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.geekcert.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

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: pod-access
  namespace: dev-team
spec:
  podSelector:
    matchLabels:
      environment: dev
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              environment: dev
        - podSelector:
            matchLabels:
              environment: testing
```

```
candidate@cli:~$ vim np.yaml
candidate@cli:~$ cat np.yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: pod-access
  namespace: dev-team
spec:
  podSelector:
    matchLabels:
      environment: dev
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              environment: dev
        - podSelector:
            matchLabels:
              environment: testing
candidate@cli:~$
candidate@cli:~$
candidate@cli:~$ kubectl create -f np.yaml -n dev-team
networkpolicy.networking.k8s.io/pod-access created
candidate@cli:~$ kubectl describe netpol -n dev-team
Name:         pod-access
Namespace:    dev-team
Created on:   2022-05-20 15:35:33 +0000 UTC
Labels:       <none>
Annotations:  <none>
Spec:
  PodSelector:     environment=dev
  Allowing ingress traffic:
    To Port: <any> (traffic allowed to all ports)
    From:
      NamespaceSelector: environment=dev
    From:
      PodSelector: environment=testing
  Not affecting egress traffic
  Policy Types: Ingress
candidate@cli:~$ cat KSSH00301/network-policy.yaml
---
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: ""
  namespace: ""
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress:
    - from: []
    - from: []
candidate@cli:~$ cp np.yaml KSSH00301/network-policy.yaml
candidate@cli:~$ cat KSSH00301/network-policy.yaml
```

```
candidate@cli:~$ cat KSSH00301/network-policy.yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: pod-access
  namespace: dev-team
spec:
  podSelector:
    matchLabels:
      environment: dev
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              environment: dev
        - podSelector:
            matchLabels:
              environment: testing
candidate@cli:~$
```

Enable audit logs in the cluster, To Do so, enable the log backend, and ensure that

1.

logs are stored at /var/log/kubernetes-logs.txt.

2.

Log files are retained for 12 days.

3.

at maximum, a number of 8 old audit logs files are retained.

4.

set the maximum size before getting rotated to 200MB

Edit and extend the basic policy to log:

1.

namespaces changes at RequestResponse

2.

Log the request body of secrets changes in the namespace kube-system.

3.

Log all other resources in core and extensions at the Request level.

4.

Log "pods/portforward", "services/proxy" at Metadata level.

5.

Omit the Stage RequestReceived

All other requests at the Metadata level

A. See the explanation below:

B. PlaceHolder

Correct Answer: A

Kubernetes auditing provides a security-relevant chronological set of records about a cluster. Kube-apiserver performs auditing. Each request on each stage of its execution generates an event, which is then pre-processed according to a

certain policy and written to a backend. The policy determines what\\'s recorded and the backends persist the records. You might want to configure the audit log as part of compliance with the CIS (Center for Internet Security) Kubernetes

Benchmark controls.

The audit log can be enabled by default using the following configuration in cluster.yml:

services:

kube-api:

audit_log:

enabled: true

When the audit log is enabled, you should be able to see the default values at /etc/kubernetes/audit-policy.yaml

The log backend writes audit events to a file in JSONlines format. You can configure the log audit backend using the following kube-apiserver flags:

--audit-log-path specifies the log file path that log backend uses to write audit events. Not specifying this flag disables log backend. - means standard out --audit-log-maxage defined the maximum number of days to retain old audit log files

--audit-log-maxbackup defines the maximum number of audit log files to retain

--audit-log-maxsize defines the maximum size in megabytes of the audit log file before it gets rotated

If your cluster\\'s control plane runs the kube-apiserver as a Pod, remember to mount the hostPath to the location of the policy file and log file, so that audit records are persisted.

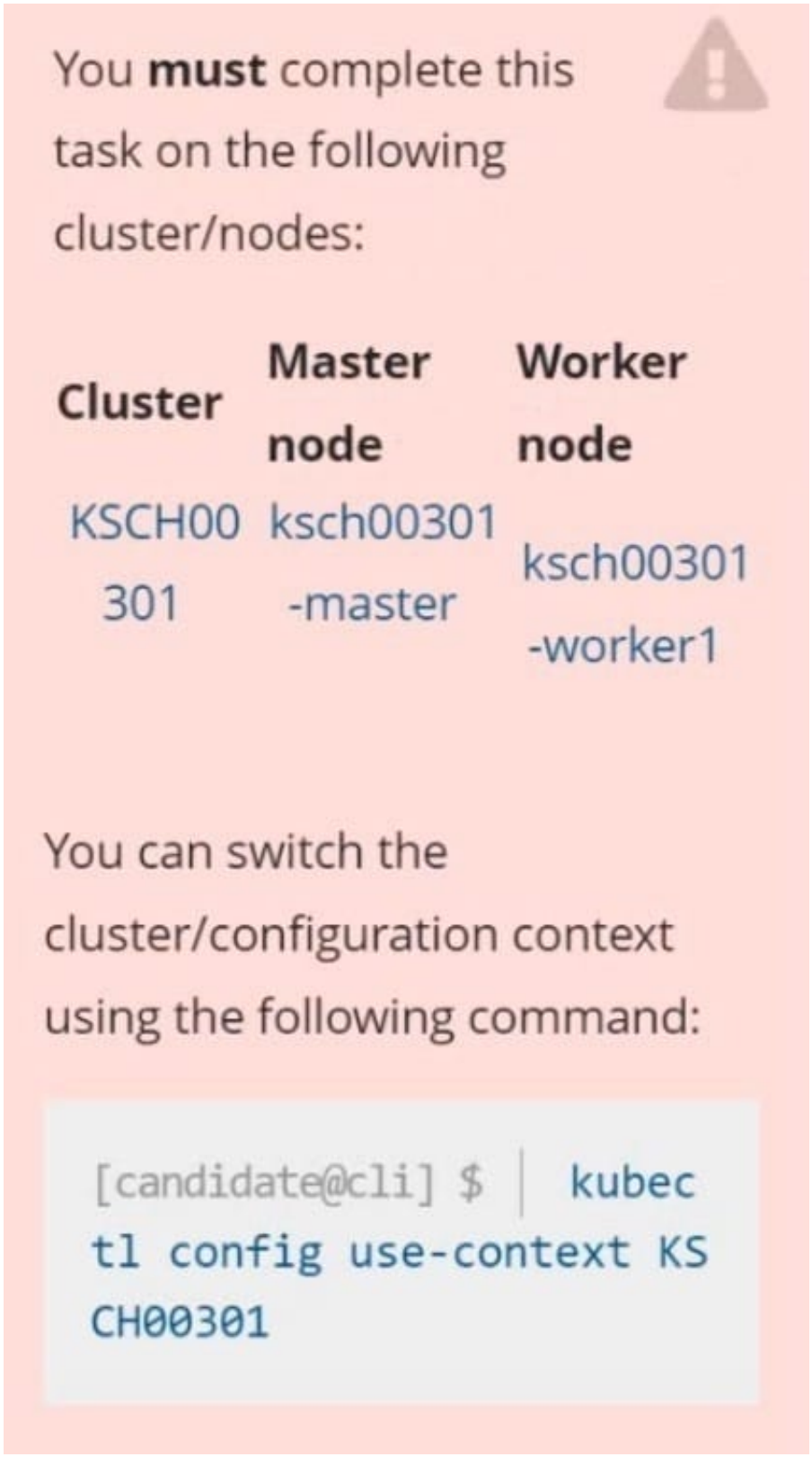
For example:

--audit-policy-file=/etc/kubernetes/audit-policy.yaml \

--audit-log-path=/var/log/audit.log

QUESTION 2

CORRECT TEXT Your organization\\'s security policy includes:



1.

ServiceAccounts must not automount API credentials

2.

ServiceAccount names must end in "-sa"

The Pod specified in the manifest file /home/candidate/KSCH00301 /pod-m

nifest.yaml fails to schedule because of an incorrectly specified ServiceAccount.

Complete the following tasks:

Task

1.

Create a new ServiceAccount named frontend-sa in the existing namespace qa. Ensure the ServiceAccount does not automount API credentials.

2.

Using the manifest file at /home/candidate/KSCH00301 /pod-manifest.yaml, create the Pod.

3.

Finally, clean up any unused ServiceAccounts in namespace qa.

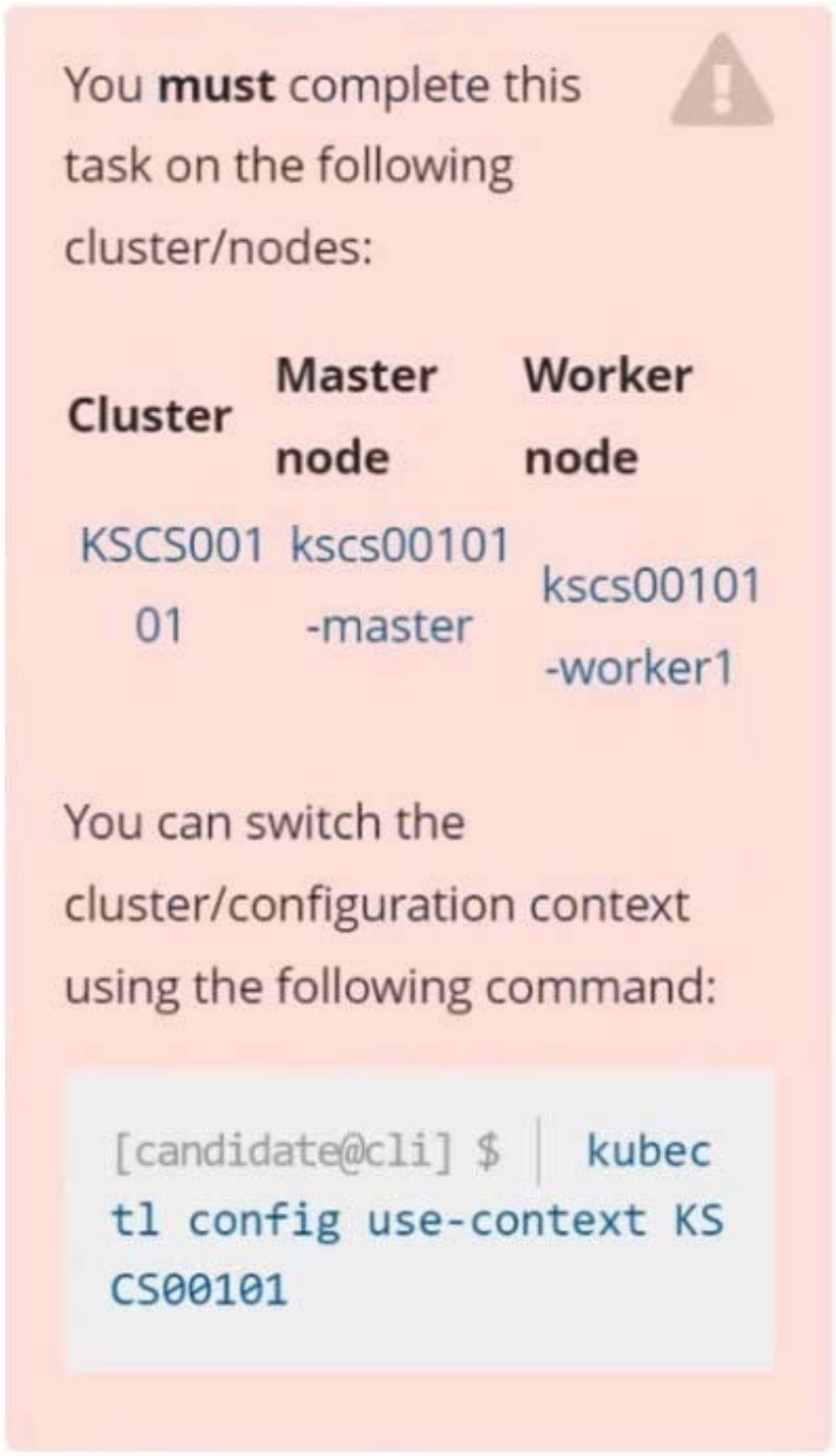
A. See the explanation below

B. PlaceHolder

Correct Answer: A

QUESTION 3

CORRECT TEXT Context



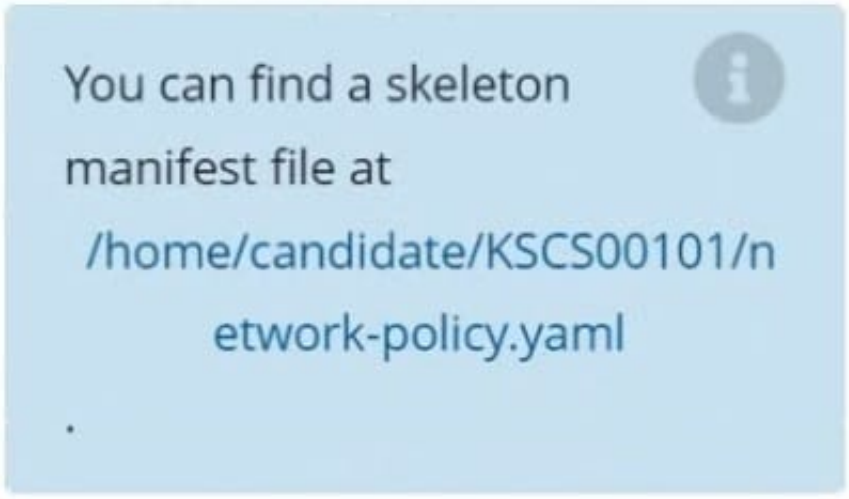
A default-deny NetworkPolicy avoids to accidentally expose a Pod in a namespace that doesn\\'t have any other NetworkPolicy defined.

Task

Create a new default-deny NetworkPolicy named defaultdeny in the namespace testing for all traffic of type Egress.

The new NetworkPolicy must deny all Egress traffic in the namespace testing.

Apply the newly created default-deny NetworkPolicy to all Pods running in namespace testing.



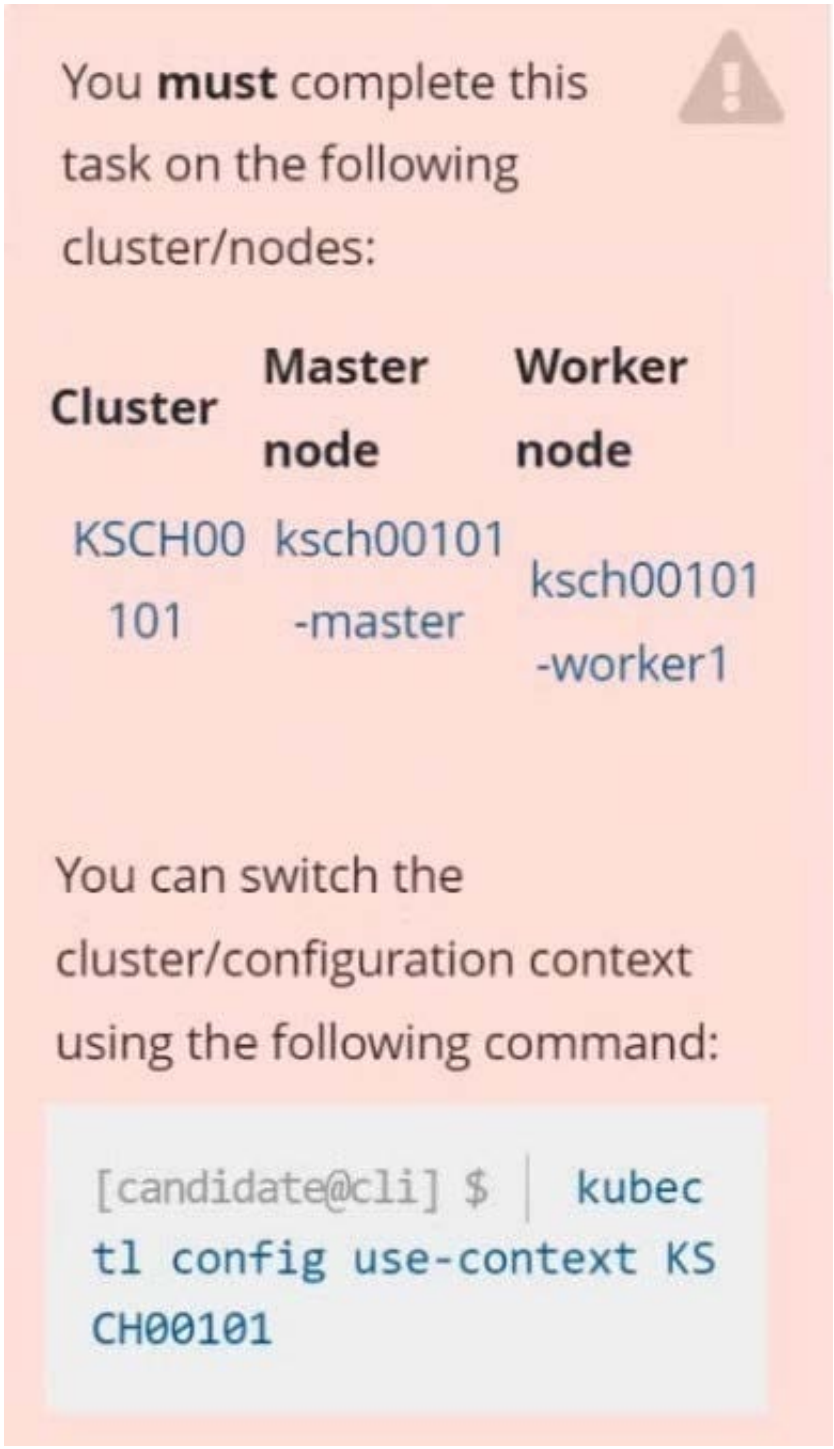
A. See explanation below.

B. PlaceHolder

Correct Answer: A

QUESTION 4

The kubeadm-created cluster\\'s Kubernetes API server was, for testing purposes, temporarily configured to allow unauthenticated and unauthorized access granting the anonymous user duster-admin access.



Task

Reconfigure the cluster\\'s Kubernetes API server to ensure that only authenticated and authorized REST requests are allowed.

Use authorization mode Node,RBAC and admission controller NodeRestriction.

Cleaning up, remove the ClusterRoleBinding for user system:anonymous.

A. See explanation below.

B. PlaceHolder

Correct Answer: A

```
candidate@cli:~$ kubectl config use-context KSCH00101
Switched to context "KSCH00101".
candidate@cli:~$ ssh ksch00101-master
Warning: Permanently added '10.240.86.190' (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@ksch00101-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  annotations:
    kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: 10.240.86.190:6443
  creationTimestamp: null
  labels:
    component: kube-apiserver
    tier: control-plane
  name: kube-apiserver
  namespace: kube-system
spec:
  containers:
    - command:
        - kube-apiserver
        - --advertise-address=10.240.86.190
        - --allow-privileged=true
        - --authorization-mode=Node,RBAC
        - --client-ca-file=/etc/kubernetes/pki/ca.crt
        - --enable-admission-plugins=AlwaysAdmit
        - --enable-bootstrap-token-auth=true
        - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
        - --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt
        - --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
"/etc/kubernetes/manifests/kube-apiserver.yaml" 128L, 4343C                  1,1           Top
```

```
root@ksch00101-master:~# cat /etc/kubernetes/admin.conf
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUMvakNDQWVhZ0F3SUJB
Z0lCQURBTkJna3Foa21HOXcwQkFRc0ZBREFWTVJNd0VRWURWUVFERXdwcmRXSmwKY201bGRHVnpNQjRYRFRJeU1ESXhO
akF3T1RVeE9Wb1hcEVE15TURJeE5EQXdOVFV4T1Zvd0ZURVRNQkVHQTFVRQpBeE1LYTNWaVpYSnVaWFJsY3pDQ0FTSXdE
UVlKS29aSWh2Y05BQUVCQlFBRGdnRVBBRENDQVFvQ2dnRUJBTlgwCm9LeUYvTGNmYTIvNzNZTktkSFdZU3JUaUx0QStr
NG1qTXpRZllzM2ttNGllalpoM0tZc3Y1bUdpNOUyQ2tYc0MKUnhlLlNiZnBDMzlla2k5V3hOSHc5eTM0OEtXUVE3VXBL
UmZRdXVxd1A1WXdDZkordlJmWGNGTXQxLzRNQVhWLwpkdjZ5YWRKSitPeFFSVjZlaHFBZHR0M3FtOFdVcW84UE5JT1E0
OEc3WWhnRUg5RHU3SFdkMS8raXVkSjNOMXl6CnNISEdtYklsWENSbEcydFV0M2RScDczSnRIS1JjS2tnMGxYM3FWSlUy
QmJRblBmK01wb0V1TXFGcmZvcWVaVWcKYlBKK3ROVmZIM1JLTkhVUnYydVJIa3ZZc2JrclhUMW8rMXFNNHZrYnFNMHlq
KzNxTUtiSyt5V3dzUTlBYUVPMApUdXR4UUdlTFp3OUE3TjZZeTFVQ0F3RUFBYU5aTUZjd0RnWURWUjBQQVFLL0JBUURB
Z0trTUE4R0ExVWRFd0VCCi93UUZNQUlCQWY4d0hBWURWUjBPQkJZRUZEcUlwLzdYbzZaNkJNVjVEK2w3bFZPcGpBOWlN
QlVHQTFVZEVRUU8KTUF5QONtdDFZbVZ5YmlWMFpYTXdEUVlKS29aSWh2Y05BQUVMQlFBRGdnRUJBSlNWNm9wNGgxYkNv
eGZLRUZ4bwoxaVlHUFlnMlhhOTNOWEZlTTY3RnA2NkdqUEc5SXBONnNHUnRnWVlyd0MyalBDeFVOb2IySWtUQlFNbDV3
cWRHCkdPS2JwVVp6Smc3YOdyS2E3RlpZWVNyVUVGRWhyd2xZWXNGME56aFBoZVcwcHJjcWtSdXNlbm55SG5YNGVOMUoK
N1NzbGZYTjJIdVFJd1VIRGl5LOJsL1ZWRmZNZnRxOGF0Z0pYSFZGTmlVcDRpNXlJTXFRNTB4ZjVqcnF1WFRmWpVdmJq
ZjEyOThxVTk3QkxHcDdRZE9QYWVKU051UStlVkMrdnpVZ2tVQVNjclVsc24xcThPNnBRbjV3TjNxdUVrCm5zQk9pckxS
c2k2alN3U1hLbGcvangvcitqd0dTc0xwWUxDZTlxalFraTdCSVRJTlN3ejd3c2hzbERuNzBFY0IKa0VBPQotLS0tLUVO
RCBDRVJUSUZJQ0FURS0tLS0tCg==
    server: https://10.240.86.190:6443
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
kind: Config
preferences: {}
users:
- name: kubernetes-admin
  user:
    client-certificate-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURJVENDQWdtZ0F3SUJBZ01
ocEdQcDB4Zk9JbkYxaGJwcThSYlBUMGxlTm5VNjBiSUpxRXVKckxJbEtXC1NVa1h1VkYzNklOZHc1ZUlOT2JxK1haaHd
hY2JURVZCMlVDVURsbDgzdG5teFQyVXJmY0pUQmhlTCtzTFAvcWYKdjdXR3BwQlZXNnhVZGFibGNuUklIMnpleUVJTEt
Tck5XbUQ0TzZzMU13blZOOVJzQ2RXTkV3VGNZRHdoUTd2OQpGcExxKL3hiSDdUTzkwY1RFd1Iwazl3cFVYdllkdklj5XN
MRkYwL3F2bDA3U3lxbGplOEI1SnNpQlhCUlZxbS9wCmNUUSs3SnZlbmdaZzlkOWd2aVJVdFFTcHBONkx4UnhkSzNKMGR
BK240SWxFZEtHRWh3TE00d0tMaldERG9scHgKYzB3WHkwVXBORGZ6UUxuRUFzVUJsbDRCQ3VkdW5QNVVDN2FuS3dJREF
RQUJBb0lCQURWRkZNSVRqYnNySTZTTWpQOGM0MTByN3RWZ251cXJVS202dHRnZWtXOWd1S1pvMnZyb3RsbG9qOGFRamF
OMTZnaEUwOXdZd2xMSDhId0tLCk1Mb2NrZnFCUytlOWo1ZmlFWGxYTG00cElCVDFRbGFJQlJRMDRyQ0JZbHdCN1VFbVB
lWjhuQ3lmR2JYTC9HM2wKcXBYTDVKdzJqcVh2MXdzcWsrdWNCRk0zZ0FYZk5YZkk1RExnV0VyNXRZRlF4VXo5UFFHOD1
pcDY1OTBkYnB1SApoOMnU2NGk4UTg1dk83OFV1Tlc2eUFZU1loZVdha093RDFwZzNPdkhxV3FhbnV1MnlrOWxaUUR0WW5
2MytBeU5DCnloNlRaRHluZ0lZdEptbDFTQ01TNEpSR2d4NXNwaCtKOC9XOGx0Ri9wMWZxbTA0bXZSRndxU3M2Y1JCQ2Z
PVVcKbFV1MGxLRUNnWUVBNWJzT0lVVzFBVndjTmJsc0pSVDNURkI2OVlxbDRYcnZRR0FZY3BhdktENnd5VmtEOTV1QQp
SaXVRS1NNKzY4REtBVmlpYllpaThJemExTkdqdC9JZDUwTGVoNklaRVg2enVpK0g3d1BSbVd6SE9ueWNmU2FmC1VQMEF
RL0RiM2lCNWJQTmJHYXNkaDNIb2JvR0NSSHZmTFFXY2tYbUVXM2ZudVlIRlJLZ2xlTEVDZllFQTVUdysKTEVTV1BESFF
mamNBNOhtNmdsMndGRjdCUGlsSGdaYVVRN25Eb3ZvRmMxalBMRWVCMWJ6OHJNWld1eGdmaHNOOQpMZ0xSUDBXdkJWd1J
sVTdMTmFLTlVzRmkxU2dvaWZsSO1ZWkMyZmpLWTY1RFE3YUUzcTdnVis4U2pIZHpoclhCCkVQc1AvWXQ3SOQrbFBMZmh
aNXNKZWFtelY3b3gveno5Y0s0U0ZKc0NnWUJ4OVk2VzFydHBoMFcvS05JS3V4SEoKMjRxRFQxbm1ObE9FdmFhakFUaTJ
ZQkxXYnIvWERsNWRjTEs4bFcxYkNYR3JwY2s3U0xKNlhZUVlXajQ2dTNJMgpEQ2ZUWlFiRWRQTzNBbWtPR2ZqWmdPcDd
pdUVkLOJDLzNpRkprcXVlenNFdFdMTHlVcjM5T0hZeWl6QVJ4Tmo2CnZuUGlma000Rkl6d3F2MHVoN0xlb1FLQmdBT1Z
XZTRZMlRwbzJ3aEswbmVkMllsMXhVNjJoZ2JiVHcvaVdhdVcKY3ZMV3dlZUlmd0Q4MVRWL2R3a29KVEM1VEJRUXQzUkk
xRFFnVmtnMHFwcUtOOGhDNGQwM05MRzIwTWdZMk94WgpjSFZzK2J4elYwVVB6V1RUbEMyVEsyamhmOHVRcndzSktxY2N
OU0ZEczZwclhocThsV213Znd3aWlBRlhLSFJRCkE3RkxBb0dCQUx3NW8rbHFVZ3hHQlpKdy9ZelBGek5TekQreVd6Um8
lc2ZEc2x6a2FvY0pHbEx2MUNndEVIc3QKeG5HMTlIYStSM1M3cDRtei9LeDJYMFRzaTZzUzVwWlR5WEx5STF5azh2TUZ
rRldacjRmeVhXV2t3SjZlVEllYwpyWFl3TWM5VFlDUGZrSFJaTm9XRlhZV3BkeTJBOXZCbFlScHZsQVZoenU2T1VZQ2w
5b2ZpCi0tLS0tRU5EIFJTQSBQUklWQVRFIEtFWS0tLS0tCg==
root@ksch00101-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
```

```
root@ksch00101-master:~# vim /etc/kubernetes/manifests/kube-apiserver.yaml
root@ksch00101-master:~# systemctl daemon-reload
sroot@ksch00101-master:~# systemctl restart kubelet.service
root@ksch00101-master:~# kubectl get nodes
error: You must be logged in to the server (Unauthorized)
root@ksch00101-master:~# exit
logout
Connection to 10.240.86.190 closed.
candidate@cli:~$ kubectl get nodes
NAME                STATUS   ROLES                  AGE   VERSION
ksch00101-master    Ready    control-plane,master   93d   v1.23.3
ksch00101-worker1   Ready    <none>                 93d   v1.23.3
candidate@cli:~$ kubectl get pod -n kube-system
NAME                                       READY   STATUS    RESTARTS       AGE
coredns-64897985d-7pnhm                    1/1     Running   1 (7h2m ago)   93d
coredns-64897985d-rr7sd                    1/1     Running   1 (7h2m ago)   93d
etcd-ksch00101-master                      1/1     Running   1 (7h2m ago)   93d
kube-apiserver-ksch00101-master            0/1     Running   0              24s
kube-controller-manager-ksch00101-master   1/1     Running   3 (42s ago)    93d
kube-flannel-ds-llktn                      1/1     Running   1 (93d ago)    93d
kube-flannel-ds-q9vnl                      1/1     Running   1 (93d ago)    93d
kube-proxy-2c4ht                           1/1     Running   1 (93d ago)    93d
kube-proxy-pmmbc                           1/1     Running   1 (93d ago)    93d
kube-scheduler-ksch00101-master            1/1     Running   3 (42s ago)    93d
candidate@cli:~$ kubectl get pod -n kube-system
NAME                                       READY   STATUS    RESTARTS       AGE
coredns-64897985d-7pnhm                    1/1     Running   1 (7h2m ago)   93d
coredns-64897985d-rr7sd                    1/1     Running   1 (7h2m ago)   93d
etcd-ksch00101-master                      1/1     Running   1 (7h2m ago)   93d
kube-apiserver-ksch00101-master            0/1     Running   0              30s
kube-controller-manager-ksch00101-master   1/1     Running   3 (48s ago)    93d
kube-flannel-ds-llktn                      1/1     Running   1 (93d ago)    93d
kube-flannel-ds-q9vnl                      1/1     Running   1 (93d ago)    93d
kube-proxy-2c4ht                           1/1     Running   1 (93d ago)    93d
kube-proxy-pmmbc                           1/1     Running   1 (93d ago)    93d
kube-scheduler-ksch00101-master            1/1     Running   3 (48s ago)    93d
candidate@cli:~$ kubectl get clusterrolebindings.rbac.authorization.k8s.io | grep anon
system:anonymous                                       ClusterRole/cluster-admin
                                                  7h1m
candidate@cli:~$ kubectl delete clusterrolebindings.rbac.authorization.k8s.io/system:anonymo
us
clusterrolebinding.rbac.authorization.k8s.io "system:anonymous" deleted
```

QUESTION 5

```
candidate@cli:~$ kubectl config use-context KSCS00101
Switched to context "KSCS00101".
candidate@cli:~$ cat /home/candidate/KSCS00101/network-policy.yaml
---
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: ""
  namespace: ""
spec:
  podSelector: {}
  policyTypes: []
candidate@cli:~$ vim /home/candidate/KSCS00101/network-policy.yaml
candidate@cli:~$
```

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: "defaultdeny"
  namespace: "testing"
spec:
  podSelector: {}
  policyTypes:
  - Egress
  egress:
  - to:
    - podSelector: {}
      namespaceSelector:
        matchLabels:
          access: testingproject
```

```
candidate@cli:~$ vim /home/candidate/KSCS00101/network-policy.yaml
candidate@cli:~$ vim /home/candidate/KSCS00101/network-policy.yaml
candidate@cli:~$ kubectl label ns testing access=testingproject
namespace/testing labeled
candidate@cli:~$ cat /home/candidate/KSCS00101/network-policy.yaml
---
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: "defaultdeny"
  namespace: "testing"
spec:
  podSelector: {}
  policyTypes:
  - Egress
  egress:
  - to:
    - podSelector: {}
      namespaceSelector:
        matchLabels:
          access: testingproject
candidate@cli:~$ kubectl create -f /home/candidate/KSCS00101/network-policy.yaml
networkpolicy.networking.k8s.io/defaultdeny created
candidate@cli:~$ kubectl -n testing describe networkpolicy
Name:         defaultdeny
Namespace:    testing
Created on:   2022-05-20 14:28:27 +0000 UTC
Labels:       <none>
Annotations:  <none>
Spec:
  PodSelector:     <none> (Allowing the specific traffic to all pods in this namespace)
  Not affecting ingress traffic
  Allowing egress traffic:
    To Port: <any> (traffic allowed to all ports)
    To:
      NamespaceSelector: access=testingproject
      PodSelector: <none>
  Policy Types: Egress
candidate@cli:~$
```

Create a RuntimeClass named gvisor-rc using the prepared runtime handler named runsc.

Create a Pods of image Nginx in the Namespace server to run on the gVisor runtime class

A. See the explanation below:

B. PlaceHolder

Correct Answer: A

Install the Runtime Class for gVisor

{ # Step 1: Install a RuntimeClass cat