Free Questions for CKS by go4braindumps

Shared by Miranda on 24-05-2024

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

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]

Options:

A- Send us your feedback on it.

B- Send us your

Answer:

A

Question 2

Question Type: MultipleChoice

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

Options:

A- Explanation:

kubesec scan k8s-deployment.yaml

cat <<EOF > 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

EOF

kubesec scan kubesec-test.yaml

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

kubesec http 8080 &

[1] 12345

{'severity':'info','timestamp':'2019-05-12T11:58:34.662+0100','caller':'server/server.go:69','message':'Starting HTTP server on port 8080'}

curl -sSX POST --data-binary @test/asset/score-0-cap-sys-admin.yml http://localhost:8080/scan

[

{

'object': 'Pod/security-context-demo.default',

'valid': true,

```
'message': 'Failed with a score of -30 points',
'score': -30,
'scoring': {
'critical': [
{
'selector': 'containers[] .securityContext .capabilities .add == SYS_ADMIN',
'reason': 'CAP_SYS_ADMIN is the most privileged capability and should always be avoided'
},
{
'selector': 'containers[] .securityContext .runAsNonRoot == true',
'reason': 'Force the running image to run as a non-root user to ensure least privilege'
},
// ...
```

Answer:

A

Question 3

Question Type: **MultipleChoice**

Service is running on port 389 inside the system, find the process-id of the process, and stores the names of all the open-files inside the /candidate/KH77539/files.txt, and also delete the binary.

Options:

A- Explanation:

root# netstat -ltnup

Active Internet connections (only servers)

Proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name

tcp 0 0 127.0.0.1:17600 0.0.0.0:* LISTEN 1293/dropbox

tcp 0 0 127.0.0.1:17603 0.0.0.0:* LISTEN 1293/dropbox

tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN 575/sshd

tcp 0 0 127.0.0.1:9393 0.0.0.0:* LISTEN 900/perl

tcp 0 0 :::80 :::* LISTEN 9583/docker-proxy

tcp 0 0 :::443 :::* LISTEN 9571/docker-proxy

udp 0 0 0.0.0.0:68 0.0.0.0:* 8822/dhcpcd

...

root# netstat -ltnup | grep ':22'

tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN 575/sshd

Thesscommand is the replacement of thenetstatcommand.

Now let's see how to use thesscommand to see which process is listening on port 22:

root# ss -ltnup 'sport = :22'

Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port

tcp LISTEN 0 128 0.0.0.0:22 0.0.0.0:* users:('sshd',pid=575,fd=3))

Answer:

A

Question 4

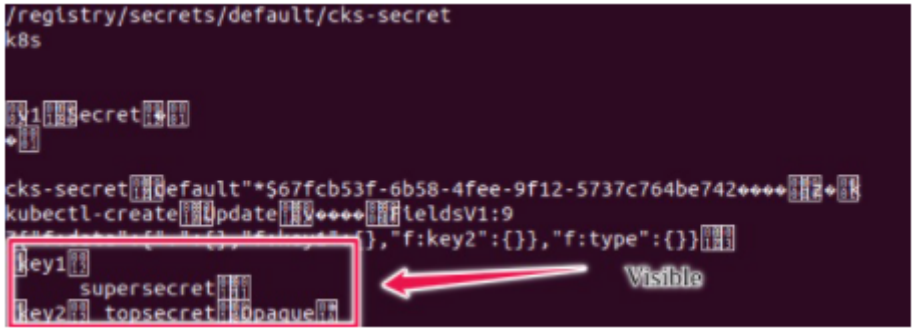
Question Type: MultipleChoice

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.

Options:

A- Explanation:

ETCD secret encryption can be verified with the help ofetcdctlcommand line utility.

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

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

Answer:

A

Question 5

Question Type: MultipleChoice

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.

Options:

A- Explanation:

$ kubectl get ing -n <namespace-of-ingress-resource>

NAME HOSTS ADDRESS PORTS AGE

cafe-ingress cafe.com 10.0.2.15 80 25s

$ kubectl describe ing <ingress-resource-name> -n <namespace-of-ingress-resource>

Name: cafe-ingress

Namespace: default

Address: 10.0.2.15

Default backend: default-http-backend:80 (172.17.0.5:8080)

Rules:

Host Path Backends

---- ---- --------

cafe.com

/tea tea-svc:80 (<none>)

/coffee coffee-svc:80 (<none>)

Annotations:

kubectl.kubernetes.io/last-applied-configuration:

{'apiVersion':'networking.k8s.io/v1','kind':'Ingress','metadata':{'annotations':{},'name':'cafe-ingress','namespace':'default','selfLink':'/apis/networking/v1/namespaces/default/ingresses/cafe-ingress'},'spec':{'rules':[{'host':'cafe.com','http':{'paths':[{'backend':{'serviceName':'tea-svc','servicePort':80},'path':'/tea'},{'backend':{'serviceName':'coffee-svc','servicePort':80},'path':'/coffee'}]}}]},'status':{'loadBalancer':{'ingress':[{'ip':'169.48.142.110'}]}}}

Events:

Type Reason Age From Message

---- ------ ---- ---- -------

Normal CREATE 1m ingress-nginx-controller Ingress default/cafe-ingress

Normal UPDATE 58s ingress-nginx-controller Ingress default/cafe-ingress

$ kubectl get pods -n <namespace-of-ingress-controller>

NAME READY STATUS RESTARTS AGE

ingress-nginx-controller-67956bf89d-fv58j 1/1 Running 0 1m

$ kubectl logs -n <namespace> ingress-nginx-controller-67956bf89d-fv58j

---

NGINX Ingress controller

Release: 0.14.0

Build: git-734361d

Repository: https://github.com/kubernetes/ingress-nginx

....

Answer:

A

Question 6

Question Type: **MultipleChoice**

Create a network policy named allow-np, that allows pod in the namespace staging to connect to port 80 of other pods in the same namespace.

Ensure that Network Policy:-

1. Does not allow access to pod not listening on port 80.

2. Does not allow access from Pods, not in namespace staging.

Options:

A- Explanation:
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
name: network-policy
spec:
podSelector: {} #selects all the pods in the namespace deployed
policyTypes:
- Ingress
ingress:
- ports: #in input traffic allowed only through 80 port only
- protocol: TCP
port: 80

Answer:

A

Question 7

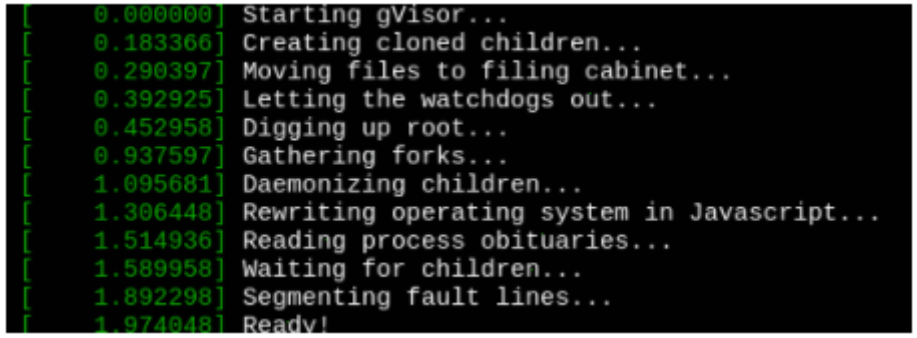
Question Type: MultipleChoice

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.

Options:

A- Explanation:



Answer:

A

Question 8

Question Type: MultipleChoice

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

Options:

A- Explanation:

```
candidate@cli:~$ kubectl config use-context KSRS00602
Switched to context "KSRS00602".
candidate@cli:~$ ssh ksrs00602-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@ksrs00602-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
  - 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:~$
```

Answer:

A