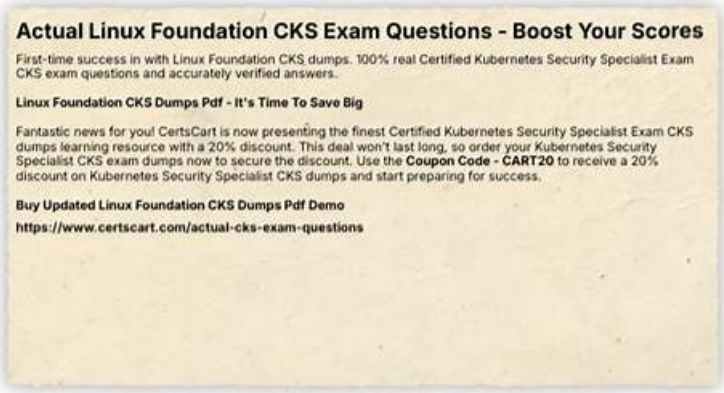
Exam CKS Dumps | Practice CKS Engine



If you want to get through the CKS practice exam quickly with less time and efforts, our learning materials is definitely your best option. One or two days' preparation and remember the correct CKS test answers, getting the certification will be simple for our candidates. Free trials of CKS Exam PDF are available for everyone and great discounts are waiting for you. Join us and realize your dream.

To take the CKS Certification Exam, candidates must have a valid CNCF (Cloud Native Computing Foundation) CKA (Certified Kubernetes Administrator) certification, which demonstrates their proficiency in Kubernetes administration. Candidates must also have experience working with Kubernetes in production environments and have a good understanding of Linux command-line tools and utilities.

>> Exam CKS Dumps <<

Desktop Based CKS Certified Kubernetes Security Specialist (CKS) Practice Test Software

Although a lot of products are cheap, but the quality is poor, perhaps users have the same concern for our CKS learning materials. Here, we solemnly promise to users that our product error rate is zero. Everything that appears in our products has been inspected by experts. In our CKS learning material, users will not even find a small error, such as spelling errors or grammatical errors. It is believed that no one is willing to buy defective products, so, the CKS study materials have established a strict quality control system.

Linux Foundation Certified Kubernetes Security Specialist (CKS) Sample Questions (Q42-Q47):

NEW QUESTION # 42
Create a PSP that will only allow the persistentvolumeclaim as the volume type in the namespace restricted.
Create a new PodSecurityPolicy named prevent-volume-policy which prevents the pods which is having different volumes mount apart from persistentvolumeclaim.
Create a new ServiceAccount named psp-sa in the namespace restricted.
Create a new ClusterRole named psp-role, which uses the newly created Pod Security Policy prevent-volume-policy Create a new ClusterRoleBinding named psp-role-binding, which binds the created ClusterRole psp-role to the created SA psp-sa.
Hint:
Also, Check the Configuration is working or not by trying to Mount a Secret in the pod maifest, it should get failed.
POD Manifest:
apiVersion: v1
kind: Pod
metadata:
name:
spec:
containers:
- name:
image:

volumeMounts:
- name:
mountPath:
volumes:
- name:
secret:
secretname:

Answer:

Explanation:
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
name: restricted
annotations:
seccomp.security.alpha.kubernetes.io/allowedProfileNames: 'docker/default,runtime/default'
apparmor.security.beta.kubernetes.io/allowedProfileNames: 'runtime/default'
seccomp.security.alpha.kubernetes.io/defaultProfileName: 'runtime/default'
apparmor.security.beta.kubernetes.io/defaultProfileName: 'runtime/default' spec:
privileged: false
Required to prevent escalations to root.
allowPrivilegeEscalation: false
This is redundant with non-root + disallow privilege escalation,
but we can provide it for defense in depth.
requiredDropCapabilities:
- ALL
Allow core volume types.
volumes:
- 'configMap'
- 'emptyDir'
- 'projected'
- 'secret'
- 'downwardAPI'
Assume that persistentVolumes set up by the cluster admin are safe to use.
- 'persistentVolumeClaim'
hostNetwork: false
hostIPC: false
hostPID: false
runAsUser:
Require the container to run without root privileges.
rule: 'MustRunAsNonRoot'
seLinux:
This policy assumes the nodes are using AppArmor rather than SELinux.
rule: 'RunAsAny'
supplementalGroups:
rule: 'MustRunAs'
ranges:
Forbid adding the root group.
- min: 1
max: 65535
fsGroup:
rule: 'MustRunAs'
ranges:
Forbid adding the root group.
- min: 1
max: 65535
readOnlyRootFilesystem: false

NEW QUESTION # 43

You need to implement a secure network policy that allows communication only between specific pods within a namespace. For example, you want to allow communication between pods that have the label 'app=frontend' and pods that have the label 'app=backend', but block all other communication within the namespace.

Answer:

Explanation:
Solution (Step by Step) :
1. Create a NetworkPolicy:
- Define a NetworkP01icy that allows communication between 'frontend' and 'backend' pods, but blocks other communication within the namespace.

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-frontend-backend
  namespace: my-namespace
spec:
  podSelector: {}  # Apply to all pods in the namespace
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: frontend
    - podSelector:
        matchLabels:
          app: backend
    ports:
    - protocol: TCP
      port: 80
  egress:
  - to:
    - podSelector:
        matchLabels:
          app: frontend
    - podSelector:
        matchLabels:
          app: backend
    ports:
    - protocol: TCP
      port: 80
```

2. Create a Frontend Pod: - Create a Pod with the label 'app=frontend'.

```
apiVersion: v1
kind: Pod
metadata:
  name: frontend-pod
  namespace: my-namespace
  labels:
    app: frontend
spec:
  containers:
  - name: nginx
    image: nginx:1.14.2
    ports:
    - containerPort: 80
```

3. Create a Backend Pod: - Create a Pod With the label 'app=backend'.

```
apiVersion: v1
kind: Pod
metadata:
  name: backend-pod
  namespace: my-namespace
  labels:
    app: backend
spec:
  containers:
  - name: nginx
    image: nginx:1.14.2
    ports:
    - containerPort: 80
```

4. Apply the YAML files: - Apply the created YAML files using ' kubectl apply -f 5. Verify the Network Policy: - Try to connect from the 'frontend-pod' to the 'backend-pod' (e.g., using ' kubectl exec -it frontend-pod bash' and 'curl backend-pod:80')- It should succeed. - Try to connect from the 'frontend-pod' to another pod in the namespace that doesn't have the Sapp-backend' label. This connection should be blocked.

NEW QUESTION # 44
You are managing a Kubernetes cluster where workloads are spread across multiple nodes- You want to configure Pod Security Policies PSPS to restrict the use of privileged containers and limit the ca abilities of containers running within your cluster.

Answer:

Explanation:
Solution (Step by Step) :
1. Create a Pod Security Policy:
- Create a PSP YAML file named restricted-psp.ya'r:

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted-psp
spec:
  # Allow only non-root users
  runAsUser:
    rule: "MustRunAsNonRoot"
  # Allow only specific capabilities
  allowedCapabilities:
    - CAP_NET_BIND_SERVICE
    - CAP_CHOWN
  # No privileged containers allowed
  privileged: false
  # Allow only specific volumes
  volumes:
    - 'hostPath'
    - 'emptyDir'
    - 'projected'
    - 'configMap'
    - 'secret'
    - 'persistentVolumeClaim'
```

2. Apply the Pod Security Policy: - Apply the PSP using 'kubectl apply -f restricted-psp.yaml' 3. Create a Deployment using the PSP: - Create a new deployment YAML file named 'test-deployment.yamr that specifies the 'restricted-psp' for the pod's security context:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: test-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: test-app
  template:
    metadata:
      labels:
        app: test-app
    spec:
      containers:
        - name: test-container
          image: nginx:latest
          securityContext:
            securityContext:
              pspName: restricted-psp
```

4. Apply the Deployment: - Apply the deployment using 'kubectl apply -f test-deployment.yaml 5. Test the Restrictions: - Try creating a pod that violates the PSP, for example, using a privileged container. The pod should fail to be created due to the PSP enforcement - Try running a command within a using the deployment that uses the PSP. You should be able to run commands but may have limitations based on the capabilities allowed by the PSP.

NEW QUESTION # 45
SIMULATION
Using the runtime detection tool Falco, Analyse the container behavior for at least 30 seconds, using filters that detect newly spawning and executing processes store the incident file art /opt/falco-incident.txt, containing the detected incidents. one per line, in

the format
[timestamp],[uid],[user-name],[processName]

- A. Sendusyoursuggestiononit

Answer: A

NEW QUESTION # 46
You are tasked With securing a Kubernetes cluster that is hosting sensitive applications. You need to implement a robust security posture, including network segmentation, secure communication, and authentication. Explain how you would leverage Pod Security Policies (PSPs) to enforce security controls for pods in the cluster. Provide a practical example of a PSP definition that enforces specific security restrictions.

Answer:

Explanation:
Solution (Step by Step) :
1. Create a Pod Security Policy:
- Define the security policy using a YAML file.
- The file will include specific rules for the policy.

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted-psp
spec:
  # RunAsUser and FSGroup are used for controlling user IDs and file system group IDs.
  runAsUser:
    rule: "RunAsAny"
  fsGroup:
    rule: "RunAsAny"
  #  The 'seLinux' field is used to define SELinux security context constraints.
  seLinux:
    rule: "RunAsAny"
  #  The 'supplementalGroups' field allows you to specify supplemental groups for the pod.
  supplementalGroups:
    rule: "RunAsAny"
  #  This section defines the allowed capabilities for the pod.
  capabilities:
    # 'drop' is used to drop capabilities that are not required.
    drop: ["SYS_CHROOT", "SYS_PTRACE", "SYS_ADMIN", "SYS_NICE", "SYS_RESOURCE", "SYS_TIME", "SYS_TTY_CONFIG", "SYS_AUDIT", "SYS_RAWIO", "SYS_MMAN", "SYS_SETFCAP", "SYS_DAC_READ_SEARCH"]
  #  The 'allowedCapabilities' field is used to define the allowed capabilities.
  allowedCapabilities: ["NET_BIND_SERVICE", "NET_BROADCAST"]
  #  This section defines the volumes that are allowed to be used by the pod.
  volumes:
  - 'configMap'
  - 'secret'
  - 'emptyDir'
  - 'persistentVolumeClaim'
  - 'hostPath'
  #  This section defines the allowed host paths that can be used by the pod.
  hostPath:
    # '' means all host paths are allowed.
    # ':/etc/hosts' means only the `/etc/hosts` path is allowed.
    # You can specify a specific directory or file using the `hostPath` field.
    # You can also specify multiple host paths using an array.
    rule: 'RunAsAny'
  #  This section defines the allowed services that can be used by the pod.
  services:
    #  The 'hostPort' field allows the pod to bind to host ports.
    hostPort:
      #  '' means all host ports are allowed.
      #  '8080' means only port 8080 is allowed.
      #  You can specify a specific port using the `hostPort` field.
      #  You can also specify multiple ports using an array.
      rule: 'RunAsAny'
  #  This section defines the allowed host namespaces that can be used by the pod.
  hostNetwork: false
  #  This section defines the allowed host processes that can be used by the pod.
  hostPID: false
  #  This section defines the allowed host IPC namespaces that can be used by the pod.
  hostIPC: false
  #  This section defines the allowed container security context.
  readOnlyRootFilesystem: false
  #  This section defines the allowed privilege escalation.
  allowPrivilegeEscalation: false
  #  This section defines the allowed security context constraints.
  securityContext:
    #  The 'runAsUser' field is used to specify the user ID that the pod will run as.
    #  The 'runAsGroup' field is used to specify the group ID that the pod will run as.
    #  The 'fsGroup' field is used to specify the file system group ID that the pod will run as.
    #  The 'seLinuxOptions' field is used to specify the SELinux security context options.
    #  The 'capabilities' field is used to specify the capabilities that the pod will have.
    #  The 'readOnlyRootFilesystem' field is used to specify whether the pod's root filesystem will be read-only.
    #  The 'allowPrivilegeEscalation' field is used to specify whether the pod can escalate privileges.
    #  The 'privileged' field is used to specify whether the pod will have privileged access.
    rule: 'RunAsAny'
```

2. Apply the Pod Security Policy: - After creating the policy, you can apply it to the cluster: bash kubectl apply -f restricted-psp.yaml 3. Use the Pod Security Policy: - You can enforce the policy on a pod using the 'securityContext' field in the pod's YAML file. - For example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      securityContext:
        #  The 'securityContext' field is used to specify the security context for the pod.
        #  The 'pspName' field is used to specify the Pod Security Policy that the pod will use.
        #  You can use the 'pspName' field to enforce the security context for the pod.
        pspName: restricted-psp
      containers:
      - name: nginx
        image: example/nginx:latest
        imagePullPolicy: Always
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 1
      maxSurge: 0
```


4. Enforce the Policy: - The PSP will enforce the specified restrictions on pods that are created using the deployment configuration.

NEW QUESTION # 47

.....

As is known to us, the high pass rate is a reflection of the high quality of CKS study torrent. The more people passed their exam, the better the study materials are. There are more than 98 percent that passed their exam, and these people both used our CKS test torrent. There is no doubt that our Certified Kubernetes Security Specialist (CKS) guide torrent has a higher pass rate than other study materials. We deeply know that the high pass rate is so important for all people, so we have been trying our best to improve our pass rate all the time. Now our pass rate has reached 99 percent. If you choose our CKS study torrent as your study tool and learn it carefully, you will find that it will be very soon for you to get the Certified Kubernetes Security Specialist (CKS) certification in a short time. Do not hesitate and buy our CKS test torrent, it will be very helpful for you.

- New CKS Test Practice □ CKS Cert Guide □ Valid CKS Test Camp □ Enter ⇛ www.pdfvce.com ⇚ and search for ➥ CKS □ to download for free □Valid Real CKS Exam
- CKS Reliable Test Voucher □ Braindump CKS Pdf □ Valid Real CKS Exam □ Easily obtain free download of （ CKS ） by searching on ▹ www.torrentvalid.com ◃ □CKS Clearer Explanation
- www.stes.tyc.edu.tw, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, cou.alnoor.edu.iq, www.wcs.edu.eu, faceliftery.blogs-service.com, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.stes.tyc.edu.tw, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, cou.alnoor.edu.iq, Disposable vapes