Lesson 07 Demo 04

Creating and Using Secrets in a Volume

Objective: To create a Kubernetes secret and mount it as a volume inside a pod for enhancing security in the Kubernetes environment

Tools required: kubeadm, kubectl, kubelet, and containerd

Prerequisites: A Kubernetes cluster should already be set up (refer to the steps in Lesson 02, Demo 01 for guidance).

Steps to be followed:

1. Create a Kubernetes secret
2. Create a pod that uses the secret as a volume

Step 1: Create a Kubernetes secret

1.1 On the master node, use the following command to create a Kubernetes secret named **mysecret**:
kubectl create secret generic mysecret --from-literal='dbpass'='simplilearn'

```
labsuser@master:~$ kubectl get nodes
NAME                      STATUS   ROLES           AGE     VERSION
master.example.com        Ready    control-plane   2d22h   v1.28.2
worker-node-1.example.com Ready    <none>          2d22h   v1.28.2
worker-node-2.example.com Ready    <none>          2d22h   v1.28.2
labsuser@master:~$ kubectl create secret generic mysecret --from-literal='dbpass'='simplilearn'
secret/mysecret created
labsuser@master:~$
```

1.2 To view the created secret, use the following commands:

kubectl get secrets mysecret -o yaml

kubectl get secrets

```
labsuser@master:~$ kubectl get secrets mysecret -o yaml
apiVersion: v1
data:
  dbpass: c2ltcGxpbGVhcm4=
kind: Secret
metadata:
  creationTimestamp: "2023-10-20T09:40:37Z"
  name: mysecret
  namespace: default
  resourceVersion: "16610"
  uid: 9250107c-ded1-484c-b424-1ce8811a0eed
type: Opaque
labsuser@master:~$ kubectl get secrets
NAME       TYPE     DATA   AGE
mysecret   Opaque   1      48s
```

1.3 View detailed information about a secret stored in Kubernetes by using the following command:

kubectl describe secret mysecret

```
labsuser@master:~$ kubectl get secrets
NAME       TYPE     DATA   AGE
mysecret   Opaque   1      21s
labsuser@master:~$ kubectl describe secret mysecret
Name:         mysecret
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
dbpass:  11 bytes
labsuser@master:~$
```

Note: The secret stored in Kubernetes is encrypted in a human-readable format.

Step 2: Create a pod that uses the secret as a volume

2.1 Create a new YAML configuration file for the pod by running the following command:
nano secret-volume.yaml

```
labsuser@master:~$ kubectl get secrets
NAME       TYPE     DATA   AGE
mysecret   Opaque   1      21s
labsuser@master:~$ kubectl describe secret mysecret
Name:         mysecret
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
dbpass:  11 bytes
labsuser@master:~$ nano secret-volume.yaml
```

2.2 Add the following code to the **secret-volume.yaml** file:

```
apiVersion: v1
kind: Pod
metadata:
  name: secret-pod
spec:
  containers:
    - name: security-container
      image: nginx
      volumeMounts:
      - name: secret-volume
        mountPath: /etc/secret-volume
  volumes:
    - name: secret-volume
      secret:
        secretName: mysecret
```

```
 GNU nano 6.2                                  secret-volume.yaml
apiVersion: v1
kind: Pod
metadata:
  name: secret-pod
spec:
  containers:
    - name: security-container
      image: nginx
      volumeMounts:
      - name: secret-volume
        mountPath: /etc/secret-volume
  volumes:
    - name: secret-volume
      secret:
        secretName: mysecret
^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo         M-A Set Mark     M-] To Bracket   M-Q Previous
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo         M-6 Copy         ^Q Where Was     M-W Next
```

2.3 Run the following command to create a pod:
kubectl apply -f secret-volume.yaml

```
labsuser@master:~$ nano secret-volume.yaml
labsuser@master:~$ kubectl apply -f secret-volume.yaml
pod/secret-pod created
labsuser@master:~$
```

2.4 View the pod using the following command:
kubectl get pods

```
labsuser@master:~$ nano secret-volume.yaml
labsuser@master:~$ kubectl apply -f secret-volume.yaml
pod/secret-pod created
labsuser@master:~$ kubectl get pods
NAME                          READY   STATUS              RESTARTS   AGE
myhttpd-5bd4687fff-c65mk      0/1     ContainerCreating   0          40m
secret-pod                    1/1     Running             0          118s
labsuser@master:~$
```

2.5 To verify that the secret is properly mounted as a volume, access the pod by starting a shell session inside it with the following command:
kubectl exec -it secret-pod -- /bin/bash

```
labsuser@master:~$ kubectl get pods
NAME           READY   STATUS    RESTARTS       AGE
my-nginx-pod   1/1     Running   1 (113m ago)   2d22h
secret-pod     1/1     Running   0              53s
labsuser@master:~$ kubectl exec -it secret-pod -- /bin/bash
root@secret-pod:/#
```

2.6 Inside the pod, navigate to the directory **secret-volume** and view the secret data using the following command:

cd /etc/secret-volume

```
labsuser@master:~$ kubectl exec -it secret-pod -- /bin/bash
root@secret-pod:/# ls
bin   dev                 docker-entrypoint.sh  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint.d  etc                   lib   lib64  media   opt  root  sbin  sys  usr
root@secret-pod:/# cd /etc/secret-volume
root@secret-pod:/etc/secret-volume# ls
dbpass
```

2.7 View the content of the **dbpass** file using the following command:

cat dbpass

```
root@secret-pod:/etc/secret-volume# cat dbpass
simplilearnroot@secret-pod:/etc/secret-volume#
root@secret-pod:/etc/secret-volume#
```

This command should display the decrypted secret value, which is **Simplilearn** since that is the value provided when you created the secret.

By following these steps, you have successfully created a Kubernetes secret and mounted it as a volume inside a pod. This helps in enhancing the security by securely storing and providing access to sensitive information or configuration data within a Kubernetes cluster.