DUMPSARENA

Certified Kubernetes Security Specialist (CKS)

Linux Foundation CKS

Version Demo

Total Demo Questions: 10

Total Premium Questions: 48

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

QUESTION NO: 1 - (SIMULATION)

Cluster: dev

Master node: master1Worker node: worker1

You can switch the cluster/configuration context using the following command:[desk@cli] $ kubectl config use-context dev

Task:Retrieve the content of the existing secret named adam in the safe namespace.

Store the username field in a file names /home/cert-masters/username.txt, and the password field in a file named /home/cert-masters/password.txt.

1. You must create both files; they don't exist yet.2. Do not use/modify the created files in the following steps, create new temporary files if needed.

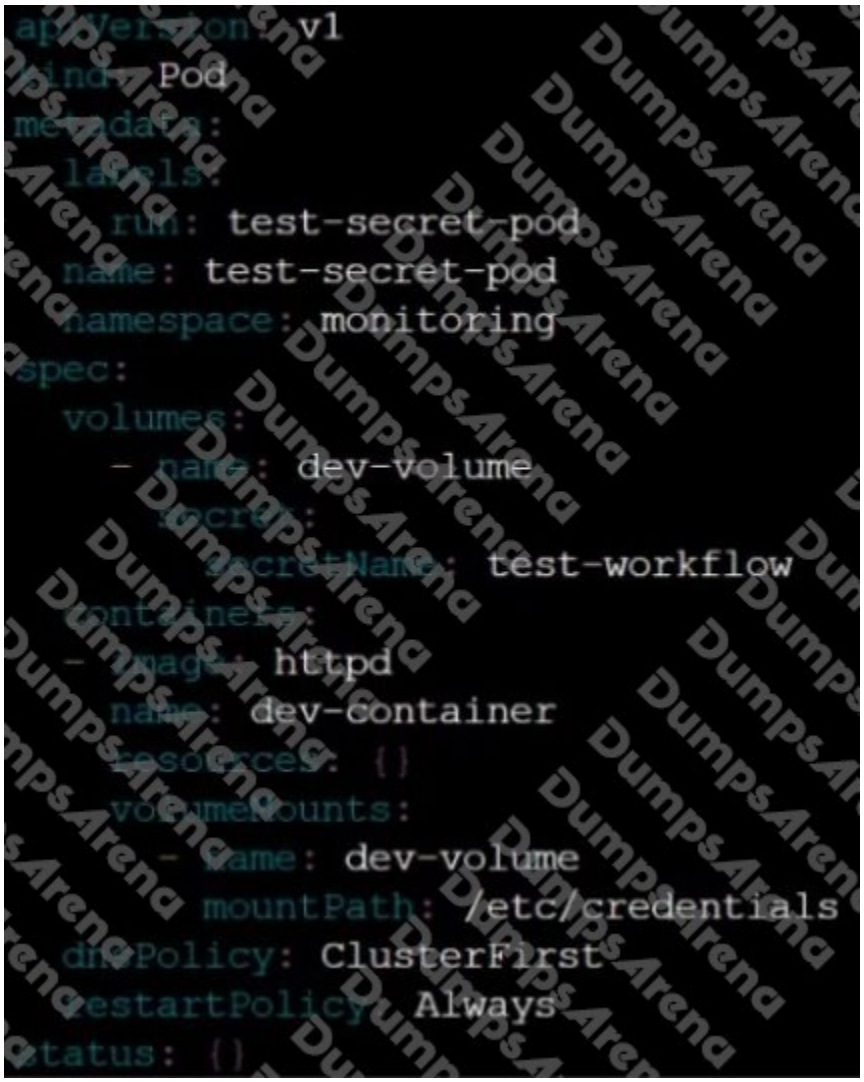
Create a new secret names newsecret in the safe namespace, with the following content:Username: dbadminPassword: moresecurepas

Finally, create a new Pod that has access to the secret newsecret via a volume:

ANSWER: Seetheexplanationbelow

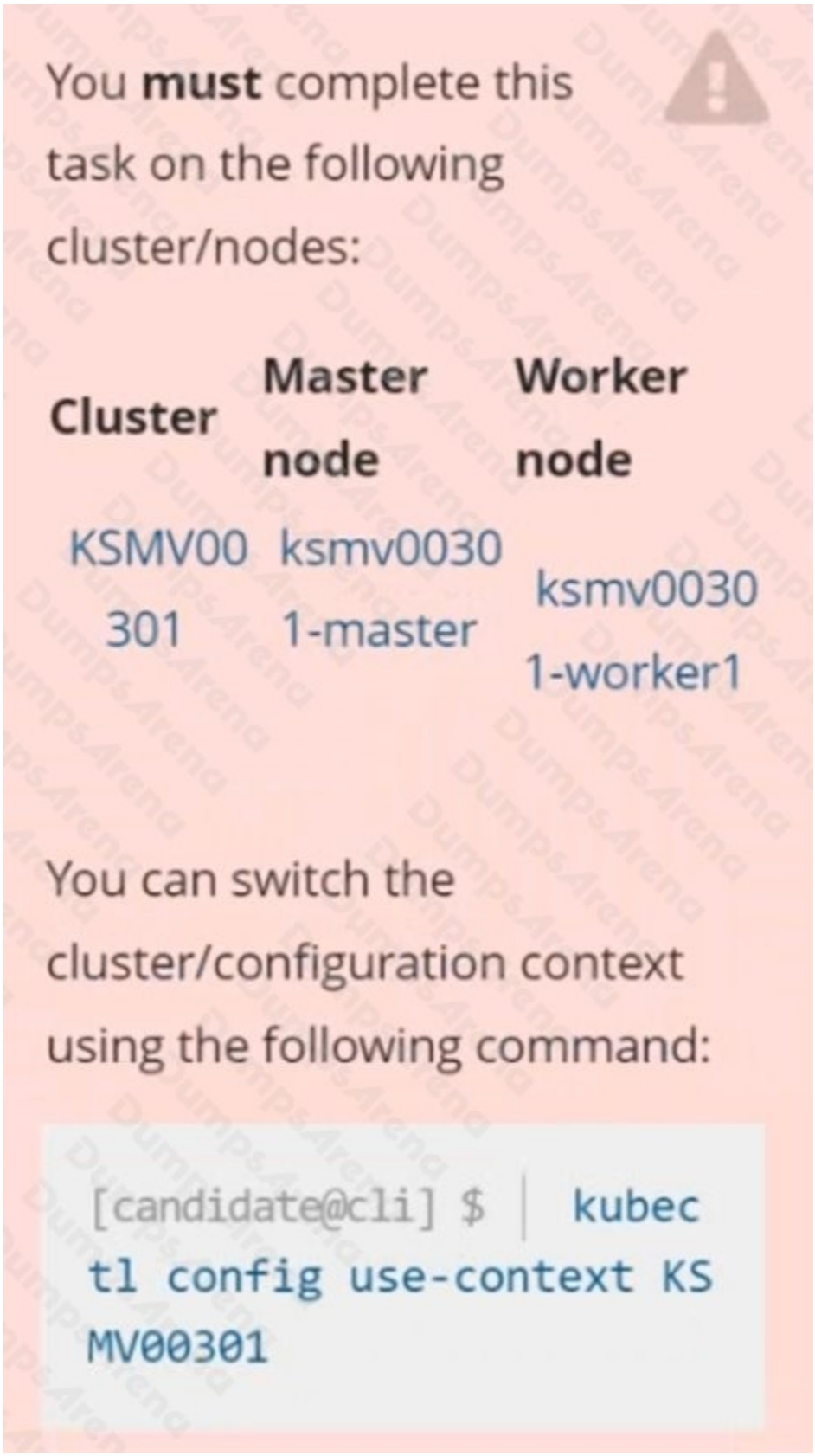
Explanation:


candidate@cli:~$ kubectl config use-context KSMV00201
Switched to context "KSMV00201".
candidate@cli:~$ kubectl get secret -n monitoring
NAME TYPE DATA AGE
dbl-test Opaque 2 6h23m
default-token-cqqf6 kubernetes.io/service-account-token 3 6h23m
candidate@cli:~$ kubectl get secret/dbl-test -n monitoring
NAME TYPE DATA AGE
dbl-test Opaque 2 6h23m
candidate@cli:~$ kubectl get secret/dbl-test -n monitoring -o yaml
apiVersion: v1
data:
password: QVU3dHhlbXFOTHZt
username: cHJvZHVjdGlvbi0x
kind: Secret
metadata:
creationTimestamp: "2022-05-20T08:37:33Z"
name: dbl-test
namespace: monitoring
resourceVersion: "2588"
uid: 659bd4ac-e0ba-4d9f-b411-816f2aedf7e6
type: Opaque
candidate@cli:~$ echo "cHJvZHVjdGlvbi0x" | base64 -d
production-1candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ echo "cHJvZHVjdGlvbi0x" | base64 -d > /home/candidate/username.txt
candidate@cli:~$ cat /home/candidate/username.txt
production-1candidate@cli:~$
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ echo "QVU3dHhlbXFOTHZt" | base64 -d
AU7txumqNLvmcandidate@cli:~$ echo "QVU3dHhlbXFOTHZt" | base64 -d > /home/candidate/password.
txt
candidate@cli:~$ cat /home/candidate/password.txt
AU7txumqNLvmcandidate@cli:~$
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ kubectl create secret generic test-workflow --from-literal=username=dev-dat
abase --from-literal=password=aV7HR7nU3JLx -n monitoring
secret/test-workflow created
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ kubectl -n monitoring run test-secret-pod --image=httpd --dry-run=client -
o yaml > test-secret-pod.yaml
candidate@cli:~$ vim test-secret-pod.yaml




```
candidate@cli:~$ kubectl  -n monitoring run test-secret-pod --image=httpd --dry-run=client
o yaml > test-secret-pod.yaml
candidate@cli:~$ vim test-secret-pod.yaml
candidate@cli:~$ cat test-secret-pod.yaml
```

QUESTION NO: 2 - (SIMULATION)


You must complete this task on the following cluster/nodes:
Cluster
Master node
Worker node
KSMV00301
ksmv00301-master
ksmv00301-worker1
You can switch the cluster/configuration context using the following command:
[candidate@cli] $ kubectl config use-context KSMV00301

Context

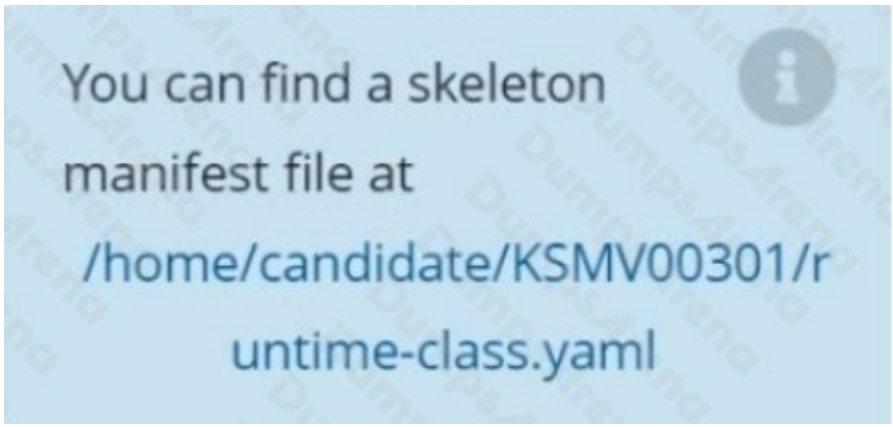
This cluster uses containerd as CRI runtime.

Containerd's default runtime handler is runc. Containerd has been prepared to support an additional runtime handler, runsc (gVisor).

Task

Create a RuntimeClass named sandboxed using the prepared runtime handler named runsc.

Update all Pods in the namespace server to run on gVisor.



ANSWER: Seetheexplanationbelow

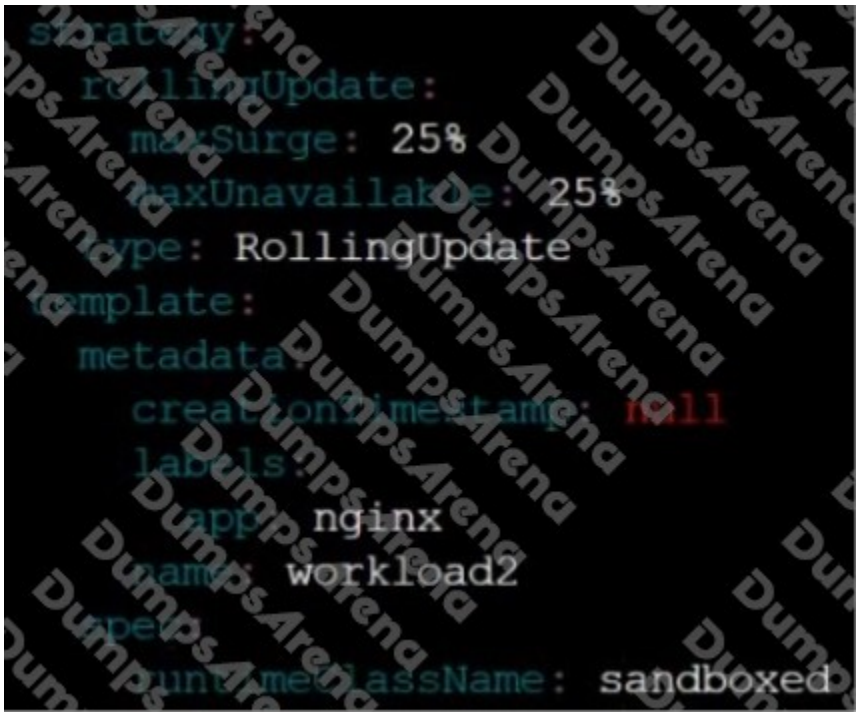
Explanation:


node.k8s.io/v1
RuntimeClass
metadata:
name:
handler
"runsc"

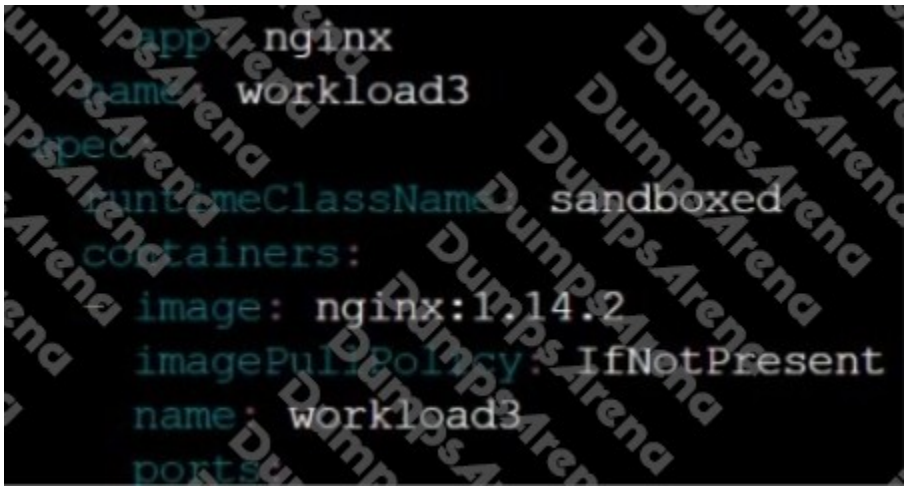

candidate@cli:~$ kubectl config use-context KSMV00301
Switched to context "KSMV00301".
candidate@cli:~$ cat /home/candidate/KSMV00301/runtime-class.yaml
---
apiVersion: node.k8s.io/v1
kind: RuntimeClass
metadata:
name: ""
handler: ""
candidate@cli:~$ vim /home/candidate/KSMV00301/runtime-class.yaml
candidate@cli:~$ cat /home/candidate/KSMV00301/runtime-class.yaml
---
apiVersion: node.k8s.io/v1
kind: RuntimeClass
metadata:
name: "sandboxed"
handler: "runsc"
candidate@cli:~$ kubectl get deployments.apps -n server
NAME READY UP-TO-DATE AVAILABLE AGE
workload1 1/1 1 1 5h43m
workload2 1/1 1 1 5h43m
workload3 1/1 1 1 5h43m
candidate@cli:~$ kubectl get pods -n server
NAME READY STATUS RESTARTS AGE
workload1-6869857dd7-s45rc 1/1 Running 0 5h43m
workload2-d4bd497d5-h44df 1/1 Running 0 5h43m
workload3-8587774495-chm56 1/1 Running 0 5h43m
candidate@cli:~$ kubectl -n server edit deployments.apps workload1


app: nginx
name: workload1
runtimeClassName: sandboxed
- image: nginx:1.14.2
imagePullPolicy: IfNotPresent
name: workload1
ports:
- containerPort: 87
protocol: TCP
resources: {}
terminationMessagePath: /dev/termination-log
terminationMessagePolicy: File
dnsPolicy: ClusterFirst
restartPolicy: Always
schedulerName: default-scheduler
securityContext: {}
terminationGracePeriodSeconds: 30
"/tmp/kubectl-edit-3385772700.yaml"

```
NAME                         READY   STATUS    RESTARTS   AGE
workload1-6869857dd7-s45rc   1/1     Running   0          5h44m
workload2-d4bd497d5-h44df    1/1     Running   0          5h44m
workload3-8587774495-chm56   1/1     Running   0          5h44m
candidate@cli:~$ kubectl -n server edit deployments.apps workload1
Edit cancelled, no changes made.
candidate@cli:~$ kubectl get pods -n server
NAME                         READY   STATUS    RESTARTS   AGE
workload1-6869857dd7-s45rc   1/1     Running   0          5h45m
workload2-d4bd497d5-h44df    1/1     Running   0          5h44m
workload3-8587774495-chm56   1/1     Running   0          5h44m
candidate@cli:~$ kubectl -n server edit deployments.apps workload2
Edit cancelled, no changes made.
candidate@cli:~$ kubectl create -f /home/candidate/KSMV00301/runtime-class.yaml
runtimeclass.node.k8s.io/sandboxed created
candidate@cli:~$ kubectl get pods -n server
NAME                         READY   STATUS    RESTARTS   AGE
workload1-6869857dd7-s45rc   1/1     Running   0          5h45m
workload2-d4bd497d5-h44df    1/1     Running   0          5h45m
workload3-8587774495-chm56   1/1     Running   0          5h45m
candidate@cli:~$ kubectl -n server edit deployments.apps workload2
```



```
NAME                         READY   STATUS    RESTARTS   AGE
workload1-6869857dd7-s45rc   1/1     Running   0          5h45m
workload2-d4bd497d5-h44df    1/1     Running   0          5h45m
workload3-8587774495-chm56   1/1     Running   0          5h45m
candidate@cli:~$ kubectl -n server edit deployments.apps workload2
deployment.apps/workload2 edited
candidate@cli:~$ kubectl get pods -n server
NAME                         READY   STATUS    RESTARTS   AGE
workload1-8d8649ff6-wvjtg    1/1     Running   0          15s
workload2-765bdb98c8-wd8cm   1/1     Running   0          4s
workload3-8587774495-chm56   1/1     Running   0          5h45m
candidate@cli:~$ kubectl -n server edit deployments.apps workload3
```




```
candidate@cli:~$ kubectl -n server edit deployments.apps workload3
deployment.apps/workload3 edited
candidate@cli:~$ kubectl get pods -n server
NAME                         READY   STATUS    RESTARTS   AGE
workload1-8d8649ff6-wvjtg    1/1     Running   0          58s
workload2-765bdb98c8-wd8cm   1/1     Running   0          47s
workload3-76c994bb4d-s6k85   1/1     Running   0          4s
candidate@cli:~$
```

QUESTION NO: 3 - (SIMULATION)

On the Cluster worker node, enforce the prepared AppArmor profile

Edit the prepared manifest file to include the AppArmor profile.

Finally, apply the manifests files and create the Pod specified on it.

Verify: Try to make a file inside the directory which is restricted.

ANSWER: Seeexplanationbelow.

Explanation:

QUESTION NO: 4 - (SIMULATION)

Create a PSP that will prevent the creation of privileged pods in the namespace.

Create a new PodSecurityPolicy named prevent-privileged-policy which prevents the creation of privileged pods.

Create a new ServiceAccount named psp-sa in the namespace default.

Create a new ClusterRole named prevent-role, which uses the newly created Pod Security Policy prevent-privileged-policy.

Create a new ClusterRoleBinding named prevent-role-binding, which binds the created ClusterRole prevent-role to the created SA psp-sa.

Also, Check the Configuration is working or not by trying to Create a Privileged pod, it should get failed.

ANSWER: Seethebelow.

Explanation:

a) Create a PSP that will prevent the creation of privileged pods in the namespace.

$ cat clusterrole-use-privileged.yaml

---

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRole

metadata:

name: use-privileged-psp

rules:

- apiGroups: ['policy']

resources: ['podsecuritypolicies']

verbs: ['use']

resourceNames:

- default-psp

---

apiVersion: rbac.authorization.k8s.io/v1

kind: RoleBinding

metadata:

name: privileged-role-bind

namespace: psp-test

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: use-privileged-psp

subjects:

- kind: ServiceAccount

name: privileged-sa

$ kubectl -n psp-test apply -f clusterrole-use-privileged.yaml

After a few moments, the privileged Pod should be created.

b) Create a new PodSecurityPolicy named prevent-privileged-policy which prevents the creation of privileged pods.

apiVersion: policy/v1beta1

kind: PodSecurityPolicy

metadata:

name: example

spec:

privileged: false # Don't allow privileged pods!

The rest fills in some required fields.

seLinux:

rule: RunAsAny

supplementalGroups:

rule: RunAsAny

runAsUser:

rule: RunAsAny

fsGroup:

rule: RunAsAny

volumes:

- '*'

And create it with kubectl:

kubectl-admin create -f example-psp.yaml

Now, as the unprivileged user, try to create a simple pod:

kubectl-user create -f- <<>

apiVersion: v1

kind: Pod

metadata:

name: pause

spec:

containers:

- name: pause

image: k8s.gcr.io/pause

EOF

The output is similar to this:

Error from server (Forbidden): error when creating "STDIN": pods "pause" is forbidden: unable to validate against any pod security policy: []

c) Create a new ServiceAccount named psp-sa in the namespace default.

$ cat clusterrole-use-privileged.yaml

---

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRole

metadata:

name: use-privileged-psp

rules:

- apiGroups: ['policy']

resources: ['podsecuritypolicies']

verbs: ['use']

resourceNames:

- default-psp

---

apiVersion: rbac.authorization.k8s.io/v1

kind: RoleBinding

metadata:

name: privileged-role-bind

namespace: psp-test

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: use-privileged-psp

subjects:

- kind: ServiceAccount

name: privileged-sa

$ kubectl -n psp-test apply -f clusterrole-use-privileged.yaml

After a few moments, the privileged Pod should be created.

d) Create a new ClusterRole named prevent-role, which uses the newly created Pod Security Policy prevent-privileged-policy.

apiVersion: policy/v1beta1

kind: PodSecurityPolicy

metadata:

name: example

spec:

privileged: false # Don't allow privileged pods!

The rest fills in some required fields.

seLinux:

rule: RunAsAny

supplementalGroups:

rule: RunAsAny

runAsUser:

rule: RunAsAny

fsGroup:

rule: RunAsAny

volumes:

- '*'

And create it with kubectl:

kubectl-admin create -f example-psp.yaml

Now, as the unprivileged user, try to create a simple pod:

kubectl-user create -f- <<>

apiVersion: v1

kind: Pod

metadata:

name: pause

spec:

containers:

- name: pause

image: k8s.gcr.io/pause

EOF

The output is similar to this:

Error from server (Forbidden): error when creating "STDIN": pods "pause" is forbidden: unable to validate against any pod security policy: []

e) Create a new ClusterRoleBinding named prevent-role-binding, which binds the created ClusterRole prevent-role to the created SA psp-sa.

apiVersion: rbac.authorization.k8s.io/v1

This role binding allows "jane" to read pods in the "default" namespace.

You need to already have a Role named "pod-reader" in that namespace.

kind: RoleBinding

metadata:

name: read-pods

namespace: default

subjects:

You can specify more than one "subject"

- kind: User

name: jane # "name" is case sensitive

apiGroup: rbac.authorization.k8s.io

roleRef:

"roleRef" specifies the binding to a Role / ClusterRole

kind: Role #this must be Role or ClusterRole

name: pod-reader # this must match the name of the Role or ClusterRole you wish to bind to

apiGroup: rbac.authorization.k8s.io

apiVersion: rbac.authorization.k8s.io/v1

kind: Role

metadata:

namespace: default

name: pod-reader

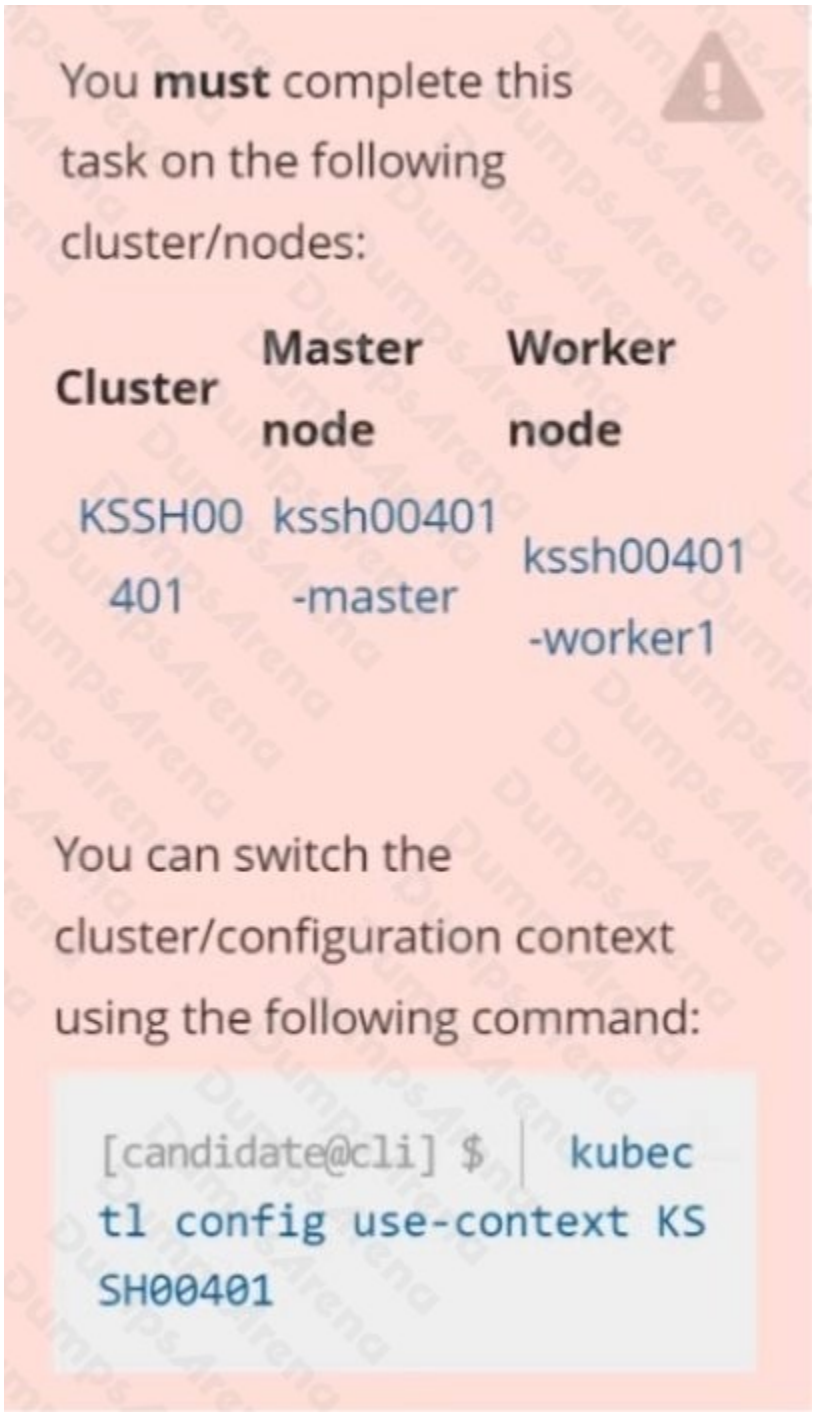
rules:

- apiGroups: [""] # "" indicates the core API group

resources: ["pods"]

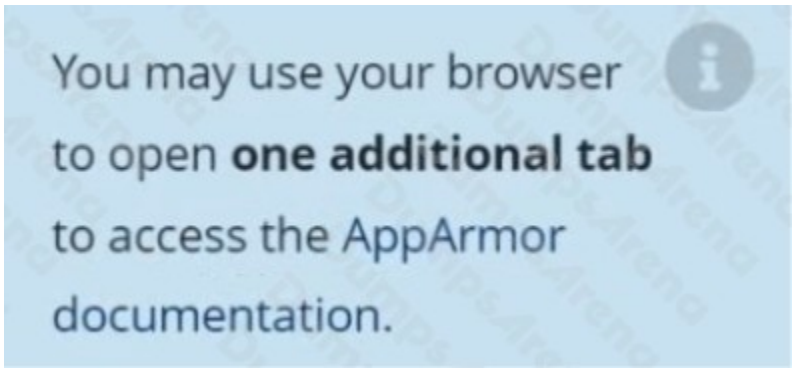
verbs: ["get", "watch", "list"]

QUESTION NO: 5 - (SIMULATION)



Context

AppArmor is enabled on the cluster's worker node. An AppArmor profile is prepared, but not enforced yet.



Task

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 /home/candidate/KSSH00401/nginx-pod.yaml to apply the AppArmor profile.

Finally, apply the manifest file and create the Pod specified in it.

ANSWER: Seetheexplanationbelow

Explanation:

```
candidate@cli:~$ kubectl config use-context KSSH00401
Switched to context "KSSH00401".
candidate@cli:~$ ssh kssh00401-worker1
Warning: Permanently added '10.240.86.172' (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@kssh00401-worker1:~# head /etc/apparmor.d/nginx_apparmor
#include <tunables/global>

profile nginx-profile-2 flags=(attach_disconnected,mediate_deleted) {
  #include <abstractions/base>
  network inet tcp,
  network inet udp,
  network inet icmp,

  deny network raw,

root@kssh00401-worker1:~# apparmor_parser -q /etc/apparmor.d/nginx_apparmor
root@kssh00401-worker1:~# exit
logout
Connection to 10.240.86.172 closed.
candidate@cli:~$ cat KSSH00401/nginx-pod.yaml
---
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
  - name: nginx-pod
    image: nginx:1.19.0
    ports:
    - containerPort: 80
candidate@cli:~$ vim KSSH00401/nginx-pod.yaml
```

```
---
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  annotations:
    container.apparmor.security.beta.kubernetes.io/nginx-pod: localhost/nginx-pr
spec:
  containers:
  - name: nginx-pod
    image: nginx:1.19.0
    ports:
    - containerPort: 80
~
```

Reference: https://kubernetes.io/docs/tutorials/clusters/apparmor/

QUESTION NO: 6 - (SIMULATION)

You can switch the cluster/configuration context using the following command:[desk@cli] $ kubectl config use-context stage Context:A PodSecurityPolicy shall prevent the creation of privileged Pods in a specific namespace.Task:1. Create a new PodSecurityPolcy named deny-policy, which prevents the creation of privileged Pods.2. Create a new ClusterRole name deny-access-role, which uses the newly created PodSecurityPolicy deny-policy.3. Create a new ServiceAccount named psd-denial-sa in the existing namespace development.Finally, create a new ClusterRoleBindind named restrict-access-bind, which binds the newly created ClusterRole deny-access-role to the newly created ServiceAccount psp-denial-sa

ANSWER: Seetheexplanationbelow

Explanation:

Create psp to disallow privileged container

k create sa psp-denial-sa -n development

namespace: development

Explanation

master1 $ vim psp.yaml

apiVersion: policy/v1beta1

kind: PodSecurityPolicy

metadata:

name: deny-policy

spec:

privileged: false # Don't allow privileged pods!

seLinux:

rule: RunAsAny

supplementalGroups:

rule: RunAsAny

runAsUser:

rule: RunAsAny

fsGroup:

rule: RunAsAny

volumes:

- '*'

master1 $ vim cr1.yaml

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRole

metadata:

name: deny-access-role

rules:

- apiGroups: ['policy']

resources: ['podsecuritypolicies']

verbs: ['use']

resourceNames:

- “deny-policy”

master1 $ k create sa psp-denial-sa -n developmentmaster1 $ vim cb1.yaml

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRoleBinding

metadata:

name: restrict-access-bing

roleRef:

kind: ClusterRole

name: deny-access-role

apiGroup: rbac.authorization.k8s.io

subjects:

Authorize specific service accounts:

- kind: ServiceAccount

name: psp-denial-sa

namespace: development

master1 $ k apply -f psp.yamlmaster1 $ k apply -f cr1.yamlmaster1 $ k apply -f cb1.yamlReference: https://kubernetes.io/docs/concepts/policy/pod-security-policy/

QUESTION NO: 7 - (SIMULATION)

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.

ANSWER: Seetheexplanationbelow:

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

QUESTION NO: 8 - (SIMULATION)

You **must** complete this task on the following cluster/nodes:

Cluster	Master node	Worker node
KSRS00101	ksrs00101-master	ksrs00101-worker1

You can switch the cluster/configuration context using the following command:

```
[candidate@cli] $ kubectl config use-context KSRS00101
```

You may use your browser to open **one additional tab** to access Falco's documentation.

Two tools are pre-installed on the cluster's worker node:

- sysdig
- falco

Using the tool of your choice (including any non pre-installed tool), analyze the container's behavior for at least 30 seconds, using filters that detect newly spawning and executing processes.

Store an incident file at /opt/KSRS00101/alerts/details, containing the detected incidents, one per line, in the following format:

```
timestamp,uid/username,proce
ssName
```

The following example shows a properly formatted incident file:

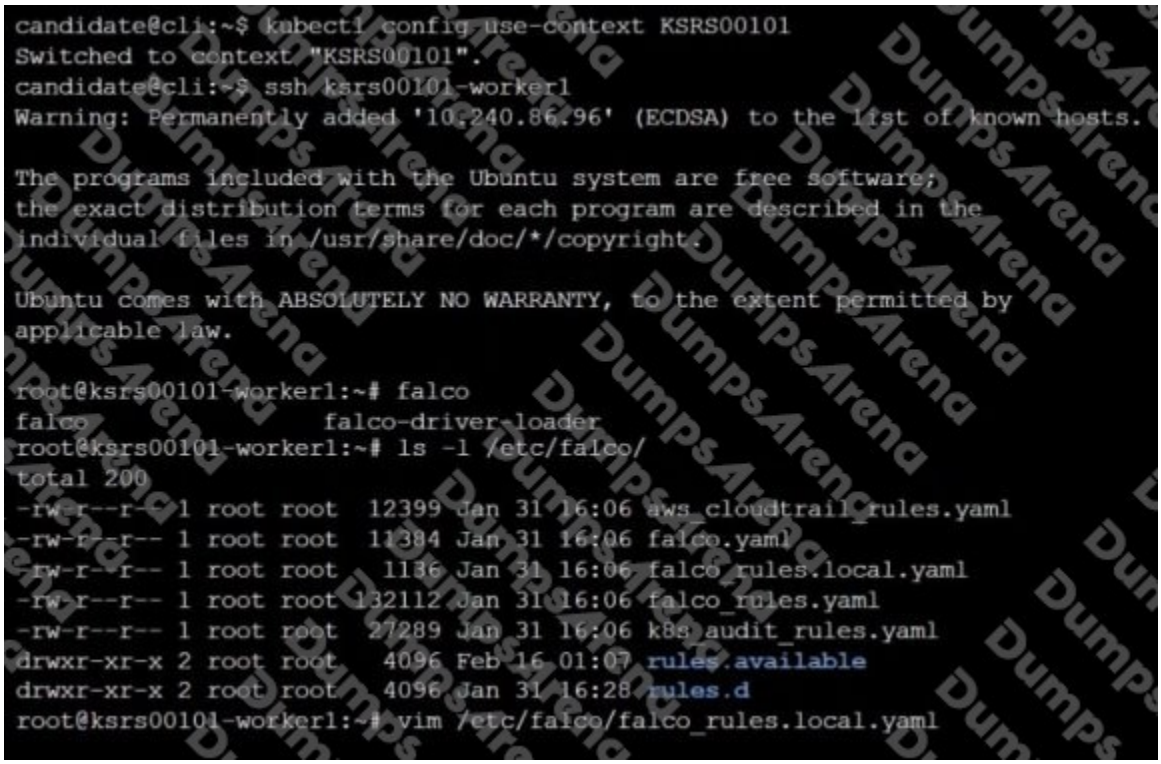
```
01:40:19.601363716,root,init
01:40:20.606013716,nobody,ba
sh
01:40:21.137163716,1000,tar
```

Keep the tool's original timestamp-format as-is.

Make sure to store the incident file on the cluster's worker node.

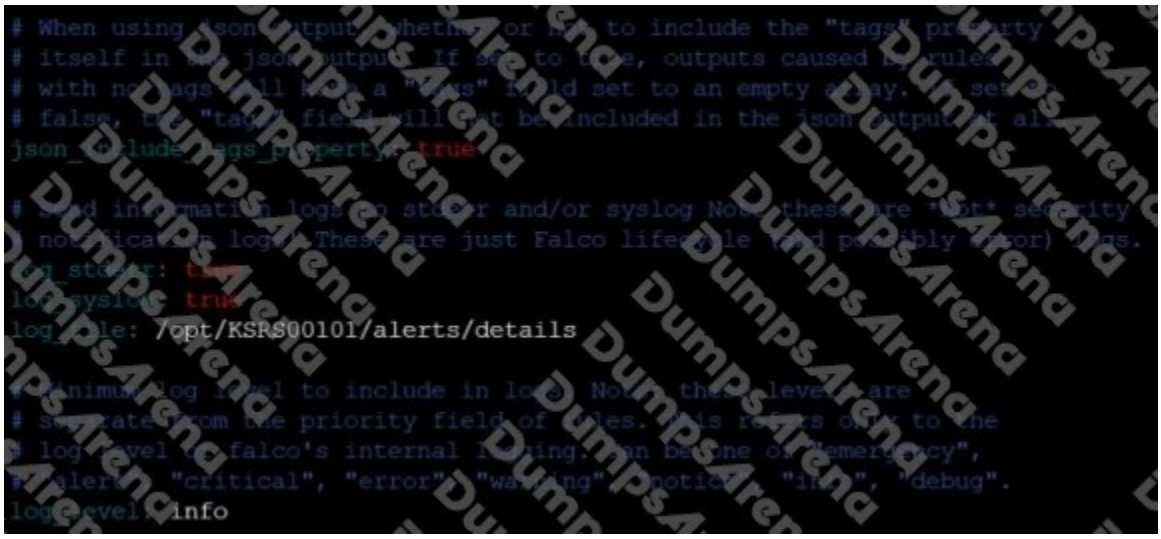
ANSWER: Seeexplanationbelow.

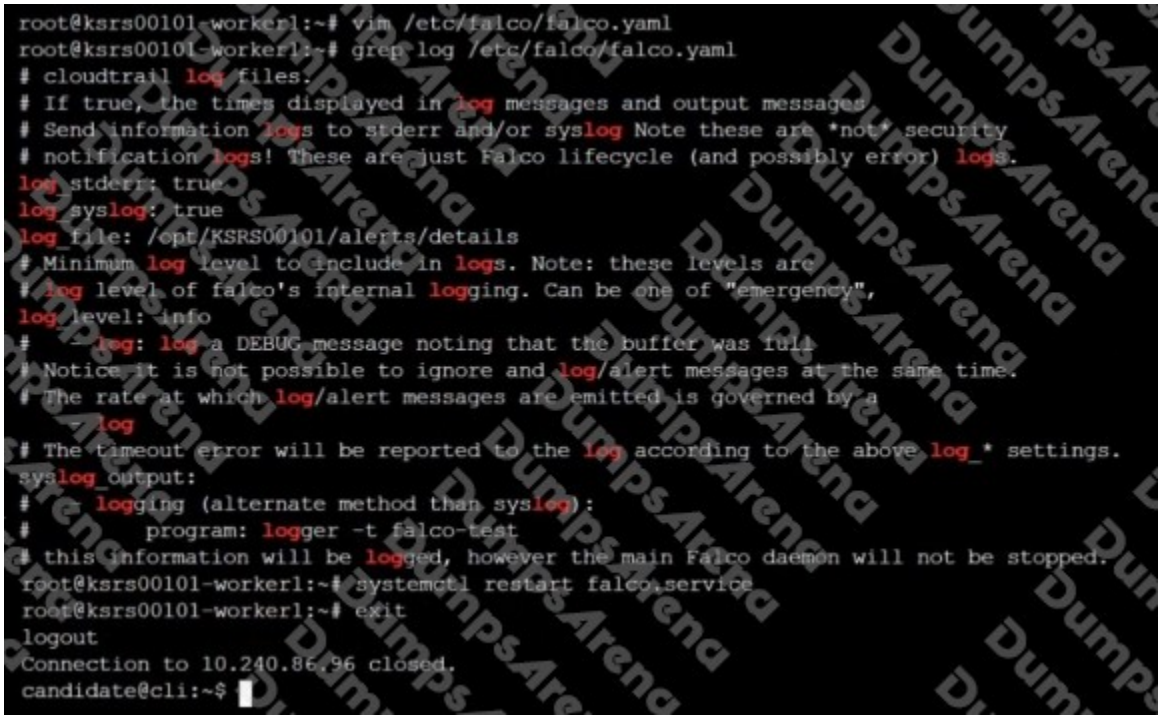
Explanation:


candidate@cli:~$ kubectl config use-context KSRS00101
Switched to context "KSRS00101".
candidate@cli:~$ ssh ksrs00101-worker1
Warning: Permanently added '10.240.86.96' (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@ksrs00101-worker1:~# falco
falco falco-driver-loader
root@ksrs00101-worker1:~# ls -l /etc/falco/
total 200
-rw-r--r-- 1 root root 12399 Jan 31 16:06 aws_cloudtrail_rules.yaml
-rw-r--r-- 1 root root 11384 Jan 31 16:06 falco.yaml
-rw-r--r-- 1 root root 1136 Jan 31 16:06 falco_rules.local.yaml
-rw-r--r-- 1 root root 132112 Jan 31 16:06 falco_rules.yaml
-rw-r--r-- 1 root root 27289 Jan 31 16:06 k8s_audit_rules.yaml
drwxr-xr-x 2 root root 4096 Feb 16 01:07 rules.available
drwxr-xr-x 2 root root 4096 Jan 31 16:28 rules.d
root@ksrs00101-worker1:~# vim /etc/falco/falco_rules.local.yaml

```
- rule: Container Drift Detected (chmod)
  desc: New executable created in a container due to chmod
  condition: >
    evt.type in (open,openat,create) and
    evt.is_open_exec=true and
    container and
    not runc_writing_exec_fifo and
    not runc_writing_var_lib_docker and
    not user_known_container_drift_activities and
    evt.rawres>=0
  output: >
    %evt.time,%user.uid,%proc.name
  priority: ERROR
```

```
root@ksrs00101-worker1:~# vim /etc/falco/falco_rules.local.yaml
root@ksrs00101-worker1:~# systemctl status falco.service
● falco.service - Falco Runtime Security
     Loaded: loaded (/lib/systemd/system/falco.service; disabled; vendor preset: enabled)
     Active: inactive (dead)
root@ksrs00101-worker1:~# systemctl enable falco.service
Created symlink /etc/systemd/system/multi-user.target.wants/falco.service → /lib/systemd/sys
tem/falco.service.
root@ksrs00101-worker1:~# systemctl start falco.service
root@ksrs00101-worker1:~# exit
logout
Connection to 10.240.86.96 closed.
candidate@cli:~$ ssh ksrs00101-worker1
Last login: Fri May 20 15:59:48 2022 from 10.240.86.88
root@ksrs00101-worker1:~# vim /etc/falco/falco.yaml
```





QUESTION NO: 9 - (SIMULATION)

Context:Cluster: prodMaster node: master1Worker node: worker1

You can switch the cluster/configuration context using the following command:

[desk@cli] $ kubectl config use-context prod

Task:Analyse and edit the given Dockerfile (based on the ubuntu:18:04 image)/home/cert_masters/Dockerfile fixing two instructions present in the file being prominent security/best-practice issues.

Analyse and edit the given manifest file/home/cert_masters/mydeployment.yaml fixing two fields present in the file being prominent security/best-practice issues.

Note: Don't add or remove configuration settings; only modify the existing configuration settings, so that two configuration settings each are no longer security/best-practice concerns.Should you need an unprivileged user for any of the tasks, use user nobody with user id 65535

ANSWER: Seetheexplanationbelow

Explanation:

1. For Dockerfile: Fix the image version & user name in Dockerfile2. For mydeployment.yaml : Fix security contexts

Explanation

[desk@cli] $ vim /home/cert_masters/Dockerfile

FROM ubuntu:latest # Remove this

FROM ubuntu:18.04 # Add this

USER root # Remove this

USER nobody # Add this

RUN apt get install -y lsof=4.72 wget=1.17.1 nginx=4.2

ENV ENVIRONMENT=testing

USER root # Remove this

USER nobody # Add this

CMD ["nginx -d"]

```
FROM ubuntu:latest  # Remove this
FROM ubuntu:18.04   # Add this
USER root           # Remove this
USER nobody         # Add this
RUN apt get install -y lsof=4.72 wget=1.17.1 nginx=4.2
ENV  ENVIRONMENT=testing
USER root           # Remove this
USER nobody         # Add this
CMD ["nginx -d"]
```

[desk@cli] $ vim /home/cert_masters/mydeployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

creationTimestamp: null

labels:

app: kafka

name: kafka

spec:

replicas: 1

selector:

matchLabels:

app: kafka

strategy: {}

template:

metadata:

creationTimestamp: null

labels:

app: kafka

spec:

containers:

- image: bitnami/kafka

name: kafka

volumeMounts:

- name: kafka-vol

mountPath: /var/lib/kafka

securityContext:

{"capabilities":{"add":["NET_ADMIN"],"drop":["all"]},"privileged": True,"readOnlyRootFilesystem": False, "runAsUser": 65535}
Delete This

{"capabilities":{"add":["NET_ADMIN"],"drop":["all"]},"privileged": False,"readOnlyRootFilesystem": True, "runAsUser": 65535}
Add This

resources: {}

volumes:

- name: kafka-vol

emptyDir: {}

status: {}

Pictorial View:[desk@cli] $ vim /home/cert_masters/mydeployment.yaml

Reference: https://kubernetes.io/docs/concepts/policy/pod-security-policy/

QUESTION NO: 10 - (SIMULATION)

Cluster: scannerMaster node: controlplaneWorker node: worker1

You can switch the cluster/configuration context using the following command:

[desk@cli] $ kubectl config use-context scanner

Given:You may use Trivy's documentation.

Task:Use the Trivy open-source container scanner to detect images with severe vulnerabilities used by Pods in the namespace nato.

Look for images with High or Critical severity vulnerabilities and delete the Pods that use those images.Trivy is pre-installed on the cluster's master node. Use cluster's master node to use Trivy.

ANSWER: Seetheexplanationbelow

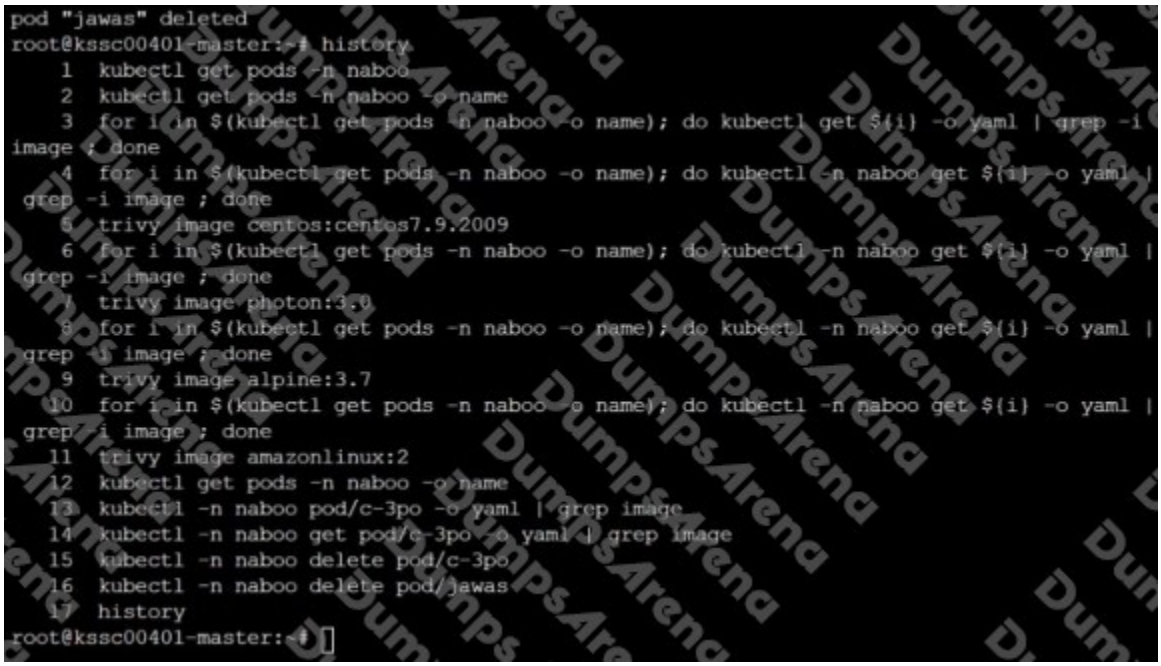
Explanation:


candidate@cli:~$ kubectl config use-context KSSC00401
Switched to context "KSSC00401".
candidate@cli:~$ ssh kssc00401-master
Warning: Permanently added '10.240.86.231' (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@kssc00401-master:~# kubectl get pods -n naboo
NAME READY STATUS RESTARTS AGE
c-3po 1/1 Running 0 6h48m
chewbacca 1/1 Running 0 6h48m
jawas 1/1 Running 0 6h48m
qui-gon-jinn 1/1 Running 0 6h48m
root@kssc00401-master:~# kubectl get pods -n naboo -o name
pod/c-3po
pod/chewbacca
pod/jawas
pod/qui-gon-jinn
root@kssc00401-master:~# for i in $(kubectl get pods -n naboo -o name)
> do
> kubectl get ${i} -o yaml | grep -i image
> done
Error from server (NotFound): pods "c-3po" not found
Error from server (NotFound): pods "chewbacca" not found
Error from server (NotFound): pods "jawas" not found
Error from server (NotFound): pods "qui-gon-jinn" not found
root@kssc00401-master:~# for i in $(kubectl get pods -n naboo -o name); do kubectl -n naboo
get ${i} -o yaml | grep -i image ; done
image: centos:centos7.9.2009
imagePullPolicy: Never
image: centos:centos7.9.2009
imageID: docker-pullable://centos@sha256:c73f515d06b0fa07bb18d8202035e739a494ce760aa7312
9f60f4bf2bd22b407
image: photon:3.0
imagePullPolicy: Never
image: photon:3.0
imageID: docker-pullable://photon@sha256:c48d61f0f3ad19215b75e2087cfbe95d7321abb454e4295
a0e6c38f563ece622
image: alpine:3.7
imagePullPolicy: Never
image: alpine:3.7
imageID: docker-pullable://alpine@sha256:8421d9a84432575381bfabd248f1eb56f3aa21d9d7cd251
1583c68c9b7511d10
image: amazonlinux:2
imagePullPolicy: Never
image: amazonlinux:2
imageID: docker-pullable://amazonlinux@sha256:246ef631c75ea83005889621119fd5cc9cbb5500e1
93707c38b6c060d597a146
root@kssc00401-master:~# trivy image centos:centos7.9.2009
2022-05-20T15:39:51.733Z INFO Need to update DB
2022-05-20T15:39:51.733Z INFO Downloading DB...
27.97 MiB / 27.97 MiB [----------] 100.00% 27.43 MiB p/s 1s

```
root@kssc00401-master:~# for i in $(kubectl get pods -n naboo -o name); do kubectl -n naboo
get ${i} -o yaml | grep -i image ; done
    image: centos:centos7.9.2009
    imagePullPolicy: Never
    image: centos:centos7.9.2009
    imageID: docker-pullable://centos@sha256:c73f515d06b0fa07bb18d8202035e739a494ce760aa7312
9f60f4bf2bd22b407
    image: photon:3.0
    imagePullPolicy: Never
    image: photon:3.0
    imageID: docker-pullable://photon@sha256:c48d61f0f3ad19215b75e2087cfbe95d7321abb454e4295
a0e6c38f563ecc622
    image: alpine:3.7
    imagePullPolicy: Never
    image: alpine:3.7
    imageID: docker-pullable://alpine@sha256:8421d9a84432575381bfabd248f1eb56f3aa21d9d7cd251
1583c68c9b7511d10
    image: amazonlinux:2
    imagePullPolicy: Never
    image: amazonlinux:2
    imageID: docker-pullable://amazonlinux@sha256:246ef631c75ea83005889621119fd5cc9cbb5500e1
93707c38b6c060d597a146
root@kssc00401-master:~# trivy image photon:3.0
2022-05-20T15:40:18.003Z        INFO    Detected OS: photon
2022-05-20T15:40:18.003Z        INFO    Detecting Photon Linux vulnerabilities...
2022-05-20T15:40:18.005Z        INFO    Number of language-specific files: 0

photon:3.0 (photon 3.0)
=======================
Total: 0 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 0, CRITICAL: 0)
```

```
root@kssc00401-master:~# kubectl get pods -n naboo -o name
pod/c-3po
pod/chewbacca
pod/jawas
pod/qui-gon-jinn
root@kssc00401-master:~# kubectl -n naboo pod/c-3po -o yaml | grep image
Error: flags cannot be placed before plugin name: -n
root@kssc00401-master:~# kubectl -n naboo get pod/c-3po -o yaml | grep image
    image: centos:centos7.9.2009
    imagePullPolicy: Never
    image: centos:centos7.9.2009
    imageID: docker-pullable://centos@sha256:c73f515d06b0fa07bb18d8202035e739a494ce760aa7312
9f60f4bf2bd22b407
root@kssc00401-master:~# kubectl -n naboo delete pod/c-3po
pod "c-3po" deleted
root@kssc00401-master:~# kubectl -n naboo delete pod/jawas
pod "jawas" deleted
```



Reference: https://github.com/aquasecurity/trivy