CKS$^{Q\&As}$

Certified Kubernetes Security Specialist (CKS) Exam

Pass Linux Foundation CKS Exam with 100% Guarantee

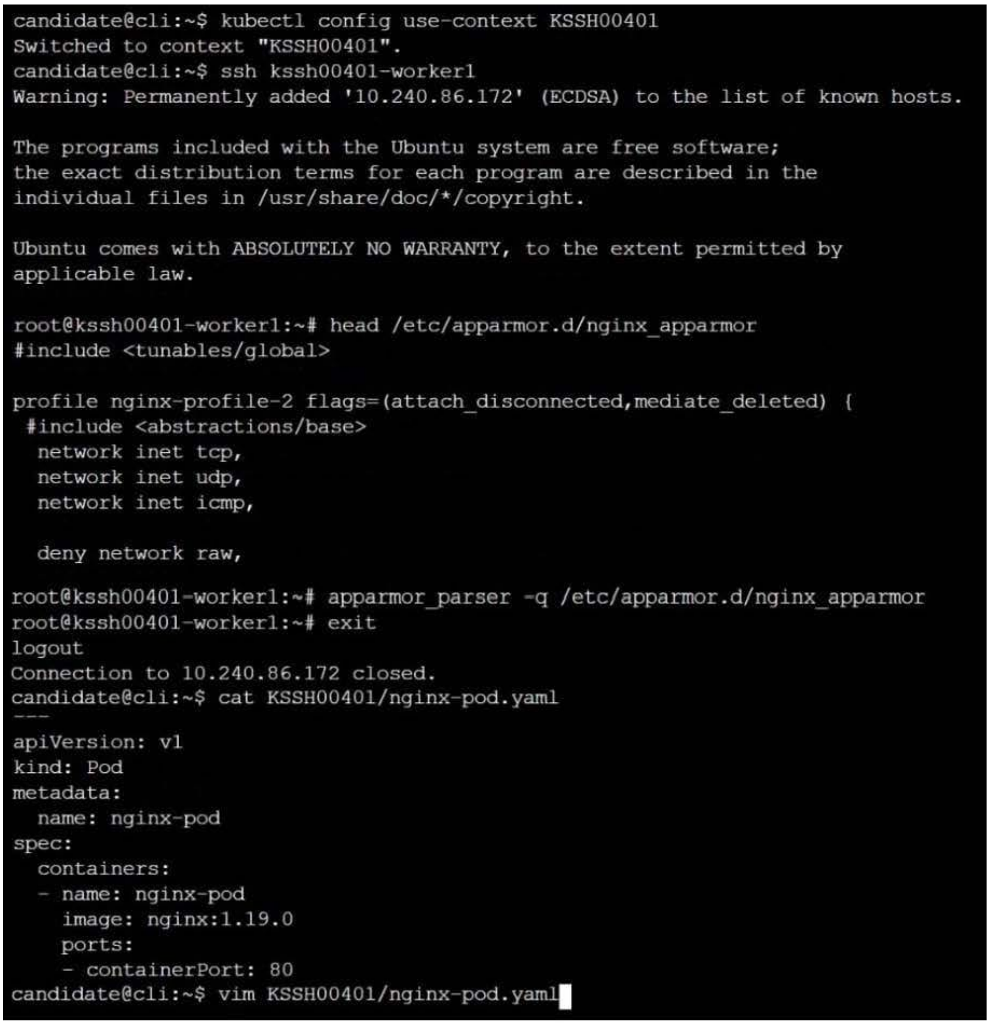
Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.leads4pass.com/cks.html

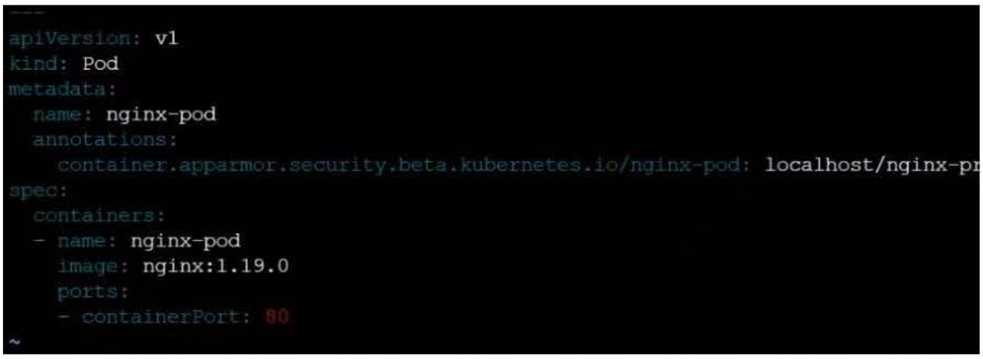
100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Linux Foundation Official Exam Center

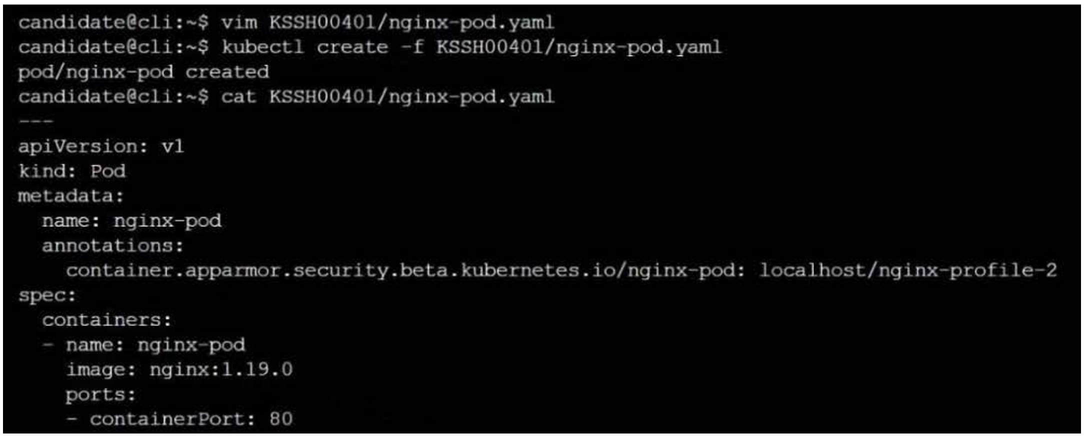
QUESTION 1



```
candidate@cli:~$ kubectl config use-context KSSH00401
Switched to context "KSSH00401".
candidate@cli:~$ ssh kssh00401-worker1
Warning: Permanently added '10.240.86.172' (ECDSA) to the list of known hosts.

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.

root@kssh00401-worker1:~# head /etc/apparmor.d/nginx_apparmor
#include <tunables/global>

profile nginx-profile-2 flags=(attach_disconnected,mediate_deleted) {
 #include <abstractions/base>
  network inet tcp,
  network inet udp,
  network inet icmp,

  deny network raw,

root@kssh00401-worker1:~# apparmor_parser -q /etc/apparmor.d/nginx_apparmor
root@kssh00401-worker1:~# exit
logout
Connection to 10.240.86.172 closed.
candidate@cli:~$ cat KSSH00401/nginx-pod.yaml
---
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
  - name: nginx-pod
    image: nginx:1.19.0
    ports:
    - containerPort: 80
candidate@cli:~$ vim KSSH00401/nginx-pod.yaml
```



```
---
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  annotations:
    container.apparmor.security.beta.kubernetes.io/nginx-pod: localhost/nginx-pr
spec:
  containers:
  - name: nginx-pod
    image: nginx:1.19.0
    ports:
    - containerPort: 80
~
```



```
candidate@cli:~$ vim KSSH00401/nginx-pod.yaml
candidate@cli:~$ kubectl create -f KSSH00401/nginx-pod.yaml
pod/nginx-pod created
candidate@cli:~$ cat KSSH00401/nginx-pod.yaml
---
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  annotations:
    container.apparmor.security.beta.kubernetes.io/nginx-pod: localhost/nginx-profile-2
spec:
  containers:
  - name: nginx-pod
    image: nginx:1.19.0
    ports:
    - containerPort: 80
```

Fix all issues via configuration and restart the affected components to ensure the new setting takes effect. Fix all of the following violations that were found against the API server:

1.

Ensure that the RotateKubeletServerCertificate argument is set to true.

2.

Ensure that the admission control plugin PodSecurityPolicy is set.

3.

Ensure that the --kubelet-certificate-authority argument is set as appropriate. Fix all of the following violations that were found against the Kubelet:

1.

Ensure the --anonymous-auth argument is set to false.

2.

Ensure that the --authorization-mode argument is set to Webhook. Fix all of the following violations that were found against the ETCD:

1.

Ensure that the --auto-tls argument is not set to true

2.

Ensure that the --peer-auto-tls argument is not set to true

Hint: Take the use of Tool Kube-Bench

A. See the below.

B. PlaceHolder

Correct Answer: A

Fix all of the following violations that were found against the API server:

a. Ensure that the RotateKubeletServerCertificate argument is set to true.

apiVersion: v1 kind: Pod metadata: creationTimestamp: null labels: component: kubelet tier: control-plane name: kubelet namespace: kube-system spec: containers:

-command:

-kube-controller-manager + - --feature-gates=RotateKubeletServerCertificate=true image: gcr.io/google_containers/kubelet-amd64:v1.6.0 livenessProbe: failureThreshold: 8 httpGet: host: 127.0.0.1 path: /healthz port: 6443 scheme: HTTPS initialDelaySeconds: 15 timeoutSeconds: 15 name: kubelet resources: requests: cpu: 250m volumeMounts:

-

mountPath: /etc/kubernetes/ name: k8s readOnly: true

-

mountPath: /etc/ssl/certs name: certs

-

mountPath: /etc/pki name: pki hostNetwork: true volumes:

-

hostPath: path: /etc/kubernetes name: k8s

-

hostPath: path: /etc/ssl/certs name: certs

-

hostPath: path: /etc/pki name: pki

b.

Ensure that the admission control plugin PodSecurityPolicy is set.

audit: "/bin/ps -ef | grep $apiserverbin | grep -v grep"

tests:

test_items:

-flag: "--enable-admission-plugins"

compare:

op: has

value: "PodSecurityPolicy"

set: true

remediation: |

Follow the documentation and create Pod Security Policy objects as per your environment.

Then, edit the API server pod specification file $apiserverconf

on the master node and set the --enable-admission-plugins parameter to a

value that includes PodSecurityPolicy :

--enable-admission-plugins=...,PodSecurityPolicy,...

Then restart the API Server.

scored: true

c. Ensure that the --kubelet-certificate-authority argument is set as appropriate. audit: "/bin/ps -ef | grep $apiserverbin | grep -v grep"

tests: test_items:

-flag: "--kubelet-certificate-authority"

set: true

remediation: |

Follow the Kubernetes documentation and setup the TLS connection between the

apiserver and kubelets. Then, edit the API server pod specification file

$apiserverconf on the master node and set the --kubelet-certificate-authority

parameter to the path to the cert file for the certificate authority.

--kubelet-certificate-authority=

scored: true

Fix all of the following violations that were found against the ETCD:

a.

Ensure that the --auto-tls argument is not set to true Edit the etcd pod specification file $etcdconf on the masternode and either remove the -- auto-tls parameter or set it to false.--auto-tls=false

b.

Ensure that the --peer-auto-tls argument is not set to true

Edit the etcd pod specification file $etcdconf on the masternode and either remove the -- peer-auto-tls parameter or set it to false.--peer-auto-tls=false

QUESTION 2

Create a network policy named allow-np, that allows pod in the namespace staging to connect to port 80 of other pods in the same namespace.

Ensure that Network Policy:

1.

Does not allow access to pod not listening on port 80.

2.

Does not allow access from Pods, not in namespace staging.

A. See the explanation below:

B. PlaceHolder

Correct Answer: A

apiVersion: networking.k8s.io/v1

kind: NetworkPolicy

metadata:

name: network-policy

spec:

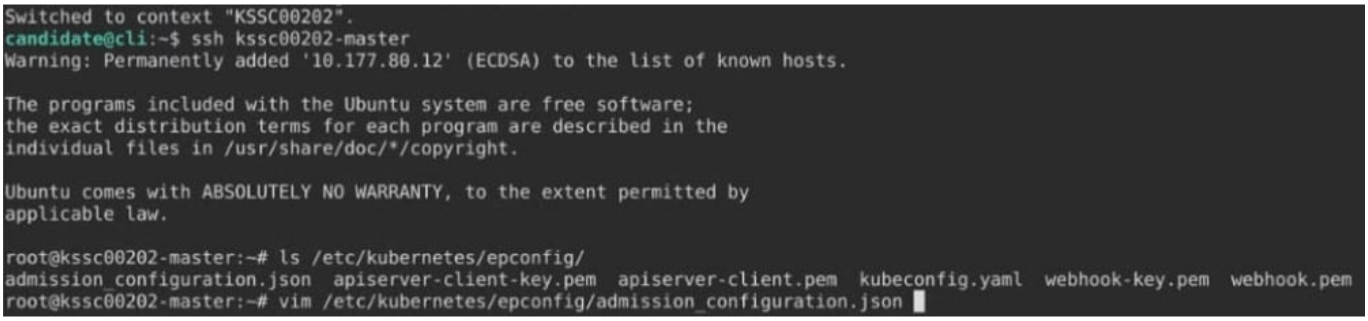
podSelector: {} #selects all the pods in the namespace deployed policyTypes:

-Ingress ingress:

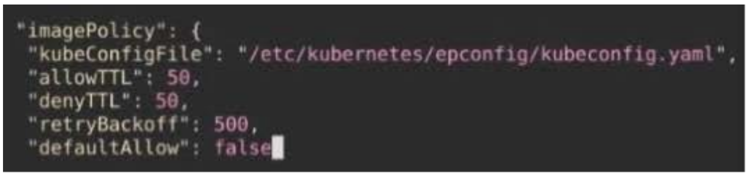
-ports: #in input traffic allowed only through 80 port only

-protocol: TCP port: 80

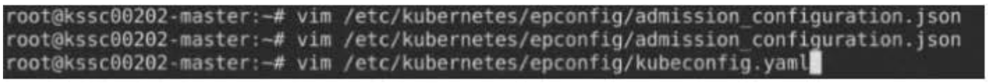
QUESTION 3



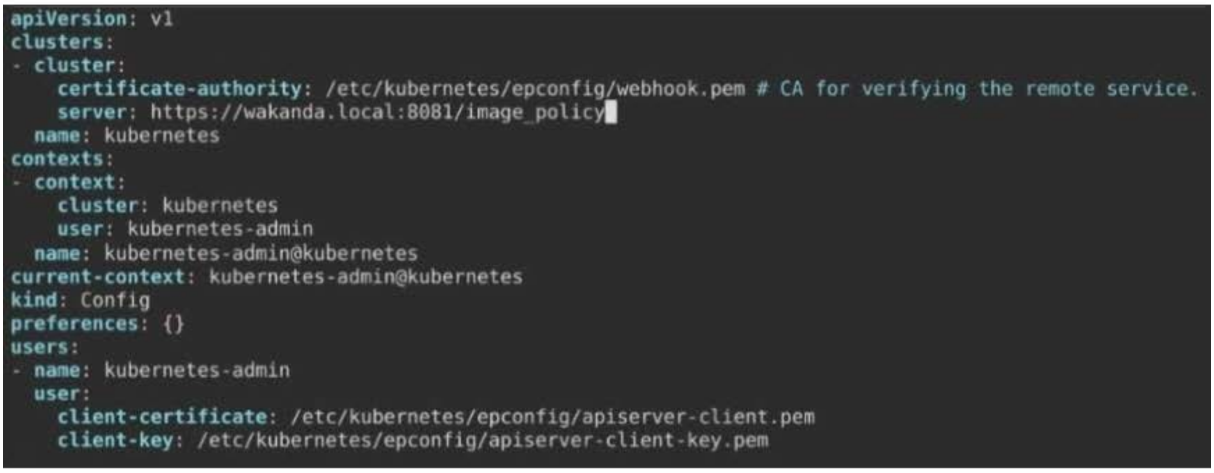
```
Switched to context "KSSC00202".
candidate@cli:~$ ssh kssc00202-master
Warning: Permanently added '10.177.80.12' (ECDSA) to the list of known hosts.

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.

root@kssc00202-master:~# ls /etc/kubernetes/epconfig/
admission_configuration.json  apiserver-client-key.pem  apiserver-client.pem  kubeconfig.yaml  webhook-key.pem  webhook.pem
root@kssc00202-master:~# vim /etc/kubernetes/epconfig/admission_configuration.json
```



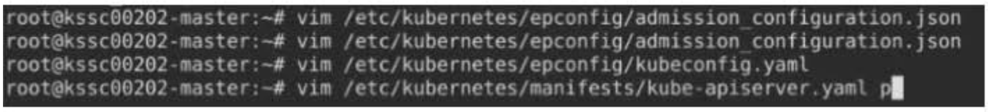
```
"imagePolicy": {
 "kubeConfigFile": "/etc/kubernetes/epconfig/kubeconfig.yaml",
 "allowTTL": 50,
 "denyTTL": 50,
 "retryBackoff": 500,
 "defaultAllow": false
```



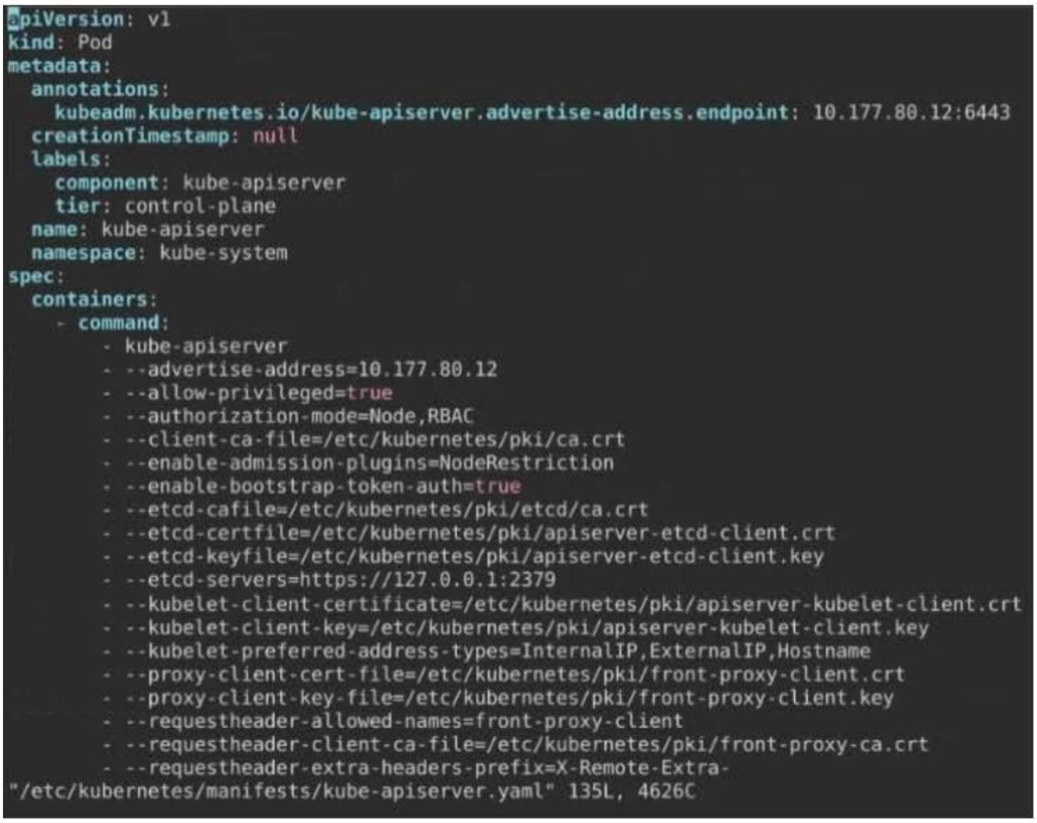
```
root@kssc00202-master:~# vim /etc/kubernetes/epconfig/admission_configuration.json
root@kssc00202-master:~# vim /etc/kubernetes/epconfig/admission_configuration.json
root@kssc00202-master:~# vim /etc/kubernetes/epconfig/kubeconfig.yaml
```



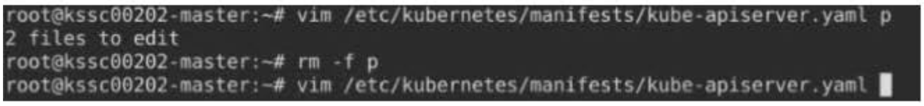
```
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /etc/kubernetes/epconfig/webhook.pem # CA for verifying the remote service.
    server: https://wakanda.local:8081/image_policy
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
kind: Config
preferences: {}
users:
- name: kubernetes-admin
  user:
    client-certificate: /etc/kubernetes/epconfig/apiserver-client.pem
    client-key: /etc/kubernetes/epconfig/apiserver-client-key.pem
```



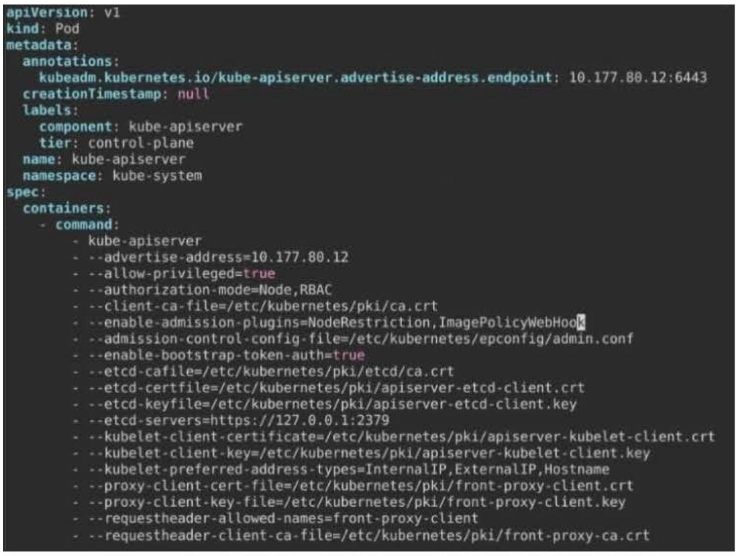
```
root@kssc00202-master:~# vim /etc/kubernetes/epconfig/admission_configuration.json
root@kssc00202-master:~# vim /etc/kubernetes/epconfig/admission_configuration.json
root@kssc00202-master:~# vim /etc/kubernetes/epconfig/kubeconfig.yaml
root@kssc00202-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml p
```



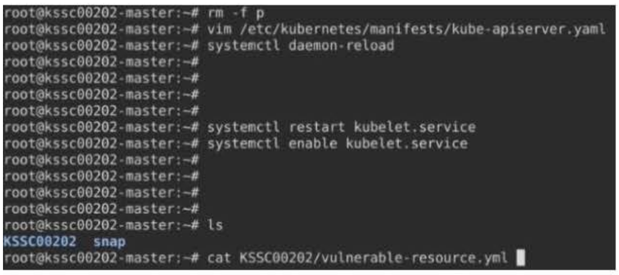
```
apiVersion: v1
kind: Pod
metadata:
  annotations:
    kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: 10.177.80.12:6443
  creationTimestamp: null
  labels:
    component: kube-apiserver
    tier: control-plane
  name: kube-apiserver
  namespace: kube-system
spec:
  containers:
  - command:
    - kube-apiserver
    - --advertise-address=10.177.80.12
    - --allow-privileged=true
    - --authorization-mode=Node,RBAC
    - --client-ca-file=/etc/kubernetes/pki/ca.crt
    - --enable-admission-plugins=NodeRestriction
    - --enable-bootstrap-token-auth=true
    - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
    - --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt
    - --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
    - --etcd-servers=https://127.0.0.1:2379
    - --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-kubelet-client.crt
    - --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key
    - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
    - --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-client.crt
    - --proxy-client-key-file=/etc/kubernetes/pki/front-proxy-client.key
    - --requestheader-allowed-names=front-proxy-client
    - --requestheader-client-ca-file=/etc/kubernetes/pki/front-proxy-ca.crt
    - --requestheader-extra-headers-prefix=X-Remote-Extra-
"/etc/kubernetes/manifests/kube-apiserver.yaml" 135L, 4626C
```



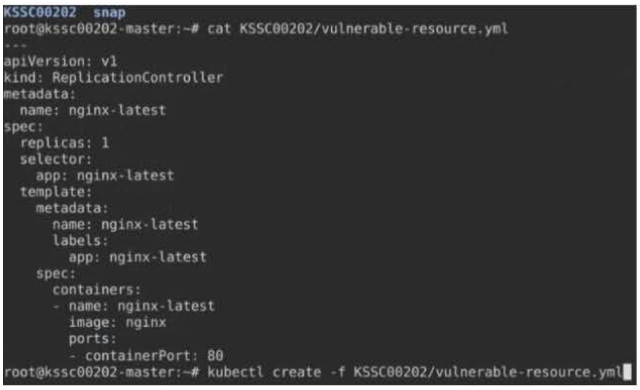
```
root@kssc00202-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml p
2 files to edit
root@kssc00202-master:~# rm -f p
root@kssc00202-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
```



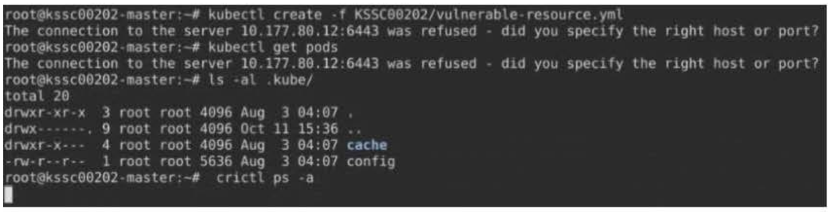
```
apiVersion: v1
kind: Pod
metadata:
  annotations:
    kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: 10.177.80.12:6443
  creationTimestamp: null
  labels:
    component: kube-apiserver
    tier: control-plane
  name: kube-apiserver
  namespace: kube-system
spec:
  containers:
  - command:
    - kube-apiserver
    - --advertise-address=10.177.80.12
    - --allow-privileged=true
    - --authorization-mode=Node,RBAC
    - --client-ca-file=/etc/kubernetes/pki/ca.crt
    - --enable-admission-plugins=NodeRestriction,ImagePolicyWebHook
    - --admission-control-config-file=/etc/kubernetes/epconfig/admin.conf
    - --enable-bootstrap-token-auth=true
    - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
    - --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt
    - --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
    - --etcd-servers=https://127.0.0.1:2379
    - --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-kubelet-client.crt
    - --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key
    - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
    - --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-client.crt
    - --proxy-client-key-file=/etc/kubernetes/pki/front-proxy-client.key
    - --requestheader-allowed-names=front-proxy-client
    - --requestheader-client-ca-file=/etc/kubernetes/pki/front-proxy-ca.crt
```



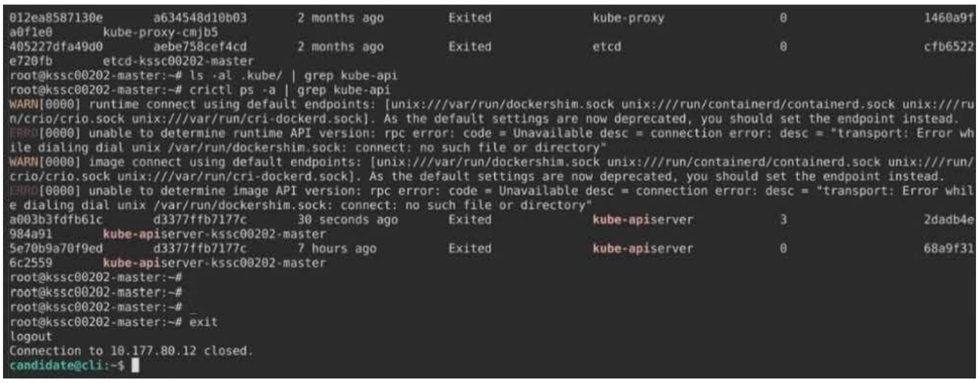
```
root@kssc00202-master:~# rm -f p
root@kssc00202-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
root@kssc00202-master:~# systemctl daemon-reload
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~# systemctl restart kubelet.service
root@kssc00202-master:~# systemctl enable kubelet.service
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~# ls
KSSC00202  snap
root@kssc00202-master:~# cat KSSC00202/vulnerable-resource.yml
```



```
KSSC00202  snap
root@kssc00202-master:~# cat KSSC00202/vulnerable-resource.yml
---
apiVersion: v1
kind: ReplicationController
metadata:
  name: nginx-latest
spec:
  replicas: 1
  selector:
    app: nginx-latest
  template:
    metadata:
      name: nginx-latest
      labels:
        app: nginx-latest
    spec:
      containers:
      - name: nginx-latest
        image: nginx
        ports:
        - containerPort: 80
root@kssc00202-master:~# kubectl create -f KSSC00202/vulnerable-resource.yml
```



```
root@kssc00202-master:~# kubectl create -f KSSC00202/vulnerable-resource.yml
The connection to the server 10.177.80.12:6443 was refused - did you specify the right host or port?
root@kssc00202-master:~# kubectl get pods
The connection to the server 10.177.80.12:6443 was refused - did you specify the right host or port?
root@kssc00202-master:~# ls -al .kube/
total 20
drwxr-xr-x  3 root root 4096 Aug  3 04:07 .
drwx------. 9 root root 4096 Oct 11 15:36 ..
drwxr-x---  4 root root 4096 Aug  3 04:07 cache
-rw-r--r--  1 root root 5636 Aug  3 04:07 config
root@kssc00202-master:~#  crictl ps -a
```



```
012ea8587130e       a634548d10b03       2 months ago        Exited              kube-proxy                0                   1460a9f
a0f1e0          kube-proxy-cmjb5
405227dfa49d0       aebe758cef4cd       2 months ago        Exited              etcd                      0                   cfb6522
e720fb          etcd-kssc00202-master
root@kssc00202-master:~# ls -al .kube/ | grep kube-api
root@kssc00202-master:~# crictl ps -a | grep kube-api
WARN[0000] runtime connect using default endpoints: [unix:///var/run/dockershim.sock unix:///run/containerd/containerd.sock unix:///ru
n/crio/crio.sock unix:///var/run/cri-dockerd.sock]. As the default settings are now deprecated, you should set the endpoint instead.
ERRO[0000] unable to determine runtime API version: rpc error: code = Unavailable desc = connection error: desc = "transport: Error wh
ile dialing dial unix /var/run/dockershim.sock: connect: no such file or directory"
WARN[0000] image connect using default endpoints: [unix:///var/run/dockershim.sock unix:///run/containerd/containerd.sock unix:///run/
crio/crio.sock unix:///var/run/cri-dockerd.sock]. As the default settings are now deprecated, you should set the endpoint instead.
ERRO[0000] unable to determine image API version: rpc error: code = Unavailable desc = connection error: desc = "transport: Error whil
e dialing dial unix /var/run/dockershim.sock: connect: no such file or directory"
a003b3fdfb61c       d3377ffb7177c       30 seconds ago      Exited              kube-apiserver            3                   2dadb4e
984a91          kube-apiserver-kssc00202-master
5e70b9a70f9ed       d3377ffb7177c       7 hours ago         Exited              kube-apiserver            0                   68a9f31
6c2559          kube-apiserver-kssc00202-master
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~# _
root@kssc00202-master:~# exit
logout
Connection to 10.177.80.12 closed.
candidate@cli:~$
```

Cluster: dev Master node: master1 Worker node: worker1 You can switch the cluster/configuration context using the following command: [desk@cli] $ kubectl config use-context dev Task:

Retrieve the content of the existing secret named adam in the safe namespace.

Store the username field in a file names /home/cert-masters/username.txt, and the password field in a file named /home/cert-masters/password.txt.

1.

You must create both files; they don\\'t exist yet.

2.

Do not use/modify the created files in the following steps, create new temporary files if needed.

Create a new secret names newsecret in the safe namespace, with the following content:

Username: dbadmin Password: moresecurepas

Finally, create a new Pod that has access to the secret newsecret via a volume:

Namespace:safe Pod name:mysecret-pod Container name:db-container Image:redis Volume name:secret-vol Mount path:/etc/mysecret

A. See the explanation below

B. PlaceHolder

Correct Answer: A

QUESTION 4

A container image scanner is set up on the cluster.

Given an incomplete configuration in the directory

/etc/kubernetes/confcontrol and a functional container image scanner with HTTPS endpoint https://test-server.local.8081/image_policy

1.

Enable the admission plugin.

2.

Validate the control configuration and change it to implicit deny.

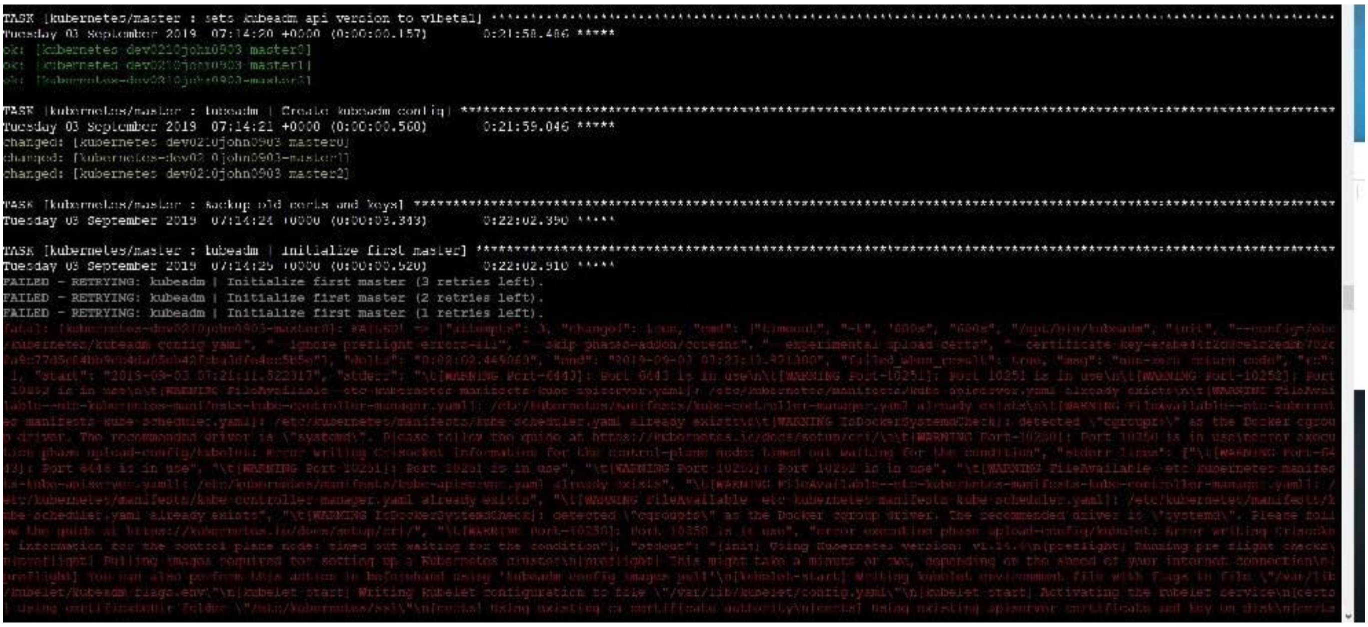
Finally, test the configuration by deploying the pod having the image tag as latest.

A. See explanation below.

B. PlaceHolder

Correct Answer: A

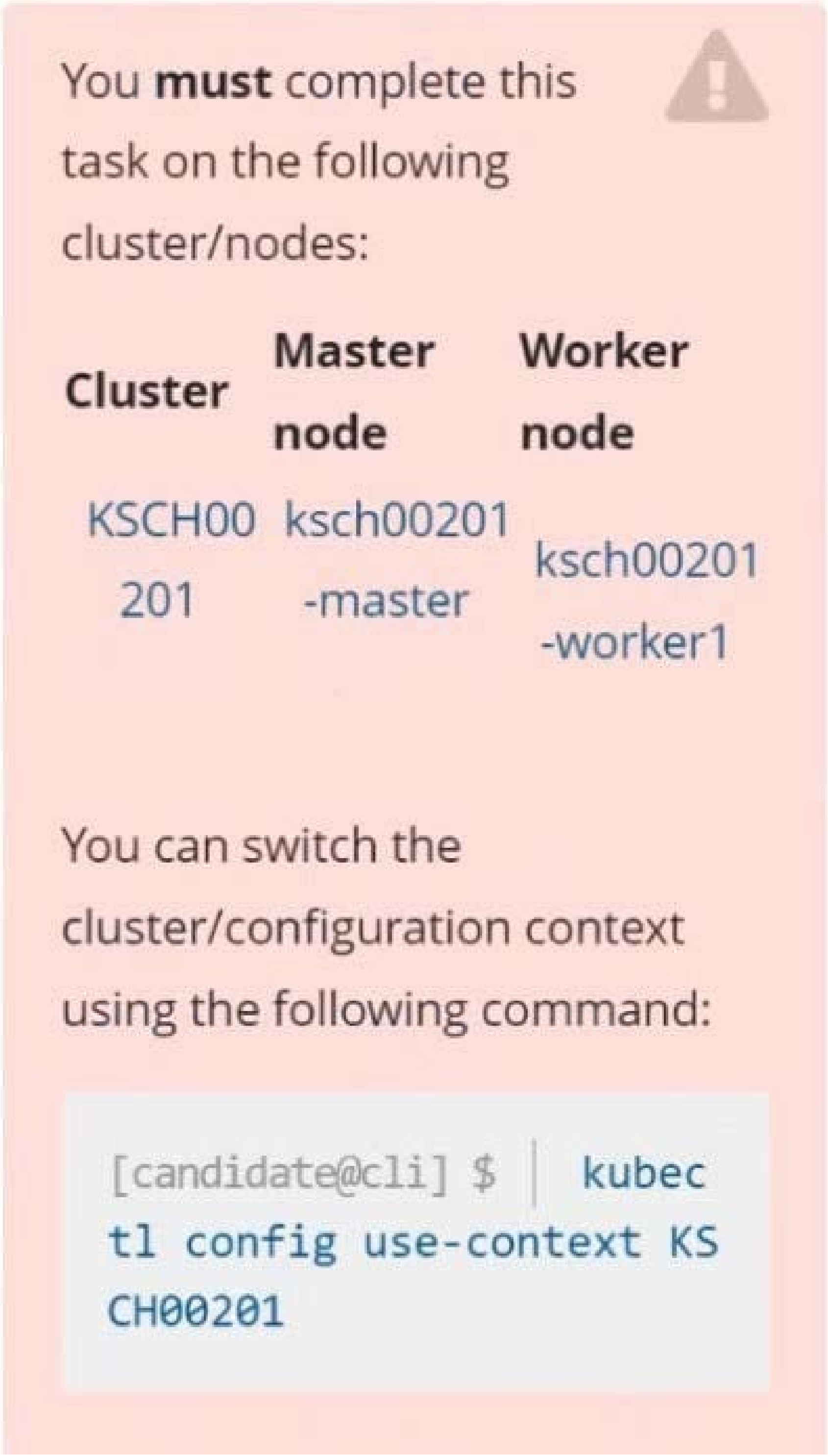
ssh-add ~/.ssh/tempprivate eval "$(ssh-agent -s)" cd contrib/terraform/aws vi terraform.tfvars terraform init terraform apply -var-file=credentials.tfvars ansible-playbook -i ./inventory/hosts ./cluster.yml -e ansible_ssh_user=core -e bootstrap_os=coreos -b --become-user=root --flush-cache -e ansible_user=core



```
TASK [kubernetes/master : sets kubeadm api version to v1beta1] ***********************************************************************************
Tuesday 03 September 2019  07:14:20 +0000 (0:00:00.157)       0:21:58.486 *****
ok: [kubernetes-dev0210john0903-master0]
ok: [kubernetes-dev0210john0903-master1]
ok: [kubernetes-dev0210john0903-master2]

TASK [kubernetes/master : kubeadm | Create kubeadm config] ***************************************************************************************
Tuesday 03 September 2019  07:14:21 +0000 (0:00:00.560)       0:21:59.046 *****
changed: [kubernetes-dev0210john0903-master0]
changed: [kubernetes-dev0210john0903-master1]
changed: [kubernetes-dev0210john0903-master2]

TASK [kubernetes/master : Backup old certs and keys] *********************************************************************************************
Tuesday 03 September 2019  07:14:24 +0000 (0:00:03.343)       0:22:02.390 *****

TASK [kubernetes/master : kubeadm | Initialize first master] *************************************************************************************
Tuesday 03 September 2019  07:14:25 +0000 (0:00:00.520)       0:22:02.910 *****
FAILED - RETRYING: kubeadm | Initialize first master (3 retries left).
FAILED - RETRYING: kubeadm | Initialize first master (2 retries left).
FAILED - RETRYING: kubeadm | Initialize first master (1 retries left).
fatal: [kubernetes-dev0210john0903-master0]: FAILED! => {"attempts": 3, "changed": true, "cmd": ["timeout", "-k", "600s", "600s", "/opt/bin/kubeadm", "init", "--config=/etc
/kubernetes/kubeadm-config.yaml", "--ignore-preflight-errors=all", "--skip-phases=addon/coredns", "--experimental-upload-certs", "--certificate-key=[illegible]
[illegible]"], "delta": "0:02:02.449060", "end": "2019-09-03 07:23:13.971300", "failed_when_result": true, "msg": "non-zero return code", "rc":
1, "start": "2019-09-03 07:21:11.522317", "stderr": "\t[WARNING Port-6443]: Port 6443 is in use\n\t[WARNING Port-10251]: Port 10251 is in use\n\t[WARNING Port-10252]: Port
10252 is in use\n\t[WARNING FileAvailable--etc-kubernetes-manifests-kube-apiserver.yaml]: /etc/kubernetes/manifests/kube-apiserver.yaml already exists\n\t[WARNING FileAvai
lable--etc-kubernetes-manifests-kube-controller-manager.yaml]: /etc/kubernetes/manifests/kube-controller-manager.yaml already exists\n\t[WARNING FileAvailable--etc-kubernet
es-manifests-kube-scheduler.yaml]: /etc/kubernetes/manifests/kube-scheduler.yaml already exists\n\t[WARNING IsDockerSystemdCheck]: detected \"cgroupfs\" as the Docker cgrou
p driver. The recommended driver is \"systemd\". Please follow the guide at https://kubernetes.io/docs/setup/cri/\n\t[WARNING Port-10250]: Port 10250 is in use\nerror execu
tion phase upload-config/kubelet: Error writing Crisocket information for the control-plane node: timed out waiting for the condition", "stderr_lines": ["\t[WARNING Port-64
43]: Port 6443 is in use", "\t[WARNING Port-10251]: Port 10251 is in use", "\t[WARNING Port-10252]: Port 10252 is in use", "\t[WARNING FileAvailable--etc-kubernetes-manifes
ts-kube-apiserver.yaml]: /etc/kubernetes/manifests/kube-apiserver.yaml already exists", "\t[WARNING FileAvailable--etc-kubernetes-manifests-kube-controller-manager.yaml]: /
etc/kubernetes/manifests/kube-controller-manager.yaml already exists", "\t[WARNING FileAvailable--etc-kubernetes-manifests-kube-scheduler.yaml]: /etc/kubernetes/manifests/k
ube-scheduler.yaml already exists", "\t[WARNING IsDockerSystemdCheck]: detected \"cgroupfs\" as the Docker cgroup driver. The recommended driver is \"systemd\". Please foll
ow the guide at https://kubernetes.io/docs/setup/cri/", "\t[WARNING Port-10250]: Port 10250 is in use", "error execution phase upload-config/kubelet: Error writing Crisocke
t information for the control plane node: timed out waiting for the condition"], "stdout": "[init] Using Kubernetes version: v1.14.[illegible]\n[preflight] Running pre-flight checks\
n[preflight] Pulling images required for setting up a Kubernetes cluster\n[preflight] This might take a minute or two, depending on the speed of your internet connection\n[
preflight] You can also perform this action in beforehand using 'kubeadm config images pull'\n[kubelet-start] Writing kubelet environment file with flags to file \"/var/lib
/kubelet/kubeadm-flags.env\"\n[kubelet-start] Writing kubelet configuration to file \"/var/lib/kubelet/config.yaml\"\n[kubelet-start] Activating the kubelet service\n[certs
] Using certificateDir folder \"/etc/kubernetes/pki\"\n[certs] Using existing ca certificate authority\n[certs] Using existing apiserver certificate and key on disk\n[certs
```

QUESTION 5

A Role bound to a Pod\\'s ServiceAccount grants overly permissive permissions. Complete the following tasks to reduce the set of permissions.



You **must** complete this task on the following cluster/nodes:

Cluster	Master node	Worker node
KSCH00201	ksch00201-master	ksch00201-worker1

You can switch the cluster/configuration context using the following command:

```
[candidate@cli] $ kubectl config use-context KSCH00201
```

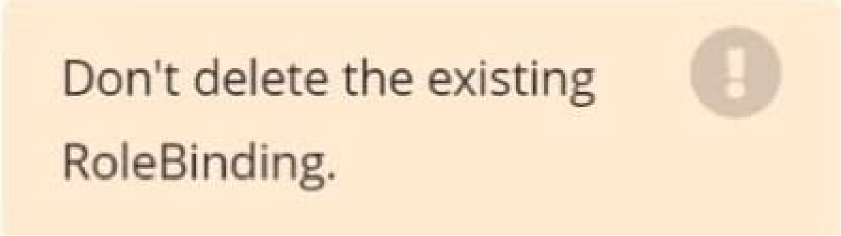
Task

Given an existing Pod named web-pod running in the namespace security.

Edit the existing Role bound to the Pod\\'s ServiceAccount sa-dev-1 to only allow performing watch operations, only on resources of type services.

Create a new Role named role-2 in the namespace security, which only allows performing update operations, only on resources of type namespaces.

Create a new RoleBinding named role-2-binding binding the newly created Role to the Pod\\'s ServiceAccount.

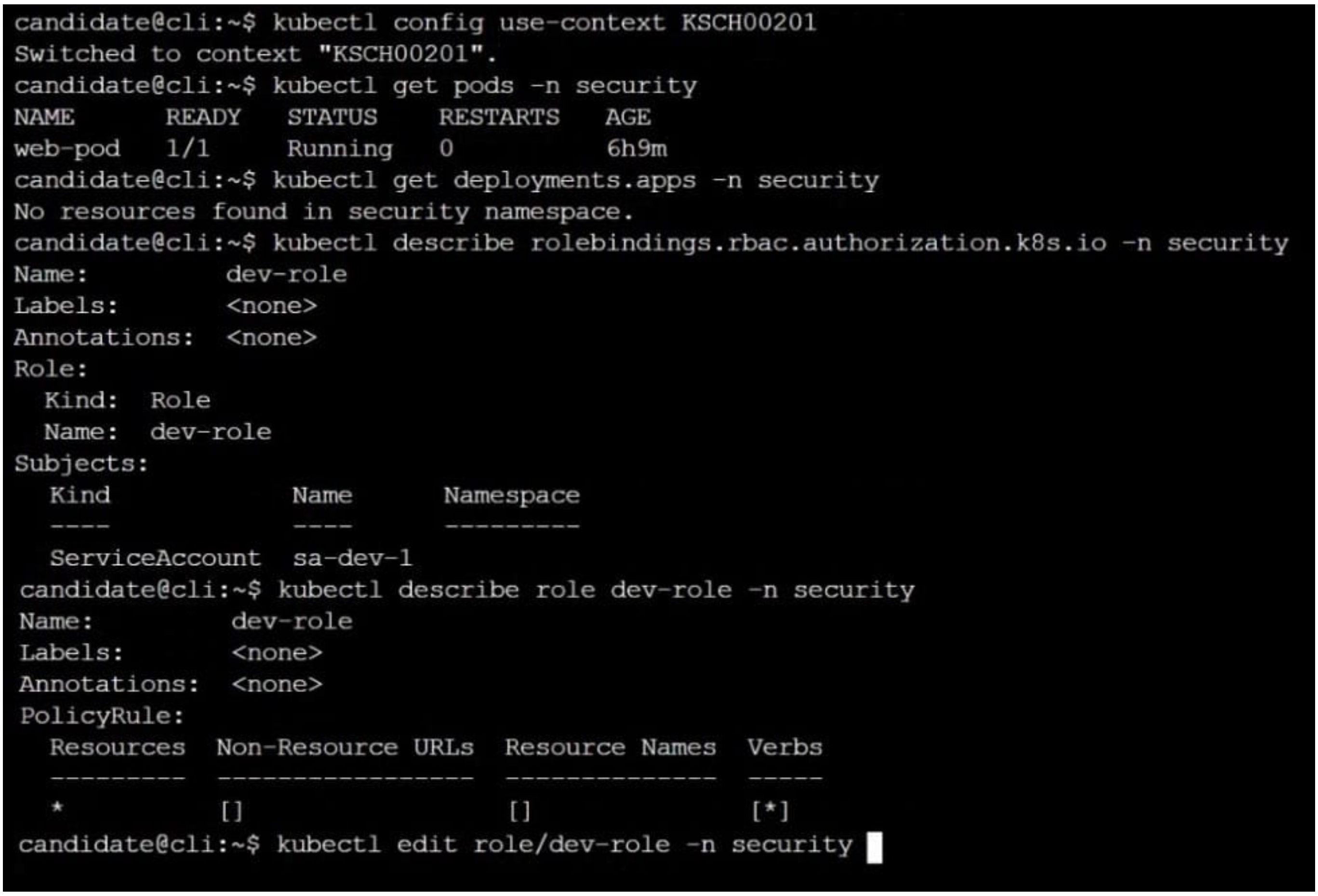


Don't delete the existing RoleBinding.

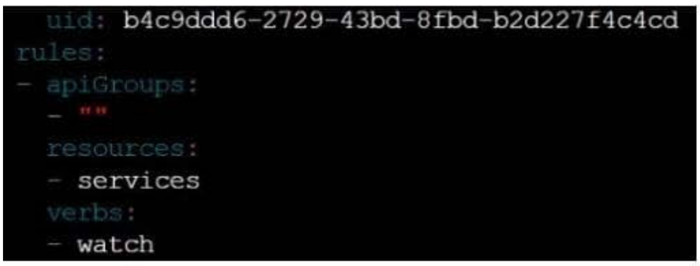
A. See the explanation below

B. PlaceHolder

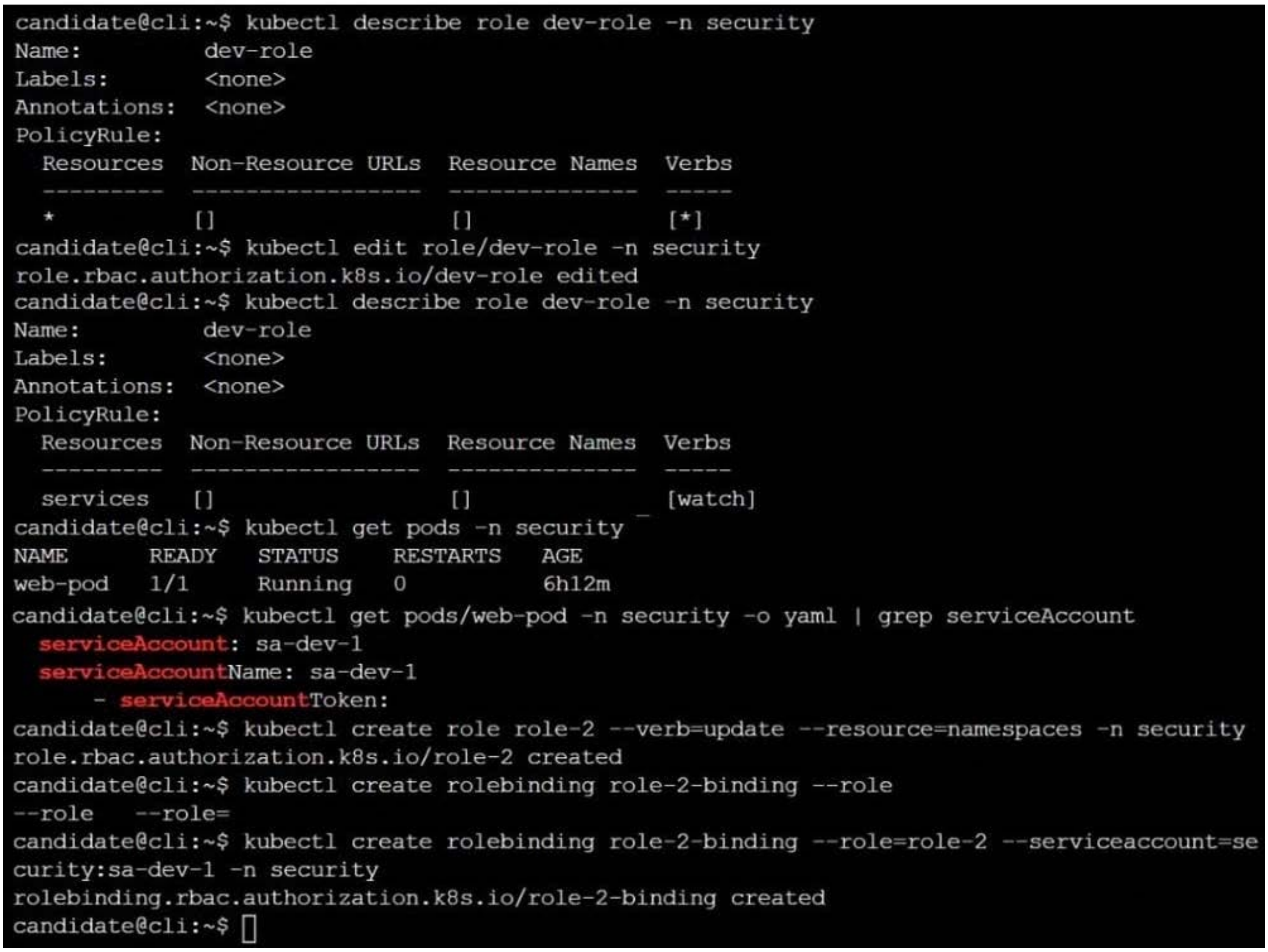
Correct Answer: A



```
candidate@cli:~$ kubectl config use-context KSCH00201
Switched to context "KSCH00201".
candidate@cli:~$ kubectl get pods -n security
NAME       READY   STATUS     RESTARTS   AGE
web-pod    1/1     Running    0          6h9m
candidate@cli:~$ kubectl get deployments.apps -n security
No resources found in security namespace.
candidate@cli:~$ kubectl describe rolebindings.rbac.authorization.k8s.io -n security
Name:         dev-role
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  Role
  Name:  dev-role
Subjects:
  Kind            Name      Namespace
  ----            ----      ---------
  ServiceAccount  sa-dev-1
candidate@cli:~$ kubectl describe role dev-role -n security
Name:         dev-role
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources  Non-Resource URLs  Resource Names  Verbs
  ---------  -----------------  --------------  -----
  *          []                 []              [*]
candidate@cli:~$ kubectl edit role/dev-role -n security
```

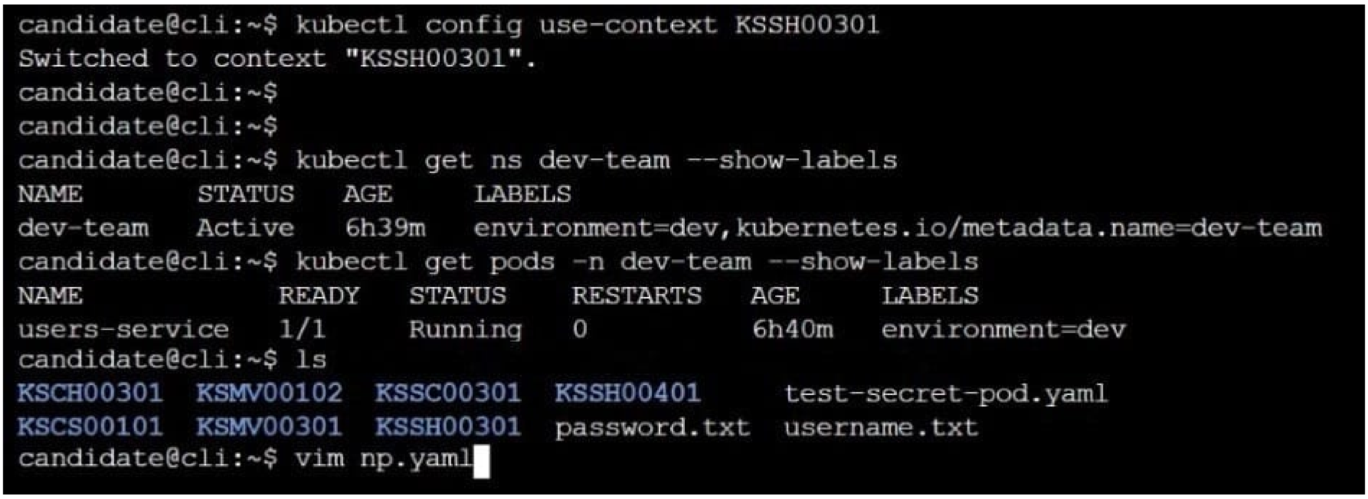


```
  uid: b4c9ddd6-2729-43bd-8fbd-b2d227f4c4cd
rules:
- apiGroups:
  - ""
  resources:
  - services
  verbs:
  - watch
```

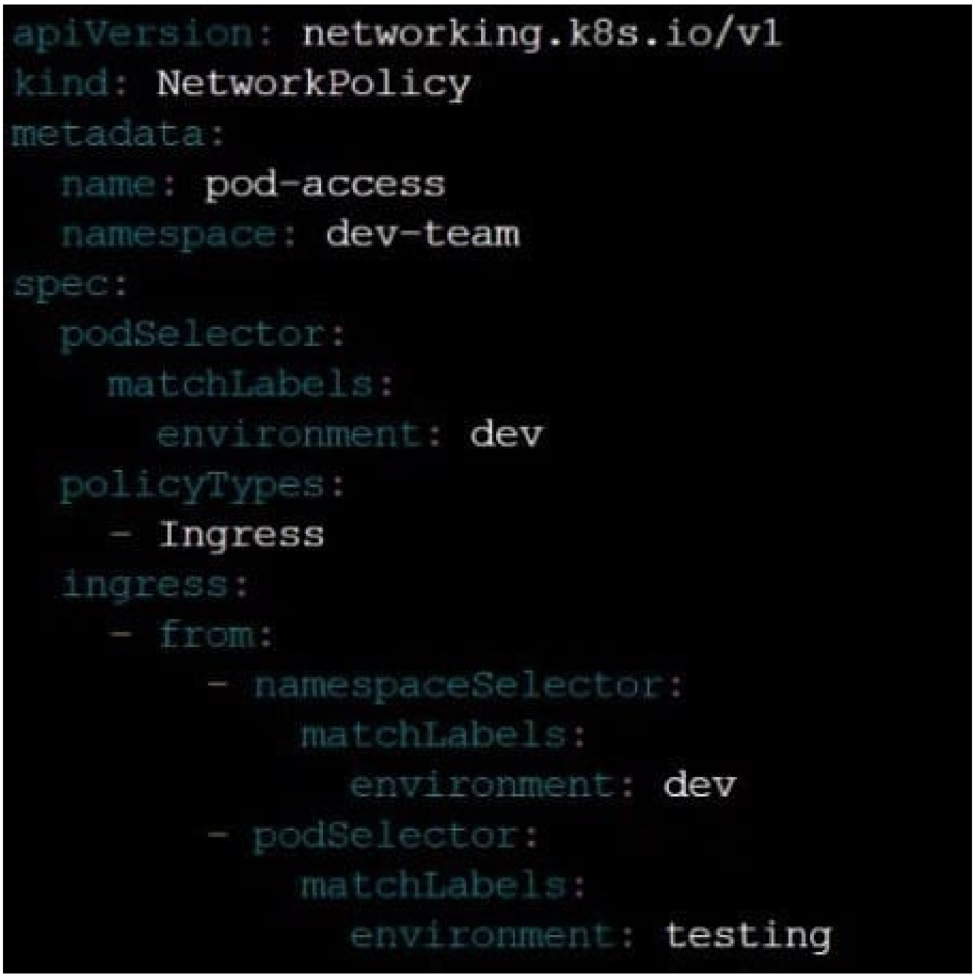


```
candidate@cli:~$ kubectl describe role dev-role -n security
Name:         dev-role
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources  Non-Resource URLs  Resource Names  Verbs
  ---------  -----------------  --------------  -----
  *          []                 []              [*]
candidate@cli:~$ kubectl edit role/dev-role -n security
role.rbac.authorization.k8s.io/dev-role edited
candidate@cli:~$ kubectl describe role dev-role -n security
Name:         dev-role
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources  Non-Resource URLs  Resource Names  Verbs
  ---------  -----------------  --------------  -----
  services   []                 []              [watch]
candidate@cli:~$ kubectl get pods -n security
NAME     READY   STATUS    RESTARTS   AGE
web-pod  1/1     Running   0          6h12m
candidate@cli:~$ kubectl get pods/web-pod -n security -o yaml | grep serviceAccount
  serviceAccount: sa-dev-1
  serviceAccountName: sa-dev-1
      - serviceAccountToken:
candidate@cli:~$ kubectl create role role-2 --verb=update --resource=namespaces -n security
role.rbac.authorization.k8s.io/role-2 created
candidate@cli:~$ kubectl create rolebinding role-2-binding --role
--role   --role=
candidate@cli:~$ kubectl create rolebinding role-2-binding --role=role-2 --serviceaccount=se
curity:sa-dev-1 -n security
rolebinding.rbac.authorization.k8s.io/role-2-binding created
candidate@cli:~$ 
```

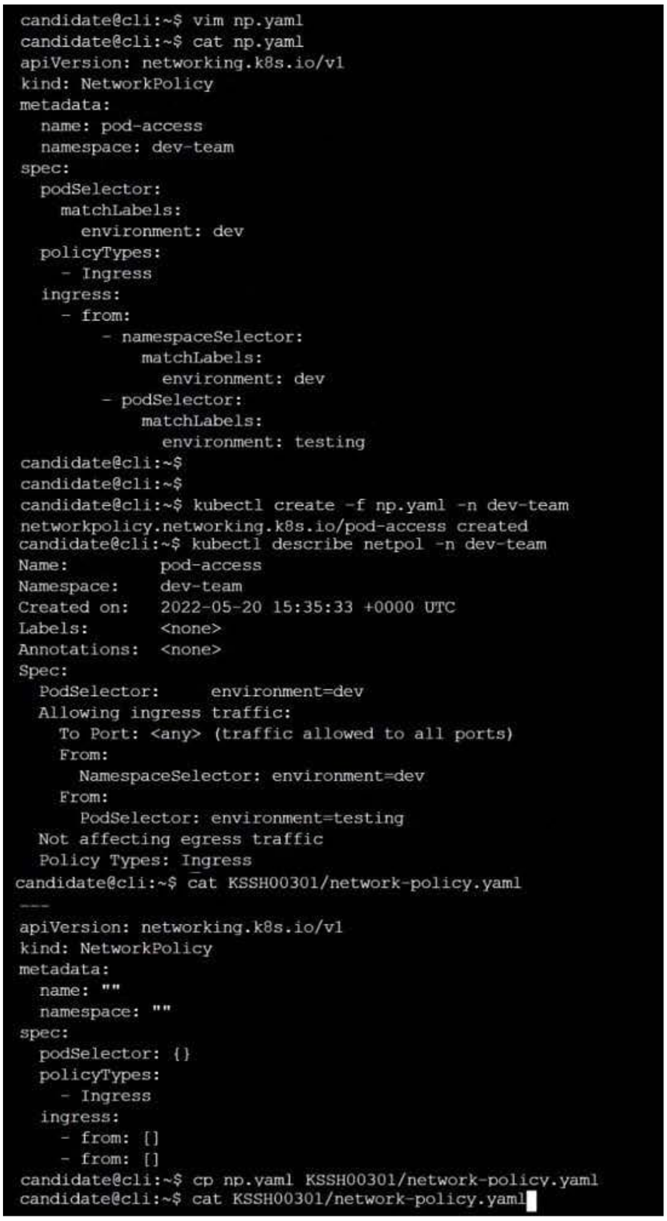
QUESTION 6



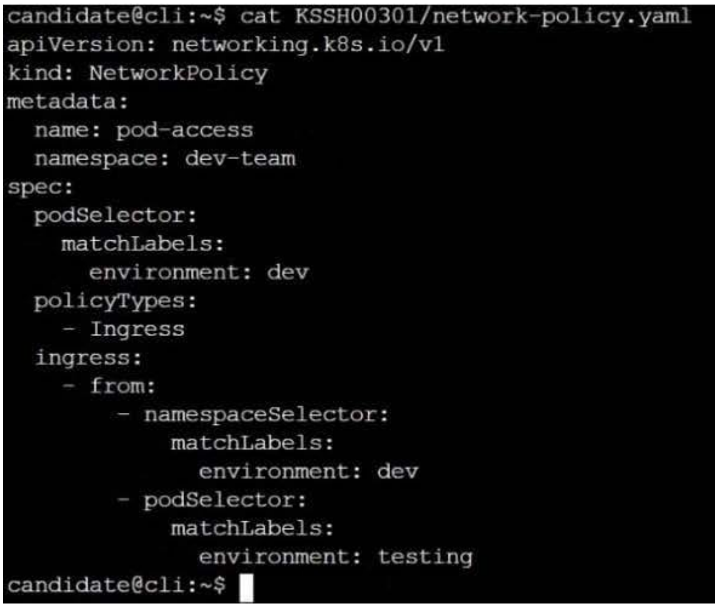
```
candidate@cli:~$ kubectl config use-context KSSH00301
Switched to context "KSSH00301".
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ kubectl get ns dev-team --show-labels
NAME       STATUS   AGE     LABELS
dev-team   Active   6h39m   environment=dev,kubernetes.io/metadata.name=dev-team
candidate@cli:~$ kubectl get pods -n dev-team --show-labels
NAME             READY   STATUS    RESTARTS   AGE     LABELS
users-service    1/1     Running   0          6h40m   environment=dev
candidate@cli:~$ ls
KSCH00301  KSMV00102  KSSC00301  KSSH00401     test-secret-pod.yaml
KSCS00101  KSMV00301  KSSH00301  password.txt  username.txt
candidate@cli:~$ vim np.yaml
```



```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: pod-access
  namespace: dev-team
spec:
  podSelector:
    matchLabels:
      environment: dev
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              environment: dev
        - podSelector:
            matchLabels:
              environment: testing
```



```
candidate@cli:~$ vim np.yaml
candidate@cli:~$ cat np.yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: pod-access
  namespace: dev-team
spec:
  podSelector:
    matchLabels:
      environment: dev
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              environment: dev
        - podSelector:
            matchLabels:
              environment: testing
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ kubectl create -f np.yaml -n dev-team
networkpolicy.networking.k8s.io/pod-access created
candidate@cli:~$ kubectl describe netpol -n dev-team
Name:         pod-access
Namespace:    dev-team
Created on:   2022-05-20 15:35:33 +0000 UTC
Labels:       <none>
Annotations:  <none>
Spec:
  PodSelector:     environment=dev
  Allowing ingress traffic:
    To Port: <any> (traffic allowed to all ports)
    From:
      NamespaceSelector: environment=dev
    From:
      PodSelector: environment=testing
  Not affecting egress traffic
  Policy Types: Ingress
candidate@cli:~$ cat KSSH00301/network-policy.yaml
---
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: ""
  namespace: ""
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress:
    - from: []
    - from: []
candidate@cli:~$ cp np.yaml KSSH00301/network-policy.yaml
candidate@cli:~$ cat KSSH00301/network-policy.yaml
```



```
candidate@cli:~$ cat KSSH00301/network-policy.yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: pod-access
  namespace: dev-team
spec:
  podSelector:
    matchLabels:
      environment: dev
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              environment: dev
        - podSelector:
            matchLabels:
              environment: testing
candidate@cli:~$
```

1.

Retrieve the content of the existing secret named default-token-xxxxx in the testing namespace.

Store the value of the token in the token.txt

2.

Create a new secret named test-db-secret in the DB namespace with the following content:

username: mysql password: password@123

Create the Pod name test-db-pod of image nginx in the namespace db that can access test-db-secret via a volume at path /etc/mysql-credentials

A. See the explanation below:

B. PlaceHolder

Correct Answer: A

To add a Kubernetes cluster to your project, group, or instance:

1.

Navigate to your:

2.

Click Add Kubernetes cluster.

3.

Click the Add existing cluster tab and fill in the details:

Get the API URL by running this command:

```
kubectl cluster-info | grep -E \\'Kubernetes master|Kubernetes control plane\\' | awk \\'/http/ {print $NF}\\'
```

uk.co.certification.simulator.questionpool.PList@113e1f90

```
kubectl get secret -o jsonpath="{[\\'data\\'][\\'ca\.crt\\']}"
```

QUESTION 7

Cluster: scanner

Master node: controlplane

Worker node: worker1

You can switch the cluster/configuration context using the following command:

```
[desk@cli] $ kubectl config use-context scanner
```

Given:

You may use Trivy\\'s documentation.

Task:

Use the Trivy open-source container scanner to detect images with severe vulnerabilities used by Pods in the namespace nato.

Look for images with High or Critical severity vulnerabilities and delete the Pods that use those images.

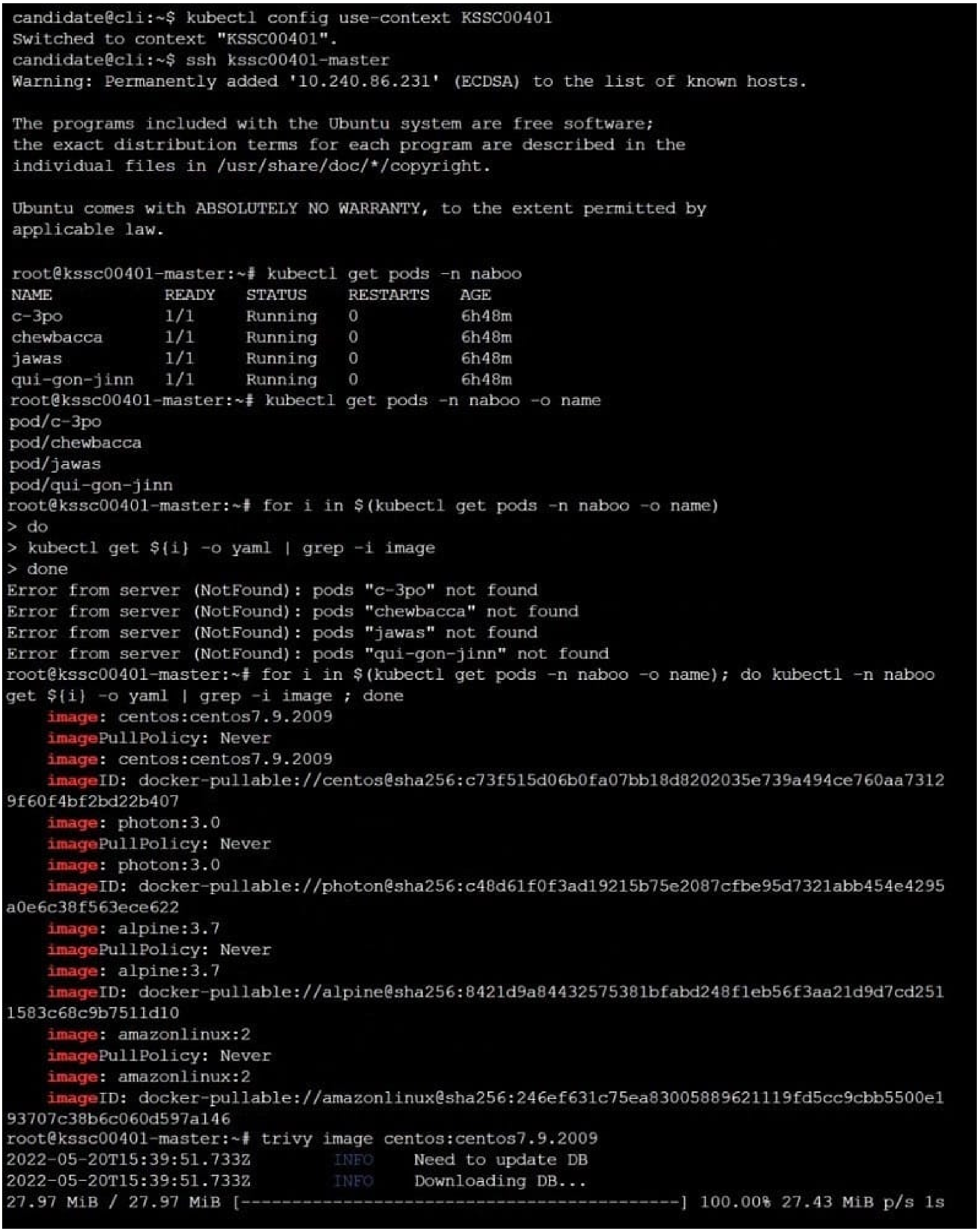
Trivy is pre-installed on the cluster\\'s master node. Use cluster\\'s master node to use Trivy.

A. See the explanation below

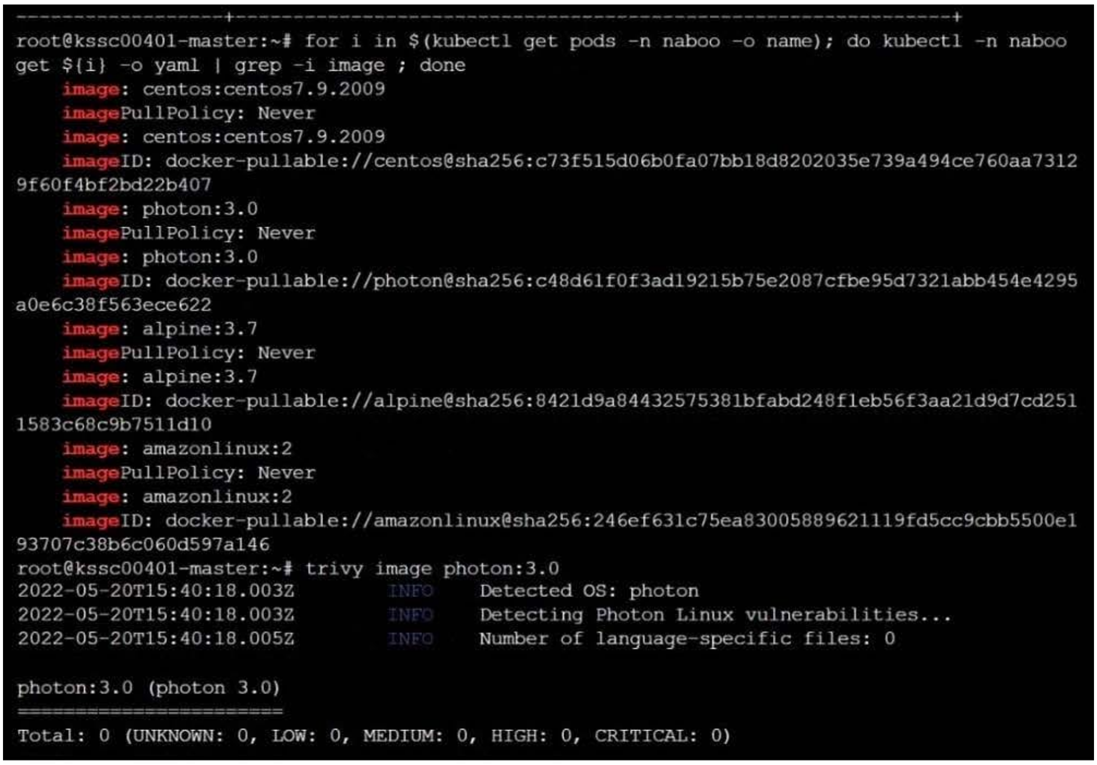
B. PlaceHolder

Correct Answer: A

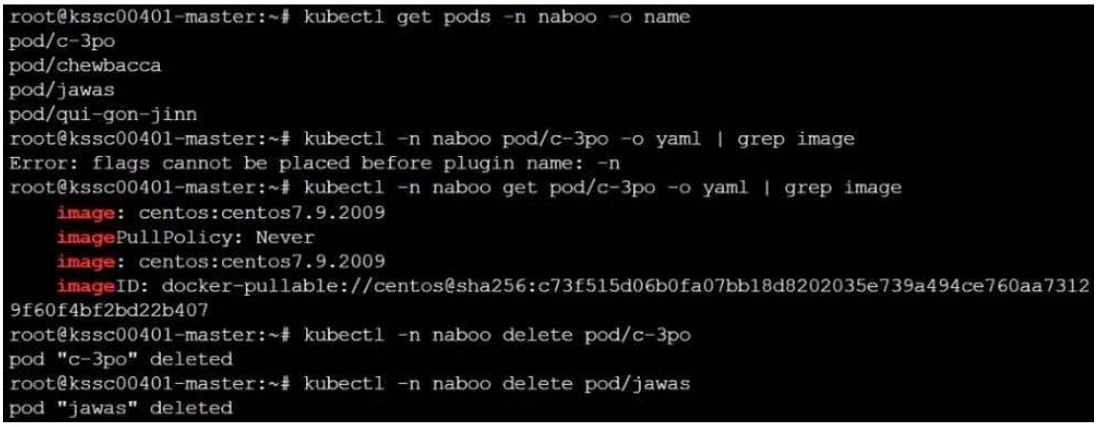
QUESTION 8



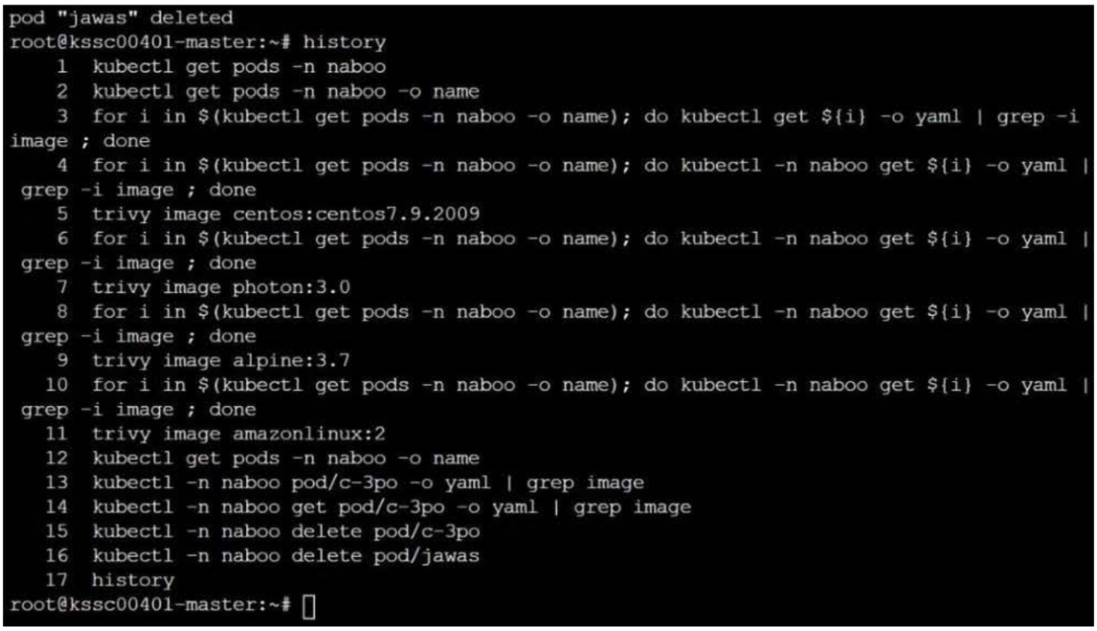
```
candidate@cli:~$ kubectl config use-context KSSC00401
Switched to context "KSSC00401".
candidate@cli:~$ ssh kssc00401-master
Warning: Permanently added '10.240.86.231' (ECDSA) to the list of known hosts.

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.

root@kssc00401-master:~# kubectl get pods -n naboo
NAME           READY   STATUS    RESTARTS   AGE
c-3po          1/1     Running   0          6h48m
chewbacca      1/1     Running   0          6h48m
jawas          1/1     Running   0          6h48m
qui-gon-jinn   1/1     Running   0          6h48m
root@kssc00401-master:~# kubectl get pods -n naboo -o name
pod/c-3po
pod/chewbacca
pod/jawas
pod/qui-gon-jinn
root@kssc00401-master:~# for i in $(kubectl get pods -n naboo -o name)
> do
> kubectl get ${i} -o yaml | grep -i image
> done
Error from server (NotFound): pods "c-3po" not found
Error from server (NotFound): pods "chewbacca" not found
Error from server (NotFound): pods "jawas" not found
Error from server (NotFound): pods "qui-gon-jinn" not found
root@kssc00401-master:~# for i in $(kubectl get pods -n naboo -o name); do kubectl -n naboo
get ${i} -o yaml | grep -i image ; done
    image: centos:centos7.9.2009
    imagePullPolicy: Never
    image: centos:centos7.9.2009
    imageID: docker-pullable://centos@sha256:c73f515d06b0fa07bb18d8202035e739a494ce760aa7312
9f60f4bf2bd22b407
    image: photon:3.0
    imagePullPolicy: Never
    image: photon:3.0
    imageID: docker-pullable://photon@sha256:c48d61f0f3ad19215b75e2087cfbe95d7321abb454e4295
a0e6c38f563ece622
    image: alpine:3.7
    imagePullPolicy: Never
    image: alpine:3.7
    imageID: docker-pullable://alpine@sha256:8421d9a84432575381bfabd248f1eb56f3aa21d9d7cd251
1583c68c9b7511d10
    image: amazonlinux:2
    imagePullPolicy: Never
    image: amazonlinux:2
    imageID: docker-pullable://amazonlinux@sha256:246ef631c75ea83005889621119fd5cc9cbb5500e1
93707c38b6c060d597a146
root@kssc00401-master:~# trivy image centos:centos7.9.2009
2022-05-20T15:39:51.733Z        INFO    Need to update DB
2022-05-20T15:39:51.733Z        INFO    Downloading DB...
27.97 MiB / 27.97 MiB [--------------------------------------------] 100.00% 27.43 MiB p/s 1s
```



```
------------------+----------------------------------------------------------------+
root@kssc00401-master:~# for i in $(kubectl get pods -n naboo -o name); do kubectl -n naboo
get ${i} -o yaml | grep -i image ; done
    image: centos:centos7.9.2009
    imagePullPolicy: Never
    image: centos:centos7.9.2009
    imageID: docker-pullable://centos@sha256:c73f515d06b0fa07bb18d8202035e739a494ce760aa7312
9f60f4bf2bd22b407
    image: photon:3.0
    imagePullPolicy: Never
    image: photon:3.0
    imageID: docker-pullable://photon@sha256:c48d61f0f3ad19215b75e2087cfbe95d7321abb454e4295
a0e6c38f563ece622
    image: alpine:3.7
    imagePullPolicy: Never
    image: alpine:3.7
    imageID: docker-pullable://alpine@sha256:8421d9a84432575381bfabd248f1eb56f3aa21d9d7cd251
1583c68c9b7511d10
    image: amazonlinux:2
    imagePullPolicy: Never
    image: amazonlinux:2
    imageID: docker-pullable://amazonlinux@sha256:246ef631c75ea83005889621119fd5cc9cbb5500e1
93707c38b6c060d597a146
root@kssc00401-master:~# trivy image photon:3.0
2022-05-20T15:40:18.003Z        INFO    Detected OS: photon
2022-05-20T15:40:18.003Z        INFO    Detecting Photon Linux vulnerabilities...
2022-05-20T15:40:18.005Z        INFO    Number of language-specific files: 0

photon:3.0 (photon 3.0)
=======================
Total: 0 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 0, CRITICAL: 0)
```

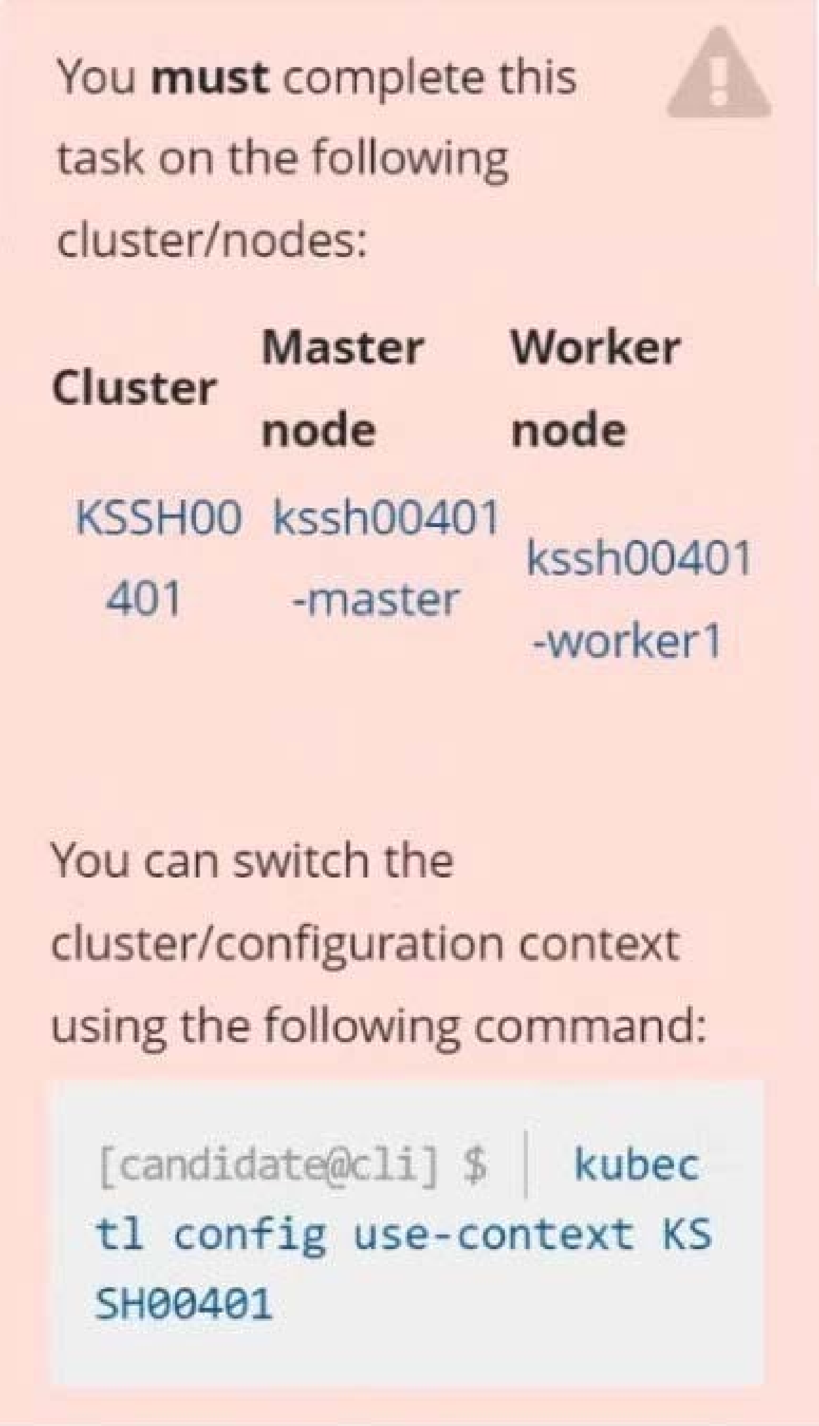


```
root@kssc00401-master:~# kubectl get pods -n naboo -o name
pod/c-3po
pod/chewbacca
pod/jawas
pod/qui-gon-jinn
root@kssc00401-master:~# kubectl -n naboo pod/c-3po -o yaml | grep image
Error: flags cannot be placed before plugin name: -n
root@kssc00401-master:~# kubectl -n naboo get pod/c-3po -o yaml | grep image
    image: centos:centos7.9.2009
    imagePullPolicy: Never
    image: centos:centos7.9.2009
    imageID: docker-pullable://centos@sha256:c73f515d06b0fa07bb18d8202035e739a494ce760aa7312
9f60f4bf2bd22b407
root@kssc00401-master:~# kubectl -n naboo delete pod/c-3po
pod "c-3po" deleted
root@kssc00401-master:~# kubectl -n naboo delete pod/jawas
pod "jawas" deleted
```



```
pod "jawas" deleted
root@kssc00401-master:~# history
    1  kubectl get pods -n naboo
    2  kubectl get pods -n naboo -o name
    3  for i in $(kubectl get pods -n naboo -o name); do kubectl get ${i} -o yaml | grep -i
image ; done
    4  for i in $(kubectl get pods -n naboo -o name); do kubectl -n naboo get ${i} -o yaml |
 grep -i image ; done
    5  trivy image centos:centos7.9.2009
    6  for i in $(kubectl get pods -n naboo -o name); do kubectl -n naboo get ${i} -o yaml |
 grep -i image ; done
    7  trivy image photon:3.0
    8  for i in $(kubectl get pods -n naboo -o name); do kubectl -n naboo get ${i} -o yaml |
 grep -i image ; done
    9  trivy image alpine:3.7
   10  for i in $(kubectl get pods -n naboo -o name); do kubectl -n naboo get ${i} -o yaml |
 grep -i image ; done
   11  trivy image amazonlinux:2
   12  kubectl get pods -n naboo -o name
   13  kubectl -n naboo pod/c-3po -o yaml | grep image
   14  kubectl -n naboo get pod/c-3po -o yaml | grep image
   15  kubectl -n naboo delete pod/c-3po
   16  kubectl -n naboo delete pod/jawas
   17  history
root@kssc00401-master:~# 
```

AppArmor is enabled on the cluster\\'s worker node. An AppArmor profile is prepared, but not enforced yet.

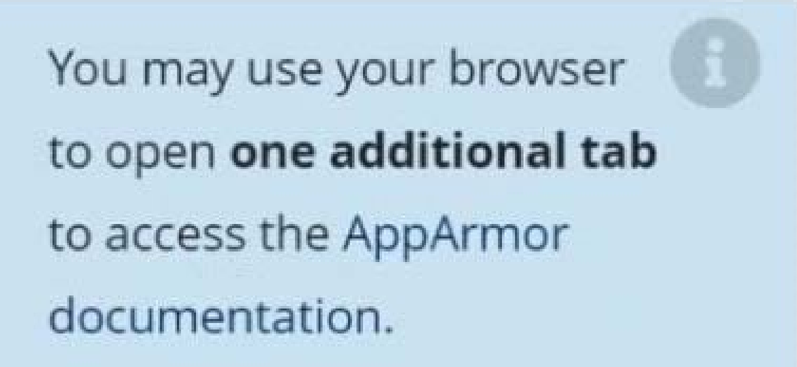


You **must** complete this task on the following cluster/nodes:

Cluster	Master node	Worker node
KSSH00401	kssh00401-master	kssh00401-worker1

You can switch the cluster/configuration context using the following command:

```
[candidate@cli] $ kubectl config use-context KSSH00401
```



You may use your browser to open **one additional tab** to access the AppArmor documentation.

Task

On the cluster\\'s worker node, enforce the prepared AppArmor profile located at /etc/apparmor.d/nginx_apparmor.

Edit the prepared manifest file located at /home/candidate/KSSH00401/nginx-pod.yaml to apply the AppArmor profile.

Finally, apply the manifest file and create the Pod specified in it.

A. See the explanation below

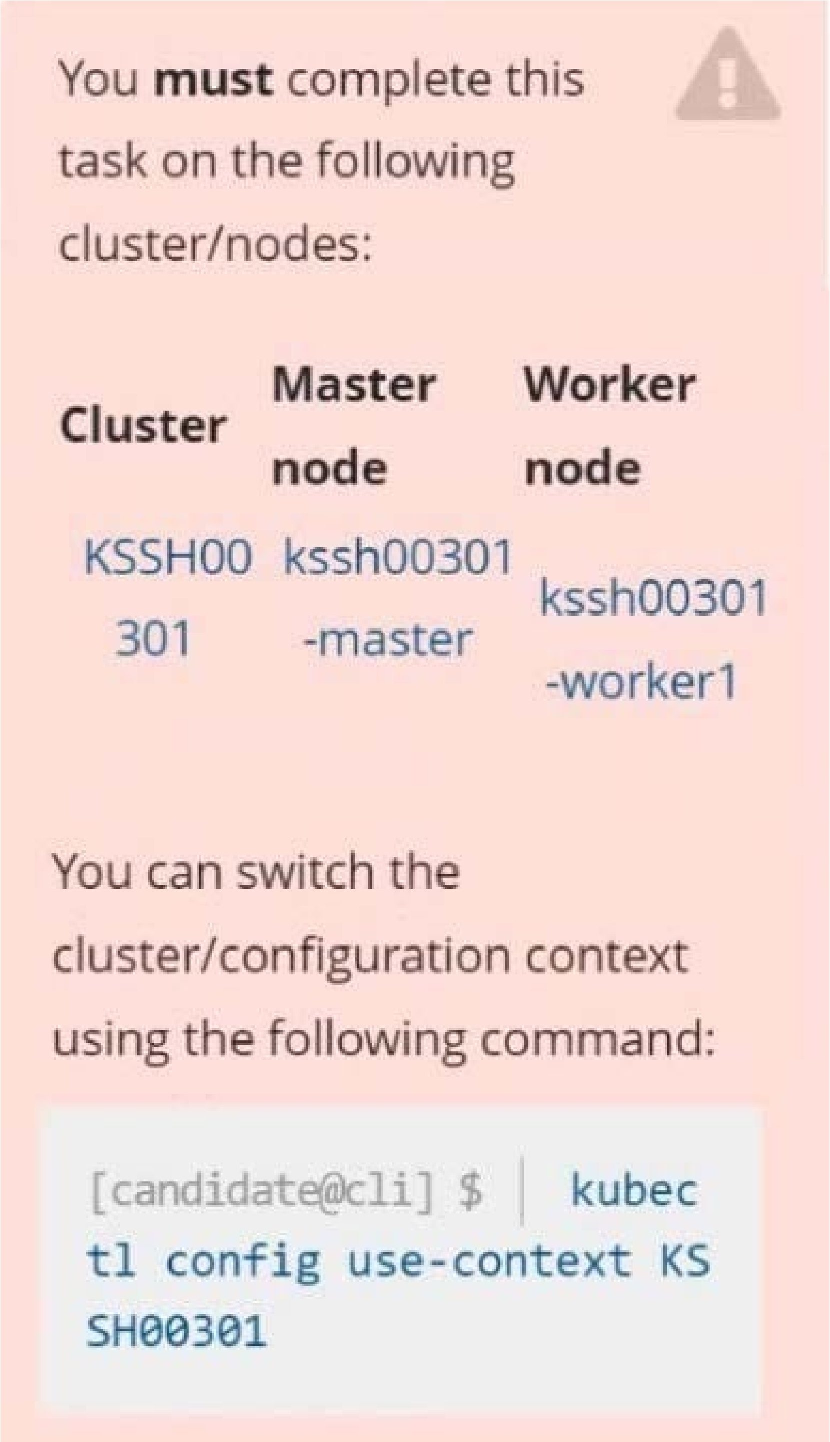
B. PlaceHolder

Correct Answer: A

QUESTION 9

CORRECT TEXT

Task



You **must** complete this task on the following cluster/nodes:

Cluster	Master node	Worker node
KSSH00301	kssh00301-master	kssh00301-worker1

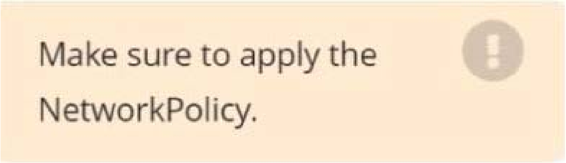
You can switch the cluster/configuration context using the following command:

```
[candidate@cli] $ kubectl config use-context KSSH00301
```

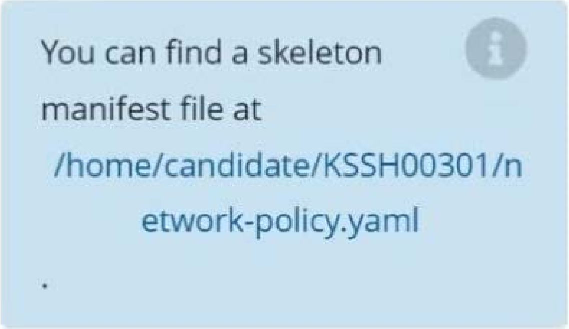
Create a NetworkPolicy named pod-access to restrict access to Pod users-service running in namespace dev-team. Only allow the following Pods to connect to Pod users-service:

1. Pods in the namespace qa

2. Pods with label environment: testing, in any namespace



Make sure to apply the NetworkPolicy.

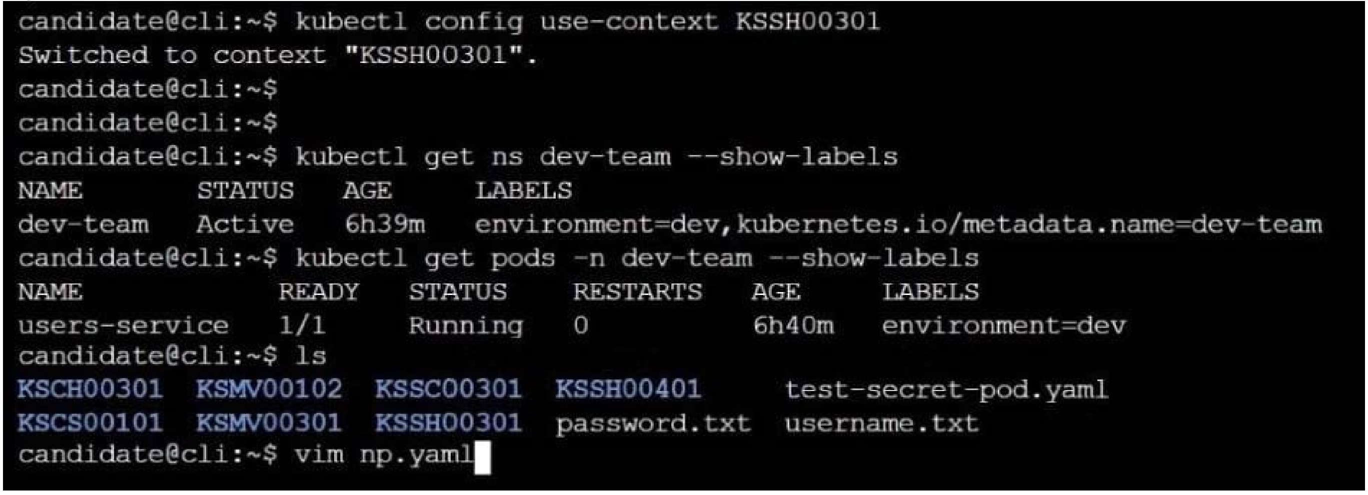


You can find a skeleton manifest file at /home/candidate/KSSH00301/network-policy.yaml
.

A. See explanation below.
B. PlaceHolder

Correct Answer: A
Explanation

Explanation/Reference:



```
candidate@cli:~$ kubectl config use-context KSSH00301
Switched to context "KSSH00301".
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ kubectl get ns dev-team --show-labels
NAME        STATUS   AGE     LABELS
dev-team    Active   6h39m   environment=dev,kubernetes.io/metadata.name=dev-team
candidate@cli:~$ kubectl get pods -n dev-team --show-labels
NAME            READY   STATUS    RESTARTS   AGE     LABELS
users-service   1/1     Running   0          6h40m   environment=dev
candidate@cli:~$ ls
KSCH00301  KSMV00102  KSSC00301  KSSH00401    test-secret-pod.yaml
KSCS00101  KSMV00301  KSSH00301  password.txt  username.txt
candidate@cli:~$ vim np.yaml
```

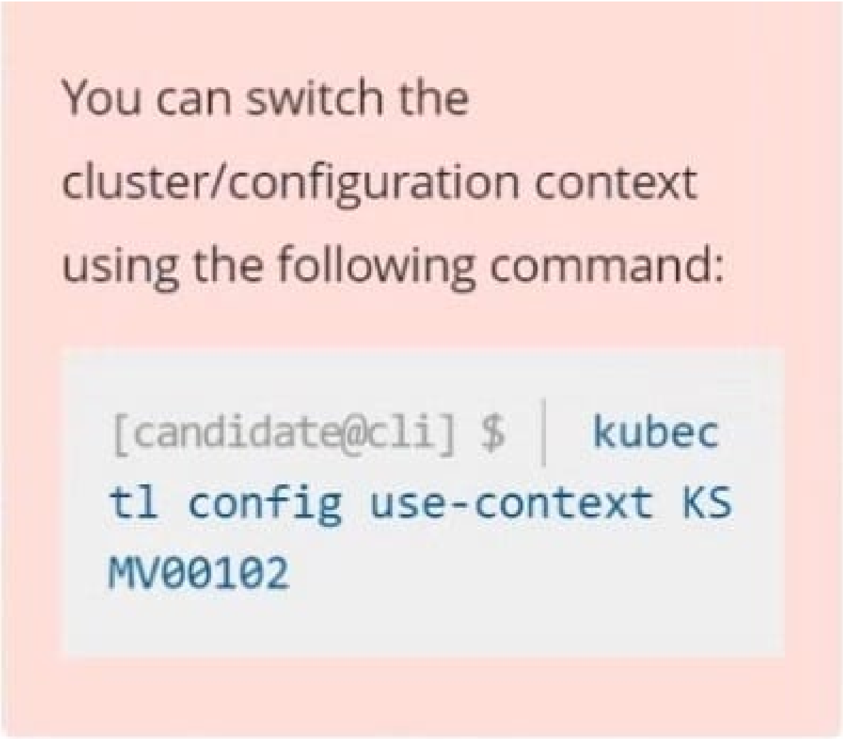
A. See explanation below.

B. PlaceHolder

Correct Answer: A

QUESTION 10

CORRECT TEXT



You can switch the cluster/configuration context using the following command:

```
[candidate@cli] $ kubectl config use-context KSMV00102
```

A PodSecurityPolicy shall prevent the creation of privileged Pods in a specific namespace.

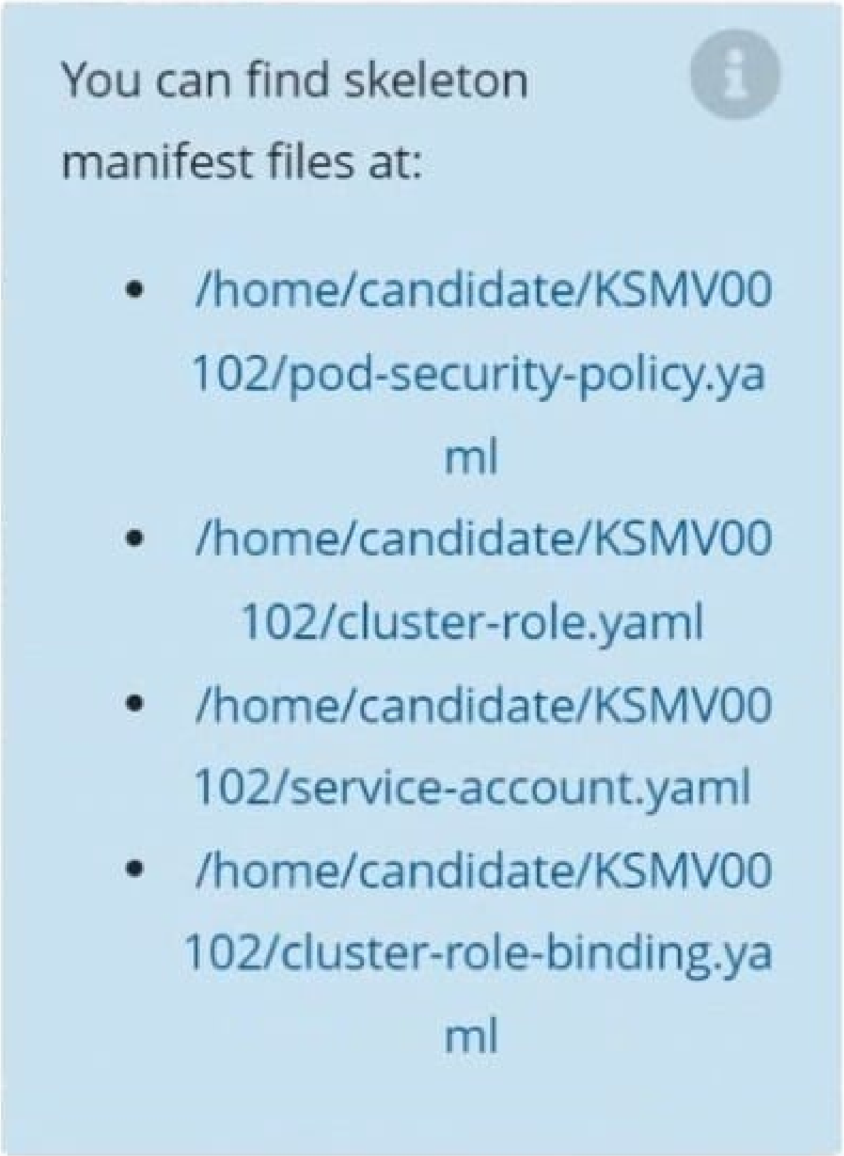
Task

Create a new PodSecurityPolicy named prevent-psp-policy,which prevents the creation of privileged Pods.

Create a new ClusterRole named restrict-access-role, which uses the newly created PodSecurityPolicy prevent-psp-policy.

Create a new ServiceAccount named psp-restrict-sa in the existing namespace staging.

Finally, create a new ClusterRoleBinding named restrict-access-bind, which binds the newly created ClusterRole restrict-access-role to the newly created ServiceAccount psp- restrict-sa.



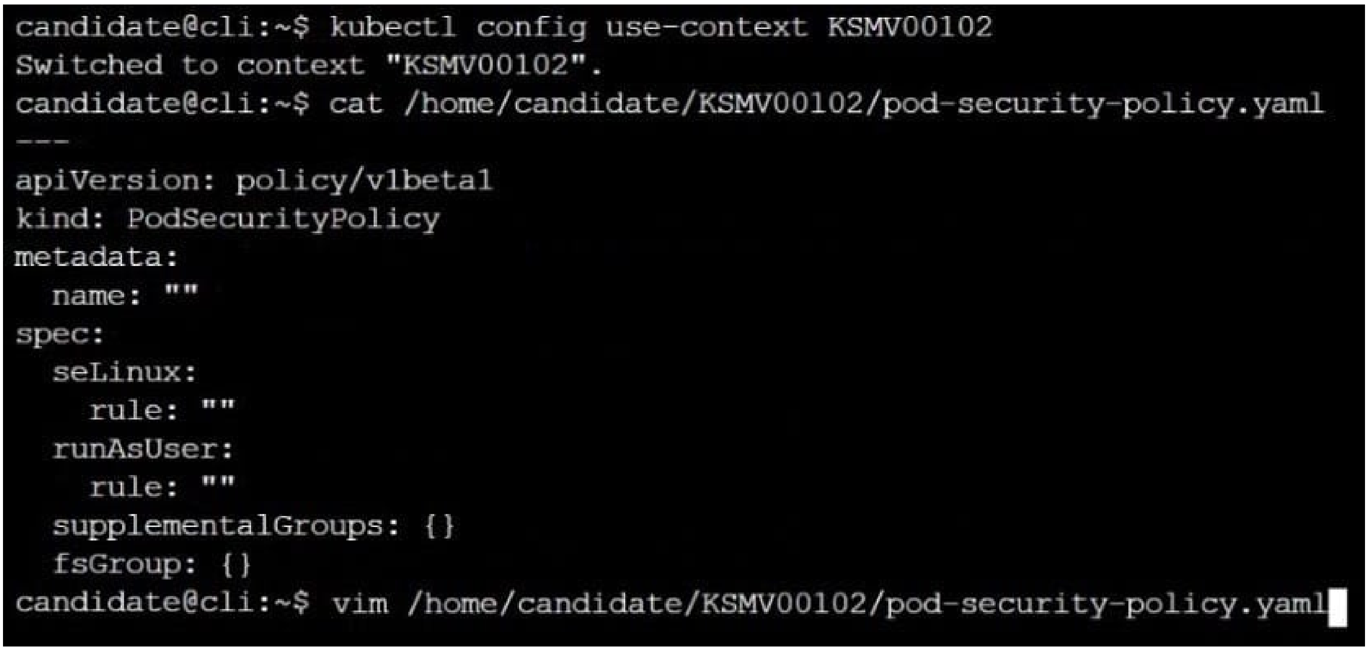
You can find skeleton manifest files at:

- /home/candidate/KSMV00102/pod-security-policy.yaml
- /home/candidate/KSMV00102/cluster-role.yaml
- /home/candidate/KSMV00102/service-account.yaml
- /home/candidate/KSMV00102/cluster-role-binding.yaml

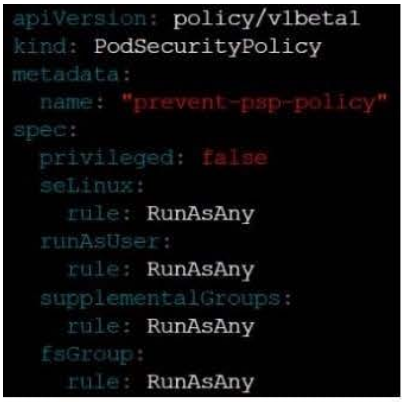
A. See explanation below.

B. PlaceHolder

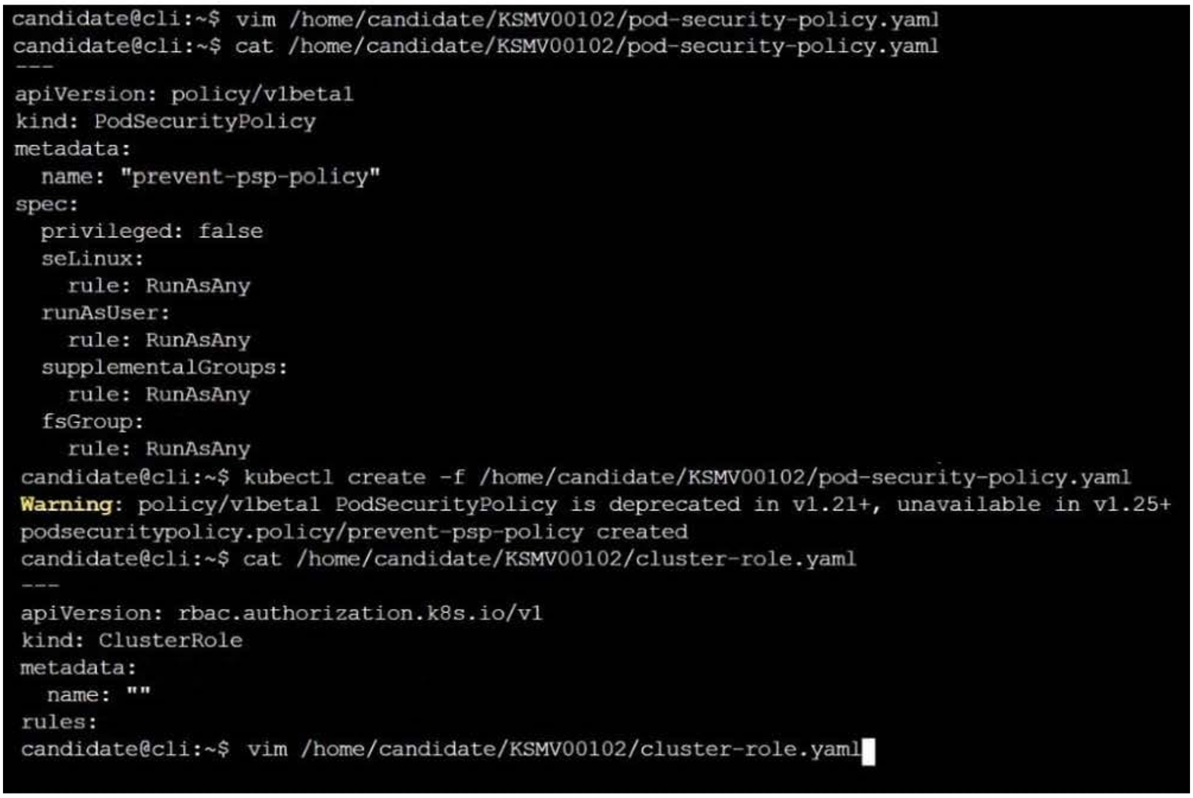
Correct Answer: A



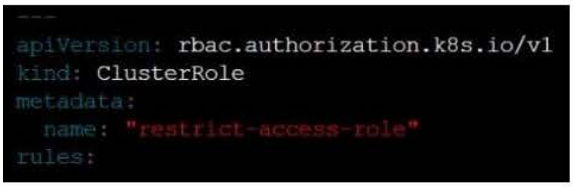
```
candidate@cli:~$ kubectl config use-context KSMV00102
Switched to context "KSMV00102".
candidate@cli:~$ cat /home/candidate/KSMV00102/pod-security-policy.yaml
---
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: ""
spec:
  seLinux:
    rule: ""
  runAsUser:
    rule: ""
  supplementalGroups: {}
  fsGroup: {}
candidate@cli:~$ vim /home/candidate/KSMV00102/pod-security-policy.yaml
```



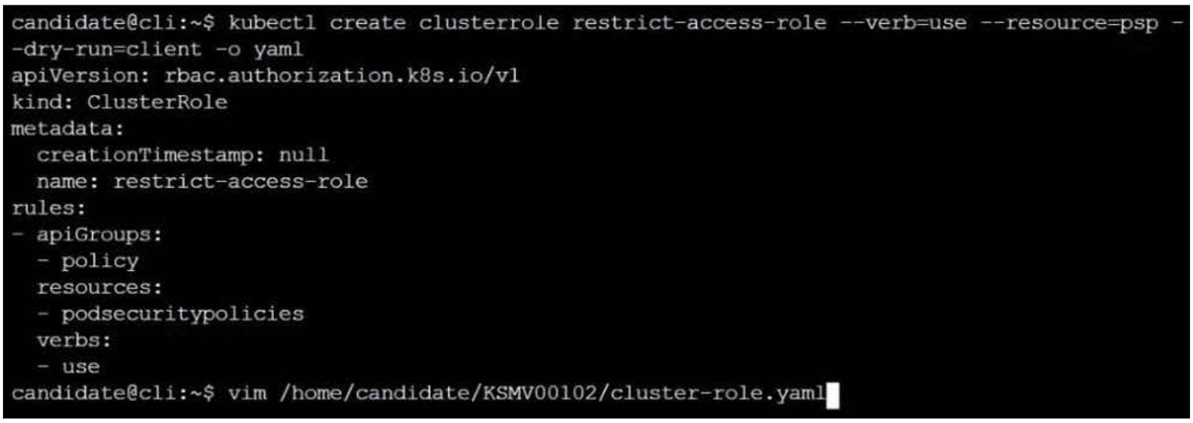
```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: "prevent-psp-policy"
spec:
  privileged: false
  seLinux:
    rule: RunAsAny
  runAsUser:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  fsGroup:
    rule: RunAsAny
```



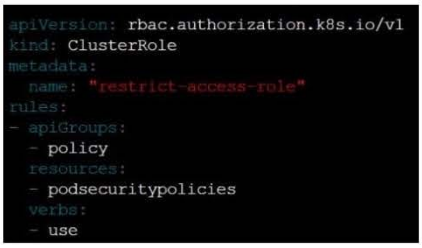
```
candidate@cli:~$ vim /home/candidate/KSMV00102/pod-security-policy.yaml
candidate@cli:~$ cat /home/candidate/KSMV00102/pod-security-policy.yaml
---
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: "prevent-psp-policy"
spec:
  privileged: false
  seLinux:
    rule: RunAsAny
  runAsUser:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  fsGroup:
    rule: RunAsAny
candidate@cli:~$ kubectl create -f /home/candidate/KSMV00102/pod-security-policy.yaml
Warning: policy/v1beta1 PodSecurityPolicy is deprecated in v1.21+, unavailable in v1.25+
podsecuritypolicy.policy/prevent-psp-policy created
candidate@cli:~$ cat /home/candidate/KSMV00102/cluster-role.yaml
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: ""
rules:
candidate@cli:~$ vim /home/candidate/KSMV00102/cluster-role.yaml
```



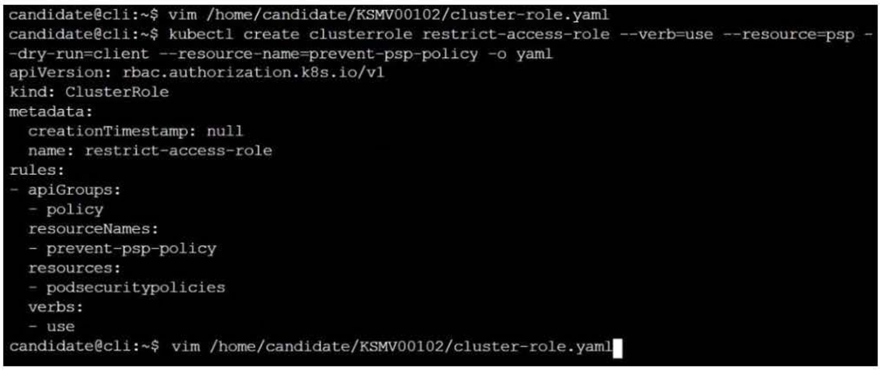
```
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: "restrict-access-role"
rules:
```



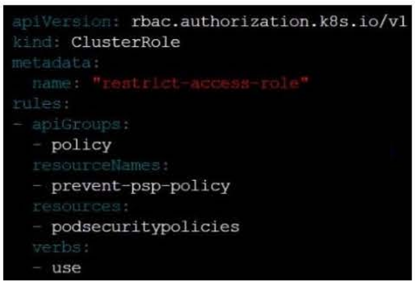
```
candidate@cli:~$ kubectl create clusterrole restrict-access-role --verb=use --resource=psp -
-dry-run=client -o yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  creationTimestamp: null
  name: restrict-access-role
rules:
- apiGroups:
  - policy
  resources:
  - podsecuritypolicies
  verbs:
  - use
candidate@cli:~$ vim /home/candidate/KSMV00102/cluster-role.yaml
```



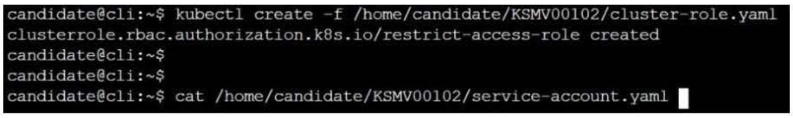
```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: "restrict-access-role"
rules:
- apiGroups:
  - policy
  resources:
  - podsecuritypolicies
  verbs:
  - use
```



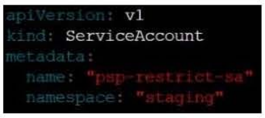
```
candidate@cli:~$ vim /home/candidate/KSMV00102/cluster-role.yaml
candidate@cli:~$ kubectl create clusterrole restrict-access-role --verb=use --resource=psp -
-dry-run=client --resource-name=prevent-psp-policy -o yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  creationTimestamp: null
  name: restrict-access-role
rules:
- apiGroups:
  - policy
  resourceNames:
  - prevent-psp-policy
  resources:
  - podsecuritypolicies
  verbs:
  - use
candidate@cli:~$ vim /home/candidate/KSMV00102/cluster-role.yaml
```



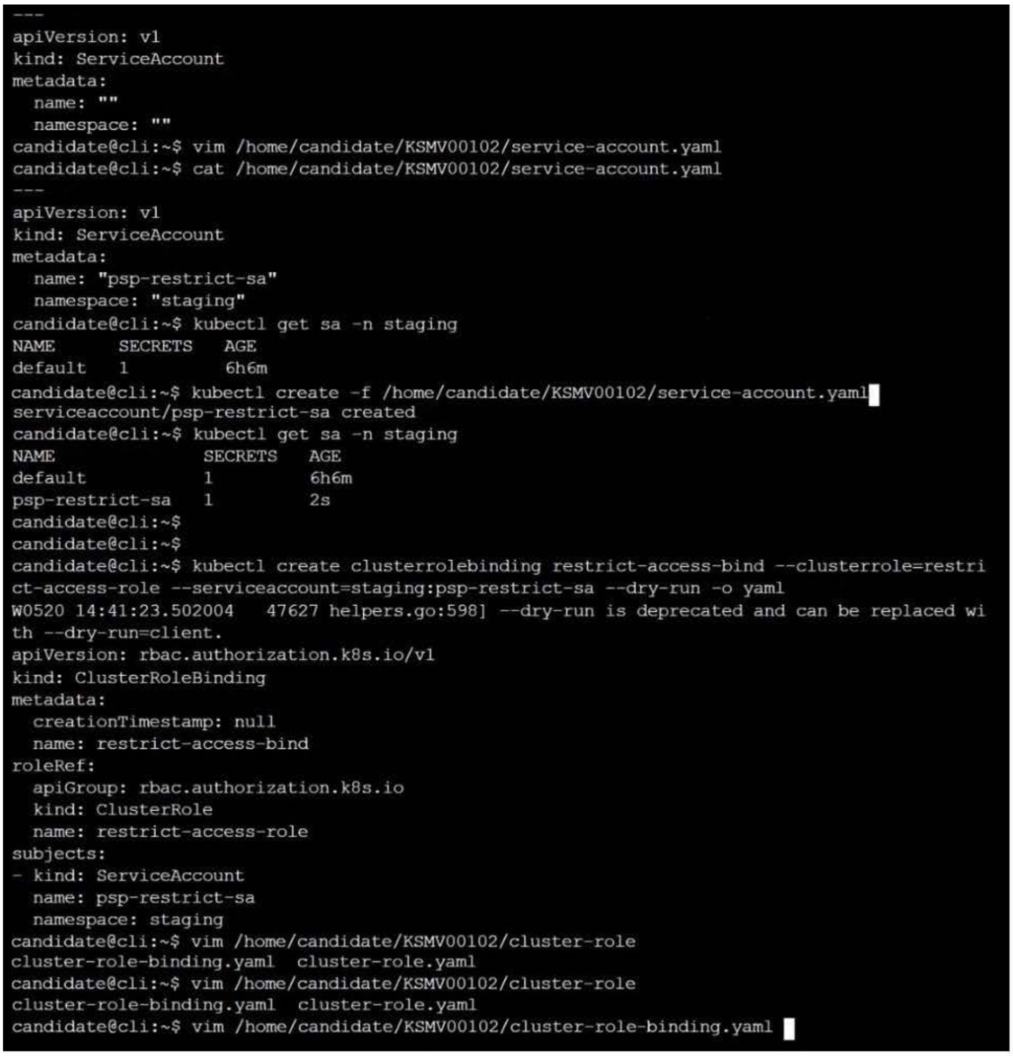
```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: "restrict-access-role"
rules:
- apiGroups:
  - policy
  resourceNames:
  - prevent-psp-policy
  resources:
  - podsecuritypolicies
  verbs:
  - use
```



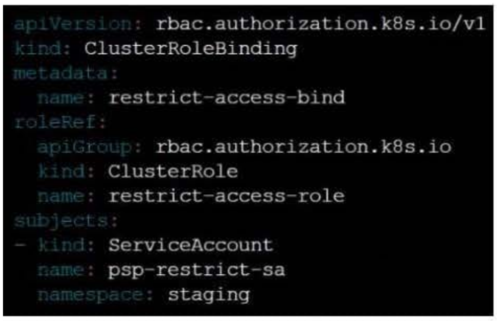
```
candidate@cli:~$ kubectl create -f /home/candidate/KSMV00102/cluster-role.yaml
clusterrole.rbac.authorization.k8s.io/restrict-access-role created
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ cat /home/candidate/KSMV00102/service-account.yaml
```



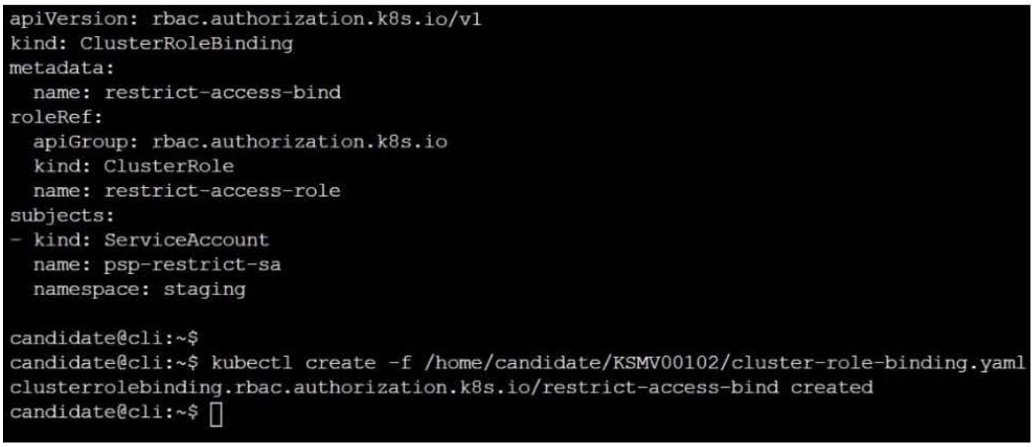
```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: "psp-restrict-sa"
  namespace: "staging"
```



```
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: ""
  namespace: ""
candidate@cli:~$ vim /home/candidate/KSMV00102/service-account.yaml
candidate@cli:~$ cat /home/candidate/KSMV00102/service-account.yaml
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: "psp-restrict-sa"
  namespace: "staging"
candidate@cli:~$ kubectl get sa -n staging
NAME      SECRETS   AGE
default   1         6h6m
candidate@cli:~$ kubectl create -f /home/candidate/KSMV00102/service-account.yaml
serviceaccount/psp-restrict-sa created
candidate@cli:~$ kubectl get sa -n staging
NAME              SECRETS   AGE
default           1         6h6m
psp-restrict-sa   1         2s
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ kubectl create clusterrolebinding restrict-access-bind --clusterrole=restri
ct-access-role --serviceaccount=staging:psp-restrict-sa --dry-run -o yaml
W0520 14:41:23.502004   47627 helpers.go:598] --dry-run is deprecated and can be replaced wi
th --dry-run=client.
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  creationTimestamp: null
  name: restrict-access-bind
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restrict-access-role
subjects:
- kind: ServiceAccount
  name: psp-restrict-sa
  namespace: staging
candidate@cli:~$ vim /home/candidate/KSMV00102/cluster-role
cluster-role-binding.yaml  cluster-role.yaml
candidate@cli:~$ vim /home/candidate/KSMV00102/cluster-role
cluster-role-binding.yaml  cluster-role.yaml
candidate@cli:~$ vim /home/candidate/KSMV00102/cluster-role-binding.yaml
```



```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restrict-access-bind
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restrict-access-role
subjects:
- kind: ServiceAccount
  name: psp-restrict-sa
  namespace: staging
```



```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: restrict-access-bind
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: restrict-access-role
subjects:
- kind: ServiceAccount
  name: psp-restrict-sa
  namespace: staging

candidate@cli:~$
candidate@cli:~$ kubectl create -f /home/candidate/KSMV00102/cluster-role-binding.yaml
clusterrolebinding.rbac.authorization.k8s.io/restrict-access-bind created
candidate@cli:~$
```