LinuxFoundation.CKS.v2022-01-21.q10

Exam Code:	CKS
Exam Name:	Certified Kubernetes Security Specialist (CKS)
Certification Provider:	Linux Foundation
Free Question Number:	10
Version:	v2022-01-21
# of views:	1134
# of Questions views:	100
https://www.freepdfdumps.com/LinuxFoundation.CKS.v2022-01-21.q10.html	

NEW QUESTION: 1
SIMULATION
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. Send us the Feedback on it.
Answer: A (LEAVE A REPLY)

NEW QUESTION: 2
SIMULATION
Create a new NetworkPolicy named deny-all in the namespace testing which denies all traffic of type ingress and egress traffic
Answer:
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 traffic
You 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: 3
SIMULATION
Enable audit logs in the cluster, To Do so, enable the log backend, and ensure that

1. logs are stored at /var/log/kubernetes/kubernetes-logs.txt.
2. Log files are retained for 5 days.
3. at maximum, a number of 10 old audit logs files are retained.
Edit and extend the basic policy to log:
1. Cronjobs changes at RequestResponse
2. Log the request body of deployments changes in the namespace kube-system.
3. Log all other resources in core and extensions at the Request level.
4. Don't log watch requests by the "system:kube-proxy" on endpoints or
A. Send us the Feedback on it.
Answer: A (LEAVE A REPLY)

NEW QUESTION: 4
SIMULATION
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 argument is set to true.
b. Ensure that the admission control plugin PodSecurityPolicy is set.
c. Ensure that the --kubelet-certificate-authority argument is set as appropriate.
Fix all of the following violations that were found against the Kubelet:- a. Ensure the --anonymous-auth argument is set to false.
b. Ensure that the --authorization-mode argument is set to 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
b. Ensure that the --peer-auto-tls argument is not set to true
Hint: Take the use of Tool Kube-Bench
Answer:
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=<ca-string>
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 master node 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 master node and either remove the --peer-auto-tls parameter or set it to false. --peer-auto-tls=false

NEW QUESTION: 5
SIMULATION
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 the --authorization-mode argument includes RBAC b. Ensure the --authorization-mode argument includes Node c. Ensure that the --profiling argument is set to false Fix all of the following violations that were found against the Kubelet:- a. Ensure the --anonymous-auth argument is set to false.
b. Ensure that the --authorization-mode argument is set to 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
Answer:
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 - Buildtime
Kubernetes
apiVersion: 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 file /etc/kubernetes/manifests/kube-apiserver.yaml on the master node and set the --authorization-mode parameter to a value that includes Node.
--authorization-mode=Node,RBAC
Audit:
/bin/ps -ef | grep kube-apiserver | grep -v grep
Expected result:
'Node,RBAC' has 'Node'
Ensure that the --profiling argument is set to false
Remediation: Edit the API server pod specification file /etc/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 argument is set to false.
Remediation: If using a Kubelet config file, edit the file to set authentication: anonymous: 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 variable.
--anonymous-auth=false

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 argument is set to 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 - Buildtime
Kubernetes
apiVersion: 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:

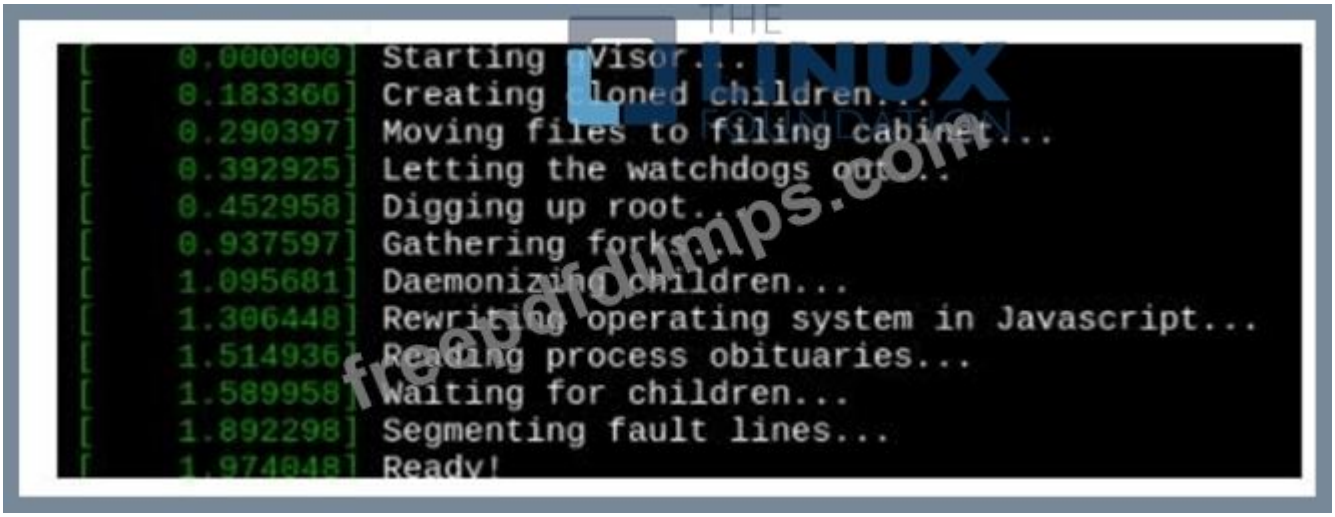
- /bin/sh
- -ec
- ETCDCTL_API=3 etcdctl --endpoints=https://[192.168.22.9]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert=/etc/kubernetes/pki/etcd/healthcheck-client.crt --key=/etc/kubernetes/pki/etcd/healthcheck-client.key get foo failureThreshold: 8 initialDelaySeconds: 15 timeoutSeconds: 15 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: 6
SIMULATION
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. Send us the Feedback on it.
Answer: (SHOW ANSWER)

NEW QUESTION: 7
SIMULATION
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.
Finally, test the configuration by deploying the pod having the image tag as latest.
A. Send us the Feedback on it.
Answer: A (LEAVE A REPLY)

NEW QUESTION: 8
SIMULATION
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:-



A. Send us your feedback on it.
Answer: A (LEAVE A REPLY)

NEW QUESTION: 9
SIMULATION
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. Send us the Feedback on it.
Answer: A (LEAVE A REPLY)

NEW QUESTION: 10
SIMULATION
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. Send us your Feedback on this.
Answer: A (LEAVE A REPLY)