

# Technologie des conteneurs

POLYCARPE Julien

BENGANA Yassine

## TP3 : Découverte de Kubernetes

Dans ce TP vous allez installer Kind, une distribution de Kubernetes pour le développement, vous familiariser avec kubectl, la CLI de Kubernetes.

**Q1 / Installez kubectl sur votre machine [en suivant la documentation](#).**

**Vérifiez l'installation en exécutant la commande `kubectl version --client`**

```
osboxes@osboxes:~$ kubectl version --client
Client Version: version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.0", GitCommit:"ab69524f795c4209
4a6630298ff53f3c3ebab7f4", GitTreeState:"clean", BuildDate:"2021-12-07T18:16:20Z", GoVersion:"go1.17.
3", Compiler:"gc", Platform:"linux/amd64"}
```

**Q2 / Afin d'utiliser facilement Kubernetes sur votre machine, vous allez installer kind (Kubernetes IN Docker) [en suivant là encore la documentation](#).**

**Utilisez la commande `kind create cluster` pour créer votre premier cluster.**

```
root@osboxes:/home/osboxes/tp-containers/kubernetes# kind create cluster
Creating cluster "kind" ...
 ✓ Ensuring node image (kindest/node:v1.23.4) 📁
 ✓ Preparing nodes 📦
 ✓ Writing configuration 📄
 ✓ Starting control-plane 🏠
 ✓ Installing CNI 🌐
 ✓ Installing StorageClass 💾
Set kubectl context to "kind-kind"
You can now use your cluster with:

kubectl cluster-info --context kind-kind

Have a question, bug, or feature request? Let us know! https://kind.sigs.k8s.io/#community 😊
root@osboxes:/home/osboxes/tp-containers/kubernetes#
```

**Q3 / Commencez par lister les Namespaces du cluster, quelle commande kubectl faut-il exécuter ?**

**Listez ensuite les éléments suivants :**

- Les pods dans le namespace default
- Les pods dans le namespace kube-system

**Quel est selon vous le rôle du namespace kube-system ?**

# Technologie des conteneurs

```
root@osboxes:/home/osboxes/tp-containers/kubernetes# kubectl get namespace
NAME                STATUS    AGE
default              Active    36s
kube-node-lease      Active    38s
kube-public           Active    38s
kube-system           Active    38s
local-path-storage   Active    30s
root@osboxes:/home/osboxes/tp-containers/kubernetes# kubectl get pods --namespace=default
No resources found in default namespace.
root@osboxes:/home/osboxes/tp-containers/kubernetes# kubectl get pods --namespace=kube-system
NAME                                READY   STATUS    RESTARTS   AGE
coredns-64897985d-4759f             1/1     Running   0           2m40s
coredns-64897985d-xtvpb             1/1     Running   0           2m40s
etcd-kind-control-plane             1/1     Running   0           2m55s
kindnet-jtvp7                       1/1     Running   0           2m40s
kube-apiserver-kind-control-plane    1/1     Running   0           2m51s
kube-controller-manager-kind-control-plane 1/1     Running   0           2m51s
kube-proxy-rvmgq                    1/1     Running   0           2m40s
kube-scheduler-kind-control-plane    1/1     Running   0           2m51s
root@osboxes:/home/osboxes/tp-containers/kubernetes#
```

Le namespace kube-system est utilisé par le moteur kubernetes a des fins de fonctionnement du cluster.

**Q4 / Écrivez un manifest (fichier .yaml) pour démarrer le pod suivant :**

- 1 seul conteneur avec l'image *gcr.io/kuar-demo/kuard-amd64:blue*
- Le conteneur écoute sur le port 8080 en http

Pour écrire les manifests, il y a mieux que Stackoverflow : [la documentation de référence de l'API Kubernetes](#).

```
! pod.yaml u x
kubernetes > kuard > ! pod.yaml > apiVersion
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: kuard
5  spec:
6    containers:
7      - name: kuard
8        image: gcr.io/kuar-demo/kuard-amd64:blue
9        ports:
10       - containerPort: 8080
11
12
```

**Q5 / Appliquez votre manifest au cluster via *kubectl apply -f*. Vérifiez que votre Pod a bien démarré. Trouvez un moyen d'accéder au port 8080 de votre conteneur (kubectl peut surement vous aider).**

# Technologie des conteneurs

The screenshot displays two windows side-by-side. The left window is a web browser showing the 'KUARD Demo' application. It features a red warning banner at the top stating 'WARNING: This server may expose sensitive and secret information. Be careful.' Below this, the application title 'kuard' is shown, followed by 'Demo application version v0.70.0-blue' and 'Serving on 10.244.0.10'. The right window is a Visual Studio Code editor titled 'podyaml - tp-containers - Visual Studio Code'. It shows a YAML manifest file for a pod named 'kuard'. The manifest includes the following details:

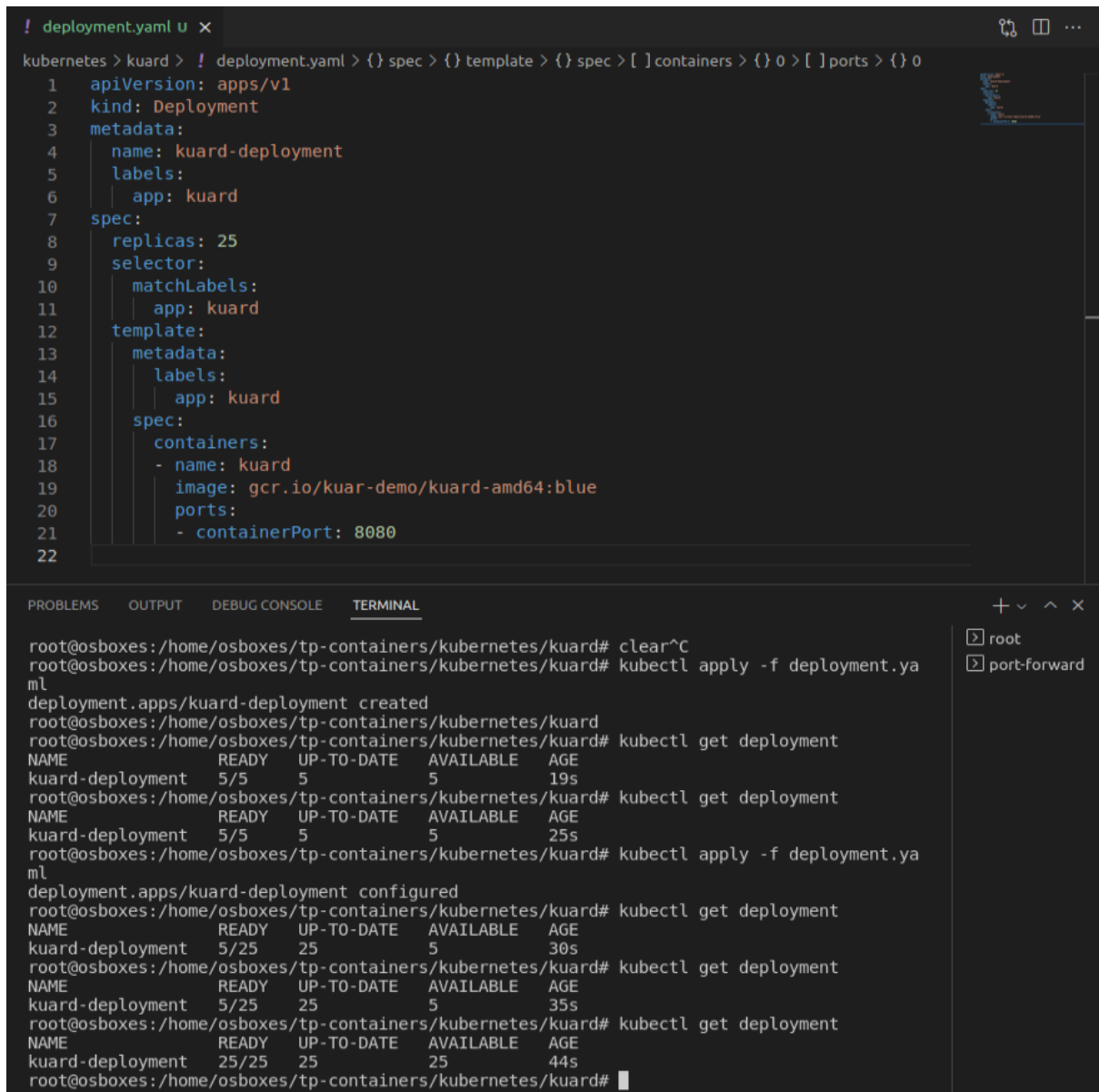
```
1 apiVersion: v1
2 kind: Pod
3 metadata:
4   name: kuard
5 spec:
6   containers:
7     - name: kuard
8       image: gcr.io/kuar-demo/kuard-amd64:blue
9       ports:
10        - containerPort: 8080
11
12
```

Below the editor, the terminal output shows the following commands and results:

```
root@osboxes:/home/osboxes/tp-containers/kubernetes/kuard# kubectl get pods
NAME      READY   STATUS    RESTARTS   AGE
kuard     1/1     Running   0          12s
root@osboxes:/home/osboxes/tp-containers/kubernetes/kuard# kubectl port-forward pod/kuard 80:8080
Forwarding from 127.0.0.1:80 -> 8080
Forwarding from [::]:80 -> 8080
Handling connection for 80
Handling connection for 80
```

**Q6 / Supprimez votre Pod et modifiez votre manifest afin d'en faire un Deployment (aidez vous de la doc de référence). Appliquez votre fichier, puis essayez de changer la valeur du champ *replicas* et de réappliquer votre fichier. Observez les pods dans votre cluster, que constatez vous ?**

# Technologie des conteneurs



The screenshot shows a VS Code editor with a file named `deployment.yaml` open. The file contains a Kubernetes Deployment manifest for an application named `kuard`. The manifest specifies 25 replicas, a selector matching the `app: kuard` label, and a container named `kuard` using the image `gcr.io/kuar-demo/kuard-amd64:blue` with port `8080` exposed.

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: kuard-deployment
5    labels:
6      app: kuard
7  spec:
8    replicas: 25
9    selector:
10     matchLabels:
11       app: kuard
12  template:
13    metadata:
14     labels:
15       app: kuard
16    spec:
17     containers:
18     - name: kuard
19       image: gcr.io/kuar-demo/kuard-amd64:blue
20       ports:
21       - containerPort: 8080
22
```

Below the editor, the terminal window shows the execution of `kubectl` commands. The first command, `kubectl apply -f deployment.yaml`, creates the deployment. Subsequent `kubectl get deployment` commands show the deployment's status as `READY` with 5/5 replicas available. The final command, `kubectl apply -f deployment.yaml`, updates the deployment, increasing the number of replicas to 25.

```
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard# clear^C
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard# kubectl apply -f deployment.yaml
deployment.apps/kuard-deployment created
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard# kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
kuard-deployment  5/5     5            5           19s
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard# kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
kuard-deployment  5/5     5            5           25s
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard# kubectl apply -f deployment.yaml
deployment.apps/kuard-deployment configured
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard# kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
kuard-deployment  5/25    25           5           30s
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard# kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
kuard-deployment  5/25    25           5           35s
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard# kubectl get deployment
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
kuard-deployment  25/25   25           25          44s
root@osboxes: /home/osboxes/tp-containers/kubernetes/kuard#
```

Les existants restent en vie, des nouveaux replicas sont ajoutés.

## Projet fil rouge

*Le but pour votre projet est d'écrire les objets Kubernetes nécessaires à l'exécution des microservices de votre application, mais pas encore à leur communication (ce sera le sujet du TP suivant).*

**Q7 / À la racine de votre projet, créez un dossier qui contiendra vos manifests kubernetes. Ajoutez ensuite un fichier `.yaml` par microservice qui compose votre application.**

Voir dossier `kubernetes/fil-rouge/`

# Technologie des conteneurs

**Q8 / Déterminez les composants Kubernetes nécessaires à l'exécution de vos services. Écrivez chacun des manifests correspondant (en ajoutant bien les informations comme le port des conteneurs). N'oubliez pas les labels.**

Voir dossier kubernetes/fil-rouge/

**Q9 / Appliquez vos manifests avec *kubectl apply -f* sur l'ensemble de votre dossier. Vérifiez que vos Pods sont bien créés**

```
root@osboxes:/home/osboxes/tp-containers/kubernetes/fil-rouge# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
details-deployment-7cd859df75-b4xgn 1/1     Running   0           7m25s
details-deployment-7cd859df75-dt28v 1/1     Running   0           7m25s
details-deployment-7cd859df75-wdnk4 1/1     Running   0           7m25s
mongodb-deployment-6749f56cd7-566pj 1/1     Running   0           7m11s
mysql-deployment-7f6fbbfffc-zctvw    1/1     Running   0           7m
productpage-deployment-7cc754d995-5fwtt 1/1     Running   0           6m43s
productpage-deployment-7cc754d995-5vw7d 1/1     Running   0           6m43s
productpage-deployment-7cc754d995-cxfrm6 1/1     Running   0           6m43s
productpage-deployment-7cc754d995-mv2t4 1/1     Running   0           6m43s
productpage-deployment-7cc754d995-z6sxx 1/1     Running   0           6m43s
ratings-deployment-blue-868d7bd475-2d7z4 1/1     Running   0           6m34s
ratings-deployment-blue-868d7bd475-2v4kd 1/1     Running   0           6m34s
ratings-deployment-blue-868d7bd475-jwnzf 1/1     Running   0           6m34s
ratings-deployment-blue-868d7bd475-rbf2b 1/1     Running   0           6m34s
ratings-deployment-blue-868d7bd475-tvll5 1/1     Running   0           6m34s
ratings-deployment-green-6c9cdf749f-2nqrj 1/1     Running   0           6m34s
ratings-deployment-green-6c9cdf749f-8b5lh 1/1     Running   0           6m33s
ratings-deployment-green-6c9cdf749f-9vsd5 1/1     Running   0           6m34s
ratings-deployment-green-6c9cdf749f-njmf 1/1     Running   0           6m34s
ratings-deployment-green-6c9cdf749f-x5krx 1/1     Running   0           6m33s
reviews-deployment-674954974-hctr6     1/1     Running   0           6m23s
reviews-deployment-674954974-qdw7x     1/1     Running   0           6m23s
reviews-deployment-674954974-wnwb7     1/1     Running   0           6m23s
reviews-deployment-674954974-zpbqd     1/1     Running   0           6m23s
reviews-deployment-674954974-zrstt     1/1     Running   0           6m23s
root@osboxes:/home/osboxes/tp-containers/kubernetes/fil-rouge#
```