## Article Directory



kubernetes exam in action exam information
2 hours
15-20 topics
The appointment time is the same as CKA, and the results will be available within 32 hours
Full score of less than 100, 87 or 93 but a passing score of 67
simulated environment
4 environments and 1 console
NAT network segment 192.168.26.0
Mock exam questions

# 1. Mirror scan ImagePolicyWebhook

Switch cluster kubectl config use-context k8s
context
A container image scanner is set up on the cluster, but It's not yet fully integrated into the cluster's configuration When complete, the container image scanner shall scall scan for and reject the use of vulnerable images.
task:
You have to complete the entire task on the cluster's master node, where all services and files have been prepared and placed
Glven an incomplete configuration in directory /etc/kubernetes/aa and a functional container image scanner with HTTPS sendpitont http://192.168.26.60:1323/image_policy

1.enable the necessary plugins to create an image policy
2. validate the control configuration and chage it to an implicit deny
3. Edit the configuration to point the provided HTTPS endpoint correct
Finally, test if the configureion is working by trying to deploy the valnerable resource /csk/1/web1.yaml
problem solving ideas
ImagePolicyWebhook

```
Keywords: image_policy, deny
1. Switch clusters, view master, sshmaster
2. ls /etc/kubernetes/xxx
3. Change true to false in vi /etc/kubernetes/xxx/xxx.yaml
The address of https in vi /etc/kubernetes/xxx/xxx.yaml
The volume needs to be mounted in
4. Enable ImagePolicyWebhook and --admission-control-config-file=
5. systemctl restart kubelet
6. kubectl run pod1 --image=nginx
```

case:

Configure /etc/kubernetes/manifests/kube-apiserver.yaml
Add ImagePolicyWebhook related policies
Restart api-server, systemctl restart kubelet
Failed to create a pod after verifying the image
Modify /etc/kubernetes/admission/admission_config.yaml policy defaultAllow: true
Revalidate the image to create the pod

```
$ ls /etc/kubernetes/aa/
admission_config.yaml apiserver-client-cert.pem apiserver-client-key.pem external-cert.pem external-key.pem kubeconf
```

```
$ cd /etc/kubernetes/aa
$ cat kubeconf
apiVersion: v1
kind: Config

# clusters refer to the remote service.
clusters:
- cluster:
    certificate-authority: /etc/kubernetes/aa/external-cert.pem # CA for verifying the remote service.
    server: http://192.168.26.60:1323/image_policy # URL of remote service to query. Must use 'https'.
  name: image-checker

contexts:
- context:
    cluster: image-checker
    user: api-server
  name: image-checker
current-context: image-checker
preferences: {}
```

```
# users refer to the API server's webhook configuration.
users:
- name: api-server
  user:
    client-certificate: /etc/kubernetes/aa/apiserver-client-cert.pem # cert for the webhook admission controller to use
    client-key: /etc/kubernetes/aa/apiserver-client-key.pem # key matching the cert


$ cat admission_config.yaml
apiVersion: apiserver.config.k8s.io/v1
kind: AdmissionConfiguration
plugins:
  - name: ImagePolicyWebhook
    configuration:
      imagePolicy:
        kubeConfigFile: /etc/kubernetes/aa/kubeconf
        allowTTL: 50
        denyTTL: 50
        retryBackoff: 500
        defaultAllow: false

#Modify api-server configuration
$ cat /etc/kubernetes/manifests/kube-apiserver.yaml
..........
  - command:
    -kube-apiserver
    - --admission-control-config-file=/etc/kubernetes/aa/admission_config.yaml #Add this line
    - --advertise-address=192.168.211.40
    - --allow-privileged=true
    - --authorization-mode=Node,RBAC
    - --client-ca-file=/etc/kubernetes/pki/ca.crt
    - --enable-admission-plugins=NodeRestriction,ImagePolicyWebhook # #Modify this line
    - --enable-bootstrap-token-auth=true
    - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
  .........
    - mountPath: /etc/kubernetes/pki
      name: k8s-certs
      readOnly: true
    - mountPath: /etc/kubernetes/aa #Add this line
      name: k8s-admission #Add this line
      readOnly: true #Add this line
.........
  - hostPath: #Add this line
      path: /etc/kubernetes/aa #Add this line
      type: DirectoryOrCreate #add this line
    name: k8s-admission #Add this line
  - hostPath:
      path: /usr/local/share/ca-certificates
      type: DirectoryOrCreate
```

```
    name: usr-local-share-ca-certificates
  - hostPath:
      path: /usr/share/ca-certificates
      type: DirectoryOrCreate
    name: usr-share-ca-certificates
status: {}


$ k get nodes
NAME STATUS ROLES AGE VERSION
master Ready control-plane, master 9d v1.20.1
node1 Ready <none> 9d v1.20.1
node2 Ready <none> 9d v1.20.1

#Create pod failed
$ k run test --image=nginx
Error from server (Forbidden): pods "test" is forbidden: Post "https://external-service:1234/check-image?timeout=30s": dial tcp: lookup external-service on 8.8.8.8:53: no such host

#Modify admission_config.yaml configuration
$ vim /etc/kubernetes/aa/admission_config.yaml
apiVersion: apiserver.config.k8s.io/v1
kind: AdmissionConfiguration
plugins:
  - name: ImagePolicyWebhook
    configuration:
      imagePolicy:
        kubeConfigFile: /etc/kubernetes/aa/kubeconf
        allowTTL: 50
        denyTTL: 50
        retryBackoff: 500
        defaultAllow: true #Modify this behavior true


#restart api-server
$ ps -ef | grep api
root 78871 39023 0 20:17 pts/3 00:00:00 grep --color=auto api

$ mv ../kube-apiserver.yaml .

#Create pod successfully
$ k run test --image=nginx
pod/test created
```

## 2. sysdig detect pod

Switch cluster kubectl config use-context k8s

you may user you brower to open one additonal tab to access sysdig's documentation ro Falco's documentation

Task:
user runtime detection tools to detect anomalous processes spawning and executing frequently in the sigle container belorging to pod redis.
Tow tools are available to use:

sysdig
falico
The tools are pre-installed on the cluster's worker node only; they are not available on the base system or the master node.
using the tool of your choice (including any non pre-install tool) analyze the container's behavior for at least 30 seconds, using filers that detect newly spawing and executing processes store an incident file at /opt/2/report, containing the detected incidents one per line in the follwing format:

```
[timestamp],[uid],[processName]
```

problem solving ideas
Sysdig User Guide

```
Keyword: sysdig
0. Remember to use sysdig -l |grep to search for relevant fields
1. Switch the cluster, query the corresponding pod, and ssh to the node host corresponding to the pod
2. Use sysdig, pay attention to the required format and time, and redirect the result to the corresponding file
3. sysdig -M 30 -p "*%evt.time,%user.uid,%proc.name" container.id=container id >/opt/2/report
```

the case

# 3. cluster role

Switch cluster kubectl config use-context k8s
context
A Role bound to a pod's serviceAccount grants overly permissive permission
Complete the following tasks to reduce the set of permissions.
task
Glven an existing Pod name web-pod running in the namespace monitoring Edit the Rolebound to the Pod's serviceAccount sa-dev-1 to only allow performing list operations, only on resources of type Endpoints
create a new Role named role-2 in the namespaces monitoring which only allows performing update operations, only on resources of type persistent voume claims.
create a new Rolebind name role role-2-binding binding the newly created Roleto the Pod's serviceAccount

problem solving ideas
RBAC

```
Keywords: role, rolebinding
1. Find the roll corresponding to rollbind and modify the permissions to list and endpoints
$ kubectl edit role role-1 -n monitoring
2. Remember --verb is permission --resource is object
```

```
$ kubectl create role role-2 --verb=update --resource=persistentvolumeclaims -n
monitoring
3. Create a binding binding to the corresponding sa
$ kubectl create rolebinding role-2-binding --role=role-2 --
serviceaccount=monitoring:sa-dev-1 -n monitoring
```

# 4. AppArmor

Switch cluster kubectl config use-context k8s

Context

AppArmor is enabled on the cluster's worker node. An AppArmor profile is prepared, but not enforced yet. You may use your browser to open one additional tab to access  An AppArmor profile is prepared, but not enforced yet.You may use your browser to open one additional tab to access

theAppArmor documentation. Tasktask

On the cluster's worker node, enforce the prepared AppArmor profile located at /etc/apparmor.d/nginx_apparmor . Edit the prepared manifest file located at /cks/4/pod1.yaml to apply the AppArmor profile. Finally, apply the manifest file and create the pod specified in itEdit the prepared manifest file located at /cks/4/pod1.yaml to apply the AppArmor profile.Finally, apply the manifest file and create the pod specified in it

problem solving ideas

apparmor

```
Keyword: apparmor
1. Switch to the cluster, remember to check the nodes, ssh to the node node
2. View the corresponding configuration file and name
$ cd /etc/apparmor.d
$ vinginx_apparmor
$ apparmor_status |grep nginx-profile-3 # If there is no grep to indicate that it is not started
$ apparmor_parser -q nginx_apparmor # Load and enable this configuration file
3. Modify the corresponding yaml application rule, open the URL copy example of the official website, and modify the container name and local configuration name
$ vi /cks/4/pod1.yaml
apiVersion: v1
kind: Pod
metadata:
  name: hello-apparmor
  annotations:
    container.apparmor.security.beta.kubernetes.io/hello:localhost/nginx-profile-3
spec:
  containers:
  - name: hello
    image: busybox
    command: [ "sh", "-c", "echo 'Hello AppArmor!' && sleep 1h" ]

4. Created after modification
$ kubectl apply -f /cks/4/pod1.yaml
```

# 5. Pod Security Policy

Switch cluster kubectl config use-context k8s63
context
A PodsecurityPolicy shall prevent the create on of privileged Pods in a specific
namespace. Tasktask

Create a new PodSecurityPolicy named prevent-psp-policy , which prevents the creation of privileged Pods.
Create a new ClusterRole named restrict-access-role, which uses the newly created PodSecurityPolicy prevent-psp-policy.
Create a new serviceAccount named pspdenial-sa in the existing namespace development.
Finally, create a new clusterRoleBinding named dany-access-bind , which binds the newly created ClusterRole restrict-access-role to the newly created serviceAccount
problem solving ideas
PodSecurityPolicy

```
Keywords: psp policy privileged
0. Toggle grouping to see if it is enabled
$ vi /etc/kubernetes/manifests/kube-apiserver.yaml
- --enable-admission-plugins=NodeRestriction,PodSecurityPolicy

$ systemctl restart kubelet

1. Copy the psp from the official website, modify and deny privileges
$ cat psp.yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
name: prevent-psp-policy
spec:
privileged: false
seLinux:
rule: RunAsAny
supplementalGroups:
rule: RunAsAny
runAsUser:
rule: RunAsAny
fsGroup:
rule: RunAsAny
volumes:
- '*'

$ kubectl create -f psp.yaml

2. Create the corresponding clusterrole
$ kubectl create clusterrole restrict-access-role --verb=use --resource=podsecuritypolicy --resource-name=prevent-psp-policy

3. Create sa to see the corresponding ns
$ kubectl create sa psp-denial-sa -n development
```

```
4. Create a binding relationship
$ kubectl create clusterrolebinding dany-access-bind --clusterrole=restrict-access-role --serviceaccount=development:psp-denial-sa
```

# 6. Network Policy

Switch cluster kubectl config use-context k8s
create a NetworkPolicy named pod-access to restrict access to Pod products-service running in namespace development . only allow the following Pods to connect to Pod productsservice :only allow the following Pods to connect to Pod productsservice:

Pods in the namespace testing
Pods with label environment: staging , in any namespace Make sure to apply the NetworkPolicy. You can find a skeleton on manifest file at /cks/6/p1.yamlYou can find a skeleton on manifest file at /cks/6/p1.yaml
problem solving ideas
NetworkPolicy

```
Keywords: NetworkPolicy
1. The host checks the label of the pod
$ kubectl get pod -n development --show-labels

2. Check the label corresponding to ns, there is no need to set it
$ kubectl label ns testing name=testing

3. Orchestrate networkpolicy strategy
$ cat /cks/6/p1.yaml
kind: NetworkPolicy
metadata:
name: "pod-access"
namespace: "development"
spec:
  podSelector:
    matchLabels:
      environment: staging
  policyTypes:
  -Ingress
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          name: testing
  - from:
    - namespaceSelector:
        matchLabels:
      podSelector:
        matchLabels:
          environment: staging
```

```
$ kubectl create -f /cks/6/p1.yaml
```

# 7. Dockerfile detection and yaml file problems

Switch cluster kubectl config use-context k8s

task

Analyze and edit the given Dockerfile (based on the ubuntu:16.04 image) /cks/7/Dockerfile fixing two instructions present in the file being prominent security/best-practice issues.

Analyze and edit the given manifest file /cks/7/deployment.yaml

fixing two fields present in the file being prominent security/best-practice issues.

problem solving ideas

```
Keywords: Dockerfile issues
1. Pay attention to the number of errors prompted by dockerfile
Note: USER root
2. Yaml problem: pay attention to the api version problem, and privileged network, mirror version, it also depends on how many mistakes are mentioned in the title
```

case:

Dockerfile

```
# build container stage 1
FROM ubuntu:20.04
ARG DEBIAN_FRONTEND=noninteractive
RUN apt-get update && apt-get install -y golang-go=2:1.13~1ubuntu2
COPY app.go.
RUN pwd
RUN CGO_ENABLED=0 go build app.go

# app container stage 2
FROM alpine:3.12.0
RUN addgroup -S appgroup && adduser -S appuser -G appgroup -h /home/appuser
RUN rm -rf /bin/*
COPY --from=0 /app /home/appuser/
USER appuser
CMD ["/home/appuser/app"]
```

# 8. pod security

Switch cluster kubectl config use-context k8s

context

It is best-practice to design containers to best teless and immutable. Tasktask

Inspect Pods running in namespace testing and delete any Pod that is either not stateless or not immutable. use the following strict interpretation of stateless and immutable:Use the following strict interpretation of stateless and immutable:

Pods being able to store data inside containers must be treated as not stateless.

You don't have to worry whether data is actually stored inside containers or not already. Pods being configured to be privileged in any way must be treated as potentially not stateless and not immutable.

problem solving ideas

```
Keyword: stateless immutable
1. get all pods
2. Check if you have the privilege privi*
3. Check if there is a volume
4. Delete the privileged network and volume
$ kubectl get pod pod1 -n testing -o jsonpath={.spec.volumes} | jq
$ kubectl get pod sso -n testing -o yaml |grep "privi.*: true"
$ kubectl delete pod xxxxx -n testing
```