

TESTPASSPORT

練習題



質 量 更 高

服 務 更 好

<https://www.testpassport.net>

免費一年更新服務

Exam : CKAD

**Title : Certified Kubernetes
Application Developer**

Version : DEMO

1.SIMULATION



Set Configuration Context:

`[student@node-1] $ | kubectl`

`config use-context k8s`

Context

A web application requires a specific version of redis to be used as a cache.

Task

Create a pod with the following characteristics, and leave it running when complete:

- The pod must run in the web namespace. The namespace has already been created
- The name of the pod should be cache
- Use the lfcncf/redis image with the 3.2 tag
- Expose port 6379

Answer:

Solution:



2.SIMULATION



Context

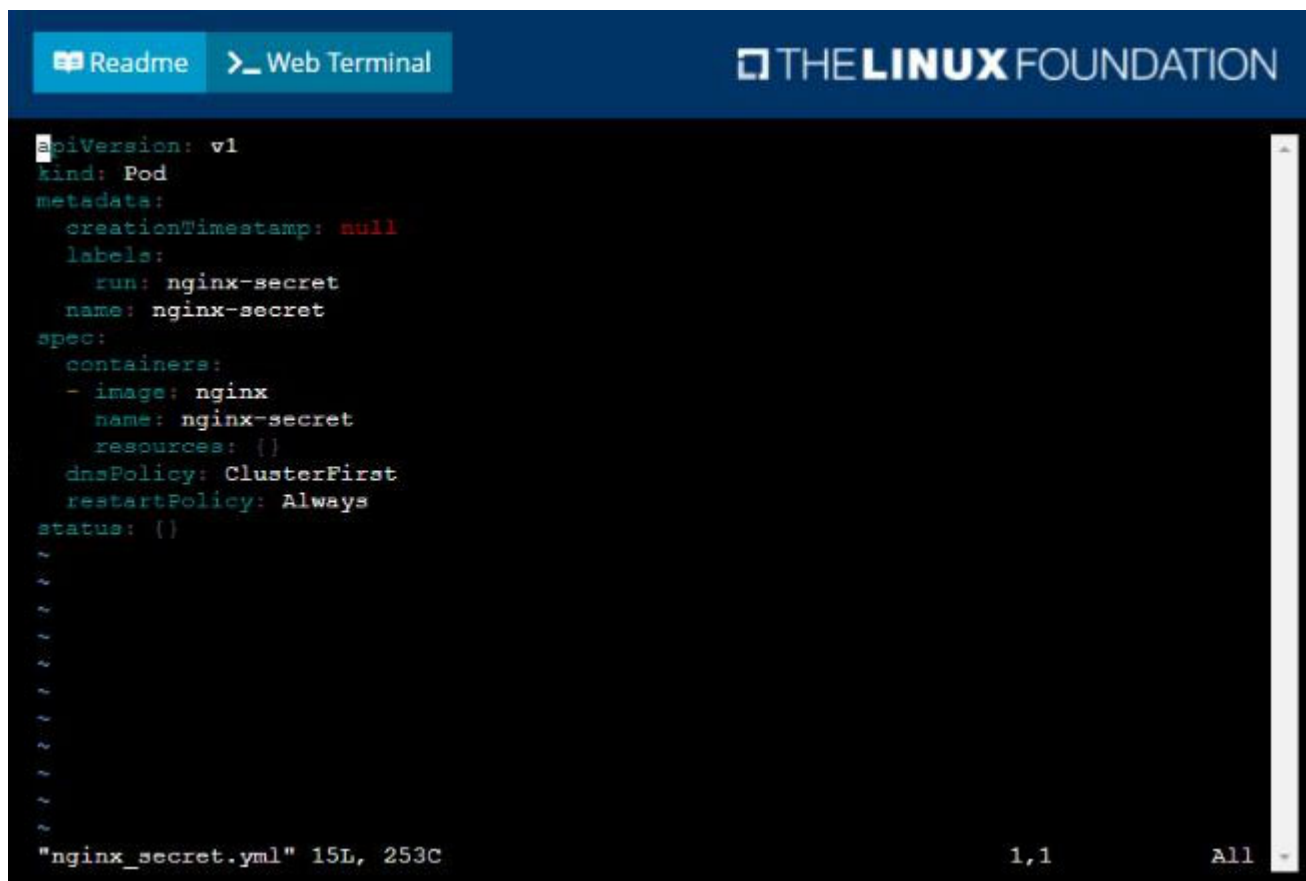
You are tasked to create a secret and consume the secret in a pod using environment variables as follow:

Task

- Create a secret named another-secret with a key/value pair; key1/value4
- Start an nginx pod named nginx-secret using container image nginx, and add an environment variable exposing the value of the secret key key 1, using COOL_VARIABLE as the name for the environment variable inside the pod

Answer:**Solution:**

```
student@node-1:~$ kubectl create secret generic some-secret --from-literal=key1=value4
secret/some-secret created
student@node-1:~$ kubectl get secret
NAME                                TYPE                                DATA  AGE
default-token-4kvr5                kubernetes.io/service-account-token  3      2d11h
some-secret                        Opaque                              1      5s
student@node-1:~$ kubectl run nginx-secret --image=nginx --dry-run=client -o yaml > nginx_secret.yml
student@node-1:~$ vim nginx_secret.yml
```



The screenshot shows a web terminal interface with a dark background. At the top, there are two tabs: 'Readme' and 'Web Terminal'. The 'Web Terminal' tab is active, displaying a YAML manifest for a pod named 'nginx-secret'. The manifest includes fields for 'apiVersion', 'kind', 'metadata' (with 'creationTimestamp' as null and 'labels' for 'run' and 'name'), 'spec' (with 'containers' for 'nginx-secret' using the 'nginx' image, 'resources' as an empty object, 'dnsPolicy' as 'ClusterFirst', and 'restartPolicy' as 'Always'), and 'status' as an empty object. The terminal also shows a status bar at the bottom indicating the file is 'nginx_secret.yml' with 15 lines and 253 characters, and a cursor at line 1, column 1.

```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-secret
    name: nginx-secret
spec:
  containers:
  - image: nginx
    name: nginx-secret
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}
```

"nginx_secret.yml" 15L, 253C 1,1 All

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-secret
    name: nginx-secret
spec:
  containers:
  - image: nginx
    name: nginx-secret
    env:
    - name: COOL_VARIABLE
      valueFrom:
        secretKeyRef:
          name: some-secret
          key: key1
~
~
~
~
~
~
~
~
-- INSERT --
16,20
All

```

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

student@node-1:~$ kubectl get pods -n web
NAME      READY   STATUS    RESTARTS   AGE
cache     1/1     Running   0           9s
student@node-1:~$ kubectl create secret generic some-secret --from-literal=key1=value4
secret/some-secret created
student@node-1:~$ kubectl get secret
NAME                TYPE          DATA   AGE
default-token-4kvr5  kubernetes.io/service-account-token  3       2d11h
some-secret          Opaque        1       5s
student@node-1:~$ kubectl run nginx-secret --image=nginx --dry-run=client -o yaml > nginx_secret.yml
student@node-1:~$ vim nginx_secret.yml
student@node-1:~$ kubectl create -f nginx_secret.yml
pod/nginx-secret created
student@node-1:~$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
liveness-http  1/1     Running   0           6h38m
nginx-101     1/1     Running   0           6h39m
nginx-secret  0/1     ContainerCreating  0           4s
poller       1/1     Running   0           6h39m
student@node-1:~$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
liveness-http  1/1     Running   0           6h38m
nginx-101     1/1     Running   0           6h39m
nginx-secret  1/1     Running   0           8s
poller       1/1     Running   0           6h39m
student@node-1:~$

```

3.SIMULATION

Set configuration context:



```
[student@node-1] $ | kubectl config  
use-context k8s
```

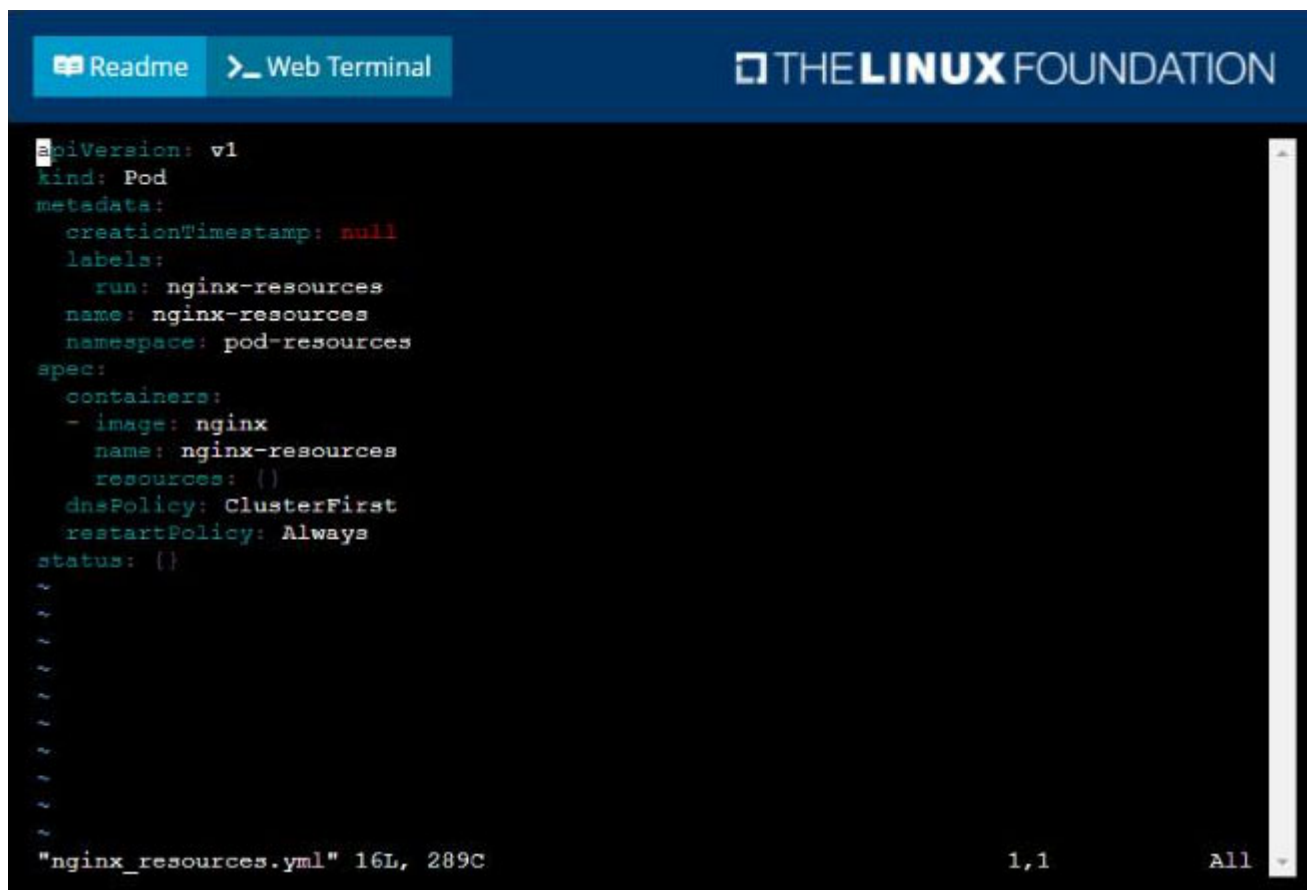
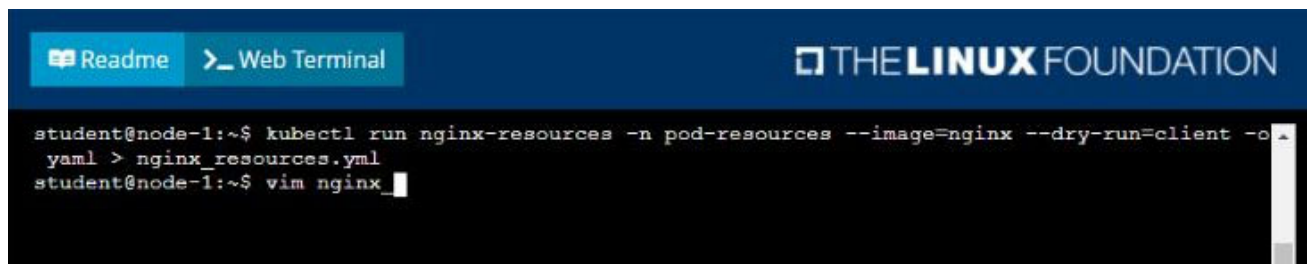
Task

You are required to create a pod that requests a certain amount of CPU and memory, so it gets scheduled to a node that has those resources available.

- Create a pod named `nginx-resources` in the `pod-resources` namespace that requests a minimum of 200m CPU and 1Gi memory for its container
- The pod should use the `nginx` image
- The `pod-resources` namespace has already been created

Answer:

Solution:



Readme
Web Terminal
THE **LINUX** FOUNDATION

```

apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-resources
    name: nginx-resources
    namespace: pod-resources
spec:
  containers:
  - image: nginx
    name: nginx-resources
    resources:
      requests:
        cpu: 200m
        memory: "1Gi"
~
~
~
~
~
~
~
~
~
~
-- INSERT --
15,22 All

```

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

student@node-1:~$ kubectl run nginx-resources -n pod-resources --image=nginx --dry-run=client -o
yaml > nginx_resources.yml
student@node-1:~$ vim nginx_resources.yml
student@node-1:~$ kubectl create -g nginx_resources.yml
Error: unknown shorthand flag: 'g' in -g
See 'kubectl create --help' for usage.
student@node-1:~$ kubectl create -f nginx_resources.yml
pod/nginx-resources created
student@node-1:~$ kubectl get pods -n pod-re

```

Readme
Web Terminal
THE **LINUX** FOUNDATION

```

student@node-1:~$ kubectl get pods -n pod-resources
NAME          READY   STATUS    RESTARTS   AGE
nginx-resources 1/1     Running   0           8s
student@node-1:~$

```

4.SIMULATION

Set configuration context:



```
[student@node-1] $ | kubectl config  
use-context k8s
```

Context

You are tasked to create a ConfigMap and consume the ConfigMap in a pod using a volume mount.

Task

Please complete the following:

- Create a ConfigMap named another-config containing the key/value pair: key4/value3
- start a pod named nginx-configmap containing a single container using the nginx image, and mount the key you just created into the pod under directory /also/a/path

Answer:

Solution:

```
student@node-1:~$ kubectl create configmap another-config --from-literal=key4=value3  
configmap/another-config created  
student@node-1:~$ kubectl get configmap  
NAME          DATA   AGE  
another-config 1       5s  
student@node-1:~$ kubectl run nginx-configmap --image=nginx --dry-run=client -o yaml > nginx_conf  
igmap.yml  
student@node-1:~$ vim nginx_configmap.yml ^C  
student@node-1:~$ mv nginx_configmap.yml nginx_configmap.yml  
student@node-1:~$ vim nginx_co
```



```

student@node-1:~$ kubectl create configmap another-config --from-literal=key4=value3
configmap/another-config created
student@node-1:~$ kubectl get configmap
NAME          DATA  AGE
another-config 1      5s
student@node-1:~$ kubectl run nginx-configmap --image=nginx --dry-run=client -o yaml > nginx_conf
igmap.yaml
student@node-1:~$ vim nginx_configmap.yaml ^C
student@node-1:~$ mv nginx_configmap.yaml nginx_configmap.yaml
student@node-1:~$ vim nginx_configmap.yaml
student@node-1:~$

```

```

student@node-1:~$ kubectl run nginx-configmap --image=nginx --dry-run=client -o yaml > nginx_conf
igmap.yaml
student@node-1:~$ vim nginx_configmap.yaml ^C
student@node-1:~$ mv nginx_configmap.yaml nginx_configmap.yaml
student@node-1:~$ vim nginx_configmap.yaml
student@node-1:~$ kubectl create f nginx_configmap.yaml
Error: must specify one of -f and -k

error: unknown command "f nginx_configmap.yaml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_configmap.yaml
error: error validating "nginx_configmap.yaml": error validating data: ValidationError(Pod.spec.c
ontainers[1]): unknown field "mountPath" in io.k8s.api.core.v1.Container; if you choose to ignor
e these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_configmap.yaml

```

[Readme](#)
[Web Terminal](#)

 THE **LINUX** FOUNDATION

```

student@node-1:~$ kubectl create f nginx_configmap.yaml
Error: must specify one of -f and -k

error: unknown command "f nginx_configmap.yaml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_configmap.yaml
error: error validating "nginx_configmap.yaml": error validating data: ValidationError(Pod.spec.c
ontainers[1]): unknown field "mountPath" in io.k8s.api.core.v1.Container; if you choose to ignor
e these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_configmap.yaml
student@node-1:~$ kubectl create -f nginx_configmap.yaml
pod/nginx-configmap created
student@node-1:~$ kubectl get pods
NAME          READY  STATUS      RESTARTS  AGE
liveness-http  1/1    Running     0          6h44m
nginx-101      1/1    Running     0          6h45m
nginx-configmap 0/1    ContainerCreating 0          5s
nginx-secret   1/1    Running     0          5m39s
poller         1/1    Running     0          6h44m
student@node-1:~$ kubectl get pods
NAME          READY  STATUS      RESTARTS  AGE
liveness-http  1/1    Running     0          6h44m
nginx-101      1/1    Running     0          6h45m
nginx-configmap 1/1    Running     0          8s
nginx-secret   1/1    Running     0          5m42s
poller         1/1    Running     0          6h45m
student@node-1:~$ l

```

5.SIMULATION

Set configuration context:



```
[student@node-1] $ | kubectl config  
use-context k8s
```

Context

Your application's namespace requires a specific service account to be used.

Task

Update the app-a deployment in the production namespace to run as the restrictedservice service account. The service account has already been created.

Answer:

Solution:

The screenshot shows a web terminal window with a dark blue header. On the left, there are two buttons: 'Readme' and 'Web Terminal'. On the right, the 'THE LINUX FOUNDATION' logo is displayed. The terminal content shows the following commands and output:

```
student@node-1:~$ kubectl get serviceaccount -n production
NAME                SECRETS  AGE
default              1        6h46m
restrictedservice    1        6h46m
student@node-1:~$ kubectl get deployment -n production
NAME    READY  UP-TO-DATE  AVAILABLE  AGE
app-a   3/3    3           3          6h46m
student@node-1:~$ kubectl set serviceaccount deployment app-a restrictedservice -n production
deployment.apps/app-a serviceaccount updated
student@node-1:~$
```