Lesson 08 Demo 02

Implementing a Pod Security Policy

Objective: To implement a Pod Security Policy to enhance container security

Tools required: kubeadm, kubectl, kubelet, and containerd

Prerequisites: A Kubernetes cluster (refer to Demo 01 from Lesson 01 for setting up a cluster)

Steps to be followed:

1. Create a Pod Security Policy
2. Create a role and role binding for PSP
3. Test the Pod Security Policy
4. Delete the resources

Step 1: Create a Pod Security Policy

1.1 Open the terminal and run the following command to create a PSP YAML file:
nano restricted-psp.yaml

```
labsuser@ip-172-31-27-233:~$ nano restricted-psp.yaml
```

1.2 Add the following code to the file:

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted-psp
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'projected'
```

```
  - 'secret'
  - 'downwardAPI'
  - 'persistentVolumeClaim'
hostNetwork: false
hostIPC: false
hostPID: false
runAsUser:
  rule: 'MustRunAsNonRoot'
seLinux:
  rule: 'RunAsAny'
supplementalGroups:
  rule: 'MustRunAs'
  ranges:
    - min: 1
      max: 65535
fsGroup:
  rule: 'MustRunAs'
  ranges:
    - min: 1
      max: 65535
```

```
  GNU nano 6.2                                                restricted-psp.yaml *
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: restricted-psp
spec:
  privileged: false
  allowPrivilegeEscalation: false
  requiredDropCapabilities:
    - ALL
  volumes:
    - 'configMap'
    - 'emptyDir'
    - 'projected'
    - 'secret'
    - 'downwardAPI'
    - 'persistentVolumeClaim'
  hostNetwork: false
  hostIPC: false
  hostPID: false
  runAsUser:
    rule: 'MustRunAsNonRoot'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
  fsGroup:
    rule: 'MustRunAs'
    ranges:
      - min: 1
        max: 65535
```

Note: Save it by pressing ctrl+s and exit by pressing ctrl+x

1.3 After saving the file, apply the Pod security policy to your Kubernetes cluster using the following command:
kubectl apply -f restricted-psp.yaml

```
labsuser@ip-172-31-27-233:~$ nano restricted-psp.yaml
labsuser@ip-172-31-27-233:~$ kubectl apply -f restricted-psp.yaml
```

Step 2: Create a Pod Security Policy

2.1 Create a role binding YAML file (**psp-rolebinding.yaml**) with the following command:
nano psp-rolebinding.yaml

```
labsuser@ip-172-31-27-233:~$ nano psp-rolebinding.yaml
```

2.2 Add the following code to the **config.yaml** file:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: psp-rolebinding
  namespace: default
subjects:
- kind: ServiceAccount
  name: default
  namespace: default
roleRef:
  kind: Role
  name: psp-role
  apiGroup: rbac.authorization.k8s.io
```

```
  GNU nano 6.2                                                    psp-rolebinding.yaml *
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: psp-rolebinding
  namespace: default
subjects:
- kind: ServiceAccount
  name: default
  namespace: default
roleRef:
  kind: Role
  name: psp-role
  apiGroup: rbac.authorization.k8s.io
```

2.3 Apply the role binding by running the following command:
kubectl apply -f psp-rolebinding.yaml

```
labsuser@ip-172-31-27-233:~$ nano psp-rolebinding.yaml
labsuser@ip-172-31-27-233:~$ kubectl apply -f psp-rolebinding.yaml
```

Step 3: Test the Pod Security Policy

3.1 Create a test pod YAML file (**test-pod.yaml**) using the command below:
nano test-pod.yml

```
labsuser@ip-172-31-27-233:~$ nano test-pod.yml
```

3.2 Add the following code to the file:

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pod
spec:
  containers:
  - name: nginx
    image: nginx
    securityContext:
      runAsUser: 1000
      runAsNonRoot: true
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: test-pod
spec:
  containers:
  - name: nginx
    image: nginx
    securityContext:
      runAsUser: 1000
      runAsNonRoot: true
```

3.3 Save the YAML file and apply it to your Kubernetes cluster using the following command:
kubectl apply -f test-pod.yaml

```
labsuser@ip-172-31-27-233:~$ nano test-pod.yml
labsuser@ip-172-31-27-233:~$ kubectl apply -f test-pod.yaml
```

Step 4: Delete the resources

4.1 Delete the test pod using the following command:
kubectl delete -f test-pod.yaml

```
labsuser@ip-172-31-27-233:~$ kubectl delete -f test-pod.yaml
```

> **Note**: The test pod is no longer needed after validating the Pod Security Policy. Deleting it frees up resources and ensures that the cluster does not contain unnecessary pods.

By following these steps, you have successfully implemented a Pod Security Policy to enhance container security within your Kubernetes environment.