



# IT認證考試題庫 專業平臺

考證寶提供最新考古題與模擬試題  
協助您高效通過認證考試

[www.kaozhengpro.com](http://www.kaozhengpro.com)

**Exam** : **CKA**

**Title** : **Certified Kubernetes  
Administrator**

**Version** : **DEMO**

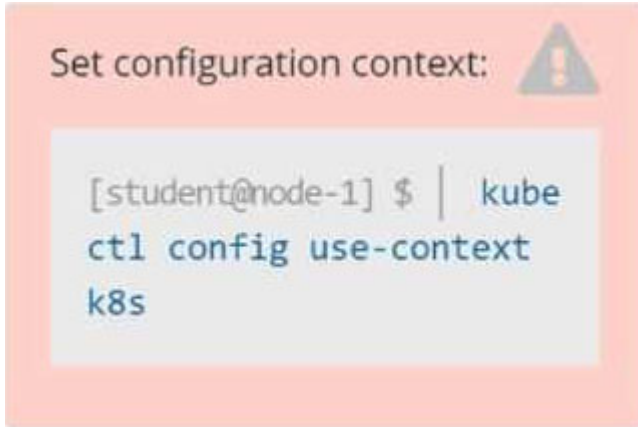
## 1.SIMULATION

Monitor the logs of pod foo and:

Extract log lines corresponding to error

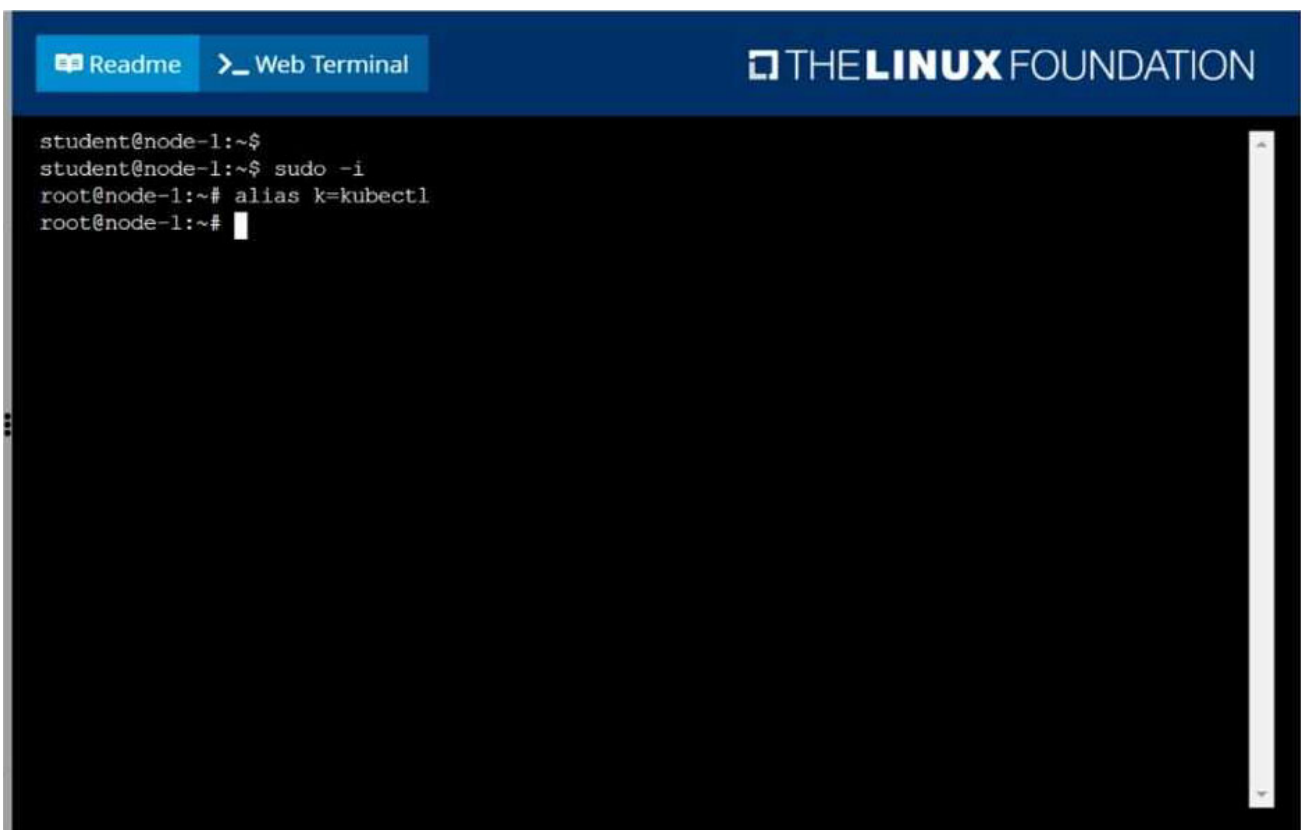
unable-to-access-website

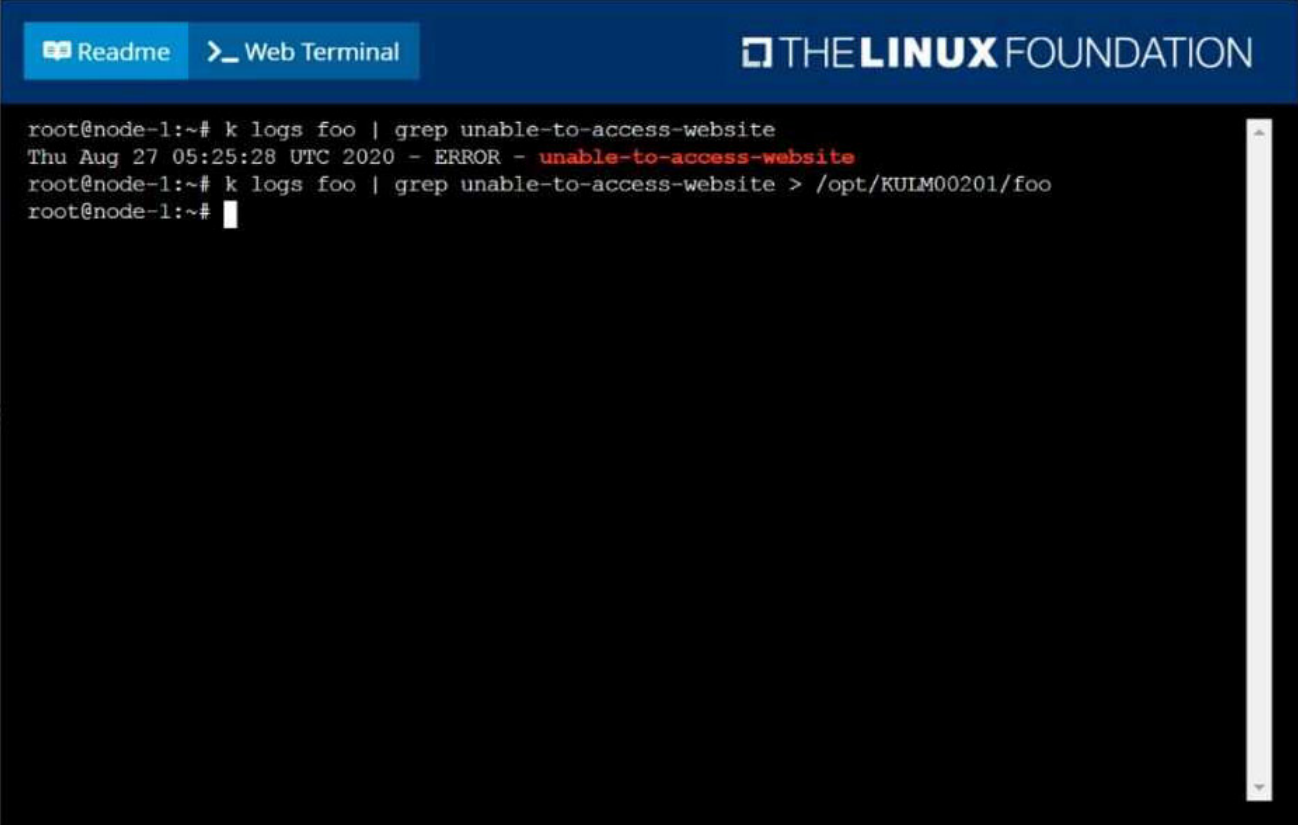
Write them to /opt/KULM00201/foo



**Answer:**

solution





```
root@node-1:~# k logs foo | grep unable-to-access-website
Thu Aug 27 05:25:28 UTC 2020 - ERROR - unable-to-access-website
root@node-1:~# k logs foo | grep unable-to-access-website > /opt/KULM00201/foo
root@node-1:~#
```

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
```

✅ 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
```

✅ Final Answer Summary:

```
kubectl config use-context k8s
```

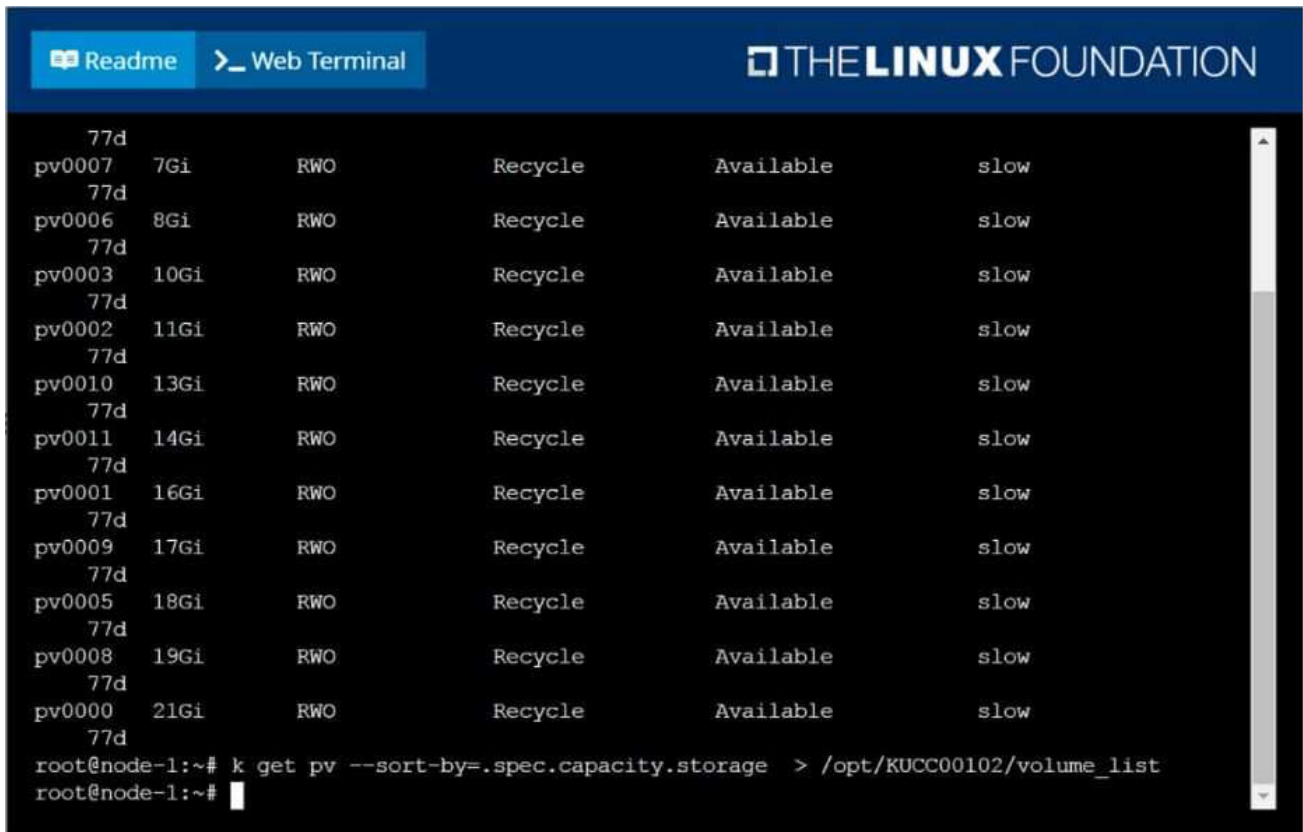
```
kubectl logs foo | grep "unable-to-access-website" > /opt/KULM00201/foo
```

## 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:**

solution



```

77d
pv0007 7Gi RWO Recycle Available slow
77d
pv0006 8Gi RWO Recycle Available slow
77d
pv0003 10Gi RWO Recycle Available slow
77d
pv0002 11Gi RWO Recycle Available slow
77d
pv0010 13Gi RWO Recycle Available slow
77d
pv0011 14Gi RWO Recycle Available slow
77d
pv0001 16Gi RWO Recycle Available slow
77d
pv0009 17Gi RWO Recycle Available slow
77d
pv0005 18Gi RWO Recycle Available slow
77d
pv0008 19Gi RWO Recycle Available slow
77d
pv0000 21Gi RWO Recycle Available slow
77d
root@node-1:~# k get pv --sort-by=.spec.capacity.storage > /opt/KUCC00102/volume_list
root@node-1:~#

```

## 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:**

solution

```
THE LINUX FOUNDATION
Readme Web Terminal
root@node-1:~# vim ds.yaml
i
```

```
THE LINUX FOUNDATION
Readme Web Terminal
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: fluentd-elasticsearch
  namespace: kube-system
  labels:
    k8s-app: fluentd-logging
spec:
  selector:
    matchLabels:
      name: fluentd-elasticsearch
  template:
    metadata:
      labels:
        name: fluentd-elasticsearch
    spec:
      tolerations:
        # this toleration is to have the daemonset runnable on master nodes
        # remove it if your masters can't run pods
        - key: node-role.kubernetes.io/master
          effect: NoSchedule
      containers:
        - name: nginx
          image: nginx
-- INSERT --
17,19 All
```



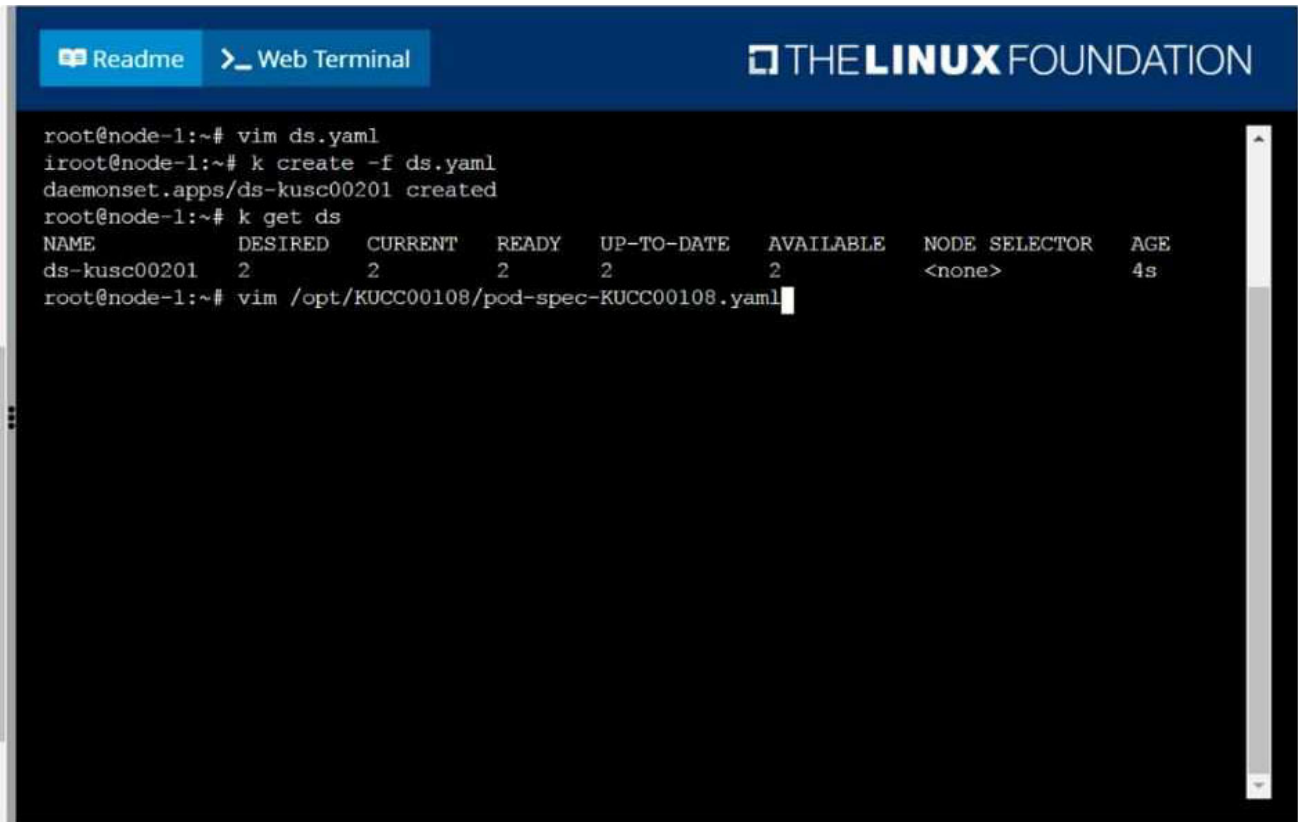
KUCC00108.yaml)

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:**

solution



The screenshot shows a web terminal interface with a dark background and light text. At the top, there are two tabs: 'Readme' and 'Web Terminal'. The 'Web Terminal' tab is active. The terminal output shows the following commands and results:

```
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
```

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
:wg

```

---

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:~#

```

## 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):



```
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:~#
```