2024

Lab 6 - Conserver les données Volumes k8s

COMPTRE RENDU

YOUSSEF BEN GHORBEL

Table des matières

Introduction

Les données des Pods sont volatiles. À chaque destruction d'un Pod, toutes les données générées sont perdues. Pour résoudre ce problème, nous introduisons le concept de Volume, qui permet de conserver les données même si le Pod est recréé.

Objectifs

- Créer un Volume local de type hostPath
- Créer un Volume local de type emptyDir
- Créer un Volume distant de type NFS

Étapes à suivre

1. Supprimer le Namespace précédent :

```
namespace "mynamespaceexercice4" deleted
```

2. Créer un nouveau Namespace :

- Créer un fichier appelé mynamespaceexercice5.yaml avec le contenu suivant :

```
[root@localhost ~]# mkdir  exercice5-volumes
[root@localhost ~]# cd exercice5-volumes/
[root@localhost exercice5-volumes]# touch mynamespaceexercice5.yaml
[root@localhost exercice5-volumes]# gedit mynamespaceexercice5.yaml
```

```
apiVersion: v1
kind: Namespace
metadata:

 name: mynamespaceexercice5
```

- Créez le Namespace dans le cluster :

```
kubectl apply -f exercice5-volumes/mynamespaceexercice5.yaml
```

```
[root@localhost exercice5-volumes]# ged
[root@localhost exercice5-volumes]# kub
namespace/mynamespaceexercice5 created
```

3. Créer un Deployment avec un Volume hostPath :

- Créez un fichier myhostpath.yaml

```
apiVersion: apps/v1 kind:
Deployment metadata:

  name: mydeploymentwithhostpath
spec:
```

```
  replicas: 3
  selector:

    matchLabels:

      app: mypodforhostpath
  template:

    metadata:
      labels:

        app: mypodforhostpath spec:

      containers:

      - name: mynginx image:
        nginx:latest ports:
        - containerPort: 80
        volumeMounts:
          - mountPath: /usr/share/nginx/html name:
            myhostpathvolume

      volumes:

        - name: myhostpathvolume
          hostPath:
            path: /myhostpath type:
            DirectoryOrCreate

---


kind: Service
apiVersion: v1
metadata:

  name: mypodforhostpathservice
spec:

  selector:

    app: mypodforhostpath
  type: NodePort

  ports:

    - protocol: TCP
      targetPort: 80
      port: 8080

      nodePort: 30001
```

- Appliquez la configuration :

```
kubectl apply -f exercice5-volumes/myhostpath.yaml -n
mynamespaceexercice5
```

4. Vérifiez le déploiement :

```
kubectl get Pods -n mynamespaceexercice5 -o wide
```

```
[root@localhost exercice5-volumes]# kubectl get Pods -n myn
NAME                                      READY   STATUS
mydeploymentwithhostpath-cd488d747-ncp7r  1/1     Running
mydeploymentwithhostpath-cd488d747-p6xh8  1/1     Running
mydeploymentwithhostpath-cd488d747-s8xhq  1/1     Running
[root@localhost exercice5-volumes]#
```

5. **Vérifiez-le dossier /myhostpath sur les nœuds :**

```
$ docker exec -it k3d-mycluster-server-0 ls /
```

```
bin  dev  etc  k3d  lib  myhostpath  output  proc  run  sbin  sys  tmp  usr  var
```

6. **Tester le Service NodePort :**

```
curl localhost:30001
```

```
[root@localhost exercice5-volumes]# curl localhost:30001
<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
<hr><center>nginx/1.27.3</center>
</body>
</html>
[root@localhost exercice5-volumes]#
```

7. **Ajoutez un fichier index.html dans le dossier /myhostpath du nœud maître :**

```
$ docker exec k3d-mycluster-server-0 sh -c "echo 'Bonjour depuis le noeud Master' > /myhostpath/index.html"
```

8. **Effectuez des requêtes pour obtenir le contenu :**

```
$ curl localhost:30001
```

```
[root@localhost exercice5-volumes]# curl localhost:30001
Bonjour depuis le noeud Master
[root@localhost exercice5-volumes]#
```

9. **Supprimer les objets Deployment et Service :**

```
kubectl delete service -n mynamespaceexercice5 mypodforhostpathservice

kubectl delete deployments.apps -n mynamespaceexercice5 mydeploymentwithhostpath
```

```
Bonjour depuis le noeud Master
[root@localhost exercice5-volumes]#  kubectl delete s
service "mypodforhostpathservice" deleted
[root@localhost exercice5-volumes]# kubectl delete de
deployment.apps "mydeploymentwithhostpath" deleted
[root@localhost exercice5-volumes]#
```

10. Créer un Deployment avec un Volume emptyDir :

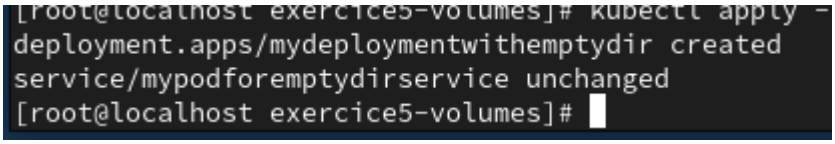
- Créer un fichier myemptydir.yaml :

```
apiVersion: apps/v1 kind:
Deployment metadata:

  name: mydeploymentwithemptydir
spec:

  replicas: 2
  selector:

    matchLabels:

      app: mypodforemptydir
  template:

    metadata:
      labels:

        app: mypodforemptydir spec:

      containers:

      - name: mynginx image:
        nginx:latest ports:
        - containerPort: 80
        volumeMounts:

          - mountPath: /usr/share/nginx/html name:
            myemptydirvolume

      initContainers:

      - name: mygit image:
        alpine/git args:
          - clone
          - --

          - https://github.com/cloudacademy/static-website-example
          - /data
        volumeMounts:
          - mountPath: /data name:
            myemptydirvolume
      volumes:

        - name: myemptydirvolume
          emptyDir:

            medium: Memory


---

kind: Service
apiVersion: v1
metadata:

  name: mypodforemptydirservice
spec:

  selector:

    app: mypodforemptydir
  type: NodePort
```

```
  ports:
    - protocol: TCP
      targetPort: 80
      port: 8080
      nodePort: 30001
```

- Appliquer la configuration :

```
kubectl apply -f exercice5-volumes/myemptydir.yaml -n
mynamespaceexercice5
```



11. Créer un Deployment avec un Volume NFS

- Créez un fichier mynfs.yaml

```
apiVersion: apps/v1 kind:
Deployment metadata:
  name: mydeploymentwithnfs
spec:
  replicas: 3
  selector:
    matchLabels:
      app: mypodfornfs
  template:
    metadata:
      labels:
        app: mypodfornfs spec:
      containers:
      - name: mynginx image:
        nginx:latest ports:
        - containerPort: 80
        volumeMounts:
          - mountPath: /usr/share/nginx/html name:
            mynfsvolume
      volumes:
      - name: mynfsvolume
        nfs:
          server: <IP_NFS-SERVER>
          path: "/nfs"
---
kind: Service
apiVersion: v1
metadata:
```

```
  name: mypodfornfsservice
spec:

  selector:

    app: mypodfornfs
  type: NodePort ports:

    - protocol: TCP
      targetPort: 80
      port: 8080

      nodePort: 30001
```

- Appliquer la configuration :

```
kubectl apply -f exercice5-volumes/mynfs.yaml -n
mynamespaceexercice5
```