

Mise en situation professionnelle

	Cours : Kubernetes
Sujet : Premier Pod	Numéro : 4
	Version : 1.1

Objectifs :

Préparer votre poste de travail

Prérequis :

aucun

Principales tâches à réaliser :

- 4 Premier Pod.....2
 - 4.1 Partie 1 : lancement de pods.....2
 - a Démarrage d’un POD influxdb.....2
 - b Démarrage un POD grafana.....2
 - c Injecter des données.....3
 - d Bonus.....3
 - e Nettoyage.....3
 - 4.2 Solution.....4
 - a 01-PodInflux.yaml.....5
 - b 02-PodTelegraf.yaml.....6
 - c 03-PodGrafana.yaml.....6
 - d Mettre à jour le dépôt gitea.....7
 - e Accéder à Grafana.....7

4 Premier Pod

4.1 Partie 1 : lancement de pods

Nous allons lancer des pods grâce à la description de PODs en YAML. Ce premier exemple ne fait appel qu'à des fonctionnalités docker. Dans un second temps nous mettrons en place un projet plus complexe faisant communiquer des POD entre eux.

InfluxDB est une base de données basée sur le temps. Elle est parfaitement adaptée pour stocker des données de monitorings.

NB :

- utiliser le fichier CheatSheet.odt mis à disposition par le formateur

a Démarrage d'un POD influxdb

Récupérer et utiliser de l'image registry.formation.local:5000/easylinux/influxdb:1.7 ou [easylinux/kubernetes:telegraf](https://registry.formation.local:5000/easylinux/kubernetes:telegraf)

Utiliser les variables d'environnement pour

- ✓ Configurer le nom de la base (**INFLUXDB_DB**)
- ✓ Configurer le nom d'utilisateur (**INFLUXDB_USER**)
- ✓ Configurer le mot de passe (**INFLUXDB_PASSWORD**)

Consulter la documentation du container influxdb.

<http://registry.formation.local/Docs/influx.html>

Grafana est un projet web pouvant se connecté à un ensemble de base de données, tel influxdb. L'outil permet de générer de nombreux tableau de bord facilement.

b Démarrage un POD grafana

Utiliser l'image registry.formation.local:5000/easylinux/grafana ou [easylinux/kubernetes:grafana](https://registry.formation.local:5000/easylinux/kubernetes:grafana)

Nous ne pouvons pour l'instant pas encore accéder à ce composant à l'extérieur de notre cluster. Nous verrons comment faire dans le prochain chapitre.

c Injecter des données

Il nous reste à configurer un pooler pour envoyer des données de monitoring dans notre base influxdb. Pour la configuration de celui-ci il conviendra de le configurer avec les variables d'environnement pour lui permettre d'envoyer ses données dans la base InfluxDB. Le port InfluxDB par défaut est le 8086. Pour obtenir l'adresse IP du POD InfluxDB vous pouvez exécuter la commande :

```
# kubectl get pods -o wide
```

Démarrer un POD météo qui envoie la température à Nantes sur influxdb

Utiliser l'image [registry.formation.local:5000/easylinux/telegraf](#) ou [easylinux/kubernetes:telegraf](#)

Utiliser les variables d'environnement pour

- ✓ Configurer l'IP de la base de données (**INFLUXDB_HOST**, **INFLUXDB_PORT**)
- ✓ Configurer le nom de la base (**INFLUXDB_NAME**)
- ✓ Configurer le nom d'utilisateur (**INFLUXDB_USER**)
- ✓ Configurer le mot de passe (**INFLUXDB_PASSWORD**)

d Bonus

Si vous en êtes arrivé là c'est bien ! Modifier le pod grafana pour exposer le port 3000.

Vous pouvez utiliser la commande :

```
$ kubectl expose pod <nom du pod grafana>
```

pour créer un service lié au POD

e Nettoyage

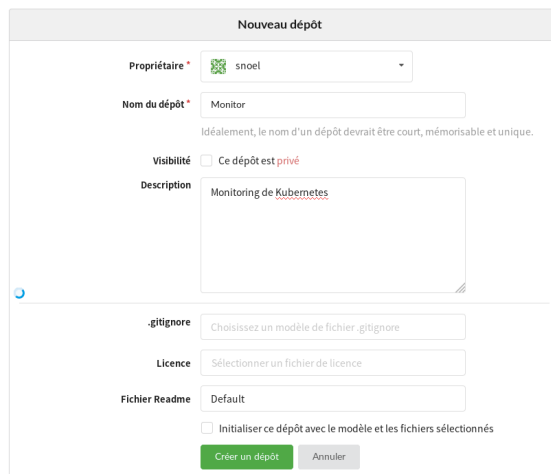
Lister puis supprimer les pods, vérifier que le service exposé est éteint (sinon le pod redémarre)

```
# kubectl get service
NAME          TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
kubernetes    ClusterIP   10.96.0.1     <none>         443/TCP    28h
# kubectl get pod
NAME          READY   STATUS    RESTARTS   AGE
influxdb      1/1     Running   0          131m
telegraf      1/1     Running   0          36m
# kubectl delete pod influxdb
pod "influxdb" deleted
# kubectl delete pod telegraf
pod "telegraf" deleted
# kubectl get pod
No resources found.
```

4.2 Solution

Nous allons créer un répertoire pour stocker notre application.

Commencer par créer un projet git (afin d'avoir l'historique)



Créer un répertoire qui recevra la définition de notre application,

```
mkdir Monitor
```

Dans ce répertoire, initialiser le projet

```
vi README.md
git init
git add README.md
git remote add origin http://registry.formation.local:3000/snoel/Monitor.git
```

Créer les fichiers Pod

a 01-PodInflux.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-influx
  labels:
    app: monitor-influx
spec:
  containers:
  - name: influxdb
    image: registry.formation.local:5000/easylinux/influxdb:1.7
    env:
      - name: INFLUXDB_DB
        value: "db0"
      - name: INFLUXDB_USER
        value: "telegraf"
      - name: INFLUXDB_PASSWORD
        value: "Secr3t"
    imagePullPolicy: IfNotPresent
  ports:
  - name : http
    containerPort : 8083
```

Lancer le pod

```
$ kubectl apply -f 01-PodInfluxDb.yaml
```

b 02-PodTelegraf.yaml

Lire l'ip assignée au Pod InfluxDB

```
$ kubectl get pod -o wide
NAME          READY   STATUS    RESTARTS   AGE   IP           NODE       NOMINATED NODE
READINESS GATES
pod-influx    1/1     Running   0           10s   172.17.0.7   minikube   <none>
<none>
```

Créer le fichier

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-telegraf
  labels:
    app: monitor-telegraf
spec:
  containers:
  - name: telegraf
    image: registry.formation.local:5000/easylinux/telegraf
    imagePullPolicy: IfNotPresent
    env:
      - name: INFLUXDB_DB
        value: "db0"
      - name: INFLUXDB_USER
        value: "telegraf"
      - name: INFLUXDB_PASSWORD
        value: "Secr3t"
      - name: INFLUXDB_HOST
        value: "172.17.0.7"
      - name: INFLUXDB_PORT
        value: "8083"
```

Lancer le pod

```
$ kubectl apply -f 02-PodTelegraf.yaml
```

c 03-PodGrafana.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-grafana
  labels:
    app: monitor-grafana
spec:
  containers:
  - name: grafana
    image: registry.formation.local:5000/easylinux/grafana
    imagePullPolicy: IfNotPresent
    ports:
      - name : http
        containerPort : 3000
```

Lancer le pod

```
$ kubectl apply -f 03-PodGrafana.yaml
```

d Mettre à jour le dépôt gitea

```
$ git add *
snoel@Asus-Serge:~/Formation/Kubernetes/Exercice/Monitor$ git commit -am "Création des pods"
[master (commit racine) ceb3b84] Création des pods
4 files changed, 51 insertions(+)
create mode 100644 01-PodInfluxDb.yml
create mode 100644 02-PodTelegraf.yml
create mode 100644 03-PodGrafana.yml
create mode 100644 README.md
$ git push origin master
```

e Accéder à Grafana

Attendre que les pods soient actifs

```
$ kubectl get pod
NAME          READY   STATUS    RESTARTS   AGE
pod-grafana   1/1     Running   0           3m49s
pod-influx    1/1     Running   0           6m34s
pod-telegraf  1/1     Running   0           4m
```

Vérifier que Grafana est à l'écoute

```
$ kubectl get pods pod-grafana --template='{{(index (index .spec.containers 0).ports 0).containerPort}}{{"\n"}}'
3000
```

Accéder au pod

```
$ kubectl port-forward pod/pod-grafana 3000:3000
Forwarding from 127.0.0.1:3000 -> 3000
```

Il faudra créer un tunnel ssh ou tester en local

