

## CKS Dumps

### Certified Kubernetes Security Specialist (CKS) Exam

<https://www.certleader.com/CKS-dumps.html>



**NEW QUESTION 1**

Create a network policy named restrict-np to restrict to pod nginx-test running in namespace testing. Only allow the following Pods to connect to Pod nginx-test:

- \* 1. pods in the namespace default
- \* 2. pods with label version:v1 in any namespace.

Make sure to apply the network policy.

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

Send us your Feedback on this.

**NEW QUESTION 2**

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).

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

Send us your feedback on it.

**NEW QUESTION 3**

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://acme.local.8081/image\\_policy](https://acme.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 the latest.

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

Send us your feedback on it.

**NEW QUESTION 4**

Given an existing Pod named test-web-pod running in the namespace test-system

Edit the existing Role bound to the Pod's Service Account named sa-backend to only allow performing get operations on endpoints.

Create a new Role named test-system-role-2 in the namespace test-system, which can perform patch operations, on resources of type statefulsets.

Create a new RoleBinding named test-system-role-2-binding binding the newly created Role to the Pod's ServiceAccount sa-backend.

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

Send us your feedback on this.

**NEW QUESTION 5**

Service is running on port 389 inside the system, find the process-id of the process, and stores the names of all the open-files inside the /candidate/KH77539/files.txt, and also delete the binary.

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

Send us your feedback on it.

**NEW QUESTION 6**

Create a RuntimeClass named gvisor-rc using the prepared runtime handler named runsc. Create a Pods of image Nginx in the Namespace server to run on the gVisor runtime class

- A. Mastered

B. Not Mastered

**Answer:** A

**Explanation:**

Install the Runtime Class for gVisor

```
{ # Step 1: Install a RuntimeClass
```

```
cat <<EOF | kubectl apply -f -
```

```
apiVersion: node.k8s.io/v1beta1
```

```
kind: RuntimeClass
```

```
metadata:
```

```
name: gvisor
```

```
handler: runsc
```

```
EOF
```

```
}
```

Create a Pod with the gVisor Runtime Class

```
{ # Step 2: Create a pod
```

```
cat <<EOF | kubectl apply -f -
```

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
name: nginx-gvisor
```

```
spec:
```

```
runtimeClassName: gvisor
```

```
containers:
```

```
- name: nginx
```

```
image: nginx
```

```
EOF
```

```
}
```

Verify that the Pod is running

```
{ # Step 3: Get the pod
```

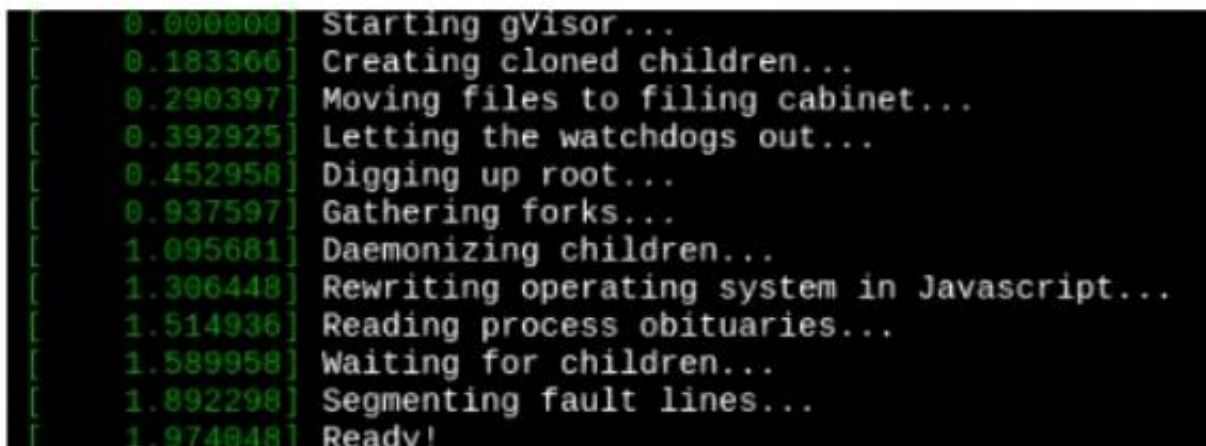
```
kubectl get pod nginx-gvisor -o wide
```

```
}
```

**NEW QUESTION 7**

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. Verify: Exec the pods and run the dmesg, you will see output like this:



```
[ 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!
```

A. Mastered

B. Not Mastered

**Answer:** A

**Explanation:**

Send us your feedback on it.

**NEW QUESTION 8**

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.

A. Mastered

B. Not Mastered

**Answer:** A

**Explanation:**

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 certificateRetrieve the certificate from the CSR:

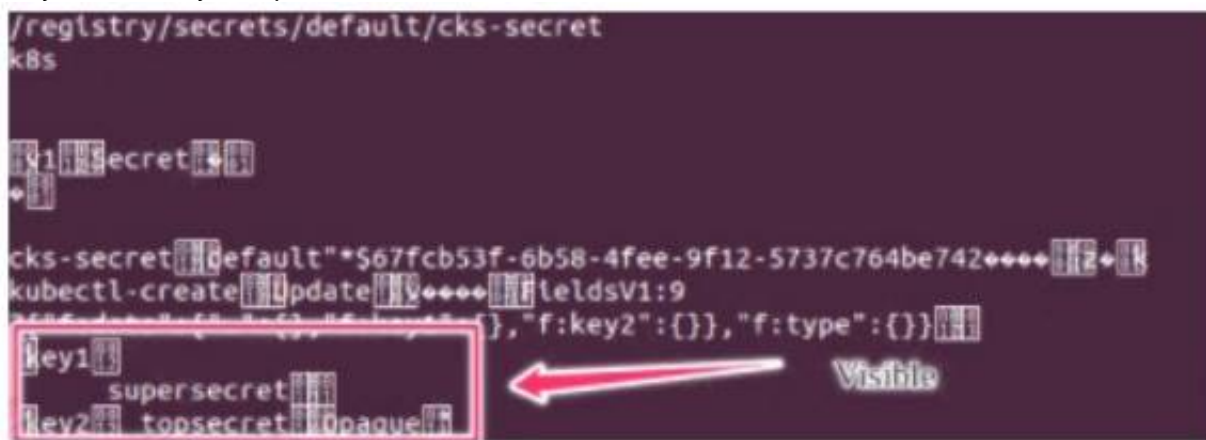
```
kubectl get csr/myuser -o yaml
```

here are the role and role-binding to give john permission to create NEW\_CRD resource: kubectlapply-froleBindingJohn.yaml--as=john

```
rolebinding.rbac.authorization.k8s.io/john_external-rosource-rbcreated
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 9

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



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.

- A. Mastered
- B. Not Mastered

**Answer:** A

#### Explanation:

Send us your feedback on it.

### NEW QUESTION 10

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. Mastered
- B. Not Mastered

**Answer:** A

#### Explanation:

Send us your suggestion on it.

### NEW QUESTION 10

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:

- \* apiVersion: v1
- \* kind: Pod
- \* metadata:
- \* name:

\* spec:  
\* containers:  
\* - name:  
\* image:  
\* volumeMounts:  
\* - name:  
\* mountPath:  
\* volumes:  
\* - name:  
\* secret:  
\* secretName:

A. Mastered  
B. Not Mastered

**Answer:** A

**Explanation:**

```
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
```

### NEW QUESTION 13

Before Making any changes build the Dockerfile with tag base:v1 Now Analyze and edit the given Dockerfile(based on ubuntu 16:04)  
Fixing two instructions present in the file, Check from Security Aspect and Reduce Size point of view.

Dockerfile:

```
FROM ubuntu:latest
RUN apt-getupdate -y
RUN apt install nginx -y
COPY entrypoint.sh /
RUN useradd ubuntu
ENTRYPOINT ["/entrypoint.sh"]
USER ubuntu
entrypoint.sh
#!/bin/bash
echo"Hello from CKS"
```

After fixing the Dockerfile, build the docker-image with the tag base:v2 To Verify: Check the size of the image before and after the build.

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

Send us your feedback on it.

**NEW QUESTION 15**

On the Cluster worker node, enforce the prepared AppArmor profile

```
#include<tunables/global>
profile nginx-deny flags=(attach_disconnected) {
#include<abstractions/base>
file,
# Deny all file writes.
deny/** w,
}
EOF'
```

Edit the prepared manifest file to include the AppArmor profile.

```
apiVersion: v1
kind: Pod
metadata:
name: apparmor-pod
spec:
containers:
- name: apparmor-pod
image: nginx
```

Finally, apply the manifests files and create the Pod specified on it. Verify: Try to make a file inside the directory which is restricted.

- A. Mastered
- B. Not Mastered

**Answer:** A

**Explanation:**

Send us your Feedback on this.

**NEW QUESTION 17**

.....

## Thank You for Trying Our Product

\* 100% Pass or Money Back

All our products come with a 90-day Money Back Guarantee.

\* One year free update

You can enjoy free update one year. 24x7 online support.

\* Trusted by Millions

We currently serve more than 30,000,000 customers.

\* Shop Securely

All transactions are protected by VeriSign!

**100% Pass Your CKS Exam with Our Prep Materials Via below:**

<https://www.certleader.com/CKS-dumps.html>