



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.passapply.com/cks.html>

100% Passing Guarantee
100% Money Back Assurance

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

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





QUESTION 1

Cluster: scanner

Master node: controlplane

Worker node: worker1

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

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

Given:

You may use Trivy's documentation.

Task:

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

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

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

A. See the explanation below

B. Placeholder

Correct Answer: A

QUESTION 2



```
candidate@cli:~$ kubectl config use-context KRSR00602
Switched to context "KRSR00602".
candidate@cli:~$ ssh krsr00602-master
Warning: Permanently added '10.240.86.243' (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@krsr00602-master:~# cat /etc/kubernetes/logpolicy/sample-policy.yaml
---
apiVersion: audit.k8s.io/v1
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"
# Edit form here below
root@krsr00602-master:~# vim /etc/kubernetes/logpolicy/sample-policy.yaml
```



```
- "/api*" # Wildcard matching.
- "/version"
# Edit form here below
- level: RequestResponse
  resources:
  - group: ""
    resources: ["cronjobs"]
- level: Request
  resources:
  - group: "" # core API group
    resources: ["pods"]
    namespaces: ["webapps"]
# Log configmap and secret changes in all other namespaces at the Metadata level.
- level: Metadata
  resources:
  - group: "" # core API group
    resources: ["secrets", "configmaps"]

# A catch-all rule to log all other requests at the Metadata level.
- level: Metadata
  # Long-running requests like watches that fall under this rule will not
  # generate an audit event in RequestReceived.
  omitStages:
  - "RequestReceived"
```



```
- "/version"
# Edit form here below
- level: RequestResponse
  resources:
    - group: ""
      resources: ["cronjobs"]
- level: Request
  resources:
    - group: "" # core API group
      resources: ["pods"]
      namespaces: ["webapps"]
# Log configmap and secret changes in all other namespaces at the Metadata level.
- level: Metadata
  resources:
    - group: "" # core API group
      resources: ["secrets", "configmaps"]

# A catch-all rule to log all other requests at the Metadata level.
- level: Metadata
  # Long-running requests like watches that fall under this rule will not
  # generate an audit event in RequestReceived.
  omitStages:
    - "RequestReceived"
root@ksrs00602-master:~# vim /etc/kubernetes/logpolicy/sample-policy.yaml
root@ksrs00602-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
```

```
labels:
  component: kube-apiserver
  tier: control-plane
name: kube-apiserver
namespace: kube-system
spec:
  containers:
    - command:
      - kube-apiserver
      - --advertise-address=10.240.86.243
      - --allow-privileged=true
      - --audit-policy-file=/etc/kubernetes/logpolicy/sample-policy.yaml
      - --audit-log-path=/var/log/kubernetes/kubernetes-logs.txt
      - --audit-log-maxbackup=1
      - --audit-log-maxage=30
      - --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
```

```
# A catch-all rule to log all other requests at the Metadata level.
- level: Metadata
  # Long-running requests like watches that fall under this rule will not
  # generate an audit event in RequestReceived.
  omitStages:
    - "RequestReceived"
root@ksrs00602-master:~# vim /etc/kubernetes/logpolicy/sample-policy.yaml
root@ksrs00602-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
root@ksrs00602-master:~# systemctl daemon-reload
root@ksrs00602-master:~# systemctl restart kubelet.service
root@ksrs00602-master:~# systemctl enable kubelet
root@ksrs00602-master:~# exit
logout
Connection to 10.240.86.243 closed.
candidate@cli:~$
```



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

```
[desk@cli] $ kubectl 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 authn/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

A. See the explanation below

B. Placeholder

Correct Answer: A

```
worker1 $ vim /var/lib/kubelet/config.yaml uk.co.certification.simulator.questionpool.PList@132b77a0 worker1 $  
systemctl restart kubelet. # To reload kubelet configssh to master1master1 $ vim /etc/kubernetes/manifests/kube-  
apiserver.yaml- -- authorizationmode=Node,RBACmaster1 $ vim /etc/kubernetes/manifests/etcd.yaml- --client-cert-  
auth=true
```

```
Explanationssh to worker1worker1 $ 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 configssh to  
master1master1 $ vim /etc/kubernetes/manifests/kube-apiserver.yaml
```




```
apiVersion: v1
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
```

QUESTION 3



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

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

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

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

    deny network raw,

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




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

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

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.

A. See the explanation below

B. Placeholder

Correct Answer: A

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

```
uk.co.certification.simulator.questionpool.PList@11602760
```

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

 Add these

```
uk.co.certification.simulator.questionpool.PList@11602c70
```

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

```
[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 and 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
```



QUESTION 4



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

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

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

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

  deny network raw,

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

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

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




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

1.

Ensure that the RotateKubeletServerCertificate argument is set to true.

2.

Ensure that the admission control plugin PodSecurityPolicy is set.

3.

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

1.

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

2.

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

1.

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

2.

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

Hint: Take the use of Tool Kube-Bench

A. See the below.

B. Placeholder

Correct Answer: A

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

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

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

-command:

-kube-controller-manager + - --feature-gates=RotateKubeletServerCertificate=true image:

gcr.io/google_containers/kubelet-amd64:v1.6.0 livenessProbe: failureThreshold: 8 httpGet: host: 127.0.0.1 path:

/healthz port: 6443 scheme: HTTPS initialDelaySeconds: 15 timeoutSeconds: 15 name: kubelet resources: requests:

cpu: 250m volumeMounts:

-



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

-

mountPath: /etc/ssl/certs name: certs

-

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

-

hostPath: path: /etc/kubernetes name: k8s

-

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

-

hostPath: path: /etc/pki name: pki

b.

Ensure that the admission control plugin PodSecurityPolicy is set.

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

tests:

test_items:

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

compare:

op: has

value: "PodSecurityPolicy"

set: true

remediation: |

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

Then, edit the API server pod specification file \$apiserverconf

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

value that includes PodSecurityPolicy :

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

Then restart the API Server.

scored: true



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

tests: test_items:

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

set: true

remediation: |

Follow the Kubernetes documentation and setup the TLS connection between the

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

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

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

`--kubelet-certificate-authority=`

scored: true

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

a.

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

b.

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

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

QUESTION 5



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

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

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



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

A. See the explanation below:

B. Placeholder

Correct Answer: A

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 defined 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

[Latest CKS Dumps](#)

[CKS VCE Dumps](#)

[CKS Exam Questions](#)