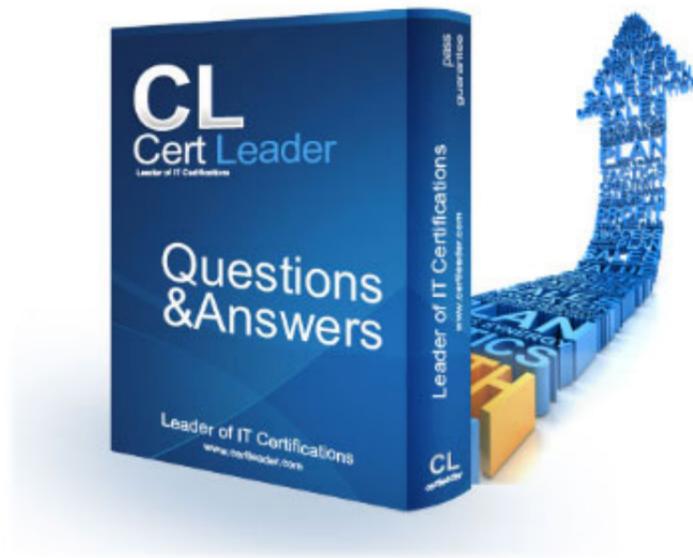


CKAD Dumps

Certified Kubernetes Application Developer (CKAD) Program

<https://www.certleader.com/CKAD-dumps.html>



NEW QUESTION 1

Exhibit:



Context

It is always useful to look at the resources your applications are consuming in a cluster. Task

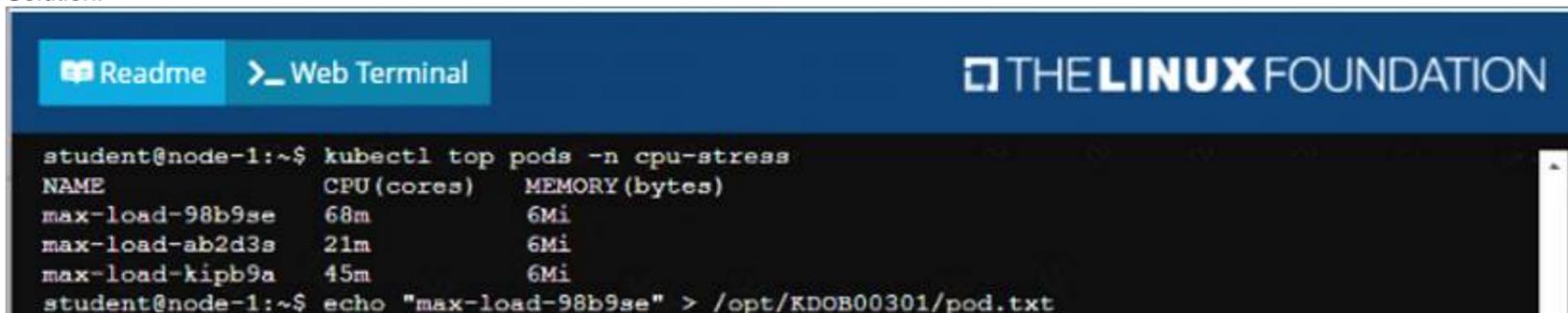
- From the pods running in namespacecpu-stress , write the name only of the pod that is consuming the most CPU to file /opt/KDOBG030I/pod.txt, which has already been created.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:



NEW QUESTION 2

Exhibit:



Context

A container within the poller pod is hard-coded to connect the nginxsvc service on port90 . As this port changes to5050 an additional container needs to be added to the poller pod which adapts the container to connect to this new port. This should be realized as an ambassador container within the pod.

Task

- Update the nginxsvc service to serve on port5050.
- Add an HAProxy container named haproxy bound to port90 tothe poller pod and deploy the enhanced pod. Use the image haproxy and inject the configuration located at /opt/KDMC00101/haproxy.cfg, with a ConfigMap named haproxy-config, mounted into the container so that haproxy.cfg is available at /usr/local/etc/haproxy/haproxy.cfg. Ensure that you update the args of the poller container to connect to localhost instead of nginxsvc so that the connection is correctly proxied to the new service endpoint. You must not modify the port of the endpoint in poller's args . The spec file used to create the initial poller pod is available in /opt/KDMC00101/poller.yaml

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution: apiVersion: apps/v1 kind: Deployment metadata:

name: my-nginx spec:

selector: matchLabels: run: my-nginx replicas: 2 template: metadata: labels:

run: my-nginx spec: containers:

- name: my-nginx image: nginx ports:

- containerPort: 90

This makes it accessible from any node in your cluster. Check the nodes the Pod is running on: kubectl apply -f ./run-my-nginx.yaml

kubectl get pods -lrun=my-nginx -o wide

NAME READY STATUS RESTARTS AGE IP NODE

my-nginx-3800858182-jr4a2 1/1 Running 0 13s 10.244.3.4 kubernetes-minion-905m

my-nginx-3800858182-kna2y 1/1 Running 0 13s 10.244.2.5 kubernetes-minion-ljyd Check your pods' IPs:

kubectl get pods -lrun=my-nginx -o yaml | grep podIP podIP: 10.244.3.4

podIP: 10.244.2.5

NEW QUESTION 3

Exhibit:



Context

A pod is running on the cluster but it is not responding. Task

The desired behavior is to have Kubernetes restart the pod when an endpoint returns an HTTP 500 on the /healthz endpoint. The service, probe-pod, should never send traffic to the pod while it is failing. Please complete the following:

- The application has an endpoint, /started, that will indicate if it can accept traffic by returning an HTTP 200. If the endpoint returns an HTTP 500, the application has not yet finished initialization.
- The application has another endpoint /healthz that will indicate if the application is still working as expected by returning an HTTP 200. If the endpoint returns an HTTP 500 the application is no longer responsive.
- Configure the probe-pod pod provided to use these endpoints
- The probes should use port 8080

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:

```
apiVersion:v1 kind:Pod metadata: labels: test:liveness
name:liveness-exec
spec: containers:
-name:liveness
image:k8s.gcr.io/busybox args:
- /bin/sh
- -c
- touch/tmp/healthy;sleep30;rm-rf/tmp/healthy;sleep600
livenessProbe: exec: command:
- cat
- /tmp/healthy initialDelaySeconds:5 periodSeconds:5
```

In the configuration file, you can see that the Pod has a single Container. The periodSeconds field specifies that the kubelet should perform a liveness probe every 5 seconds. The initialDelaySeconds field tells the kubelet that it should wait 5 seconds before performing the first probe. To perform a probe, the kubelet executes the command cat /tmp/healthy in the target container. If the command succeeds, it returns 0, and the kubelet considers the container to be alive and healthy. If the command returns a non-zero value, the kubelet kills the container and restarts it.

When the container starts, it executes this command:

```
/bin/sh -c"touch /tmp/healthy; sleep 30; rm -rf /tmp/healthy; sleep 600"
```

For the first 30 seconds of the container's life, there is a /tmp/healthy file. So during the first 30 seconds, the command cat /tmp/healthy returns a success code.

After 30 seconds, cat /tmp/healthy returns a failure code

Create the Pod:

```
kubectl apply -f https://k8s.io/examples/pods/probe/exec-liveness.yaml Within 30 seconds, view the Pod events:
```

```
kubectl describe pod liveness-exec
```

The output indicates that no liveness probes have failed yet:

```
FirstSeen LastSeen Count From SubobjectPath Type Reason Message
```

```
-----
24s 24s 1 {default-scheduler } Normal Scheduled Successfully assigned liveness-exec to worker0
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "k8s.gcr.io/busybox" 23s 23s 1 {kubelet worker0} spec.containers{liveness}
Normal Pulled Successfully pulled image
"k8s.gcr.io/busybox"
```

```
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e; Security:[seccomp=unconfined]
```

```
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started container with docker id 86849c15382e
```

After 35 seconds, view the Pod events again: kubectl describe pod liveness-exec

At the bottom of the output, there are messages indicating that the liveness probes have failed, and the containers have been killed and recreated.

```
FirstSeen LastSeen Count From SubobjectPath Type Reason Message
```

```
-----
37s 37s 1 {default-scheduler } Normal Scheduled Successfully assigned liveness-exec to worker0
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulling pulling image "k8s.gcr.io/busybox" 36s 36s 1 {kubelet worker0} spec.containers{liveness}
Normal Pulled Successfully pulled image
"k8s.gcr.io/busybox"
```

```
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Created Created container with docker id 86849c15382e; Security:[seccomp=unconfined]
```

```
36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Started Started container with docker id 86849c15382e
```

```
2s 2s 1 {kubelet worker0} spec.containers{liveness} Warning Unhealthy Liveness probe failed: cat: can't open '/tmp/healthy': No such file or directory
```

Wait another 30 seconds, and verify that the container has been restarted: kubectl get pod liveness-exec

```
The output shows that RESTARTS has been incremented: NAME READY STATUS RESTARTS AGE
```

```
liveness-exec 1/1 Running 1 1m
```

NEW QUESTION 4

Exhibit:

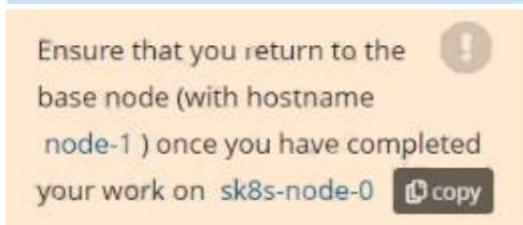
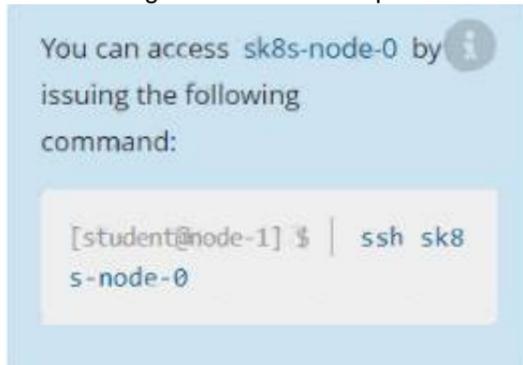


Context

A project that you are working on has a requirement for persistent data to be available. Task

To facilitate this, perform the following tasks:

- Create a file on node sk8s-node-0 at /opt/KDSP00101/data/index.html with the content Acct=Finance
- Create a PersistentVolume named task-pv-volume using hostPath and allocate 1Gi to it, specifying that the volume is at /opt/KDSP00101/data on the cluster's node. The configuration should specify the access mode of ReadWriteOnce . It should define the StorageClass name exam for the PersistentVolume , which will be used to bind PersistentVolumeClaim requests to this PersistentVolume.
- Create a PersistentVolumeClaim named task-pv-claim that requests a volume of at least 100Mi and specifies an access mode of ReadWriteOnce
- Create a pod that uses the PersistentVolumeClaim as a volume with a label app: my-storage-app mounting the resulting volume to a mountPath /usr/share/nginx/html inside the pod

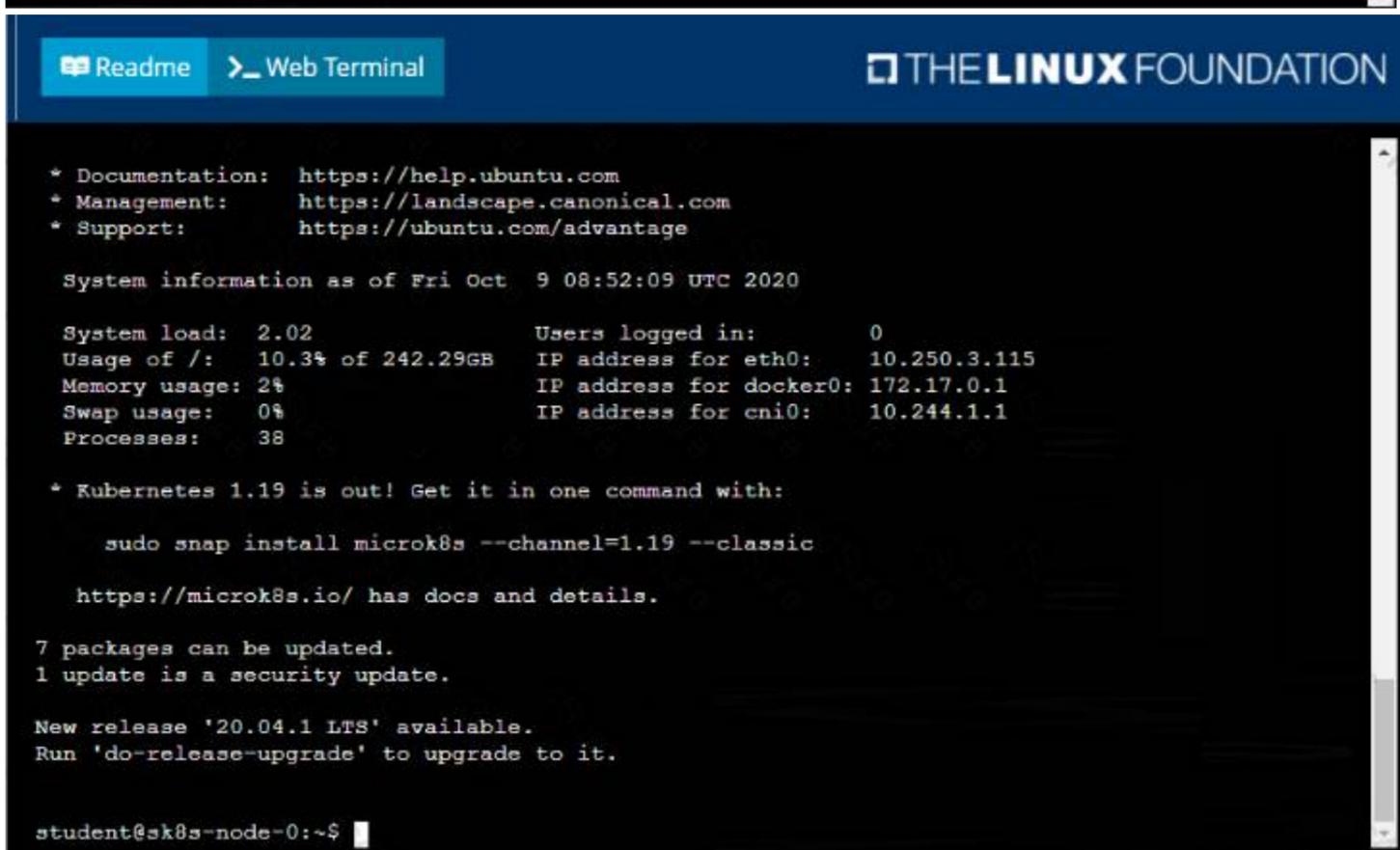
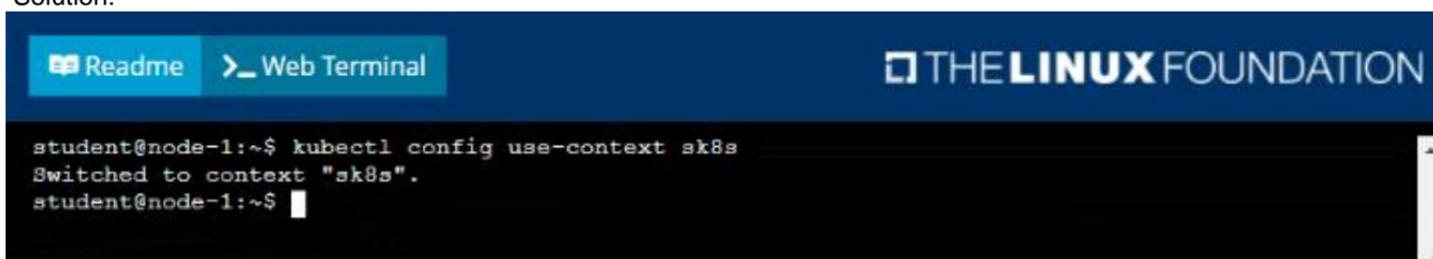


- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Solution:



Readme Web Terminal THE LINUX FOUNDATION

```

apiVersion: v1
kind: Pod
metadata:
  name: mypod
  labels:
    app: my-storage-app
spec:
  containers:
  - name: myfrontend
    image: nginx
    volumeMounts:
    - mountPath: "/usr/share/nginx/html"
      name: mypod
  volumes:
  - name: mypod
    persistentVolumeClaim:
      claimName: task-pv-clai
  
```

17,32 All

```

student@sk8s-node-0:~$ kubectl create -f pod.yml
pod/mypod created
student@sk8s-node-0:~$ kubectl get
  
```

Readme Web Terminal THE LINUX FOUNDATION

```

student@sk8s-node-0:~$ kubectl get pods
NAME      READY   STATUS             RESTARTS   AGE
mypod    0/1     ContainerCreating   0           4s
student@sk8s-node-0:~$ kubectl get pods
NAME      READY   STATUS             RESTARTS   AGE
mypod    0/1     ContainerCreating   0           8s
student@sk8s-node-0:~$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
mypod    1/1     Running   0           10s
student@sk8s-node-0:~$ logout
Connection to 10.250.3.115 closed.
student@node-1:~$
  
```

NEW QUESTION 10
.....

Thank You for Trying Our Product

* 100% Pass or Money Back

All our products come with a 90-day Money Back Guarantee.

* One year free update

You can enjoy free update one year. 24x7 online support.

* Trusted by Millions

We currently serve more than 30,000,000 customers.

* Shop Securely

All transactions are protected by VeriSign!

100% Pass Your CKAD Exam with Our Prep Materials Via below:

<https://www.certleader.com/CKAD-dumps.html>