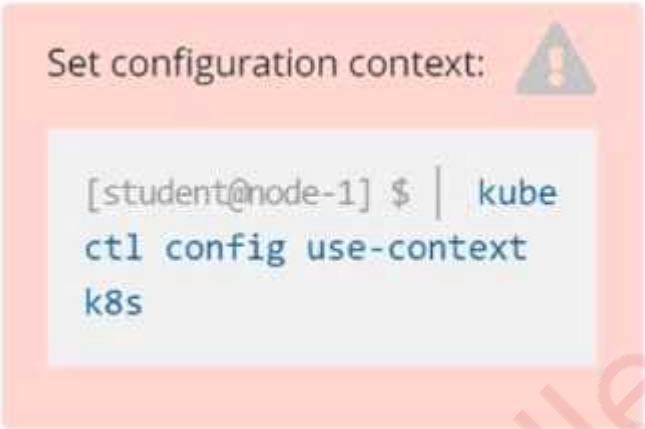
Linux Foundation

CKA Exam

Linux Foundation Certified Kubernetes Administrator Exam

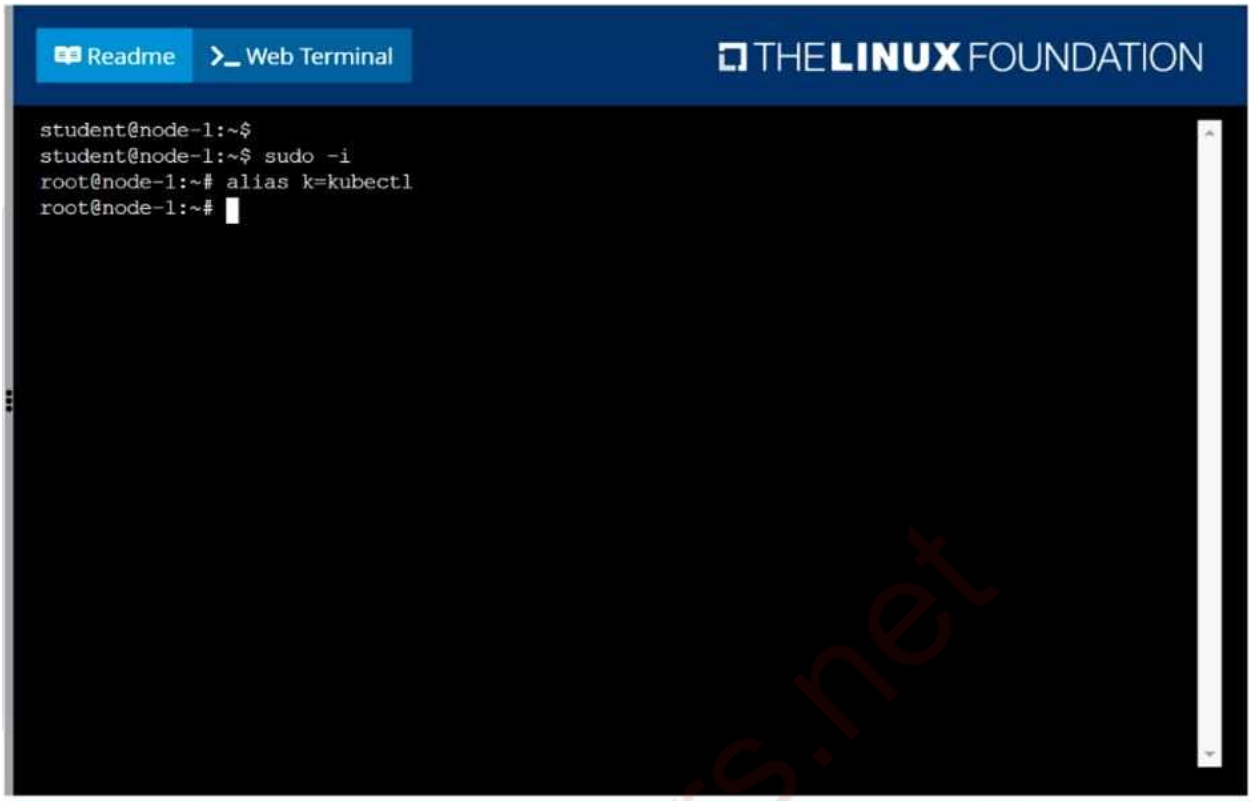
Question: 1

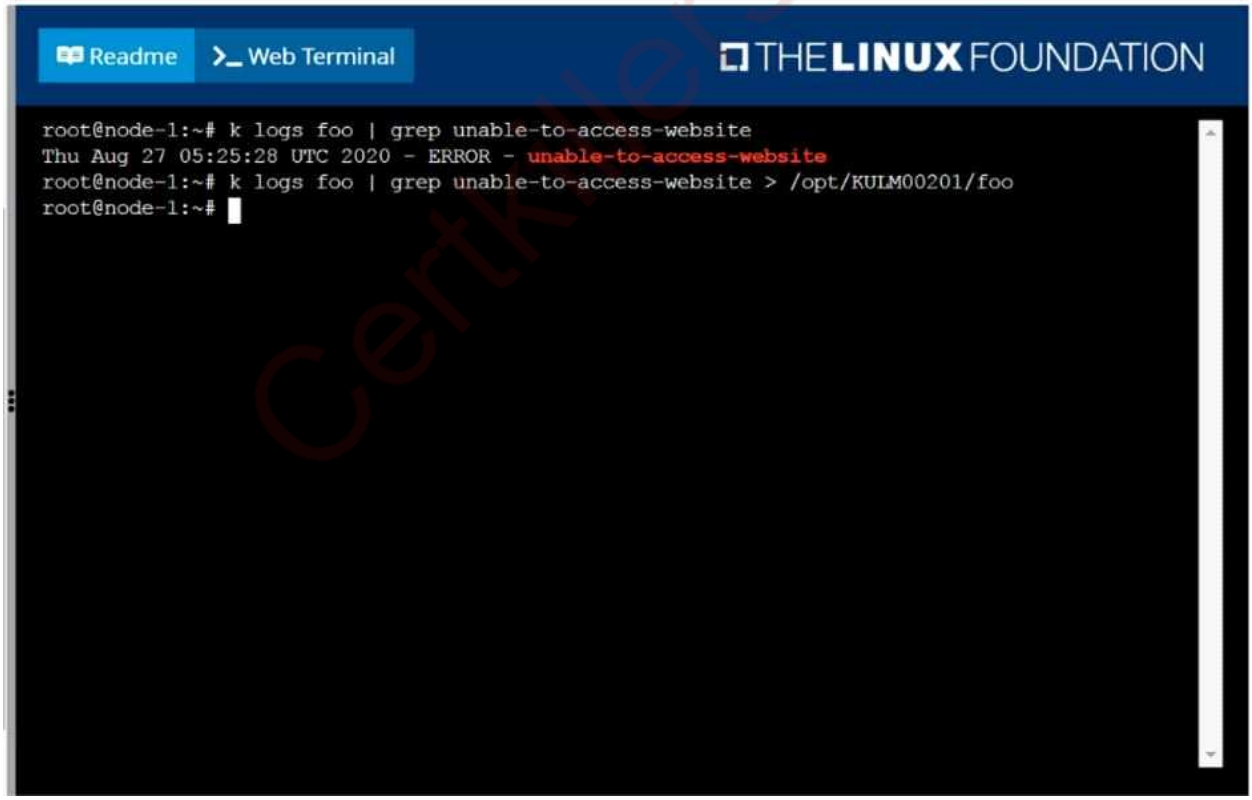
SIMULATION
Monitor the logs of pod foo and:
• Extract log lines corresponding to error unable-to-access-website
• Write them to /opt/KULM00201/foo



Solution

solution



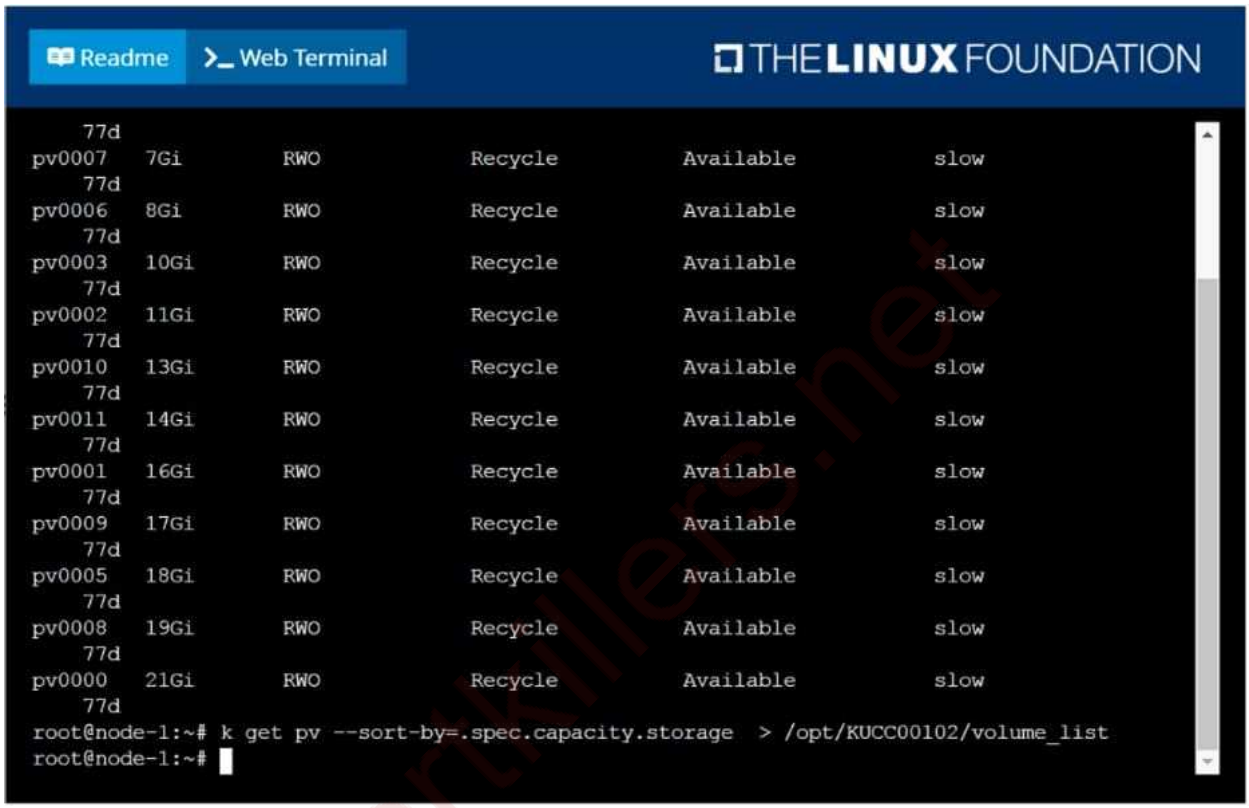


Question: 2

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.

Solution

solution

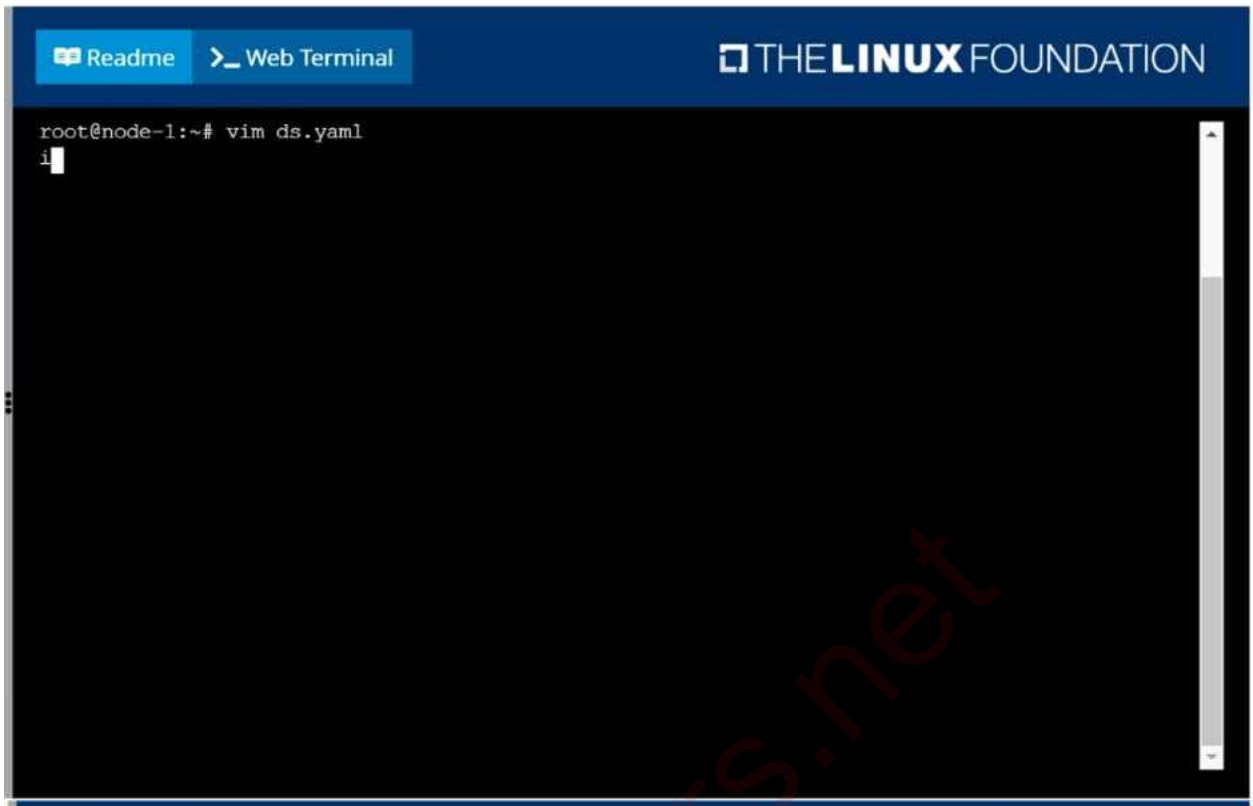


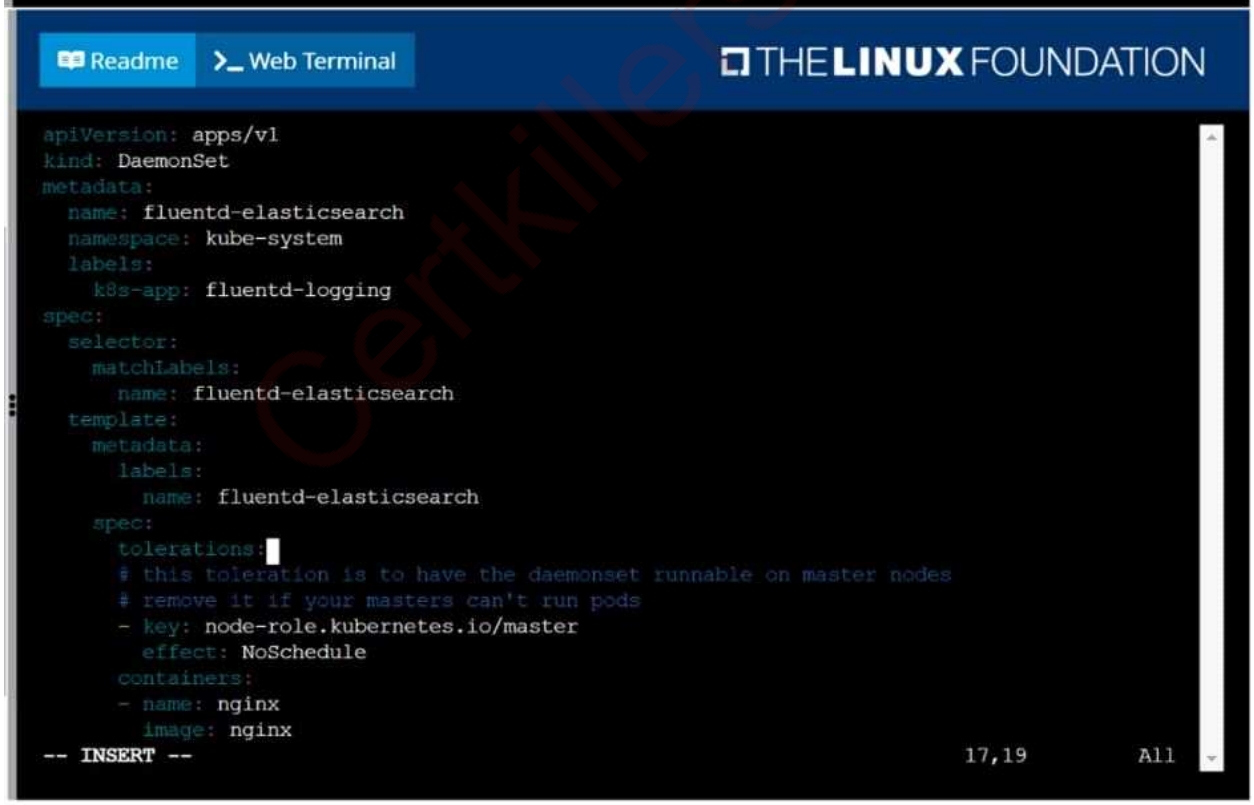
Question: 3

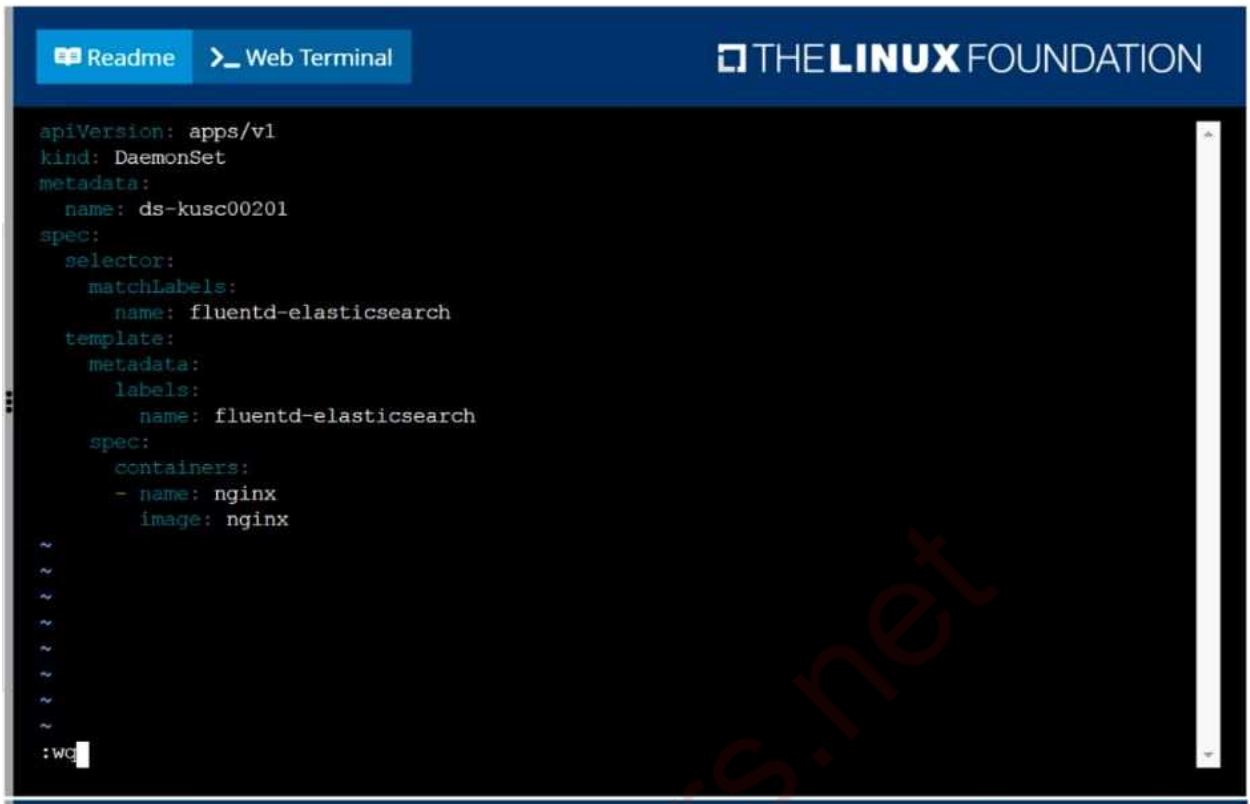
SIMULATION
Ensure a single instance of pod nginx is running on each node of the Kubernetes cluster where nginx also represents the Image name which has to be used. Do not override any taints currently in place. Use DaemonSet to complete this task and use ds-kusc00201 as DaemonSet name.

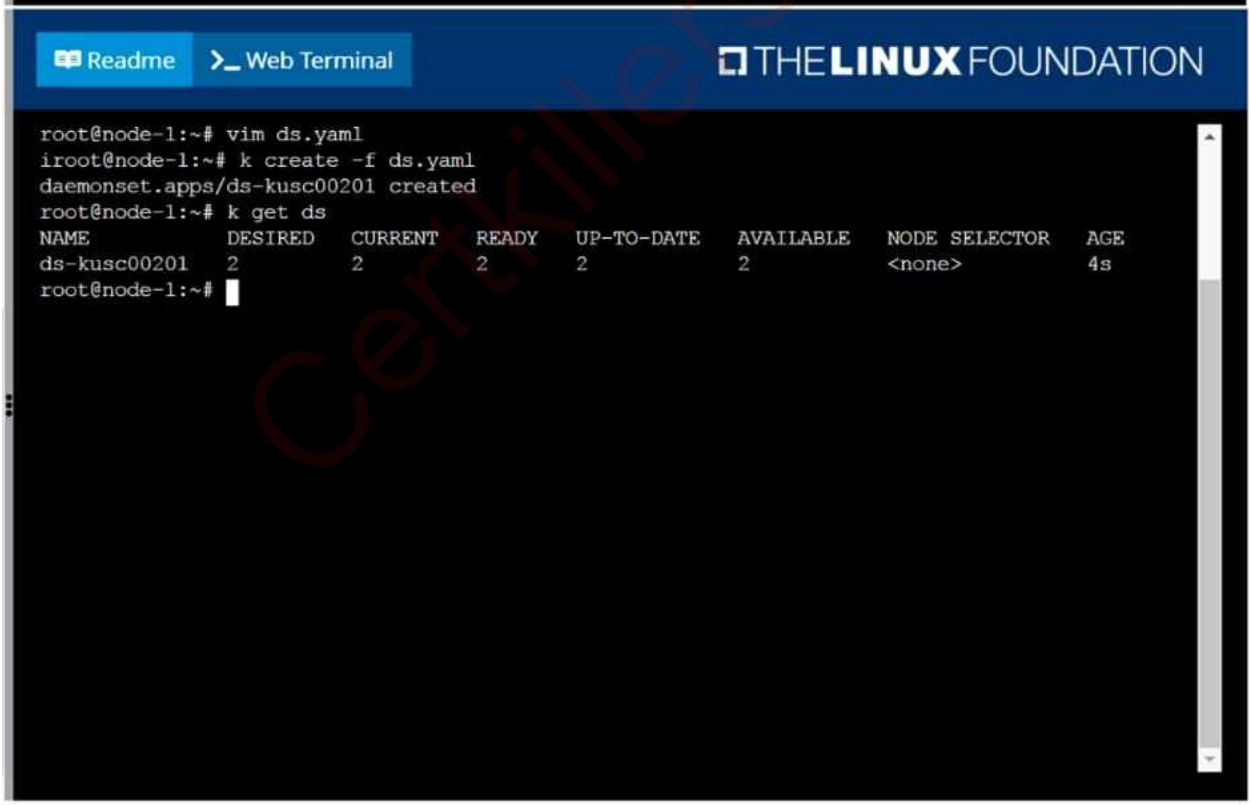
Solution

solution







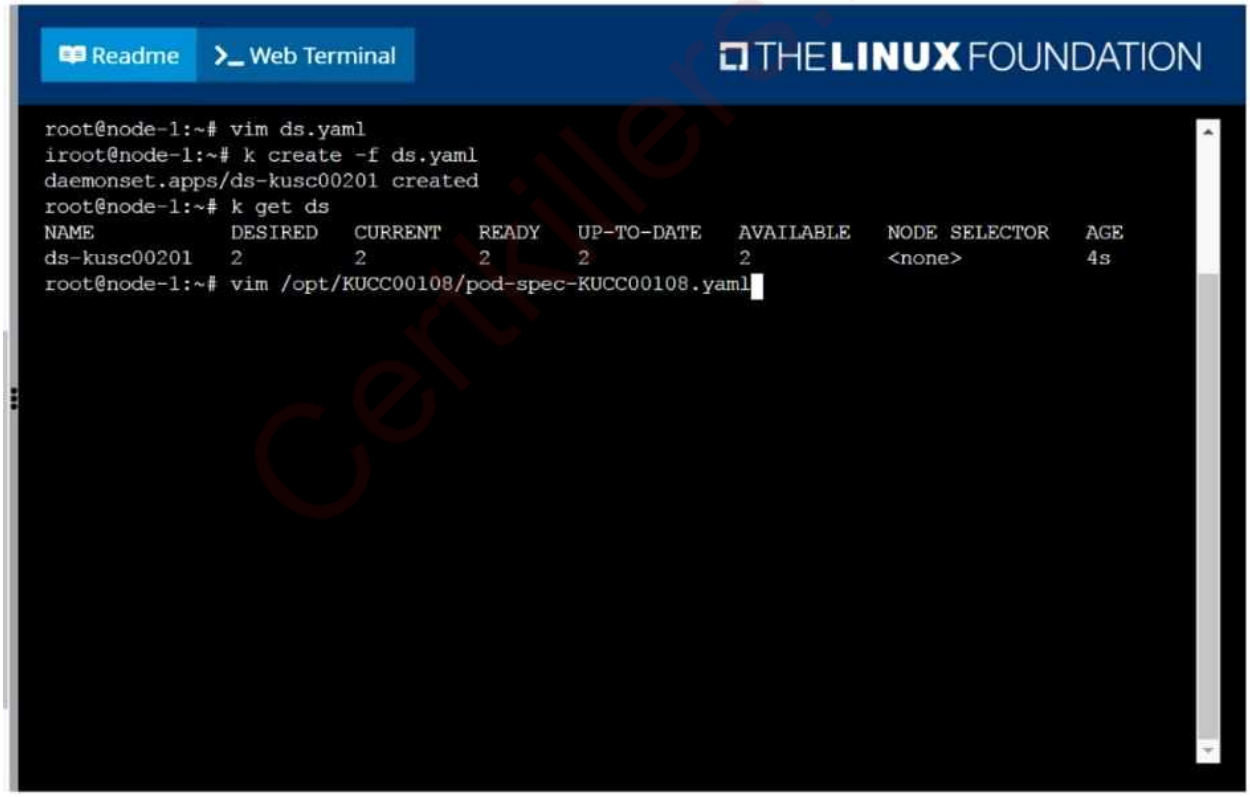


Question: 4

SIMULATION
Perform the following tasks:
• Add an init container to hungry-bear (which has been defined in spec file /opt/KUCC00108/pod-spec-KUC
C00108.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

Solution

solution



Readme | Web Terminal — THE LINUX FOUNDATION

```
apiVersion: v1
kind: Pod
metadata:
  name: hungry-bear
spec:
  volumes:
    - name: workdir
      emptyDir:
  containers:
  - name: checker
    image: alpine
    command: ["/bin/sh", "-c", "if [ -f /workdir/calm.txt ];
              then sleep 100000; else exit 1; fi"]
    volumeMounts:
    - name: workdir
      mountPath: /workdir
  initContainers:
  - name: create
    image: alpine
    command: ["/bin/sh", "-c", "touch /workdir/calm.txt"]
    volumeMounts:
    - name: workdir
      mountPath: /workdir
:wq
```

Readme | Web Terminal — THE LINUX FOUNDATION

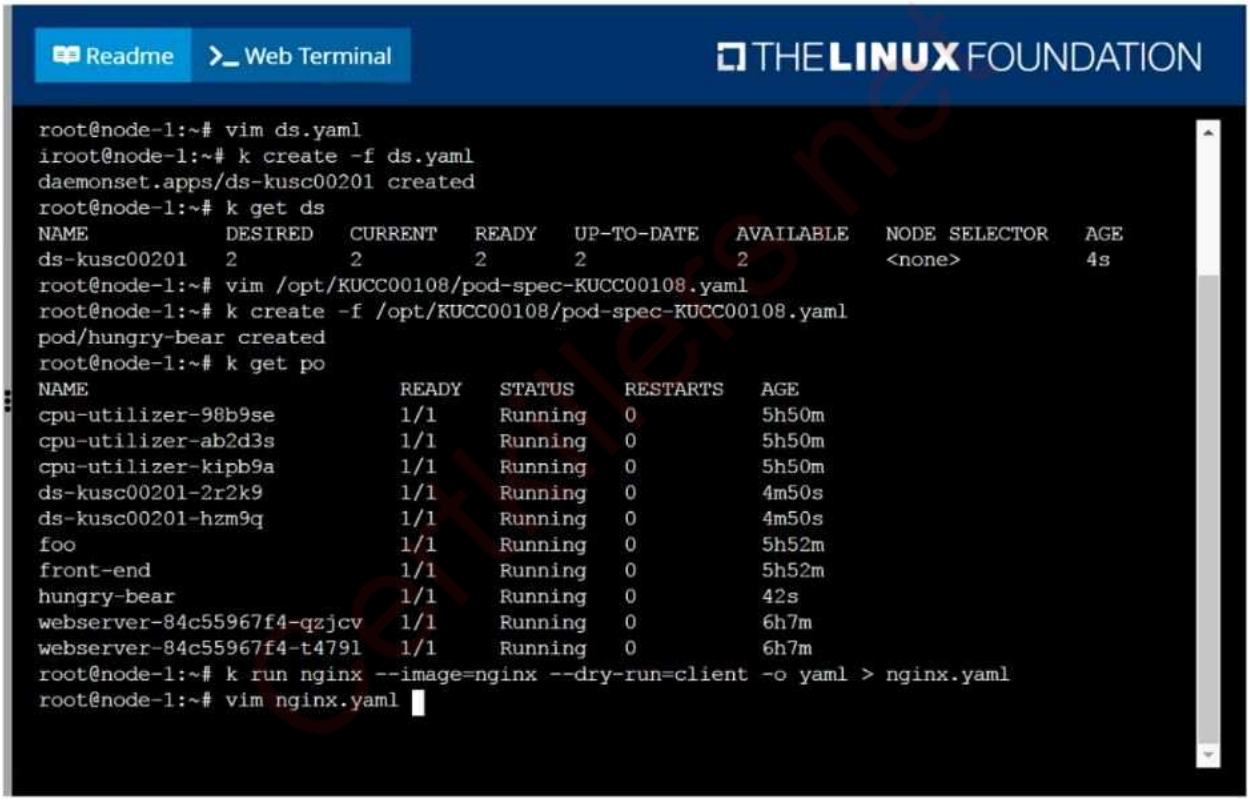
```
root@node-1:~# vim ds.yaml
iroot@node-1:~# k create -f ds.yaml
daemonset.apps/ds-kusc00201 created
root@node-1:~# k get ds
NAME           DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
ds-kusc00201   2         2         2       2            2           <none>          4s
root@node-1:~# vim /opt/KUCC00108/pod-spec-KUCC00108.yaml
root@node-1:~# k create -f /opt/KUCC00108/pod-spec-KUCC00108.yaml
pod/hungry-bear created
root@node-1:~#
```

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.

Solution

solution



Readme | Web Terminal — THE LINUX FOUNDATION

```
apiVersion: v1
kind: Pod
metadata:
  name: kucc8
spec:
  containers:
  - image: nginx
    name: nginx
  - image: redis
    name: redis
  - image: memcached
    name: memcached
~
~
~
~
~
~
~
~
~
~
~
~
:w
```

Readme | Web Terminal — THE LINUX FOUNDATION

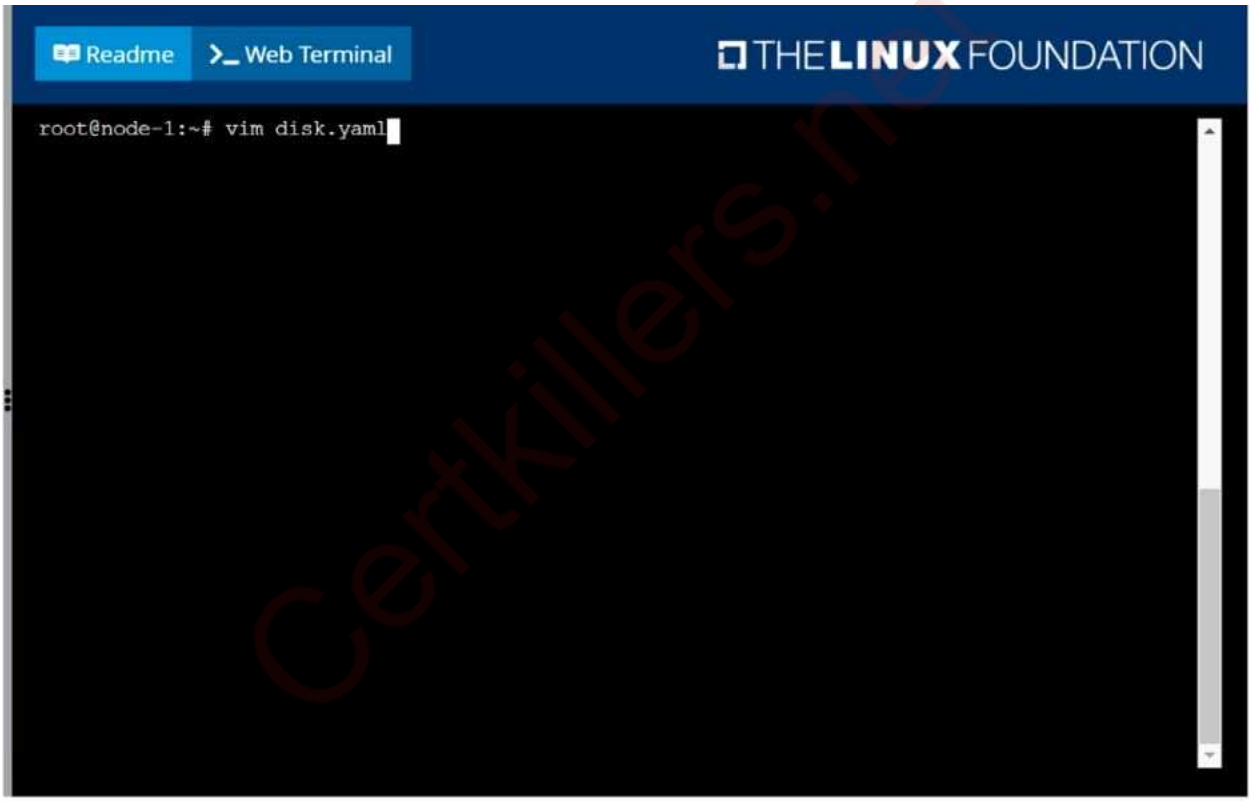
```
cpu-utilizer-98b9se          1/1     Running             0          5h51m
cpu-utilizer-ab2d3s          1/1     Running             0          5h51m
cpu-utilizer-kipb9a          1/1     Running             0          5h51m
ds-kusc00201-2r2k9           1/1     Running             0          6m12s
ds-kusc00201-hzm9q           1/1     Running             0          6m12s
foo                          1/1     Running             0          5h54m
front-end                    1/1     Running             0          5h53m
hungry-bear                  1/1     Running             0          2m4s
kucc8                        0/3     ContainerCreating   0          4s
webserver-84c55967f4-qzjcv   1/1     Running             0          6h9m
webserver-84c55967f4-t4791   1/1     Running             0          6h9m
root@node-1:~# k get po
NAME                         READY   STATUS    RESTARTS   AGE
cpu-utilizer-98b9se          1/1     Running   0          5h52m
cpu-utilizer-ab2d3s          1/1     Running   0          5h52m
cpu-utilizer-kipb9a          1/1     Running   0          5h52m
ds-kusc00201-2r2k9           1/1     Running   0          6m31s
ds-kusc00201-hzm9q           1/1     Running   0          6m31s
foo                          1/1     Running   0          5h54m
front-end                    1/1     Running   0          5h54m
hungry-bear                  1/1     Running   0          2m23s
kucc8                        3/3     Running   0          23s
webserver-84c55967f4-qzjcv   1/1     Running   0          6h9m
webserver-84c55967f4-t4791   1/1     Running   0          6h9m
root@node-1:~#
```

Question: 6

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

Solution

solution



Readme | Web Terminal — THE LINUX FOUNDATION

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-kusc00101
spec:
  containers:
  - name: nginx
    image: nginx
    imagePullPolicy: IfNotPresent
  nodeSelector:
    disk: ssd
~
~
~
~
~
~
~
~
~
~
~
~
~
"disk.yaml" [New] 11L, 176C written
```

Readme | Web Terminal — THE LINUX FOUNDATION

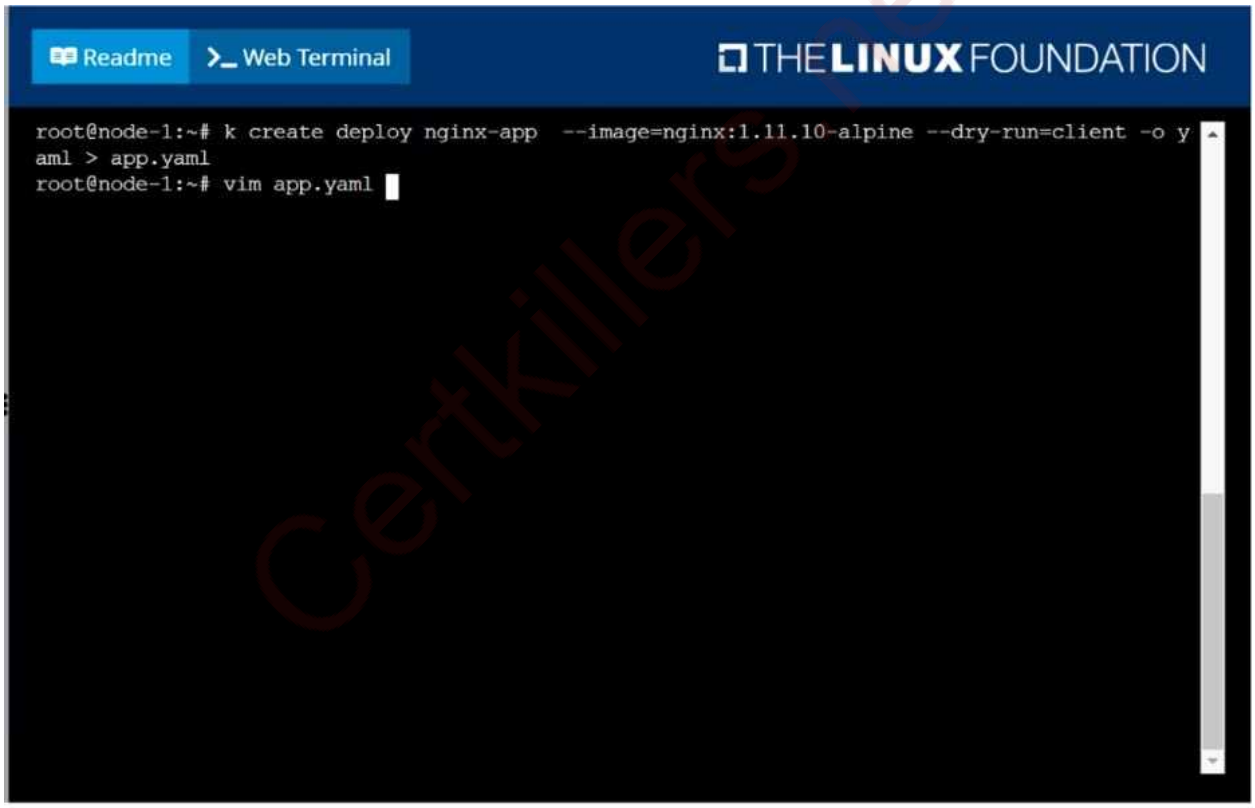
```
root@node-1:~# vim disk.yaml
root@node-1:~# k create -f disk.yaml
pod/nginx-kusc00101 created
root@node-1:~# k get po
NAME                          READY   STATUS    RESTARTS   AGE
cpu-utilizer-98b9se           1/1     Running   0          5h59m
cpu-utilizer-ab2d3s           1/1     Running   0          5h59m
cpu-utilizer-kipb9a           1/1     Running   0          5h59m
ds-kusc00201-2r2k9            1/1     Running   0          13m
ds-kusc00201-hzm9q            1/1     Running   0          13m
foo                           1/1     Running   0          6h1m
front-end                     1/1     Running   0          6h1m
hungry-bear                   1/1     Running   0          9m37s
kucc8                         3/3     Running   0          7m37s
nginx-kusc00101               1/1     Running   0          9s
webserver-84c55967f4-qzjcv    1/1     Running   0          6h16m
webserver-84c55967f4-t479l    1/1     Running   0          6h16m
root@node-1:~#
```

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.

Solution

solution



Readme Web Terminal THE LINUX FOUNDATION

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx-app
  template:
    metadata:
      labels:
        app: nginx-app
    spec:
      containers:
      - image: nginx:1.11.10-alpine
        name: nginx
~
~
~
~
~
~
~
"app.yaml
```

Readme Web Terminal THE LINUX FOUNDATION

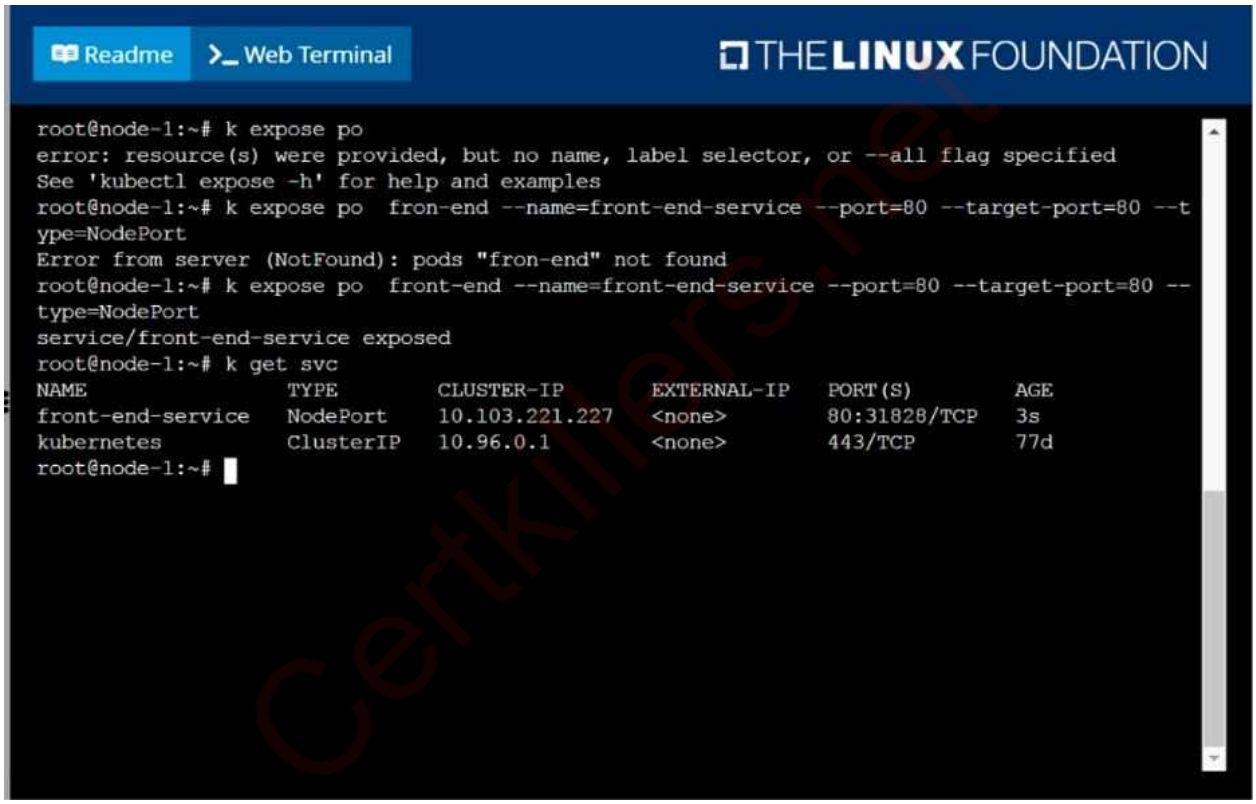
```
root@node-1:~# k create deploy nginx-app  --image=nginx:1.11.10-alpine --dry-run=client -o y
aml > app.yaml
root@node-1:~# vim app.yaml
root@node-1:~# k create -f app.yaml
deployment.apps/nginx-app created
root@node-1:~#
root@node-1:~#
root@node-1:~# k set image deploy nginx-app nginx=nginx:1.11.13-alpine  --record
deployment.apps/nginx-app image updated
root@node-1:~# k rollout undo deploy nginx-app
deployment.apps/nginx-app rolled back
root@node-1:~#
```

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.

Solution

solution

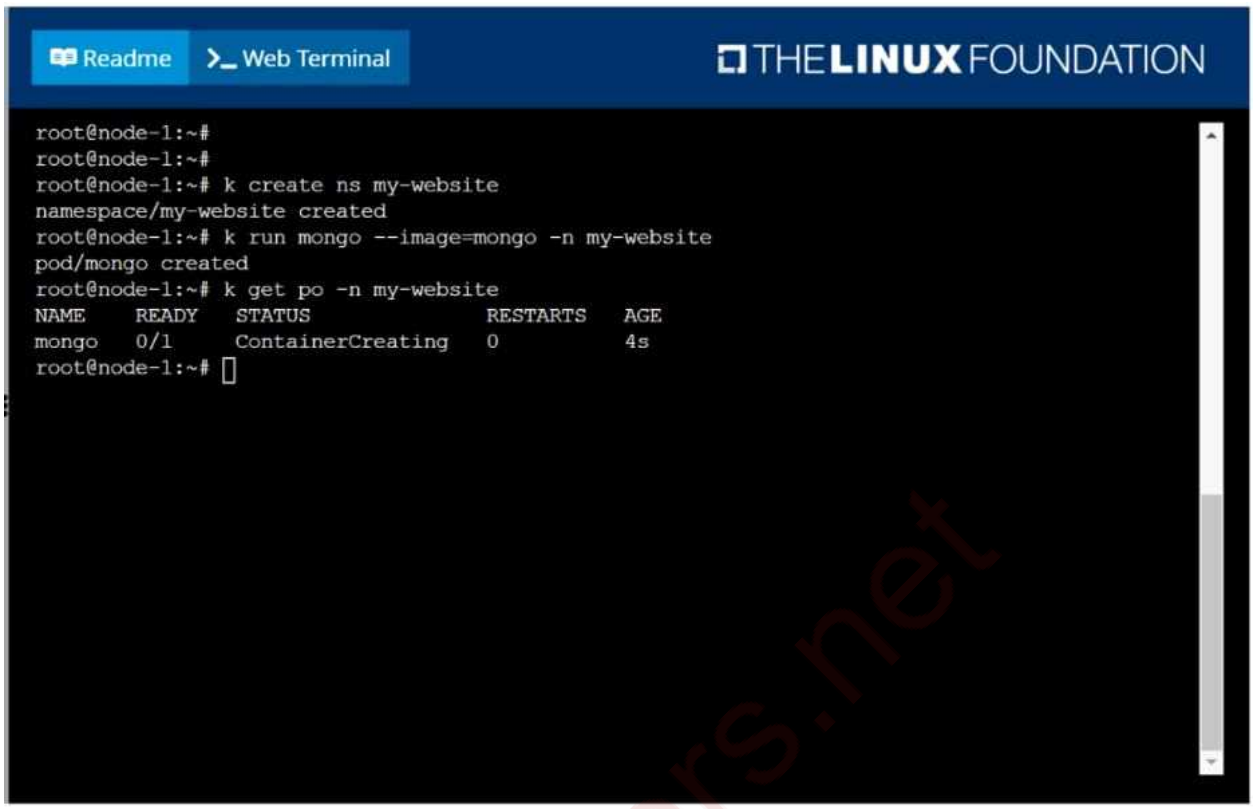


Question: 9

SIMULATION
Create a pod as follows:
• Name: mongo
• Using Image: mongo
• In a new Kubernetes namespace named: my-website

Solution

solution



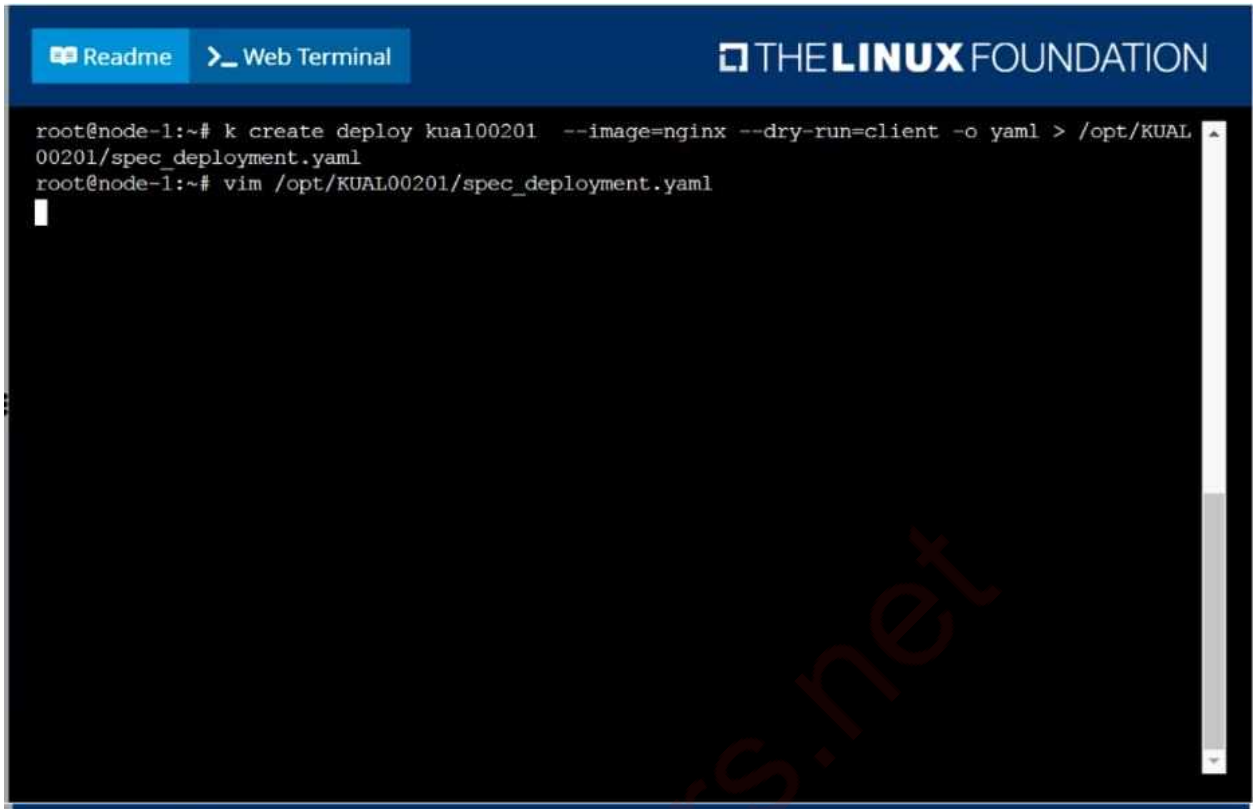
Question: 10

SIMULATION

Create a deployment spec file that will:
• Launch 7 replicas of the nginx Image with the label
app_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.

Solution

solution

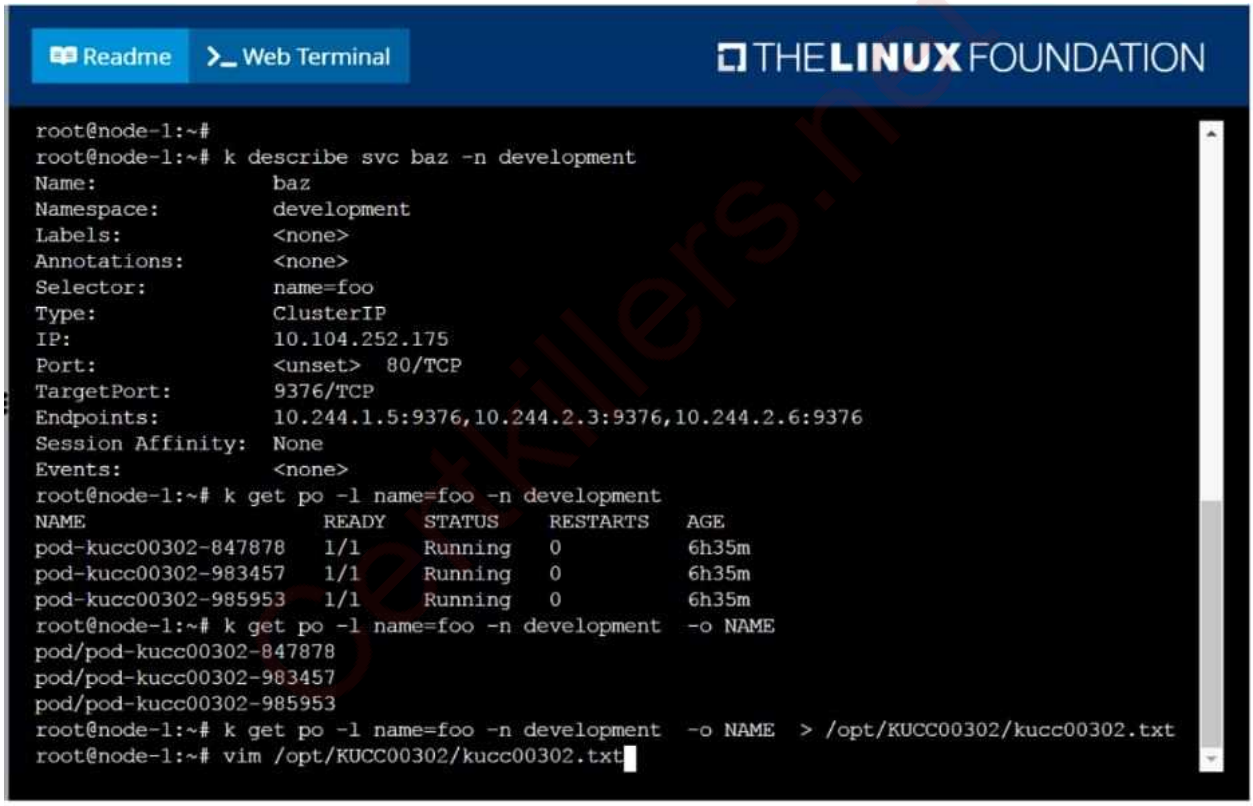


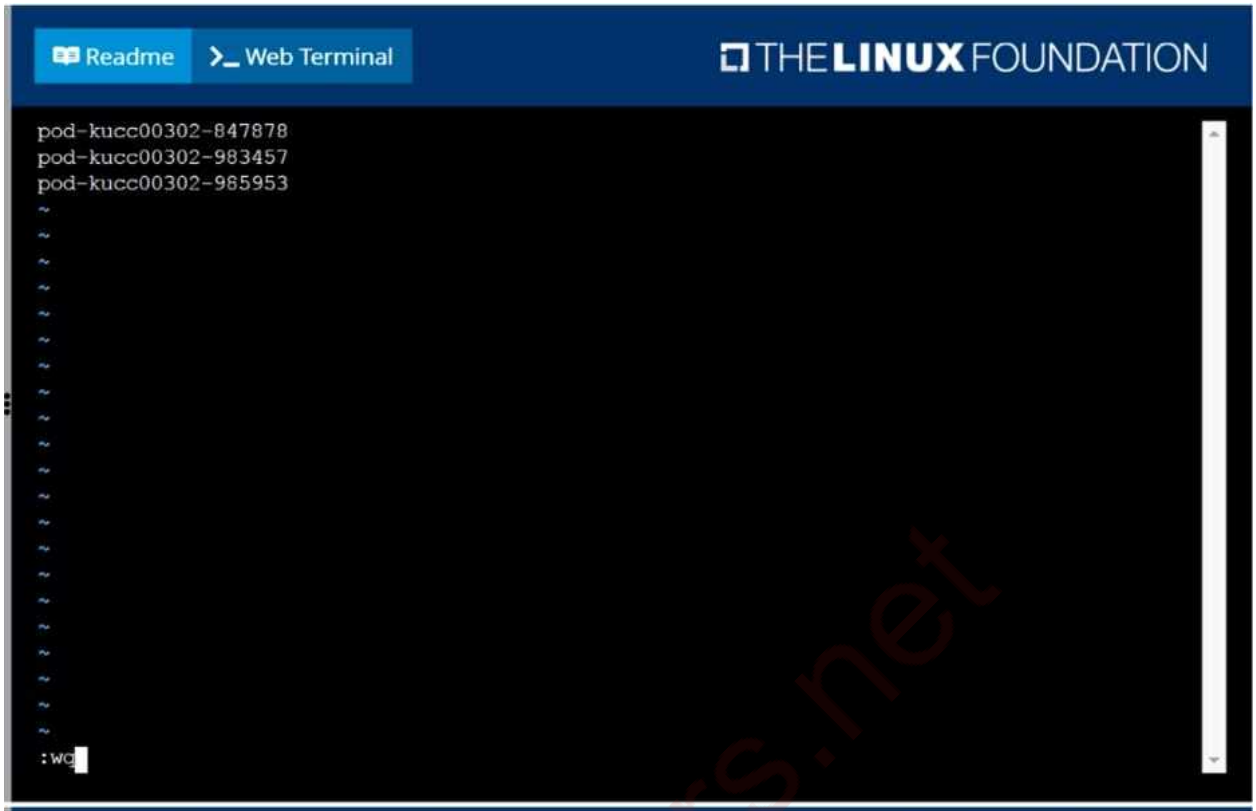
Question: 11

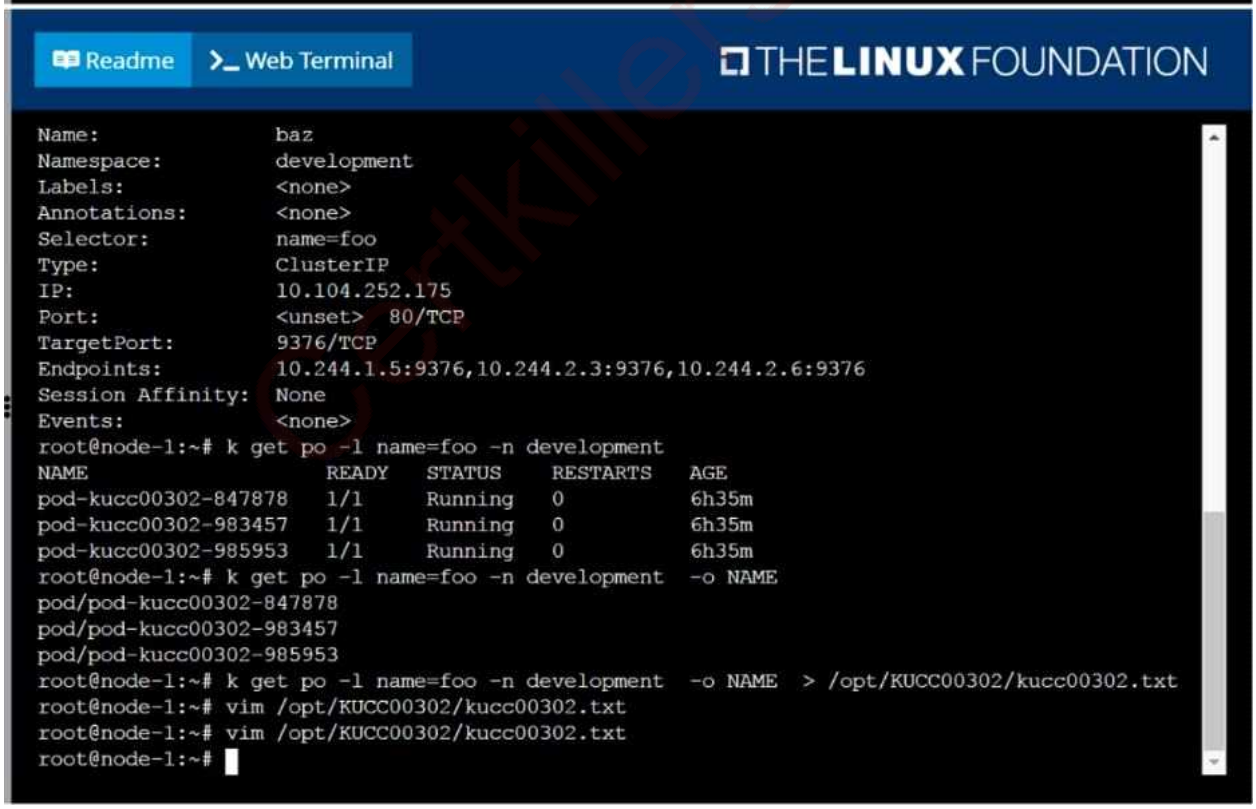
SIMULATION
Create a file:
/opt/KUCC00302/kucc00302.txt that lists all pods that implement service baz in namespace development.
The format of the file should be one pod name per line.

Solution

solution




Readme
Web Terminal
THE LINUX FOUNDATION
pod-kucc00302-847878
pod-kucc00302-983457
pod-kucc00302-985953
:wq

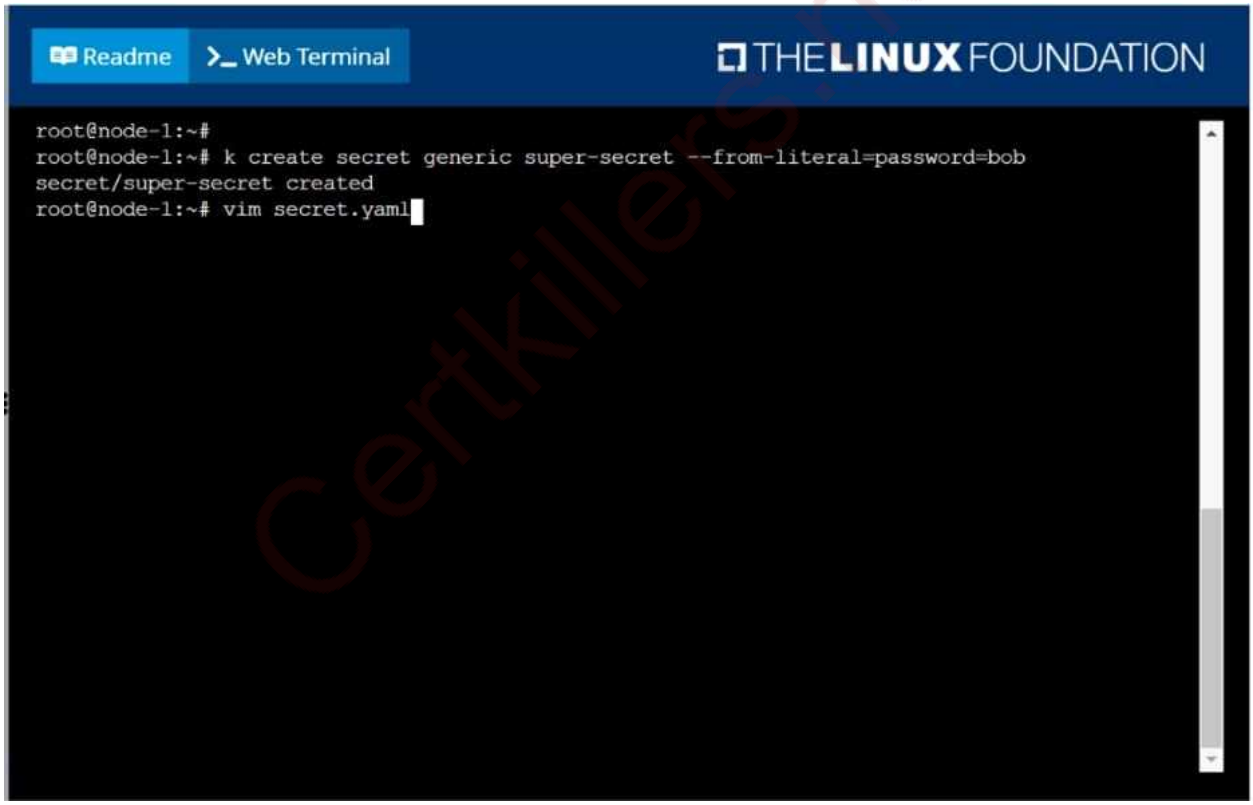

Readme
Web Terminal
THE LINUX FOUNDATION
Name: baz
Namespace: development
Labels: <none>
Annotations: <none>
Selector: name=foo
Type: ClusterIP
IP: 10.104.252.175
Port: <unset> 80/TCP
TargetPort: 9376/TCP
Endpoints: 10.244.1.5:9376,10.244.2.3:9376,10.244.2.6:9376
Session Affinity: None
Events: <none>
root@node-1:~# k get po -l 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
root@node-1:~# k get po -l name=foo -n development -o NAME
pod/pod-kucc00302-847878
pod/pod-kucc00302-983457
pod/pod-kucc00302-985953
root@node-1:~# k get po -l name=foo -n development -o NAME > /opt/KUCC00302/kucc00302.txt
root@node-1:~# vim /opt/KUCC00302/kucc00302.txt
root@node-1:~# vim /opt/KUCC00302/kucc00302.txt
root@node-1:~#

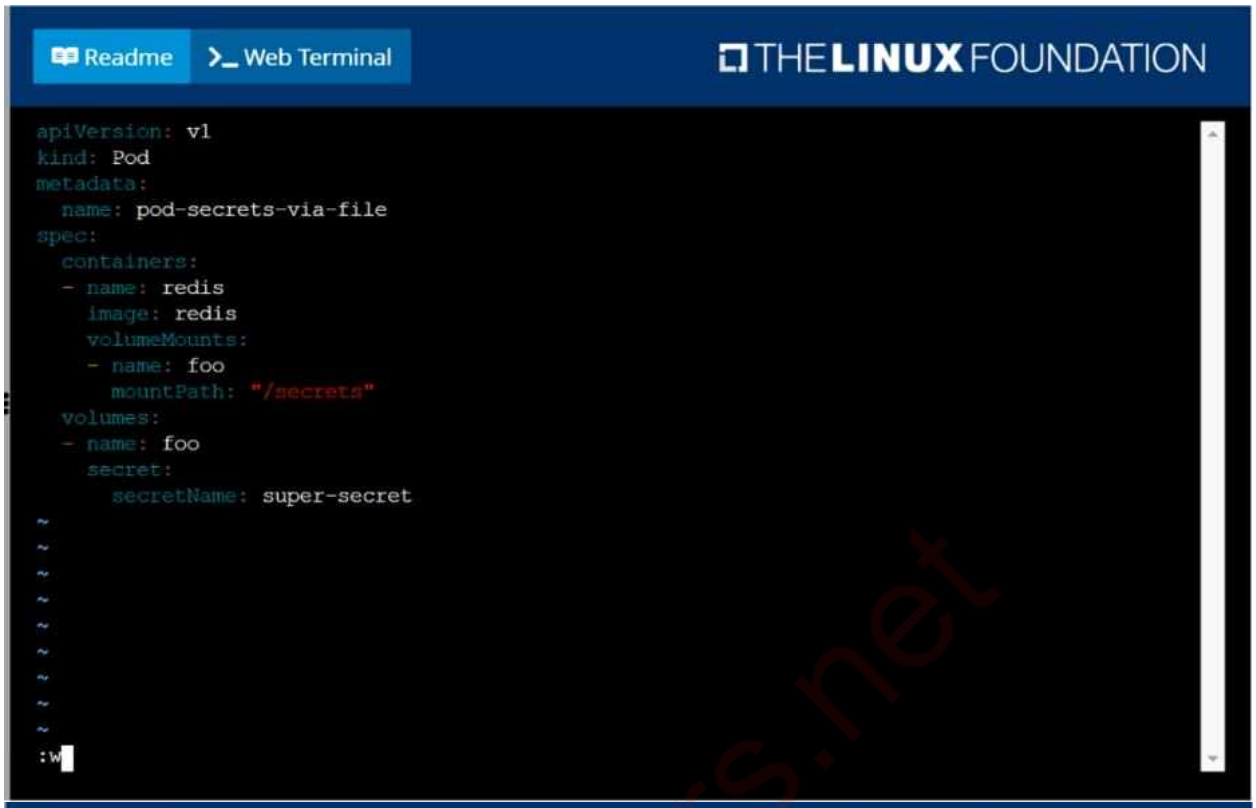
Question: 12

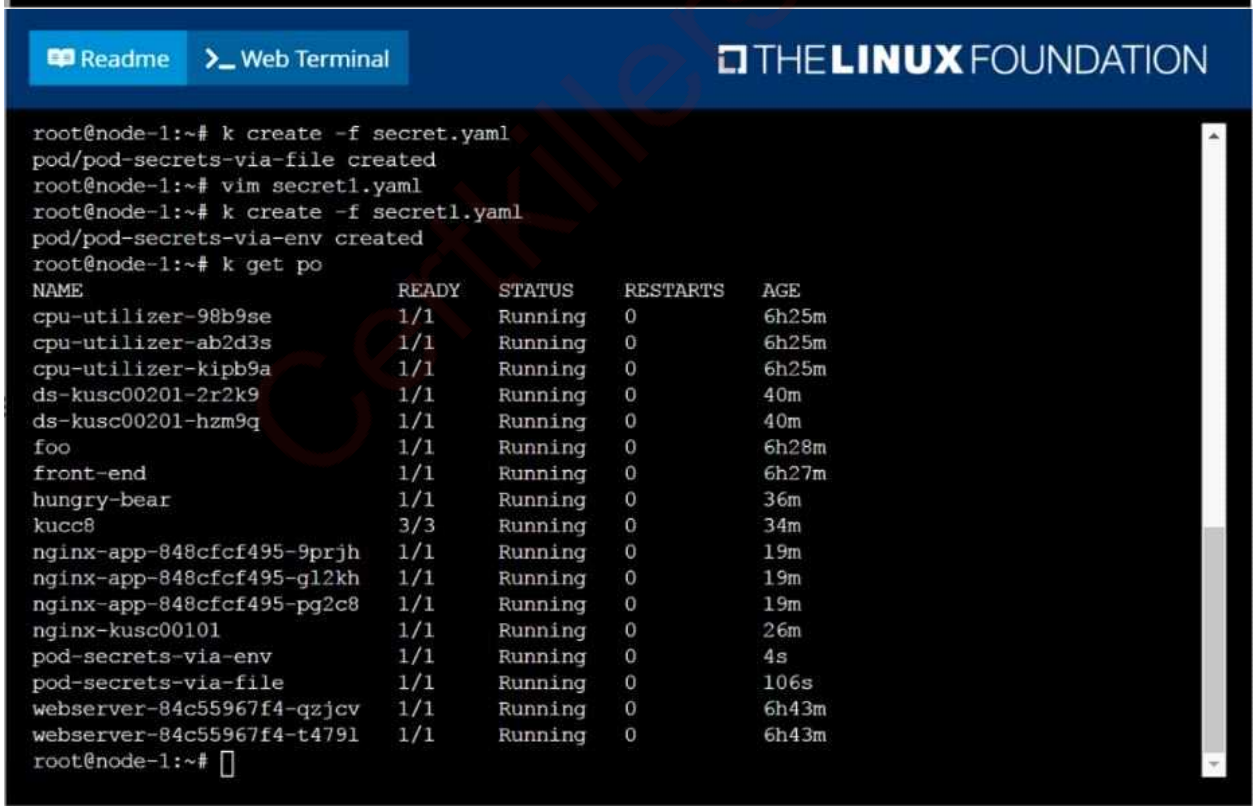
SIMULATION
Create a Kubernetes secret as follows:
• Name: super-secret
• password: bob
Create a pod named pod-secrets-via-file, using the redis Image, which mounts a secret named super-secret at /secrets.
Create a second pod named pod-secrets-via-env, using the redis Image, which exports password as CONFIDENTIAL

Solution

solution





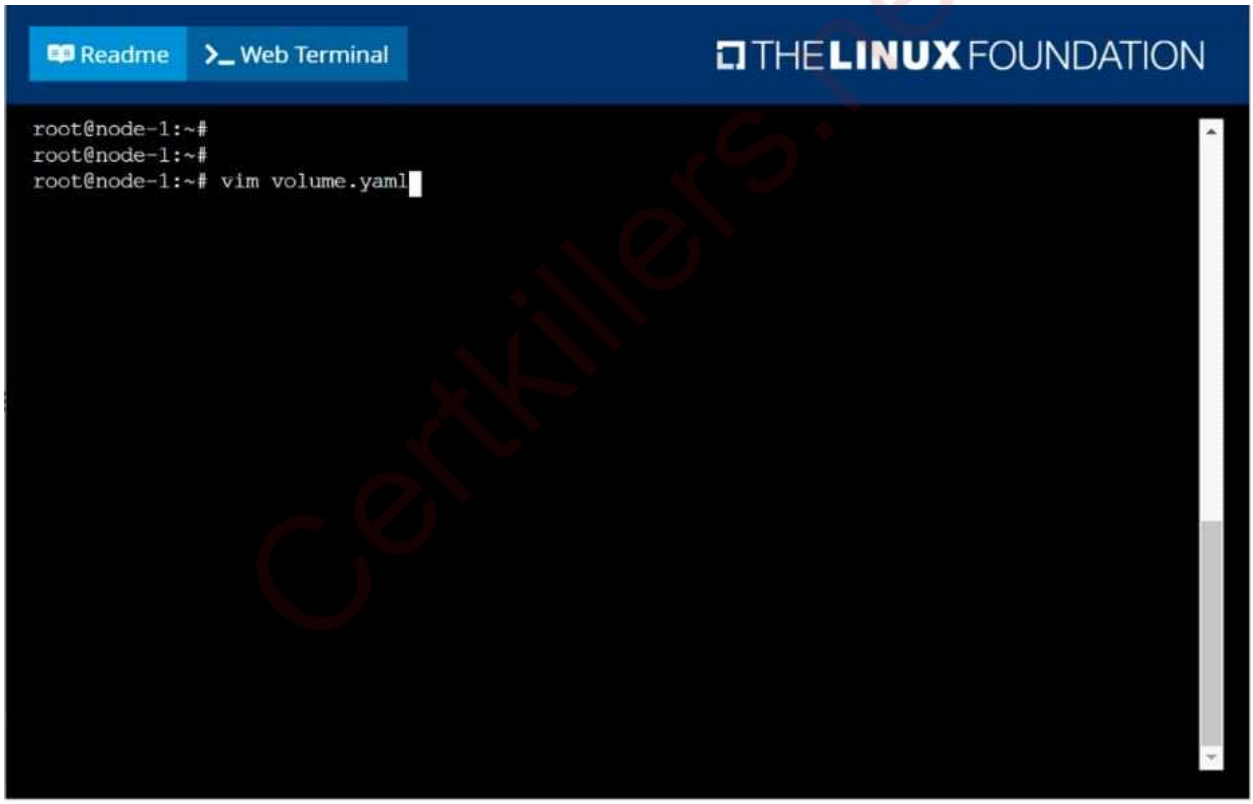


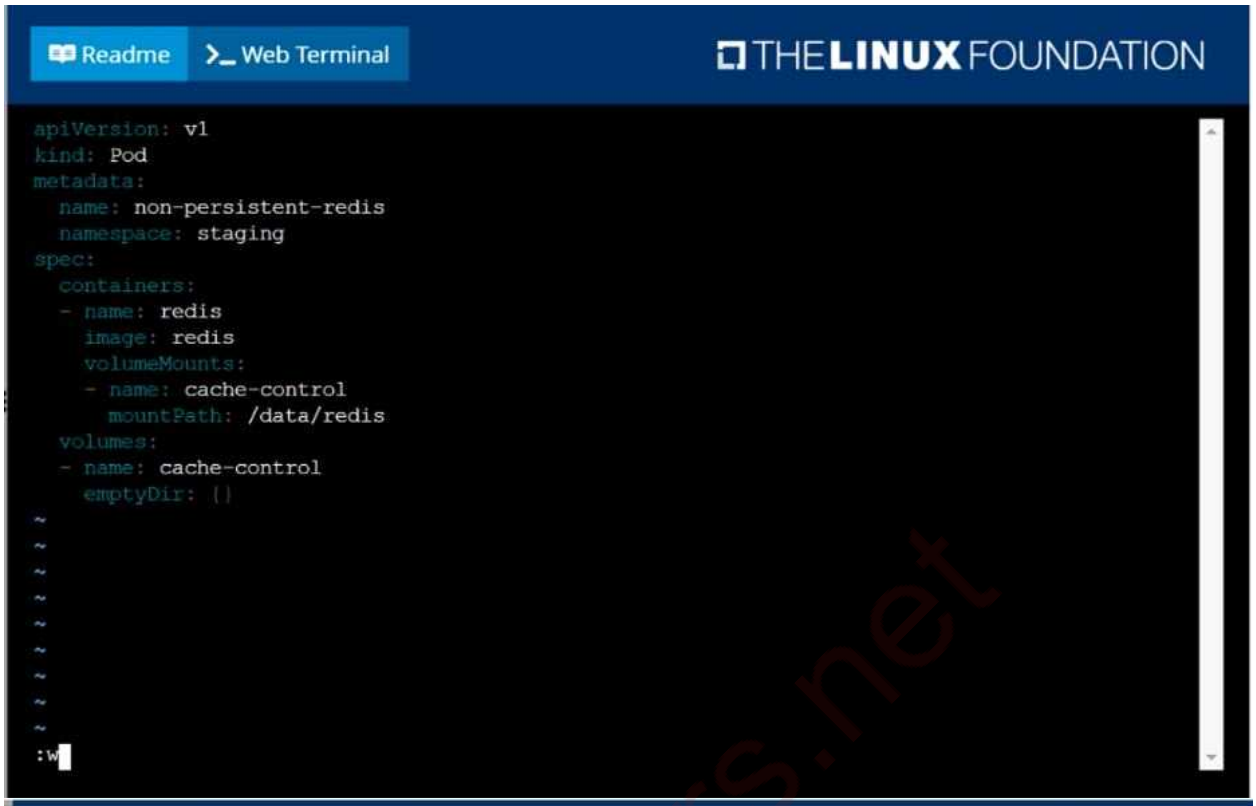
Question: 13

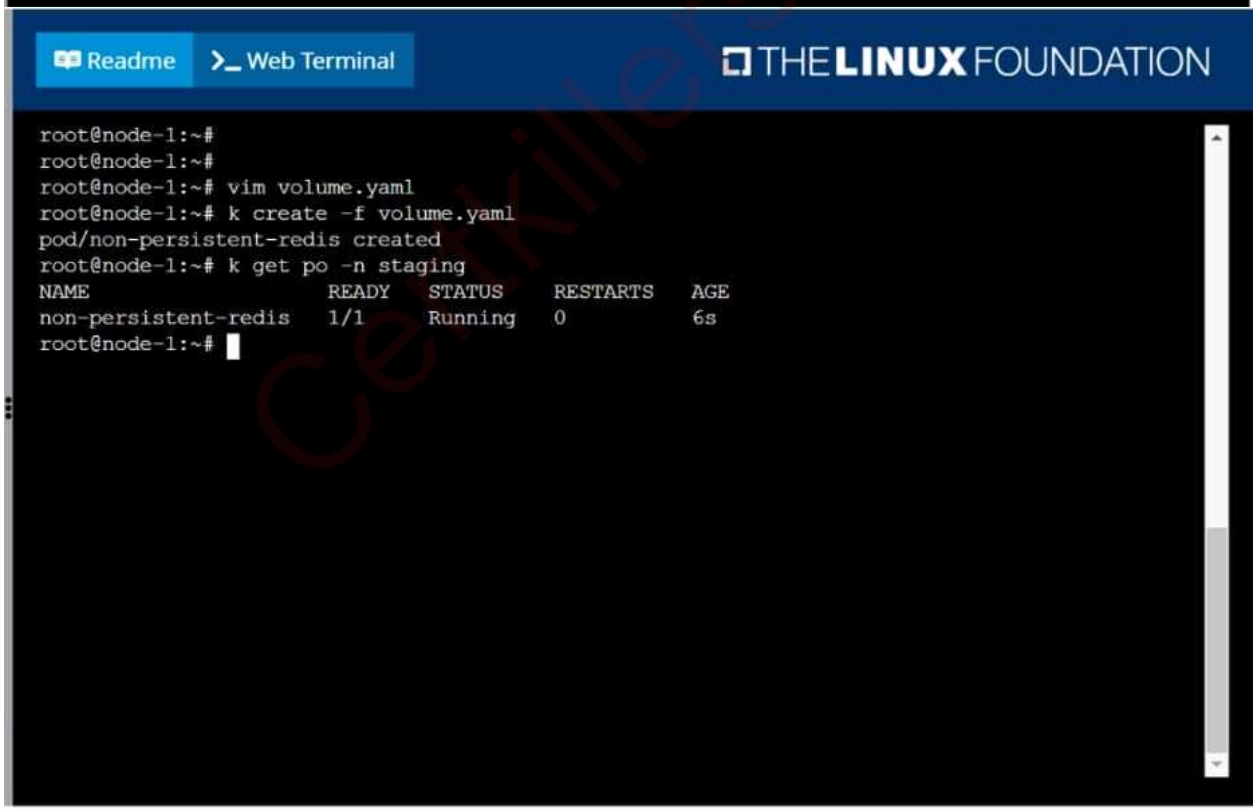
SIMULATION
Create a pod as follows:
• Name: non-persistent-redis
• container Image: redis
• Volume with name: cache-control
• Mount path: /data/redis
The pod should launch in the staging namespace and the volume must not be persistent.

Solution

solution





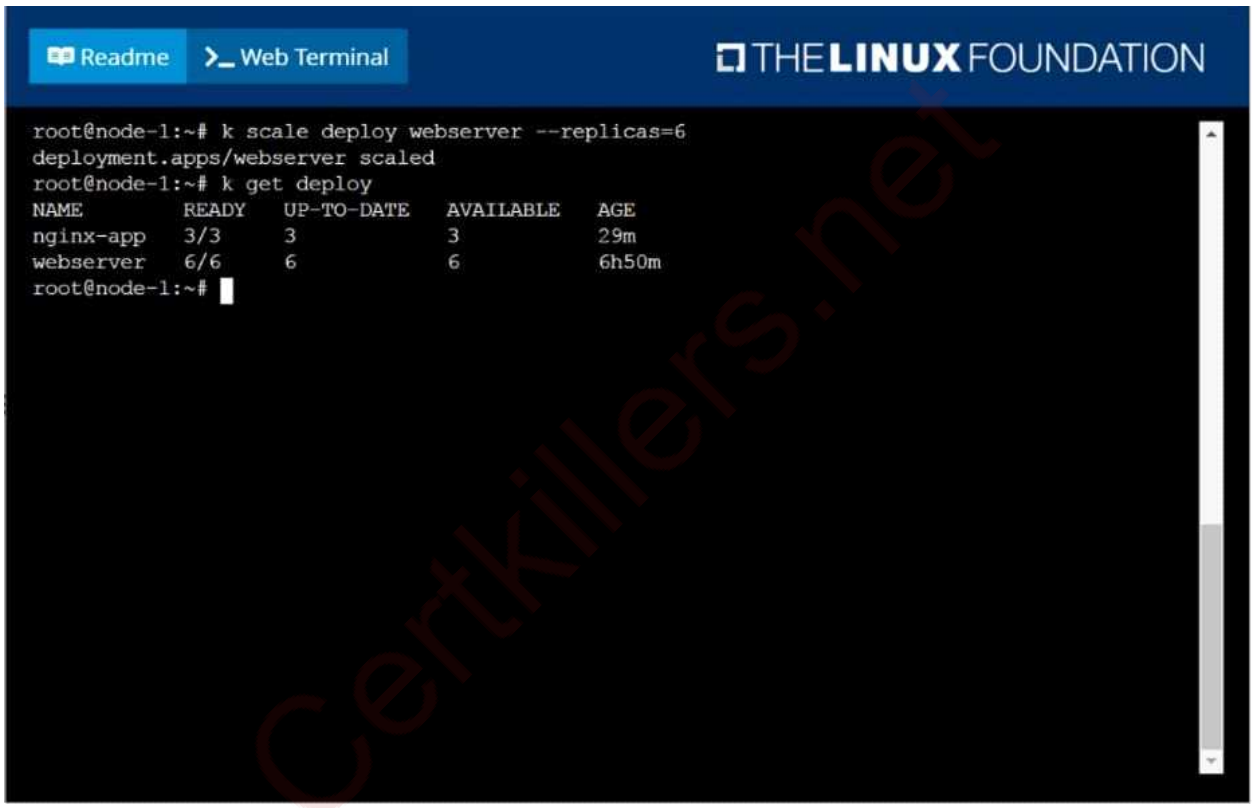


Question: 14

SIMULATION
Scale the deployment webserver to 6 pods.

Solution

solution

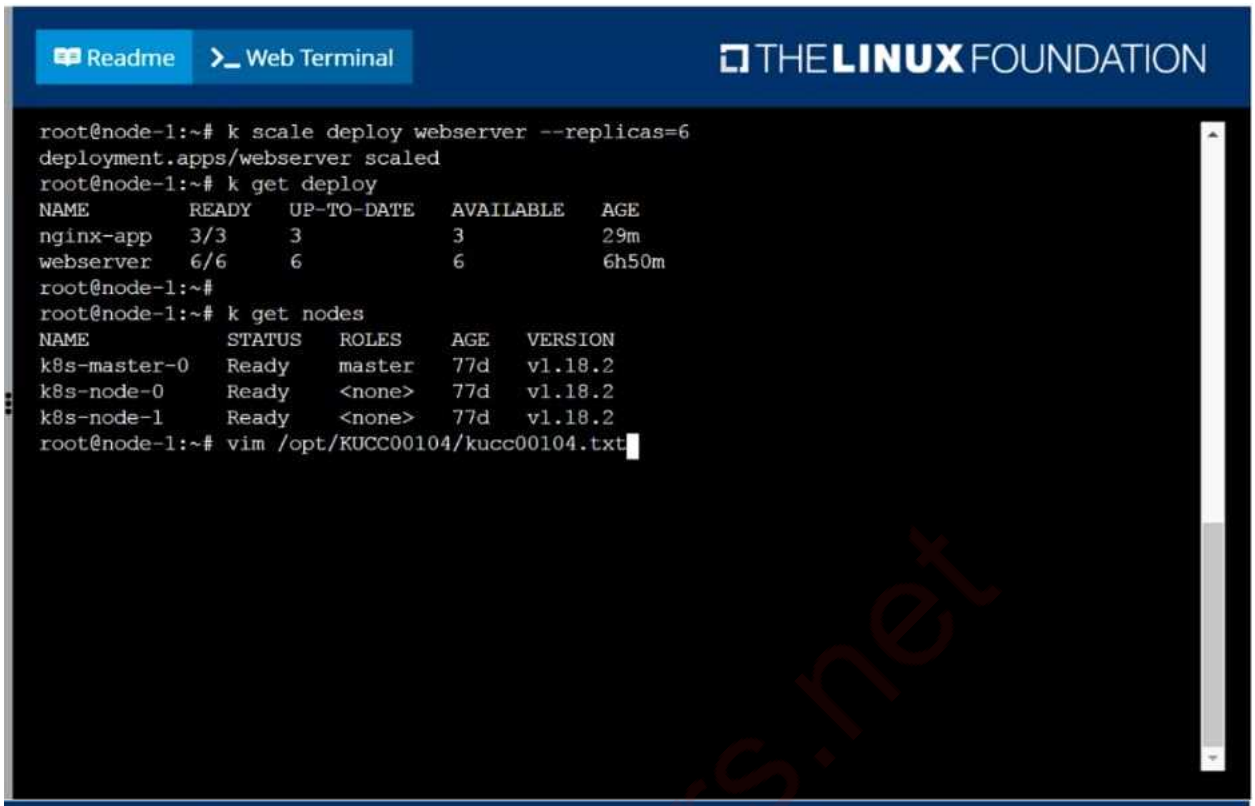


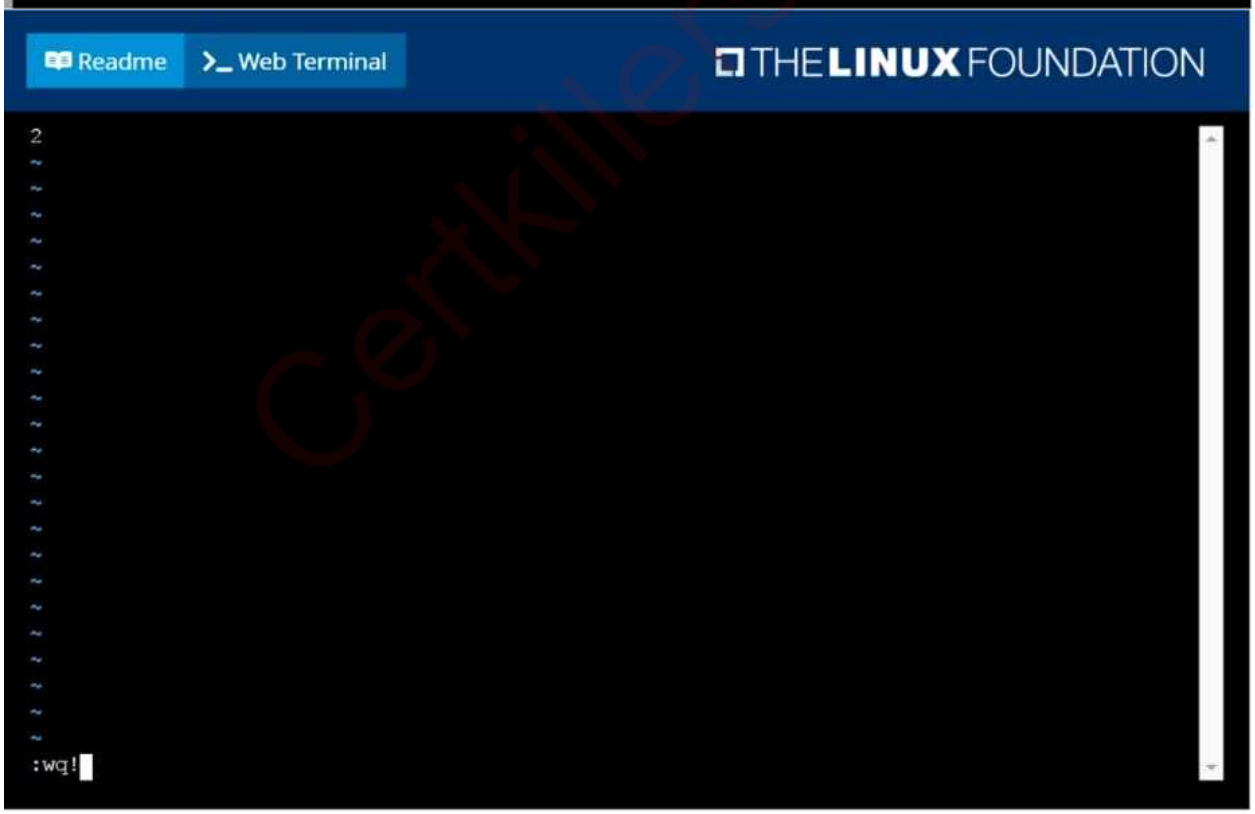
Question: 15

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

Solution

solution



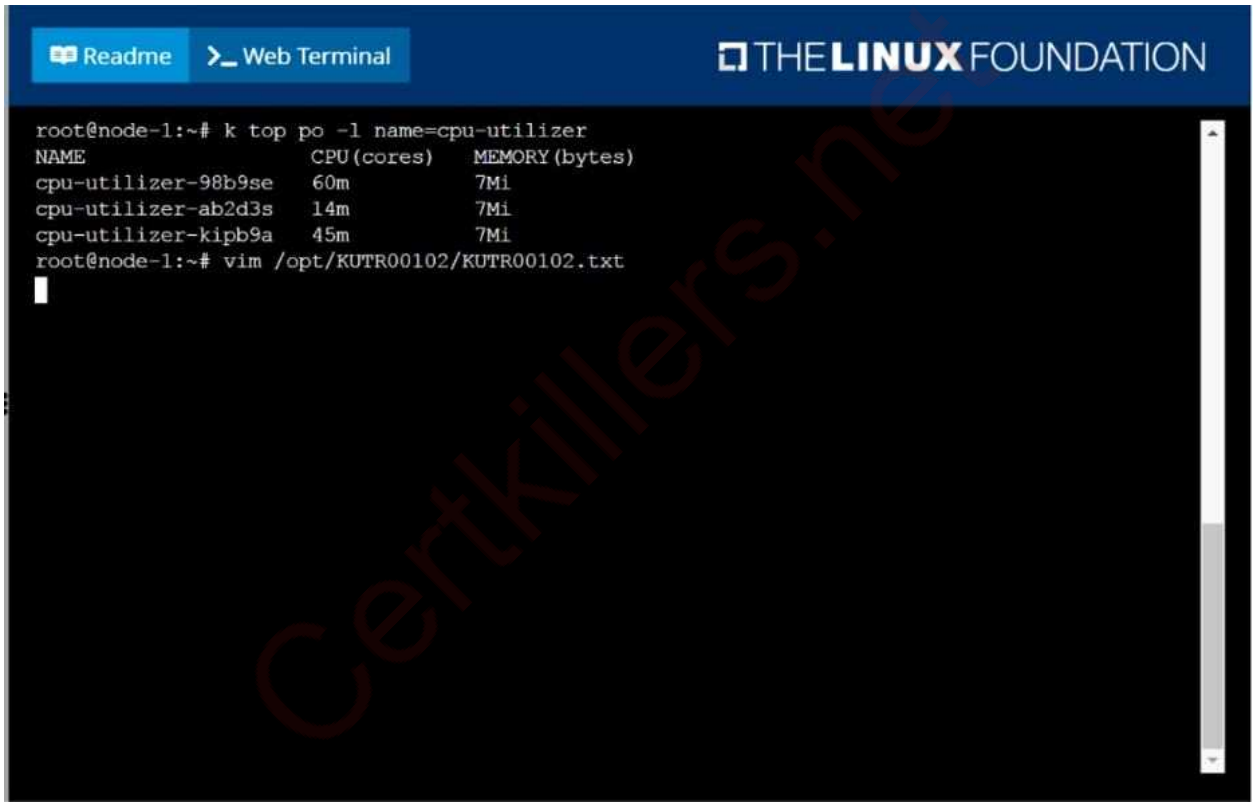


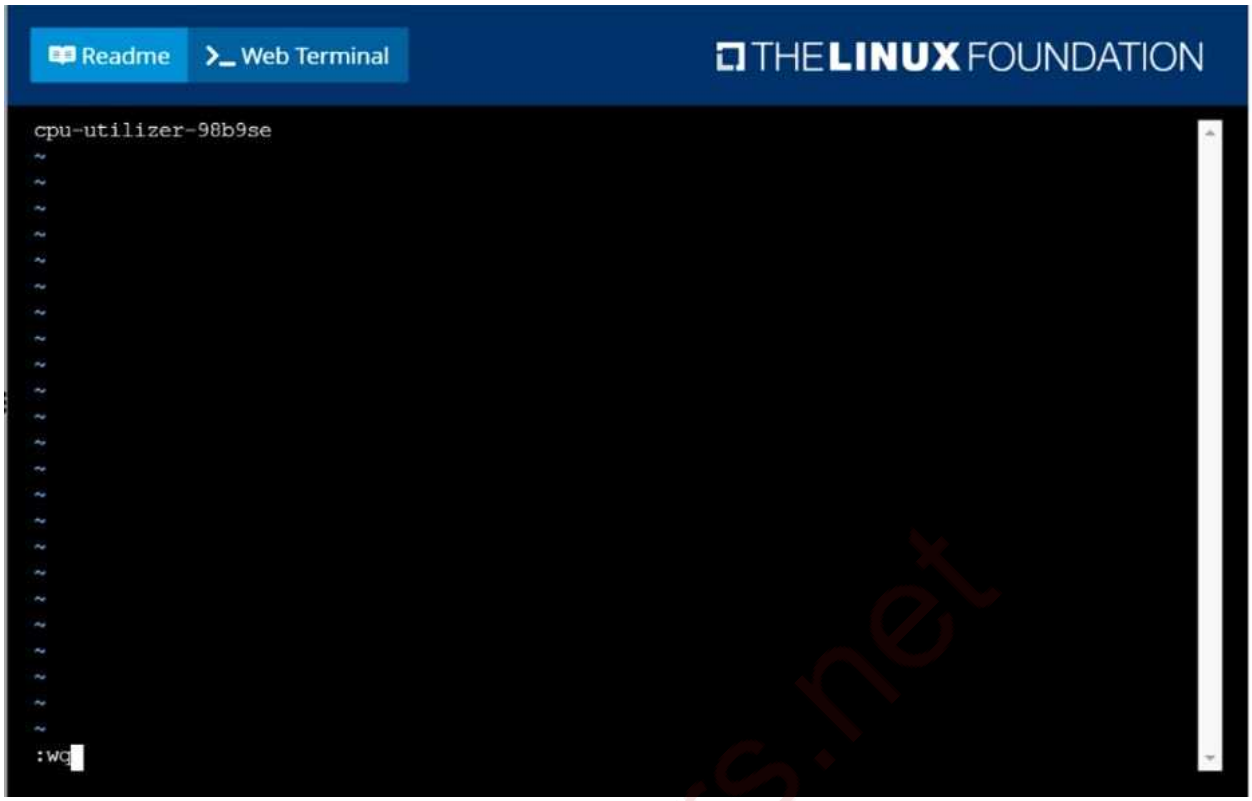
Question: 16

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).

Solution

solution



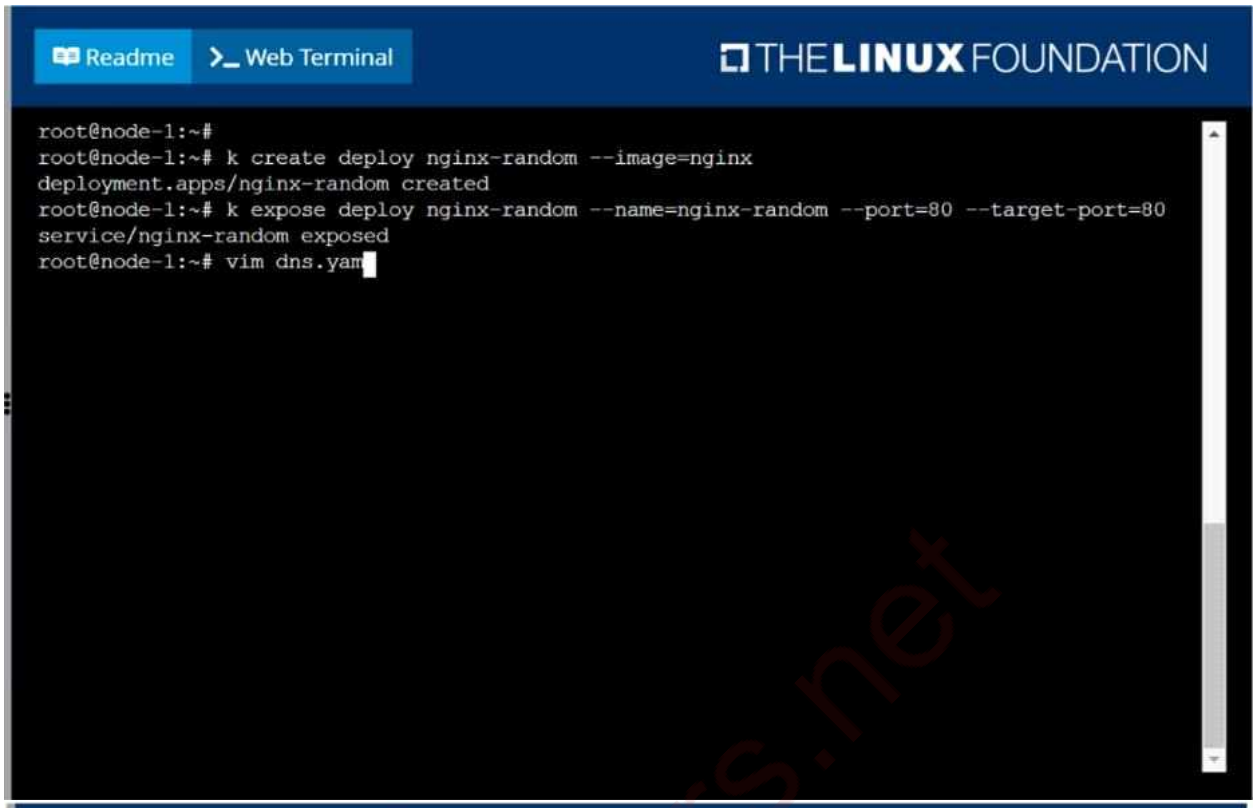


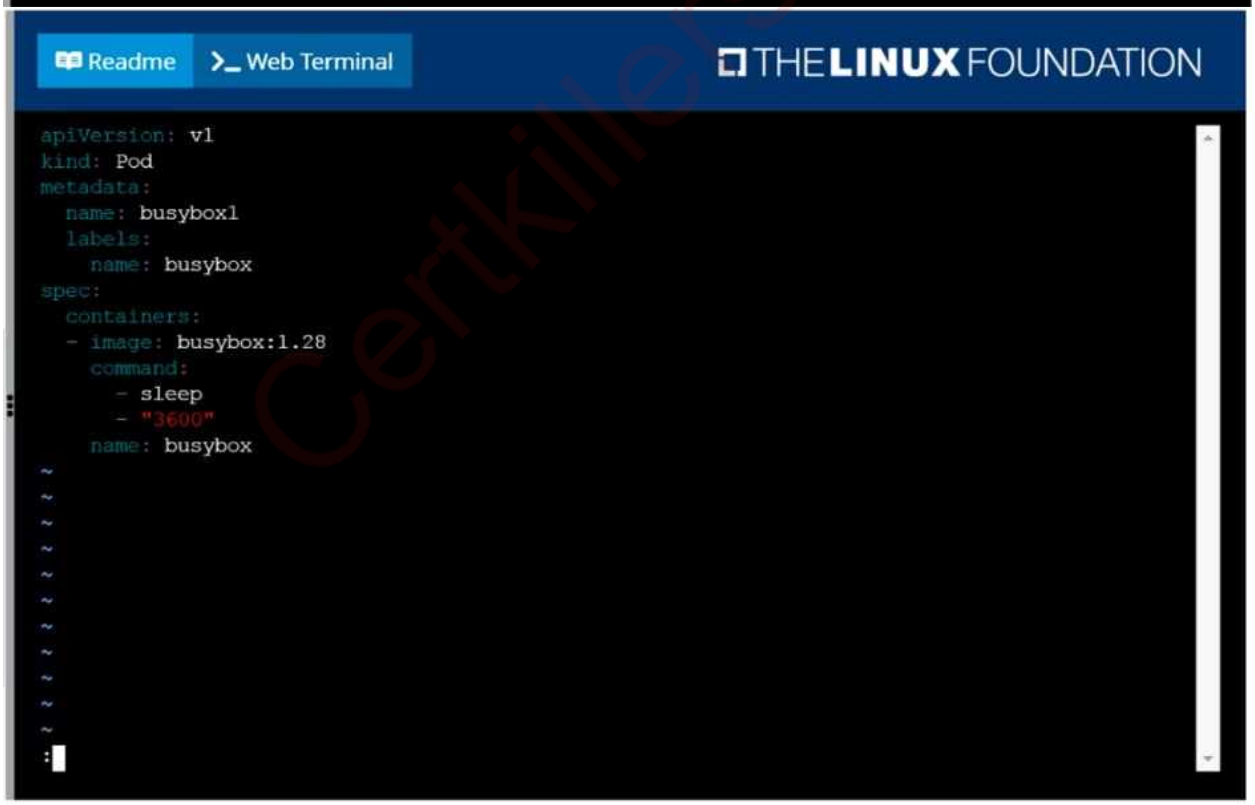
Question: 17

SIMULATION
Create a deployment as follows:
• Name: nginx-random
• Exposed via a service nginx-random
• Ensure that the service & 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 & pod and write the output to /opt/KUNW00601/service.dns and /opt/KUNW00601/pod.dns respectively.

Solution

Solution:



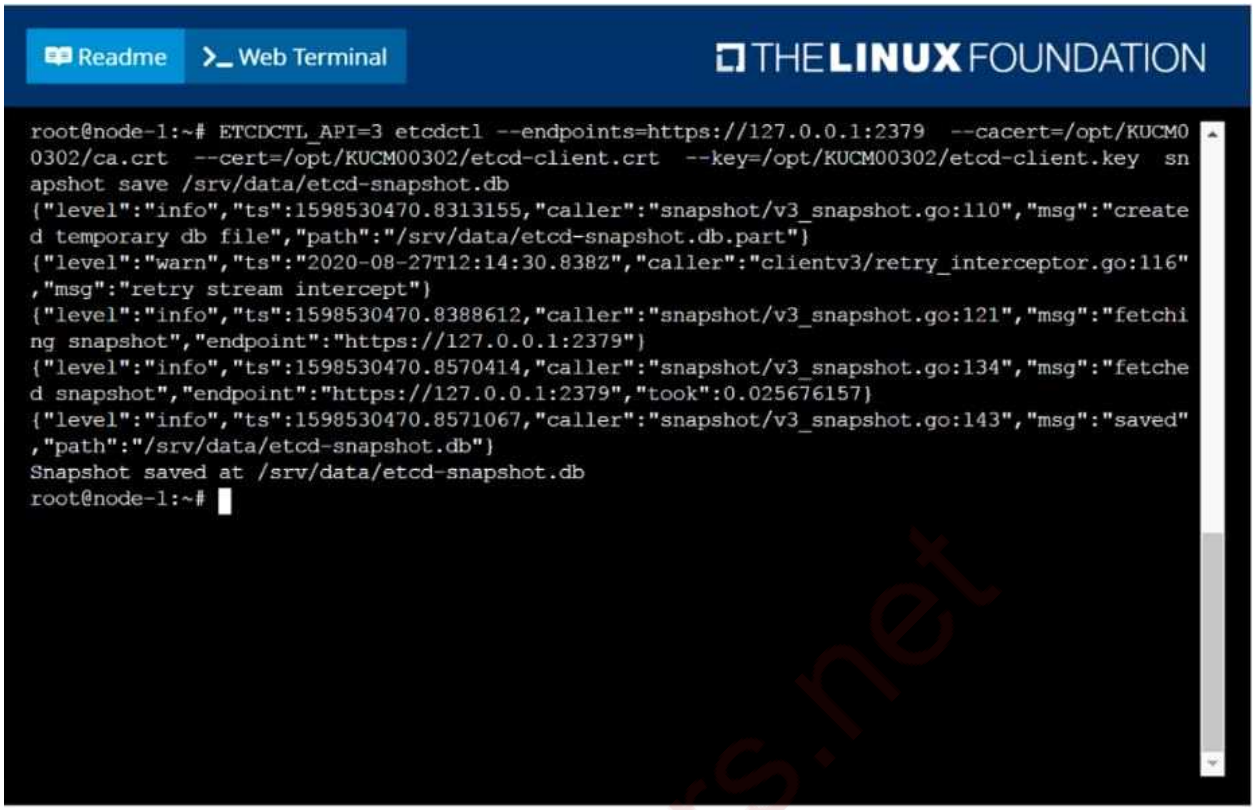


Question: 18

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

Solution

solution

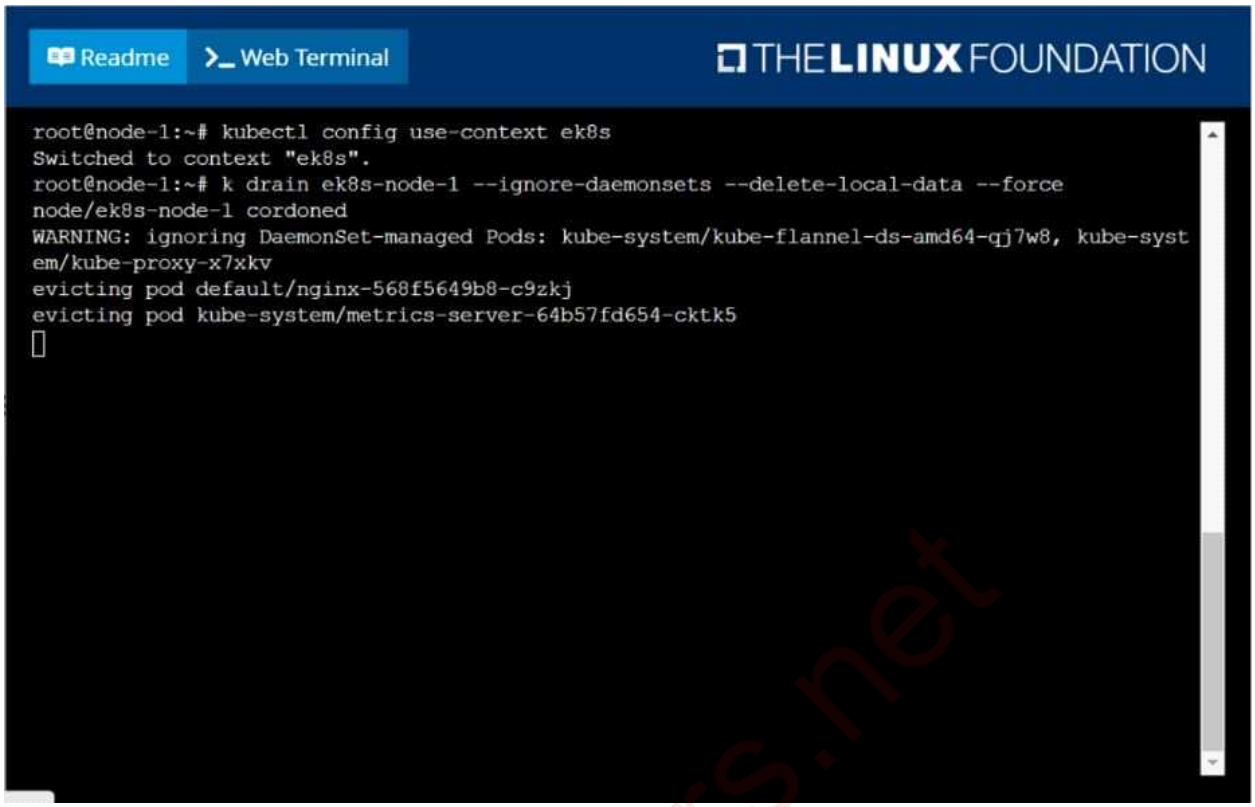


Question: 19

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

Solution

solution

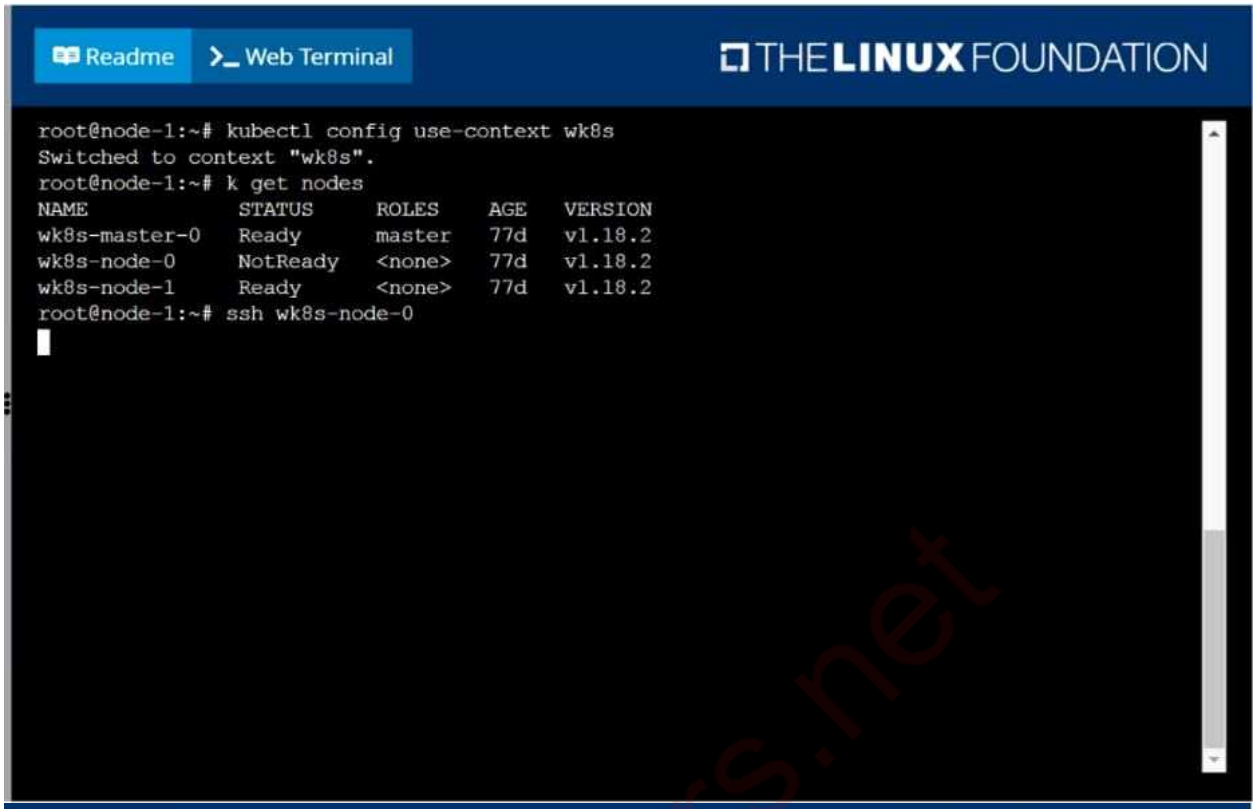


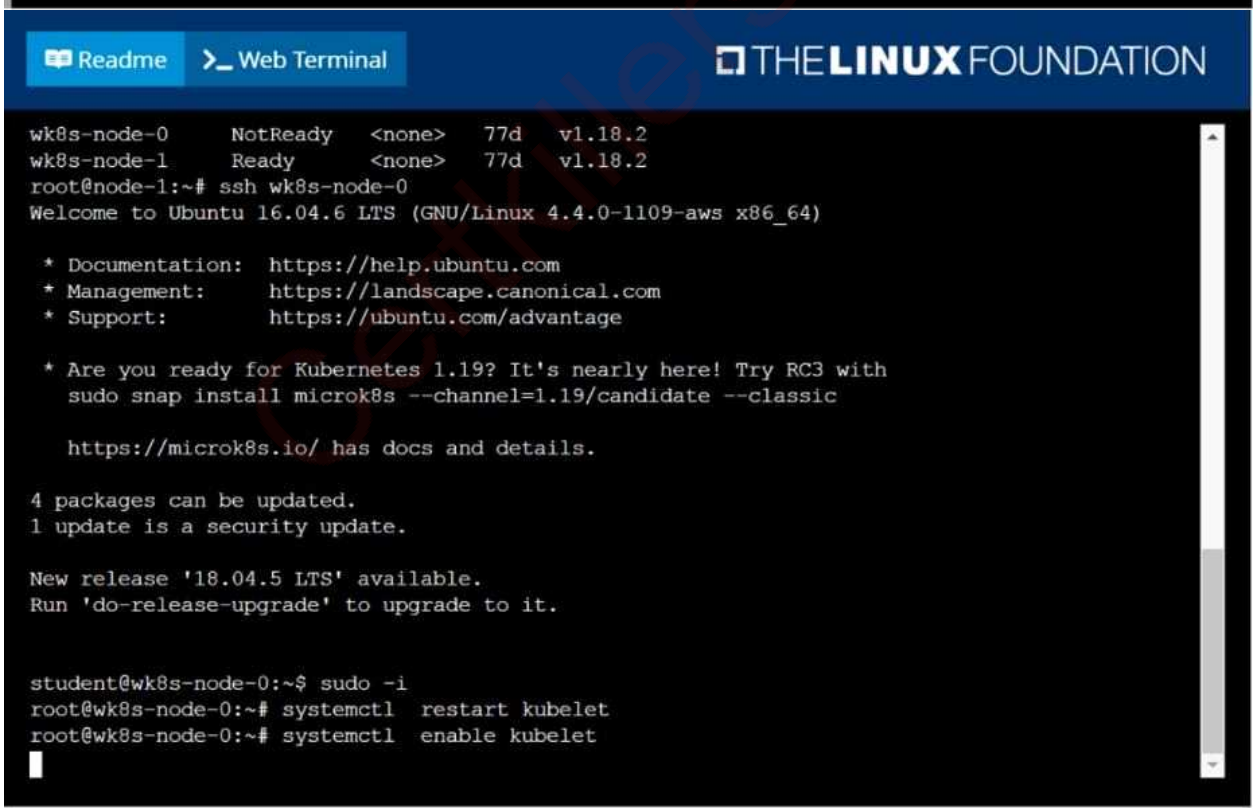
Question: 20

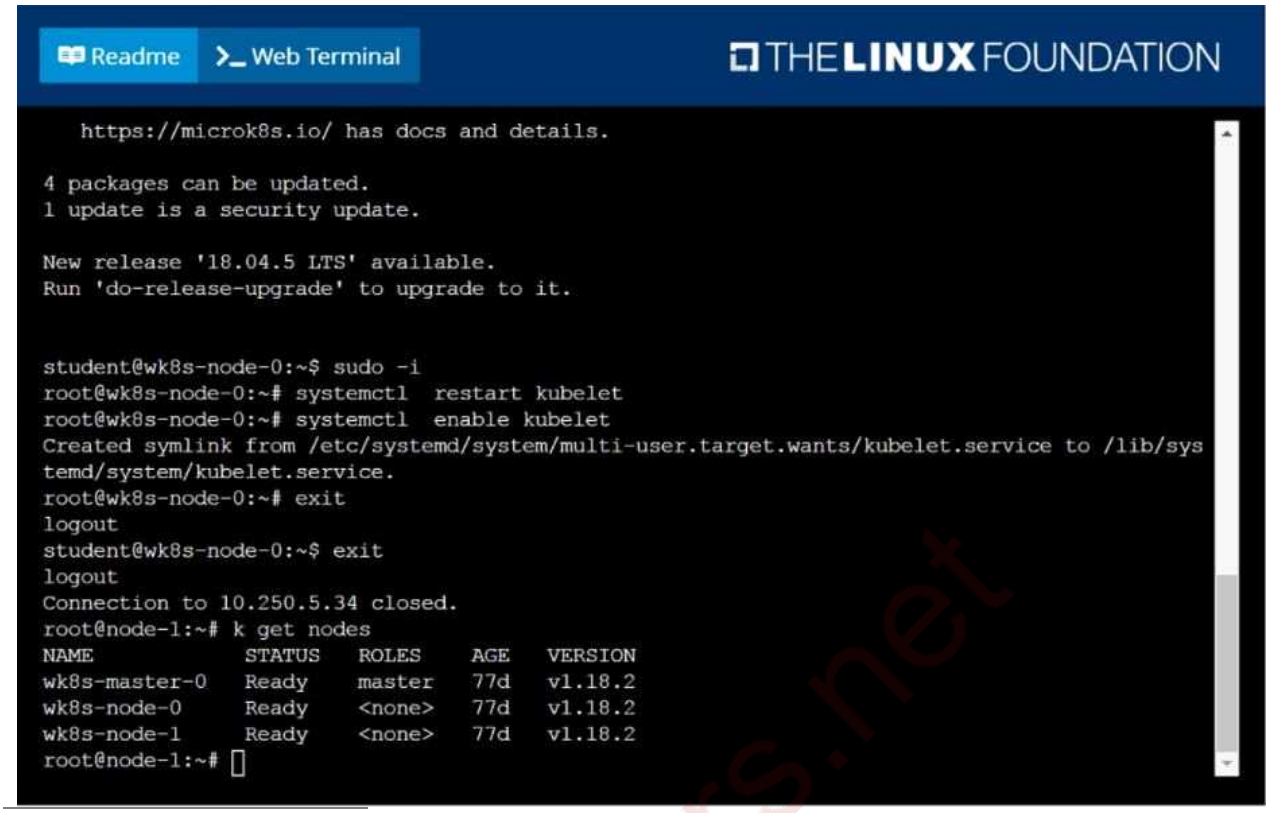
SIMULATION
A Kubernetes worker node, named wk8s-node-0 is in state NotReady. Investigate why this is the case, and perform any appropriate steps to bring the node to a Ready state, ensuring that any changes are made permanent.
You can ssh to the failed node using:
[student@node-1] $ | ssh Wk8s-node-0
You can assume elevated privileges on the node with the following command:
[student@w8ks-node-0] $ | sudo –i

Solution

solution


Readme
Web Terminal
THE LINUX FOUNDATION
root@node-1:~# kubectl config use-context wk8s
Switched to context "wk8s".
root@node-1:~# k get nodes
NAME STATUS ROLES AGE VERSION
wk8s-master-0 Ready master 77d v1.18.2
wk8s-node-0 NotReady <none> 77d v1.18.2
wk8s-node-1 Ready <none> 77d v1.18.2
root@node-1:~# ssh wk8s-node-0


Readme
Web Terminal
THE LINUX FOUNDATION
wk8s-node-0 NotReady <none> 77d v1.18.2
wk8s-node-1 Ready <none> 77d v1.18.2
root@node-1:~# ssh wk8s-node-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 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-0:~$ sudo -i
root@wk8s-node-0:~# systemctl restart kubelet
root@wk8s-node-0:~# systemctl enable kubelet

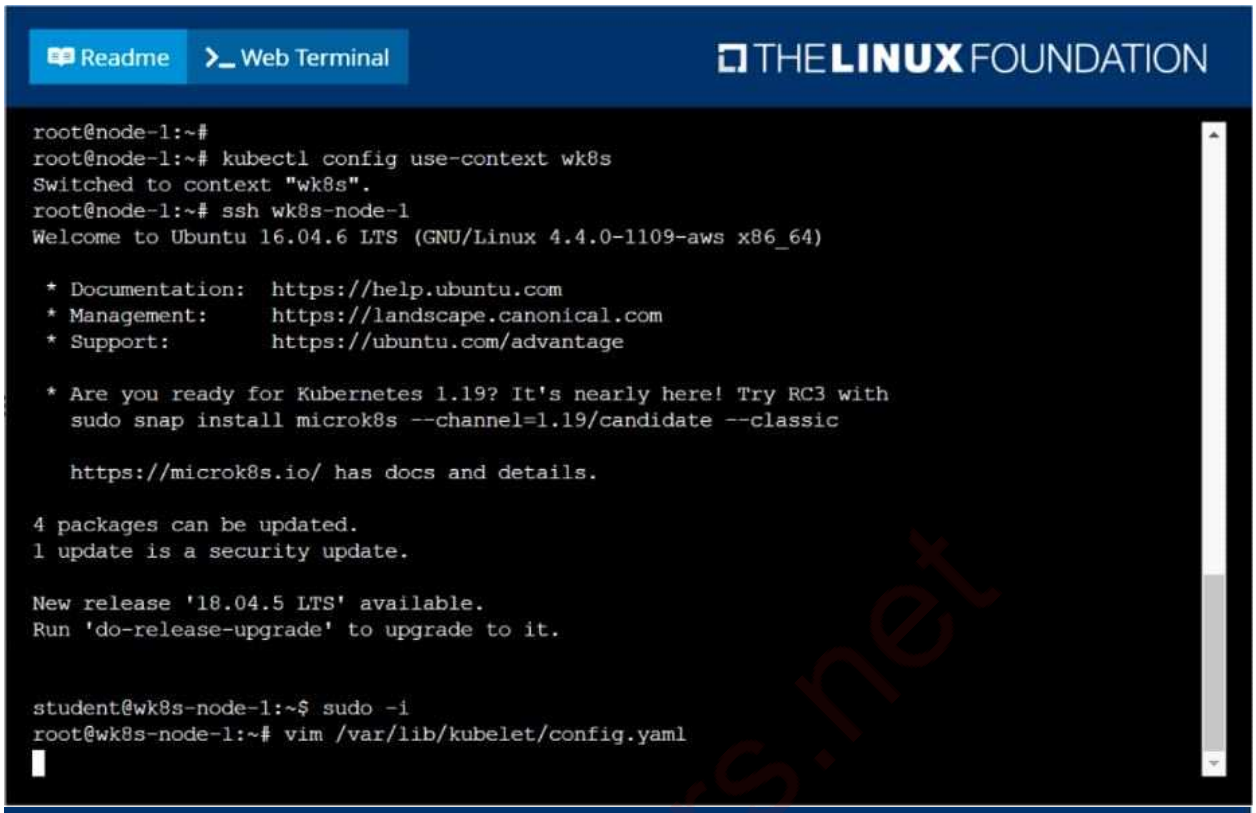


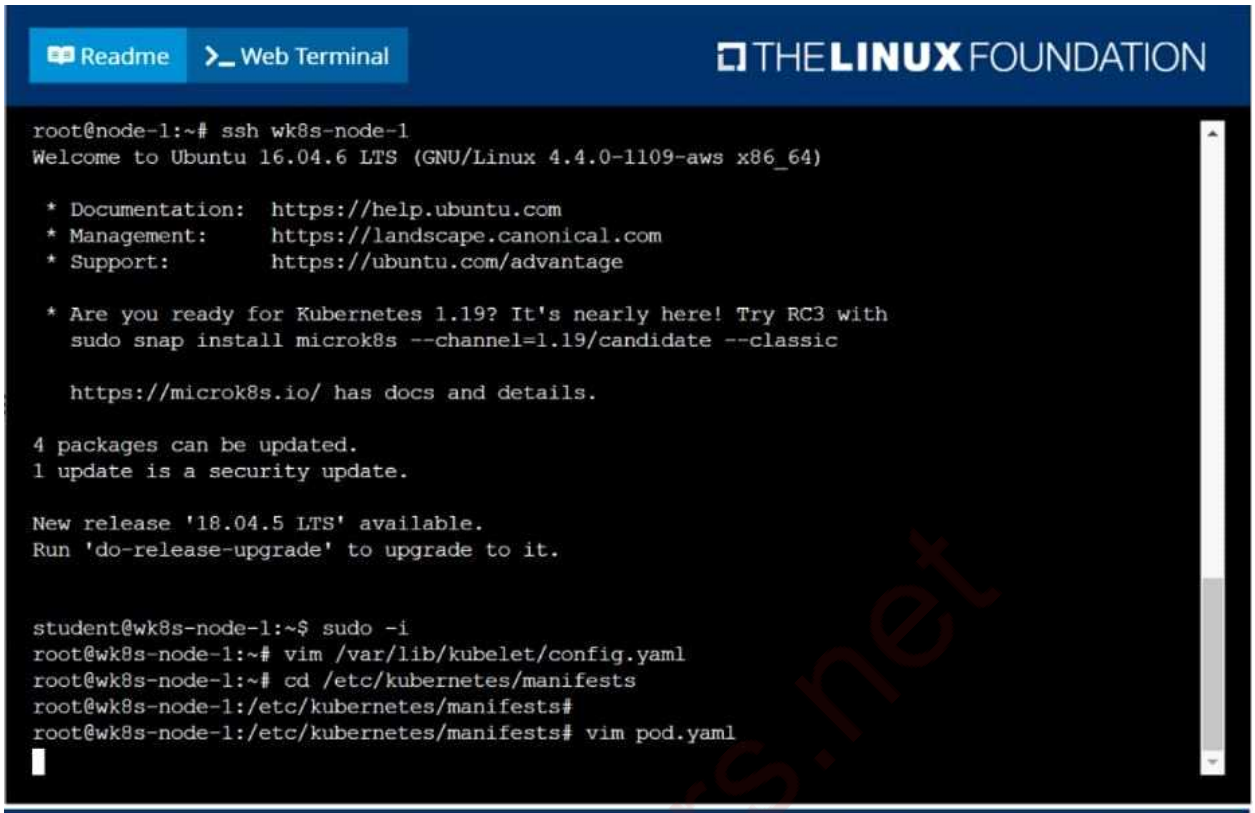
Question: 21

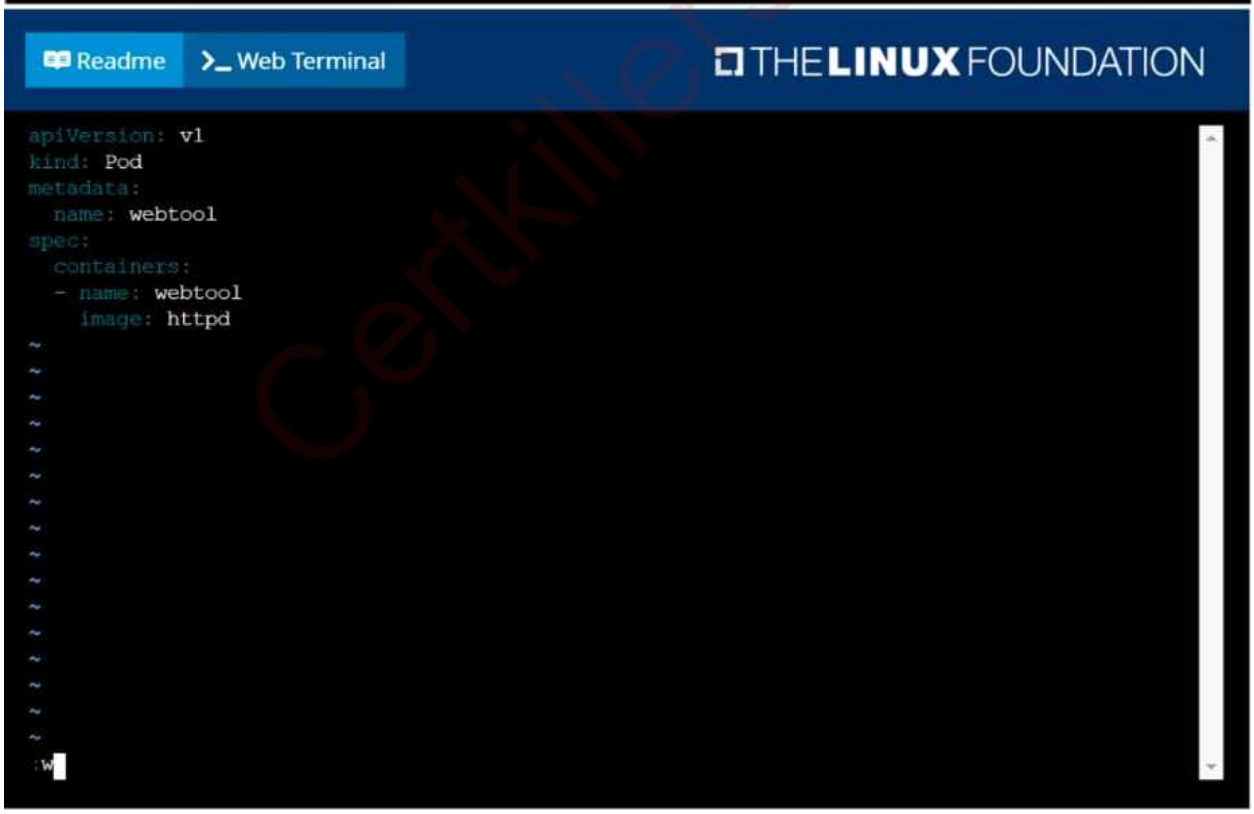
SIMULATION
Configure the kubelet systemd- managed service, on the node labelled with name=wk8s-node-1, to launch a pod containing a single container of Image httpd named webtool automatically. Any spec files required should be placed in the /etc/kubernetes/manifests directory on the node.
You can ssh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

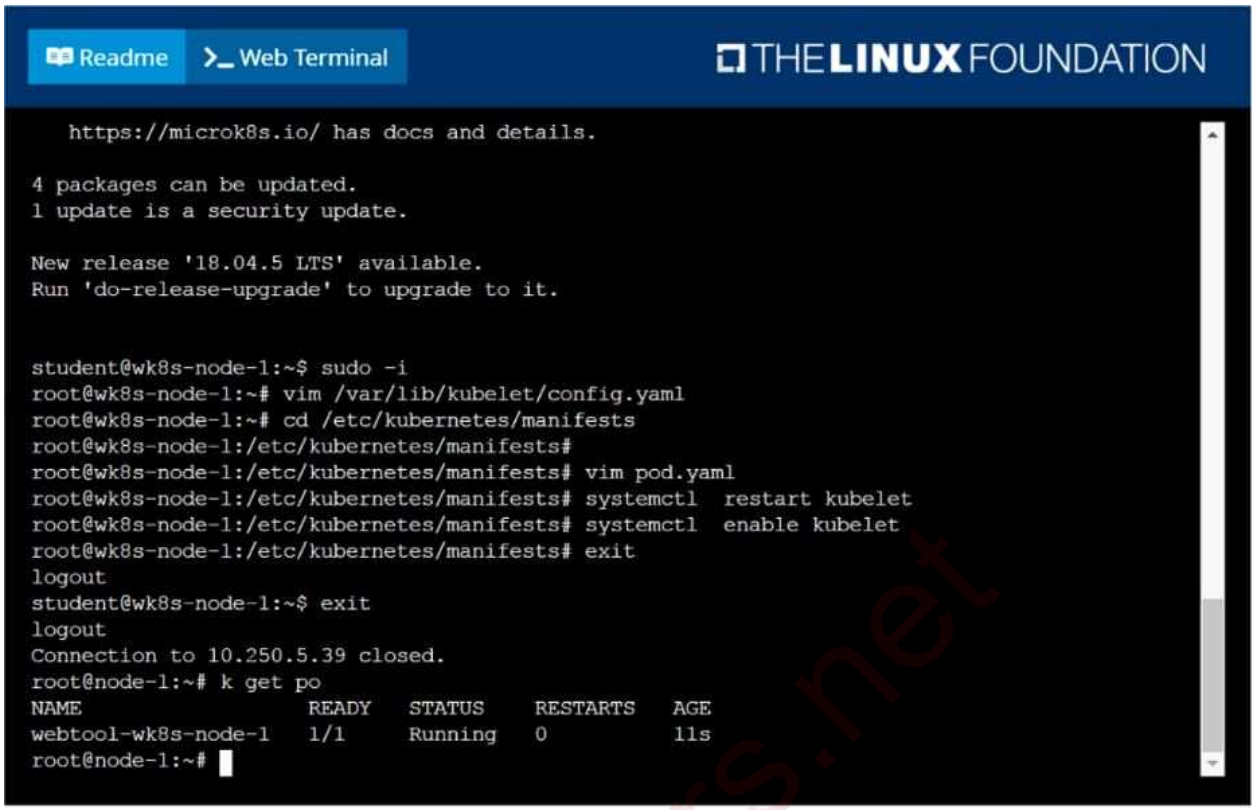
Solution

solution









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.

Solution

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:

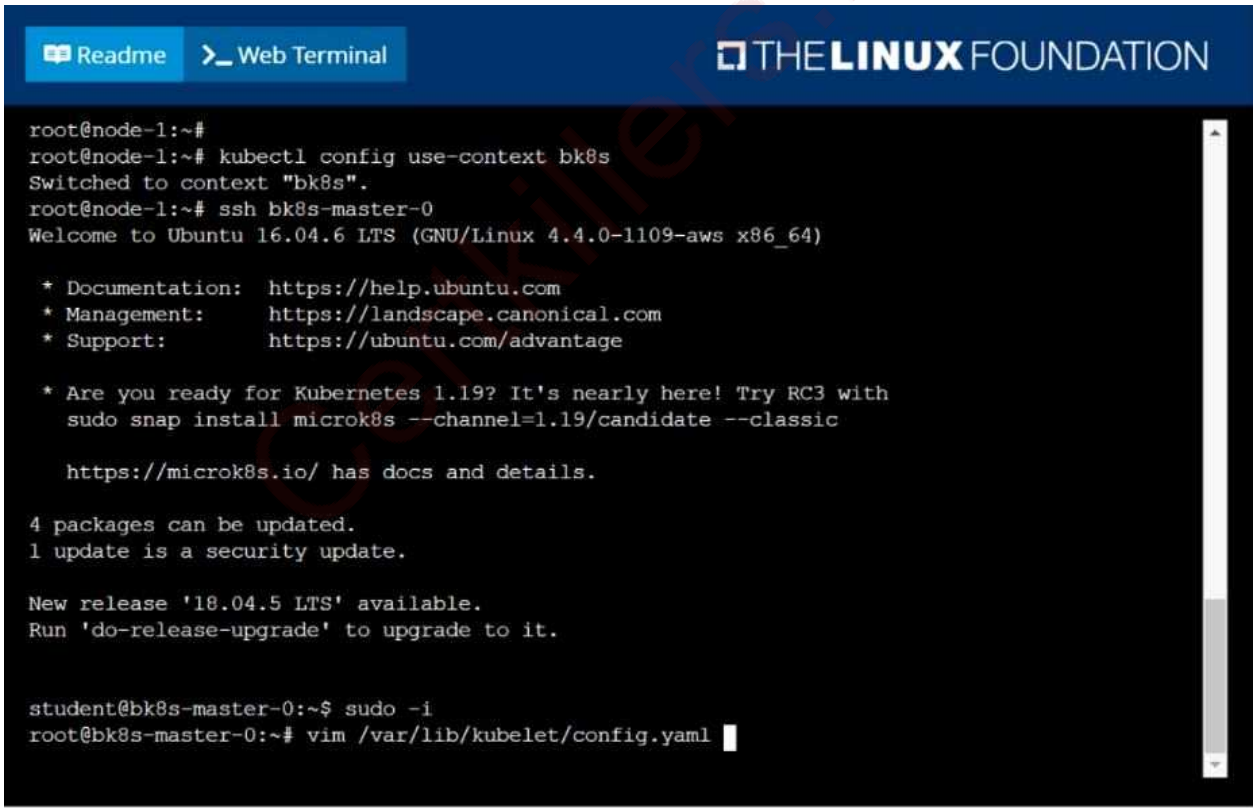
https://docs.projectcalico.org/v3.14/manifests/calico.yaml
Docker is already installed on both nodes and apt has been configured so that you can install the required tools.

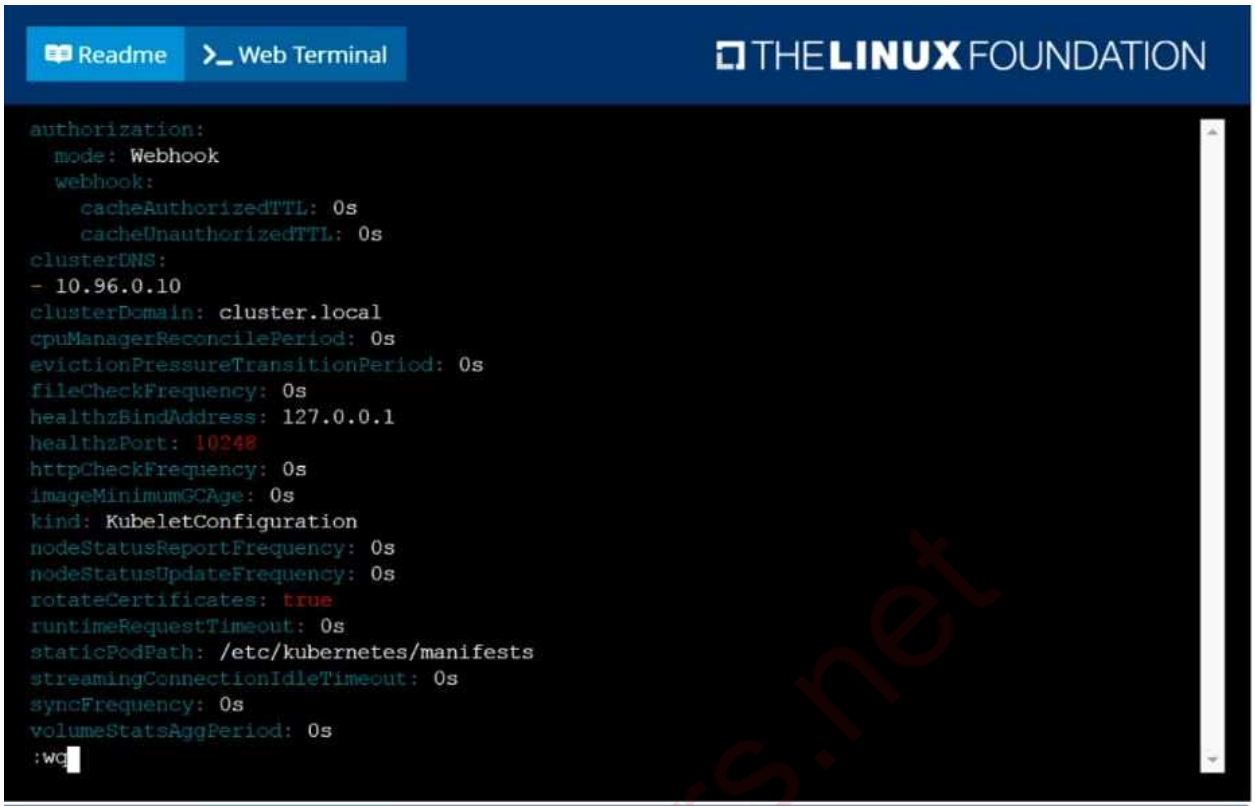
Question: 23

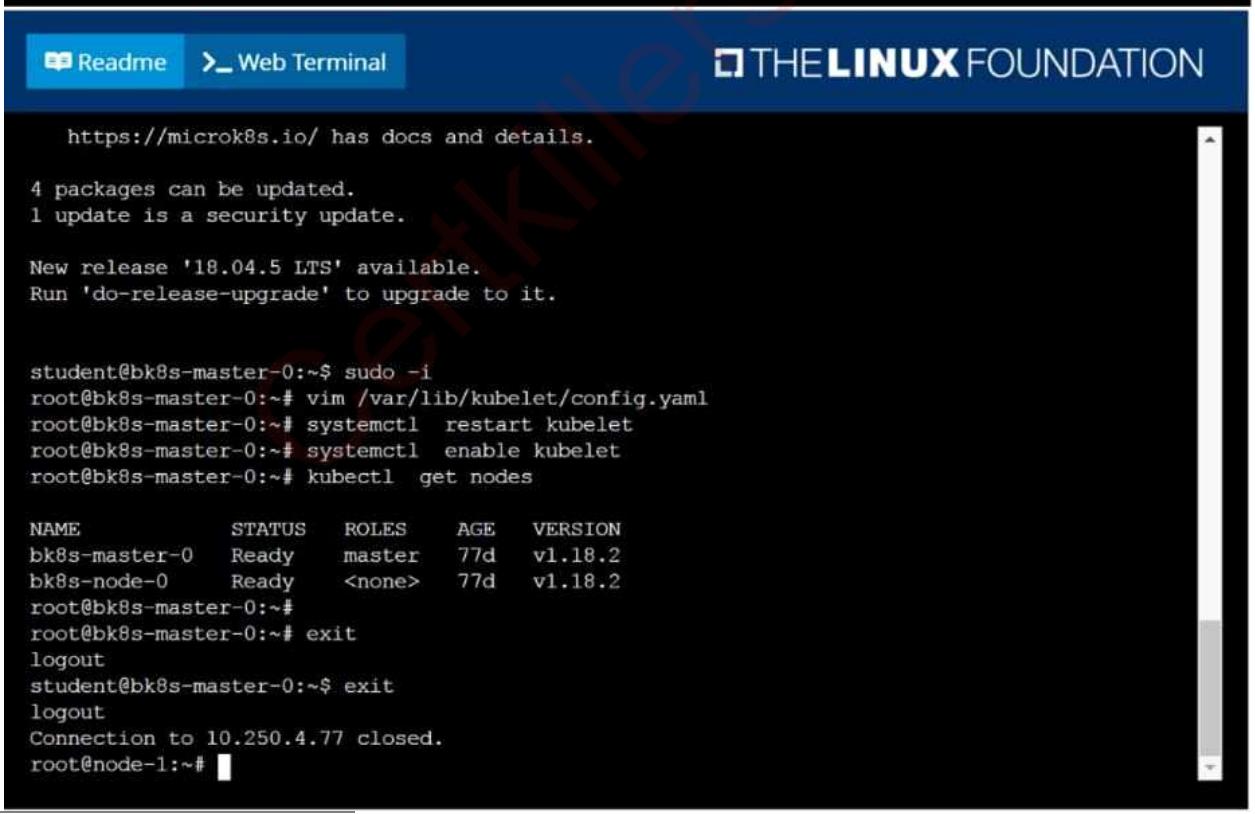
SIMULATION
Given a partially-functioning Kubernetes cluster, identify symptoms of failure on the cluster.
Determine the node, the failing service, and take actions to bring up the failed service and restore the health of the cluster. Ensure that any changes are made permanently.
You can ssh to the relevant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

Solution

solution




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


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.
student@bk8s-master-0:~$ sudo -i
root@bk8s-master-0:~# vim /var/lib/kubelet/config.yaml
root@bk8s-master-0:~# systemctl restart kubelet
root@bk8s-master-0:~# systemctl enable kubelet
root@bk8s-master-0:~# kubectl get nodes
NAME STATUS ROLES AGE VERSION
bk8s-master-0 Ready master 77d v1.18.2
bk8s-node-0 Ready <none> 77d v1.18.2
root@bk8s-master-0:~#
root@bk8s-master-0:~# exit
logout
student@bk8s-master-0:~$ exit
logout
Connection to 10.250.4.77 closed.
root@node-1:~#

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.

Solution

solution
Persistent Volume
A persistent volume is a piece of storage in a Kubernetes cluster. PersistentVolumes are a cluster-level 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.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: app-data
spec:
  capacity:
    storage: 2Gi
  accessModes:
    - ReadWriteMany
  hostPath:
      path: /srv/app-data
  storageClassName: shared
~
~
~
~
~
~
~
~
~
~
~
~
"app-data.yaml" 12L, 194C
```

2. Save the file and create the persistent volume.

```
njerry191@cloudshell:~ (extreme-clone-265411)$ kubectl create -f pv.yaml
persistentvolume/pv created
```

3. View the persistent volume.

```
njerry191@cloudshell:~ (extreme-clone-265411)$ kubectl get pv
NAME    CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   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

```
njerry191@cloudshell:~ (extreme-clone-265411)$ kubectl get pvc
NAME    STATUS    VOLUME    CAPACITY    ACCESS MODES    STORAGECLASS
pv      Bound     pv        512m        RWX             shared
```

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

```
njerry191@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
```

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

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

Solution

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

Question: 26

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

Solution

kubectl run nginx --image=nginx --restart=Never --labels=env=test --namespace=engineering --dry-run -o yaml > nginx-pod.yaml
kubectl run nginx --image=nginx --restart=Never --labels=env=test --namespace=engineering --dry-run -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

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

Solution

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

Question: 28

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

Solution

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

Question: 29

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

Solution

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

Question: 30

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

Solution

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

Question: 31

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

Solution

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

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

Solution

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

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

Solution

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

Question: 34

Check the image version in pod without the describe command

Solution

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

Question: 35

List the nginx pod with custom columns POD_NAME and POD_STATUS

Solution

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

Question: 36

List all the pods sorted by name

Solution

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

Question: 37

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

Solution

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

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.

Solution

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

Get IP address of the pod – "nginx-dev"

Solution

Kubect1 get po -o wide
Using JsonPath
kubect1 get pods -o=jsonpath='{range
.items[*]}{.metadata.name}{"\t"}{.status.podIP}{"\n"}{end}'

Question: 40

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

Solution

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

Question: 41

Check the Image version of nginx-dev pod using jsonpath

Solution

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

Question: 42

Create a busybox pod and add “sleep 3600” command

Solution

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

Question: 43

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

Solution

// 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

List the nginx pod with custom columns POD_NAME and POD_STATUS

Solution

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

Question: 45

List all the pods sorted by name

Solution

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

Question: 46

List all the pods sorted by created timestamp

Solution

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

Question: 47

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

Solution

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

Question: 48

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

Solution

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