

LAB 3 : Manipulation des Pods

Conditions préalables

- Cluster Kubernetes fonctionnel

Exercice 1 : Premier Pod

Dans cet exercice, vous allez créer une spécification pour votre premier Pod et le lancer.

1. Création de la spécification

Créez un fichier yaml **www_pod.yaml** définissant un Pod ayant les propriétés suivantes:

- nom du Pod: **www**
- image du container: **nginx:1.16-alpine**
- nom du container: **nginx**

2. Lancement du Pod

Lancez le Pod à l'aide de **kubectl**

3. Vérification

Listez les Pods lancés et assurez-vous que le Pod **www** apparait bien dans cette liste.

4. Détails du Pod

Observez les détails du Pod à l'aide de **kubectl** et retrouvez l'information de l'image utilisée par le container **nginx**.

5. Lancement d'un shell dans le container **nginx**

Lancez un shell dans le container **nginx** et vérifiez la version du binaire **nginx**.

Note: utilisez la commande **nginx -v** .

6. Suppression du Pod

Toujours à l'aide de **kubectl**, supprimez le Pod.

Exercice 2 : l'application wordpress

Dans cet exercice vous allez créer un Pod contenant 2 containers permettant de lancer une application wordpress.

1. Création de la spécification

Créez un fichier yaml `wordpress_pod.yaml` définissant un Pod ayant les propriétés suivantes:

- nom du Pod: wp
- un premier container:
 - nommé wordpress
 - basé sur l'image `wordpress:4.9-apache`
 - définissant la variable d'environnement `WORDPRESS_DB_PASSWORD` avec pour valeur `mysqlpwd` (cf note)
 - définissant la variable d'environnement `WORDPRESS_DB_HOST` avec pour valeur `127.0.0.1` (cf note)
- un second container:
 - nommé mysql
 - basé sur l'image `mysql:5.7`
 - définissant la variable d'environnement `MYSQL_ROOT_PASSWORD` avec pour valeur `mysqlpwd` (cf note)

Note: chaque container peut définir une clé env, celui contenant une liste de paires name/value

2. Lancement du Pod

Lancez le Pod à l'aide de `kubectl`

3. Vérification du status du Pod

Utilisez `kubectl` pour vérifier l'état du Pod.

Au bout de quelques secondes, il devrait être dans l'état Running (le temps que les images des containers soient téléchargées depuis le DockerHub).

4. Accès à l'application

- Forwardez le port 80 du container wordpress sur le port 8080 de la machine hôte.
- Lancez un navigateur sur `http://localhost:8080`
- En ouvrant un navigateur sur l'URL indiquée, vous obtiendrez l'interface web de setup de Wordpress.

5. Suppression du Pod

A l'aide de `kubectl` supprimez le Pod wp.

Exercice 3 : contrainte de déploiement

Dans cet exercice, vous allez ajouter une contrainte afin de déployer un Pod sur un node en particulier.

1. Définition d'un Pod Mysql

La spécification suivante définit un Pod contenant un container mysql.

```
apiVersion: v1
kind: Pod
metadata:
  name: db
spec:
  containers:
  - image: mysql:5.7
    name: mysql
    env:
    - name: MYSQL_ROOT_PASSWORD
      value: mysqlpwd
    volumeMounts:
    - name: data
      mountPath: /var/lib/mysql
  volumes:
  - name: data
    emptyDir: {}
```

2. Ajout d'une contrainte de déploiement

Pour des raisons de performances, nous souhaitons déployer ce Pod sur un node contenant un disk SSD. Nous allons alors ajouter cette contrainte de déploiement via la clé **affinity** dans la spécification du Pod.

3. Déploiement du Pod

- Créez le Pod mysql en vérifiant le noeud de déploiement.
- Vérifiez l'état de déploiement du Pod à l'aide de la commande `describe`. Qu'observez-vous ?

4. Ajout d'un label sur l'un des nodes du cluster

- Ajoutez un label `disktype` avec la valeur `SSD` sur le nœud `worker-node1`
- Vérifiez l'état de déploiement du Pod à l'aide de la commande `describe`. Qu'observez-vous ?