



Kubernetes

Docker jour 3



Kezako

Kubernetes (K8s) est un système open source pour automatiquement déployer, scaler et manager le cycle de vie d'applications conteneurisées.

Il groupe les conteneurs qui forment une application en unité logique pour le discovery et le management. Il est construit sur 15 ans d'XP de Borg (l'outil de gestion de datacenter de Google), et open-source depuis 2016.



Declaratif vs impératif

Assis-toi ! VS J'aimerais que tu sois assis.

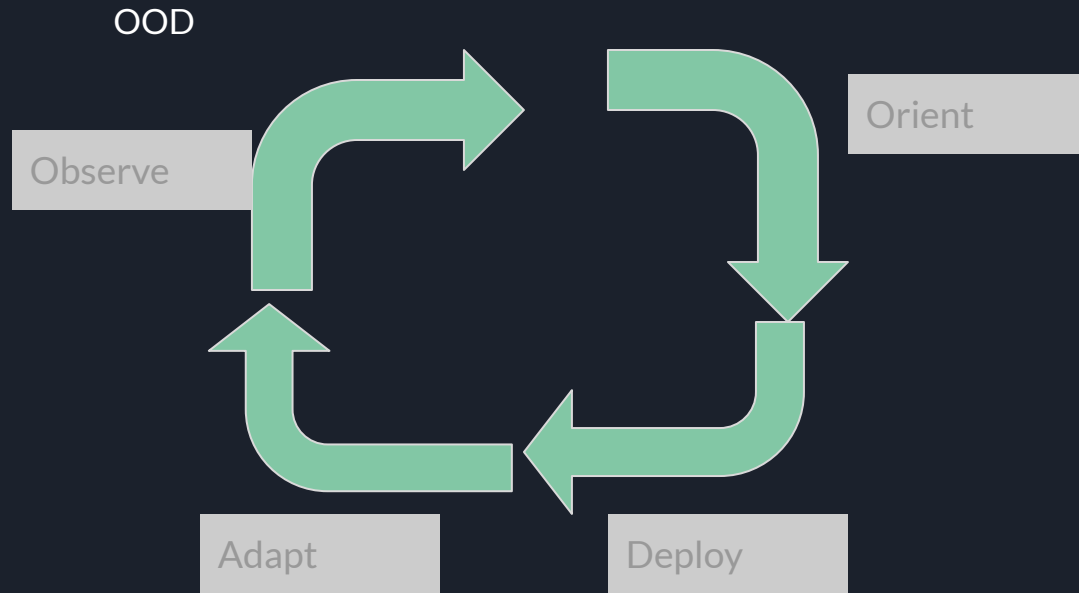
Description de l'état désiré VS Suite de commandes



Declaratif.sh

```
#!/usr/bin/env bash
if [! -f $(pwd)/file-declarative.txt ]; then
    echo "declarative" >> file-declarative.txt
    echo "File created"
else
    echo "no action needed, file exists"
fi
```

Control loop





Réconciliation

ça va pas -> on corrige



TP

install kubectl (kubecontroleur)

<https://kubernetes.io/docs/tasks/tools/install-kubectl/>

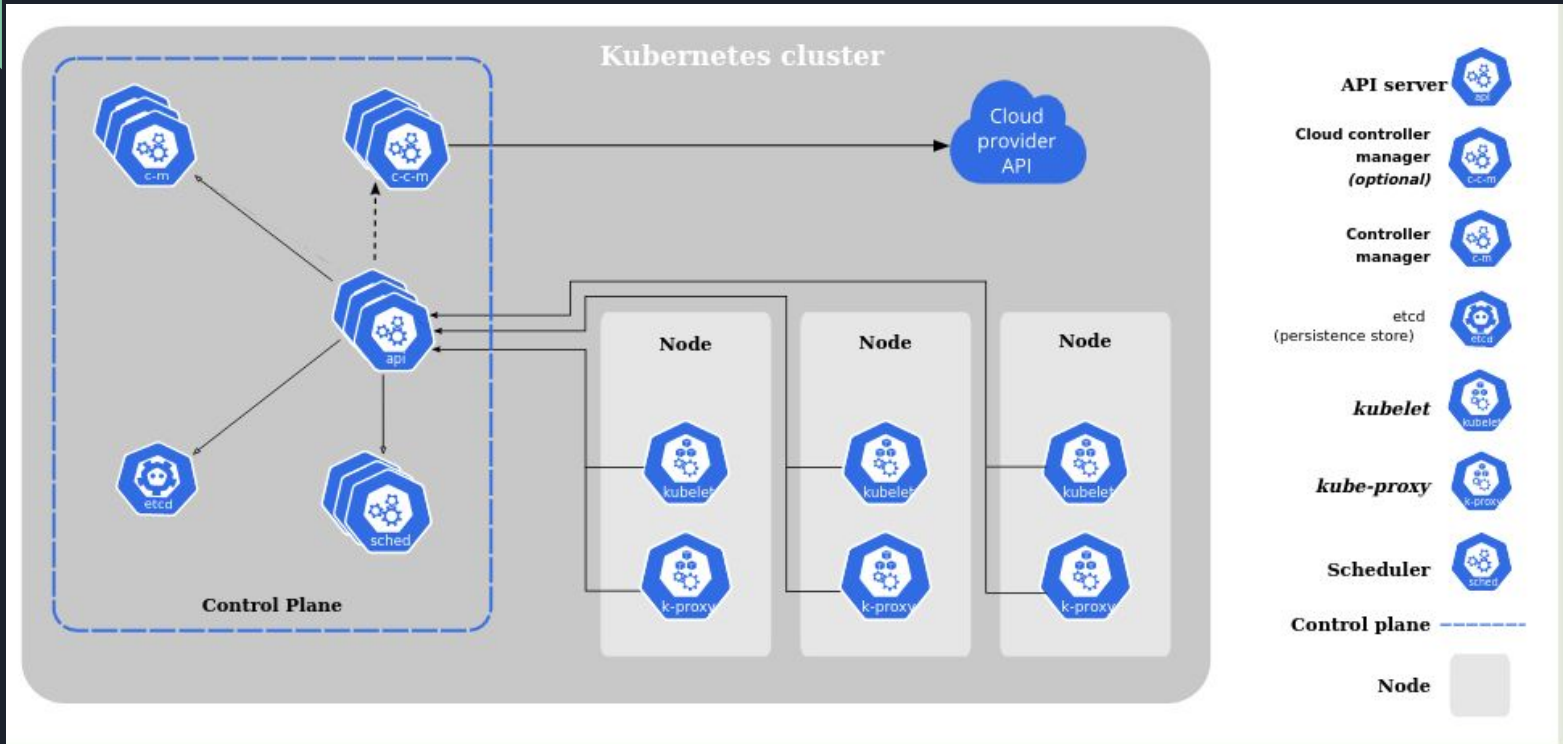
```
$ gcloud components install kubectl
```

```
$ gcloud container clusters get-credentials cluster-test --zone europe-west1-d --project  
formation-docker-inter
```

```
$ kubectl get nodes
```

```
$ gcloud compute instances list
```

Architecture





Composants du control-plane

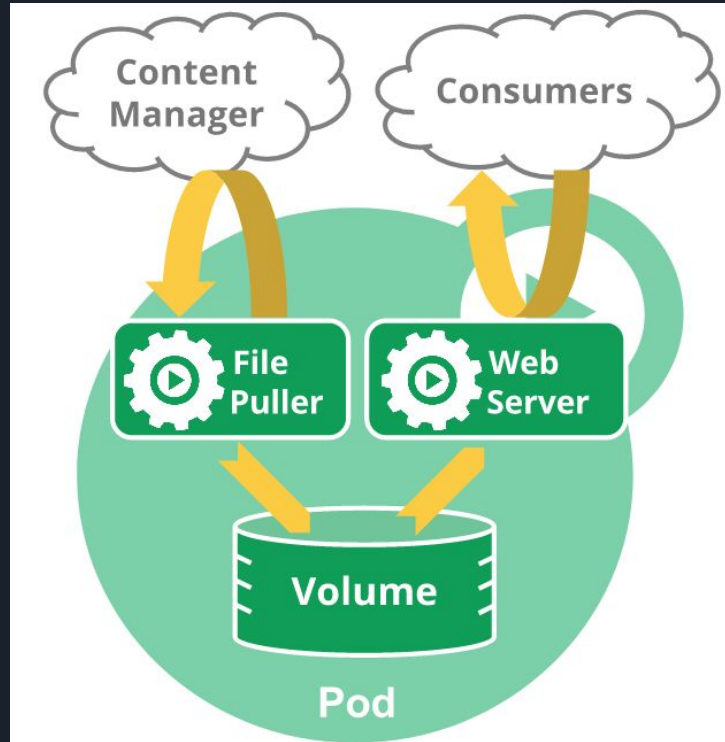
- `kube-apiserver`
- `etcd`
- `kube-scheduler`
- `kube-controller-manager`
 - `Node controller`: Responsible for noticing and responding when nodes go down.
 - `Replication controller`: Responsible for maintaining the correct number of pods for every replication controller object in the system.
 - `Endpoints controller`: Populates the Endpoints object (that is, joins Services & Pods).
 - `Service Account & Token controllers`: Create default accounts and API access tokens for new namespaces
- `cloud-controller-manager`




Components : Noeud

- kubelet
- kube-proxy
- Container runtime

Pod





On ne déploie pas un pod, on déploie un contrôleur

- Deployment and ReplicaSet
- StatefulSet;
- DaemonSet Job and CronJob.



TP 3

```
$ git clone https://github.com/GoogleCloudPlatform/kubernetes-engine-samples
```

```
$ cd kubernetes-engine-samples/hello-app
```

```
$ docker build -t gcr.io/formation-docker-inter/hello-app-{prénom}:v1 .
```

```
$ docker run --rm gcr.io/formation-docker-inter/hello-app-{prénom}:v1
```

```
$ docker push gcr.io/formation-docker-inter/hello-app-{prénom}:v1
```

```
$ alias k=kubectl
```

```
$ k get pods --watch
```



On déploie une application

```
$ kubectl create deployment hello-app-{prénom}  
--image=gcr.io/formation-docker-inter/hello-app-{prénom}:v1
```

```
$ kubectl scale deployment hello-app-fabien --replicas=2
```

```
$ kubectl autoscale deployment hello-app-fabien --cpu-percent=80 --min=1 --max=5
```

```
$ kubectl expose deployment hello-app-fabien --name=hello-app-fabien-service  
--type=LoadBalancer --port 80 --target-port 8080
```

```
$ kubectl get services | k get svc
```

```
$
```



On l'expose sur internet

```
$gcloud compute addresses create helloweb-ip-fabien --region europe-west1
$#on va regarder l'IP qu'on a reçu : xx.Xx.xx.xX
$ gcloud compute addresses list
$ kubectl edit service hello-app-fabien-service
> ++  type: LoadBalancer
      loadBalancerIP: "YOUR.IP.ADDRESS.HERE"
$ curl 35.233.43.100:80
```