



# CKS<sup>Q&As</sup>

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.passapply.com/cks.html>

100% Passing Guarantee  
100% Money Back Assurance

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

- ⚙️ **Instant Download** After Purchase
- ⚙️ **100% Money Back** Guarantee
- ⚙️ **365 Days** Free Update
- ⚙️ **800,000+** Satisfied Customers





## QUESTION 1

Cluster: qa-cluster

Master node: master Worker node: worker1 You can switch the cluster/configuration context using the following command: [desk@cli] \$ kubectl config use-context qa-cluster

Task:

Create a NetworkPolicy named restricted-policy to restrict access to Pod product running in namespace dev.

Only allow the following Pods to connect to Pod products-service:

1.

Pods in the namespace qa

2.

Pods with label environment: stage, in any namespace

A. See the below.

B. Placeholder

Correct Answer: A

---

## QUESTION 2

```
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.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:~# systemctl restart kubelet.service
root@kssc00202-master:~# systemctl enable kubelet.service
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~#
root@kssc00202-master:~# ls
KSSC00202 snap
root@kssc00202-master:~# cat KSSC00202/vulnerable-resource.yaml
```

```
KSSC00202 snap
root@kssc00202-master:~# cat KSSC00202/vulnerable-resource.yaml
---
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:~# kubectcl create -f KSSC00202/vulnerable-resource.yaml
```

```
root@kssc00202-master:~# kubectcl create -f KSSC00202/vulnerable-resource.yaml
The connection to the server 10.177.80.12:6443 was refused - did you specify the right host or port?
root@kssc00202-master:~# kubectcl 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 .
drwxr-xr-x 9 root root 4096 Oct 11 15:36 ..
drwxr-xr-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 a6be758cef4cd 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/docker.sock unix:///run/containerd/containerd.sock unix:///run/crio.sock unix:///var/run/cri-dockerd.sock]. As the default settings are now deprecated, you should set the endpoint instead.
WARN[0000] unable to determine runtime API version: rpc error: code = Unavailable desc = connection error: desc = "transport: Error while dialing dial unix /var/run/docker.sock: connect: no such file or directory"
WARN[0000] image connect using default endpoints: [unix:///var/run/docker.sock unix:///run/containerd/containerd.sock unix:///run/crio.sock unix:///var/run/cri-dockerd.sock]. As the default settings are now deprecated, you should set the endpoint instead.
WARN[0000] unable to determine image API version: rpc error: code = Unavailable desc = connection error: desc = "transport: Error while dialing dial unix /var/run/docker.sock: connect: no such file or directory"
a003b3dfb61c d3377fb7177c 30 seconds ago Exited kube-apiserver 3 2dad4e
984a91 kube-apiserver-kssc00202-master
5e70b9a70f9ed d3377fb7177c 7 hours ago Exited kube-apiserver 0 68a9f31
6c2559 kube-apiserver-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 3

Create a PSP that will prevent the creation of privileged pods in the namespace.

Create a new PodSecurityPolicy named prevent-privileged-policy which prevents the creation of privileged pods.

Create a new ServiceAccount named psp-sa in the namespace default.

Create a new ClusterRole named prevent-role, which uses the newly created Pod Security Policy prevent-privileged-policy.

Create a new ClusterRoleBinding named prevent-role-binding, which binds the created ClusterRole prevent-role to the created SA psp-sa.

Also, Check the Configuration is working or not by trying to Create a Privileged pod, it should get failed.

A. See the below.

B. Placeholder

Correct Answer: A

---



Create a PSP that will prevent the creation of privileged pods in the namespace. \$ cat clusterrole-use-privileged.yaml  
apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRole metadata: name: use-privileged-psp rules:

-apiGroups: ['policy']

resources: ['podsecuritypolicies']

verbs: ['use']

resourceNames:

-default-psp

apiVersion: rbac.authorization.k8s.io/v1 kind: RoleBinding metadata: name: privileged-role-bind namespace: psp-test  
roleRef: apiGroup: rbac.authorization.k8s.io kind: ClusterRole name: use-privileged-psp subjects:

-kind: ServiceAccount name: privileged-sa \$ kubectl -n psp-test apply -f clusterrole-use-privileged.yaml

After a few moments, the privileged Pod should be created.

Create a new PodSecurityPolicy named prevent-privileged-policy which prevents the creation of privileged pods.

apiVersion: policy/v1beta1

kind: PodSecurityPolicy

metadata:

name: example

spec:

privileged: false # Don't allow privileged pods!

# The rest fills in some required fields.

seLinux:

rule: RunAsAny

supplementalGroups:

rule: RunAsAny

runAsUser:

rule: RunAsAny

fsGroup:

rule: RunAsAny

volumes:

-[\*]





And create it with kubectl:

```
kubectl-admin create -f example-psp.yaml
```

Now, as the unprivileged user, try to create a simple pod:

```
kubectl-user create -f-
```