Linux-Foundation

Exam Questions CKS

Certified Kubernetes Security Specialist (CKS) Exam

NEW QUESTION 1
Create a new NetworkPolicy named deny-all in the namespace testing which denies all traffic of type ingress and egress traffic

A. Mastered
B. Not Mastered

Answer: A

Explanation:
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 trafficYou 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 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
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 thAe PI server:
* a. Ensure the --authorization-mode argument includes RBAC
* b. Ensure the --authorization-mode argument includes Node
* c. Ensure that the --profiling argumentissettofalse
Fix all of the following violations that were found against the Kubelet:
* a. Ensure the --anonymous-auth argumentissettofalse.
* b. Ensure that the --authorization-mode argumentissetto 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

A. Mastered
B. Not Mastered

Answer: A

Explanation:

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 - BuildtimeKubernetesapiVersion: 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 fil/eetc/kubernetes/manifests/kube-apiserver.yaml on the master node and set the --authorization-mode parameter to a value that includeNs ode.

--authorization-mode=Node,RBAC

Audit:

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

Expected result:

'Node,RBAC' has 'Node'

Ensure that the --profiling argumentissettofalse

Remediation: Edit the API server pod specification fil/eetc/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 argumentissettofalse.

Remediation: If using a Kubelet config file, edit the file to set authenticationa:nonymous: 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

--anonymous-auth=false

variable.

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 argumentissetto 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 - BuildtimeKubernetesapiVersion: 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: {}
```

NEW QUESTION 4
Use the kubesec docker images to scan the given YAML manifest, edit and apply the advised changes, and passed with a score of 4 points.
kubesec-test.yaml

```
 apiVersion: v1
 kind: Pod
 metadata:
 name: kubesec-demo
 spec:
 containers:
 - name: kubesec-demo
 image: gcr.io/google-samples/node-hello:1.0
 securityContext:
 readOnlyRootFilesystem:true
```

Hint: docker run -i kubesec/kubesec:512c5e0 scan /dev/stdin< kubesec-test.yaml

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Send us your feedback on it.

NEW QUESTION 5

Using the runtime detection tool Falco, Analyse the container behavior for at least 20 seconds, using filters that detect newly spawning and executing processes in a single container of Nginx.
store the incident file art /opt/falco-incident.txt, containing the detected incidents. one per line, in the format [timestamp],[uid],[processName]

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Send us your feedback on it.

NEW QUESTION 6
Create a Pod name Nginx-pod inside the namespace testing, Create a service for the Nginx-pod named nginx-svc, using the ingress of your choice, run the ingress on tls, secure port.

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Send us your feedback on it.

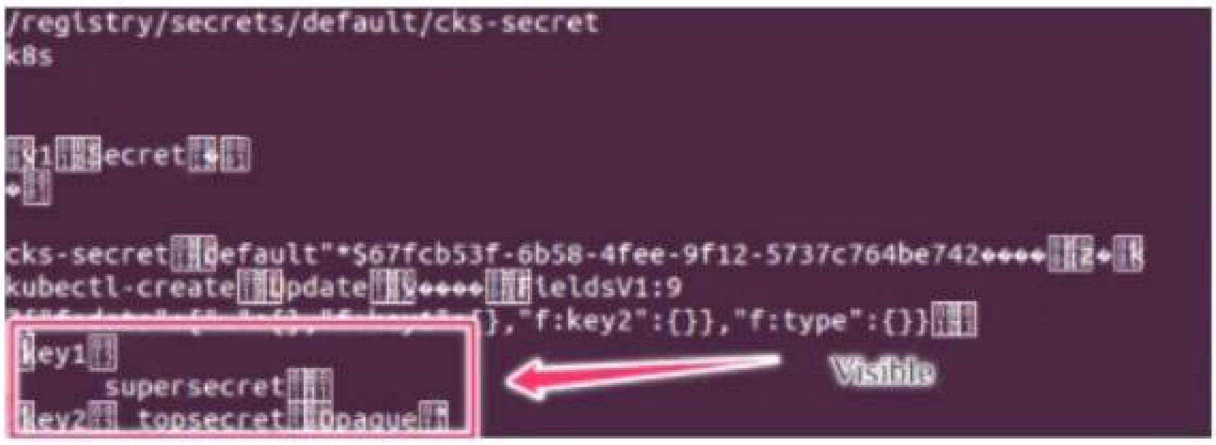
NEW QUESTION 7
A container image scanner is set up on the cluster. Given an incomplete configuration in the directory
/etc/kubernetes/confcontrol and a functional container image scanner with HTTPS endpoint https://test-server.local.8081/image_policy
* 1. Enable the admission plugin.
* 2. Validate the control configuration and change it to implicit deny.

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Finally, test the configuration by deploying the pod having the image tag as latest. Send us your Feedback on this.

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


v1 Secret


cks-secret default"*$67fcb53f-6b58-4fee-9f12-5737c764be742 z k
kubectl-create Update V fieldsV1:9
},"f:key2":{}},"f:type":{}}
key1
     supersecret
key2 topsecret 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.

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Send us your feedback on it.

NEW QUESTION 10
......