

## LinuxFoundation.CKS.v2023-05-08.q22

|                                                                                                                                                           |                                                |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------|
| Exam Code:                                                                                                                                                | CKS                                            |
| Exam Name:                                                                                                                                                | Certified Kubernetes Security Specialist (CKS) |
| Certification Provider:                                                                                                                                   | Linux Foundation                               |
| Free Question Number:                                                                                                                                     | 22                                             |
| Version:                                                                                                                                                  | v2023-05-08                                    |
| # of views:                                                                                                                                               | 1764                                           |
| # of Questions views:                                                                                                                                     | 220                                            |
| <a href="https://www.freepdfdumps.com/LinuxFoundation.CKS.v2023-05-08.q22.html">https://www.freepdfdumps.com/LinuxFoundation.CKS.v2023-05-08.q22.html</a> |                                                |

### NEW QUESTION: 1

Context

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

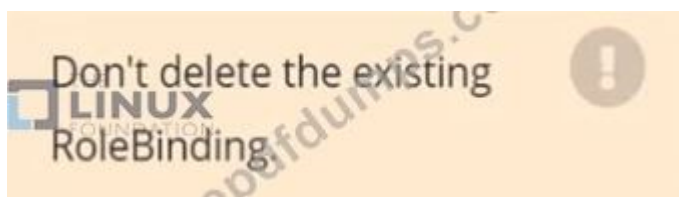
Task

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.



**Answer:**

```

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:~$

```

## NEW QUESTION: 2

You can switch the cluster/configuration context using the following command: [desk@cli] \$ kubectl config use-context stage Context: A PodSecurityPolicy shall prevent the creation of privileged Pods in a specific namespace. Task: 1. Create a new PodSecurityPolicy named deny-policy, which prevents the creation of privileged Pods. 2. Create a new ClusterRole name deny-access-role, which uses the newly created PodSecurityPolicy deny-policy. 3. Create a new ServiceAccount named psd-denial-sa in the existing namespace development. Finally, create a new ClusterRoleBindind named restrict-access-bind, which binds the newly created ClusterRole deny-access-role to the newly created ServiceAccount psp-denial-sa

### Answer:

Create psp to disallow privileged container

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRole

metadata:

name: deny-access-role

rules:

- apiGroups: ['policy']

resources: ['podsecuritypolicies']

```
verbs: ['use']
resourceNames:
- "deny-policy"
k create sa psp-denial-sa -n development
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
name: restrict-access-bing
roleRef:
kind: ClusterRole
name: deny-access-role
apiGroup: rbac.authorization.k8s.io
subjects:
- kind: ServiceAccount
name: psp-denial-sa
namespace: development
Explanation
master1 $ vim psp.yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
name: deny-policy
spec:
privileged: false # Don't allow privileged pods!
seLinux:
rule: RunAsAny
supplementalGroups:
rule: RunAsAny
runAsUser:
rule: RunAsAny
fsGroup:
rule: RunAsAny
volumes:
- '*'
master1 $ vim cr1.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
name: deny-access-role
rules:
- apiGroups: ['policy']
```

```
resources: ['podsecuritypolicies']
verbs: ['use']
resourceNames:
- "deny-policy"
master1 $ k create sa psp-denial-sa -n development master1 $ vim cb1.yaml apiVersion:
rbac.authorization.k8s.io/v1 kind: ClusterRoleBinding metadata:
name: restrict-access-bing
roleRef:
kind: ClusterRole
name: deny-access-role
apiGroup: rbac.authorization.k8s.io
subjects:
# Authorize specific service accounts:
- kind: ServiceAccount
name: psp-denial-sa
namespace: development
master1 $ k apply -f psp.yaml master1 $ k apply -f cr1.yaml master1 $ k apply -f cb1.yaml
Reference: https://kubernetes.io/docs/concepts/policy/pod-security-policy/
```

### NEW QUESTION: 3

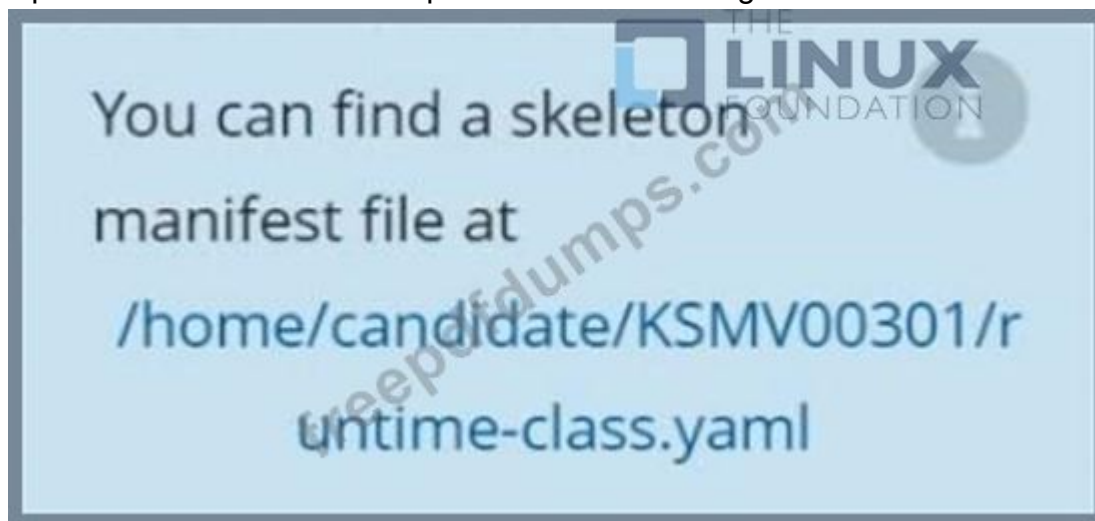
Context

This cluster uses containerd as CRI runtime.

Containerd's default runtime handler is runc. Containerd has been prepared to support an additional runtime handler, runsc (gVisor).

Task

Create a RuntimeClass named sandboxed using the prepared runtime handler named runsc. Update all Pods in the namespace server to run on gVisor.



**Answer:**

```
candidate@cli:~$ kubectl config use-context KSMV00301
Switched to context "KSMV00301".
candidate@cli:~$ cat /home/candidate/KSMV00301/runtime-class.yaml
---
apiVersion: node.k8s.io/v1
kind: RuntimeClass
metadata:
  name: ""
  handler: ""
candidate@cli:~$ vim /home/candidate/KSMV00301/runtime-class.yaml
```

```
apiVersion: node.k8s.io/v1
kind: RuntimeClass
metadata:
  name: "sandboxed"
  handler: "runsc"

```

```

candidate@cli:~$ kubectl config use-context KSMV00301
Switched to context "KSMV00301".
candidate@cli:~$ cat /home/candidate/KSMV00301/runtime-class.yaml
---
apiVersion: node.k8s.io/v1
kind: RuntimeClass
metadata:
  name: ""
handler: ""
candidate@cli:~$ vim /home/candidate/KSMV00301/runtime-class.yaml
candidate@cli:~$ cat /home/candidate/KSMV00301/runtime-class.yaml
---
apiVersion: node.k8s.io/v1
kind: RuntimeClass
metadata:
  name: "sandboxed"
handler: "runsc"
candidate@cli:~$ kubectl get deployments.apps -n server
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
workload1            1/1      1              1            5h43m
workload2            1/1      1              1            5h43m
workload3            1/1      1              1            5h43m
candidate@cli:~$ kubectl get pods -n server
NAME                                READY    STATUS      RESTARTS    AGE
workload1-6869857dd7-s45rc          1/1      Running     0            5h43m
workload2-d4bd497d5-h44df           1/1      Running     0            5h43m
workload3-8587774495-chm56          1/1      Running     0            5h43m
candidate@cli:~$ kubectl -n server edit deployments.apps workload1

```

```
template:
  metadata:
    creationTimestamp: null
    labels:
      app: nginx
      name: workload1
  spec:
    runtimeClassName: sandboxed
    containers:
    - image: nginx:1.14.2
      imagePullPolicy: IfNotPresent
      name: workload1
      ports:
      - containerPort: 80
        protocol: TCP
      resources: {}
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
    dnsPolicy: ClusterFirst
    restartPolicy: Always
    schedulerName: default-scheduler
    securityContext: {}
    terminationGracePeriodSeconds: 30
status:
  "/tmp/kubectl-edit-3385772700.yaml"
```

```

NAME                                READY   STATUS    RESTARTS   AGE
workload1-6869857dd7-s45rc         1/1     Running   0           5h44m
workload2-d4bd497d5-h44df          1/1     Running   0           5h44m
workload3-8587774495-chm56         1/1     Running   0           5h44m
candidate@cli:~$ kubectl -n server edit deployments.apps workload1
Edit cancelled, no changes made.
candidate@cli:~$ kubectl get pods -n server
NAME                                READY   STATUS    RESTARTS   AGE
workload1-6869857dd7-s45rc         1/1     Running   0           5h45m
workload2-d4bd497d5-h44df          1/1     Running   0           5h44m
workload3-8587774495-chm56         1/1     Running   0           5h44m
candidate@cli:~$ kubectl -n server edit deployments.apps workload2
Edit cancelled, no changes made.
candidate@cli:~$ kubectl create -f /home/candidate/KSMV00301/runtime-class.yaml
runtimeclass.node.k8s.io/sandboxed created
candidate@cli:~$ kubectl get pods -n server
NAME                                READY   STATUS    RESTARTS   AGE
workload1-6869857dd7-s45rc         1/1     Running   0           5h45m
workload2-d4bd497d5-h44df          1/1     Running   0           5h45m
workload3-8587774495-chm56         1/1     Running   0           5h45m
candidate@cli:~$ kubectl -n server edit deployments.apps workload2

```

```

strategy:
  rollingUpdate:
    maxSurge: 25%
    maxUnavailable: 25%
  type: RollingUpdate
template:
  metadata:
    creationTimestamp: null
    labels:
      app: nginx
    name: workload2
  spec:
    runtimeClassName: sandboxed

```

```

NAME                                READY   STATUS    RESTARTS   AGE
workload1-6869857dd7-s45rc         1/1     Running   0           5h45m
workload2-d4bd497d5-h44df          1/1     Running   0           5h45m
workload3-8587774495-chm56         1/1     Running   0           5h45m
candidate@cli:~$ kubectl -n server edit deployments.apps workload2
deployment.apps/workload2 edited
candidate@cli:~$ kubectl get pods -n server
NAME                                READY   STATUS    RESTARTS   AGE
workload1-8d8649ff6-wvjtg          1/1     Running   0           15s
workload2-765bdb98c8-wd8cm          1/1     Running   0           4s
workload3-8587774495-chm56         1/1     Running   0           5h45m
candidate@cli:~$ kubectl -n server edit deployments.apps workload3

```

```

app: nginx
name: workload3
spec:
  runtimeClassName: sandboxed
  containers:
  - image: nginx:1.14.2
    imagePullPolicy: IfNotPresent
    name: workload3

```

```

candidate@cli:~$ kubectl -n server edit deployments.apps workload3
deployment.apps/workload3 edited
candidate@cli:~$ kubectl get pods -n server
NAME                                READY   STATUS    RESTARTS   AGE
workload1-8d8649ff6-wvjtg           1/1     Running   0           58s
workload2-765bdb98c8-wd8cm          1/1     Running   0           47s
workload3-76c994bb4d-s6k85          1/1     Running   0           4s
candidate@cli:~$

```

#### NEW QUESTION: 4

use the Trivy to scan the following images,

1. amazonlinux:1
2. k8s.gcr.io/kube-controller-manager:v1.18.6

Look for images with HIGH or CRITICAL severity vulnerabilities and store the output of the same in /opt/trivy-vulnerable.txt

A. Send us your suggestion

B. Send us your suggestion on it.

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

#### NEW 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

**Answer:**

```
master1 $ 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
```

```
$ vim netpol.yaml
```

apiVersion: networking.k8s.io/v1

kind: NetworkPolicy

metadata:

name: deny-network

namespace: test

spec:

podSelector: {}

policyTypes:

- Ingress

- Egress

master1 \$ k apply -f netpol.yaml

Explanation

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

master1 \$ k apply -f netpol1.yaml Reference: [https://kubernetes.io/docs/concepts/services-](https://kubernetes.io/docs/concepts/services-networking/network-policies/)

networking/network-policies/ Explanation 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

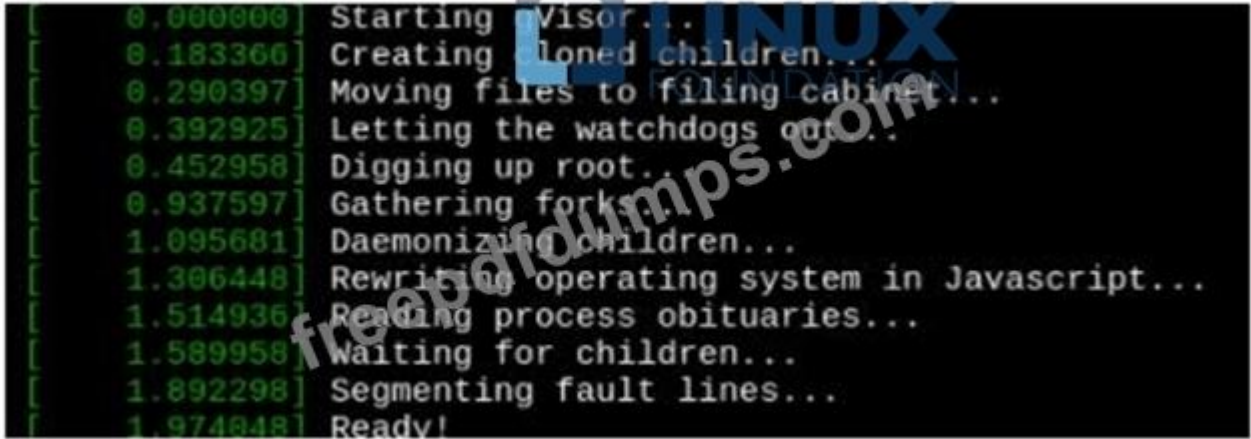
master1 \$ k apply -f netpol1.yaml Reference: [https://kubernetes.io/docs/concepts/services-](https://kubernetes.io/docs/concepts/services-networking/network-policies/)

networking/network-policies/

**NEW QUESTION: 6**

Create a RuntimeClass named untrusted using the prepared runtime handler named runsc. Create a Pods of image alpine:3.13.2 in the Namespace default to run on the gVisor runtime class.

**Answer:**



```
[ 0.000000] Starting gvisor...  
[ 0.183366] Creating cloned children...  
[ 0.290397] Moving files to filing cabinet...  
[ 0.392925] Letting the watchdogs out...  
[ 0.452958] Digging up root...  
[ 0.937597] Gathering forks...  
[ 1.095681] Daemonizing children...  
[ 1.306448] Rewriting operating system in Javascript...  
[ 1.514936] Reading process obituaries...  
[ 1.589958] Waiting for children...  
[ 1.892298] Segmenting fault lines...  
[ 1.974048] Ready!
```

**NEW QUESTION: 7**

Given an existing Pod named nginx-pod running in the namespace test-system, fetch the service-account-name used and put the content in /candidate/KSC00124.txt Create a new Role named dev-test-role in the namespace test-system, which can perform update operations, on resources of type namespaces.

Create a new RoleBinding named dev-test-role-binding, which binds the newly created Role to the Pod's ServiceAccount ( found in the Nginx pod running in namespace test-system).

**Answer:**

```

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

```

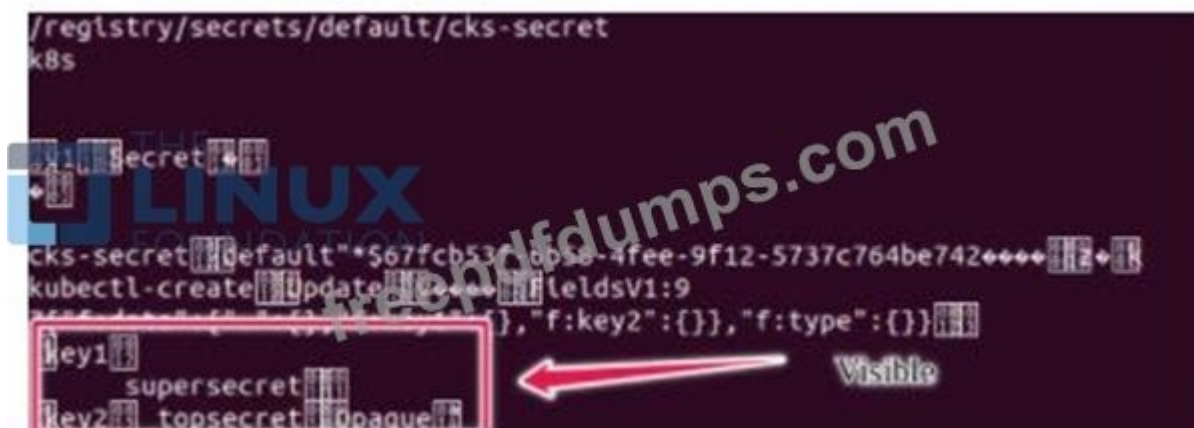
```

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:~$

```

## NEW QUESTION: 8

Secrets stored in the etcd is not secure at rest, you can use the etcdctl command utility to find the secret value for e.g:- ETCDCTL\_API=3 etcdctl get /registry/secrets/default/cks-secret --cacert="ca.crt" --cert="server.crt" --key="server.key" Output



```

/registry/secrets/default/cks-secret
k8s
{"key1": "supersecret", "key2": "topsecret", "type": "Opaque"}

```

Using the Encryption Configuration, Create the manifest, which secures the resource secrets using the provider AES-CBC and identity, to encrypt the secret-data at rest and ensure all secrets are encrypted with the new configuration.

## Answer:

ETCD secret encryption can be verified with the help of etcdctl command line utility.

ETCD secrets are stored at the path `/registry/secrets/$namespace/$secret` on the master node. The below command can be used to verify if the particular ETCD secret is encrypted or not.

```
# ETCDCTL_API=3 etcdctl get /registry/secrets/default/secret1 [...] | hexdump -C
```

### NEW QUESTION: 9

Create a new ServiceAccount named `backend-sa` in the existing namespace `default`, which has the capability to list the pods inside the namespace `default`.

Create a new Pod named `backend-pod` in the namespace `default`, mount the newly created `sa backend-sa` to the pod, and Verify that the pod is able to list pods.

Ensure that the Pod is running.

#### Answer:

A service account provides an identity for processes that run in a Pod.

When you (a human) access the cluster (for example, using `kubectl`), you are authenticated by the `apiserver` as a particular User Account (currently this is usually `admin`, unless your cluster administrator has customized your cluster). Processes in containers inside pods can also contact the `apiserver`. When they do, they are authenticated as a particular Service Account (for example, `default`).

When you create a pod, if you do not specify a service account, it is automatically assigned the default service account in the same namespace. If you get the raw json or yaml for a pod you have created (for example, `kubectl get pods/<podname> -o yaml`), you can see the `spec.serviceAccountName` field has been automatically set.

You can access the API from inside a pod using automatically mounted service account credentials, as described in [Accessing the Cluster](#). The API permissions of the service account depend on the authorization plugin and policy in use.

In version 1.6+, you can opt out of automounting API credentials for a service account by setting `automountServiceAccountToken: false` on the service account:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: build-robot
automountServiceAccountToken: false
```

...

In version 1.6+, you can also opt out of automounting API credentials for a particular pod:

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  serviceAccountName: build-robot
  automountServiceAccountToken: false
```

...

The pod spec takes precedence over the service account if both specify a `automountServiceAccountToken` value.

### NEW QUESTION: 10

#### Task

Analyze and edit the given Dockerfile `/home/candidate/KSSC00301/Dockerfile` (based on the `ubuntu:16.04` image), fixing two instructions present in the file that are prominent security/best-practice issues.

Analyze and edit the given manifest file `/home/candidate/KSSC00301/deployment.yaml`, fixing two fields present in the file that are prominent security/best-practice issues.

Don't add or remove configuration settings; only modify the existing configuration settings, so that **two** configuration settings each are no longer security/best-practice concerns.

Should you need an unprivileged user for any of the tasks, use user `nobody` with user id `65535`.

#### Answer:

```
candidate@cli:~$ kubectl config use-context KSSC00301
Switched to context "KSSC00301".
candidate@cli:~$ vim KSSC00301/Dockerfile
```

```

FROM ubuntu:16.04

USER root
RUN apt-get update && \
    apt-get install -yq --no-install-recommends runit=2.1.2-3ubuntu1 wget=1.17.1-1ubuntu1.5 \
    \
        chrpath=0.16-1 tzdata=2020a-0ubuntu0.16.04 lsof=4.89+dfsg-0.1 lshw=02.17-1.1ubuntu3 \
    \
        sysstat=11.2.0-1ubuntu0.3 net-tools=1.60-26ubuntu1 numactl=2.0.11-1ubuntu1.1 \
        bzip2=1.0.6-8ubuntu0.2 && \
    apt-get autoremove && apt-get clean && \
    rm -rf /var/lib/apt/lists/* /tmp/* /var/tmp/*

ARG CB_VERSION=6.5.1
ARG CB_RELEASE_URL=https://packages.couchbase.com/releases/6.5.1
ARG CB_PACKAGE=couchbase-server-enterprise_6.5.1-ubuntu16.04_amd64.deb
ARG CB_SHA256=80427193137e5cb5a4795b2675b1c450claf8cfla5c634d917f66416f2047e66

ENV PATH=$PATH:/opt/couchbase/bin:/opt/couchbase/bin/tools:/opt/couchbase/bin/install

RUN groupadd -g 1000 couchbase && useradd couchbase -u 1000 -g couchbase -M

SHELL ["/bin/bash", "-o", "pipefail", "-c"]
RUN export INSTALL_DONT_START_SERVER=1 && \
    wget -N --no-verbose $CB_RELEASE_URL/$CB_PACKAGE && \
    echo "$CB_SHA256 $CB_PACKAGE" | sha256sum -c - && \
    dpkg -i ./CB_PACKAGE && rm -f ./CB_PACKAGE

COPY scripts/run /etc/service/couchbase-server/run
RUN chown -R couchbase:couchbase /etc/service

COPY scripts/dummy.sh /usr/local/bin/
RUN ln -s dummy.sh /usr/local/bin/iptables-save && \
    ln -s dummy.sh /usr/local/bin/lvdisplay && \
    ln -s dummy.sh /usr/local/bin/vgdisplay && \
    ln -s dummy.sh /usr/local/bin/pvdisplay

RUN chrpath -r "$ORIGIN/../lib" /opt/couchbase/bin/curl

COPY scripts/entrypoint.sh /
ENTRYPOINT ["/entrypoint.sh"]
USER nobody
CMD ["couchbase-server"]

EXPOSE 8091 8092 8093 8094 8095 8096 11207 11210 11211 18091 18092 18093 18094 18095 18096
VOLUME /opt/couchbase/var

```

```

candidate@cli:~$ kubectl config use-context KSSC00301
Switched to context "KSSC00301"
candidate@cli:~$ vim KSSC00301/Dockerfile
candidate@cli:~$ vim KSSC00301/deployment.yaml

```

```

securityContext:
  capabilities: {'add': ['NET_BIND_SERVICE'], 'drop': ['all']}, 'privileged': F
  also, 'readOnlyRootFilesystem': True, 'runAsUser': 65535
  resources:
    limits:
      cpu: 2
      memory: 1024Mi
    requests:
      cpu: 1
      memory: 512Mi
  volumes:
    - name: database-storage

```

## NEW QUESTION: 11

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

b. Ensure that the admission control plugin PodSecurityPolicy is set.

c. Ensure that the --kubelet-certificate-authority argument is set as appropriate.

Fix all of the following violations that were found against the Kubelet:- a. Ensure the --anonymous-auth argument is set to false.

b. Ensure that the --authorization-mode argument is set to Webhook.

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

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

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

Hint: Take the use of Tool Kube-Bench

### Answer:

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=<ca-string>

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 master node 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 master node and either remove the --peer-auto-tls parameter or set it to false. --peer-auto-tls=false

## NEW QUESTION: 12

You can switch the cluster/configuration context using the following command: [desk@cli] \$

kubectcl config use-context dev Context: A CIS Benchmark tool was run against the kubeadm

created cluster and found multiple issues that must be addressed. Task: Fix all issues via

configuration and restart the affected components to ensure the new settings take effect. Fix all of the following violations that were found against the API server: 1.2.7 authorization-mode

argument is not set to AlwaysAllow FAIL 1.2.8 authorization-mode argument includes Node FAIL

1.2.7 authorization-mode argument includes RBAC FAIL Fix all of the following violations that

were found against the Kubelet: 4.2.1 Ensure that the anonymous-auth argument is set to false

FAIL 4.2.2 authorization-mode argument is not set to AlwaysAllow FAIL (Use Webhook

autumn/authz where possible) Fix all of the following violations that were found against etcd: 2.2

Ensure that the client-cert-auth argument is set to true

### Answer:

worker1 \$ vim /var/lib/kubelet/config.yaml

anonymous:

enabled: true #Delete this

enabled: false #Replace by this

authorization:

mode: AlwaysAllow #Delete this

mode: Webhook #Replace by this

```
worker1 $ systemctl restart kubelet. # To reload kubelet config ssh to master1 master1 $  
vim /etc/kubernetes/manifests/kube-apiserver.yaml - -- authorization-mode=Node,RBAC master1  
$ vim /etc/kubernetes/manifests/etcd.yaml - --client-cert-auth=true Explanation ssh to worker1  
worker1 $ vim /var/lib/kubelet/config.yaml apiVersion: kubelet.config.k8s.io/v1beta1  
authentication:  
anonymous:  
enabled: true #Delete this  
enabled: false #Replace by this  
webhook:  
cacheTTL: 0s  
enabled: true  
x509:  
clientCAFile: /etc/kubernetes/pki/ca.crt  
authorization:  
mode: AlwaysAllow #Delete this  
mode: Webhook #Replace by this  
webhook:  
cacheAuthorizedTTL: 0s  
cacheUnauthorizedTTL: 0s  
cgroupDriver: systemd  
clusterDNS:  
- 10.96.0.10  
clusterDomain: cluster.local  
cpuManagerReconcilePeriod: 0s  
evictionPressureTransitionPeriod: 0s  
fileCheckFrequency: 0s  
healthzBindAddress: 127.0.0.1  
healthzPort: 10248  
httpCheckFrequency: 0s  
imageMinimumGCAge: 0s  
kind: KubeletConfiguration  
logging: {}  
nodeStatusReportFrequency: 0s  
nodeStatusUpdateFrequency: 0s  
resolvConf: /run/systemd/resolve/resolv.conf  
rotateCertificates: true  
runtimeRequestTimeout: 0s  
staticPodPath: /etc/kubernetes/manifests  
streamingConnectionIdleTimeout: 0s  
syncFrequency: 0s  
volumeStatsAggPeriod: 0s
```

worker1 \$ systemctl restart kubelet. # To reload kubelet config ssh to master1 master1 \$  
vim /etc/kubernetes/manifests/kube-apiserver.yaml

```
kind: Pod
metadata:
  annotations:
    kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: 172.17.0.22:6443
  labels:
    component: kube-apiserver
    tier: control-plane
  name: kube-apiserver
  namespace: kube-system
spec:
  containers:
  - command:
    - kube-apiserver
    - --advertise-address=172.17.0.22
    - --allow-privileged=true
    # - --authorization-mode=AlwaysAllow # Delete This
    - --authorization-mode=Node,RBAC # Replace by this line
    - --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
    - --insecure-port=0
```

master1 \$ vim /etc/kubernetes/manifests/etcd.yaml

```
kind: Pod
metadata:
  annotations:
    kubeadm.kubernetes.io/etcd.advertise-client-urls: https://172.17.0.29:2379
  creationTimestamp: null
  labels:
    component: etcd
    tier: control-plane
  name: etcd
  namespace: kube-system
spec:
  containers:
  - command:
    - etcd
    - --advertise-client-urls=https://172.17.0.29:2379
    - --cert-file=/etc/kubernetes/pki/etcd/server.crt
    - --client-cert-auth=true #Add this line
    - --data-dir=/var/lib/etcd
    - --initial-advertise-peer-urls=https://172.17.0.29:2380
    - --initial-cluster=controlplane=https://172.17.0.29:2380
    - --key-file=/etc/kubernetes/pki/etcd/server.key
    - --listen-client-urls=https://127.0.0.1:2379,https://172.17.0.29:2379
    - --listen-metrics-urls=http://127.0.0.1:2381
    - --listen-peer-urls=https://172.17.0.29:2380
    - --name=controlplane
    - --peer-cert-file=/etc/kubernetes/pki/etcd/peer.crt
    - --peer-client-cert-auth=true
    - --peer-key-file=/etc/kubernetes/pki/etcd/peer.key
    - --peer-trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
    - --snapshot-count=10000
    - --trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
  image: k8s.gcr.io/etcd:3.4.9-1
```

### NEW QUESTION: 13

You can switch the cluster/configuration context using the following command: [desk@cli] \$  
kubectl config use-context test-account Task: Enable audit logs in the cluster.

To do so, enable the log backend, and ensure that:

1. logs are stored at /var/log/Kubernetes/logs.txt
2. log files are retained for 5 days
3. at maximum, a number of 10 old audit log files are retained

A basic policy is provided at /etc/Kubernetes/logpolicy/audit-policy.yaml. It only specifies what not to log. Note: The base policy is located on the cluster's master node.

Edit and extend the basic policy to log: 1. Nodes changes at RequestResponse level 2. The request body of persistentvolumes changes in the namespace frontend 3. ConfigMap and Secret changes in all namespaces at the Metadata level Also, add a catch-all rule to log all other requests at the Metadata level Note: Don't forget to apply the modified policy.

#### Answer:

```
$ vim /etc/kubernetes/log-policy/audit-policy.yaml
```

```
- level: RequestResponse
```

```
userGroups: ["system:nodes"]
```

```
- level: Request
```

```
resources:
```

```
- group: "" # core API group
```

```
resources: ["persistentvolumes"]
```

```
namespaces: ["frontend"]
```

```
- level: Metadata
```

```
resources:
```

```
- group: ""
```

```
resources: ["configmaps", "secrets"]
```

```
- level: Metadata
```

```
$ vim /etc/kubernetes/manifests/kube-apiserver.yaml Add these
```

```
- --audit-policy-file=/etc/kubernetes/log-policy/audit-policy.yaml
```

```
- --audit-log-path=/var/log/kubernetes/logs.txt
```

```
- --audit-log-maxage=5
```

```
- --audit-log-maxbackup=10
```

Explanation

```
[desk@cli] $ ssh master1 [master1@cli] $ vim /etc/kubernetes/log-policy/audit-policy.yaml
```

```
apiVersion: audit.k8s.io/v1 # This is required.
```

```
kind: Policy
```

```
# Don't generate audit events for all requests in RequestReceived stage.
```

```
omitStages:
```

```
- "RequestReceived"
```

```
rules:
```

```

# Don't log watch requests by the "system:kube-proxy" on endpoints or services
- level: None
users: ["system:kube-proxy"]
verbs: ["watch"]
resources:
- group: "" # core API group
resources: ["endpoints", "services"]
# Don't log authenticated requests to certain non-resource URL paths.
- level: None
userGroups: ["system:authenticated"]
nonResourceURLs:
- "/api*" # Wildcard matching.
- "/version"
# Add your changes below
- level: RequestResponse
userGroups: ["system:nodes"] # Block for nodes
- level: Request
resources:
- group: "" # core API group
resources: ["persistentvolumes"] # Block for persistentvolumes
namespaces: ["frontend"] # Block for persistentvolumes of frontend ns
- level: Metadata
resources:
- group: "" # core API group
resources: ["configmaps", "secrets"] # Block for configmaps & secrets
- level: Metadata # Block for everything else
[master1@cli] $ vim /etc/kubernetes/manifests/kube-apiserver.yaml
apiVersion: v1
kind: Pod
metadata:
annotations:
kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: 10.0.0.5:6443 labels:
component: kube-apiserver
tier: control-plane
name: kube-apiserver
namespace: kube-system
spec:
containers:
- command:
- kube-apiserver
- --advertise-address=10.0.0.5

```

- --allow-privileged=true
- --authorization-mode=Node,RBAC
- --audit-policy-file=/etc/kubernetes/log-policy/audit-policy.yaml #Add this
- --audit-log-path=/var/log/kubernetes/logs.txt #Add this
- --audit-log-maxage=5 #Add this
- --audit-log-maxbackup=10 #Add this

...

output truncated

Note: log volume & policy volume is already mounted in vim /etc/kubernetes/manifests/kube-apiserver.yaml so no need to mount it. Reference: <https://kubernetes.io/docs/tasks/debug-application-cluster/audit/>

### NEW QUESTION: 14

Enable audit logs in the cluster, To Do so, enable the log backend, and ensure that

1. logs are stored at /var/log/kubernetes-logs.txt.
2. Log files are retained for 12 days.
3. at maximum, a number of 8 old audit logs files are retained.
4. set the maximum size before getting rotated to 200MB

Edit and extend the basic policy to log:

1. namespaces changes at RequestResponse
2. Log the request body of secrets changes in the namespace kube-system.
3. Log all other resources in core and extensions at the Request level.
4. Log "pods/portforward", "services/proxy" at Metadata level.
5. Omit the Stage RequestReceived

All other requests at the Metadata level

### Answer:

Kubernetes auditing provides a security-relevant chronological set of records about a cluster.

Kube-apiserver performs auditing. Each request on each stage of its execution generates an event, which is then pre-processed according to a certain policy and written to a backend. The policy determines what's recorded and the backends persist the records.

You might want to configure the audit log as part of compliance with the CIS (Center for Internet Security) Kubernetes Benchmark controls.

The audit log can be enabled by default using the following configuration in cluster.yml:

services:

kube-api:

audit\_log:

enabled: true

When the audit log is enabled, you should be able to see the default values

at /etc/kubernetes/audit-policy.yaml The log backend writes audit events to a file in JSONlines format. You can configure the log audit backend using the following kube-apiserver flags:

--audit-log-path specifies the log file path that log backend uses to write audit events. Not specifying this flag disables log backend. - means standard out  
--audit-log-maxage defines the maximum number of days to retain old audit log files  
--audit-log-maxbackup defines the maximum number of audit log files to retain  
--audit-log-maxsize defines the maximum size in megabytes of the audit log file before it gets rotated If your cluster's control plane runs the kube-apiserver as a Pod, remember to mount the hostPath to the location of the policy file and log file, so that audit records are persisted. For example:  
--audit-policy-file=/etc/kubernetes/audit-policy.yaml \  
--audit-log-path=/var/log/audit.log

### NEW QUESTION: 15

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:- a. Ensure the --authorization-mode argument includes RBAC b. Ensure the --authorization-mode argument includes Node c. Ensure that the --profiling argument is set to false Fix all of the following violations that were found against the Kubelet:- a. Ensure the --anonymous-auth argument is set to false.

b. Ensure that the --authorization-mode argument is set to Webhook.

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

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

Hint: Take the use of Tool Kube-Bench

#### Answer:

API server:

Ensure the --authorization-mode argument includes RBAC

Turn on Role Based Access Control. Role Based Access Control (RBAC) allows fine-grained control over the operations that different entities can perform on different objects in the cluster. It is recommended to use the RBAC authorization mode.

Fix - Buildtime

Kubernetes

apiVersion: v1

kind: Pod

metadata:

creationTimestamp: null

labels:

component: kube-apiserver

tier: control-plane

name: kube-apiserver

namespace: kube-system

spec:

containers:  
- command:  
+ - kube-apiserver  
+ - --authorization-mode=RBAC,Node  
image: gcr.io/google\_containers/kube-apiserver-amd64:v1.6.0  
livenessProbe:  
failureThreshold: 8  
httpGet:  
host: 127.0.0.1  
path: /healthz  
port: 6443  
scheme: HTTPS  
initialDelaySeconds: 15  
timeoutSeconds: 15  
name: kube-apiserver-should-pass  
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

Ensure the --authorization-mode argument includes Node

Remediation: Edit the API server pod specification file /etc/kubernetes/manifests/kube-apiserver.yaml on the master node and set the --authorization-mode parameter to a value that includes Node.

--authorization-mode=Node,RBAC

Audit:

```
/bin/ps -ef | grep kube-apiserver | grep -v grep
```

Expected result:

'Node,RBAC' has 'Node'

Ensure that the --profiling argument is set to false

Remediation: Edit the API server pod specification file /etc/kubernetes/manifests/kube-apiserver.yaml on the master node and set the below parameter.

--profiling=false

Audit:

```
/bin/ps -ef | grep kube-apiserver | grep -v grep
```

Expected result:

'false' is equal to 'false'

Fix all of the following violations that were found against the Kubelet:- Ensure the --anonymous-auth argument is set to false.

Remediation: If using a Kubelet config file, edit the file to set authentication: anonymous: enabled to false. If using executable arguments, edit the kubelet service file /etc/systemd/system/kubelet.service.d/10-kubeadm.conf on each worker node and set the below parameter in KUBELET\_SYSTEM\_PODS\_ARGS variable.

--anonymous-auth=false

Based on your system, restart the kubelet service. For example:

```
systemctl daemon-reload
```

```
systemctl restart kubelet.service
```

Audit:

```
/bin/ps -fC kubelet
```

Audit Config:

```
/bin/cat /var/lib/kubelet/config.yaml
```

Expected result:

'false' is equal to 'false'

2) Ensure that the --authorization-mode argument is set to Webhook.

Audit

```
docker inspect kubelet | jq -e '[0].Args[] | match("--authorization-mode=Webhook").string'
```

Returned Value: --authorization-mode=Webhook Fix all of the following violations that were found against the ETCD:- a. Ensure that the --auto-tls argument is not set to true Do not use self-signed certificates for TLS. etcd is a highly-available key value store used by Kubernetes deployments for persistent storage of all of its REST API objects. These objects are sensitive in nature and should not be available to unauthenticated clients. You should enable the client authentication via valid certificates to secure the access to the etcd service.

Fix - Buildtime

Kubernetes

apiVersion: v1

kind: Pod  
metadata:  
annotations:  
scheduler.alpha.kubernetes.io/critical-pod: ""  
creationTimestamp: null  
labels:  
component: etcd  
tier: control-plane  
name: etcd  
namespace: kube-system  
spec:  
containers:  
- command:  
+ - etcd  
+ - --auto-tls=true  
image: k8s.gcr.io/etcd-amd64:3.2.18  
imagePullPolicy: IfNotPresent  
livenessProbe:  
exec:  
command:  
- /bin/sh  
- -ec  
- ETCDCTL\_API=3 etcdctl --endpoints=https://[192.168.22.9]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt --key=/etc/kubernetes/pki/etcd/healthcheck-client.key get foo failureThreshold: 8 initialDelaySeconds: 15 timeoutSeconds: 15 name: etcd-should-fail resources: {} volumeMounts:  
- mountPath: /var/lib/etcd  
name: etcd-data  
- mountPath: /etc/kubernetes/pki/etcd  
name: etcd-certs  
hostNetwork: true  
priorityClassName: system-cluster-critical  
volumes:  
- hostPath:  
path: /var/lib/etcd  
type: DirectoryOrCreate  
name: etcd-data  
- hostPath:  
path: /etc/kubernetes/pki/etcd  
type: DirectoryOrCreate

name: etcd-certs

status: {}

Explanation:

```
candidate@cli:~$ kubectl delete sa/podrunner -n qa
serviceaccount "podrunner" deleted
candidate@cli:~$ kubectl config use-context KSCS00201
Switched to context "KSCS00201".
candidate@cli:~$ ssh kscs00201-master
Warning: Permanently added '10.240.86.194' (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@kscs00201-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
root@kscs00201-master:~# systemctl daemon-reload
root@kscs00201-master:~# systemctl restart kubelet.service
root@kscs00201-master:~# systemctl enable kubelet.service
root@kscs00201-master:~# systemctl status kubelet.service
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
   Drop-In: /etc/systemd/system/kubelet.service.d
            └─10-kubeadm.conf
   Active: active (running) since Fri 2022-05-20 14:19:31 UTC; 29s ago
     Docs: https://kubernetes.io/docs/home/
  Main PID: 134205 (kubelet)
    Tasks: 16 (limit: 76200)
   Memory: 34.5M
    CGroup: /system.slice/kubelet.service
            └─134205 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kub

May 20 14:19:35 kscs00201-master kubelet[134205]: I0520 14:19:35.420825 134205 reconciler.>
May 20 14:19:35 kscs00201-master kubelet[134205]: I0520 14:19:35.420863 134205 reconciler.>
May 20 14:19:35 kscs00201-master kubelet[134205]: I0520 14:19:35.420907 134205 reconciler.>
May 20 14:19:35 kscs00201-master kubelet[134205]: I0520 14:19:35.420928 134205 reconciler.>
May 20 14:19:36 kscs00201-master kubelet[134205]: I0520 14:19:36.572353 134205 request.go:>
May 20 14:19:37 kscs00201-master kubelet[134205]: I0520 14:19:37.112347 134205 prober_mana>
May 20 14:19:37 kscs00201-master kubelet[134205]: E0520 14:19:37.185076 134205 kubelet.go:>
May 20 14:19:37 kscs00201-master kubelet[134205]: I0520 14:19:37.645798 134205 kubelet.go:>
May 20 14:19:38 kscs00201-master kubelet[134205]: I0520 14:19:38.184062 134205 kubelet.go:>
May 20 14:19:40 kscs00201-master kubelet[134205]: I0520 14:19:40.036042 134205 prober_mana>
lines 1-22/22 (END)
```

```
de Agent
et.service; enabled; vendor preset: enabled)
ce.d
```

5-20 14:19:31 UTC; 29s ago

```
trap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf kubeconfig=/etc/kubernetes/kubelet
5]: I0520 14:19:35.420825 134205 reconciler.go:221] "operationExecutor.VerifyControllerAtt
5]: I0520 14:19:35.420863 134205 reconciler.go:221] "operationExecutor.VerifyControllerAtt
5]: I0520 14:19:35.420907 134205 reconciler.go:221] "operationExecutor.VerifyControllerAtt
5]: I0520 14:19:35.420928 134205 reconciler.go:157] "Reconciler: start to sync state"
5]: I0520 14:19:36.572353 134205 request.go:665] Waited for 1.049946364s due to client-sid
5]: I0520 14:19:37.112347 134205 prober_manager.go:255] "Failed to trigger a manual run" p
5]: E0520 14:19:37.165076 134205 kubelet.go:1711] "Failed creating a mirror pod for" err="
5]: I0520 14:19:37.645798 134205 kubelet.go:1693] "Trying to delete pod" pod="kube-system/
5]: I0520 14:19:38.184062 134205 kubelet.go:1698] "Deleted mirror pod because it is outdat
5]: I0520 14:19:40.036042 134205 prober_manager.go:255] "Failed to trigger a manual run" p
~
~
lines 1-22/22 (END)
```

```
let.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/lib/kubelet/config.yaml --
o:221] "operationExecutor.VerifyControllerAttachedVolume started for volume \"kube-proxy\"
o:221] "operationExecutor.VerifyControllerAttachedVolume started for volume \"lib-modules\"
o:221] "operationExecutor.VerifyControllerAttachedVolume started for volume \"flannel-cfg\"
o:157] "Reconciler: start to sync state"
65] Waited for 1.049946364s due to client-side throttling, not priority and fairness, reques
er.go:255] "Failed to trigger a manual run" probe="Readiness"
711] "Failed creating a mirror pod for" err="pods \"kube-apiserver-kscs00201-master\" alrea
693] "Trying to delete pod" pod="kube-system/kube-apiserver-kscs00201-master" podUID=bb91e1
698] "Deleted mirror pod because it is outdated" pod="kube-system/kube-apiserver-kscs00201-
er.go:255] "Failed to trigger a manual run" probe="Readiness"
~
~
root@kscs00201-master:~# vim /var/lib/kubelet/config.yaml
```

```

apiVersion: kubelet.config.k8s.io/v1beta1
authentication:
  anonymous:
    enabled: false
  webhook:
    cacheTTL: 0s
    enabled: true
    clientCAFile: /etc/kubernetes/pki/ca.crt
authorization:
  mode: Webhook
  webhook:
    cacheAuthorizedTTL: 0s
    cacheUnauthorizedTTL: 0s
cgroupDriver: systemd
clusterDNS:

```

```

~
~
root@kscs00201-master:~# vim /var/lib/kubelet/config.yaml
root@kscs00201-master:~# vim /var/lib/kubelet/config.yaml
root@kscs00201-master:~# vim /etc/kubernetes/manifests/etcd.yaml
root@kscs00201-master:~# systemctl daemon-reload
root@kscs00201-master:~# systemctl restart kubelet.service
root@kscs00201-master:~# systemctl status kubelet.service

```

```

● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
   Drop-In: /etc/systemd/system/kubelet.service.d
            └─10-kubeadm.conf
   Active: active (running) since Fri 2022-05-20 14:22:29 UTC; 4s ago
     Docs: https://kubernetes.io/docs/home/
   Main PID: 135849 (kubelet)
    Tasks: 17 (limit: 76200)
   Memory: 38.0M
   CGroup: /system.slice/kubelet.service
           └─135849 /usr/bin/kubelet --bootstrap kubeconfig=/etc/kubernetes/bootstrap-kub

```

|                 |                  |                                        |        |              |
|-----------------|------------------|----------------------------------------|--------|--------------|
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330232 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330259 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330304 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330354 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330378 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330397 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330415 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330433 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330452 | 135849 | reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330463 | 135849 | reconciler.> |

```

lines 1-22/22 (END)

```

```
May 20 14:22:30 kscs00201-master kubelet[135849]: I0520 14:22:30.330463E 135849 reconciler.>  
root@kscs00201-master:~#  
root@kscs00201-master:~#  
root@kscs00201-master:~#  
root@kscs00201-master:~# exit  
logout  
Connection to 10.240.86.194 closed.  
candidate@cli:~$
```



## NEW QUESTION: 16

Context

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

Answer:

```
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-ppp
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
```



```
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=restrict-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 with --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-binding.yaml cluster-role-binding.yaml
candidate@cli:~$ vim /home/candidate/KSMV00102/cluster-role-binding.yaml cluster-role-binding.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:~$
```

**Valid CKS Dumps** shared by Actual4test.com for Helping Passing CKS Exam!

Actual4test.com now offer the **newest CKS exam dumps**, the Actual4test.com CKS exam questions have been updated and answers have been corrected get the newest Actual4test.com CKS dumps with Test Engine here:

[https://www.actual4test.com/CKS\\_examcollection.html](https://www.actual4test.com/CKS_examcollection.html) (66 Q&As Dumps, **30%OFF** Special Discount: **Freepdfdumps**)

### NEW QUESTION: 17

Create a new NetworkPolicy named deny-all in the namespace testing which denies all traffic of type ingress and egress traffic

#### Answer:

You can create a "default" isolation policy for a namespace by creating a NetworkPolicy that selects all pods but does not allow any ingress traffic to those pods.

---

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

```
kind: NetworkPolicy
```

```
metadata:
```

```
name: default-deny-ingress
```

```
spec:
```

```
podSelector: {}
```

```
policyTypes:
```

```
- Ingress
```

You can create a "default" egress isolation policy for a namespace by creating a NetworkPolicy that selects all pods but does not allow any egress traffic from those pods.

---

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

```
kind: NetworkPolicy
```

```
metadata:
```

```
name: allow-all-egress
```

```
spec:
```

```
podSelector: {}
```

```
egress:
```

```
- {}
```

```
policyTypes:
```

```
- Egress
```

Default deny all ingress and all egress traffic

You can create a "default" policy for a namespace which prevents all ingress AND egress traffic by creating the following NetworkPolicy in that namespace.

---

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

```
kind: NetworkPolicy
```

```
metadata:
```

```
name: default-deny-all
```

```
spec:
```

```
podSelector: {}
```

```
policyTypes:
```

```
- Ingress
```

```
- Egress
```

This ensures that even pods that aren't selected by any other NetworkPolicy will not be allowed ingress or egress traffic.

### NEW QUESTION: 18

You can switch the cluster/configuration context using the following command: [desk@cli] \$  
kubectl config use-context prod-account Context: A Role bound to a Pod's ServiceAccount grants overly permissive permissions. Complete the following tasks to reduce the set of permissions.  
Task: Given an existing Pod named web-pod running in the namespace database. 1. Edit the existing Role bound to the Pod's ServiceAccount test-sa to only allow performing get operations, only on resources of type Pods. 2. Create a new Role named test-role-2 in the namespace database, which only allows performing update operations, only on resources of type statuefulsets. 3. Create a new RoleBinding named test-role-2-bind binding the newly created Role to the Pod's ServiceAccount. Note: Don't delete the existing RoleBinding.

### Answer:

```
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:~$
```



## NEW QUESTION: 19

Create a User named john, create the CSR Request, fetch the certificate of the user after approving it.

Create a Role name john-role to list secrets, pods in namespace john

Finally, Create a RoleBinding named john-role-binding to attach the newly created role john-role to the user john in the namespace john. To Verify: Use the kubectl auth CLI command to verify the permissions.

**Answer:**

se kubectl to create a CSR and approve it.

Get the list of CSRs:

```
kubectl get csr
```

Approve the CSR:

```
kubectl certificate approve myuser
```

Get the certificate

Retrieve the certificate from the CSR:

```
kubectl get csr/myuser -o yaml
```

here are the role and role-binding to give john permission to create NEW\_CRD resource:

```
kubectl apply -f roleBindingJohn.yaml --as=john
```

rolebinding.rbac.authorization.k8s.io/john\_external-resource-rb created kind: RoleBinding

apiVersion: rbac.authorization.k8s.io/v1 metadata:

name: john\_crd

namespace: development-john

subjects:

- kind: User

name: john

apiGroup: rbac.authorization.k8s.io

roleRef:

kind: ClusterRole

name: crd-creation

kind: ClusterRole

apiVersion: rbac.authorization.k8s.io/v1

metadata:

name: crd-creation

rules:

- apiGroups: ["kubernetes-client.io/v1"]

resources: ["NEW\_CRD"]

verbs: ["create, list, get"]

## **NEW QUESTION: 20**

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.

### **Answer:**

```
ssh-add ~/.ssh/tempprivate
```

```
eval "$(ssh-agent -s)"
```

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

Ensure that the  
--authorization  
-mode  
1.2.7 argument is not FAIL  
set to  
AlwaysAllow  
Ensure that the  
--authorization  
1.2.8 -mode FAIL  
argument  
includes Node  
Ensure that the  
--authorization  
1.2.9 -mode FAIL  
argument  
includes RBAC



freendtdumps.com

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

Ensure that the  
anonymous-auth  
4.2.1 th FAIL  
argument is set  
to false  
Ensure that the  
authorization  
4.2.2 -mode FAIL  
argument is not  
set to  
AlwaysAllow



Use Webhook  
authentication/authorization  
where possible.



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

Ensure that the  
2.2 --client-cert-auth FAIL  
argument is set  
to true



## Answer:

```
candidate@cli:~$ kubectl delete sa/podrunner -n qa
serviceaccount "podrunner" deleted
candidate@cli:~$ kubectl config use-context KSCS00201
Switched to context "KSCS00201".
candidate@cli:~$ ssh kscs00201-master
Warning: Permanently added '10.240.86.194' (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@kscs00201-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
root@kscs00201-master:~# systemctl daemon-reload
root@kscs00201-master:~# systemctl restart kubelet.service
root@kscs00201-master:~# systemctl enable kubelet.service
root@kscs00201-master:~# systemctl status kubelet.service
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
   Drop-In: /etc/systemd/system/kubelet.service.d
            └─10-kubeadm.conf
   Active: active (running) since Fri 2022-05-20 14:19:31 UTC; 29s ago
     Docs: https://kubernetes.io/docs/home/
  Main PID: 134205 (kubelet)
    Tasks: 16 (limit: 76200)
   Memory: 36.5M
   CGroup: /system.slice/kubelet.service
            └─134205 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kub

May 20 14:19:35 kscs00201-master kubelet[134205]: I0520 14:19:35.420825 134205 reconciler.>
May 20 14:19:35 kscs00201-master kubelet[134205]: I0520 14:19:35.420863 134205 reconciler.>
May 20 14:19:35 kscs00201-master kubelet[134205]: I0520 14:19:35.420907 134205 reconciler.>
May 20 14:19:35 kscs00201-master kubelet[134205]: I0520 14:19:35.420928 134205 reconciler.>
May 20 14:19:36 kscs00201-master kubelet[134205]: I0520 14:19:36.572353 134205 request.go:>
May 20 14:19:37 kscs00201-master kubelet[134205]: I0520 14:19:37.112347 134205 prober_mana>
May 20 14:19:37 kscs00201-master kubelet[134205]: E0520 14:19:37.185076 134205 kubelet.go:>
May 20 14:19:37 kscs00201-master kubelet[134205]: I0520 14:19:37.645798 134205 kubelet.go:>
May 20 14:19:38 kscs00201-master kubelet[134205]: I0520 14:19:38.184062 134205 kubelet.go:>
May 20 14:19:40 kscs00201-master kubelet[134205]: I0520 14:19:40.036042 134205 prober_mana>
lines 1-22/22 (END)
```

```
de Agent
et.service; enabled; vendor preset: enabled)
ce.d
```

5-20 14:19:31 UTC; 29s ago

```
trap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf kubeconfig=/etc/kubernetes/kubelet
5]: I0520 14:19:35.420825 134205 reconciler.go:221] "operationExecutor.VerifyControllerAtt
5]: I0520 14:19:35.420863 134205 reconciler.go:221] "operationExecutor.VerifyControllerAtt
5]: I0520 14:19:35.420907 134205 reconciler.go:221] "operationExecutor.VerifyControllerAtt
5]: I0520 14:19:35.420928 134205 reconciler.go:157] "Reconciler: start to sync state"
5]: I0520 14:19:36.572353 134205 request.go:665] "Waited for 1.049946364s due to client-sid
5]: I0520 14:19:37.112347 134205 prober_manager.go:255] "Failed to trigger a manual run" p
5]: E0520 14:19:37.165076 134205 kubelet.go:1711] "Failed creating a mirror pod for" err="
5]: I0520 14:19:37.645798 134205 kubelet.go:1693] "Trying to delete pod" pod="kube-system/
5]: I0520 14:19:38.184062 134205 kubelet.go:1698] "Deleted mirror pod because it is outdat
5]: I0520 14:19:40.036042 134205 prober_manager.go:255] "Failed to trigger a manual run" p
~
~
lines 1-22/22 (END)
```

```
let.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/lib/kubelet/config.yaml --
o:221] "operationExecutor.VerifyControllerAttachedVolume started for volume \"kube-proxy\"
o:221] "operationExecutor.VerifyControllerAttachedVolume started for volume \"lib-modules\"
o:221] "operationExecutor.VerifyControllerAttachedVolume started for volume \"flannel-cfg\"
o:157] "Reconciler: start to sync state"
65] "Waited for 1.049946364s due to client-side throttling, not priority and fairness, reques
er.go:255] "Failed to trigger a manual run" probe="Readiness"
711] "Failed creating a mirror pod for" err="pods \"kube-apiserver-kscs00201-master\" alrea
693] "Trying to delete pod" pod="kube-system/kube-apiserver-kscs00201-master" podUID=bb91e1
698] "Deleted mirror pod because it is outdated" pod="kube-system/kube-apiserver-kscs00201-
er.go:255] "Failed to trigger a manual run" probe="Readiness"
~
~
root@kscs00201-master:~# vim /var/lib/kubelet/config.yaml
```

```

apiVersion: kubelet.config.k8s.io/v1beta1
authentication:
  anonymous:
    enabled: false
  webhook:
    cacheTTL: 0s
    enabled: true
    x509:
      clientCAFile: /etc/kubernetes/pki/ca.crt
authorization:
  mode: Webhook
  webhook:
    cacheAuthorizedTTL: 0s
    cacheUnauthorizedTTL: 0s
cgroupDriver: systemd
clusterDNS:

```

```

~
~
root@kscs00201-master:~# vim /var/lib/kubelet/config.yaml
root@kscs00201-master:~# vim /var/lib/kubelet/config.yaml
root@kscs00201-master:~# vim /etc/kubernetes/manifests/etcd.yaml
root@kscs00201-master:~# systemctl daemon-reload
root@kscs00201-master:~# systemctl restart kubelet.service
root@kscs00201-master:~# systemctl status kubelet.service

```

```

● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
   Drop-In: /etc/systemd/system/kubelet.service.d
            └─10-kubeadm.conf
   Active: active (running) since Fri 2022-05-20 14:22:29 UTC; 4s ago
     Docs: https://kubernetes.io/docs/home/
   Main PID: 135849 (kubelet)
    Tasks: 17 (limit: 76200)
   Memory: 38.0M
   CGroup: /system.slice/kubelet.service
           └─135849 /usr/bin/kubelet --bootstrap kubeconfig=/etc/kubernetes/bootstrap-kub

```

|                 |                  |                                        |                     |
|-----------------|------------------|----------------------------------------|---------------------|
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330232 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330259 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330304 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330354 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330378 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330397 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330415 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330433 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330452 | 135849 reconciler.> |
| May 20 14:22:30 | kscs00201-master | kubelet[135849]: I0520 14:22:30.330463 | 135849 reconciler.> |

```

lines 1-22/22 (END)

```

```
May 20 14:22:30 kscs00201-master kubelet[135849]: I0520 14:22:30.330463E 135849 reconciler.>
root@kscs00201-master:~#
root@kscs00201-master:~#
root@kscs00201-master:~#
root@kscs00201-master:~# exit
logout
Connection to 10.240.86.194 closed.
candidate@cli:~$
```

## NEW QUESTION: 22

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 manifest, it should get failed.

POD Manifest:

apiVersion: v1

kind: Pod

metadata:

name:

spec:

containers:

- name:

image:

volumeMounts:

- name:

mountPath:

volumes:

- name:

secret:

secretname:

**Answer:**

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

**Valid CKS Dumps** shared by Actual4test.com for Helping Passing CKS Exam!

Actual4test.com now offer the **newest CKS exam dumps**, the Actual4test.com CKS exam **questions have been updated** and **answers have been corrected** get the **newest**

Actual4test.com CKS dumps with Test Engine here:

[https://www.actual4test.com/CKS\\_examcollection.html](https://www.actual4test.com/CKS_examcollection.html) (**66** Q&As Dumps, **30%OFF** Special Discount: **Freepdfdumps**)