Exam Questions CKS

Certified Kubernetes Security Specialist (CKS) Exam

https://www.2passeasy.com/dumps/CKS/

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
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. Mastered
B. Not Mastered

Answer: A

Explanation:
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 thi flag disables log backend. - means standard out

- --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 location of the policy file and log file, so that audit records are persisted. For example:-hostPath-to the

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

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
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 argumentissettotrue.
* b. Ensure that the admission control plugin PodSecurityPolicyisset.
* c. Ensure that the --kubelet-certificate-authority argumentissetasappropriate.
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 argumentisnotsettotrue
* b. Ensure that the --peer-auto-tls argumentisnotsettotrue
Hint: Take the use of Tool Kube-Bench

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Fix all of the following violations that were found against the API server:
* a. Ensure that the RotateKubeletServerCertificate argumentissettotrue.

```
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 PodSecurityPolicyisset.

```
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 argumentissetasappropriate.
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 argumentisnotsettotrue
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 argumentisnotsettotrue
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

NEW QUESTION 5
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 6
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 7
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 8

Analyze and edit the given Dockerfile

```
 FROM ubuntu:latest
 RUN apt-getupdate -y
 RUN apt-install nginx -y
 COPY entrypoint.sh /
 ENTRYPOINT ["/entrypoint.sh"]
 USER ROOT
```

Fixing two instructions present in the file being prominent security best practice issues

Analyze and edit the deployment manifest file

```
 apiVersion: v1
 kind: Pod
 metadata:
 name: security-context-demo-2
 spec:
securityContext:
 runAsUser: 1000
 containers:
- name: sec-ctx-demo-2
image: gcr.io/google-samples/node-hello:1.0
 securityContext:
 runAsUser: 0
privileged:True
allowPrivilegeEscalation:false
```

Fixing two fields present in the file being prominent security best practice issues

Don't add or remove configuration settings; only modify the existing configuration settings

A. Mastered
B. Not Mastered

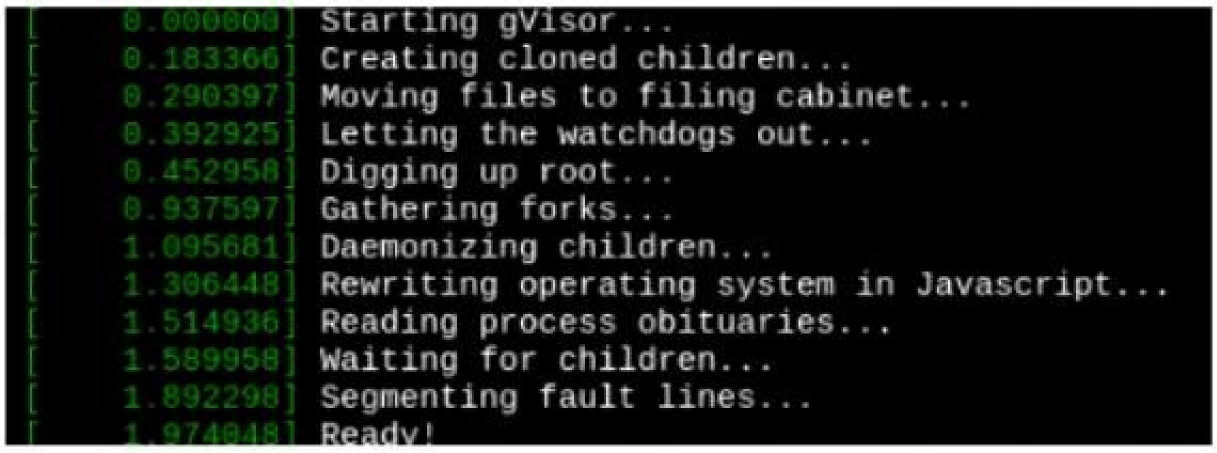
Answer: A

Explanation:
Whenever you need an unprivileged user for any of the tasks, use user test-user with the user id 5487 Send us the Feedback on it.

NEW QUESTION 9

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

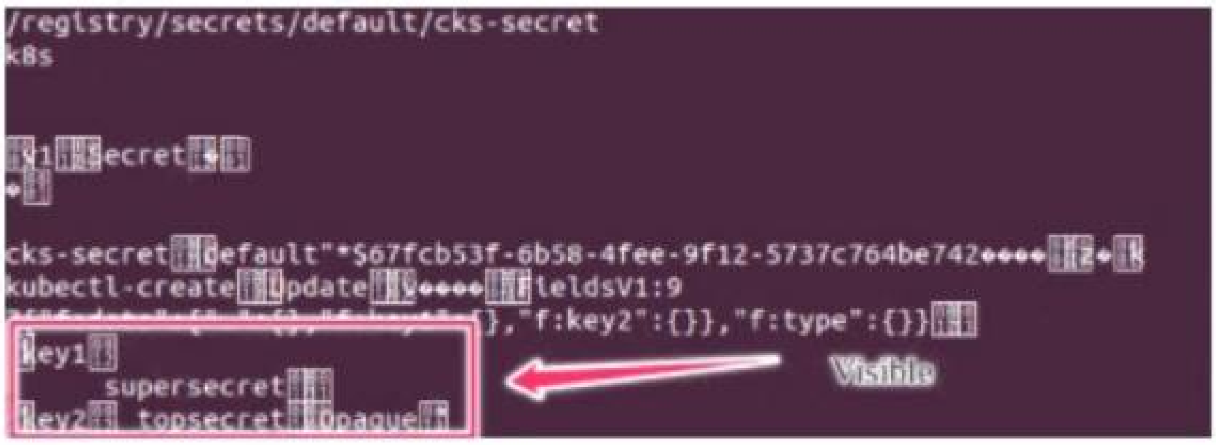
Answer: A

Explanation:
Send us your feedback on it.

NEW QUESTION 10

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
```

Visible

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
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 15

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 19

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 20

......