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.geekcert.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

CORRECT TEXT Context

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

Cluster	Master node	Worker node
KSCS00101	kscs00101-master	kscs00101-worker1

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

```
[candidate@cli] $ kubectl config use-context KSCS00101
```

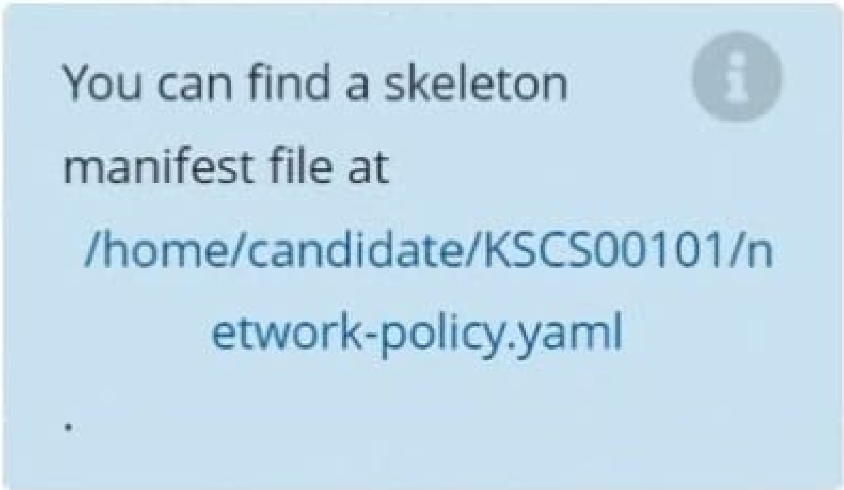
A default-deny NetworkPolicy avoids to accidentally expose a Pod in a namespace that doesn\\'t have any other NetworkPolicy defined.

Task

Create a new default-deny NetworkPolicy named defaultdeny in the namespace testing for all traffic of type Egress.

The new NetworkPolicy must deny all Egress traffic in the namespace testing.

Apply the newly created default-deny NetworkPolicy to all Pods running in namespace testing.



A. See explanation below.

B. PlaceHolder

Correct Answer: A

QUESTION 2

Create a PSP that will only allow the persistentvolumeclaim as the volume type in the namespace restricted.

Create a new PodSecurityPolicy named prevent-volume-policy which prevents the pods which is having different volumes mount apart from persistentvolumeclaim.

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

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

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

Hint:

Also, Check the Configuration is working or not by trying to Mount a Secret in the pod maifest, it should get failed.

POD Manifest:

1.

apiVersion: v1

2.

kind: Pod

3.

metadata:

4.

name:

5.

spec:

6.

containers:

7.

- name:

8.

image:

9.

volumeMounts: 10.- name: 11.mountPath: 12.volumes: 13.- name: 14.secret: 15.secretName:

A. See the below:

B. PlaceHolder

Correct Answer: A

apiVersion: policy/v1beta1

kind: PodSecurityPolicy

metadata:

name: restricted

annotations:

seccomp.security.alpha.kubernetes.io/allowedProfileNames:

\\'docker/default,runtime/default\\'

apparmor.security.beta.kubernetes.io/allowedProfileNames: \\'runtime/default\\'
seccomp.security.alpha.kubernetes.io/defaultProfileName: \\'runtime/default\\'
apparmor.security.beta.kubernetes.io/defaultProfileName: \\'runtime/default\\' spec:

privileged: false

Required to prevent escalations to root.

allowPrivilegeEscalation: false

This is redundant with non-root + disallow privilege escalation, # but we can provide it for defense in depth.

requiredDropCapabilities:

-ALL

Allow core volume types.

volumes:

-\\'configMap\\'

-\\'emptyDir\\'

-\\'projected\\'

-\\'secret\\'

-\\'downwardAPI\\'

Assume that persistentVolumes set up by the cluster admin are safe to use.

-\\'persistentVolumeClaim\\'

hostNetwork: false

hostIPC: false

hostPID: false

runAsUser:

Require the container to run without root privileges.

rule: \\'MustRunAsNonRoot\\'

seLinux:

This policy assumes the nodes are using AppArmor rather than SELinux.

rule: \\'RunAsAny\\'

supplementalGroups:

rule: \\'MustRunAs\\'

ranges:

Forbid adding the root group.

-

min: 1

max: 65535

fsGroup:

rule: \\'MustRunAs\\'

ranges:

Forbid adding the root group.

-

min: 1

max: 65535

readOnlyRootFilesystem: false

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

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 5

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

[desk@cli] $ kubectl config use-context dev

A default-deny NetworkPolicy avoid to accidentally expose a Pod in a namespace that doesn\\'t have any other NetworkPolicy defined.

Task: Create a new default-deny NetworkPolicy named deny-network in the namespace test for all traffic of type Ingress + Egress

The new NetworkPolicy must deny all Ingress + Egress traffic in the namespace test.

Apply the newly created default-deny NetworkPolicy to all Pods running in namespace test.

You can find a skeleton manifests file at /home/cert_masters/network-policy.yaml

A. See the explanation below

B. PlaceHolder

Correct Answer: A

master1 $ k get pods -n test --show-labels uk.co.certification.simulator.questionpool.PList@132b47c0 $ vim netpol.yaml uk.co.certification.simulator.questionpool.PList@132b4af0 master1 $ k apply -f netpol.yaml

controlplane $ k get pods -n test --show-labels NAME READY STATUS RESTARTS AGE LABELS test-pod 1/1 Running 0 34s role=test,run=test-pod testing 1/1 Running 0 17d run=testing master1 $ vim netpol1.yaml apiVersion: networking.k8s.io/v1 kind: NetworkPolicy metadata: name: deny-network namespace: test spec: podSelector: {} policyTypes:

-Ingress

-Egress