Lab 1 : Installer un cluster K3d

Cette préparation d'environnement cible la mise en place d'un cluster Kubernetes à partir de l'outil K3d qui s'appuie sur la distribution légère K3s. K3d sera utilisé comme solution *DinD* => Docker in Docker. Cette approche *DinD* permet de déployer un cluster Kubernetes multi-nœuds directement sur votre poste développeur. Tous les nœuds (maître et de travail) sont encapsulés dans un conteneur Docker. L'avantage est de pouvoir profiter des performances des conteneurs (rapidité et occupation mémoire réduite) pour créer des nœuds.

Comme précisé en introduction, l'ensemble des expérimentations ont été testées depuis macOS et Linux. L'adaptation sous Windows n'est pas insurmontable, il faudra adapter certains scripts.

Il est important de signaler que cette préparation d'environnement ne peut être appliquée pour une mise en production. Elle est dédiée au poste du développeur qui souhaite s'assurer que les configurations fonctionnent correctement.

But

- Installer les outils de gestion kubectl et K9s
- Créer un cluster K3d

Étapes à suivre

L'outil de ligne de commande Kubernetes, **kubectl**, vous permet d'exécuter des commandes sur des clusters Kubernetes. Vous pouvez utiliser kubectl pour déployer des applications, inspecter et gérer les ressources du cluster et afficher les journaux.

On commencera par l'installer en premier pour qu'il soit automatiquement configuré lors de l'installation du cluster k3d.

Installation à l'aide du gestionnaire de packages

```
# cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
EOF
```

```
# yum install -y kubectl
```

Installation de k3d

K3d sera donc utilisé pour créer notre cluster Kubernetes. Il s'agit d'un outil en ligne de commande qui encapsule la création des nœuds dans des conteneurs Docker.

Ci-dessous sont données les instructions d'installation de K3d pour Linux et macOS.

Pour installer K3d :

```
$ wget -q -O - https://raw.githubusercontent.com/k3d-io/k3d/main/install.sh
| bash
```

- Pour s'assurer que K3d est correctement installé, exécuter les deux commandes suivantes :

```
$ k3d version
k3d version v5.4.9
k3s version v1.25.7-k3s1 (default)
```

Nous allons créer un cluster Kubernetes composé de trois nœuds dont un sera dédié au nœud maître et les deux autres seront dédiés aux nœuds de travail. Sous K3d, un nœud de travail est intitulé agent et un nœud maître est intitulé server.

- Créer un cluster Kubernetes via K3d qui s'appelera `mycluster` :

```
$ k3d cluster create mycluster --agents 2 --servers 1
```

Cette commande crée un cluster Kubernetes appelé `mycluster`. Il contient deux nœuds de travail (`--agents 2`) et un (1) nœud maître (`--servers 1`).

- Consulter les conteneurs Docker qui ont été créés :

```
$ docker ps
```

```
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE                            COMMAND                  CREATED          STATUS          PORTS                                  NAMES
366520e2e83a   ghcr.io/k3d-io/k3d-proxy:5.4.9   "/bin/sh -c nginx-pr…"   39 seconds ago   Up 28 seconds   80/tcp, 0.0.0.0:33907->6443/tcp   k3d-mycluster-serverlb
aee702c74efe   rancher/k3s:v1.25.7-k3s1         "/bin/k3d-entrypoint…"   39 seconds ago   Up 32 seconds                                     k3d-mycluster-agent-1
33ada18b2213   rancher/k3s:v1.25.7-k3s1         "/bin/k3d-entrypoint…"   39 seconds ago   Up 32 seconds                                     k3d-mycluster-agent-0
5d6817df3507   rancher/k3s:v1.25.7-k3s1         "/bin/k3d-entrypoint…"   40 seconds ago   Up 38 seconds                                     k3d-mycluster-server-0
[root@localhost ~]#
```

Les deux nœuds de travail sont encapsulés par les deux conteneurs nommés `k3d-mycluster-agent-0` et `k3d-mycluster-agent-1`, le nœud maître est encapsulé par un (1) conteneur nommé `k3d-mycluster-server-0`, un conteneur `k3d-mycluster-serverlb` qui sert d'équilibreur de charge (*LoadBalancer*) pour le cluster K8s et finalement un conteneur `k3d-mycluster-tools` qui fournit des outils spécifiques comme par exemple l'importation d'images Docker vers un cluster K3s.

Si vous installez kubectl après k3d

Afin que nous puissions accéder au cluster Kubernetes, nous devons récupérer un fichier d'accès qui contiendra des informations comme les autorisations pour les outils clients. Ce fichier d'accès permet de communiquer avec le composant *API Server* d'un cluster.

- Se placer à la racine du dossier du dépôt de ce tutoriel et exécuter la ligne de commande suivante pour récupérer ce fichier d'accès :

```
$ k3d kubeconfig get mycluster > k3s.yaml
```

Nous avons désormais un cluster Kubernetes, mais nous ne disposns pas encore des outils pour interagir avec celui-ci. Nous détaillons ci-après comment installer les outils de gestion **kubectl** et K9s sur votre poste de développeur. Leurs utilisations seront détaillées dans l'exercice suivant.

kubectl et K9s sont des outils qui communiquent avec le composant *API Server* et nécessitent d'accéder au fichier *k3s.yaml* obtenu précédemment.

Pour installer **kubectl** procédez comme l'étape indiquée précédemment

```
$ export KUBECONFIG=$PWD/k3s.yaml
$ kubectl version --client
```

Pour tester si **kubectl** est correctement installé :

```
$ kubectl top nodes
NAME                       CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
k3d-mycluster-agent-0      26m          1%     102Mi           6%
k3d-mycluster-agent-1      36m          1%     176Mi           11%
k3d-mycluster-server-0     82m          4%     408Mi           27%
```

La première ligne de commande permet d'indiquer à **kubectl** où se trouve le fichier d'accès au cluster Kubernetes. Cette commande n'est à réaliser qu'une seule fois à l'ouvertue de votre terminal. Le seconde ligne de commande permet d'obtenir des informations sur les ressources utilisées par des objets gérés par Kubernetes (ici l'objet est un nœud).

Installation K9s (Facultatif)

K9s est un gestionnaire de cluster Kubernetes qui a la particularité de fonctionner dans la console. L'interface utilisateur est très simpliste, mais permet de retourner en continu l'état du cluster.

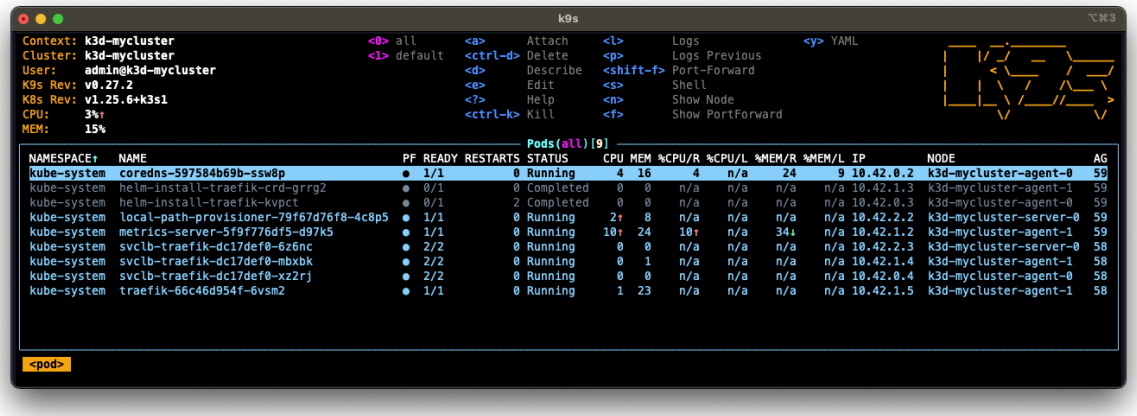
Linux : pour installer **K9s** :

```
$ wget https://github.com/derailed/k9s/releases/download/v0.25.15/k9s_Linux_x86_64.tar.gz
$ tar xzf k9s_Linux_x86_64.tar.gz
```

- Pour tester si **K9s** est correctement installé, depuis un autre terminal :

```
$ export KUBECONFIG=./k3s.yaml
$ k9s
```

Vous devriez obtenir le même résultat que sur la figure ci-dessous.



Bilan de l'exercice

À cette étape, vous disposez :

- d'un cluster Kubernetes avec trois nœuds dont un pour le maître et deux autres pour les nœuds de travail ;
- de deux outils de contrôle pour notre cluster Kubernetes.