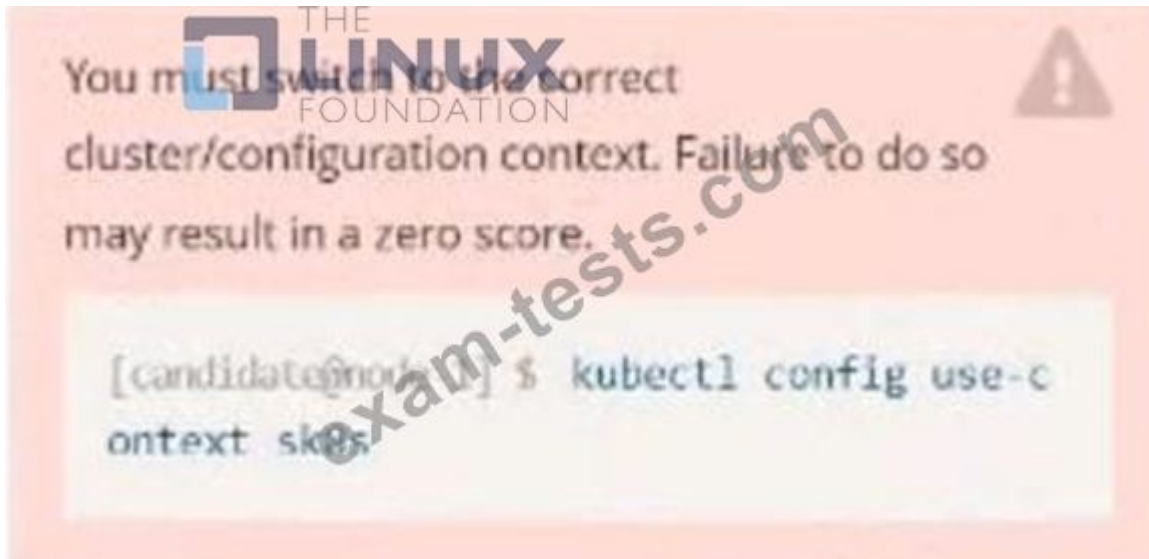


LinuxFoundation.CKAD.v2023-06-01.q33

Exam Code:	CKAD
Exam Name:	Linux Foundation Certified Kubernetes Application Developer Exam
Certification Provider:	Linux Foundation
Free Question Number:	33
Version:	v2023-06-01
# of views:	1540
# of Questions views:	330
https://www.exam-tests.com/CKAD-exam/LinuxFoundation.CKAD.v2023-06-01.q33.html	

NEW QUESTION: 1

Refer to Exhibit.



Task:

Update the Pod ckad00018-newpod in the ckad00018 namespace to use a NetworkPolicy allowing the Pod to send and receive traffic only to and from the pods web and db



Answer:

Solution:

```
candidate@node-1:~$ kubectl config use-context nk8s
Switched to context "nk8s".
candidate@node-1:~$ kubectl describe netpol -n ckad00018
```

```

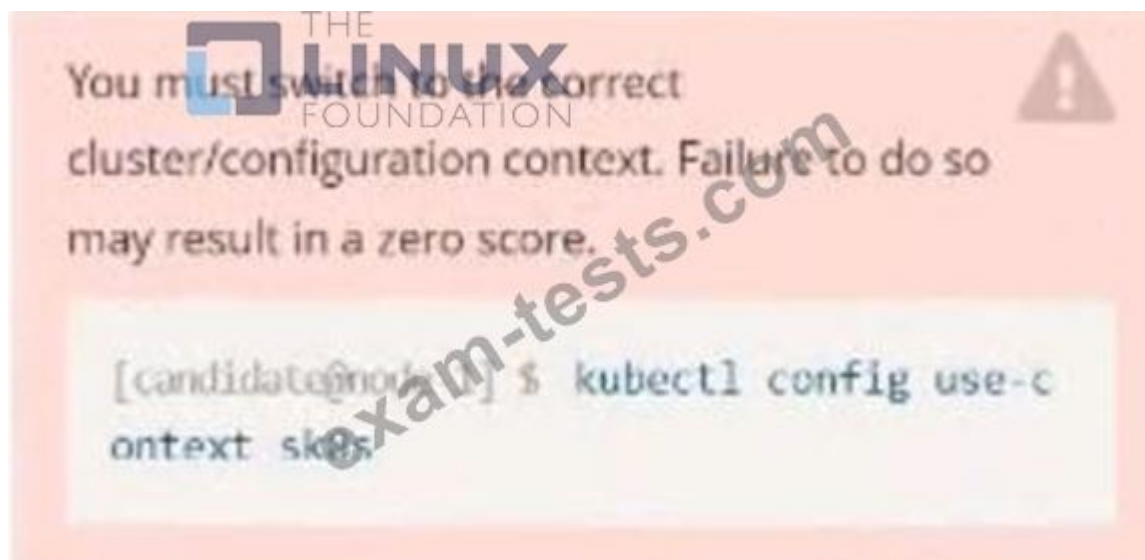
e Edit View Terminal Tabs Help
me: all-access
namespace: ckad00018
Created on: 2022-09-24 04:27:37 +0000 UTC
Labels: <none>
Annotations: <none>
Spec:
PodSelector: all-access=true
Allowing ingress traffic:
  To Port: <any> (traffic allowed to all ports)
  From: <any> (traffic not restricted by source)
Allowing egress traffic:
  To Port: <any> (traffic allowed to all ports)
  To: <any> (traffic not restricted by destination)
Policy Types: Ingress, Egress

Name: default-deny
Namespace: ckad00018
Created on: 2022-09-24 04:27:37 +0000 UTC
Labels: <none>
Annotations: <none>
Spec:
PodSelector: <none> (Allowing the specific traffic to all pods in this namespace)
Allowing ingress traffic:
  <none> (Selected pods are isolated for ingress connectivity)
Not affecting egress traffic
Policy Types: Ingress
candidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 web-access=true
pod/ckad00018-newpod labeled
candidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 db-access=true
pod/ckad00018-newpod labeled
candidate@node-1:~$

```

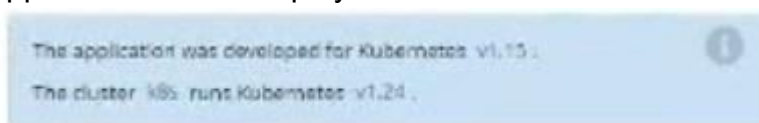
NEW QUESTION: 2

Refer to Exhibit.



Task:

1) Fix any API deprecation issues in the manifest file `-/credible-mite/www.yaml` so that this application can be deployed on cluster K8s.



2) Deploy the application specified in the updated manifest file `-/credible-mite/www.yaml` in namespace cobra

Answer:

Solution:

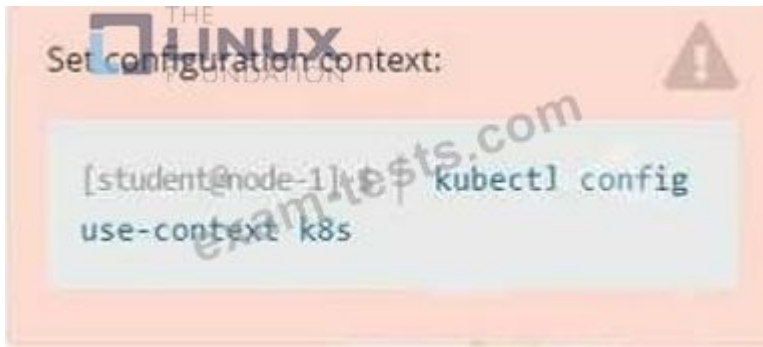
```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/credible-mite/www.yaml
```

```
File Edit View Terminal Tabs Help
apiVersion: apps/v1
kind: Deployment
metadata:
  name: www-deployment
  namespace: cobra
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: "nginx:stable"
          ports:
            - containerPort: 80
          volumeMounts:
            - mountPath: /var/log/nginx
              name: logs
          env:
            - name: NGINX_ENTRYPOINT_QUIET_LOGS
              value: "1"
      volumes:
        - name: logs
          emptyDir: {}
```

```
File Edit View Terminal Tabs Help
deployment.apps/expose created
candidate@node-1:~$ kubectl get pods -n ckad00014
NAME                                READY   STATUS              RESTARTS   AGE
expose-85dd99d4d9-25675             0/1     ContainerCreating   0           6s
expose-85dd99d4d9-4fhcc             0/1     ContainerCreating   0           6s
expose-85dd99d4d9-fl7j              0/1     ContainerCreating   0           6s
expose-85dd99d4d9-tt6rm             0/1     ContainerCreating   0           6s
expose-85dd99d4d9-vjd8b             0/1     ContainerCreating   0           6s
expose-85dd99d4d9-vtzpq             0/1     ContainerCreating   0           6s
candidate@node-1:~$ kubectl get deploy -n ckad00014
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
expose  6/6     6            6           15s
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/credible-mite/www.yaml
candidate@node-1:~$ vim ~/credible-mite/www.yaml
candidate@node-1:~$ kubectl apply -f ~/credible-mite/www.yaml
deployment.apps/www-deployment created
candidate@node-1:~$ kubectl get pods -n cobra
NAME                                READY   STATUS              RESTARTS   AGE
www-deployment-d899c6b49-d6ccg      1/1     Running            0           6s
www-deployment-d899c6b49-f796l      0/1     ContainerCreating   0           6s
www-deployment-d899c6b49-ztfcw      0/1     ContainerCreating   0           6s
candidate@node-1:~$ kubectl get deploy -n cobra
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
www-deployment  3/3     3            3           11s
candidate@node-1:~$ kubectl get pods -n cobra
NAME                                READY   STATUS              RESTARTS   AGE
www-deployment-d899c6b49-d6ccg      1/1     Running            0           14s
www-deployment-d899c6b49-f796l      1/1     Running            0           14s
www-deployment-d899c6b49-ztfcw      1/1     Running            0           14s
candidate@node-1:~$
```

NEW QUESTION: 3

Refer to Exhibit.



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:



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

1,1

All

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-configmap
  name: nginx-configmap
spec:
  containers:
    - image: nginx
      name: nginx-configmap
      volumeMounts:
        - name: myvol
          mountPath: /also/a/path
  volumes:
    - name: myvol
      configMap:
        name: another-config
```

13,6

All

```

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:~$

```

NEW QUESTION: 4

Context



Context

Developers occasionally need to submit pods that run periodically.

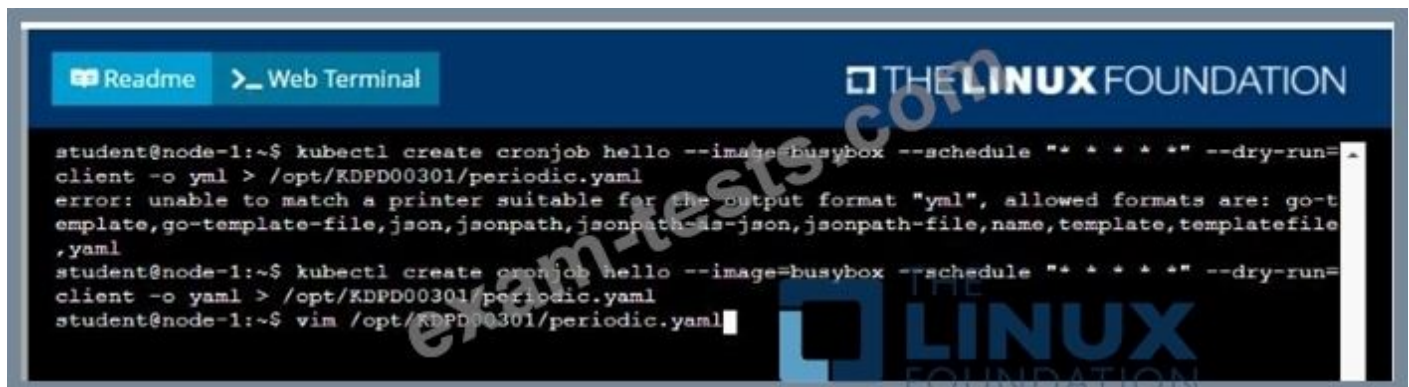
Task

Follow the steps below to create a pod that will start at a predetermined time and]which runs to completion only once each time it is started:

- * Create a YAML formatted Kubernetes manifest `/opt/KDPD00301/periodic.yaml` that runs the following shell command: `date` in a single busybox container. The command should run every minute and must complete within 22 seconds or be terminated by Kubernetes. The Cronjob name and container name should both be `hello`
- * Create the resource in the above manifest and verify that the job executes successfully at least once

Answer:

Solution:



Readme
Web Terminal

THE LINUX FOUNDATION

```

apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-c", "date"]
          restartPolicy: Never
  schedule: '*/* * * * *'
  startingDeadlineSeconds: 22
  concurrencyPolicy: Allow

```

19,26 All

Readme
Web Terminal

THE LINUX FOUNDATION

```

student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml". Allowed formats are: go-t
emplate, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, yaml, template, templatefile
, yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
student@node-1:~$ kubectl create -f /opt/KDPD00301/periodic.yaml
cronjob.batch/hello created
student@node-1:~$ kubectl get cronjob

```

NAME	SCHEDULE	SUSPEND	ACTIVE	LAST SCHEDULE	AGE
hello	*/* * * * *	False	0	<none>	6s

THE LINUX FOUNDATION

NEW QUESTION: 5

Context

THE LINUX FOUNDATION

You must switch to the correct cluster/configuration context. Failure to do so may result in a zero score.

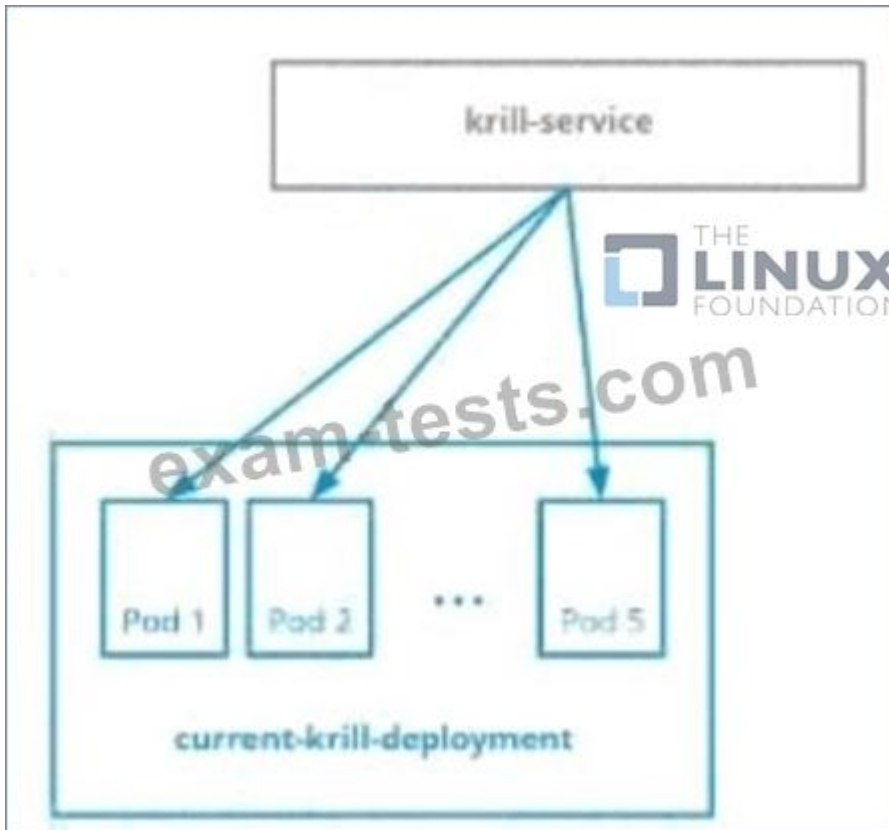
```
[candidate@node-1] $ kubectl config use-c
ontext skws
```

Context

You are asked to prepare a Canary deployment for testing a new application release.

Task:

A Service named krill-Service in the goshawk namespace points to 5 pod created by the Deployment named current-krill-deployment

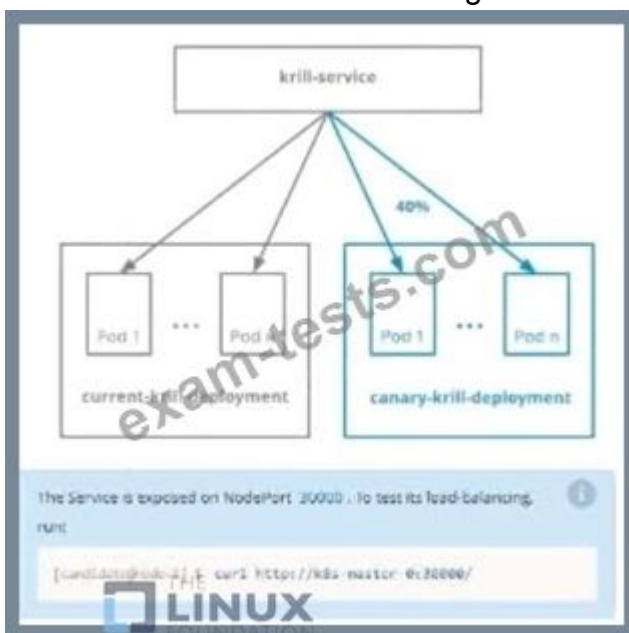


1) Create an identical Deployment named canary-kill-deployment, in the same namespace.

2) Modify the Deployment so that:

-A maximum number of 10 pods run in the goshawk namespace.

-40% of the krill-service 's traffic goes to the canary-krill-deployment pod(s)



Answer:

Solution:

```

candidate@node-1:~/humane-storks$ kubectl scale deploy canary-krill-deployment --replicas 4 -n goshawk
deployment.apps/canary-krill-deployment scaled
candidate@node-1:~/humane-storks$ kubectl get deploy -n goshawk
NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
canary-krill-deployment            4/4    4            4           4s
current-krill-deployment           5/5    5            5           7h22m
candidate@node-1:~/humane-storks$ wget https://k8s.io/examples/
2022-09-24 11:43:51-- https://k8s.io/examples/admin/resource/quota-pod.yaml
solving k8s.io (k8s.io)... 34.107.204.206, 2600:1901::26f3:
Connecting to k8s.io (k8s.io)[34.107.204.206]:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://kubernetes.io/examples/admin/resource/quota-pod.yaml [following]
2022-09-24 11:43:52-- https://kubernetes.io/examples/admin/resource/quota-pod.yaml
solving kubernetes.io (kubernetes.io)... 147.75.40.148
Connecting to kubernetes.io (kubernetes.io)[147.75.40.148]:443... connected.
HTTP request sent, awaiting response... 200 OK
Content-length: 90 [application/x-yaml]
Saving to: 'quota-pod.yaml'

quota-pod.yaml           100%[=====] 90 ---KB/s   in 0s

2022-09-24 11:43:52 (15.0 MB/s) - 'quota-pod.yaml' saved [90/90]

candidate@node-1:~/humane-storks$ vim quota-pod.yaml
File Edit View Terminal Tabs Help
2022-09-24 11:43:52 (15.0 MB/s) - 'quota-pod.yaml' saved [90/90]

candidate@node-1:~/humane-storks$ vim quota-pod.yaml
candidate@node-1:~/humane-storks$ kubectl create -f quota-pod.yaml
resourcequota/pod-demo created
candidate@node-1:~/humane-storks$ kubectl get quota -n go
No resources found in go namespace.
candidate@node-1:~/humane-storks$ kubectl get quota -n goshawk
NAME      AGE  REQUEST  LIMIT
pod-demo  19s  pods: 9/10
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
current-krill-deployment-fb7c7995c-kvtjr
app.kubernetes.io/name=current
app.kubernetes.io/part-of=krill
pod-template-hash=fb7c7995c
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
current-krill-deployment-fb7c7995c-4whfm
app.kubernetes.io/name=current
app.kubernetes.io/part-of=krill
pod-template-hash=fb7c7995c
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-dfk7l
app.kubernetes.io/name=canary
app.kubernetes.io/part-of=krill
pod-template-hash=5f78fd4786
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-z5zrt
app.kubernetes.io/name=canary
app.kubernetes.io/part-of=krill
pod-template-hash=5f78fd4786
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-2774b
app.kubernetes.io/name=canary
app.kubernetes.io/part-of=krill
pod-template-hash=5f78fd4786
candidate@node-1:~/humane-storks$

```

NEW QUESTION: 6



Context

You sometimes need to observe a pod's logs, and write those logs to a file for further analysis.

Task

Please complete the following;

- * Deploy the counter pod to the cluster using the provided YAMLSpec file

at /opt/KDOB00201/counter.yaml

- * Retrieve all currently available application logs from the running pod and store them in the file /opt/KDOB00201/log_Output.txt, which has already been created

Answer:

See the solution below.

Explanation

Solution:

```
pod/counter created
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
counter       1/1     Running   0           10s
liveness-http 1/1     Running   0           6h45m
nginx-101     1/1     Running   0           6h46m
nginx-configmap 1/1     Running   0           107s
nginx-secret  1/1     Running   0           7m21s
poller        1/1     Running   0           6h46m
student@node-1:~$ kubectl logs counter
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828fbc5e8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ 
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
```

```
Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828f8e5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfec9a894a0604fc4b814b37d0a200a4
11: 5493cd16a1790a5fb9512b0c9d4c5dd1
12: 03f169e93e6143438e6dfe4ecb3cc9ed
13: 764b37fe611373c42d0b47154041f6eb
14: 1a56fbc1896b0ee6394136166281839e
15: ecc492eb17715de090c47345a98a98a3
16: 7974a6bec0fb44b6b8bbfc71a32f8a74
17: 9ae01bef01748b12cc9f97a01f72cd6
18: 23fb22ee34d4272e4c0e005f1774515f
19: ec7e1a5d314da9a0ad4e1b3be5a7acae
20: 0bccdd8ee02cd42029e8162cd1c1197c
21: d6851ea43546216b95bcb81ced997102
22: 7ed9a38ea8bf0d86206569481442af44
23: 29b8416ddc63dbfcb987ab3c8198e9fe
24: 1f2062001df51a108ab25010f506716f
student@node-1:~$
```

NEW QUESTION: 7



Context

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

Task

* From the pods running in namespace `cpu-stress`, write the name only of the pod that is consuming the most CPU to file `/opt/KDOB030/pod.txt`, which has already been created.

Answer:

See the solution below.

Explanation

Solution:

```
Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl top pods -n cpu-stress
NAME          CPU(cores)   MEMORY(bytes)
max-load-98b9se 68m          6Mi
max-load-ab2d3s 21m          6Mi
max-load-kipb9a 45m          6Mi
student@node-1:~$ echo "max-load-98b9se" > /opt/K50800301/pod.txt
```

NEW QUESTION: 8

Refer to Exhibit.

No configuration context change is required for this task.

Task:

A Dockerfile has been prepared at `~/human-stork/build/Dockerfile`

1) Using the prepared Dockerfile, build a container image with the name `macque` and tag `3.0`. You may install and use the tool of your choice.

Multiple image builders and tools have been pre-installed in the base system, including: `docker`, `skopeo`, `colima`, `img`, and `podman`.

Please do not push the built image to a registry, run a container, or otherwise consume it.

2) Using the tool of your choice export the built container image in OC-format and store it at `~/human stork/macque 3.0 tar`

Answer:

Solution:

```
candidate@node-1:~$ cd humane-stork/build/
candidate@node-1:~/humane-stork/build$ ls -l
total 16
-rw-r--r-- 1 candidate candidate 201 Sep 24 04:21 Dockerfile
-rw-r--r-- 1 candidate candidate 644 Sep 24 04:21 text1.html
-rw-r--r-- 1 candidate candidate 813 Sep 24 04:21 text2.html
-rw-r--r-- 1 candidate candidate 383 Sep 24 04:21 text3.html
candidate@node-1:~/humane-stork/build$ sudo docker build -t macaque:3.0 .
Sending build context to Docker daemon  6.144kB
Step 1/5 : FROM docker.io/lfcccncf/nginx:mainline
--> ea335eea17ab
Step 2/5 : ADD text1.html /usr/share/nginx/html/
--> 8967ee9ee5d8
Step 3/5 : ADD text2.html /usr/share/nginx/html/
--> cb0554422f26
Step 4/5 : ADD text3.html /usr/share/nginx/html/
--> 62e879ab821e
Step 5/5 : COPY text2.html /usr/share/nginx/html/index.html
--> 331c8a94372c
Successfully built 331c8a94372c
Successfully tagged macaque:3.0
candidate@node-1:~/humane-stork/build$ sudo docker save macaque:3.0 > ~/humane-stork/macaque-3.0.tar
candidate@node-1:~/humane-stork/build$ cd ..
candidate@node-1:~/humane-stork$ ls -l
total 142532
drwxr-xr-x 2 candidate candidate 4096 Sep 24 04:21 build
-rw-rw-r-- 1 candidate candidate 145948672 Sep 24 11:39 macaque-3.0.tar
candidate@node-1:~/humane-stork$
```

```

File Edit View Terminal Tabs Help
pod/ckad00018-newpod labeled
candidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 db-access=true
pod/ckad00018-newpod labeled
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/chief-cardinal/nosql.yaml
candidate@node-1:~$ vim ~/chief-cardinal/nosql.yaml
candidate@node-1:~$ kubectl apply -f ~/chief-cardinal/nosql.yaml
deployment.apps/nosql configured
candidate@node-1:~$ kubectl get pods -n crayfish
NAME                                READY   STATUS    RESTARTS   AGE
nosql-74ccc7d64-lkqlg               1/1     Running   0           3m2s
candidate@node-1:~$ kubectl get deploy -n crayfish
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
nosql   1/1     1             1           7h16m
candidate@node-1:~$ cd humane-stork/build/
candidate@node-1:~/humane-stork/build$ ls -l
total 16
-rw-r--r-- 1 candidate candidate 201 Sep 24 04:21 Dockerfile
-rw-r--r-- 1 candidate candidate 644 Sep 24 04:21 text1.html
-rw-r--r-- 1 candidate candidate 813 Sep 24 04:21 text2.html
-rw-r--r-- 1 candidate candidate 383 Sep 24 04:21 text3.html
candidate@node-1:~/humane-stork/build$ sudo docker build -t macaque:3.0 .
Sending build context to Docker daemon 6.144kB
Step 1/5 : FROM docker.io/lfccnf/nginx:mainline
--> ea335eeal7ab
Step 2/5 : ADD text1.html /usr/share/nginx/html/
--> 8967ee9ee5d0
Step 3/5 : ADD text2.html /usr/share/nginx/html/
--> cb0554422f26
Step 4/5 : ADD text3.html /usr/share/nginx/html/

```

```

File Edit View Terminal Tabs Help
candidate@node-1:~$ vim ~/chief-cardinal/nosql.yaml
candidate@node-1:~$ kubectl apply -f ~/chief-cardinal/nosql.yaml
deployment.apps/nosql configured
candidate@node-1:~$ kubectl get pods -n crayfish
NAME                                READY   STATUS    RESTARTS   AGE
nosql-74ccc7d64-lkqlg               1/1     Running   0           3m2s
candidate@node-1:~$ kubectl get deploy -n crayfish
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
nosql   1/1     1             1           7h16m
candidate@node-1:~$ cd humane-stork/build/
candidate@node-1:~/humane-stork/build$ ls -l
total 16
-rw-r--r-- 1 candidate candidate 201 Sep 24 04:21 Dockerfile
-rw-r--r-- 1 candidate candidate 644 Sep 24 04:21 text1.html
-rw-r--r-- 1 candidate candidate 813 Sep 24 04:21 text2.html
-rw-r--r-- 1 candidate candidate 383 Sep 24 04:21 text3.html
candidate@node-1:~/humane-stork/build$ sudo docker build -t macaque:3.0 .
Sending build context to Docker daemon 6.144kB
Step 1/5 : FROM docker.io/lfccnf/nginx:mainline
--> ea335eeal7ab
Step 2/5 : ADD text1.html /usr/share/nginx/html/
--> 8967ee9ee5d0
Step 3/5 : ADD text2.html /usr/share/nginx/html/
--> cb0554422f26
Step 4/5 : ADD text3.html /usr/share/nginx/html/
--> 62e879ab821e
Step 5/5 : COPY text2.html /usr/share/nginx/html/index.html
--> 331c8a94372c
Successfully built 331c8a94372c
Successfully tagged macaque:3.0
candidate@node-1:~/humane-stork/build$ sudo docker save macaque:3.0 > ~/humane-stork/macaque-3.0.tar

```

NEW QUESTION: 9

Context



Task

Create a new deployment for running nginx with the following parameters;

- * Run the deployment in the kdpd00201 namespace. The namespace has already been created
- * Name the deployment frontend and configure with 4 replicas
- * Configure the pod with a container image of lfcncf/nginx:1.13.7
- * Set an environment variable of NGINX__PORT=8080 and also expose that port for the container above

Answer:

Solution:



```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: api
  name: api
  namespace: kdpd00201
spec:
  replicas: 4
  selector:
    matchLabels:
      app: api
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: api
    spec:
      containers:
        - image: lfccncf/nginx:1.13.7-alpine
          name: nginx
          resources: {}
status: {}
~
"nginx_deployment.yml" 25L, 421C
```

4,1

All

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: api
  name: api
  namespace: kdpd00201
spec:
  replicas: 4
  selector:
    matchLabels:
      app: api
  template:
    metadata:
      labels:
        app: api
    spec:
      containers:
        - image: lfccncf/nginx:1.13.7-alpine
          name: nginx
          ports:
            - containerPort: 8080
          env:
            - name: NGINX_PORT
              value: "8080"
```

23,8

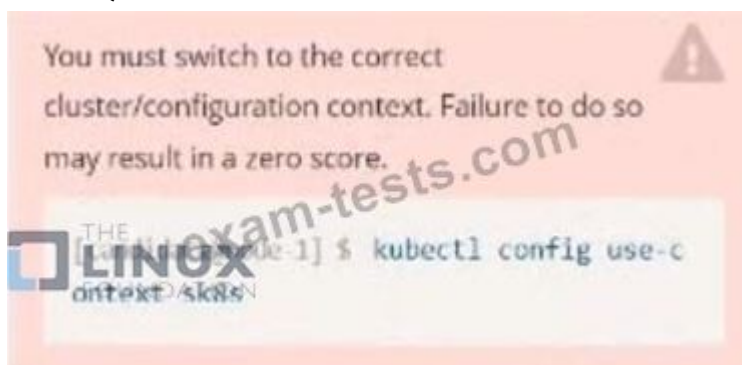
All

```
Readme Web Terminal THE LINUX FOUNDATION

student@node-1:~$ kubectl create deployment api --image=lfcncf/nginx:1.13.7-alpine --replicas=4
-n kdpd00201 --dry-run=client -o yaml > nginx_deployment.yml
student@node-1:~$ vim nginx_deployment.yml
student@node-1:~$ kubectl create nginx_deployment.yml
Error: must specify one of -f and -k

error: unknown command "nginx_deployment.yml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_deployment.yml
error: error validating "nginx_deployment.yml": error validating data: ValidationError(Deployment.spec.template.spec): unknown field "env" in io.k8s.api.core.v1.PodSpec; if you choose to ignore these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_deployment.yml
student@node-1:~$ kubectl create -f nginx_deployment.yml
deployment.apps/api created
student@node-1:~$ kubectl get pods -n kdpd00201
NAME                                READY    STATUS    RESTARTS   AGE
api-745677f7dc-7hnvm                1/1      Running   0           13s
api-745677f7dc-9q5vp                1/1      Running   0           13s
api-745677f7dc-fd4gk                1/1      Running   0           13s
api-745677f7dc-mbnpc                1/1      Running   0           13s
student@node-1:~$
```

NEW QUESTION: 10

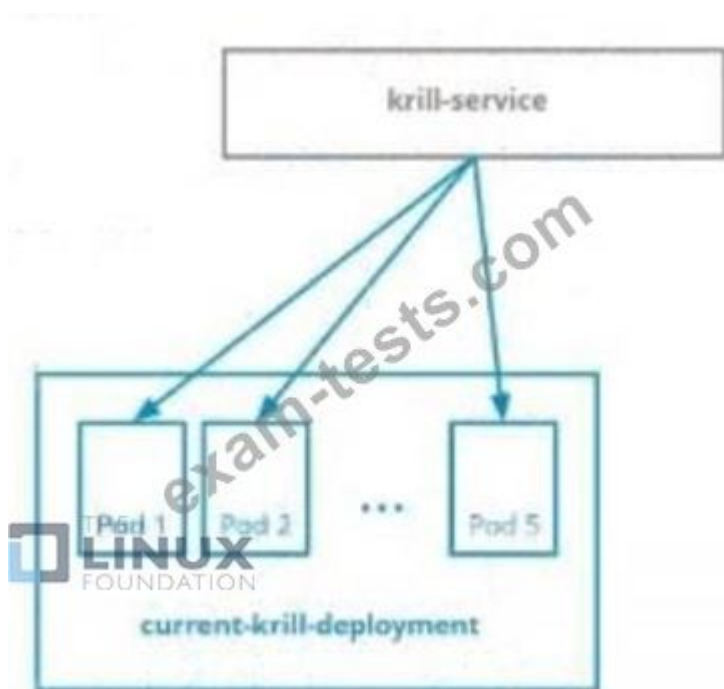


Context

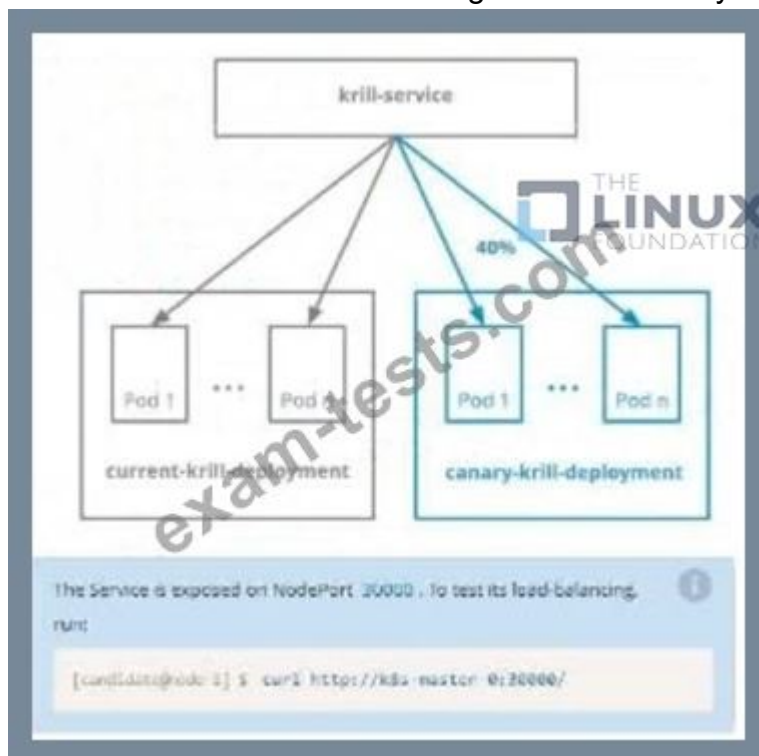
You are asked to prepare a Canary deployment for testing a new application release.

Task:

A Service named krill-Service in the goshark namespace points to 5 pod created by the Deployment named current-krill-deployment



- 1) Create an identical Deployment named **canary-kill-deployment**, in the same namespace.
- 2) Modify the Deployment so that:
 - A maximum number of 10 pods run in the goshawk namespace.
 - 40% of the krill-service 's traffic goes to the canary-krill-deployment pod(s)



Answer:

See the solution below.

Explanation

Solution:

```

candidate@node-1:~/humane-storks$ kubectl scale deploy canary-krill-deployment --replicas 4 -n goshawk
deployment.apps/canary-krill-deployment scaled
candidate@node-1:~/humane-storks$ kubectl get deploy -n goshawk
NAME                    READY    UP-TO-DATE    AVAILABLE    AGE
canary-krill-deployment 4/4      4             4            46s
current-krill-deployment 5/5      5             5            7h22m
candidate@node-1:~/humane-storks$ wget https://k8s.io/examples/

```

Text Description automatically generated

```

candidate@node-1:~/humane-storks$ wget https://k8s.io/examples/admin/resource/quota-pod.yaml
2022-09-24 11:43:51-- https://k8s.io/examples/admin/resource/quota-pod.yaml
solving k8s.io (k8s.io)... 34.107.204.206, 2600:1901:0:26f3::
connecting to k8s.io (k8s.io)[34.107.204.206]:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://kubernetes.io/examples/admin/resource/quota-pod.yaml [following]
2022-09-24 11:43:52-- https://kubernetes.io/examples/admin/resource/quota-pod.yaml
solving kubernetes.io (kubernetes.io)... 147.75.40.148
connecting to kubernetes.io (kubernetes.io)[147.75.40.148]:443... connected.
HTTP request sent, awaiting response... 200 OK
Content-Length: 90 [application/x-yaml]
Saving to: 'quota-pod.yaml'

quota-pod.yaml           100%[=====] 90 --.-KB/s  in 0s

2022-09-24 11:43:52 (15.0 MB/s) - 'quota-pod.yaml' saved [90/90]

candidate@node-1:~/humane-storks$ vim quota-pod.yaml

```

File Edit View Terminal Tabs Help

```

2022-09-24 11:43:52 (15.0 MB/s) - 'quota-pod.yaml' saved [90/90]

candidate@node-1:~/humane-storks$ vim quota-pod.yaml
candidate@node-1:~/humane-storks$ kubectl create -f quota-pod.yaml
resourcequota/pod-demo created
candidate@node-1:~/humane-storks$ kubectl get quota -n go
No resources found in go namespace.
candidate@node-1:~/humane-storks$ kubectl get quota -n goshawk
NAME      AGE  REQUEST  LIMIT
pod-demo  19s  pods: 9/10
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
current-krill-deployment-fb7c7995c-kvtjr
app.kubernetes.io/name="current"
app.kubernetes.io/part-of="krill"
pod-template-hash="fb7c7995c"
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
current-krill-deployment-fb7c7995c-4whfm
app.kubernetes.io/name="current"
app.kubernetes.io/part-of="krill"
pod-template-hash="fb7c7995c"
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-dfk7l
app.kubernetes.io/name="canary"
app.kubernetes.io/part-of="krill"
pod-template-hash="5f78fd4786"
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-z5zrt
app.kubernetes.io/name="canary"
app.kubernetes.io/part-of="krill"
pod-template-hash="5f78fd4786"
candidate@node-1:~/humane-storks$ curl http://k8s-master-0:30000/
canary-krill-deployment-5f78fd4786-2774b
app.kubernetes.io/name="canary"
app.kubernetes.io/part-of="krill"
pod-template-hash="5f78fd4786"
candidate@node-1:~/humane-storks$

```

NEW QUESTION: 11

Context

Set configuration context:

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

Context

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

Task

* From the pods running in namespace cpu-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.

Answer:

Solution:

Readme Web Terminal

THE LINUX FOUNDATION

```
student@node-1:~$ kubectl top pods -n cpu-stress  
NAME          CPU(cores)   MEMORY(bytes)  
max-load-98b9se 68m          6Mi  
max-load-ab2d3s 21m          4Mi  
max-load-kipb9a 45m          6Mi  
student@node-1:~$ echo "max-load-98b9se" > /opt/KDOBG0030I/pod.txt
```

NEW QUESTION: 12

Context

Set configuration context:

```
[student@node-1] $ kubectl config  
se-context nk8s
```

Task

A deployment is failing on the cluster due to an incorrect image being specified. Locate the deployment, and fix the problem.

Answer:

create deploy hello-deploy --image=nginx --dry-run=client -o yaml > hello-deploy.yaml Update deployment image to nginx:1.17.4: kubectl set image deploy/hello-deploy nginx=nginx:1.17.4

NEW QUESTION: 13



Context

A container within the poller pod is hard-coded to connect the nginxsvc service on port 90 . As this port changes to 5050 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 port 5050.
- * Add an HAproxy container named haproxy bound to port 90 to the 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 See the solution below.

Answer:

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 -l run=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:

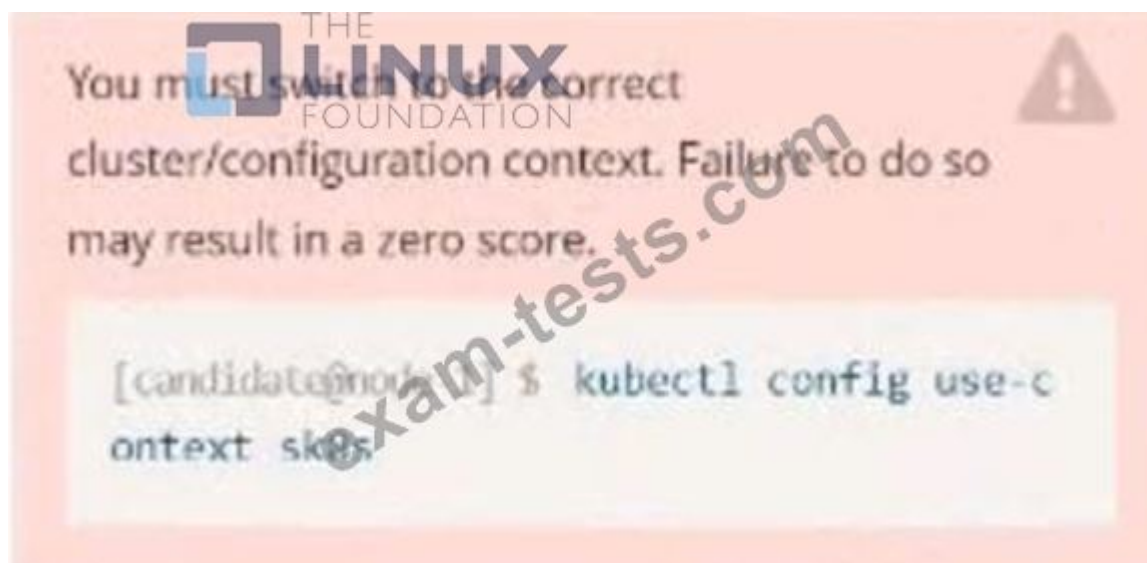
```
kubectl get pods -l run=my-nginx -o yaml | grep podIP
```

```
podIP: 10.244.3.4
```

```
podIP: 10.244.2.5
```

NEW QUESTION: 14

Refer to Exhibit.



Task:

Modify the existing Deployment named broker-deployment running in namespace quetzal so that its containers.

1) Run with user ID 30000 and

2) Privilege escalation is forbidden

The broker-deployment manifest file can be found at:

```
~/daring-moccasin/broker-deployment.yaml
```

Answer:

Solution:

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim
```

```
File Edit View Terminal Tabs Help
containers:
- name: broker
  image: redis:alpine
  ports:
  - containerPort: 6379
  securityContext:
    runAsUser: 30000
    privileged: false
```

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/daring-moccasin/broker-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/daring-moccasin/broker-deployment.yaml
deployment.apps/broker-deployment configured
candidate@node-1:~$ kubectl get pods -n quetzal
```

NAME	READY	STATUS	RESTARTS	AGE
broker-deployment-65446d6d94-868p6	1/1	Running	0	30s
broker-deployment-65446d6d94-8dn7l	1/1	Running	0	32s
broker-deployment-65446d6d94-p4h4l	1/1	Running	0	31s

```
candidate@node-1:~$ kubectl get deployment -n quetzal
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
broker-deployment	3/3	3	3	7h3m

```
candidate@node-1:~$
```

NEW QUESTION: 15

Exhibit:

Set configuration context:

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



Context

You sometimes need to observe a pod's logs, and write those logs to a file for further analysis.

Task

Please complete the following;

- * Deploy the counter pod to the cluster using the provided YAMLSpec file

at /opt/KDOB00201/counter.yaml

- * Retrieve all currently available application logs from the running pod and store them in the file /opt/KDOB00201/log_Output.txt, which has already been created

A. Solution:

```
pod/counter created  
student@node-1:~$ kubectl get pods  
NAME          READY   STATUS    RESTARTS   AGE  
counter       1/1     Running   0           10s  
liveness-http 1/1     Running   0           6h45m  
nginx-101     1/1     Running   0           6h46m  
nginx-configmap 1/1     Running   0           107s  
nginx-secret  1/1     Running   0           7m21s  
poller        1/1     Running   0           6h46m  
student@node-1:~$ kubectl logs counter  
1: 2b305101817ae25ca60ae46510fb6d11  
2: 3648cf2eae95ab680dba8f195f891af4  
3: 65c8bbd4dbf70bf81f2a0984a3a44ede  
4: 40d3a9c8e46f5533bb4828fbc5c8d038  
5: 390442d2530a90c3602901e3fe999ac8  
6: b71d95187417e139effb33af77681040  
7: 66a8e55a6491e756d2d0549ad6ab90a  
8: ff2b3d583b64125d2f9129c443bb37ff  
9: b6c6a12b6e77944ed8baaaf6c242dae4  
10: bfc9a894a0604fc4b814b37d0a200a4  
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt  
student@node-1:~$  
  
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt  
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt  
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
```

```

student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828f8e5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
11: 5493cd16a1790a5fb9512b0c9d4c5dd1
12: 03f169e93e6143438e6dfe4ecb3cc9ed
13: 764b37fe611373c42d0b47154041f6eb
14: 1a56fbe1896b0ee6394136166281839e
15: ecc492eb17715de090c47345a98d98d3
16: 7974a6bec0fb44b6b8bbfc71aa3f8e74
17: 9ae01bef01748b12cd9f97a5f9f72cd6
18: 23fb22ee34d4272e4c9e005f1774515f
19: ec7e1a5d314da9a0ad45d53be5a7acae
20: 0bccdd8ee02cd42029e8162cd1c1197c
21: d6851ea43546216b95bcb81ced997102
22: 7ed9a38ea8bf0d86206569481442af44
23: 29b8416ddc63dbfcb987ab3c8198e9fe
24: 1f2062001df51a108ab25010f506716f
student@node-1:~$

```

B. Solution:

```

pod/counter created
student@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
counter       1/1     Running   0           10s
liveness-http 1/1     Running   0           6h45m
nginx-101     1/1     Running   0           6h46m
nginx-configmap 1/1     Running   0           107s
nginx-secret  1/1     Running   0           7m21s
poller        1/1     Running   0           6h46m
student@node-1:~$ kubectl logs counter
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828f8e5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$

```

```
Readme Web Terminal THE LINUX FOUNDATION
student@node-1:~$ kubectl logs counter > /opt/KDOB00201/log_output.txt
student@node-1:~$ cat /opt/KDOB00201/log_output.txt
1: 2b305101817ae25ca60ae46510fb6d11
2: 3648cf2eae95ab680dba8f195f891af4
3: 65c8bbd4dbf70bf81f2a0984a3a44ede
4: 40d3a9c8e46f5533bb4828f8e5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
11: 5493cd16a1790a5fb9512b0c9d4c5dd1
12: 03f169e93e6143438e6dfe4ecb3cc9ed
13: 764b37fe611373c42d0b47154041f6eb
14: 1a56fbe1896b0ee6394136166281839e
15: ecc492eb17715de090c47345a98d98d3
16: 7974a6bec0fb44b6b8bbfc71aa3f8e74
17: 9ae01bef01748b12cd9f97a5f9f72cd6
18: 23fb22ee34d4272e4c9e005f1774515f
19: ec7e1a5d314da9a0ad45d53be5a7acae
20: 0bccdd8ee02cd42029e8162cd1c1197c
21: d6851ea43546216b95bcb81ced997102
22: 7ed9a38ea8bf0d86206569481442af44
23: 29b8416ddc63dbfcb987ab3c8198e9fe
24: 1f2062001df51a108ab25010f506716f
student@node-1:~$
```

Answer: A ([LEAVE A REPLY](#))

NEW QUESTION: 16

Exhibit:



Context

Developers occasionally need to submit pods that run periodically.

Task

Follow the steps below to create a pod that will start at a predetermined time and which runs to completion only once each time it is started:

- * Create a YAML formatted Kubernetes manifest `/opt/KDPD00301/periodic.yaml` that runs the following shell command: `date` in a single busybox container. The command should run every minute and must complete within 22 seconds or be terminated by Kubernetes. The Cronjob name and container name should both be `hello`

* Create the resource in the above manifest and verify that the job executes successfully at least once

A. Solution:

ReadmeWeb Terminal

THE LINUX FOUNDATION

```
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, name, template, templatefile
, yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
```

ReadmeWeb Terminal

THE LINUX FOUNDATION

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-c", "date"]
              restartPolicy: Never
          schedule: "*/* * * * *"
          startingDeadlineSeconds: 22
          concurrencyPolicy: Allow
```

19,26All

ReadmeWeb Terminal

THE LINUX FOUNDATION

```
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate, go-template-file, json, jsonpath, jsonpath-as-json, jsonpath-file, name, template, templatefile
, yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
student@node-1:~$ kubectl create -f /opt/KDPD00301/periodic.yaml
cronjob.batch/hello created
student@node-1:~$ kubectl get cronjob
NAME SCHEDULE SUSPEND ACTIVE LAST SCHEDULE AGE
hello */1 * * * * False 0 <none> 6s
student@node-1:~$
```

B. Solution:

```
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate,go-template-file,json,jsonpath,jsonpath-as-json,jsonpath-file,name,template,templatefile
,yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
```

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-c", "date"]
              restartPolicy: Never
          schedule: '* */1 * * * *'
          startingDeadlineSeconds: 22
          concurrencyPolicy: Allow
```

Answer: A ([LEAVE A REPLY](#))

Valid CKAD Dumps shared by BraindumpsPass.com for Helping Passing CKAD Exam!

BraindumpsPass.com now offer the **newest CKAD exam dumps**, the BraindumpsPass.com

CKAD exam **questions have been updated** and **answers have been corrected** get the

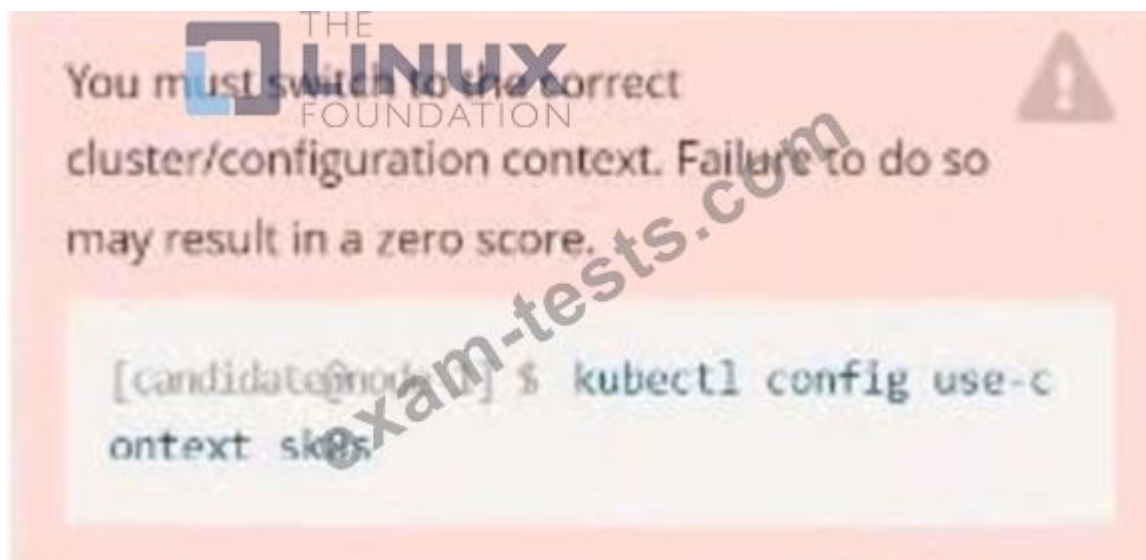
newest BraindumpsPass.com CKAD dumps with Test Engine here:

<https://www.braindumpsPASS.com/Linux-Foundation/CKAD-practice-exam-dumps.html> (193

Q&As Dumps, **40%OFF** Special Discount: **Exam-Tests**)

NEW QUESTION: 17

Refer to Exhibit.



Task:

Update the Deployment app-1 in the frontend namespace to use the existing ServiceAccount app.

Answer:

Solution:

```
File Edit View Terminal Tabs Help
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

candidate@node-1:~$ vi ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl config use-context sk8s
Switched to context "sk8s".
candidate@node-1:~$ vim .vimrc
candidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/spicy-pikachu/backend-deployment.yaml
deployment.apps/backend-deployment configured
candidate@node-1:~$ kubectl get pods -n staging
NAME                                READY   STATUS    RESTARTS   AGE
backend-deployment-59d449b99d-cxct6 1/1     Running   0           20s
backend-deployment-59d449b99d-h2zjq 0/1     Running   0           9s
backend-deployment-78976f74f5-b8c85 1/1     Running   0           6h40m
backend-deployment-78976f74f5-flfsj 1/1     Running   0           6h40m
candidate@node-1:~$ kubectl get deploy -n staging
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
backend-deployment  3/3     3            3           6h40m
candidate@node-1:~$ kubectl get deploy -n staging
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
backend-deployment  3/3     3            3           6h41m
candidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl set serviceaccount deployment/app-1 app -n frontend
deployment.apps/app-1 serviceaccount updated
candidate@node-1:~$
```

NEW QUESTION: 18



Context

A user has reported an application is unteachable due to a failing livenessProbe .

Task

Perform the following tasks:

- * Find the broken pod and store its name and namespace to /opt/KDOB00401/broken.txt in the format:

```
<namespace>/<pod>
```

The output file has already been created

- * Store the associated error events to a file /opt/KDOB00401/error.txt, The output file has already been created. You will need to use the -o wide output specifier with your command
- * Fix the issue.

The associated deployment could be running in any of the following namespaces:

- qa
- test
- production
- alan



Answer:

See the solution below.

Explanation

Solution:

Create the Pod:

kubectl create

-f <http://k8s.io/docs/tasks/configure-pod-container/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
"gcr.io/google_containers/busybox"
23s 23s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image
"gcr.io/google_containers/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 "gcr.io/google_containers/busybox"

36s 36s 1 {kubelet worker0} spec.containers{liveness} Normal Pulled Successfully pulled image "gcr.io/google_containers/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	1 m

NEW QUESTION: 19

Refer to Exhibit.



Given a container that writes a log file in format A and a container that converts log files from format A to format B, create a deployment that runs both containers such that the log files from the first container are converted by the second container, emitting logs in format B.

Task:

- * Create a deployment named deployment-xyz in the default namespace, that:
- * Includes a primary lfcncf/busybox:1 container, named logger-dev
- * includes a sidecar lfcncf/fluentd:v0.12 container, named adapter-zen
- * Mounts a shared volume /tmp/log on both containers, which does not persist when the pod is deleted
- * Instructs the logger-dev container to run the command

```
while true; do
  echo "i luv cncf" >> /
  tmp/log/input.log;
  sleep 10;
done
```

which should output logs to /tmp/log/input.log in plain text format, with example values:

```
i luv cncf
i luv cncf
i luv cncf
```

- * The adapter-zen sidecar container should read /tmp/log/input.log and output the data to /tmp/log/output.* in Fluentd JSON format. Note that no knowledge of Fluentd is required to complete this task: all you will need to achieve this is to create the ConfigMap from the spec file provided at /opt/KDMC00102/fluentd-configmap.yaml, and mount that ConfigMap to /fluentd/etc in the adapter-zen sidecar container

Answer:

Solution:



```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: deployment-xyz
  name: deployment-xyz
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deployment-xyz
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: deployment-xyz
    spec:
      containers:
      - image: lfccncf/busybox:1
        name: busybox
        resources: {}
status: {}
~
~
"deployment_xyz.yml" 24L, 434C
```



All

```
kind: Deployment
metadata:
  labels:
    app: deployment-xyz
  name: deployment-xyz
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deployment-xyz
  template:
    metadata:
      labels:
        app: deployment-xyz
    spec:
      volumes:
      - name: myvol1
        emptyDir: {}
      containers:
      - image: lfccncf/busybox:1
        name: logger-dev
        volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
      - image: lfccncf/fluentd:v0.12
        name: adapter-zen
3 lines yanked
```



27,22

Bot

```
replicas: 1
selector:
  matchLabels:
    app: deployment-xyz
template:
  metadata:
    labels:
      app: deployment-xyz
  spec:
    volumes:
      - name: myvol1
        emptyDir: {}
    containers:
      - image: lfccncf/busybox:1
        name: logger-dev
        command: ["/bin/sh", "-c", "while [ true ]; do echo 'i luv cnf' >> /tmp/log/input.log; sleep 10; done"]
        volumeMounts:
          - name: myvol1
            mountPath: /tmp/log
      - image: lfccncf/fluentd:v0.12
        name: adapter-zen
        command: ["/bin/sh", "-c", "tail -f /tmp/log/input.log >> /tmp/log/output.log"]
        volumeMounts:
          - name: myvol1
            mountPath: /tmp/log
```

29,83

Bot

```
metadata:
  labels:
    app: deployment-xyz
spec:
  volumes:
    - name: myvol1
      emptyDir: {}
    - name: myvol2
      configMap:
        name: logconf
  containers:
    - image: lfccncf/busybox:1
      name: logger-dev
      command: ["/bin/sh", "-c", "while [ true ]; do echo 'i luv cnf' >> /tmp/log/input.log; sleep 10; done"]
      volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
    - image: lfccncf/fluentd:v0.12
      name: adapter-zen
      command: ["/bin/sh", "-c", "tail -f /tmp/log/input.log >> /tmp/log/output.log"]
      volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
        - name: myvol2
          mountPath: /fluentd/etc
```

37,33

Bot

```
student@node-1:~$ kubectl create -f deployment_xyz.yml
deployment.apps/deployment-xyz created
student@node-1:~$ kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz 0/1      1             0           5s
student@node-1:~$ kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz 0/1      1             0           9s
student@node-1:~$ kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz 1/1      1             1          12s
student@node-1:~$
```

NEW QUESTION: 20



Context

Developers occasionally need to submit pods that run periodically.

Task

Follow the steps below to create a pod that will start at a predetermined time and which runs to completion only once each time it is started:

- * Create a YAML formatted Kubernetes manifest /opt/KDPD00301/periodic.yaml that runs the following shell command: date in a single busybox container. The command should run every minute and must complete within 22 seconds or be terminated by Kubernetes. The Cronjob name and container name should both be hello
- * Create the resource in the above manifest and verify that the job executes successfully at least once See the solution below.

Answer:

Explanation

Solution:

```
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate,go-template-file,json,jsonpath,jsonpath-as-json,jsonpath-file,name,template,templatefile
,yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
```

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  jobTemplate:
    metadata:
      name: hello
    spec:
      template:
        spec:
          containers:
            - image: busybox
              name: hello
              args: ["/bin/sh", "-c", "echo hello"]
          restartPolicy: Never
  schedule: '*/* * * * *'
  startingDeadlineSeconds: 60
  concurrencyPolicy: Allow
```

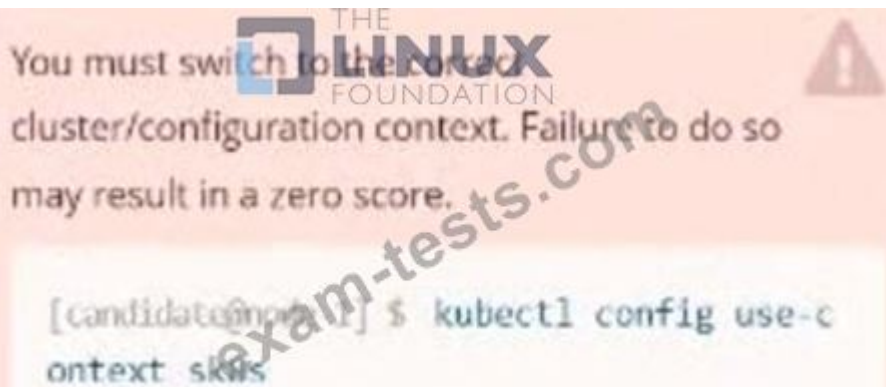
19,26

All

```
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
error: unable to match a printer suitable for the output format "yaml", allowed formats are: go-t
emplate,go-template-file,json,jsonpath,jsonpath-as-json,jsonpath-file,name,template,templatefile
,yaml
student@node-1:~$ kubectl create cronjob hello --image=busybox --schedule "* * * * *" --dry-run=
client -o yaml > /opt/KDPD00301/periodic.yaml
student@node-1:~$ vim /opt/KDPD00301/periodic.yaml
student@node-1:~$ kubectl create -f /opt/KDPD00301/periodic.yaml
cronjob.batch/hello created
student@node-1:~$ kubectl get cronjob
NAME      SCHEDULE      SUSPEND  NEXT FIRE  LAST SCHEDULE  AGE
hello     */1 * * * *   False    0          <none>         6s
student@node-1:~$
```

NEW QUESTION: 21

Context



Task:

Create a Deployment named `expose` in the existing `ckad00014` namespace running 6 replicas of a Pod. Specify a single container using the `ifccncf/nginx: 1.13.7` image Add an environment variable named `NGINX_PORT` with the value `8001` to the container then expose port `8001`

Answer:

Solution:

[illegible]

File Edit View Terminal Tabs Help

```

piVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: expose
  name: expose
  namespace: ckad00014
spec:
  replicas: 6
  selector:
    matchLabels:
      app: expose
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: expose
    spec:
      containers:
        - image: lfccncf/nginx:1.13.7
          name: nginx
          ports:
            - containerPort: 8001
          env:
            - name: NGINX_PORT
              value: "8001"

```

```

File Edit View Terminal Tabs Help
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl create deploy expose -n ckad00014 --image lfccncf/nginx:1.13.7 --dry-run=client -o yaml > dep.yaml
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$ vim dep.yaml
candidate@node-1:~$ kubectl create -f dep.yaml
deployment.apps/expose created
candidate@node-1:~$ kubectl get pods -n ckad00014
NAME                                READY    STATUS    RESTARTS   AGE
expose-85dd99d4d9-25675             0/1      ContainerCreating    0           6s
expose-85dd99d4d9-4fhcc             0/1      ContainerCreating    0           6s
expose-85dd99d4d9-fl7d7j           0/1      ContainerCreating    0           6s
expose-85dd99d4d9-tt6rm            0/1      ContainerCreating    0           6s
expose-85dd99d4d9-vjd8b            0/1      ContainerCreating    0           6s
expose-85dd99d4d9-vtzpq            0/1      ContainerCreating    0           6s
candidate@node-1:~$ kubectl get deploy -n ckad00014
NAME    READY    UP-TO-DATE    AVAILABLE    AGE
expose  6/6      6              6            15s
candidate@node-1:~$

```

NEW QUESTION: 22

Exhibit:



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

A. 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

```

Readme
Web Terminal
THE LINUX FOUNDATION

```

apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-secret
  name: nginx-secret
spec:
  containers:
    - image: nginx
      name: nginx-secret
  resources:
    requests:
      cpu: 100m
  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
cmicha  1/1     Running   0           7s
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.yaml
student@node-1:~$ vim nginx_secret.yaml
student@node-1:~$ kubectl create -f nginx_secret.yaml
pod/nginx-secret created
student@node-1:~$ kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
liveness-http  1/1     Running   0           4h38m
nginx-101     1/1     Running   0           4h39m
nginx-secret  1/1     Running   0           8s
poller       1/1     Running   0           4h39m
student@node-1:~$
```

B. 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.yaml
student@node-1:~$ vim nginx_secret.yaml
```

```
Readme Web Terminal THE LINUX FOUNDATION

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.yaml" 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
```

```
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.yaml
student@node-1:~$ vim nginx_secret.yaml
student@node-1:~$ kubectl create -f nginx_secret.yaml
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:~$
```

Answer: B ([LEAVE A REPLY](#))

NEW QUESTION: 23

Exhibit:

Set configuration context:

```
[student@node-1]# kubectl config  
use-context nk8s
```

Task

You have rolled out a new pod to your infrastructure and now you need to allow it to communicate with the web and storage pods but nothing else. Given the running pod kdsn00201 -newpod edit it to use a network policy that will allow it to send and receive traffic only to and from the web and storage pods.

All work on this item should be conducted in the kdsn00201 namespace.

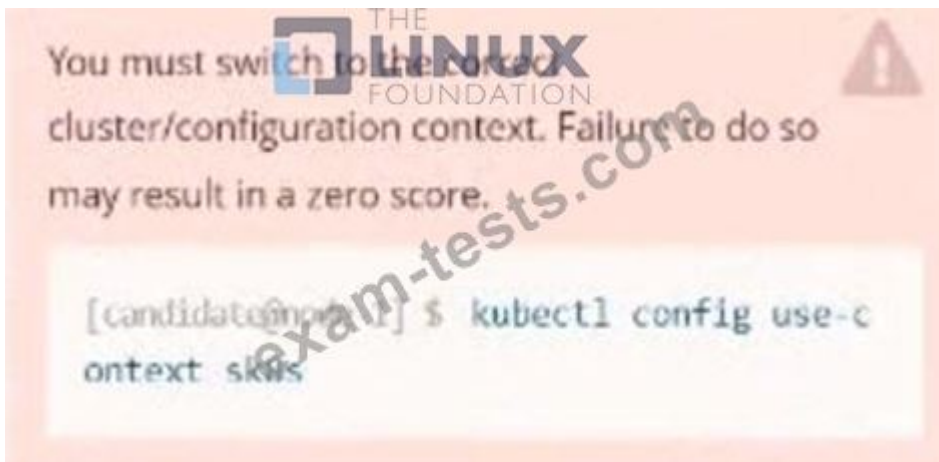
All required NetworkPolicy resources are already created and ready for use as appropriate. You should not create, modify or delete any network policies whilst completing this item.

A. Pending

Answer: ([SHOW ANSWER](#))

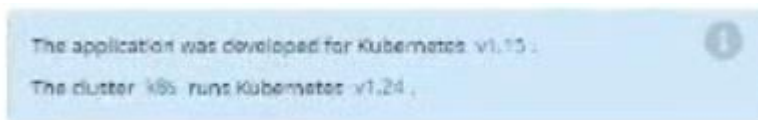
NEW QUESTION: 24

Context



Task:

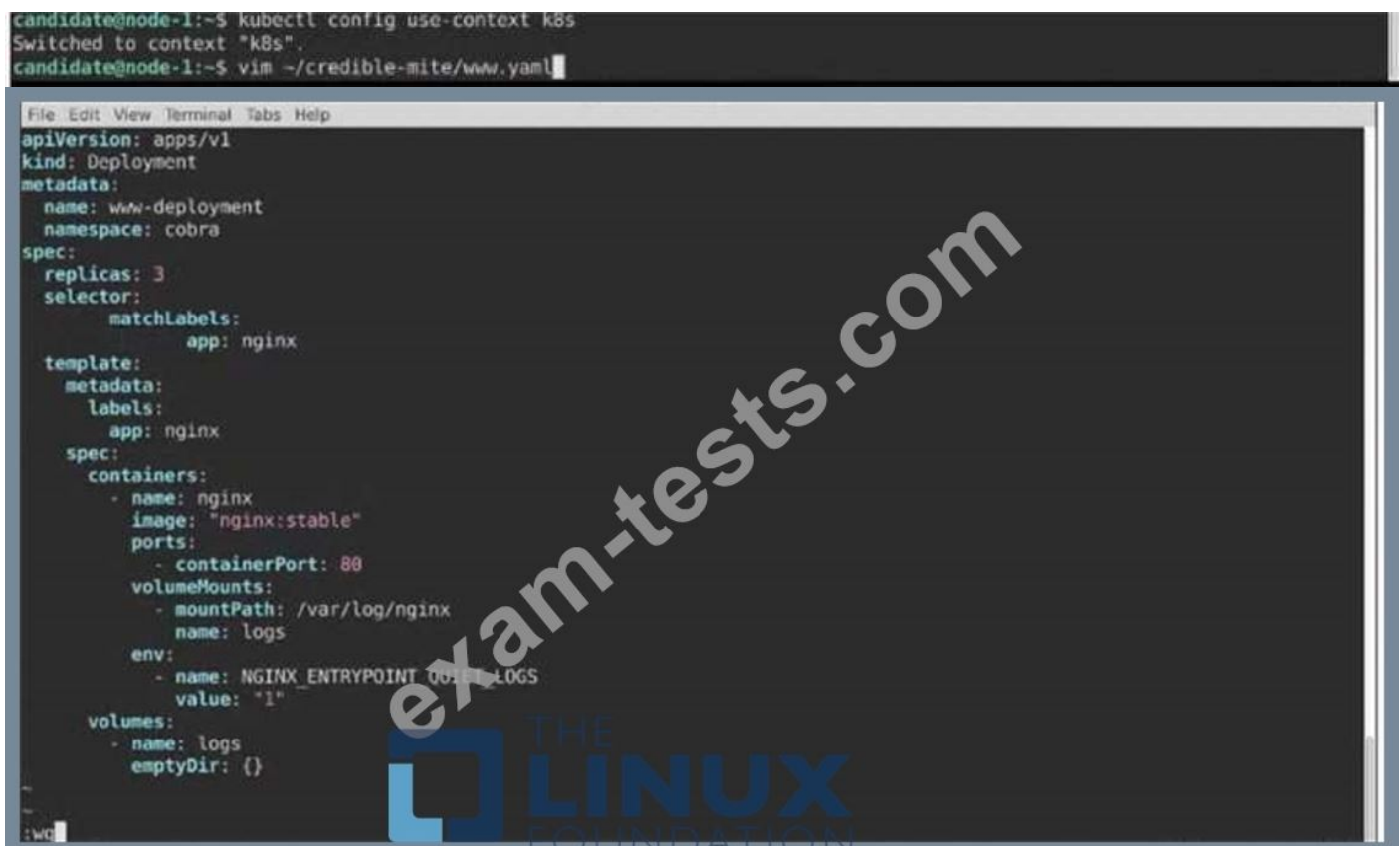
1) Fix any API depreciation issues in the manifest file `-/credible-mite/www.yaml` so that this application can be deployed on cluster K8s.



2) Deploy the application specified in the updated manifest file `-/credible-mite/www.yaml` in namespace cobra

Answer:

Solution:



```
File Edit View Terminal Tabs Help
deployment.apps/expose created
candidate@node-1:~$ kubectl get pods -n ckad00014
NAME                                READY   STATUS    RESTARTS   AGE
expose-85dd99d4d9-2567z             0/1     ContainerCreating   0           6s
expose-85dd99d4d9-4fhcc             0/1     ContainerCreating   0           6s
expose-85dd99d4d9-flid7j            0/1     ContainerCreating   0           6s
expose-85dd99d4d9-tt6rm             0/1     ContainerCreating   0           6s
expose-85dd99d4d9-vjd8b            0/1     ContainerCreating   0           6s
expose-85dd99d4d9-vtzpq            0/1     ContainerCreating   0           6s
candidate@node-1:~$ kubectl get deploy -n ckad00014
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
expose  6/6     6            6           15s
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/credible-mite/www.yaml
candidate@node-1:~$ vim ~/credible-mite/www.yaml
candidate@node-1:~$ kubectl apply -f ~/credible-mite/www.yaml
deployment.apps/www-deployment created
candidate@node-1:~$ kubectl get pods -n cobra
NAME                                READY   STATUS    RESTARTS   AGE
www-deployment-d899c6b49-d6ccg      1/1     Running   0           6s
www-deployment-d899c6b49-f796l      0/1     ContainerCreating   0           6s
www-deployment-d899c6b49-ztfcw      0/1     ContainerCreating   0           6s
candidate@node-1:~$ kubectl get deploy -n cobra
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
www-deployment  3/3     3            3           11s
candidate@node-1:~$ kubectl get pods -n cobra
NAME                                READY   STATUS    RESTARTS   AGE
www-deployment-d899c6b49-d6ccg      1/1     Running   0           14s
www-deployment-d899c6b49-f796l      1/1     Running   0           14s
www-deployment-d899c6b49-ztfcw      1/1     Running   0           14s
candidate@node-1:~$
```

NEW QUESTION: 25

Exhibit:



Given a container that writes a log file in format A and a container that converts log files from format A to format B, create a deployment that runs both containers such that the log files from the first container are converted by the second container, emitting logs in format B.

Task:

- * Create a deployment named deployment-xyz in the default namespace, that:

- * Includes a primary

lfccncf/busybox:1 container, named logger-dev

- * includes a sidecar lfccncf/fluend:v0.12 container, named adapter-zen

- * Mounts a shared volume /tmp/log on both containers, which does not persist when the pod is deleted

- * Instructs the logger-dev

container to run the command

```
while true; do
echo "i luv cncf" >> /
tmp/log/input.log;
sleep 10;
done
```



which should output logs to /tmp/log/input.log in plain text format, with example values:

```
i luv cncf
i luv cncf
i luv cncf
```

* The adapter-zen sidecar container should read /tmp/log/input.log and output the data to /tmp/log/output.* in Fluentd JSON format. Note that no knowledge of Fluentd is required to complete this task: all you will need to achieve this is to create the ConfigMap from the spec file provided at /opt/KDMC00102/fluentd-configmap.p.yaml, and mount that ConfigMap to /fluentd/etc in the adapter-zen sidecar container

A. Solution:

A screenshot of a web terminal interface. At the top, there are two tabs: 'Readme' and 'Web Terminal'. The 'Web Terminal' tab is active. The terminal shows a prompt 'student@node-1:~\$' followed by the command 'kubectl create deployment deployment-xyz --image=lfcncf/busybox:1 --dry-run=c'. The output is 'lient -o yaml > deployment_xyz.yml'. The prompt then shows 'student@node-1:~\$ vim deployment_xyz.yml'. The terminal has a dark background with light text. The Linux Foundation logo is visible in the top right corner of the terminal window.

```
student@node-1:~$ kubectl create deployment deployment-xyz --image=lfcncf/busybox:1 --dry-run=c
lient -o yaml > deployment_xyz.yml
student@node-1:~$ vim deployment_xyz.yml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: deployment-xyz
  name: deployment-xyz
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deployment-xyz
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: deployment-xyz
    spec:
      containers:
      - image: lfcncf/busybox:1
        name: busybox
        resources: {}
status: {}
```

"deployment_xyz.yml" 24L, 434C

3,1

All

```
kind: Deployment
metadata:
  labels:
    app: deployment-xyz
  name: deployment-xyz
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deployment-xyz
  template:
    metadata:
      labels:
        app: deployment-xyz
    spec:
      volumes:
      - name: myvol1
        emptyDir: {}
      containers:
      - image: lfcncf/busybox:1
        name: logger-dev
        volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
      - image: lfcncf/fluentd:v0.12
        name: adapter-zen
```

3 lines yanked

27,22

Bot

```
replicas: 1
selector:
  matchLabels:
    app: deployment-xyz
template:
  metadata:
    labels:
      app: deployment-xyz
  spec:
    volumes:
      - name: myvol1
        emptyDir: {}
    containers:
      - image: lfccncf/busybox:1
        name: logger-dev
        command: ["/bin/sh", "-c", "while [ true ]; do echo 'i luv cncf' >> /tmp/log/input.log; sleep 10; done"]
        volumeMounts:
          - name: myvol1
            mountPath: /tmp/log
      - image: lfccncf/fluentd:v0.12
        name: adapter-zen
        command: ["/bin/sh", "-c", "tail -f /tmp/log/input.log >> /tmp/log/output.log"]
        volumeMounts:
          - name: myvol1
            mountPath: /tmp/log
```

29,83

Bot

```
metadata:
  labels:
    app: deployment-xyz
spec:
  volumes:
    - name: myvol1
      emptyDir: {}
    - name: myvol2
      configMap:
        name: logconf
  containers:
    - image: lfccncf/busybox:1
      name: logger-dev
      command: ["/bin/sh", "-c", "while [ true ]; do echo 'i luv cncf' >> /tmp/log/input.log; sleep 10; done"]
      volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
    - image: lfccncf/fluentd:v0.12
      name: adapter-zen
      command: ["/bin/sh", "-c", "tail -f /tmp/log/input.log >> /tmp/log/output.log"]
      volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
        - name: myvol2
          mountPath: /fluentd/etc
```

37,33

Bot

```

student@node-1:~$ kubectl create -f deployment xyz.yml
deployment.apps/deployment-xyz created
student@node-1:~$ kubectl get deployment
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
deployment-xyz      0/1      1              0            5s
student@node-1:~$ kubectl get deployment
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
deployment-xyz      0/1      1              0            9s
student@node-1:~$ kubectl get deployment
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
deployment-xyz      1/1      1              1            12s
student@node-1:~$

```

B. Solution:

Readme Web Terminal

THE LINUX FOUNDATION

```

student@node-1:~$ kubectl create deployment deployment-xyz --image=lfcncf/busybox:1 --dry-run=c
lient -o yaml > deployment_xyz.yml
student@node-1:~$ vim deployment_xyz.yml

```

Readme Web Terminal

THE LINUX FOUNDATION

```

apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: deployment-xyz
  name: deployment-xyz
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deployment-xyz
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: deployment-xyz
    spec:
      containers:
      - image: lfcncf/busybox:1
        name: busybox
        resources: {}
status: {}
~
~
"deployment_xyz.yml" 24L, 434C

```

3,1

All

```
kind: Deployment
metadata:
  labels:
    app: deployment-xyz
    name: deployment-xyz
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deployment-xyz
  template:
    metadata:
      labels:
        app: deployment-xyz
    spec:
      volumes:
        - name: myvol1
          emptyDir: {}
      containers:
        - image: lfccncf/busybox:1
          name: logger-dev
          volumeMounts:
            - name: myvol1
              mountPath: /tmp/log
        - image: lfccncf/fluentd:v0.12
          name: adapter-zen
3 lines yanked
```

27,22

Bot

```
metadata:
  labels:
    app: deployment-xyz
spec:
  volumes:
    - name: myvol1
      emptyDir: {}
    - name: myvol2
      configMap:
        name: logconf
  containers:
    - image: lfccncf/busybox:1
      name: logger-dev
      command: ["/bin/sh", "-c", "while true; do echo 'i luv cncf' >> /tmp/log/input.log; sleep 10; done"]
      volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
    - image: lfccncf/fluentd:v0.12
      name: adapter-zen
      command: ["/bin/sh", "-c", "tail -f /tmp/log/input.log >> /tmp/log/output.log"]
      volumeMounts:
        - name: myvol1
          mountPath: /tmp/log
        - name: myvol2
          mountPath: /fluentd/etc
```

37,33

Bot

```
student@node-1:~$ kubectl create -f deployment xyz.yml
deployment.apps/deployment-xyz created
student@node-1:~$ kubectl get deployment
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz       0/1     1             0           5s
student@node-1:~$ kubectl get deployment
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz       0/1     1             0           9s
student@node-1:~$ kubectl get deployment
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
deployment-xyz       1/1     1             1          12s
student@node-1:~$
```

Answer: A ([LEAVE A REPLY](#))

NEW QUESTION: 26

Refer to Exhibit.



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  1       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
```

exam-tests.com

et

exam-tests.com

THE

ReadmeWeb 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
.yaml
student@node-1:~$ vim nginx_secret.yaml
student@node-1:~$ kubectl create -f nginx_secret.yaml
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:~$
```

NEW QUESTION: 27

Exhibit:



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

A. 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_configmap.yaml
student@node-1:~$ vim nginx_configmap.yaml ^C
student@node-1:~$ mv nginx_configmap.yaml nginx_configmap.yaml
student@node-1:~$ vim nginx_co
```

Readme

Web Terminal

THE LINUX FOUNDATION

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

"nginx_configmap.yaml" 15L, 262C
```

1,1

All

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-configmap
    name: nginx-configmap
spec:
  containers:
  - image: nginx
    name: nginx-configmap
    volumeMounts:
    - name: myvol
      mountPath: /also/a/path
  volumes:
  - name: myvol
    configMap:
      name: another-config
```

13,6

All

```
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
```



```

apiVersion: v1
kind: Pod
metadata:
  labels:
    run: nginx-configmap
    name: nginx-configmap
spec:
  containers:
  - image: nginx
    name: nginx-configmap
    volumeMounts:
    - name: myvol
      mountPath: /also/a/path
  volumes:
  - name: myvol
    configMap:
      name: another-config

```

13,6

All

```

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
student@node-1:~$ mv nginx_configmap.yaml nginx_configmap.yaml
student@node-1:~$ vim nginx_configmap.yaml
student@node-1:~$

```

```

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:~$

```

Answer: ([SHOW ANSWER](#))

NEW QUESTION: 28

Refer to Exhibit.



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

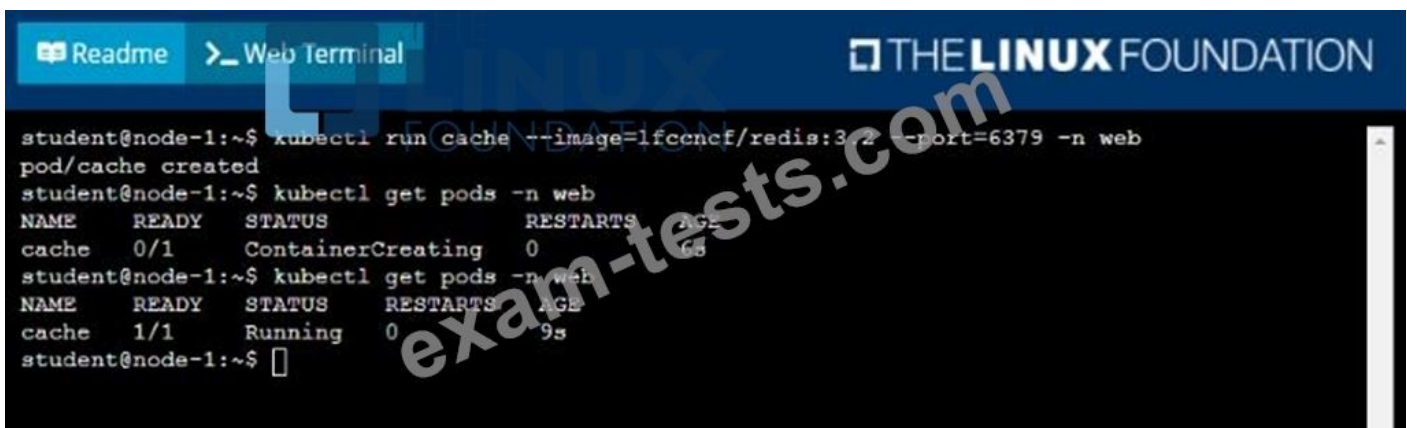
- * The name of the pod should be cache

- * Use the lfcncf/redis image with the 3.2 tag

- * Expose port 6379

Answer:

Solution:



NEW QUESTION: 29

Refer to Exhibit.



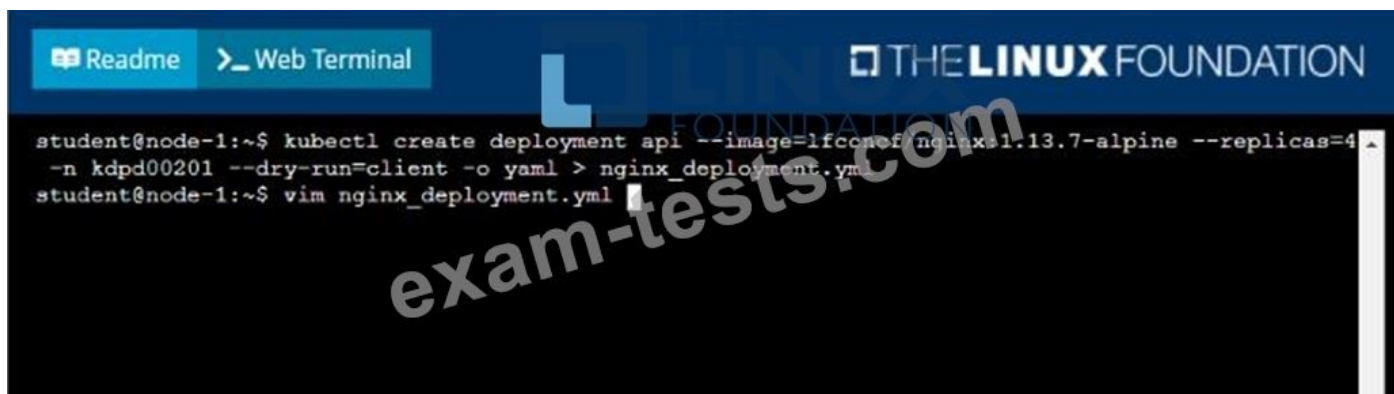
Task

Create a new deployment for running nginx with the following parameters;

- * Run the deployment in the kdpd00201 namespace. The namespace has already been created
- * Name the deployment frontend and configure with 4 replicas
- * Configure the pod with a container image of lfcncf/nginx:1.13.7
- * Set an environment variable of NGINX__PORT=8080 and also expose that port for the container above

Answer:

Solution:



Readme Web Terminal

THE LINUX FOUNDATION

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: api
  name: api
  namespace: kdpd00201
spec:
  replicas: 4
  selector:
    matchLabels:
      app: api
  template:
    metadata:
      labels:
        app: api
    spec:
      containers:
        - image: lfccncf/nginx:1.13.7-alpine
          name: nginx
          ports:
            - containerPort: 8080
          env:
            - name: NGINX_PORT
              value: "8080"
```

23,8 All

Readme Web Terminal

THE LINUX FOUNDATION

```
student@node-1:~$ kubectl create deployment api --image=lfccncf/nginx:1.13.7-alpine --replicas=4
-k n kdpd00201 --dry-run=client -o yaml > nginx_deployment.yml
student@node-1:~$ vim nginx_deployment.yml
student@node-1:~$ kubectl create nginx_deployment.yml
Error: must specify one of -f and -k

error: unknown command "nginx_deployment.yml"
See 'kubectl create -h' for help and examples
student@node-1:~$ kubectl create -f nginx_deployment.yml
error: error validating "nginx_deployment.yml": error validating data: ValidationError(Deployment.spec.template.spec): unknown field "env" in io.k8s.api.core.v1.PodSpec; if you choose to ignore these errors, turn validation off with --validate=false
student@node-1:~$ vim nginx_deployment.yml
student@node-1:~$ kubectl create -f nginx_deployment.yml
deployment.apps/api created
student@node-1:~$ kubectl get pods -n kdpd00201
```

NAME	READY	STATUS	RESTARTS	AGE
api-745677f7dc-7hnm	1/1	Running	0	13s
api-745677f7dc-9q5vp	1/1	Running	0	13s
api-745677f7dc-fd4gk	1/1	Running	0	13s
api-745677f7dc-mbnpc	1/1	Running	0	13s

```
student@node-1:~$
```

NEW QUESTION: 30

Exhibit:

Set configuration context:



```
[student@node-1] $ kubectl config  
use-context sk8s
```



THE
LINUX
FOUNDATION

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:

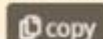
- * Create a file on node sk8s-node-0 at /opt/KDSP00101/data/index.html with the content Acct=Finance
- * Create a PersistentVolume named task-pv-volume using hostPath and allocate 1Gi to it, specifying that the volume is at /opt/KDSP00101/data on the cluster's node. The configuration should specify 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

You can access sk8s-node-0 by
issuing the following
command:

```
[student@node-1] $ ssh sk8  
s-node-0
```



Ensure that you return to the
base node (with hostname
node-1) once you have completed
your work on sk8s-node-0



A. Solution:

Readme Web Terminal



THE LINUX FOUNDATION

```
student@node-1:~$ kubectl config use-context sk8s
Switched to context "sk8s".
student@node-1:~$
```

Readme Web Terminal

THE LINUX FOUNDATION

```
* Documentation: https://help.ubuntu.com
* Management:   https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage
```

System information as of Fri Oct 9 08:52:09 UTC 2020

```
System load: 2.02           Users logged in:
Usage of /: 10.3% of 242.29GB IP address for eth0: 10.250.3.115
Memory usage: 2%           IP address for dock0: 172.17.0.1
Swap usage: 0%             IP address for ens0: 10.244.1.1
Processes: 38
```

* Kubernetes 1.19 is out! Get it in one command with:

```
sudo snap install microk8s --channel 1.19 --classic
```

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

7 packages can be updated.

1 update is a security update.

New release '20.04.1 LTS' available.

Run 'do-release-upgrade' to upgrade to it.

```
student@sk8s-node-0:~$
```

Readme Web Terminal

THE LINUX FOUNDATION

```
student@sk8s-node-0:~$ echo 'Acct=Finance' > /opt/XDSP00101/data/index.html
student@sk8s-node-0:~$ vim pv.yml
```

```
-- INSERT --
```

0,1

All

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  storageClassName: storage
  hostPath:
    path: /opt/KDSP00101/data
    type: Directory
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: task-pv-claim
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: storage
```

```

student@sk8s-node-0:~$ kubectl create -f pv.yml
persistentvolume/task-pv-volume created
student@sk8s-node-0:~$ kubectl create -f pvc.yml
persistentvolumeclaim/task-pv-claim created
student@sk8s-node-0:~$ kubectl get pv
NAME              CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   AGE
task-pv-volume    1Gi        RWO             Retain            Bound    default/task-pv-claim  storage        11s
student@sk8s-node-0:~$ kubectl get pvc
NAME              STATUS   VOLUME          CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim     Bound    task-pv-volume  1Gi        RWO             storage        9s
student@sk8s-node-0:~$ vim pod.yml

```

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-claim

```

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           6s
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:~$

```

B. Solution:

Readme Web Terminal THE LINUX FOUNDATION

```

student@node-1:~$ kubectl config use-context sk8s
Switched to context "sk8s".
student@node-1:~$

```

ReadmeWeb Terminal

THE LINUX FOUNDATION

```
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

System information as of Fri Oct 9 08:52:09 UTC 2020

System load: 2.02          Users logged in:
Usage of /: 10.3% of 242.29GB IP address for eth0: 10.250.3.115
Memory usage: 2%          IP address for dock-0: 172.17.0.1
Swap usage: 0%            IP address for cn0: 10.244.1.1
Processes: 38

* Kubernetes 1.19 is out! Get it in one command with:

  sudo snap install microk8s --channel 1.19 --classic

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

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

New release '20.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

student@sk8s-node-0:~$
```

ReadmeWeb Terminal

THE LINUX FOUNDATION

```
student@sk8s-node-0:~$ echo 'Acct=Finance' > /opt/XDSP00101/data/index.html
student@sk8s-node-0:~$ vim pv.yml
```

ReadmeWeb Terminal

THE LINUX FOUNDATION

```
-- INSERT --
```

0,1All

```

apiVersion: v1
kind: PersistentVolume
metadata:
  name: task-pv-volume
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteOnce
  storageClassName: storage
  hostPath:
    path: /opt/KDSP00101/data
    type: Directory

```



```

student@sk8s-node-0:~$ kubectl create -f pv.yml
persistentvolume/task-pv-volume created
student@sk8s-node-0:~$ kubectl create -f pvc.yml
persistentvolumeclaim/task-pv-claim created
student@sk8s-node-0:~$ kubectl get pv
NAME                CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                STORAGECLASS   AGE
task-pv-volume      1Gi        RWO            Retain           Bound    default/task-pv-claim storage         11s
student@sk8s-node-0:~$ kubectl get pvc
NAME                STATUS   VOLUME          CAPACITY   ACCESS MODES   STORAGECLASS   AGE
task-pv-claim       Bound    task-pv-volume  1Gi        RWO            storage         9s
student@sk8s-node-0:~$ vim pod.yml

```

```

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-claim

```



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 6s
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:~$
```

Answer: ([SHOW ANSWER](#))

NEW QUESTION: 31

Context

```
Set configuration context:
[student@node-1] kubectl config
use-context sk8s
```

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:

```
Readme Web Terminal THE LINUX FOUNDATION
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:~$
```

Valid CKAD Dumps shared by BraindumpsPass.com for Helping Passing CKAD Exam!
BraindumpsPass.com now offer the **newest CKAD exam dumps**, the BraindumpsPass.com CKAD exam **questions have been updated** and **answers have been corrected** get the **newest** BraindumpsPass.com CKAD dumps with Test Engine here:
<https://www.braindumpsPass.com/Linux-Foundation/CKAD-practice-exam-dumps.html> (193 Q&As Dumps, **40%OFF** Special Discount: **Exam-Tests**)

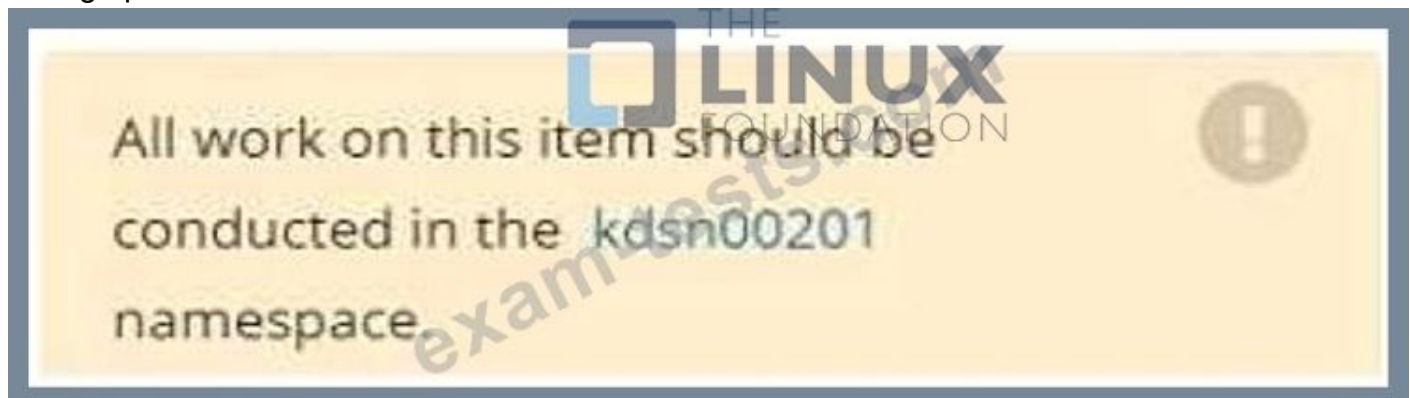
NEW QUESTION: 32

Context



Task

You have rolled out a new pod to your infrastructure and now you need to allow it to communicate with the web and storage pods but nothing else. Given the running pod kdsn00201 -newpod edit it to use a network policy that will allow it to send and receive traffic only to and from the web and storage pods.



All required NetworkPolicy resources are already created and ready for use as appropriate. You should not create, modify or delete any network policies whilst completing this item.

Answer:

apiVersion: networking.k8s.io/v1

kind: NetworkPolicy

metadata:

name: internal-policy

namespace: default

spec:

podSelector:

matchLabels:

name: internal

policyTypes:

- Egress

- Ingress

ingress:

- {}

egress:

- to:

- podSelector:

matchLabels:

name: mysql

ports:

- protocol: TCP

port: 3306

- to:

- podSelector:

matchLabels:

name: payroll

ports:

- protocol: TCP
port: 8080
- ports:
- port: 53
protocol: UDP
- port: 53
protocol: TCP

NEW QUESTION: 33



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:

See the solution below.

Explanation

Solution:



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

"nginx_resources.yml" 16L, 289C
```

1,1

All

```
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"
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}

-- INSERT --
```

15,22

All

```
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
```



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

NAME	READY	STATUS	RESTARTS	AGE
nginx-resources	1/1	Running	0	8s

Valid CKAD Dumps shared by BraindumpsPass.com for Helping Passing CKAD Exam!
BraindumpsPass.com now offer the **newest CKAD exam dumps**, the BraindumpsPass.com
CKAD exam **questions have been updated** and **answers have been corrected** get the
newest BraindumpsPass.com CKAD dumps with Test Engine here:
<https://www.braindumpsPASS.com/Linux-Foundation/CKAD-practice-exam-dumps.html> (193
Q&As Dumps, **40%OFF** Special Discount: **Exam-Tests**)