the deployment
kubectl patch deployment busybox-logger -n priority \
# -type='merge' \
# p '{"spec": {"template": {"spec": {"priorityClassName": "high-priority"}}}}'
# Verify
kubectl get deployment busybox-logger -n priority -o jsonpath='{.spec.template.spec.priorityClassName}'

kubectl get priorityclass high-priority

## Question: 78

SIMULATION
You must connect to the correct host.
Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000060

Task

Install Argo CD in the cluster by performing the following tasks:
Add the official Argo CD Helm repository with the name argo

The Argo CD CRDs have already been pre-installed in the cluster
Generate a template of the Argo CD Helm chart version 7.7.3 for the argocd namespace and save it to ~/argo-helm.yaml
. Configure the chart to not install CRDs.

Answer: See the solution below.

Explanation:

Task Summary

SSH into cka000060

Add the Argo CD Helm repo named argo

Generate a manifest (~/argo-helm.yaml) for Argo CD version 7.7.3

Target namespace: argocd

Do not install CRDs

Just generate, don't install

Step-by-Step Solution

1 QSH into the correct host

```
ssh cka000060
```

Required — skipping this = zero score

2 (2Add the Argo CD Helm repository

```
helm repo add argo https://argoproj.github.io/argo-helm
helm repo update
```

This adds the official Argo Helm chart source.

3 Qenerate Argo CD Helm chart template (version 7.7.3)

Use the helm template command to generate a manifest and write it to ~/argo-helm.yaml.

```
helm template argocd argo/argo-cd \
- -version 7.7.3 \
- -namespace argocd \
- -set crds.install=false \
- ~/argo-helm.yaml
```

argocd → Release name (can be anything; here it's same as the namespace)

- -set crds.install=false → Disables CRD installation
- ~/argo-helm.yaml → Save to required file

4 Oerify the generated file (optional but smart)

```
head ~/argo-helm.yaml
```

Check that it contains valid Kubernetes YAML and does not include CRDs.

Final Command Summary

```
ssh cka000060
helm repo add argo https://argoproj.github.io/argo-helm
helm repo update

helm template argocd argo/argo-cd \
- -version 7.7.3 \
- -namespace argocd \
- -set crds.install=false \
```

- ~/argo-helm.yaml

head ~/argo-helm.yaml # Optional verification

## Question: 79

SIMULATION

You must connect to the correct host.

Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000047

Task

A MariaDB Deployment in the mariadb namespace has been deleted by mistake. Your task is to restore the Deployment ensuring data persistence. Follow these steps:

Create a PersistentVolumeClaim (PVC ) named mariadb in the mariadb namespace with the following specifications:

Access mode ReadWriteOnce

Storage 250Mi

You must use the existing retained PersistentVolume (PV ).

Failure to do so will result in a reduced score.

There is only one existing PersistentVolume .

Edit the MariaDB Deployment file located at ~/mariadb-deployment.yaml to use PVC you created in the previous step.

Apply the updated Deployment file to the cluster.

Ensure the MariaDB Deployment is running and stable.

Answer: See the solution below.

Explanation:

Task Overview

You're restoring a MariaDB deployment in the mariadb namespace with persistent data.

Tasks:

SSH into cka000047

Create a PVC named mariadb:

Namespace: mariadb

Access mode: ReadWriteOnce

Storage: 250Mi

Use the existing retained PV (there's only one)

Edit ~/mariadb-deployment.yaml to use the PVC

Apply the deployment

Verify MariaDB is running and stable

Step-by-Step Solution

1 QSH into the correct host

ssh cka000047

Required — skipping = zero score

2 Qnspect the existing PersistentVolume

kubectl get pv

Identify the only existing PV, e.g.:

NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS

mariadb-pv 250Mi RWO Retain Available <none> manual

Ensure the status is Available, and it is not already bound to a claim.

3 Qreate the PVC to bind the retained PV

Create a file mariadb-pvc.yaml:

cat <<EOF > mariadb-pvc.yaml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: mariadb

namespace: mariadb

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 250Mi

volumeName: mariadb-pv # Match the PV name exactly

EOF

Apply the PVC:

kubectl apply -f mariadb-pvc.yaml

This binds the PVC to the retained PV.

4 Qditthe MariaDB Deployment YAML

Open the file:

nano ~/mariadb-deployment.yaml

Look under the spec.template.spec.containers.volumeMounts and spec.template.spec.volumes sections and update them like so:

Add this under the container:

yaml

CopyEdit

volumeMounts:

- name: mariadb-storage

mountPath: /var/lib/mysql

And under the pod spec:

volumes:

- name: mariadb-storage

persistentVolumeClaim:

claimName: mariadb

These lines mount the PVC at the MariaDB data directory.

5 (ZApply the updated Deployment

kubectl apply -f ~/mariadb-deployment.yaml

6 Oerify the Deployment is running and stable

kubectl get pods -n mariadb

kubectl describe pod -n mariadb <mariadb-pod-name>

Ensure the pod is in Running state and volume is mounted.

Final Command Summary

ssh cka000047

kubectl get pv # Find the retained PV

# Create PVC

cat <<EOF > mariadb-pvc.yaml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: mariadb

namespace: mariadb

spec:

accessModes:

# ReadWriteOnce

resources:

requests:

storage: 250Mi

volumeName: mariadb-pv

EOF

kubectl apply -f mariadb-pvc.yaml

# Edit Deployment

nano ~/mariadb-deployment.yaml

# Add volumeMount and volume to use the PVC as described

kubectl apply -f ~/mariadb-deployment.yaml

kubectl get pods -n mariadb

## Question: 80

SIMULATION

You must connect to the correct host.

Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000056

Task

Review and apply the appropriate NetworkPolicy from the provided YAML samples.

Ensure that the chosen NetworkPolicy is not overly permissive, but allows communication between the frontend and backend Deployments, which run in the frontend and backend namespaces respectively.

First, analyze the frontend and backend Deployments to determine the specific requirements for the NetworkPolicy that needs to be applied.

Next, examine the NetworkPolicy YAML samples located in the ~/netpol folder.

Failure to comply may result in a reduced score.

Do not delete or modify the provided samples. Only apply one of them.

Finally, apply the NetworkPolicy that enables communication between the frontend and backend Deployments, without being overly permissive.

Answer: See the solution below.

Explanation:

Task Summary

Connect to host cka000056

Review existing frontend and backend Deployments

Choose one correct NetworkPolicy from the ~/netpol directory

The policy must:

Allow traffic only from the frontend Deployment to the backend Deployment

Avoid being overly permissive

Apply the correct NetworkPolicy without modifying any sample files

Step-by-Step Instructions

Step 1: SSH into the correct node

ssh cka000056

Step 2: Inspect the frontend Deployment

Check the labels used in the frontend Deployment:

kubectl get deployment -n frontend -o yaml

Look under metadata.labels or spec.template.metadata.labels. Note the app or similar label (e.g., app: frontend).

Step 3: Inspect the backend Deployment

kubectl get deployment -n backend -o yaml

Again, find the labels assigned to the pods (e.g., app: backend).

Step 4: List and review the provided NetworkPolicies

List the available files:

ls ~/netpol

Check the contents of each policy file:

cat ~/netpol/<file-name>.yaml

Look for a policy that:

Has kind: NetworkPolicy

Applies to the backend namespace

Uses a podSelector that matches the backend pods

Includes an ingress.from rule that references the frontend namespace using a namespaceSelector (and optionally a podSelector)

Does not allow traffic from all namespaces or all pods

Here's what to look for in a good match:

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-frontend-to-backend
  namespace: backend
spec:
  podSelector:
    matchLabels:
      app: backend
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          name: frontend
```

Even better if the policy includes:

```
  - namespaceSelector:
      matchLabels:
        name: frontend
    podSelector:
      matchLabels:
        app: frontend
```

This limits access to pods in the frontend namespace with a specific label.

Step 5: Apply the correct NetworkPolicy

Once you've identified the best match, apply it:

kubectl apply -f ~/netpol/<chosen-file>.yaml

Apply only one file. Do not alter or delete any existing sample.

ssh cka000056

kubectl get deployment -n frontend -o yaml
kubectl get deployment -n backend -o yaml

ls ~/netpol
cat ~/netpol/*.yaml # Review carefully

kubectl apply -f ~/netpol/<chosen-file>.yaml

Command Summary ssh cka000056

kubectl get deployment -n frontend -o yaml
kubectl get deployment -n backend -o yaml

ls ~/netpol
cat ~/netpol/*.yaml # Review carefully

kubectl apply -f ~/netpol/<chosen-file>.yaml

## Question: 81

SIMULATION

You must connect to the correct host.
Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000059

Context

A kubeadm provisioned cluster was migrated to a new machine. It needs configuration changes to run successfully.

Task

Fix a single-node cluster that got broken during machine migration.

First, identify the broken cluster components and investigate what breaks them.

The decommissioned cluster used an external etcd server.

Next, fix the configuration of all broken cluster

Answer: See the solution below.

Explanation:

Task Summary

SSH into node: cka000059

Cluster was migrated to a new machine

It uses an external etcd server

Identify and fix misconfigured components

Bring the cluster back to a healthy state

Step-by-Step Solution

Step 1: SSH into the correct host

ssh cka000059

Step 2: Check the cluster status

Run:

kubectl get nodes

If it fails, the kubelet or kube-apiserver is likely broken.

Check kubelet status:

sudo systemctl status kubelet

Also, check pod statuses in the control plane:

sudo crictl ps -a | grep kube or:

docker ps -a | grep kube

Look especially for failures in kube-apiserver or kube-controller-manager.

Step 3: Inspect the kube-apiserver manifest

Since this is a kubeadm-based cluster, manifests are in:

ls /etc/kubernetes/manifests

Open kube-apiserver.yaml:

bash

CopyEdit

sudo nano /etc/kubernetes/manifests/kube-apiserver.yaml

Look for the --etcd-servers= flag. If the external etcd endpoint has changed (likely, due to migration), this needs to be fixed.

Example of incorrect configuration:

```
# -etcd-servers=https://192.168.1.100:2379
```

If the IP has changed, update it to the correct IP or hostname of the external etcd server.

Also ensure the correct client certificate and key paths are still valid:

```
# -etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
# -etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt
# -etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
```

If the files are missing or the path is wrong due to migration, correct those as well.

Step 4: Save and exit, and let static pod restart

Static pod changes will be picked up automatically by the kubelet (watch for /etc/kubernetes/manifests changes).

Check again:

```
docker ps | grep kube-apiserver
#  or
crictl ps | grep kube-apiserver
```

Step 5: Confirm API is healthy

Once kube-apiserver is up, try:

```
kubectl get componentstatuses
kubectl get nodes
```

If these commands work and return valid statuses, the control plane is functional again.

Step 6: Check controller-manager and scheduler (optional)

If still broken, check the other static pods in /etc/kubernetes/manifests/ and correct paths if necessary.

Also verify that /etc/kubernetes/kubelet.conf and /etc/kubernetes/admin.conf are present and valid.

Command Summary

```
ssh cka000059

#  Check system and kubelet
sudo systemctl status kubelet
docker ps -a | grep kube # or crictl ps -a | grep kube

#  Check manifests
ls /etc/kubernetes/manifests
sudo nano /etc/kubernetes/manifests/kube-apiserver.yaml
#  Fix --etcd-servers and certificate paths if needed

#  Watch pods restart and confirm:
kubectl get nodes
kubectl get componentstatuses
```

# Question: 82

SIMULATION

You must connect to the correct host.

Failure to do so may result in a zero score.

[candidate@base] $ ssh Cka000055

Task

Verify the cert-manager application which has been deployed to your cluster .

Using kubectl, create a list of all cert-manager Custom Resource Definitions (CRDs ) and save it to ~/resources.yaml .

You must use kubectl 's default output format.

Do not set an output format.
Failure to do so will result in a reduced score.

Using kubectl, extract the documentation for the subject specification field of the Certificate Custom Resource and save it to ~/subject.yaml.

Answer: See the solution below.

Explanation:

Task Summary
You need to:
SSH into the correct node: cka000055
Use kubectl to list all cert-manager CRDs, and save that list to ~/resources.yaml

Do not use any output format flags like -o yaml
Extract the documentation for the spec.subject field of the Certificate custom resource and save it to ~/subject.yaml

Step-by-Step Instructions
Step 1: SSH into the node

ssh cka000055

Step 2: List cert-manager CRDs and save to a file
First, identify all cert-manager CRDs:

kubectl get crds | grep cert-manager
Then extract them without specifying an output format:

kubectl get crds | grep cert-manager | awk '{print $1}' | xargs kubectl get crd > ~/resources.yaml
This saves the default kubectl get output to the required file without formatting flags.

Step 3: Get documentation for spec.subject in the Certificate CRD
Run the following command:

kubectl explain certificate.spec.subject > ~/subject.yaml
This extracts the field documentation and saves it to the specified file.
If you're not sure of the resource, verify it exists:

kubectl get crd certificates.cert-manager.io

Final Command Summary

ssh cka000055

kubectl get crds | grep cert-manager | awk '{print $1}' | xargs kubectl get crd > ~/resources.yaml

kubectl explain certificate.spec.subject > ~/subject.yaml

## Question: 83

SIMULATION

Quick Reference

ConfigMaps,

Documentation Deployments,

Namespace

You must connect to the correct host . Failure to do so may result in a zero score.

[candidate@base] $ ssh cka000048b

Task

An NGINX Deployment named nginx-static is running in the nginx-static namespace. It is configured using a ConfigMap named nginx-config .

First, update the nginx-config ConfigMap to also allow TLSv1.2. connections.

You may re-create, restart, or scale resources as necessary.

You can use the following command to test the changes: [candidate@cka000048b] $ curl -- tls-max 1.2 https://web.k8s.local

Answer: See the solution below.

Explanation:

Task Summary

SSH into cka000048b

Update the nginx-config ConfigMap in the nginx-static namespace to allow TLSv1.2

Ensure the nginx-static Deployment picks up the new config

Verify the change using the provided curl command

Step-by-Step Instructions

Step 1: SSH into the correct host

ssh cka000048b

Step 2: Get the ConfigMap

kubectl get configmap nginx-config -n nginx-static -o yaml > nginx-config.yaml

Open the file for editing:

nano nginx-config.yaml

Look for the TLS configuration in the data field. You are likely to find something like:

ssl_protocols TLSv1.3;

Modify it to include TLSv1.2 as well:

ssl_protocols TLSv1.2 TLSv1.3;

Save and exit the file.

Now update the ConfigMap:

kubectl apply -f nginx-config.yaml

Step 3: Restart the NGINX pods to pick up the new ConfigMap

Pods will not reload a ConfigMap automatically unless it's mounted in a way that supports dynamic reload and the app is watching for it (NGINX typically doesn't by default).

The safest way is to restart the pods:

Option 1: Roll the deployment

kubectl rollout restart deployment nginx-static -n nginx-static

Option 2: Delete pods to force recreation

kubectl delete pod -n nginx-static -l app=nginx-static

Step 4: Verify using curl

Use the provided curl command to confirm that TLS 1.2 is accepted:

curl --tls-max 1.2 https://web.k8s.local

A successful response means the TLS configuration is correct.

Final Command Summary

ssh cka000048b

kubectl get configmap nginx-config -n nginx-static -o yaml > nginx-config.yaml nano nginx-config.yaml # Modify to include "ssl_protocols TLSv1.2 TLSv1.3;" kubectl apply -f nginx-config.yaml

kubectl rollout restart deployment nginx-static -n nginx-static # or
kubectl delete pod -n nginx-static -l app=nginx-static

curl --tls-max 1.2 https://web.k8s.local