CKS Q&As

Certified Kubernetes Security Specialist (CKS) Exam

Pass Linux Foundation CKS Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

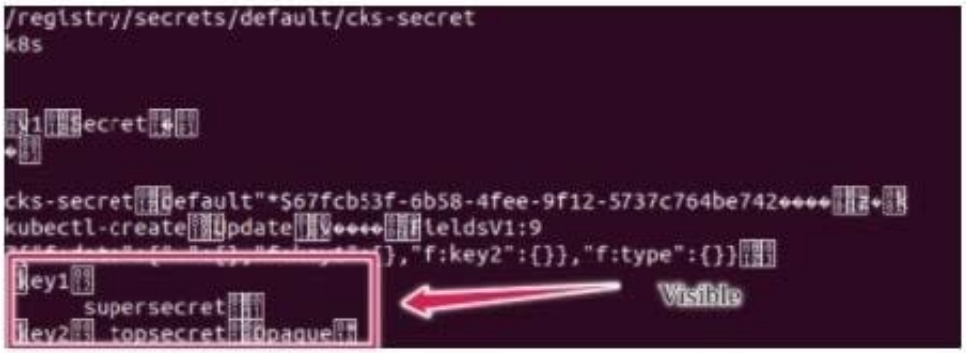
https://www.geekcert.com/cks.html

100% Passing Guarantee
100% Money Back Assurance

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

QUESTION 1

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. See explanation below.

B. PlaceHolder

Correct Answer: A

1.

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

2.

ETCD secrets are stored at the path /registry/secrets/$namespace/$secret on the master node.

3.

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

QUESTION 2

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.

A. See the below:

B. PlaceHolder

Correct Answer: A

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/ -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.

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.yamlAdd 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

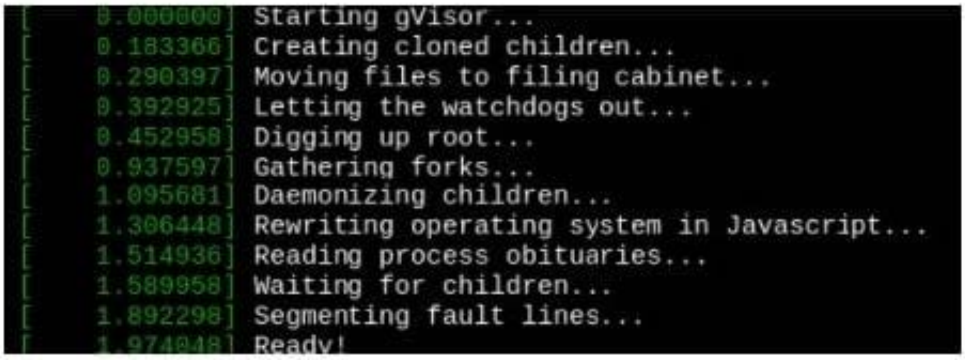
Create a RuntimeClass named untrusted using the prepared runtime handler named runsc.

Create a Pods of image alpine:3.13.2 in the Namespace default to run on the gVisor runtime class.

A. See the explanation below:

B. PlaceHolder

Correct Answer: A



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:~$
```

1.

Retrieve the content of the existing secret named default-token-xxxxx in the testing namespace.

Store the value of the token in the token.txt

2.

Create a new secret named test-db-secret in the DB namespace with the following content:

username: mysql password: password@123

Create the Pod name test-db-pod of image nginx in the namespace db that can access test-db-secret via a volume at path /etc/mysql-credentials

A. See the explanation below:

B. PlaceHolder

Correct Answer: A

To add a Kubernetes cluster to your project, group, or instance:

1.

Navigate to your:

2.

Click Add Kubernetes cluster.

3.

Click the Add existing cluster tab and fill in the details:

Get the API URL by running this command:

kubectl cluster-info | grep -E \\'Kubernetes master|Kubernetes control plane\\' | awk \\'/http/ {print $NF}\\'

uk.co.certification.simulator.questionpool.PList@113e1f90

kubectl get secret -o jsonpath="{[\\'data\\'][\\'ca\.crt\\']}"