

LinuxFoundation.CKAD.v2023-04-13.q16

Exam Code:	CKAD
Exam Name:	Linux Foundation Certified Kubernetes Application Developer Exam
Certification Provider:	Linux Foundation
Free Question Number:	16
Version:	v2023-04-13
# of views:	833
# of Questions views:	160
https://www.freeqas.com/qa/Linux-Foundation/CKAD/LinuxFoundation.CKAD.v2023-04-13.q16.html	

NEW QUESTION: 1

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:



ReadmeWeb 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.yml" 15L, 253C1,1All

ReadmeWeb 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: key!
```

-- INSERT --16,20All

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.yml
student@node-1:~$ vim nginx_secret.yml
student@node-1:~$ kubectl create -f nginx_secret.yml
pod/nginx-secret created
student@node-1:~$ kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
liveness-http   1/1     Running   0           6h38m
nginx-101       1/1     Running   0           6h39m
nginx-secret    0/1     ContainerCreating   0           4s
poller          1/1     Running   0           6h39m
student@node-1:~$ kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
liveness-http   1/1     Running   0           6h38m
nginx-101       1/1     Running   0           6h39m
nginx-secret    1/1     Running   0           8s
poller          1/1     Running   0           6h39m
student@node-1:~$
```

NEW QUESTION: 2

Refer to 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

Answer:

Solution:

To deploy the counter pod to the cluster using the provided YAML spec file, you can use the kubectl apply command. The apply command creates and updates resources in a cluster.

```
kubectl apply -f /opt/KDOB00201/counter.yaml
```

This command will create the pod in the cluster. You can use the kubectl get pods command to check the status of the pod and ensure that it is running.

```
kubectl get pods
```

To retrieve all currently available application logs from the running pod and store them in the file /opt/KDOB00201/log_Output.txt, you can use the kubectl logs command.

The logs command retrieves logs from a container in a pod.

```
kubectl logs -f <pod-name> > /opt/KDOB00201/log_Output.txt
```

Replace <pod-name> with the name of the pod.

You can also use -f option to stream the logs.

```
kubectl logs -f <pod-name> > /opt/KDOB00201/log_Output.txt &
```

This command will retrieve the logs from the pod and write them to the /opt/KDOB00201/log_Output.txt file.

Please note that the above command will retrieve all logs from the pod, including previous logs. If you want to retrieve only the new logs that are generated after running the command, you can add the --since flag to the kubectl logs command and specify a duration, for example --since=24h for logs generated in the last 24 hours.

Also, please note that, if the pod has multiple containers, you need to specify the container name using -c option.

kubectl logs -f <pod-name> -c <container-name> > /opt/KDOB00201/log_Output.txt The above command will redirect the logs of the specified container to the file.

```
student@node-1:~$ kubectl create -f /opt/KDOB00201/counter.yaml
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: 40d3a9c8e46f5533bb4828fbc5c84038
5: 390442d2530a90c360290a7e959ac8
6: b71d95187417e139effb33af7681040
7: 66a8e55a6491e756d2d0540ad6ab90a7
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: 40d3a9c8e46f5533bb4828fbc5c8d038
5: 390442d2530a90c3602901e3fe999ac8
6: b71d95187417e139effb33af77681040
7: 66a8e55a6491e756d2d0549ad6ab90a7
8: ff2b3d583b64125d2f9129c443bb37ff
9: b6c6a12b6e77944ed8baaf6c242dae4
10: bfcc9a894a0604fc4b814b37d0a200a4
11: 5493cd16a1790a5fb9512b0c9d4c5dd1
12: 03f169e93e6143438e6dfe4ecb3cc9ed
13: 764b37fe611373c42d0b47154041f6eb
14: 1a56fbc1896b0ee6394136166281839e
15: ecc492eb17715de090c47345a98d98d3
16: 7974a6bec0fb44b6b8bbfc71aa3fbc74
17: 9ae01bef01748b12cc5f97a5f9f72cd6
18: 23fb22ee34d4272e4c9e005f1774515f
19: ec7e1a5d314da9a0ad45d53be5a7acae
20: 0bccdd8ee02cd42029e8162cd1c1197c
21: d6851ea43546216b95bcb81ced997102
22: 7ed9a38ea8bf0d86206569481442af44
23: 29b8416ddc63dbfcb987ab3c8198e9fe
24: 1f2062001df51a108ab2501ff506716f
student@node-1:~$
```

NEW QUESTION: 3

Refer to Exhibit.



Context

You have been tasked with scaling an existing deployment for availability, and creating a service to expose the deployment within your infrastructure.

Task

Start with the deployment named kdsn00101-deployment which has already been deployed to the namespace kdsn00101 . Edit it to:

- * Add the func=webFrontEnd key/value label to the pod template metadata to identify the pod for the service definition
- * Have 4 replicas

Next, create a deployment in namespace kdsn00101 a service that accomplishes the following:

- * Exposes the service on TCP port 8080
- * is mapped to the pods defined by the specification of kdsn00101-deployment
- * Is of type NodePort
- * Has a name of cherry

Answer:

Solution:

```
student@node-1:~$ kubectl edit deployment kdsn00101-deployment -n kdsn00101
```

THE LINUX FOUNDATION

Readme Web Terminal THE LINUX FOUNDATION

```
Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: "2020-10-09T08:50:39Z"
  generation: 1
  labels:
    app: nginx
  name: kdsn00101-deployment
  namespace: kdsn00101
  resourceVersion: "4744"
  selfLink: /apis/apps/v1/namespaces/kdsn00101/deployments/kdsn00101-deployment
  uid: 8d3ace00-7761-4189-ba10-fbc676c311bf
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
        func: webFrontEnd
    spec:
      containers:
      - image: nginx:latest
        imagePullPolicy: Always
        name: nginx
        ports:
        - containerPort: 80
```

1,1 Top

THE LINUX FOUNDATION

Readme Web Terminal THE LINUX FOUNDATION

```
uid: 8d3ace00-7761-4189-ba10-fbc676c311bf
spec:
  progressDeadlineSeconds: 600
  replicas: 4
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: nginx
        func: webFrontEnd
    spec:
      containers:
      - image: nginx:latest
        imagePullPolicy: Always
        name: nginx
        ports:
        - containerPort: 80
```

```
student@node-1:~$ kubectl edit deployment kdsn00101-deployment -n kdsn00101
deployment.apps/kdsn00101-deployment edited
student@node-1:~$ kubectl get deployment kdsn00101-deployment -n kdsn00101
NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
kdsn00101-deployment    4/4     4            4           7h17m
student@node-1:~$ kubectl expose deployment kdsn00101-deployment -n kdsn00101 --type NodePort --
port 8080 --name cherry
service/cherry exposed
```

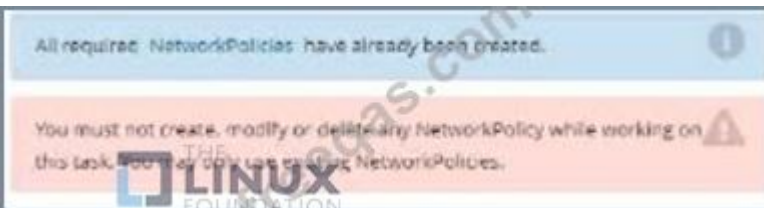
NEW QUESTION: 4

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
```

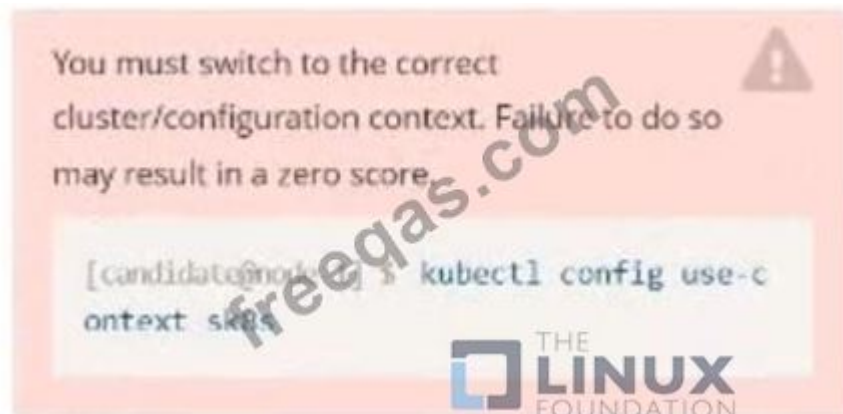
```

Edit View Terminal Tabs Help
me:      all-access
amespace: ckad00018
eated on: 2022-09-24 04:27:37 +0000 UTC
bels:    <none>
notations: <none>
ec:
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

me:      default-deny
amespace: ckad00018
eated on: 2022-09-24 04:27:37 +0000 UTC
bels:    <none>
notations: <none>
ec:
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
ndidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 web-access=true
d/ckad00018-newpod labeled
ndidate@node-1:~$ kubectl label pod ckad00018-newpod -n ckad00018 db-access=true
d/ckad00018-newpod labeled
ndidate@node-1:~$
```

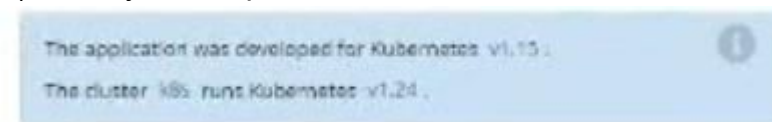
NEW QUESTION: 5

Refer to Exhibit.



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:

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

```
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: {}
```

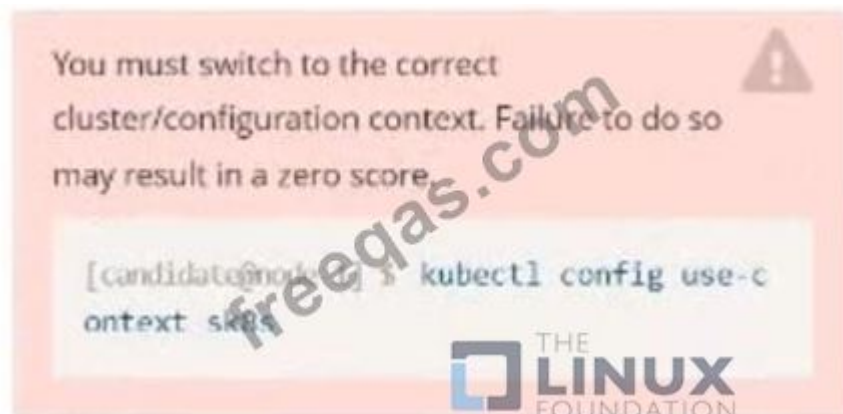
```

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:~$ 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: 6

Refer to Exhibit.



Task:

Create a Deployment named expose in the existing ckad00014 namespace running 6 replicas of a Pod.

Specify a single container using the lfcncf/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:

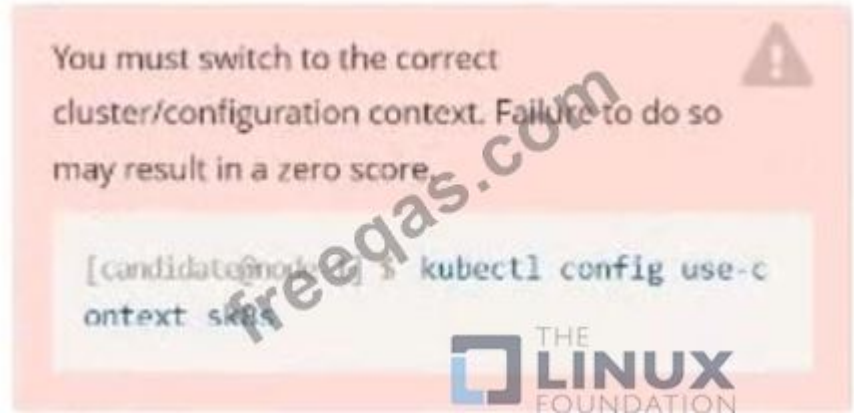
```
switched to context "k8s".
candidate@node-1:~$ kubectl create deploy expose -n ckad00014 --image lfcncf/nginx:1.13.7 --dry-run=client -o yaml > d
p.yaml
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
candidate@node-1:~$
```

```
File Edit View Terminal Tabs Help
apiVersion: 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: lfcncf/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:~$
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: 7

Refer to Exhibit.



Task:

1) First update the Deployment cka00017-deployment in the ckad00017 namespace:

*To run 2 replicas of the pod

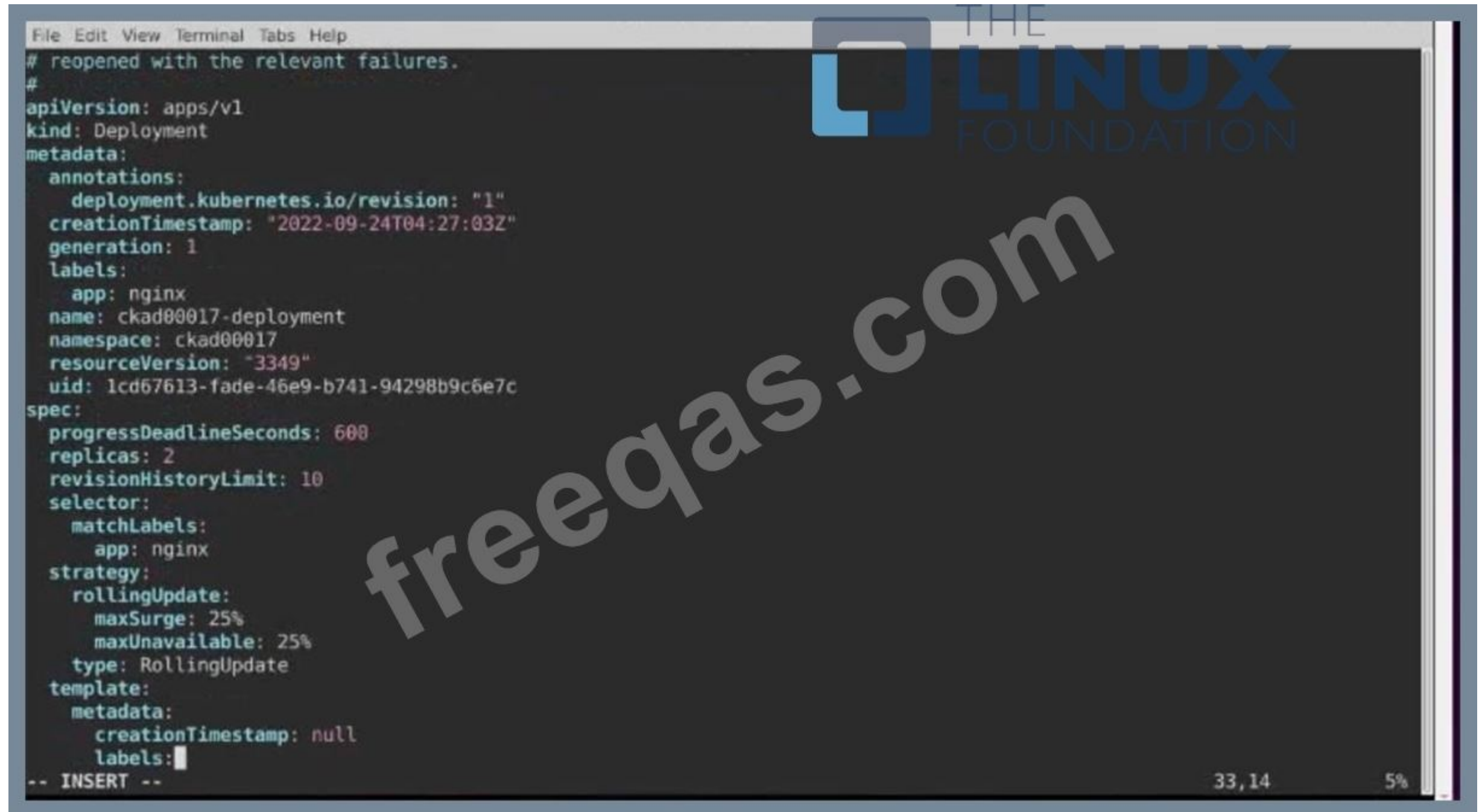
*Add the following label on the pod:

Role userUI

2) Next, Create a NodePort Service named cherry in the ckad00017 namespace exposing the ckad00017-deployment Deployment on TCP port 8888

Answer:

Solution:



```
File Edit View Terminal Tabs Help
# reopened with the relevant failures.
#
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: "2022-09-24T04:27:03Z"
  generation: 1
  labels:
    app: nginx
  name: ckad00017-deployment
  namespace: ckad00017
  resourceVersion: "3349"
  uid: lcd67613-fade-46e9-b741-94298b9c6e7c
spec:
  progressDeadlineSeconds: 600
  replicas: 2
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
-- INSERT --
```

33,14 5%

```
name: ckad00017-deployment
namespace: ckad00017
resourceVersion: "3349"
uid: 1cd67613-fade-46e9-b741-94298b9c6e7c
```

```
spec:
```

```
  progressDeadlineSeconds: 600
```

```
  replicas: 2
```

```
  revisionHistoryLimit: 10
```

```
  selector:
```

```
    matchLabels:
```

```
      app: nginx
```

```
  strategy:
```

```
    rollingUpdate:
```

```
      maxSurge: 25%
```

```
      maxUnavailable: 25%
```

```
    type: RollingUpdate
```

```
  template:
```

```
    metadata:
```

```
      creationTimestamp: null
```

```
      labels:
```

```
        app: nginx
```

```
        role: userUI
```

```
    spec:
```

```
      containers:
```

```
        - image: nginx:latest
```

```
          imagePullPolicy: Always
```

```
          name: nginx
```

```
          ports:
```

```
            - containerPort: 80
```

```
              protocol: TCP
```

```
          resources: {}
```

```
-- INSERT --
```

freeqas.com



File Edit View Terminal Tabs Help

```
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 deploy app-1 app -n frontend
deployment.apps/app-1 serviceaccount updated

candidate@node-1:~\$ kubectl config use-context k8s

Switched to context "k8s".

candidate@node-1:~\$ vim ~/prompt-escargot/buffalo-deployment.yaml

candidate@node-1:~\$ vim ~/prompt-escargot/buffalo-deployment.yaml

candidate@node-1:~\$ kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml

deployment.apps/buffalo-deployment configured

candidate@node-1:~\$ kubectl get pods -n gorilla

NAME	READY	STATUS	RESTARTS	AGE
buffalo-deployment-776844df7f-r5fsb	1/1	Running	0	6h38m
buffalo-deployment-859898c6f5-zx5gj	0/1	ContainerCreating	0	8s

candidate@node-1:~\$ kubectl get deploy -n gorilla

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
buffalo-deployment	1/1	1	1	6h38m

candidate@node-1:~\$ kubectl config use-context k8s

Switched to context "k8s".

candidate@node-1:~\$ kubectl edit deploy ckad00017-deployment -n ckad00017

deployment.apps/ckad00017-deployment edited

candidate@node-1:~\$



THE
LINUX
FOUNDATION

```

candidate@node-1:~$ kubectl get pods -n gorilla
NAME                                READY   STATUS    RESTARTS   AGE
buffalo-deployment-776844df7f-r5fsb 1/1     Running   0           6h38m
buffalo-deployment-859898c6f5-zx5gj 0/1     ContainerCreating 0           8s
candidate@node-1:~$ kubectl get deploy -n gorilla
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
buffalo-deployment 1/1     1             1           6h38m
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl edit deploy ckad00017-deployment -n ckad00017
deployment.apps/ckad00017-deployment edited
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad0001
ckad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
service/cherry exposed
candidate@node-1:~$

```

```

candidate@node-1:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP    77d
candidate@node-1:~$ kubectl get svc -n ckad00017
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
cherry       NodePort    10.100.100.176 <none>        8888:30683/TCP 24s
candidate@node-1:~$ kubectl expose service deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
Error from server (NotFound): services "deploy" not found
Error from server (NotFound): services "ckad00017-deployment" not found
candidate@node-1:~$ kubectl get svc -n ckad00017
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
cherry       NodePort    10.100.100.176 <none>        8888:30683/TCP 46s
candidate@node-1:~$

```

```

File Edit View Terminal Tabs Help
candidate@node-1:~$ kubectl expose service deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
Error from server (NotFound): services "deploy" not found
Error from server (NotFound): services "ckad00017-deployment" not found
candidate@node-1:~$ kubectl get svc -n ckad00017
NAME      TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
cherry    NodePort    10.100.100.176   <none>           8888:30683/TCP   46s
candidate@node-1:~$ history
 1 vi ~/spicy-pikachu/backend-deployment.yaml
 2 kubectl config use-context sk8s
 3 vim .vimrc
 4 vim ~/spicy-pikachu/backend-deployment.yaml
 5 kubectl apply -f ~/spicy-pikachu/backend-deployment.yaml
 6 kubectl get pods -n staging
 7 kubectl get deploy -n staging
 8 vim ~/spicy-pikachu/backend-deployment.yaml
 9 kubectl config use-context k8s
10 kubectl set serviceaccount deploy app-1 app -n frontend
11 kubectl config use-context k8s
12 vim ~/prompt-escargot/buffalo-deployment.yaml
13 kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml
14 kubectl get pods -n gorilla
15 kubectl get deploy -n gorilla
16 kubectl config use-context k8s
17 kubectl edit deploy ckad00017-deployment -n ckad00017
18 kubectl expose deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
19 kubectl get svc
20 kubectl get svc -n ckad00017
21 kubectl expose service deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
22 kubectl get svc -n ckad00017
23 history
candidate@node-1:~$

```

NEW QUESTION: 8

Refer to Exhibit.



Set Configuration Context:

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

Config use-context k8s

Context

A user has reported an aopticaun 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>



```
<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.



Answer:

To find the broken pod and store its name and namespace to /opt/KDOB00401/broken.txt, you can use the kubectl get pods command and filter the output by the status of the pod.

kubectl get pods --field-selector=status.phase=Failed -o jsonpath='{.items[*].metadata.namespace}/{.items[*].metadata.name}' > /opt/KDOB00401/broken.txt This command will list all pods with a status of Failed and output their names and namespaces in the format <namespace>/<pod>. The output is then written to the /opt/KDOB00401/broken.txt file.

To store the associated error events to a file /opt/KDOB00401/error.txt, you can use the kubectl describe command to retrieve detailed information about the pod, and the grep command to filter the output for error events.

kubectl describe pods <pod-name> --namespace <pod-namespace> | grep -i error -B5 -A5 > /opt/KDOB00401/error.txt Replace <pod-name> and <pod-namespace> with the name and namespace of the broken pod you found in the previous step.

This command will output detailed information about the pod, including error events. The grep command filters the output for lines containing "error" and also prints 5 lines before and after the match.

To fix the issue, you need to analyze the error events and find the root cause of the issue.

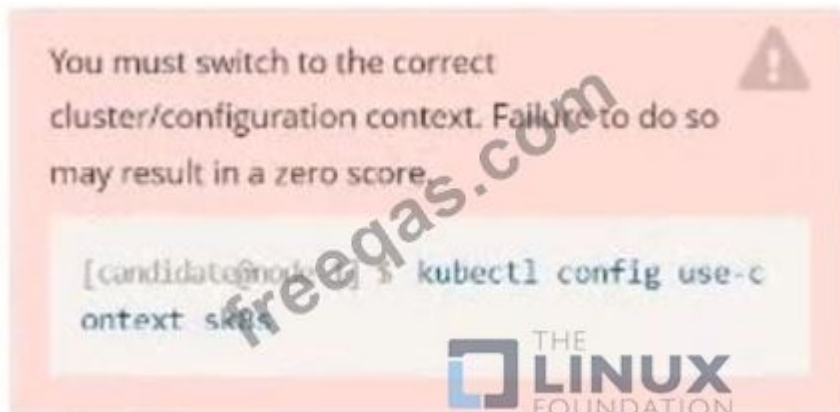
It could be that the application inside the pod is not running, the container image is not available, the pod has not enough resources, or the liveness probe configuration is incorrect.

Once you have identified the cause, you can take appropriate action, such as restarting the application, updating the container image, increasing the resources, or modifying the liveness probe configuration.

After fixing the issue, you can use the kubectl get pods command to check the status of the pod and ensure

NEW QUESTION: 9

Refer to Exhibit.



Task:

1) Create a secret named app-secret in the default namespace containing the following single key-value pair:

Key3: value1

2) Create a Pod named nginx-secret in the default namespace. Specify a single container using the nginx:stable image.

Add an environment variable named BEST_VARIABLE consuming the value of the secret key3.

Answer:

Solution:

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl create secret generic app-secret -n default --from-literal=key3=value1
secret/app-secret created
candidate@node-1:~$ kubectl get secrets
NAME          TYPE      DATA   AGE
app-secret    Opaque    1        4s
candidate@node-1:~$ kubectl run nginx-secret -n default --image=nginx:stable --dry-run=client -o yaml > sec.yaml
candidate@node-1:~$ vim sec.yaml
```

```
File Edit View Terminal Tabs Help
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx-secret
  name: nginx-secret
  namespace: default
spec:
  containers:
  - image: nginx:stable
    name: nginx-secret
    env:
    - name: BEST_VARIABLE
      valueFrom:
        secretKeyRef:
          name: app-secret
          key: key3
```

```
Switched to context "k8s".
candidate@node-1:~$ kubectl create secret generic app-secret -n default --from-literal=key3=value1
secret/app-secret created
candidate@node-1:~$ kubectl get secrets
NAME          TYPE      DATA   AGE
app-secret    Opaque    1        4s
candidate@node-1:~$ kubectl run nginx-secret -n default --image=nginx:stable --dry-run=client -o yaml > sec.yaml
candidate@node-1:~$ vim sec.yaml
candidate@node-1:~$ kubectl create -f sec.yaml
pod/nginx-secret created
candidate@node-1:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-secret  1/1     Running   0           7s
candidate@node-1:~$
```

NEW QUESTION: 10

Refer to 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

Answer:

Solution:



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: 20
  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:~$
```

NEW QUESTION: 11

Refer to Exhibit.

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 s1
```

THE LINUX FOUNDATION

Task:

The pod for the Deployment named nosql in the craytish namespace fails to start because its container runs out of resources.

Update the nosql Deployment so that the Pod:

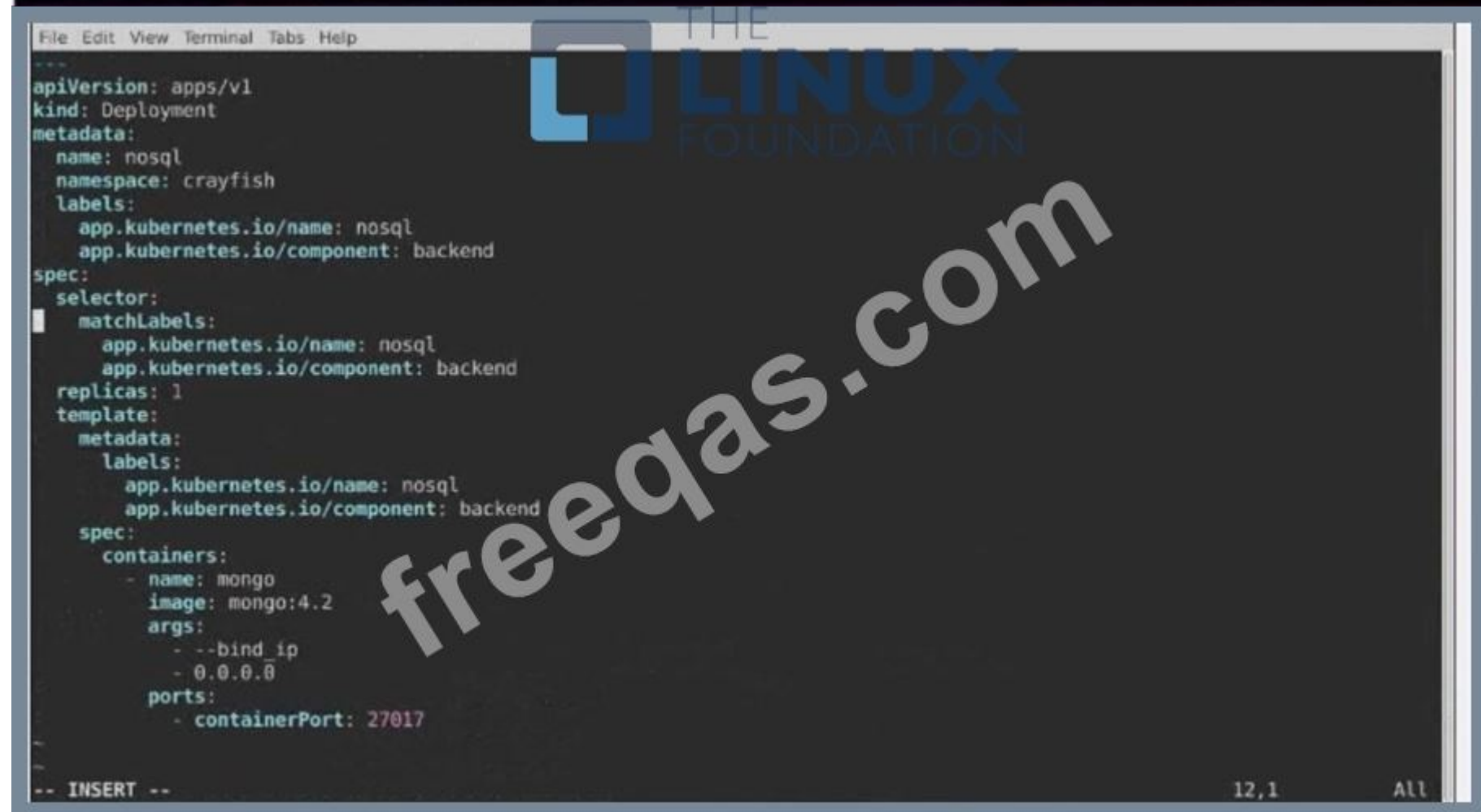
- 1) Request 160M of memory for its Container
- 2) Limits the memory to half the maximum memory constraint set for the crayfah name space.

★ The nosql Deployment's manifest file can be found at
~/chief-cardinal/nosql.yaml.

Answer:

Solution:

```
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/chief-cardinal/nosql.yaml
```



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nosql
  namespace: crayfish
  labels:
    app.kubernetes.io/name: nosql
    app.kubernetes.io/component: backend
spec:
  selector:
    matchLabels:
      app.kubernetes.io/name: nosql
      app.kubernetes.io/component: backend
  replicas: 1
  template:
    metadata:
      labels:
        app.kubernetes.io/name: nosql
        app.kubernetes.io/component: backend
    spec:
      containers:
        - name: mongo
          image: mongo:4.2
          args:
            - --bind_ip
            - 0.0.0.0
          ports:
            - containerPort: 27017
```

File Edit View Terminal Tabs Help

```
- name: mongo
  image: mongo:4.2
  args:
    - --bind_ip
    - 0.0.0.0
  ports:
    - containerPort: 27017
  resources:
    requests:
      memory: "160Mi"
    limits:
      memory: "320Mi"
```

freeqas.com



:wq

```

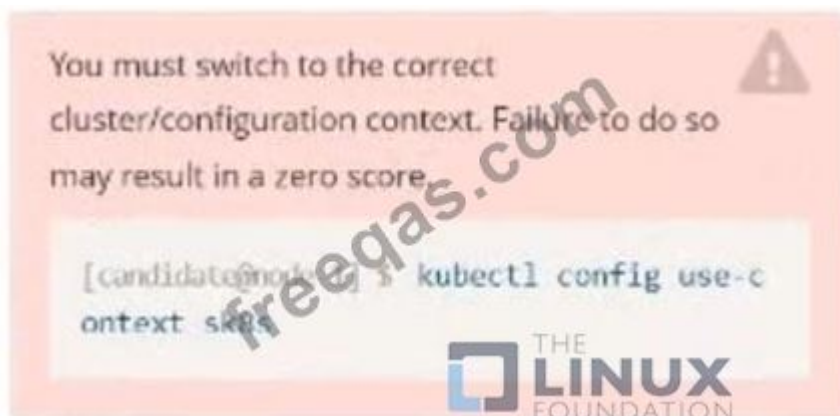
To: <any> (traffic not restricted by destination)
Policy Types: Ingress, Egress

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:~$ 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-74cccf7d64-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

```

NEW QUESTION: 12

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 is manifest file can be found at:

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

Answer:

Solution:

```

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

```

```
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 deploy -n quetzal
NAME              READY   UP-TO-DATE   AVAILABLE   AGE
broker-deployment 3/3      3             3           7h3m
candidate@node-1:~$
```

NEW QUESTION: 13

Refer to Exhibit.



Set Configuration Context:

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

```
Config use-context k8s
```

```
Context
```

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

Task

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

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

Answer:

Solution:

To have Kubernetes automatically restart a pod when an endpoint returns an HTTP 500 on the /healthz endpoint, you will need to configure liveness and readiness probes on the pod.

First, you will need to create a livenessProbe and a readinessProbe in the pod's definition yaml file. The livenessProbe will check the /healthz endpoint, and if it returns an HTTP 500, the pod will be restarted. The readinessProbe will check the /started endpoint, and if it returns an HTTP 500, the pod will not receive traffic.

Here's an example of how you can configure the liveness and readiness probes in the pod definition yaml file:

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
name: probe-pod
```

```
spec:
```

```
containers:
```

```
- name: probe-pod
```

```
image: <image-name>
```

```
ports:
```

```
- containerPort: 8080
```

```
livenessProbe:
```

```
httpGet:
  path: /healthz
  port: 8080
  initialDelaySeconds: 15
  periodSeconds: 10
  failureThreshold: 3
readinessProbe:
  httpGet:
    path: /started
    port: 8080
    initialDelaySeconds: 15
    periodSeconds: 10
    failureThreshold: 3
```

The httpGet specifies the endpoint to check and the port to use. The initialDelaySeconds is the amount of time the pod will wait before starting the probe. periodSeconds is the amount of time between each probe check, and the failureThreshold is the number of failed probes before the pod is considered unresponsive.

You can use kubectl to create the pod by running the following command:

```
kubectl apply -f <filename>.yaml
```

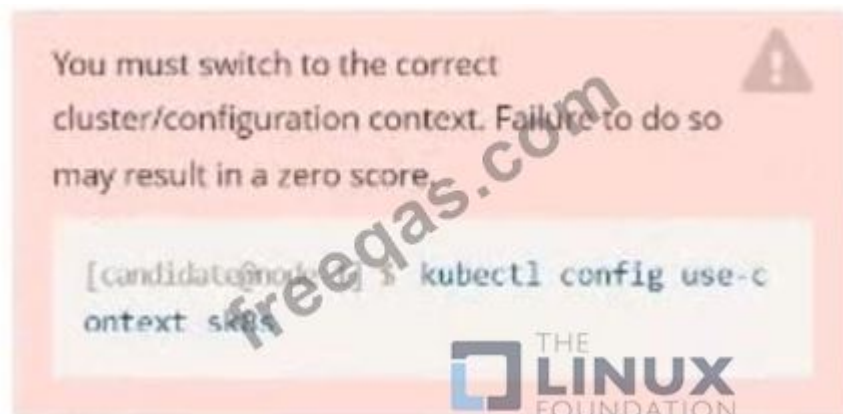
Once the pod is created, Kubernetes will start monitoring it using the configured liveness and readiness probes. If the /healthz endpoint returns an HTTP 500, the pod will be restarted. If the /started endpoint returns an HTTP 500, the pod will not receive traffic.

Please note that if the failure threshold is set to 3, it means that if the probe fails 3 times consecutively it will be considered as a failure.

The above configuration assumes that the application is running on port 8080 and the endpoints are available on the same port.

NEW QUESTION: 14

Refer to Exhibit.

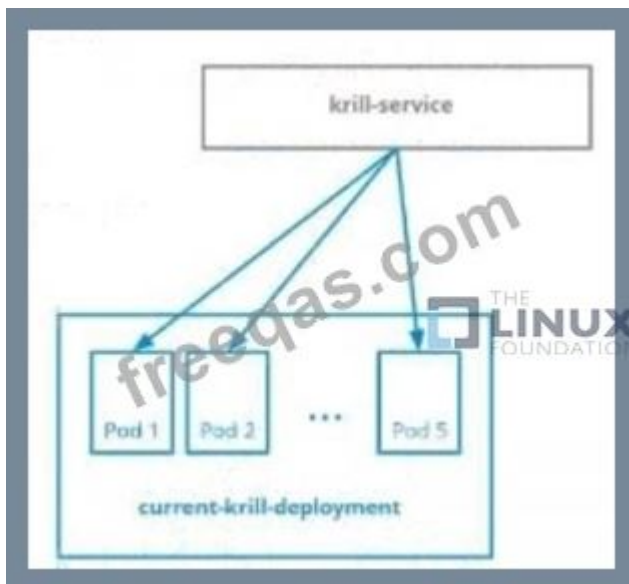


Context

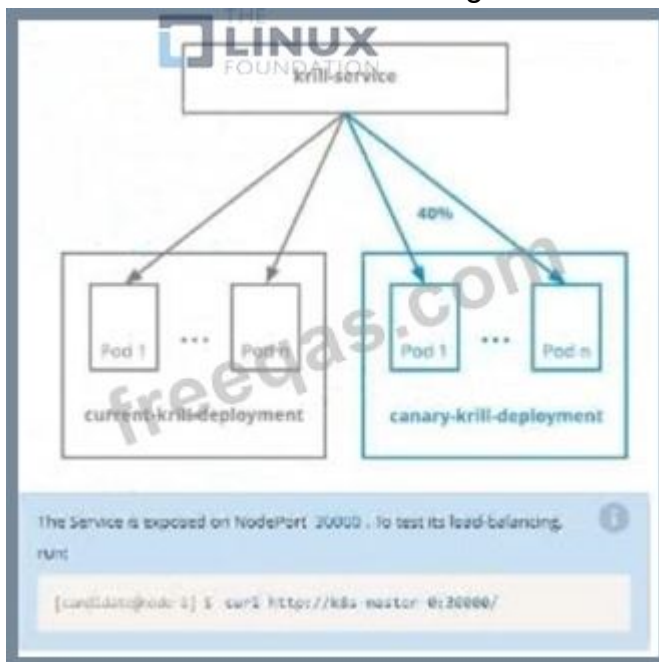
You are asked to prepare a Canary deployment for testing a new application release.

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:

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/
```

File Edit View Terminal Tabs Help

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

Resolving 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

Resolving 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

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

```

022-09-24 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: 15

Refer to Exhibit.



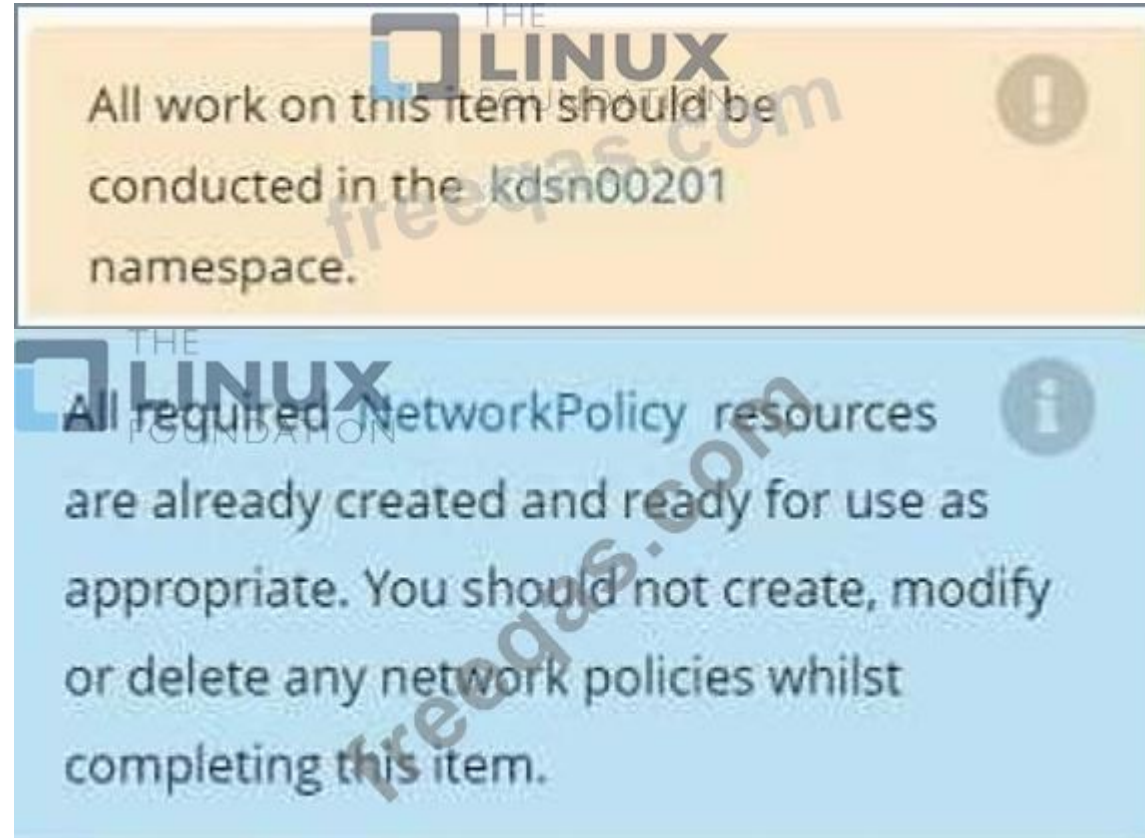
Set Configuration Context:

[student@node-1] \$ | kubectl

Config use-context k8s

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.



Answer:

To allow a pod to send and receive traffic only to and from specific pods, you can use network policies in Kubernetes.

First, you will need to create a network policy that defines the allowed traffic. You can create a network policy yaml file with the following rules:

```
apiVersion: networking.k8s.io/v1
```

```
kind: NetworkPolicy
```

```
metadata:
```

```
name: newpod-network-policy
```

```
namespace: default
```

```
spec:
```

```
podSelector:
```

```
matchLabels:
```

```
app: kdsn00201-newpod
```

```
ingress:
```

```
- from:
```

```
- podSelector:
```

```
matchLabels:
```

```
app: web
```

```
- podSelector:
```

matchLabels:

app: storage

This policy will only allow incoming traffic to the pod with the label app=kdsn00201-newpod from pods with the label app=web or app=storage. If you have different labels on your web and storage pods please update the matchLabels accordingly.

Once you have created the network policy, you can apply it to the cluster by running the following command:

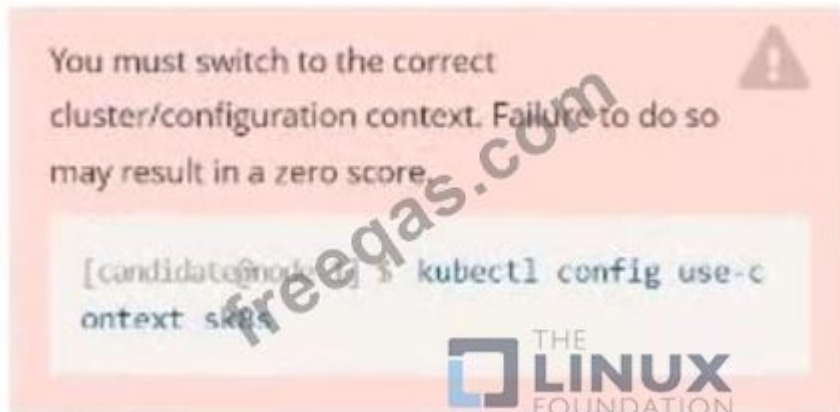
```
kubectl apply -f <network-policy-file>.yaml
```

This will apply the network policy to the cluster, and the newpod will only be able to send and receive traffic to and from the web and storage pods.

Please note that, NetworkPolicy resource is not available by default, you need to enable the NetworkPolicy feature on your Kubernetes cluster. This feature is enabled by default on some clusters and must be explicitly enabled on others. You can check if NetworkPolicy is available by running the command `kubectl api-versions | grep networking` Also, you need to ensure that the pods that you want to allow traffic to and from are running on the same namespace.

NEW QUESTION: 16

Refer to Exhibit.



Task:

A pod within the Deployment named buffalo-deployment and in namespace gorilla is logging errors.

1) Look at the logs identify errors messages.

Find errors, including User "system:serviceaccount:gorilla:default" cannot list resource "deployment" [...] in the namespace "gorilla"

2) Update the Deployment buffalo-deployment to resolve the errors in the logs of the Pod.

The buffalo-deployment 'S manifest can be found at -/prompt/escargot/buffalo-deployment.yaml

Answer:

Solution:

deployment.apps/backend-deployment configured

andidate@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

andidate@node-1:~\$ kubectl get deploy -n staging

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
backend-deployment	3/3	3	3	6h40m

andidate@node-1:~\$ kubectl get deploy -n staging

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
backend-deployment	3/3	3	3	6h41m

andidate@node-1:~\$ vim ~/spicy-pikachu/backend-deployment.yaml

andidate@node-1:~\$ kubectl config use-context k8s

Switched to context "k8s".

andidate@node-1:~\$ kubectl set serviceaccount deploy app-1 app -n frontend

deployment.apps/app-1 serviceaccount updated

andidate@node-1:~\$ kubectl config use-context k8s

Switched to context "k8s".

andidate@node-1:~\$ vim ~/prompt-escargot/buffalo-deployment.yaml

andidate@node-1:~\$ vim ~/prompt-escargot/buffalo-deployment.yaml

andidate@node-1:~\$ kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml

deployment.apps/buffalo-deployment configured

andidate@node-1:~\$ kubectl get pods -n gorilla

NAME	READY	STATUS	RESTARTS	AGE
buffalo-deployment-776844df7f-r5fsb	1/1	Running	0	6h38m
buffalo-deployment-859898c6f5-zx5gj	0/1	ContainerCreating	0	8s

andidate@node-1:~\$ kubectl get deploy -n gorilla

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
buffalo-deployment	1/1	1	1	6h38m

andidate@node-1:~\$



```

File Edit View Terminal Tabs Help
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           28s
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 deploy app-1 app -n frontend
deployment.apps/app-1 serviceaccount updated
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
candidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml
deployment.apps/buffalo-deployment configured
candidate@node-1:~$ kubectl get pods -n go

```

```

File Edit View Terminal Tabs Help
candidate@node-1:~$ kubectl get pods -n staging
NAME                                READY   STATUS    RESTARTS   AGE
backend-deployment-59d449b99d-cxct6 1/1     Running   0           28s
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 deploy app-1 app -n frontend
deployment.apps/app-1 serviceaccount updated
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
candidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml
deployment.apps/buffalo-deployment configured
candidate@node-1:~$ kubectl get pods -n gorilla
NAME                                READY   STATUS    RESTARTS   AGE
buffalo-deployment-776844df7f-r5fsb 1/1     Running   0           6h38m
buffalo-deployment-859898c6f5-zx5gj 0/1     ContainerCreating 0           8s
candidate@node-1:~$ kubectl get deploy -n gorilla
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
buffalo-deployment  1/1     1             1           6h38m
candidate@node-1:~$

```

File Edit View Terminal Tabs Help

```
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 deploy app-1 app-1 frontend
deployment.apps/app-1 serviceaccount updated
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
candidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
candidate@node-1:~$ kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml
deployment.apps/buffalo-deployment configured
candidate@node-1:~$ kubectl get pods -n gorilla
NAME READY STATUS RESTARTS AGE
buffalo-deployment-776844df7f-rp1sb 1/1 Running 0 6h38m
buffalo-deployment-859898c6f5-zx5qj 0/1 ContainerCreating 0 8s
candidate@node-1:~$ kubectl get deploy -n gorilla
NAME READY UP-TO-DATE AVAILABLE AGE
buffalo-deployment 1/1 1 1 6h38m
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl edit deploy ckad00017-deployment -n ckad00017
```

File Edit View Terminal Tabs Help

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
```

```
#
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
    creationTimestamp: "2022-09-24T04:27:03Z"
  generation: 1
  labels:
    app: nginx
    name: ckad00017-deployment
    namespace: ckad00017
    resourceVersion: "3349"
    uid: 1cd67613-fade-46e9-b741-94298b9c6e7c
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: nginx
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      -- INSERT --
```

19/31

Top

```

resourceVersion: "3349"
uid: 1cd67613-fade-46e9-b741-94298b9c6e7c
ec:
progressDeadlineSeconds: 600
replicas: 2
revisionHistoryLimit: 10
selector:
  matchLabels:
    app: nginx
strategy:
  rollingUpdate:
    maxSurge: 25%
    maxUnavailable: 25%
  type: RollingUpdate
template:
  metadata:
    creationTimestamp: null
    labels:
      app: nginx
  spec:
    containers:
      - image: nginx:latest
        imagePullPolicy: Always
        name: nginx
        ports:
          - containerPort: 80
            protocol: TCP
        resources: {}
    terminationMessagePath: /dev/termination-log
    terminationMessagePolicy: File
    dnsPolicy: ClusterFirst

```



```

ckend-deployment-59d449b99d-h2zjq 0/1 Running 0 9s
ckend-deployment-78976f74f5-b8c85 1/1 Running 0 6h40m
ckend-deployment-78976f74f5-flfsj 1/1 Running 0 6h40m
ndidate@node-1:~$ kubectl get deploy -n staging
NAME READY UP-TO-DATE AVAILABLE AGE
ckend-deployment 3/3 3 3 6h40m
ndidate@node-1:~$ kubectl get deploy -n staging
NAME READY UP-TO-DATE AVAILABLE AGE
ckend-deployment 3/3 3 3 6h41m
ndidate@node-1:~$ vim ~/spicy-pikachu/backend-deployment.yaml
ndidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
ndidate@node-1:~$ kubectl set serviceaccount deploy app-1 app -n frontend
deployment.apps/app-1 serviceaccount updated
ndidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
ndidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
ndidate@node-1:~$ vim ~/prompt-escargot/buffalo-deployment.yaml
ndidate@node-1:~$ kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml
deployment.apps/buffalo-deployment configured
ndidate@node-1:~$ kubectl get pods -n gorilla
NAME READY STATUS RESTARTS AGE
bfalo-deployment-776844df7f-r5fqb 1/1 Running 0 6h38m
bfalo-deployment-859898c6f5-zx5gj 0/1 ContainerCreating 0 8s
ndidate@node-1:~$ kubectl get deploy -n gorilla
NAME READY UP-TO-DATE AVAILABLE AGE
bfalo-deployment 1/1 1 1 6h38m
ndidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
ndidate@node-1:~$ kubectl edit deploy ckad00017-deployment -n ckad00017
deployment.apps/ckad00017-deployment edited

```

```

ffalo-deployment 1/1 1 6h38m
candidate@node-1:~$ kubectl config use-context k8s
Switched to context "k8s".
candidate@node-1:~$ kubectl edit deploy ckad00017-deployment -n ckad00017
deployment.apps/ckad00017-deployment edited
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose deploy ckad00017-deployment -n ckad00017
ad00014 ckad00015 ckad00017
candidate@node-1:~$ kubectl expose service/cherry
service/cherry exposed
candidate@node-1:~$ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 77d
candidate@node-1:~$ kubectl get svc -n ckad00017
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
cherry NodePort 10.100.100.176 <none> 8888:30683/TCP 24s
candidate@node-1:~$ kubectl expose service deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
Error from server (NotFound): services "deploy" not found
Error from server (NotFound): services "ckad00017-deployment" not found
candidate@node-1:~$ kubectl get svc -n ckad00017
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
cherry NodePort 10.100.100.176 <none> 8888:30683/TCP 46s
candidate@node-1:~$ history
1 vi ~/spicy-pikachu/backend-deployment.yaml
2 kubectl config use-context k8s
3 vim .vimrc
4 vim ~/spicy-pikachu/backend-deployment.yaml
5 kubectl apply -f ~/spicy-pikachu/backend-deployment.yaml
6 kubectl get pods -n staging
7 kubectl get deploy -n staging
8 vim ~/spicy-pikachu/backend-deployment.yaml
9 kubectl config use-context k8s
10 kubectl set serviceaccount deploy app1 app -n frontend
11 kubectl config use-context k8s
12 vim ~/prompt-escargot/buffalo-deployment.yaml
13 kubectl apply -f ~/prompt-escargot/buffalo-deployment.yaml
14 kubectl get pods -n gorilla
15 kubectl get deploy -n gorilla
16 kubectl config use-context k8s
17 kubectl edit deploy ckad00017-deployment -n ckad00017
18 kubectl expose deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
19 kubectl get svc
20 kubectl get svc -n ckad00017
21 kubectl expose service deploy ckad00017-deployment -n ckad00017 --name=cherry --port=8888 --type=NodePort
22 kubectl get svc -n ckad00017
23 history
candidate@node-1:~$

```

Valid CKAD Dumps shared by PrepPdf.com for Helping Passing CKAD Exam! PrepPdf.com now offer the **newest CKAD exam dumps**, the PrepPdf.com CKAD exam questions have been updated and answers have been corrected get the **newest** PrepPdf.com CKAD dumps with Test Engine here: <https://www.preppdf.com/Linux-Foundation/CKAD-prepaway-exam-dumps.html> (239 Q&As Dumps, **40%OFF** Special Discount: **Exam-Tests**)

Valid CKAD Dumps shared by PrepPdf.com for Helping Passing CKAD Exam! PrepPdf.com now offer the **newest CKAD exam dumps**, the PrepPdf.com CKAD exam questions have been updated and answers have been corrected get the **newest** PrepPdf.com CKAD dumps with Test Engine here: <https://www.preppdf.com/Linux-Foundation/CKAD-prepaway-exam-dumps.html> (239 Q&As Dumps, **40%OFF** Special Discount: **Exam-Tests**)