



CKS^{Q&As}

Certified Kubernetes Security Specialist (CKS) Exam

Pass Linux Foundation CKS Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.passapply.com/cks.html>

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Linux Foundation Official Exam Center

- ⚙️ **Instant Download** After Purchase
- ⚙️ **100% Money Back** Guarantee
- ⚙️ **365 Days** Free Update
- ⚙️ **800,000+** Satisfied Customers





QUESTION 1

Context:

Cluster: prod

Master node: master1

Worker 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

A. See the explanation below

B. Placeholder

Correct Answer: A

1. For Dockerfile: Fix the image version and 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" ]
```



Text

```
[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

```
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: {}
```

QUESTION 2

Create a new NetworkPolicy named deny-all in the namespace testing which denies all traffic of type ingress and egress traffic

A. See the explanation below:

B. Placeholder

Correct Answer: A

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.

QUESTION 3

Given an existing Pod named nginx-pod running in the namespace test-system, fetch the service-account-name used and put the content in /candidate/KSC00124.txt

Create a new Role named dev-test-role in the namespace test-system, which can perform update operations, on resources of type namespaces.

Create a new RoleBinding named dev-test-role-binding, which binds the newly created Role to the Pod's ServiceAccount (found in the Nginx pod running in namespace test- system).

A. See explanation below.

B. Placeholder

Correct Answer: A



Explanation/Reference:

```
candidate@cli:~$ kubectl config use-context KSCH00201
Switched to context "KSCH00201".
candidate@cli:~$ kubectl get pods -n security
NAME      READY   STATUS    RESTARTS   AGE
web-pod   1/1     Running   0           6h9m
candidate@cli:~$ kubectl get deployments.apps -n security
No resources found in security namespace.
candidate@cli:~$ kubectl describe rolebindings.rbac.authorization.k8s.io -n security
Name:      dev-role
Labels:     <none>
Annotations: <none>
Role:
  Kind: Role
  Name: dev-role
Subjects:
  Kind      Name      Namespace
  ----      -
  ServiceAccount sa-dev-1
candidate@cli:~$ kubectl describe role dev-role -n security
Name:      dev-role
Labels:     <none>
Annotations: <none>
PolicyRule:
  Resources  Non-Resource URLs  Resource Names  Verbs
  ----
  *          []                []              [*]
```

```
uid: b4c9ddd6-2729-43bd-8fbd-b2d227f4c4cd
rules:
- apiGroups:
  - ""
  resources:
  - services
  verbs:
  - watch
```

```
candidate@cli:~$ kubectl describe role dev-role -n security
Name:      dev-role
Labels:     <none>
Annotations: <none>
PolicyRule:
  Resources  Non-Resource URLs  Resource Names  Verbs
  ----
  *          []                []              [*]
candidate@cli:~$ kubectl edit role/dev-role -n security
role.rbac.authorization.k8s.io/dev-role edited
candidate@cli:~$ kubectl describe role dev-role -n security
Name:      dev-role
Labels:     <none>
Annotations: <none>
PolicyRule:
  Resources  Non-Resource URLs  Resource Names  Verbs
  ----
  services   []                []              [watch]
candidate@cli:~$ kubectl get pods -n security
NAME      READY   STATUS    RESTARTS   AGE
web-pod   1/1     Running   0           6h12m
candidate@cli:~$ kubectl get pods/web-pod -n security -o yaml | grep serviceAccount
serviceAccount: sa-dev-1
serviceAccountName: sa-dev-1
- serviceAccountToken:
candidate@cli:~$ kubectl create role role-2 --verb=update --resource=namespaces -n security
role.rbac.authorization.k8s.io/role-2 created
candidate@cli:~$ kubectl create rolebinding role-2-binding --role
--role --role=
candidate@cli:~$ kubectl create rolebinding role-2-binding --role=role-2 --serviceaccount=se
curity:sa-dev-1 -n security
rolebinding.rbac.authorization.k8s.io/role-2-binding created
candidate@cli:~$
```


**QUESTION 4**

CORRECT TEXT

Task

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



Cluster	Master node	Worker node
KSSH00301	kssh00301-master	kssh00301-worker1

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

```
[candidate@cli] $ | kubectl config use-context KSSH00301
```

Create a NetworkPolicy named pod-access to restrict access to Pod users-service running in namespace dev-team.



Only allow the following Pods to connect to Pod users-service:

1.

Pods in the namespace qa

2.

Pods with label environment: testing, in any namespace

Make sure to apply the
NetworkPolicy.

You can find a skeleton
manifest file at
`/home/candidate/KSSH00301/network-policy.yaml`

A. See explanation below.

B. Placeholder

Correct Answer: A

Explanation

Explanation/Reference:

```
candidate@cli:~$ kubectl config use-context KSSH00301
Switched to context "KSSH00301".
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ kubectl get ns dev-team --show-labels
NAME          STATUS   AGE      LABELS
dev-team      Active   6h39m    environment=dev,kubernetes.io/metadata.name=dev-team
candidate@cli:~$ kubectl get pods -n dev-team --show-labels
NAME          READY   STATUS    RESTARTS   AGE      LABELS
users-service 1/1     Running   0           6h40m    environment=dev
candidate@cli:~$ ls
KSCH00301  KSMV00102  KSSC00301  KSSH00401  test-secret-pod.yaml
KSCS00101  KSMV00301  KSSH00301  password.txt username.txt
candidate@cli:~$ vim np.yaml
```

A. See explanation below.

B. Placeholder

Correct Answer: A



QUESTION 5

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. See explanation below.

B. Placeholder

Correct Answer: A

[CKS PDF Dumps](#)

[CKS Practice Test](#)

[CKS Study Guide](#)