

TD04 | K8s - Manipulation kubectl

⚠ Utilisation de ChatGPT/OpenAI tout autre IA = 0/20 sur les TP2/3

Objectifs

L'objectif de ce TD est de revoir toutes les commandes utiles que vous utiliserez en entreprise en tant que développeur pour utiliser/déployer dans Kubernetes.



Tip

Utiliser la commande `kubectl get/create/... --help` quand vous êtes bloqué !

Quelques liens utiles :

- [kubectl Cheat Sheet](#)
- [Get a Shell to a Running Container](#)
- [Configure Access to Multiple Clusters](#)
- [Accessing Clusters](#)
- [Use Port Forwarding to Access Applications in a Cluster](#)

Prérequis

- ✔ Cluster Kubernetes déployé avec Kind
- ✔ Kubectl pour effectuer toutes les commandes

Concepts de base

1. Créer un `namespace` appelé 'monnamespace' et un pod avec une image nginx appelé nginx sur ce namespace.
2. Créer le pod qui vient d'être décrit en utilisant YAML (`-o yaml`)
3. Créez un pod busybox (en utilisant la commande kubectl) qui exécute la commande "env". Exécutez-la et voyez la sortie
4. Obtenir le YAML pour un nouveau namespace appelé 'monns' sans le créer
5. Créer le YAML pour un nouveau `ResourceQuota` appelé 'monrq' avec des limites strictes de 1 CPU, 1G de mémoire et 2 pods sans le créer.
6. Afficher les pods sur tous les namespaces
7. Créer un pod avec l'image `nginx` appelé `nginx` et exposer le trafic sur le port 80
8. Changer l'image du pod en `nginx:1.7.1`. Observez que le conteneur sera redémarré dès que l'image sera retirée.
9. Obtenir l'adresse IP du pod nginx créé à l'étape précédente, utiliser une image busybox temporaire pour `wget` son '/'.
10. Obtenir des informations sur le pod nginx, y compris des détails sur les problèmes potentiels (par exemple, le pod n'a pas démarré).

Pods Multi-Containers

1. Créez un Pod avec deux conteneurs, tous deux avec l'image busybox et la commande `echo hello ; sleep 3600`. Connectez-vous au deuxième conteneur et exécutez 'ls'

Conception de pods

Label et annotations

1. Créez 3 pods avec les noms nginx1,nginx2,nginx3. Tous les pods doivent avoir le label `app=v1`
2. Affichez tous les labels des pods.
3. Modifier les labels du pod `nginx2` pour qu'il soit `app=v2`
4. Obtenir le label "app" pour les pods (afficher une colonne avec les étiquettes APP)
5. Affichez uniquement les pods `app=v2`
6. Ajouter un nouveau label `tier=web` à tous les pods ayant des labels `app=v2` **ou** `app=v1`.
7. Ajouter une annotation `owner : marketing` à tous les pods ayant l'étiquette `app=v2`.

Placement du Pod

1. Créer un pod qui sera déployé sur un Noeud ayant le label `accelerator=nvidia-tesla-p100`
2. Créer un pod qui sera placé sur le nœud `control-plane`. Utiliser le `nodeSelector` et les `tolerations`

Déploiement

1. Créer un déploiement avec l'image `nginx:1.18.0`, appelé `nginx`, ayant 2 replica, définissant le port 80 comme le port que ce conteneur expose (ne pas créer de service pour ce déploiement).
2. Afficher le yaml de ce déploiement
3. Afficher le YAML de l'ensemble des `replica set` créé par ce déploiement
4. Vérifier l'état d'avancement du déploiement
5. Mettre à jour l'image nginx vers `nginx:1.19.8`
6. Vérifier l'historique des déploiements et confirmer que les replica sont correctes.
7. Annuler le dernier déploiement et vérifier que les nouveaux pods ont l'ancienne image (nginx:1.18.0)

Merci de votre attention
