Université de Picardie Jules Verne

Informatique – Master CCM

INSSET – Saint-Quentin

Conteneurs Applicatifs et Micro-Services M2

C. Drocourt

cyril.drocourt@u-picardie.fr

Cours 5.3 : Kubernetes – Les déploiements

V2023.1

Table des matières

1 - Introduction

Il est plus pratique de passer par un « Deployment » qui va :

- créer les Pods,
- gérer le nombre d'instances,
- gérer la scalabilité,
- gérer le ReplicaSet associé,

Pour obtenir le fichier YAML correspondant au Deployment d'un seul Pod lié à l'image « nginx » :

```
root@maitre:~# kubectl create deployment my-deploy --image nginx --dry-run -o yaml
```

2 - Fichier de déploiement

Le plus simple est de créer un fichier de description d'un déploiement :

```
root@maitre:~# cat monnginx.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: monnginx
spec:
  selector:
    matchLabels:
      run: monnginx
  replicas: 2
  template:
    metadata:
      labels:
        run: monnginx
    spec:
      containers:
      - name: monnginx
        image: nginx
        ports:
        - containerPort: 80
```

La description du fichier est la suivante :

- Le « kind » est maintenant un Deployment,
- Ce « Deployment » contient une section « metadata » et « spec »,
- La section « selector » indique comment le « deployment » va reconnaître les pods, ici, via le label « run:monnginx ».
- Le « template » va permettre de définir les Pods,
- Ce dernier possède également une section « metadata » et « spec »,

On peut maintenant déployer l'application :

```
root@maitre:~# kubectl create -f monnginx.yaml
deployment.apps/monnginx created
```

3 - Vérifications

On peut vérifier les déploiements en cours :

```
root@maitre:~# kubectl get deployments
NAME        DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
monnginx    2         2         2            2           52m
```

On peut obtenir plis d'informations :

```
root@maitre:~# kubectl get deployments -o wide
NAME        DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
monnginx    2         2         2            2           52m
```

On peut également obtenir des informations supplémentaires sur le déploiement avec la commande :

```
root@maitre:~# kubectl describe deployments monnginx
```

On vérifie les « pods » :

```
root@maitre:~# kubectl get pods
NAME                          READY   STATUS              RESTARTS
AGE
monnginx-6d88c97b4c-pw8wk     0/1       ContainerCreating     0
14s
monnginx-6d88c97b4c-xkg54     0/1       ContainerCreating     0
14s
```

Pour afficher les labels associés on peut utiliser l'option « --show-labels » :

```
root@maitre:~# kubectl get pods --show-labels
NAME                          ...  LABELS
monnginx-6d88c97b4c-pw8wk        ...       app=nginx,pod-template-
hash=d55b94fd
monnginx-6d88c97b4c-xkg54        ...       app=nginx,pod-template-
hash=d55b94fd
```

Il est possible de demander plus d'information, notamment les adresses IPs :

```
root@maitre:~# kubectl get pods -o wide
NAME                          READY  STATUS   REST  AGE  IP           NODE
NOMINATED
monnginx-6d88c97b4c-pw8wk  1/1     Running  0       40s  10.244.1.3  noeud1
<none>
monnginx-6d88c97b4c-xkg54  1/1     Running  0       40s  10.244.2.4  noeud2
<none>
```

On peut donc tester une requête sur nos « pods » :

```
root@maitre:~# curl -I 10.244.1.3
HTTP/1.1 200 OK
Server: nginx/1.15.6
Date: Thu, 15 Nov 2018 09:16:55 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 06 Nov 2018 13:32:09 GMT
Connection: keep-alive
ETag: "5be197d9-264"
Accept-Ranges: bytes
```

4 - Modification

4.1 - Scalabilité d'un Deployment

Il est possible de modifier le nombre de « pods » d'un « Deployment » sur la ligne de commande, en indiquant le fichier YAML d'origine :

```
root@maitre:~# kubectl scale --replicas=3 -f monnginx.yml
```

ou en indiquant le nom du « Deployment » correspondant :

```
root@maitre:~# kubectl scale deployment.apps/monnginx --replicas=3
```

Il est possible demander à Kubernetes de réaliser un « autoscale » basé par exemple sur le critère du CPU :

```
root@maitre:~# kubectl autoscale deployment monnginx --min=2 --max=6 --cpu-percent=80
```

Pour visualiser le HPA :

```
root@maitre:~# kubectl get hpa
NAME       REFERENCE              TARGETS         MINPODS   MAXPODS   REPLICAS
AGE
monnginx   Deployment/monnginx   <unknown>/80%   2         6         3
7m37s
```

On peut éditer la configuration de l'autoscale par la commande :

```
root@maitre:~# kubectl edit hpa monnginx
```

Pour supprimer complètement un autoscale :

```
root@maitre:~# kubectl delete hpa monnginx
```

4.2 - Suppression d'un Deployment

Pour supprimer un deployment :

```
root@maitre:~# kubectl delete deployment.apps/monnginx
```

ou

```
root@maitre:~# kubectl delete deployment monnginx
```

4.3 - Kompose

Il existe un outil nommé « Kompose » permettant d'utiliser les fichiers « docker-compose », pour l'installer :

```
root@maitre:~# curl -L https://github.com/kubernetes/kompose/releases/download/v1.16.0/kompose-linux-amd64 -o kompose
root@maitre:~# chmod +x kompose
```

Pour travailler avec, il est possible :

- D'activer directement les « deployment » à partir de ce fichier,
- De convertir ce fichier avant, le plus simple est alors de créer un répertoire :

```
root@maitre:~# mkdir app1
root@maitre:~# mv docker-compose.yml app1
root@maitre:~# cd app1/
root@maitre:~# ../kompose convert
```

Le programme a généré plusieurs fichiers « yaml » qu'il faut modifier :

- les versions doivent être passées en v1,
- les labels peuvent être simplifiés,

Il est ensuite possible de l'appliquer à l'aide de la commande :

```
root@maitre:~# kubectl apply -f app1-madb-persistentvolumeclaim.yaml,db-deployment.yaml,wordpress-deployment.yaml,wordpress-service.yaml
```