

Kubernetes pour les dev

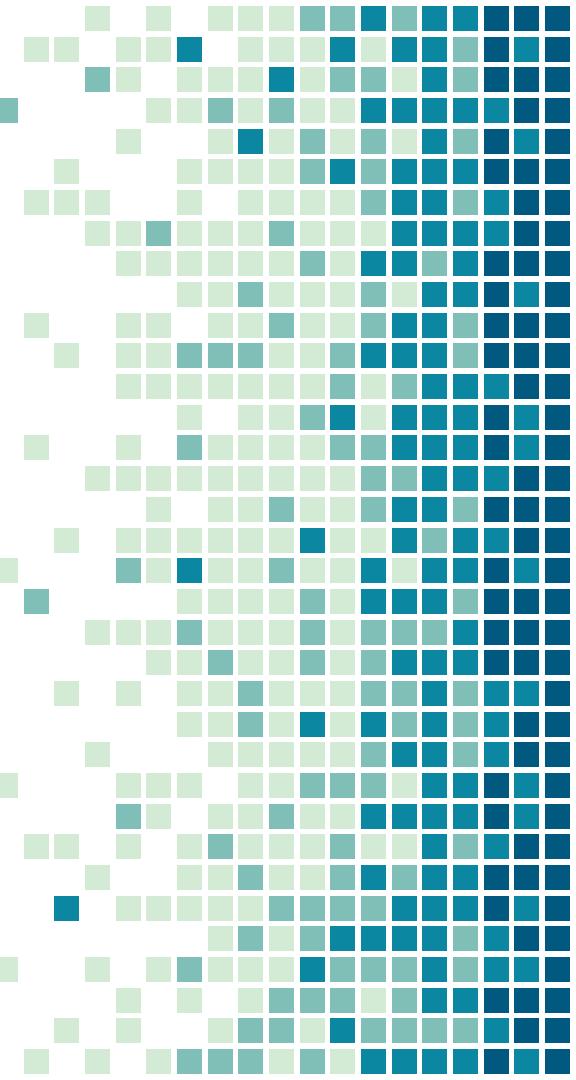


-- Cyril zarak Duval, root CRI/ACU 2020

diabolocom
Customer interaction. Augmented.

Présentation

Quelques généralités

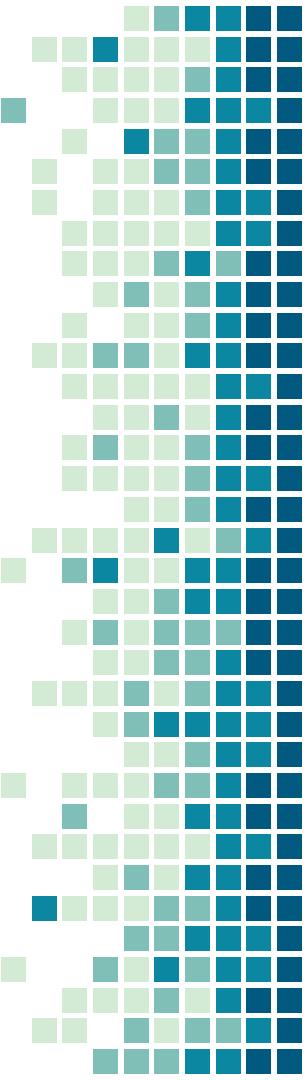


Pourquoi écouter cette présentation ?

- Développer c'est bien, sans production ça sert à rien

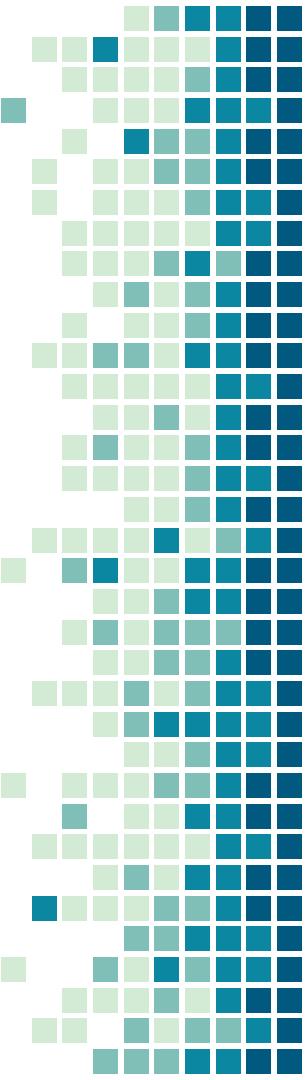
Pourquoi écouter cette présentation ?

- Développer c'est bien, sans production ça sert à rien
- Notions de DevOps



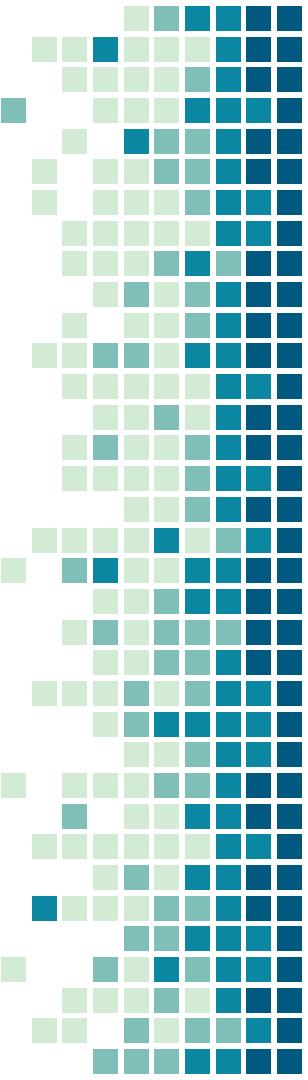
Pourquoi écouter cette présentation ?

- Développer c'est bien, sans production ça sert à rien
- Notions de DevOps
- Connaître les bons outils = gain de temps



Pourquoi écouter cette présentation ?

- Développer c'est bien, sans production ça sert à rien
- Notions de DevOps
- Connaître les bons outils = gain de temps
- Sujet difficile à aborder



Pourquoi écouter cette présentation ?

- Développer c'est bien, sans production ça sert à rien
- Notions de DevOps
- Connaître les bons outils = gain de temps
- Sujet difficile à aborder
- QCM à la fin

Qu'est-ce que Kubernetes ?

→ Projet de Google

Qu'est-ce que Kubernetes ?

- Projet de Google
 - ◆ Adaptation de Borg en open source

Qu'est-ce que Kubernetes ?

- Projet de Google
 - ◆ Adaptation de Borg en open source
- Sorti en 2015

Qu'est-ce que Kubernetes ?

- Projet de Google
 - ◆ Adaptation de Borg en open source
- Sorti en 2015
- Orchestrateur de conteneurs

Qu'est-ce que Kubernetes ?

- Projet de Google
 - ◆ Adaptation de Borg en open source
- Sorti en 2015
- Orchestrateur de conteneurs
- Plateforme de déploiement, mise en production

Qu'est-ce que Kubernetes ?

- Projet de Google
 - ◆ Adaptation de Borg en open source
- Sorti en 2015
- Orchestrateur de conteneurs
- Plateforme de déploiement, mise en production
- Gestion de la montée en charge

Qu'est-ce que Kubernetes ?

- Projet de Google
 - ◆ Adaptation de Borg en open source
- Sorti en 2015
- Orchestrateur de conteneurs
- Plateforme de déploiement, mise en production
- Gestion de la montée en charge
- Haute disponibilité

Quels problèmes cherche à résoudre Kubernetes ?

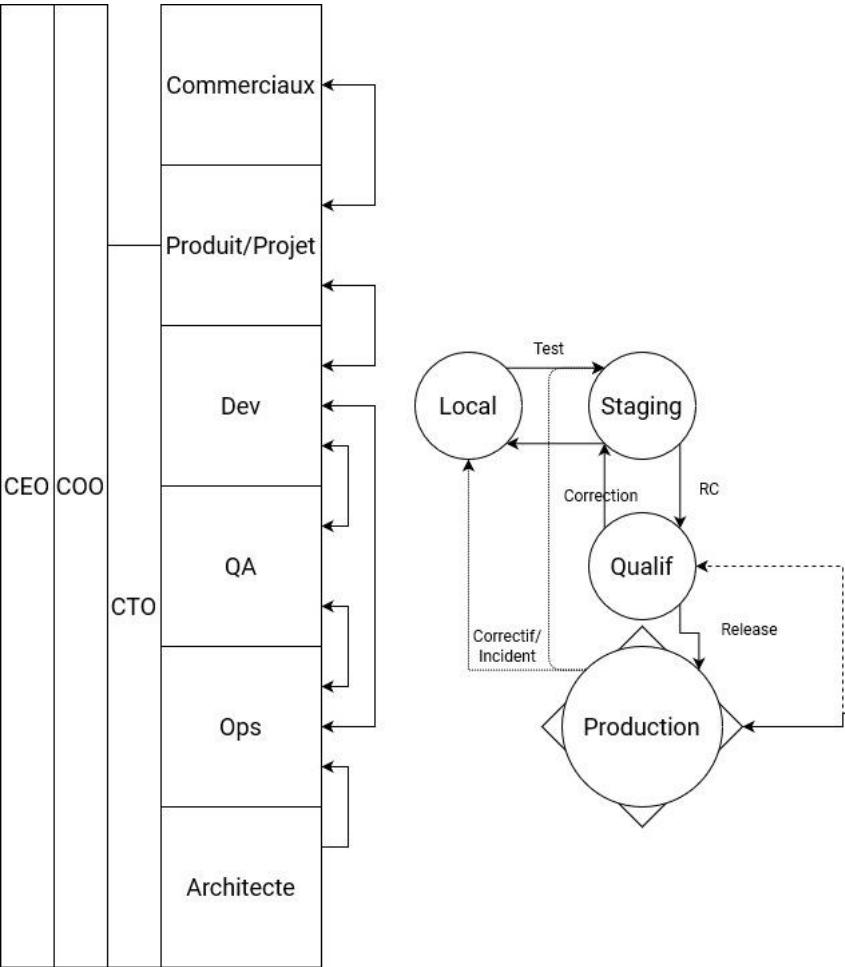
Sans surprise, ça ne sert pas à rien



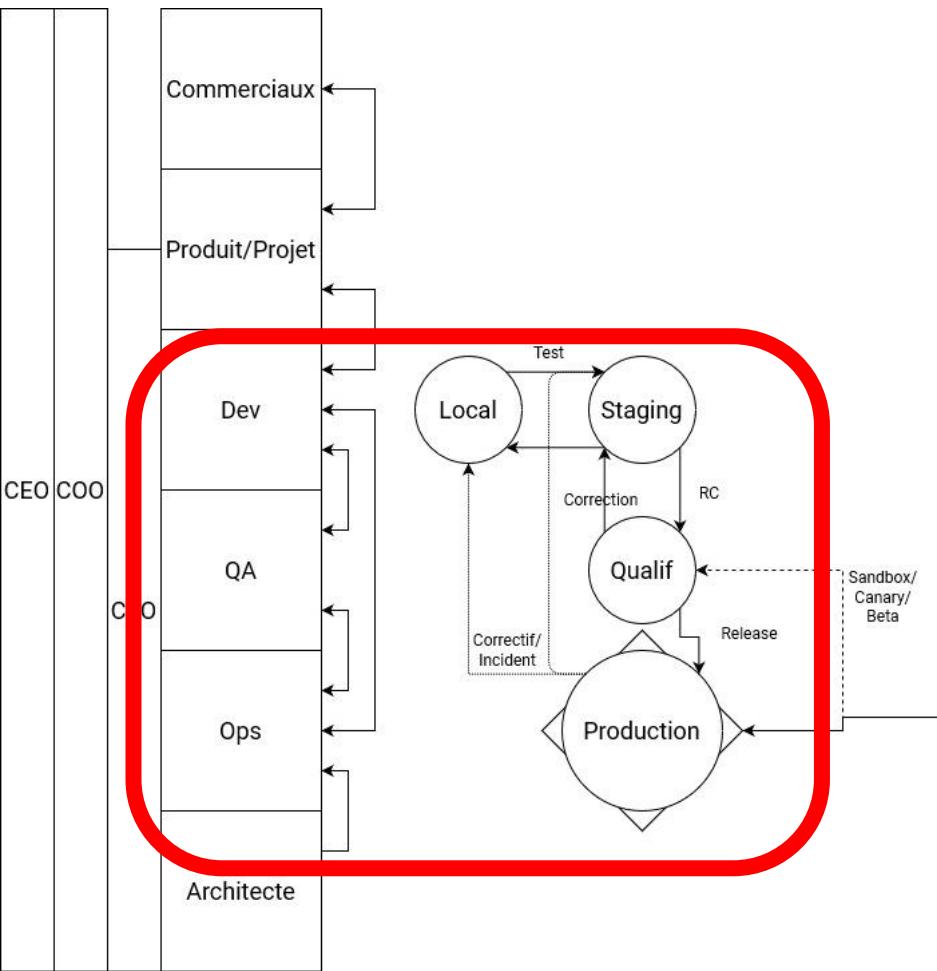
Après le dev, l'ops

- Cycle de vie du développement en entreprise
 - ◆ Exemple simplifié
 - ◆ PME
 - ◆ ~SaaS

Après le dev, l'ops



Le devops



Le DevOps

- Le dev travaille sur son laptop
 - ◆ Son OS
 - ◆ Ses bibliothèques
 - ◆ Son environnement
 - ◆ Sa stack réseau
 - ◆ ...

Le DevOps

- Le dev travaille sur son laptop
 - ◆ Son OS
 - ◆ Ses bibliothèques
 - ◆ Son environnement
 - ◆ Sa stack réseau
 - ◆ ...
- Envoyer sur la prod peut poser des problèmes

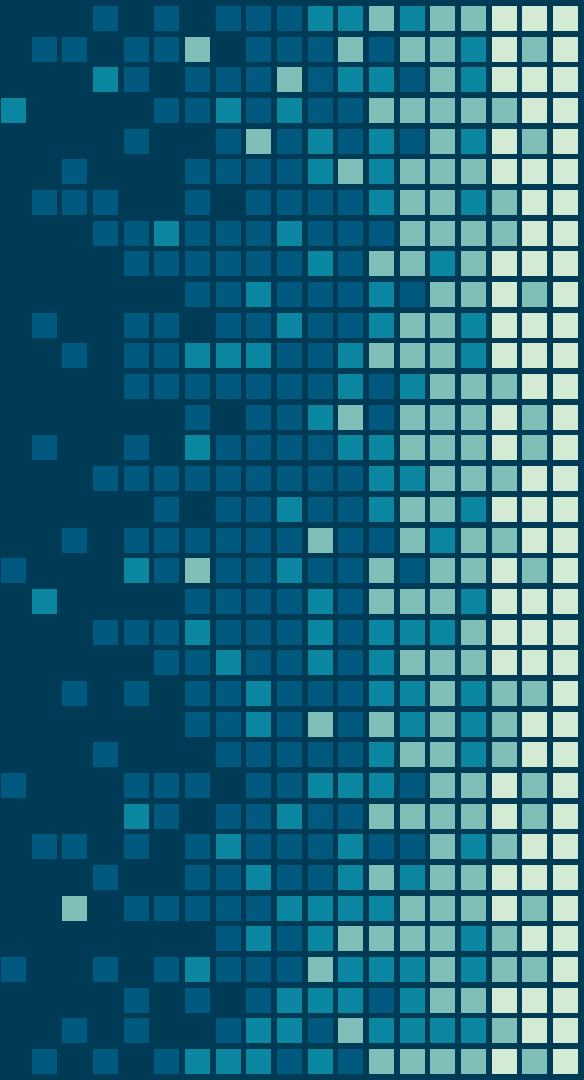
Le DevOps

- Le dev travaille sur son laptop
 - ◆ Son OS
 - ◆ Ses bibliothèques
 - ◆ Son environnement
 - ◆ Sa stack réseau
 - ◆ ...
- Envoyer sur la prod peut poser des problèmes
- On va pas envoyer son laptop sur la prod

Le DevOps - docker

- Le dev travaille sur son laptop avec docker

Démystification de Docker par l'exemple



Le DevOps - docker

- Le dev travaille sur son laptop avec docker
- Image docker :
 - ◆ Contrôle de l'OS
 - FROM debian:11

Le DevOps - docker

- Le dev travaille sur son laptop avec docker
- Image docker :
 - ◆ Contrôle de l'OS
 - ◆ Contrôle du réseau

Le DevOps - docker

- Le dev travaille sur son laptop avec docker
- Image docker :
 - ◆ Contrôle de l'OS
 - ◆ Contrôle du réseau
 - ◆ Contrôle des dépendances
 - FROM python:3-alpine
 - RUN pip install ansible

Le DevOps - docker

- Le dev travaille sur son laptop avec docker
- Image docker :
 - ◆ Contrôle de l'OS
 - ◆ Contrôle du réseau
 - ◆ Contrôle des dépendances
 - ◆ Contrôle des versions
 - FROM python:3-alpine
 - RUN pip install ansible==2.10

Le DevOps - docker

- Le dev travaille sur son laptop avec docker
- Image docker :
 - ◆ Contrôle de l'OS
 - ◆ Contrôle du réseau
 - ◆ Contrôle des dépendances
 - ◆ Contrôle des versions
- Livrable = lien vers le docker registry

Le DevOps - docker

- Le dev travaille sur son laptop avec docker
- Image docker :
 - ◆ Contrôle de l'OS
 - ◆ Contrôle du réseau
 - ◆ Contrôle des dépendances
 - ◆ Contrôle des versions
- Livrable = lien vers le docker registry
 - ◆ 2-3 broutilles (variables env, ports, etc)

Le DevOps - docker

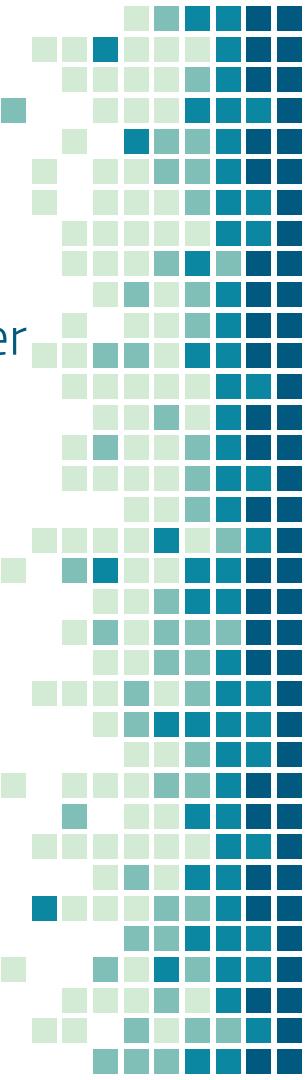
- Image docker = application, ~binaire

Le DevOps - docker

- Image docker = application, ~binaire
- ```
docker run \
-p 9200:9200 \
-p 9300:9300 \
-e "discovery.type=single-node" \
-v /srv/es/data:/usr/share/elasticsearch/data \
docker.elastic.co/elasticsearch/elasticsearch:7.15.2
```

# Le DevOps - docker

- Image docker = application, ~binaire
- ```
docker run \
-p 9200:9200 \
-p 9300:9300 \
-e "discovery.type=single-node" \
-v /srv/es/data:/usr/share/elasticsearch/data \
docker.elastic.co/elasticsearch/elasticsearch:7.15.2
```
- Manque de tooling pour le déploiement, fichier de conf



Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker
-

```
● ● ●

1 services:
2   es01:
3     image: docker.elastic.co/elasticsearch/elasticsearch:7.15.2
4     container_name: elasticsearch
5     environment:
6       - node.name=node01
7       - discovery.type=single-node
8       - bootstrap.memory_lock=true
9       - "ES_JAVA_OPTS=-Xms512m -Xmx512m"
10    volumes:
11      - /srv/es/data:/usr/share/elasticsearch/data
12    ports:
13      - 9200:9200
14      - 9300:9300
```

Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker

Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker
- Pas de notions de :

Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker
- Pas de notions de :
 - ◆ Scaling horizontal

Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker
- Pas de notions de :
 - ◆ Scaling horizontal
 - ◆ Haute disponibilité

Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker
- Pas de notions de :
 - ◆ Scaling horizontal
 - ◆ Haute disponibilité
 - ◆ Répartition de charge

Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker
- Pas de notions de :
 - ◆ Scaling horizontal
 - ◆ Haute disponibilité
 - ◆ Répartition de charge
 - ◆ Multi-noeuds

Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker
- Pas de notions de :
 - ◆ Scaling horizontal
 - ◆ Haute disponibilité
 - ◆ Répartition de charge
 - ◆ Multi-noeuds
 - ◆ Déploiement (Rolling Release, rollback, ...)

Le DevOps - docker-compose

- docker-compose.yml = fichier de conf déclaratif de la CLI docker
- Pas de notions de :
 - ◆ Scaling horizontal
 - ◆ Haute disponibilité
 - ◆ Répartition de charge
 - ◆ Multi-noeuds
 - ◆ Déploiement (Rolling Release, rollback, ...)
 - ◆ ...

Le DevOps - Kubernetes

- Kubernetes :
 - ◆ Multi-noeuds

Le DevOps - Kubernetes

- Kubernetes :
 - ◆ Multi-noeuds
 - ◆ Déclaratif

Le DevOps - Kubernetes

- Kubernetes :
 - ◆ Multi-noeuds
 - ◆ Déclaratif
 - ◆ Notions de services, secrets, configuration, réseau, ...

Le DevOps - Kubernetes

- Kubernetes :
 - ◆ Multi-noeuds
 - ◆ Déclaratif
 - ◆ Notions de services, secrets, configuration, réseau, ...
- Production-ready

Le DevOps - Kubernetes

→ Pour le dev :

Le DevOps - Kubernetes

- Pour le dev :
 - ◆ Dev avec docker en local

Le DevOps - Kubernetes

- Pour le dev :
 - ◆ Dev avec docker en local
 - Voir k3s/minikube/KinD

Le DevOps - Kubernetes

- Pour le dev :
 - ◆ Dev avec docker en local
 - Voir k3s/minikube/KinD
 - ◆ Intégration de l'image docker dans les ressources k8s

Le DevOps - Kubernetes

- Pour le dev :
 - ◆ Dev avec docker en local
 - Voir k3s/minikube/KinD
 - ◆ Intégration de l'image docker dans les ressources k8s
 - ◆ Environment staging vs production similaires :

Le DevOps - Kubernetes

- Pour le dev :
 - ◆ Dev avec docker en local
 - Voir k3s/minikube/KinD
 - ◆ Intégration de l'image docker dans les ressources k8s
 - ◆ Environment staging vs production similaires :
 - Même ressources k8s

Le DevOps - Kubernetes

- Pour le dev :
 - ◆ Dev avec docker en local
 - Voir k3s/minikube/KinD
 - ◆ Intégration de l'image docker dans les ressources k8s
 - ◆ Environment staging vs production similaires :
 - Même ressources k8s
 - Différence scalabilité, ressources hardware, ...

Le DevOps - Kubernetes

- Pour le dev :
 - ◆ Dev avec docker en local
 - Voir k3s/minikube/KinD
 - ◆ Intégration de l'image docker dans les ressources k8s
 - ◆ Environment staging vs production similaires :
 - Même ressources k8s
 - Différence scalabilité, ressources hardware, ...
 - ◆ Workflow avec le tooling de k8s

Le DevOps - Kubernetes

- Ressources k8s déclaratives :
 - ◆ Mise en commun dev et ops

Le DevOps - Kubernetes

- Ressources k8s déclaratives :
 - ◆ Mise en commun dev et ops
 - ◆ Templating :

Le DevOps - Kubernetes

- Ressources k8s déclaratives :
 - ◆ Mise en commun dev et ops
 - ◆ Templating :
 - Templates identiques staging/production

Le DevOps - Kubernetes

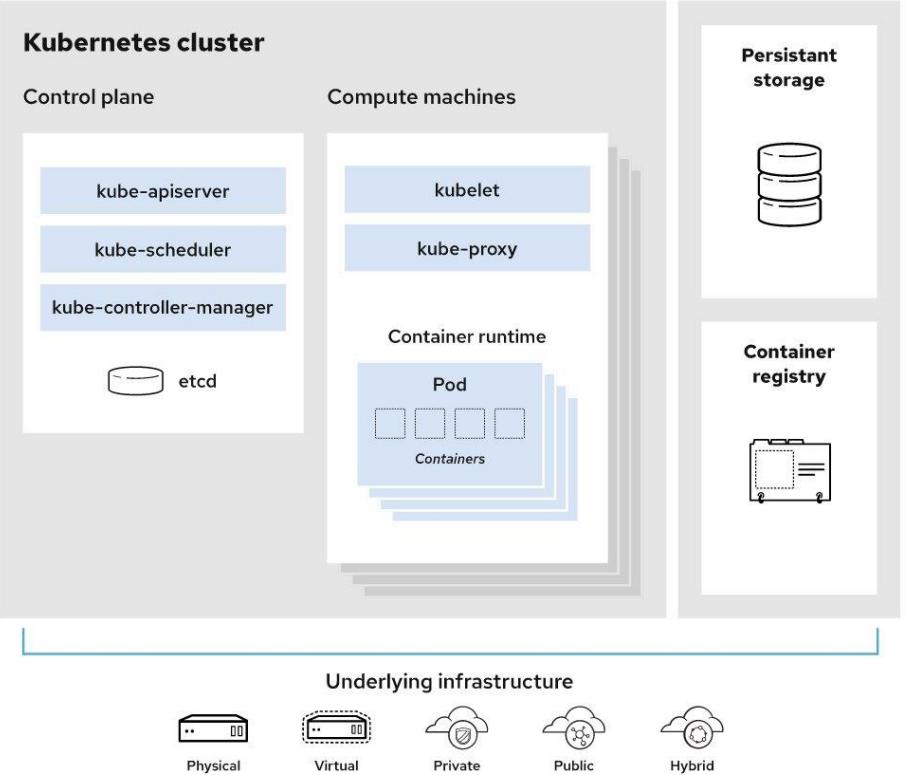
- Ressources k8s déclaratives :
 - ◆ Mise en commun dev et ops
 - ◆ Templating :
 - Templates identiques staging/production
 - Valeurs qui changent

A quoi ressemble Kubernetes ?

Une chimère à plusieurs visages



Kubernetes 101 - Architecture



Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources
 - ◆ Création, suppression, modification, ...

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources
 - ◆ Création, suppression, modification, ...
- Composants de k8s = ressources

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources
 - ◆ Création, suppression, modification, ...
- Composants de k8s = ressources
 - ◆ nodes

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources
 - ◆ Création, suppression, modification, ...
- Composants de k8s = ressources
 - ◆ nodes
 - ◆ namespaces

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources
 - ◆ Création, suppression, modification, ...
- Composants de k8s = ressources
 - ◆ nodes
 - ◆ namespaces
 - ◆ persistentvolumes

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources
 - ◆ Création, suppression, modification, ...
- Composants de k8s = ressources
 - ◆ nodes
 - ◆ namespaces
 - ◆ persistentvolumes
 - ◆ configmaps

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources
 - ◆ Création, suppression, modification, ...
- Composants de k8s = ressources
 - ◆ nodes
 - ◆ namespaces
 - ◆ persistentvolumes
 - ◆ configmaps
 - ◆ pods

Kubernetes 101 - Les ressources

- Interaction avec l'API de Kubernetes via des ressources
- Interaction avec k8s sur les ressources
 - ◆ Création, suppression, modification, ...
- Composants de k8s = ressources
 - ◆ nodes
 - ◆ namespaces
 - ◆ persistentvolumes
 - ◆ configmaps
 - ◆ pods
 - ◆ Pas de "container"

Kubernetes 101 - Le pod

- Plus petite unité d'exécution de Kubernetes

Kubernetes 101 - Le pod

- Plus petite unité d'exécution de Kubernetes
- Assimilable à un conteneur docker

Kubernetes 101 - Le pod

- Plus petite unité d'exécution de Kubernetes
- Assimilable à un conteneur docker
 - ◆ 90% des cas pod = conteneur docker

Kubernetes 101 - Le pod

- Plus petite unité d'exécution de Kubernetes
- Assimilable à un conteneur docker
 - ◆ 90% des cas pod = conteneur docker
- Spécificités :

Kubernetes 101 - Le pod

- Plus petite unité d'exécution de Kubernetes
- Assimilable à un conteneur docker
 - ◆ 90% des cas pod = conteneur docker
- Spécificités :
 - ◆ Potentiellement plusieurs containers par pod

Kubernetes 101 - Le pod

- Plus petite unité d'exécution de Kubernetes
- Assimilable à un conteneur docker
 - ◆ 90% des cas pod = conteneur docker
- Spécificités :
 - ◆ Potentiellement plusieurs containers par pod
 - Principal + sidecar

Kubernetes 101 - Le pod

- Plus petite unité d'exécution de Kubernetes
- Assimilable à un conteneur docker
 - ◆ 90% des cas pod = conteneur docker
- Spécificités :
 - ◆ Potentiellement plusieurs containers par pod
 - Principal + sidecar
 - Des init containers (migration DB)

Kubernetes 101 - Le pod

- Plus petite unité d'exécution de Kubernetes
- Assimilable à un conteneur docker
 - ◆ 90% des cas pod = conteneur docker
- Spécificités :
 - ◆ Potentiellement plusieurs containers par pod
 - Principal + sidecar
 - Des init containers (migration DB)
 - ◆ Même network namespaces(7)

Kubernetes 101 - Le pod

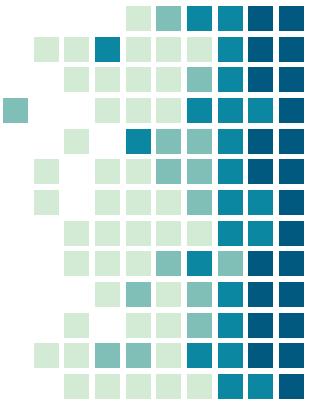
- Plus petite unité d'exécution de Kubernetes
- Assimilable à un conteneur docker
 - ◆ 90% des cas pod = conteneur docker
- Spécificités :
 - ◆ Potentiellement plusieurs containers par pod
 - Principal + sidecar
 - Des init containers (migration DB)
 - ◆ Même network namespaces(7)
 - ◆ Même shared volumes

Kubernetes 101 - kubectl



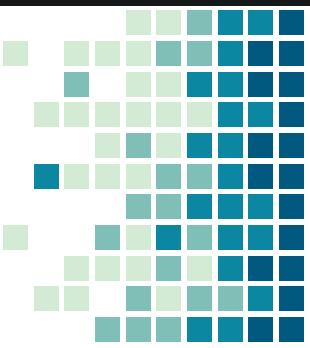
```
1 $ kubectl get nodes
2 NAME                  STATUS   ROLES      AGE     VERSION
3 km01-d01-rdb.dev dblc.io  Ready    controlplane  609d  v1.20.5
4 km02-d01-rdb.dev dblc.io  Ready    controlplane  609d  v1.20.5
5 kw01-d01-rdb.dev dblc.io  Ready    worker      609d  v1.20.5
6 kw02-d01-rdb.dev dblc.io  Ready    worker      609d  v1.20.5
7 kw03-d01-rdb.dev dblc.io  Ready    worker      609d  v1.20.5
8 kw04-d01-rdb.dev dblc.io  Ready    worker      609d  v1.20.5
9 kw05-d01-rdb.dev dblc.io  Ready    worker      498d  v1.20.5
```

Kubernetes 101 - kubectl



```
1 $ k get node -o wide
```

	NAME	STATUS	ROLES	AGE	VERSION	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE	KERNEL-VERSION	CONTAINER-RUNTIME
2	km01-d01-rdb.dev.dblc.io	Ready	controlplane	609d	v1.20.5	10.3.196.50	<none>	Debian GNU/Linux 9 (stretch)	4.9.0-7-amd64	docker://19.3.7
3	km02-d01-rdb.dev.dblc.io	Ready	controlplane	609d	v1.20.5	10.3.196.51	<none>	Debian GNU/Linux 9 (stretch)	4.9.0-7-amd64	docker://19.3.7
5	kw01-d01-rdb.dev.dblc.io	Ready	worker	609d	v1.20.5	10.3.196.52	<none>	Debian GNU/Linux 9 (stretch)	4.9.0-7-amd64	docker://19.3.7
6	kw02-d01-rdb.dev.dblc.io	Ready	worker	609d	v1.20.5	10.3.196.53	<none>	Debian GNU/Linux 9 (stretch)	4.9.0-7-amd64	docker://19.3.7
7	kw03-d01-rdb.dev.dblc.io	Ready	worker	609d	v1.20.5	10.3.196.54	<none>	Debian GNU/Linux 9 (stretch)	4.9.0-7-amd64	docker://19.3.7
8	kw04-d01-rdb.dev.dblc.io	Ready	worker	609d	v1.20.5	10.3.196.55	<none>	Debian GNU/Linux 9 (stretch)	4.9.0-7-amd64	docker://19.3.7
9	kw05-d01-rdb.dev.dblc.io	Ready	worker	498d	v1.20.5	10.3.196.56	<none>	Debian GNU/Linux 9 (stretch)	4.9.0-7-amd64	docker://19.3.7



Kubernetes 101 - kubectl

```
● ● ●  
1 $ kubectl get po  
2 NAME                  READY   STATUS    RESTARTS   AGE  
3 infinite-2loops        2/2     Running   0          4m8s  
4 infinite-loop          1/1     Running   0          4m3s
```

Kubernetes 101 - kubectl



```
1 $ kubectl logs infinite-loop
2 going to sleep for 10s every 10s
3 Sat Nov 20 14:16:54 UTC 2021
4 Sat Nov 20 14:17:04 UTC 2021
5 Sat Nov 20 14:17:14 UTC 2021
6 Sat Nov 20 14:17:24 UTC 2021
7 Sat Nov 20 14:17:34 UTC 2021
8 Sat Nov 20 14:17:44 UTC 2021
9 Sat Nov 20 14:17:54 UTC 2021
10 Sat Nov 20 14:18:04 UTC 2021
11 Sat Nov 20 14:18:14 UTC 2021
12 Sat Nov 20 14:18:24 UTC 2021
13 Sat Nov 20 14:18:34 UTC 2021
```



```
1 $ kubectl describe po infinite-loop
2 Name:           infinite-loop
3 Namespace:      default
4 Priority:      0
5 Node:          chlorine/192.168.1.18
6 Start Time:   Sat, 20 Nov 2021 15:16:43 +0100
7 Labels:        <none>
8 Annotations:   <none>
9 Status:        Running
10 IP:           10.42.0.50
11 IPs:
12 IP:  10.42.0.50
13 Containers:
14   container1:
15     Container ID: docker://ba89414b1d829dd6f54777a87af5b37fe1f0395c68779762d0b2ecbcd266b0da
16     Image:         pause
17     Image ID:     docker://sha256:d18c91061232135b43592c31f7fd1ca73e4afee1e616ca8795f1dc9fa8de8c79
18     Port:          <none>
19     Host Port:    <none>
20     State:        Running
21     Started:     Sat, 20 Nov 2021 15:16:44 +0100
22     Ready:        True
23     Restart Count: 0
24     Environment:  <none>
25     Mounts:
26       /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-bqvk2 (ro)
27 Conditions:
28   Type      Status
29   Initialized  True
30   Ready      True
31   ContainersReady  True
32   PodScheduled  True
```

Kubernetes 101 - kubectl

```
● ● ●

1 $ kubectl exec -it infinite-loop -- bash
2 bash-5.1$ hostname
3 infinite-loop
4 bash-5.1$ ps aux
5 PID   USER      TIME  COMMAND
6     1 root      0:00 bash -c echo going to sleep for 10s every 10s; while true; do sleep 10; date; done
7    547 root      0:00 sleep 10
8    548 root      0:00 bash
9    555 root      0:00 ps aux
10 bash-5.1$ env
11 KUBERNETES_SERVICE_PORT_HTTPS=443
12 KUBERNETES_SERVICE_PORT=443
13 HOSTNAME=infinite-loop
14 PWD=/
15 HOME=/root
16 KUBERNETES_PORT_443_TCP=tcp://10.43.0.1:443
17 TERM=xterm
18 SHLVL=1
19 KUBERNETES_PORT_443_TCP_PROTO=tcp
20 KUBERNETES_PORT_443_TCP_ADDR=10.43.0.1
21 KUBERNETES_SERVICE_HOST=10.43.0.1
22 KUBERNETES_PORT=tcp://10.43.0.1:443
23 KUBERNETES_PORT_443_TCP_PORT=443
24 PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
25 _=/usr/bin/env
```

Kubernetes 101 - kubectl

```
● ● ●  
1 $ kubectl get po  
2 NAME          READY   STATUS    RESTARTS   AGE  
3 infinite-loop  1/1     Running   0          44m  
4 infinite-2loops 2/2     Running   0          44m  
5 $ kubectl delete po infinite-loop  
6 pod "infinite-loop" deleted  
7 $ kubectl get po  
8 NAME          READY   STATUS    RESTARTS   AGE  
9 infinite-2loops 2/2     Running   0          45m
```

Kubernetes 101 - docker ps



```
1 $ docker ps --format 'table {{.Image}}\t{{.Command}}\t{{.RunningFor}}\t{{.Names}}'
2 IMAGE           COMMAND      CREATED     NAMES
3 d18c91061232  "bash -c 'echo going..."  53 minutes ago   k8s_container2_infinite-2loops_default_8af0ba4c-e6c8-4831-9537-6192c896c72f_0
4 d18c91061232  "bash -c 'echo going..."  53 minutes ago   k8s_container1_infinite-2loops_default_8af0ba4c-e6c8-4831-9537-6192c896c72f_0
5 rancher/pause:3.1  "/pause"        53 minutes ago   k8s_POD_infinite-2loops_default_8af0ba4c-e6c8-4831-9537-6192c896c72f_0
6 465db341a9e5   "entry"         2 hours ago    k8s_lb-port-443_svclb-traefik-krzqs_kube-system_6bc33121-d8c8-459f-ba55-8529835ea77d_1
7 465db341a9e5   "entry"         2 hours ago    k8s_lb-port-80_svclb-traefik-krzqs_kube-system_6bc33121-d8c8-459f-ba55-8529835ea77d_1
8 deaf4b1027ed   "/entrypoint.sh --gl..."  2 hours ago    k8s_traefik_traefik-97b44b794-jj86m_kube-system_bbef9cda-3ae8-41d9-b520-2f9cd318b811_1
9 rancher/pause:3.1  "/pause"        2 hours ago    k8s_POD_svclb-traefik-krzqs_kube-system_6bc33121-d8c8-459f-ba55-8529835ea77d_2
10 3885a5b7f138   "/coredns -conf /etc..."  2 hours ago   k8s_coredns_coredns-7448499f4d-gpksg_kube-system_5202957b-d36f-44cc-a9b2-a5f941f8bfa8_1
11 rancher/pause:3.1  "/pause"        2 hours ago   k8s_POD_traefik-97b44b794-jj86m_kube-system_bbef9cda-3ae8-41d9-b520-2f9cd318b811_1
12 rancher/pause:3.1  "/pause"        2 hours ago   k8s_POD_coredns-7448499f4d-gpksg_kube-system_5202957b-d36f-44cc-a9b2-a5f941f8bfa8_2
13 9dd718864ce6   "/metrics-server"  2 hours ago   k8s_metrics-server_metrics-server-86cbb8457f-t54g4_kube-system_63f4afe1-3a25-47ad-a5d7-6cb9cef4e90_2
14 148c19256271   "local-path-provision..."  2 hours ago   k8s_local-path-provisioner_local-path-provisioner-5ff76fc89d-sbwvg_kube-system_8a0cb047-019b-459d-950a-db33bd6e62b5_2
15 rancher/pause:3.1  "/pause"        2 hours ago   k8s_POD_metrics-server-86cbb8457f-t54g4_kube-system_63f4afe1-3a25-47ad-a5d7-6cb9cef4e90_2
16 rancher/pause:3.1  "/pause"        2 hours ago   k8s_POD_local-path-provisioner-5ff76fc89d-sbwvg_kube-system_8a0cb047-019b-459d-950a-db33bd6e62b5_2
```

Kubernetes 101 - docker ps

```
1 $ kubectl get po --all-namespaces  
2 NAMESPACE      NAME                READY   STATUS    RESTARTS   AGE  
3 kube-system    helm-install-traefik-crd-7kcct  0/1     Completed  0          8d  
4 kube-system    helm-install-traefik-cljb2    0/1     Completed  0          8d  
5 kube-system    metrics-server-86ccb8457f-t54g4  1/1     Running   2          8d  
6 kube-system    local-path-provisioner-5ff76fc89d-sbwvg 1/1     Running   2          8d  
7 kube-system    svclb-traefik-krzqs    2/2     Running   2          8d  
8 kube-system    coredns-7448499f4d-gpksg   1/1     Running   1          8d  
9 kube-system    traefik-97b44b794-jj86m    1/1     Running   1          8d  
10 default       infinite-2loops   2/2     Running   0          57m
```

Kubernetes 101 - namespaces



```
1 $ kubectl get ns
2 NAME                  STATUS   AGE
3 kube-system            Active   8d
4 default                Active   8d
5 kube-public             Active   8d
6 kube-node-lease         Active   8d
```

Comment créer des ressources ?

Créons des pods, et des créateurs de pods



Kubernetes

- Ressources k8s déclaratives
- YAML

Kubernetes

- Ressources k8s déclaratives
- YAML
- 3 informations obligatoires

Kubernetes

- Ressources k8s déclaratives
- YAML
- 3 informations obligatoires
 - ◆ apiVersion

Kubernetes

- Ressources k8s déclaratives
- YAML
- 3 informations obligatoires
 - ◆ apiVersion
 - v1
 - v1beta1
 - rbac.authorization.k8s.io/v1
 - k3s.cattle.io/v1

Kubernetes

- Ressources k8s déclaratives
- YAML
- 3 informations obligatoires
 - ◆ apiVersion
 - ◆ kind

Kubernetes

- Ressources k8s déclaratives
- YAML
- 3 informations obligatoires
 - ◆ apiVersion
 - ◆ kind
 - ◆ metadata.name

Kubernetes

- Ressources k8s déclaratives
- YAML
- 3 informations obligatoires
 - ◆ apiVersion
 - ◆ kind
 - ◆ metadata.name
- 1 quasi-obligatoire

Kubernetes

- Ressources k8s déclaratives
- YAML
- 3 informations obligatoires
 - ◆ apiVersion
 - ◆ kind
 - ◆ metadata.name
- 1 quasi-obligatoire
 - ◆ spec

Kubernetes - kind: pod

→ Créons un pod manuellement

Kubernetes - kind: pod

→ Création manuelle

Kubernetes - kind: pod

- Création manuelle
- Cas très basique

Kubernetes - kind: pod

- Création manuelle
- Cas très basique
 - ◆ API complète

Kubernetes - kind: pod

- Création manuelle
- Cas très basique
- ◆ API complète
- Pas de scaling automatique

Kubernetes - kind: pod

- Création manuelle
- Cas très basique
 - ◆ API complète
- Pas de scaling automatique
 - ◆ Pas de déploiement en HA

Kubernetes - kind: pod

- Création manuelle
- Cas très basique
 - ◆ API complète
- Pas de scaling automatique
 - ◆ Pas de déploiement en HA
 - ◆ Pas de loadbalancing
 - ◆ ...

Kubernetes - kind: pod

→ 2 containers



```
1 $ cat pod-2containers.yml
2 apiVersion: v1
3 kind: Pod
4 metadata:
5   name: infinite-2loops
6 spec:
7   containers:
8     - image: pause
9       imagePullPolicy: Never
10      name: container1
11     - image: pause
12       imagePullPolicy: Never
13      name: container2
```

Kubernetes - kind: pod

- 2 containers
- Besoin de préciser le container pour certaines opérations



```
1 $ k logs infinite-2loops
2 error: a container name must be specified for pod infinite-2loops,
choose one of: [container1 container2]
3 $ k logs -c container1 infinite-2loops
```



```
1 $ cat pod-2containers.yml
2 apiVersion: v1
3 kind: Pod
4 metadata:
5   name: infinite-2loops
6 spec:
7   containers:
8     - image: pause
9       imagePullPolicy: Never
10      name: container1
11     - image: pause
12       imagePullPolicy: Never
13      name: container2
```

Kubernetes - kind: pod

→ Prenons un exemple un peu plus complexe

```
[zarak@chlorine ~/misc/kubernetes/basic-service
└$ cd ..
[zarak@chlorine ~/misc/kubernetes
└$ ls
basic-refresh basic-service pause pod-2containers.yml pod-2services.yml pod-simple.yml
[zarak@chlorine ~/misc/kubernetes
└$ cat pod-2services.yml
apiVersion: v1
kind: Pod
metadata:
  name: basic-service
spec:
  volumes:
    - name: shared-workdir
      emptyDir: {}
  containers:
    - image: basic-refresh
      imagePullPolicy: Never
      name: refresh
      env:
        - name: WORKDIR
          value: /srv
      volumeMounts:
        - name: shared-workdir
          mountPath: /srv
    - image: basic-service
      imagePullPolicy: Never
      name: service
      env:
        - name: WORKDIR
          value: /mnt
      volumeMounts:
        - name: shared-workdir
          mountPath: /mnt
[zarak@chlorine ~/misc/kubernetes
└$ k apply -f pod-2services.yml
pod/basic-service created
[zarak@chlorine ~/misc/kubernetes
└$ ]
```

Kubernetes - kind: pod

→ 2 containers dans le pod

Kubernetes - kind: pod

- 2 containers dans le pod
- Chacun leur propre FS

Kubernetes - kind: pod

- 2 containers dans le pod
- Chacun leur propre FS
 - ◆ Volume précisé à côté des containers

Kubernetes - kind: pod

- 2 containers dans le pod
- Chacun leur propre FS
 - ◆ Volume précisé à côté des containers
 - ◆ Source = emptyDir, tmpfs vide

Kubernetes - kind: pod

- 2 containers dans le pod
- Chacun leur propre FS
 - ◆ Volume précisé à côté des containers
 - ◆ Source = emptyDir, tmpfs vide
 - ◆ VolumeMounts dans chaque container pour préciser où le mount

Kubernetes - kind: pod

- 2 containers dans le pod
- Chacun leur propre FS
 - ◆ Volume précisé à côté des containers
 - ◆ Source = emptyDir, tmpfs vide
 - ◆ VolumeMounts dans chaque container pour préciser où le mount
- Ajout de variables d'environnement

Kubernetes - kind: pod

- 2 containers dans le pod
- Chacun leur propre FS
 - ◆ Volume précisé à côté des containers
 - ◆ Source = emptyDir, tmpfs vide
 - ◆ VolumeMounts dans chaque container pour préciser où le mount
- Ajout de variables d'environnement
- Beaucoup d'autres options

Kubernetes - kind: pod

- 2 containers dans le pod
- Chacun leur propre FS
 - ◆ Volume précisé à côté des containers
 - ◆ Source = emptyDir, tmpfs vide
 - ◆ VolumeMounts dans chaque container pour préciser où le mount
- Ajout de variables d'environnement
- Beaucoup d'autres options
 - ◆ Lire la doc

On doit créer nos pods manuellement ?

Bien sûr que non, on est stylé, on fait du
kube



k8s - workload controllers

- Ressource haut-niveau de gestion des pods

k8s - workload controllers

- Ressource haut-niveau de gestion des pods
- Précise quel template de pod on veut créer et gérer

k8s - workload controllers

- Ressource haut-niveau de gestion des pods
- Précise quel template de pod on veut créer et gérer
- S'occupe de les créer et contrôler leur cycle de vie

k8s - workload controllers

- Ressource haut-niveau de gestion des pods
- Précise quel template de pod on veut créer et gérer
- S'occupe de les créer et contrôler leur cycle de vie
- Plusieurs types, plusieurs utilisations :

k8s - workload controllers

- Ressource haut-niveau de gestion des pods
- Précise quel template de pod on veut créer et gérer
- S'occupe de les créer et contrôler leur cycle de vie
- Plusieurs types, plusieurs utilisations :
 - ◆ ReplicaSet
 - ◆ Deployment
 - ◆ StatefulSet
 - ◆ DaemonSet
 - ◆ Job/CronJob

Kubernetes - kind: ReplicaSet

- Prends :
 - ◆ Un template de Pod

Kubernetes - kind: ReplicaSet

- Prends :
 - ◆ Un template de Pod
 - ◆ Un sélecteur de Pods

Kubernetes - kind: ReplicaSet

- Prends :
 - ◆ Un template de Pod
 - ◆ Un sélecteur de Pods
 - ◆ Un nombre x de répliques (replica)

Kubernetes - kind: ReplicaSet

- Prends :
 - ◆ Un template de Pod
 - ◆ Un sélecteur de Pods
 - ◆ Un nombre x de répliques (replica)
- Crée à partir du template des Pods jusqu'à atteindre x répliques

Kubernetes - kind: ReplicaSet

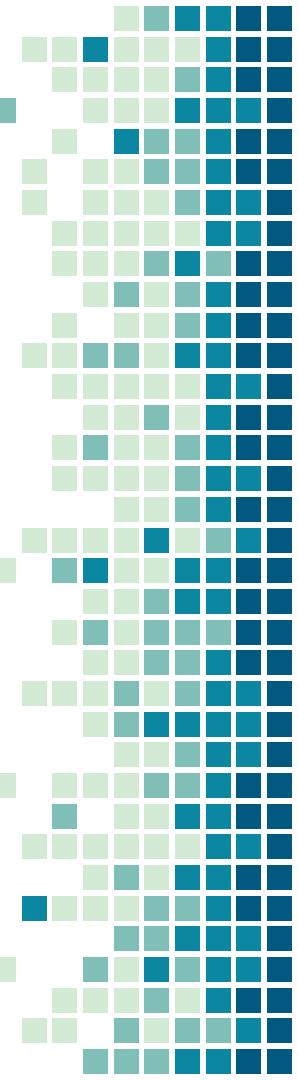
- Prends :
 - ◆ Un template de Pod
 - ◆ Un sélecteur de Pods
 - ◆ Un nombre x de répliques (replica)
- Crée à partir du template des Pods jusqu'à atteindre x répliques
- Nombre n de Pods actuels déterminé par le sélecteur

Kubernetes - kind: ReplicaSet

- Prends :
 - ◆ Un template de Pod
 - ◆ Un sélecteur de Pods
 - ◆ Un nombre x de répliques (replica)
- Crée à partir du template des Pods jusqu'à atteindre x répliques
- Nombre n de Pods actuels déterminé par le sélecteur
- Si $n > x$, termine des pods

Kubernetes - kind: ReplicaSet

- Prends :
 - ◆ Un template de Pod
 - ◆ Un sélecteur de Pods
 - ◆ Un nombre x de répliques (replica)
- Crée à partir du template des Pods jusqu'à atteindre x répliques
- Nombre n de Pods actuels déterminé par le sélecteur
- Si $n > x$, termine des pods
- Si $x > n$, crée des pods



Kubernetes - kind: ReplicaSet

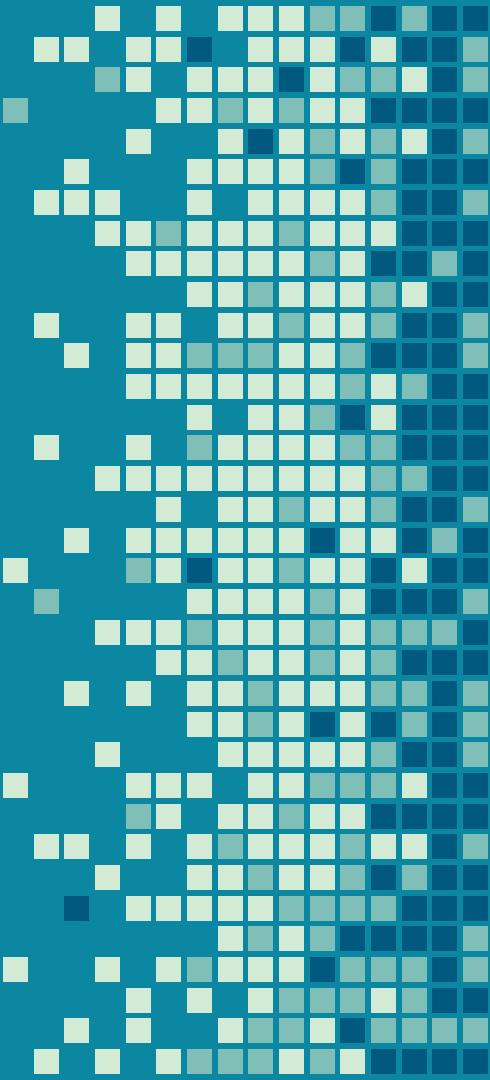
```
zarak@chlorine ~/misc/kubernetes
└─$ k get po
No resources found in default namespace.
zarak@chlorine ~/misc/kubernetes
└─$ k apply -f replicaset.yml
replicaset.apps/webservice created
zarak@chlorine ~/misc/kubernetes
└─$
```

Kubernetes - kind: ReplicaSet

```
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: "2021-12-05T16:32:06Z"
  generateName: webservice-
  labels:
    app: webservice
    name: webservice-85sqf
    namespace: default
  ownerReferences:
    - apiVersion: apps/v1
      blockOwnerDeletion: true
      controller: true
      kind: ReplicaSet
      name: webservice
      uid: ebd2c6fe-76d6-43bc-902c-0c27a756eb7f
  resourceVersion: "247383"
  uid: 0721a489-c23b-487d-a037-d984290a2192
spec:
  containers:
    - image: webservice
      imagePullPolicy: Never
      name: webservice
      resources: {}
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
      volumeMounts:
        - mountPath: /var/run/secrets/kubernetes.io/serviceaccount
          name: kube-api-access-jttv8
          readOnly: true
      dnsPolicy: ClusterFirst
      enableServiceLinks: true
      nodeName: chlorine
      preemptionPolicy: PreemptLowerPriority
      priority: 0
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      serviceAccount: default
      serviceAccountName: default
      terminationGracePeriodSeconds: 30
      tolerations:
        - effect: NoExecute
          key: node.kubernetes.io/not-ready
          operator: Exists
          tolerationSeconds: 300
        - effect: NoExecute
          key: node.kubernetes.io/unreachable
          operator: Exists
          tolerationSeconds: 300
      volumes:
        - name: kube-api-access-jttv8
          projected:
            defaultToken: true
            readOnly: true
            sources:
              - NORMAL webservice-pod.vml
              - "webservice-pod.yaml" 106L 27748
```

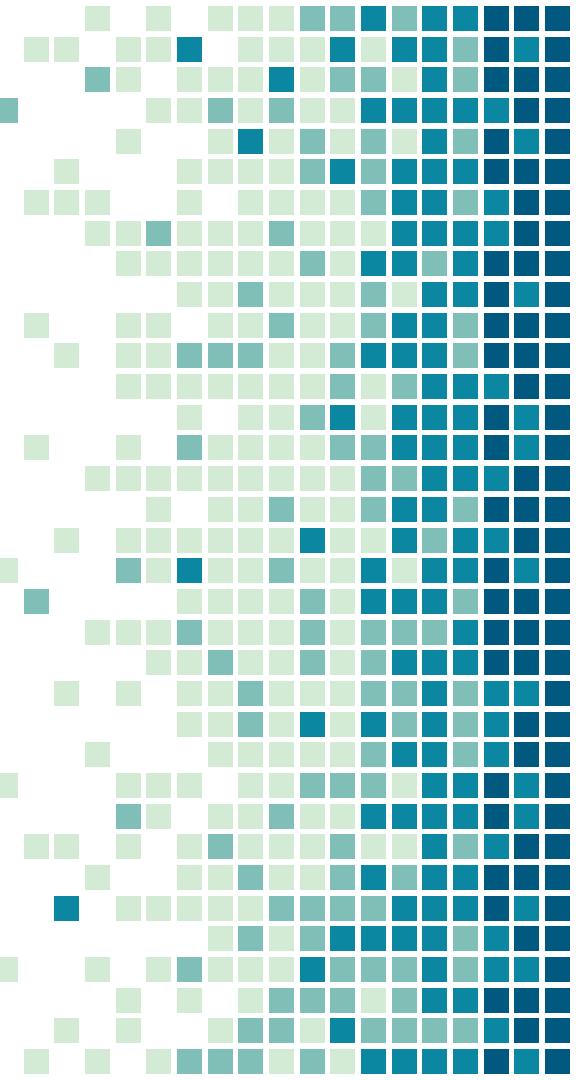
“ Comment accéder aux pods automatiquement ?

On va pas récupérer leur IP à chaque pour la filer au client ...



Notion de service

Une abstraction par dessus qui fait plaisir



k8s - Exposer un service

- Comment exposer un service avec docker ?
 - ◆ `docker run -p 0.0.0.0:80:9200`

k8s - Exposer un service

- Comment exposer un service avec docker ?
 - ◆ docker run -p 0.0.0.0:80:9200
 - ◆ docker run -p 127.0.0.1:9200:9200 + reverse proxy

k8s - Exposer un service

- Comment exposer un service avec docker ?
 - ◆ docker run -p 0.0.0.0:80:9200
 - ◆ docker run -p 127.0.0.1:9200:9200 + reverse proxy
 - ◆ docker network --create toto +
docker run --network toto <service> +
docker run --network toto -p 0.0.0.0:80:80
-e redirect_to=<service> <reverse proxy>

k8s - Exposer un service

- Comment exposer un service avec docker qui a plusieurs backends ?

k8s - Exposer un service

- Comment exposer un service avec docker qui a plusieurs backends ?

◆ ~~docker run -p 0.0.0.0:80:9200 +~~
~~docker run -p 0.0.0.0:80:9200~~

k8s - Exposer un service

- Comment exposer un service avec docker qui a plusieurs backends ?
 - ◆ ~~docker run -p 0.0.0.0:80:9200 +~~
~~docker run -p 0.0.0.0:80:9200~~
 - ◆ docker run -p 127.0.0.1:9200:9200 +
docker run -p 127.0.0.1:9201:9200 +
reverse proxy

k8s - Exposer un service

- Comment exposer un service avec docker qui a plusieurs backends ?
 - ◆ ~~docker run -p 0.0.0.0:80:9200 +~~
~~docker run -p 0.0.0.0:80:9200~~
 - ◆ docker run -p 127.0.0.1:9200:9200 +
docker run -p 127.0.0.1:9201:9200 +
reverse proxy
- Comment gérer la scalabilité ?

k8s - Exposer un service

- Comment exposer un service avec docker qui a plusieurs backends ?
 - ◆ ~~docker run -p 0.0.0.0:80:9200 +~~
~~docker run -p 0.0.0.0:80:9200~~
 - ◆ docker run -p 127.0.0.1:9200:9200 +
docker run -p 127.0.0.1:9201:9200 +
reverse proxy
- Comment gérer la scalabilité ?
 - ◆ ↗_(ツ)_/↖

k8s - Exposer un service

- Avec k8s ?
- Objet service

k8s - Exposer un service

- Avec k8s ?
- Objet service
- Abstraction du/des backend(s) pour offrir un entrypoint unique

k8s - Exposer un service

- Avec k8s ?
- Objet service
- Abstraction du/des backend(s) pour offrir un endpoint unique
 - ◆ 1, 3, 1000 backends ? 1 endpoint

k8s - Exposer un service

- Avec k8s ?
- Objet service
- Abstraction du/des backend(s) pour offrir un entrypoint unique
 - ◆ 1, 3, 1000 backends ? 1 endpoint
- "Équivalent" d'un meilleur port binding de docker

k8s - Exposer un service

→ kind: Service

k8s - Exposer un service

- kind: Service
- spec.selector

k8s - Exposer un service

- kind: Service
- spec.selector
- ports:
 - ◆ source->destination

k8s - Exposer un service

- kind: Service
- spec.selector
- ports:
 - ◆ source->destination
- Loadbalance entre les pods matchés

k8s - Exposer un service

- kind: Service
- spec.selector
- ports:
 - ◆ source->destination
- Loadbalance entre les pods matchés
 - ◆ Loadbalancing = random entre Pods disponibles

k8s - Exposer un service

- kind: Service
- spec.selector
- ports:
 - ◆ source->destination
- Loadbalance entre les pods matchés
 - ◆ Loadbalancing = random entre Pods disponibles
- Résultat exposé ?

k8s - Exposer un service

- kind: Service
- spec.selector
- ports:
 - ◆ source->destination
- Loadbalance entre les pods matchés
 - ◆ Loadbalancing = random entre Pods disponibles
- Résultat exposé ?
 - ◆ Dépend du type

k8s - Exposer un service

- kind: Service
- spec.selector
- ports:
 - ◆ source->destination
- Loadbalance entre les pods matchés
 - ◆ Loadbalancing = random entre Pods disponibles
- Résultat exposé ?
 - ◆ Dépend du type
 - ◆ Par défaut clusterIP

k8s - Exposer un service

```
zarak@chlorine ~/misc/kubernetes
└─$ ls
basic-refresh pause          pod-2services.yml  replicaset.yml  webservice
basic-service pod-2containers.yml  pod-simple.yml    service.yml   webservice-pod.yml
└─$ vim service.yml
zarak@chlorine ~/misc/kubernetes
└─$ k apply -f service.yml
service/webservice created
└─$ k get service
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes ClusterIP  10.43.0.1    <none>        443/TCP   24d
webservice ClusterIP  10.43.69.232  <none>        80/TCP    2s
└─$ curl 10.43.69.232
root@zarak@chlorine ~/misc/kubernetes
└─$ curl 10.43.69.232/whoami
```

k8s - Exposer un service

- kind: Service
- spec.type: NodePort
- Port mappé sur tous les noeuds vers le service
- exemple

k8s - Exposer un service

- kind: Service
- spec.type: LoadBalancer
- Demande au loadbalancer externe (du cloud provider) de fournir un entrypoint
- Pour exposer publiquement un service

k8s - Exposer un service

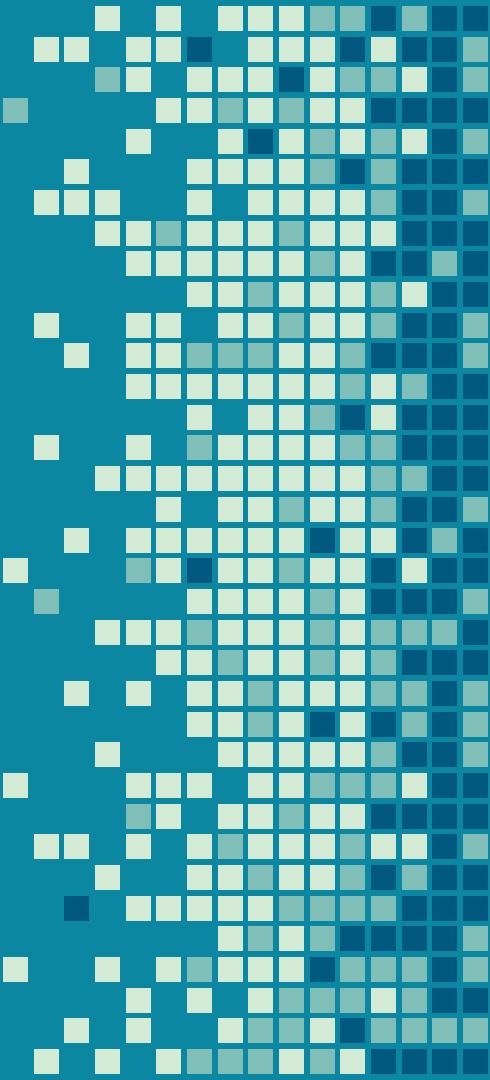
- kind: Service
- spec.type: LoadBalancer
- Demande au loadbalancer externe (du cloud provider) de fournir un entrypoint
- Pour exposer publiquement un service
- ClusterIP = dans le cluster, local

k8s - Exposer un service

- kind: Service
- spec.type: LoadBalancer
- Demande au loadbalancer externe (du cloud provider) de fournir un entrypoint
- Pour exposer publiquement un service
- ClusterIP = dans le cluster, local
- LoadBalancer = accessible hors cluster, public

“ *Oui mais comment accéder aux pods automatiquement via le service?*

On va pas récupérer la clusterIP à chaque fois ...



k8s - Exposer un service

- kind: Service
- LoadBalancer = accessible hors cluster, public

k8s - Exposer un service

- kind: Service
- LoadBalancer = accessible hors cluster, public
 - ◆ Dépend du cloudprovider

k8s - Exposer un service

- kind: Service
- LoadBalancer = accessible hors cluster, public
 - ◆ Dépend du cloudprovider
 - ◆ Possibilité d'avoir une IP fixe

k8s - Exposer un service

- kind: Service
- LoadBalancer = accessible hors cluster, public
 - ◆ Dépend du cloudprovider
 - ◆ Possibilité d'avoir une IP fixe
 - ◆ ... et des DNS

k8s - Exposer un service

- kind: Service
- LoadBalancer = accessible hors cluster, public
 - ◆ Dépend du cloudprovider
 - ◆ Possibilité d'avoir une IP fixe
 - ◆ ... et des DNS
- clusterIP = dans le cluster, local

k8s - Exposer un service

- kind: Service
- LoadBalancer = accessible hors cluster, public
 - ◆ Dépend du cloudprovider
 - ◆ Possibilité d'avoir une IP fixe
 - ◆ ... et des DNS
- clusterIP = dans le cluster, local
 - ◆ Si architecture micro-services, comment trouver un autre micro service ?

k8s - Exposer un service

- kind: Service
- LoadBalancer = accessible hors cluster, public
 - ◆ Dépend du cloudprovider
 - ◆ Possibilité d'avoir une IP fixe
 - ◆ ... et des DNS
- clusterIP = dans le cluster, local
 - ◆ Si architecture micro-services, comment trouver un autre micro service ?
 - ◆ Dans le cluster, local = DNS interne

k8s - DNS

- Kube-dns
- Allocation automatique d'IP pour les pods et services

k8s - DNS

- Kube-dns
- Allocation automatique d'IP pour les pods et services
 - ◆ Pourquoi pas d'enregistrements DNS ?

k8s - DNS

- Kube-dns
- Allocation automatique d'IP pour les pods et services
 - ◆ Pourquoi pas d'enregistrements DNS ?

```
1 $ k get service --all-namespaces
2 NAMESPACE   NAME      TYPE        CLUSTER-IP    EXTERNAL-IP  PORT(S)          AGE
3 default     kubernetes ClusterIP  10.43.0.1    <none>       443/TCP         24d
4 kube-system kube-dns   ClusterIP  10.43.0.10   <none>       53/UDP,53/TCP,9153/TCP 24d
5 kube-system metrics-server ClusterIP  10.43.99.215 <none>       443/TCP         24d
6 kube-system traefik    LoadBalancer 10.43.168.43  192.168.1.18  80:31123/TCP,443:31573/TCP 24d
7 default     webservice LoadBalancer 10.43.69.232 <pending>    80:30000/TCP    60m
```

k8s

—
DNS

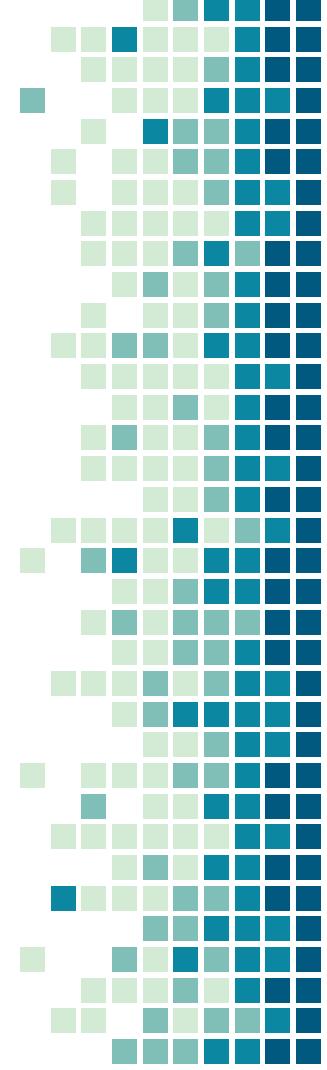
```
1 $ k get svc -o wide
2 NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE      SELECTOR
3 kubernetes  ClusterIP  10.43.0.1    <none>        443/TCP    29d      <none>
4 webservice ClusterIP  10.43.128.18  <none>        80/TCP     9s       app=webservice
5 $ dig webservice.default.svc.cluster.local @10.43.0.10
6
7 ; <>>> DiG 9.16.23 <>>> webservice.default.svc.cluster.local @10.43.0.10
8 ;; global options: +cmd
9 ;; Got answer:
10;; WARNING: .local is reserved for Multicast DNS
11;; You are currently testing what happens when an mDNS query is leaked to DNS
12;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10832
13;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
14;; WARNING: recursion requested but not available
15
16;; OPT PSEUDOSECTION:
17; EDNS: version: 0, flags:; udp: 4096
18; COOKIE: 96045e0b4a294191 (echoed)
19;; QUESTION SECTION:
20;webservice.default.svc.cluster.local. IN A
21
22;; ANSWER SECTION:
23 webservice.default.svc.cluster.local. 5 IN A      10.43.128.18
24
25;; Query time: 3 msec
26;; SERVER: 10.43.0.10#53(10.43.0.10)
27;; WHEN: Sat Dec 11 17:34:59 CET 2021
28;; MSG SIZE  rcvd: 129
29
```



k8s

DNS

```
1 $ k get svc
2 NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
3 kubernetes  ClusterIP  10.43.0.1    <none>        443/TCP     29d
4 webservice ClusterIP  10.43.128.18  <none>        80/TCP      2m29s
5 $ k get po
6 NAME      READY  STATUS      RESTARTS      AGE
7 webservice-85sqf  1/1   Running    3           6d
8 webservice-bwb6d  1/1   Running    3           5d22h
9 webservice-24w2f  1/1   Running    3           6d
10 infinite-loop  1/1   Running    0           75s
11 $ k exec -it infinite-loop -- bash
12 bash-5.1$ curl webservice/whoami && echo
13 webservice-24w2f
14 bash-5.1$ curl webservice/whoami && echo
15 webservice-85sqf
16 bash-5.1$ exit
17 $ dig webservice @10.43.0.10
18
19 ; <>>> DiG 9.16.23 <>>> webservice @10.43.0.10
20 ; global options: +cmd
21 ; Got answer:
22 ; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 2660
23 ; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1
24
25 ;; OPT PSEUDOSECTION:
26 ; EDNS: version: 0, flags:; udp: 4096
27 ; COOKIE: 2ff146690e27900c (echoed)
28 ;; QUESTION SECTION:
29 ;webservice.          IN  A
30
31 ;; AUTHORITY SECTION:
32 .            30  IN  SOA a.root-servers.net. nstld.verisign-grs.com. 2021121100 1800
33             900 604800 86400
34
35 ;; Query time: 23 msec
36 ;; SERVER: 10.43.0.10#53(10.43.0.10)
37 ;; WHEN: Sat Dec 11 17:37:44 CET 2021
38 ;; MSG SIZE  rcvd: 126
```



k8s - Exposer un service HTTP

→ Service = port plus IP

k8s - Exposer un service HTTP

- Service = port plus IP
- Exposition publique = une IP externe

k8s - Exposer un service HTTP

- Service = port plus IP
- Exposition publique = une IP externe
- Si plusieurs services HTTP (port 80/443), plusieurs IP externes

k8s - Exposer un service HTTP

- Service = port plus IP
- Exposition publique = une IP externe
- Si plusieurs services HTTP (port 80/443), plusieurs IP externes
- Solutions possibles :

k8s - Exposer un service HTTP

- Service = port plus IP
- Exposition publique = une IP externe
- Si plusieurs services HTTP (port 80/443), plusieurs IP externes
- Solutions possibles :
 - ◆ Gros reverse proxy interne

k8s - Exposer un service HTTP

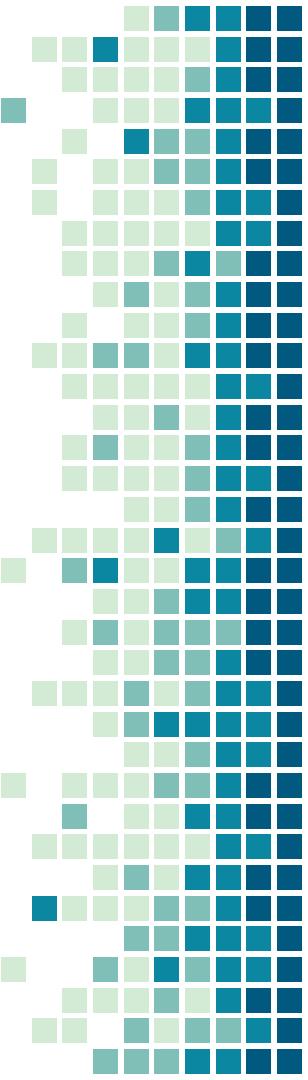
- Service = port plus IP
- Exposition publique = une IP externe
- Si plusieurs services HTTP (port 80/443), plusieurs IP externes
- Solutions possibles :
 - ◆ Gros reverse proxy interne
 - Replicaset + service + bricoles

k8s - Exposer un service HTTP

- Service = port plus IP
- Exposition publique = une IP externe
- Si plusieurs services HTTP (port 80/443), plusieurs IP externes
- Solutions possibles :
 - ◆ Gros reverse proxy interne
 - Replicaset + service + bricoles
 - Reload de la config générale à chaque modif

k8s - Exposer un service HTTP

- Service = port plus IP
- Exposition publique = une IP externe
- Si plusieurs services HTTP (port 80/443), plusieurs IP externes
- Solutions possibles :
 - ◆ Gros reverse proxy interne
 - Replicaset + service + bricoles
 - Reload de la config générale à chaque modif
 - ◆ La même chose mais intégré Kubernetes : l'ingress



k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s



k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s
- Signifie entrée (opposition à egress)

k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s
- Signifie entrée (opposition à egress)
- Reverse proxy pour gérer la répartition du trafic entrant

k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s
- Signifie entrée (opposition à egress)
- Reverse proxy pour gérer la répartition du trafic entrant
 - ◆ Le service LoadBalancer géré par l'ingress controller

k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s
- Signifie entrée (opposition à egress)
- Reverse proxy pour gérer la répartition du trafic entrant
 - ◆ Le service LoadBalancer géré par l'ingress controller
 - ◆ Création de ressources de type "Ingress" par service à exposer

k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s
- Signifie entrée (opposition à egress)
- Reverse proxy pour gérer la répartition du trafic entrant
 - ◆ Le service LoadBalancer géré par l'ingress controller
 - ◆ Création de ressources de type "Ingress" par service à exposer
 - ◆ Pas de contrôle direct du reverse proxy, abstraction

k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s
- Signifie entrée (opposition à egress)
- Reverse proxy pour gérer la répartition du trafic entrant
 - ◆ Le service LoadBalancer géré par l'ingress controller
 - ◆ Création de ressources de type "Ingress" par service à exposer
 - ◆ Pas de contrôle direct du reverse proxy, abstraction
 - ◆ Répartition du trafic via le contenu

k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s
- Signifie entrée (opposition à egress)
- Reverse proxy pour gérer la répartition du trafic entrant
 - ◆ Le service LoadBalancer géré par l'ingress controller
 - ◆ Création de ressources de type "Ingress" par service à exposer
 - ◆ Pas de contrôle direct du reverse proxy, abstraction
 - ◆ Répartition du trafic via le contenu
 - HTTP : header Host + request line

k8s - HTTP - Ingress

- Pas l'ancêtre de Pokémon GO
- Objet k8s
- Signifie entrée (opposition à egress)
- Reverse proxy pour gérer la répartition du trafic entrant
 - ◆ Le service LoadBalancer géré par l'ingress controller
 - ◆ Création de ressources de type "Ingress" par service à exposer
 - ◆ Pas de contrôle direct du reverse proxy, abstraction
 - ◆ Répartition du trafic via le contenu
 - HTTP : header Host + request line
 - HTTPS : SNI (+ request line si terminaison SSL/TLS)

k8s - HTTP - Ingress

- Déployé par l'ops dans le cluster = pas la responsabilité du dev de gérer l'ingress controller

k8s - HTTP - Ingress

- Déployé par l'ops dans le cluster = pas la responsabilité du dev de gérer l'ingress controller
- Le dev gère les ingress

k8s - HTTP - Ingress

- Déployé par l'ops dans le cluster = pas la responsabilité du dev de gérer l'ingress controller
- Le dev gère les ingress
- Pas à se soucier du controller sous-jacent = abstraction

k8s - HTTP - Ingress

- Déployé par l'ops dans le cluster = pas la responsabilité du dev de gérer l'ingress controller
- Le dev gère les ingress
- Pas à se soucier du controller sous-jacent = abstraction
- Pas besoin d'écrire une configuration spécifique nginx/haproxy/traefik/...

k8s - HTTP - Ingress

- Déployé par l'ops dans le cluster = pas la responsabilité du dev de gérer l'ingress controller
- Le dev gère les ingress
- Pas à se soucier du controller sous-jacent = abstraction
- Pas besoin d'écrire une configuration spécifique nginx/haproxy/traefik/...
- Ingress controller déployé de base dans k3s/minikube

k8s - HTTP - Ingress

- Déployé par l'ops dans le cluster = pas la responsabilité du dev de gérer l'ingress controller
- Le dev gère les ingress
- Pas à se soucier du controller sous-jacent = abstraction
- Pas besoin d'écrire une configuration spécifique nginx/haproxy/traefik/...
- Ingress controller déployé de base dans k3s/minikube
- Exposons proprement notre webservice :

k8s - HTTP - Ingress

- Déployé par l'ops dans le cluster = pas la responsabilité du dev de gérer l'ingress controller
- Le dev gère les ingress
- Pas à se soucier du controller sous-jacent = abstraction
- Pas besoin d'écrire une configuration spécifique nginx/haproxy/traefik/...
- Ingress controller déployé de base dans k3s/minikube
- Exposons proprement notre webservice :
 - ◆ Création d'ingress

k8s - HTTP - Ingress

- Possibilité de contrôler :
 - ◆ L'hostname
 - ◆ Les routes
 - ◆ Les ports
 - ◆ ...

k8s - HTTP - Ingress

→ Possibilité de contrôler :

- ◆ L'hostname
- ◆ Les routes
- ◆ Les ports
- ◆ ...



```
1 $ k get ingress --all-namespaces
2 NAMESPACE     NAME          CLASS      HOSTS      ADDRESS        PORTS      AGE
3 default       webservice   <none>    *           192.168.1.18   80          112s
```

Comment savoir à qui dispatch ?

Les probes, savoir qui est prêt



k8s - Probes - problématique

- Un service load balance entre plusieurs pods

k8s - Probes - problématique

- Un service load balance entre plusieurs pods
- Un pod peut prendre beaucoup de temps à être prêt

k8s - Probes - problématique

- Un service load balance entre plusieurs pods
- Un pod peut prendre beaucoup de temps à être prêt
 - ◆ Lancement du pod, chargement en mémoire de données, connexions aux DB à établir, ...

k8s - Probes - problématique

- Un service load balance entre plusieurs pods
- Un pod peut prendre beaucoup de temps à être prêt
 - ◆ Lancement du pod, chargement en mémoire de données, connexions aux DB à établir, ...
- Comment ne pas envoyer du trafic à un pod en train de boot ?

k8s - Probes - problématique

- Un service load balance entre plusieurs pods
- Un pod peut prendre beaucoup de temps à être prêt
 - ◆ Lancement du pod, chargement en mémoire de données, connexions aux DB à établir, ...
- Comment ne pas envoyer du trafic à un pod en train de boot ?
- Savoir quand il est prêt

k8s - Probes - problématique

- Un service load balance entre plusieurs pods
- Un pod peut prendre beaucoup de temps à être prêt
 - ◆ Lancement du pod, chargement en mémoire de données, connexions aux DB à établir, ...
- Comment ne pas envoyer du trafic à un pod en train de boot ?
- Savoir quand il est prêt
- Le signaler

k8s - Probes - problématique

- Un service load balance entre plusieurs pods
- Un pod peut prendre beaucoup de temps à être prêt
 - ◆ Lancement du pod, chargement en mémoire de données, connexions aux DB à établir, ...
- Comment ne pas envoyer du trafic à un pod en train de boot ?
- Savoir quand il est prêt
- Le signaler
- Comment savoir ?

k8s - Probes - problématique

- Un service load balance entre plusieurs pods
- Un pod peut prendre beaucoup de temps à être prêt
 - ◆ Lancement du pod, chargement en mémoire de données, connexions aux DB à établir, ...
- Comment ne pas envoyer du trafic à un pod en train de boot ?
- Savoir quand il est prêt
- Le signaler
- Comment savoir ?
 - ◆ En testant

k8s - Probes

- 3 types de Probes :
 - ◆ StartupProbe

k8s - Probes

- 3 types de Probes :
 - ◆ StartupProbe
 - ◆ LivenessProbe

k8s - Probes

- 3 types de Probes :
 - ◆ StartupProbe
 - ◆ LivenessProbe
 - ◆ ReadinessProbe

k8s - Probes

- 3 types de Probes :
 - ◆ StartupProbe
 - ◆ LivenessProbe
 - ◆ ReadinessProbe
- Définie dans la spec d'un Pod

k8s - Probes

- 3 types de Probes :
 - ◆ StartupProbe
 - ◆ LivenessProbe
 - ◆ ReadinessProbe
- Définie dans la spec d'un Pod
- Effectuent un test à interval régulier

k8s - Probes

- 3 types de Probes :
 - ◆ StartupProbe
 - ◆ LivenessProbe
 - ◆ ReadinessProbe
- Définie dans la spec d'un Pod
- Effectuent un test à interval régulier
- Gérées par Kubernetes

k8s - Probes

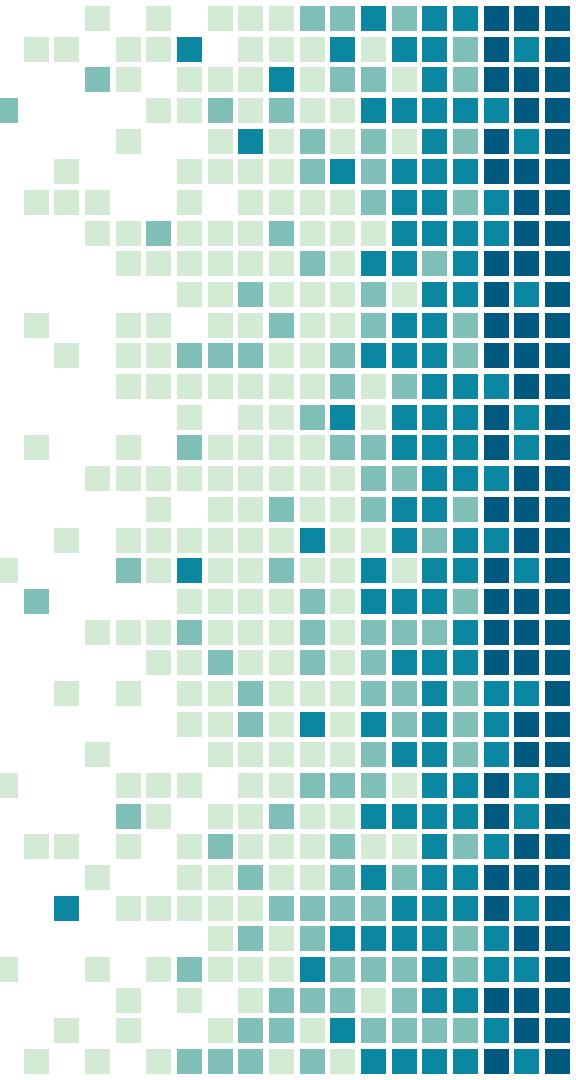
- 3 types de Probes :
 - ◆ StartupProbe
 - ◆ LivenessProbe
 - ◆ ReadinessProbe
- Définie dans la spec d'un Pod
- Effectuent un test à interval régulier
- Gérées par Kubernetes
 - ◆ Gestion du trafic voire restart d'une app deadlock

k8s - Probes

- [Exemple de StartupProbe](#)
- [Exemple de ReadinessProbe](#)
- [Exemple de LivenessProbe](#)

Les autres workload controllers

Soyons plus intelligents que le replicaset



k8s - workload controllers

- Plusieurs types, plusieurs utilisations :
 - ◆ ReplicaSet
 - ◆ Deployment
 - ◆ StatefulSet
 - ◆ DaemonSet
 - ◆ Job/CronJob

k8s - job

- Comme un Pod
 - ◆ kind: Job évidemment

k8s - job

- Comme un Pod
 - ◆ kind: Job évidemment
- Le container ne doit pas être un daemon

k8s - job

- Comme un Pod
 - ◆ kind: Job évidemment
- Le container ne doit pas être un daemon
- C'est un job : il fait son job et se casse

k8s - job

- Comme un Pod
 - ◆ kind: Job évidemment
- Le container ne doit pas être un daemon
- C'est un job : il fait son job et se casse
- Kube relance le job si le container exitcode != 0

k8s - CronJob

- Un controller qui lance des Job selon des critères de périodicité

k8s - CronJob

- Un controller qui lance des Job selon des critères de périodicité
- ... comme cron

k8s - CronJob

- Un controller qui lance des Job selon des critères de périodicité
- ... comme cron
- Rien de plus à dire

k8s - DaemonSet

- Un controller qui lance un Pod sur chacun des noeuds du cluster k8s

k8s - DaemonSet

- Un controller qui lance un Pod sur chacun des noeuds du cluster k8s
- On rajoute un noeud : run d'un Pod

k8s - DaemonSet

- Un controller qui lance un Pod sur chacun des noeuds du cluster k8s
- On rajoute un noeud : run d'un Pod
- On supprime un noeud : stop le Pod

k8s - DaemonSet

- Un controller qui lance un Pod sur chacun des noeuds du cluster k8s
- On rajoute un noeud : run d'un Pod
- On supprime un noeud : stop le Pod
- Utile pour :
 - ◆ Le monitoring

k8s - DaemonSet

- Un controller qui lance un Pod sur chacun des noeuds du cluster k8s
- On rajoute un noeud : run d'un Pod
- On supprime un noeud : stop le Pod
- Utile pour :
 - ◆ Le monitoring
 - ◆ Le stockage local

k8s - DaemonSet

- Un controller qui lance un Pod sur chacun des noeuds du cluster k8s
- On rajoute un noeud : run d'un Pod
- On supprime un noeud : stop le Pod
- Utile pour :
 - ◆ Le monitoring
 - ◆ Le stockage local
 - ◆ La collection de logs

k8s - Deployment

- Un controller plus intelligent que ReplicaSet

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods
- Modifie le(s)s Replicaset(s) pour modifier les pods

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods
- Modifie le(s)s Replicaset(s) pour modifier les pods
- Exemple

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods
- Modifie le(s)s Replicaset(s) pour modifier les pods
- Exemple
- Gère les rolling release par défaut :

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods
- Modifie le(s)s Replicaset(s) pour modifier les pods
- Exemple
- Gère les rolling release par défaut :
 - ◆ Modification des Pods étape par étape

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods
- Modifie le(s)s Replicaset(s) pour modifier les pods
- Exemple
- Gère les rolling release par défaut :
 - ◆ Modification des Pods étape par étape
 - ◆ Démarrage d'un nouveau Pod

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods
- Modifie le(s)s Replicaset(s) pour modifier les pods
- Exemple
- Gère les rolling release par défaut :
 - ◆ Modification des Pods étape par étape
 - ◆ Démarrage d'un nouveau Pod
 - ◆ Arrêt d'un ancien

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods
- Modifie le(s)s Replicaset(s) pour modifier les pods
- Exemple
- Gère les rolling release par défaut :
 - ◆ Modification des Pods étape par étape
 - ◆ Démarrage d'un nouveau Pod
 - ◆ Arrêt d'un ancien
 - ◆ Etc

k8s - Deployment

- Un controller plus intelligent que ReplicaSet
- Utilise des Replicaset pour générer les pods
- Modifie le(s)s Replicaset(s) pour modifier les pods
- Exemple
- Gère les rolling release par défaut :
 - ◆ Modification des Pods étape par étape
 - ◆ Démarrage d'un nouveau Pod
 - ◆ Arrêt d'un ancien
 - ◆ Etc
 - ◆ Exemple

k8s - Deployment

→ L'ancien rs non supprimé

k8s - Deployment

- L'ancien rs non supprimé
 - ◆ Scale à 0 Pods

k8s - Deployment

- L'ancien rs non supprimé
 - ◆ Scale à 0 Pods
 - ◆ Toujours accessible

k8s - Deployment

- L'ancien rs non supprimé
 - ◆ Scale à 0 Pods
 - ◆ Toujours accessible
- kubectl rollout

k8s - Deployment

- L'ancien rs non supprimé
 - ◆ Scale à 0 Pods
 - ◆ Toujours accessible
- kubectl rollout
 - ◆ Voir l'avancement d'un déploiement

k8s - Deployment

- L'ancien rs non supprimé
 - ◆ Scale à 0 Pods
 - ◆ Toujours accessible
- kubectl rollout
 - ◆ Voir l'avancement d'un déploiement
 - ◆ Voir l'historique des déploiements

k8s - Deployment

- L'ancien rs non supprimé
 - ◆ Scale à 0 Pods
 - ◆ Toujours accessible
- kubectl rollout
 - ◆ Voir l'avancement d'un déploiement
 - ◆ Voir l'historique des déploiements
 - ◆ Rollback vers une ancienne version

k8s - Deployment

- L'ancien rs non supprimé
 - ◆ Scale à 0 Pods
 - ◆ Toujours accessible
- kubectl rollout
 - ◆ Voir l'avancement d'un déploiement
 - ◆ Voir l'historique des déploiements
 - ◆ Rollback vers une ancienne version
 - ◆ Pause/Reprendre un déploiement

k8s - Deployment

- L'ancien rs non supprimé
 - ◆ Scale à 0 Pods
 - ◆ Toujours accessible
- kubectl rollout
 - ◆ Voir l'avancement d'un déploiement
 - ◆ Voir l'historique des déploiements
 - ◆ Rollback vers une ancienne version
 - ◆ Pause/Reprendre un déploiement
- Ne pas utiliser de ReplicaSet directement, mais utiliser un Deployment

k8s - StatefulSets

- Un genre de Deployment, pour des données stateful

k8s - StatefulSets

- Un genre de Deployment, pour des données stateful
- Garantie sur l'unicité et l'ordre de déploiement des Pods

k8s - StatefulSets

- Un genre de Deployment, pour des données stateful
- Garantie sur l'unicité et l'ordre de déploiement des Pods
- Pas de fongibilité des Pods

k8s - StatefulSets

- Un genre de Deployment, pour des données stateful
- Garantie sur l'unicité et l'ordre de déploiement des Pods
- Pas de fongibilité des Pods
- Utile pour assurer l'association entre Pod et stockage persistant

k8s - StatefulSets

- Un genre de Deployment, pour des données stateful
- Garantie sur l'unicité et l'ordre de déploiement des Pods
- Pas de fongibilité des Pods
- Utile pour assurer l'association entre Pod et stockage persistant
- Bon exemple : elasticsearch

k8s - StatefulSets

```
1 $ k get po
2 NAME          READY  STATUS   RESTARTS  AGE
3 infinite-loop 1/1    Running  2          2d23h
4 complex-webservice-0 0/1    Running  0          23s
5 $ k get po
6 NAME          READY  STATUS   RESTARTS  AGE
7 infinite-loop 1/1    Running  2          2d23h
8 complex-webservice-0 1/1    Running  0          37s
9 complex-webservice-1 0/1    Running  0          12s
10 $ k get po
11 NAME          READY  STATUS   RESTARTS  AGE
12 infinite-loop 1/1    Running  2          2d23h
13 complex-webservice-0 1/1    Running  0          66s
14 complex-webservice-1 1/1    Running  0          41s
15 complex-webservice-2 0/1    Running  0          16s
```

Et après ?

Aller plus loin



k8s - Controller / Operator

- Un controller surveille des ressources et prends des actions

k8s - Controller / Operator

- Un controller surveille des ressources et prends des actions
 - ◆ Création d'une ressource particulière

k8s - Controller / Operator

- Un controller surveille des ressources et prends des actions
 - ◆ Création d'une ressource particulière
 - ◆ Modification/suppression

k8s - Controller / Operator

- Un controller surveille des ressources et prends des actions
 - ◆ Création d'une ressource particulière
 - ◆ Modification/suppression
 - ◆ Événement sur la cible

k8s - Controller / Operator

- Un controller surveille des ressources et prends des actions
 - ◆ Création d'une ressource particulière
 - ◆ Modification/suppression
 - ◆ Événement sur la cible
 - ◆ Exemple : Ingress controller

k8s - Controller / Operator

- Un controller surveille des ressources et prends des actions
 - ◆ Création d'une ressource particulière
 - ◆ Modification/suppression
 - ◆ Événement sur la cible
 - ◆ Exemple : Ingress controller
 - ◆ Exemple : workload controller (Deployment, ReplicaSet, ..)

k8s - Controller / Operator

- Un controller surveille des ressources et prends des actions
 - ◆ Création d'une ressource particulière
 - ◆ Modification/suppression
 - ◆ Événement sur la cible
 - ◆ Exemple : Ingress controller
 - ◆ Exemple : workload controller (Deployment, ReplicaSet, ..)
- Un Operator est un Controller qui opère avec des CRD

k8s - Controller / Operator

- Un controller surveille des ressources et prends des actions
 - ◆ Création d'une ressource particulière
 - ◆ Modification/suppression
 - ◆ Événement sur la cible
 - ◆ Exemple : Ingress controller
 - ◆ Exemple : workload controller (Deployment, ReplicaSet, ..)
- Un Operator est un Controller qui opère avec des CRD
 - ◆ Custom Resource Definition

k8s - Operator par l'exemple

- k8s gère une ressource de type secret

k8s - Operator par l'exemple

- k8s gère une ressource de type secret
 - ◆ Comme une ConfigMap, mais base64 encoded

k8s - Operator par l'exemple

- k8s gère une ressource de type secret
 - ◆ Comme une ConfigMap, mais base64 encoded
 - ◆ Sémantique

k8s - Operator par l'exemple

- k8s gère une ressource de type secret
 - ◆ Comme une ConfigMap, mais base64 encoded
 - ◆ Sémantique
 - ◆ Possible de créer une var d'env dans un Pod à partir d'un Secret



```
1 $ cat secret.yml
2 ---
3
4 apiVersion: v1
5 kind: Secret
6 metadata:
7   name: not-so-super-secret
8 type: Opaque
9 data:
10  username: YWRtaW4=
11  password: cGFzc3dvcmQ=
12 $ k apply -f secret.yml
13 secret/not-so-super-secret created
14 $ k get secrets
15 NAME          TYPE          DATA  AGE
16 default-token-k8w67  kubernetes.io/service-account-token  3      5m35s
17 not-so-super-secret  Opaque          2      7s
18 $ k get secrets not-so-super-secret -o YAML
19 apiVersion: v1
20 data:
21  password: cGFzc3dvcmQ=
22  username: YWRtaW4=
23 kind: Secret
24 metadata:
25  annotations:
26    kubectl.kubernetes.io/last-applied-configuration: |
27      {"apiVersion":"v1","data":{"password":"cGFzc3dvcmQ=","username":"YWRtaW4="},"kind":"Secret","metadata": {"annotations":{},"name":"not-so-super-secret","namespace":"default"},"type":"Opaque"}
28  creationTimestamp: "2021-12-15T10:51:54Z"
29  name: not-so-super-secret
30  namespace: default
31  resourceVersion: "986"
32  uid: a629883c-5c02-4dec-ab5d-a1e674ddd757
33 type: Opaque
34 $ k get secrets not-so-super-secret -o JSON | jq -r .data.password | base64 -d
35 password%
```

k8s - Operator par l'exemple

- k8s gère une ressource de type secret
 - ◆ Comme une ConfigMap, mais base64 encoded
 - ◆ Sémantique
 - ◆ Possible de créer une var d'env dans un Pod à partir d'un Secret
 - ◆ Possible de mount un secret comme un fichier dans un Pod

k8s - Operator par l'exemple

- k8s gère une ressource de type secret
 - ◆ Comme une ConfigMap, mais base64 encoded
 - ◆ Sémantique
 - ◆ Possible de créer une var d'env dans un Pod à partir d'un Secret
 - ◆ Possible de mount un secret comme un fichier dans un Pod
- Problème : comment stocker le secret dans git ?

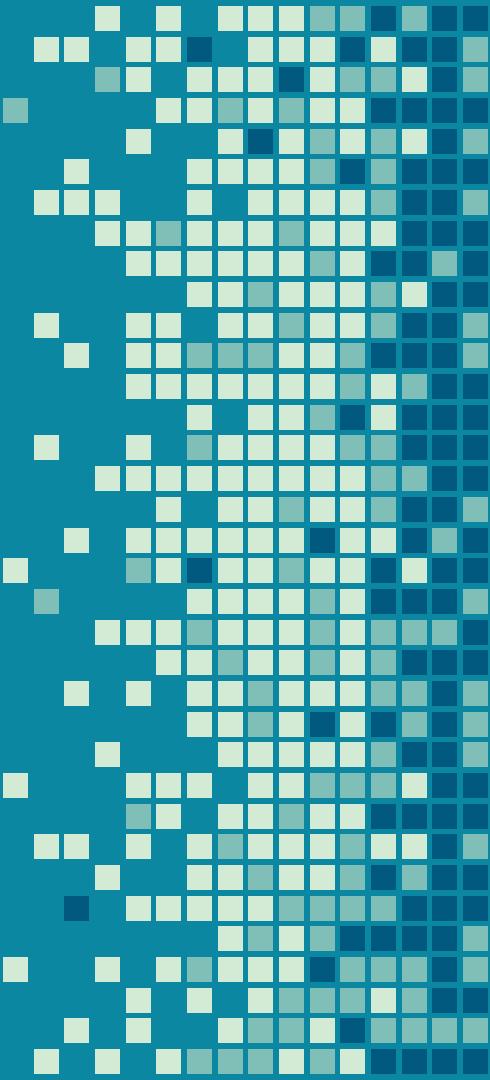
k8s - Operator par l'exemple

- k8s gère une ressource de type secret
 - ◆ Comme une ConfigMap, mais base64 encoded
 - ◆ Sémantique
 - ◆ Possible de créer une var d'env dans un Pod à partir d'un Secret
 - ◆ Possible de mount un secret comme un fichier dans un Pod
- Problème : comment stocker le secret dans git ?
 - ◆ Une solution possible : sealed-secrets

“

Problem: "I can manage all my K8s config in git, except Secrets."

Solution: Encrypt your Secret into a SealedSecret, which is safe to store - even to a public repository. The SealedSecret can be decrypted only by the controller running in the target cluster and nobody else (not even the original author) is able to obtain the original Secret from the SealedSecret.



k8s - Operator par l'exemple

- sealed-secrets projet de bitnami

k8s - Operator par l'exemple

- sealed-secrets projet de bitnami
- Fonctionne avec une CRD et un operator

k8s - Operator par l'exemple

- sealed-secrets projet de bitnami
- Fonctionne avec une CRD et un operator
- Secret chiffré avec kubeseal

k8s - Operator par l'exemple

- sealed-secrets projet de bitnami
- Fonctionne avec une CRD et un operator
- Secret chiffré avec kubeseal
- Mis dans k8s en tant que SealedSecret

k8s - Operator par l'exemple

- sealed-secrets projet de bitnami
- Fonctionne avec une CRD et un operator
- Secret chiffré avec kubeseal
- Mis dans k8s en tant que SealedSecret
 - ◆ Utilise l'API [bitnami .com/v1alpha1](https://bitnami.com/v1alpha1)

k8s -

```
1 $ kubeseal --controller-name sealed-secrets --format yaml < secret.yml > sealed-secret.yaml
2 $ cat sealed-secret.yaml
3 apiVersion: bitnami.com/v1alpha1
4 kind: SealedSecret
5 metadata:
6   creationTimestamp: null
7   name: not-so-super-secret
8   namespace: default
9 spec:
10   encryptedData:
11     password: AgCMD7Wwo7C5UjE06AfQGx5Iy1xXVftZwr8ESyb8XxCHQMz/xPTfRzTefSIKoQ1gBWeT2
     /HUKAc+52yFBm0F1ZqIhWEWSUOA0XHx7moWwYWWX0wgIq66
     /TDUoNxpLuziwHsyhEUDbp0aecl45wthpBX2-KqAU4yo+mHHjfPpsF1HhBjvTp+HYRowbCFpSzMG+Qiuhsqj+OVV8jK7IwaCioRv9uPPdm+BTVQLdApYefi
     HcMchd8Pv2zxEUvxEvgVpRA61oh3yTIYRUoASjvXCLDx6gyIeIgyz1qNeUVla67MpbeqJyMU/c8dZV2
     /yRlwKLhs1wq03rP2AXNDNy81QDXzFMRNktk7guqSYmCYqwBy06dzghYccvf4juU
     /dRKqm+Hsc1TP64kjOsVjoA3mzwticg27jwunsJfyhHMrqumSV4YzPM+YK00U4kqvt5C3SF1FNqCkiplA7LViF4wD8l
     /Jm66qkhZDJLAQAIdfJlZ4rZrevdJqlxxI+ogoDyFgkA0zNzHkQUzCQAiDcKZ04wd3zPXHd7EyK
     /qXgqJdeVwT8915gM2aTTzL5xGu8V33fCty4IuM6NtVBys8EfVp6dSa70PesNL7zyGRuoV6o91lfUwURW8rf08zZH8ICdDpokISVAgVdfdg240niDf8zLLP
     XE93ybRkoTa4cGXukU5Nwxp9Ns7P4S1byMs0673nA==

12   username: AgBqSE+7FrIpngjGm31EKhSVbjw2h/1oo9FvP2FtpQAzthUrta3NqqzY-l0NA0urCAQtwRvm47TrBQhyH/u3KzvDSr0HVrJ7abs
     /V3uM/0T9sTfLI13zvKy0QGdvZA4XAsx+QucoPUplmP5qw3teNjAx7s0CL18enNQ0m9JWVEz40quqk3G2Rh0BljqcqsrsOs+0fITFXahy4U3or1NP
     /ccT4B+ovh2gPtsLvYJ4LRV7FnPVeEKB8i3IYF4qJdC7/mJ770WDDeQmKbe7Pc5q/w2r4To2UxegJxQMJd8Ar32tE/I02kTlvh9Qn1obKa0wfDbttRQ+8fxTt
     /LMLHUGmlHIfjYmiaRqqgTnA94K+f5YfIBa5w6kYmrLP1ZqBv1fjZQXAUwDceHGbGvwydeofqj3Hbwq+PGNbK81hG7GgKnaofLRxiID+f9SuVYuNNuxNLh
     M82SRL/qkgz0pPjI0oJRAwzon7kHwPeTlQaJ0p
     /k2rbSEN6ATxUatR0d0Gol20hw85u6BnEVsvdEuIKx9dEz9Yv1M+29tBdIV1j8wMPSMgRoroNQz54yRUCihy+WqjHrnbwTTloPt76eyzNrzy6p9DSwmRVt3
     /h5fsLAHIT3e6dH/3jahstU8QRmljv070z34wocndhpVDLMMhAXD14X4SeQgdpc10/wktm+Jf4ov0aA7VzsKu7Do==

13   template:
14     data: null
15   metadata:
16     creationTimestamp: null
17   name: not-so-super-secret
18   namespace: default
19   type: Opaque
20 $ k get secrets
21 NAME          TYPE           DATA   AGE
22 default-token-k8w67  kubernetes.io/service-account-token  3      20m
23 $ k apply -f sealed-secret.yaml
24 sealedsecret.bitnami.com/not-so-super-secret created
25 $ k get sealedsecrets.bitnami.com
26 NAME          AGE
27 not-so-super-secret  5s
28 $ k get secrets
29 NAME          TYPE           DATA   AGE
30 default-token-k8w67  kubernetes.io/service-account-token  3      20m
31 not-so-super-secret  Opaque        2      8s
32 $ k -n kube-system logs sealed-secrets-7569f57679-w2f84
33 2021/12/15 11:06:58 Updating default/not-so-super-secret
34 2021/12/15 11:06:58 Event{v1.ObjectReference{Kind:"SealedSecret", Namespace:"default", Name:"not-so-super-secret",
     UID:"e1f94994-c7a1-45c5-b463-0282f1b8cbfd", APIVersion:"bitnami.com/v1alpha1", ResourceVersion:"1628", FieldPath:""}}, type: 'Normal' reason: 'Unsealed' SealedSecret unsealed successfully
```

k8s - Operator par l'exemple

- sealed-secrets est un operator

k8s - Operator par l'exemple

- sealed-secrets est un operator
- Il fourni sa CRD : SealedSecret

k8s - Operator par l'exemple

- sealed-secrets est un operator
- Il fourni sa CRD : SealedSecret
- L'operator tourne utilise l'API de kubernetes pour surveiller les modifications de SealedSecrets

k8s - Operator par l'exemple

- sealed-secrets est un operator
- Il fourni sa CRD : SealedSecret
- L'operator tourne utilise l'API de kubernetes pour surveiller les modifications de SealedSecrets
- Si création d'un SealedSecret, il le déchiffre et créé un Secret

k8s - Operator par l'exemple

- sealed-secrets est un operator
- Il fourni sa CRD : SealedSecret
- L'operator tourne utilise l'API de kubernetes pour surveiller les modifications de SealedSecrets
- Si création d'un SealedSecret, il le déchiffre et créé un Secret
- Si suppression d'un SealedSecret, il supprime le Secret associé

Kubernetes et templating

Nous n'avons pas les mêmes valeurs



Templating

- Plusieurs micro-services

Templating

- Plusieurs micro-services
- Globalement identiques à quelques exceptions près

Templating

- Plusieurs micro-services
- Globalement identiques à quelques exceptions près
 - ◆ Image docker

Templating

- Plusieurs micro-services
- Globalement identiques à quelques exceptions près
 - ◆ Image docker
 - ◆ Certaines variables d'environnement peut-être ?

Templating

- Plusieurs micro-services
- Globalement identiques à quelques exceptions près
 - ◆ Image docker
 - ◆ Certaines variables d'environnement peut-être ?
 - ◆ ...

Templating

- Plusieurs micro-services
- Globalement identiques à quelques exceptions près
 - ◆ Image docker
 - ◆ Certaines variables d'environnement peut-être ?
 - ◆ ...
- Nécessaire de copier x fois Deployment/Service/Ingress/... ?

Templating

- Plusieurs micro-services
- Globalement identiques à quelques exceptions près
 - ◆ Image docker
 - ◆ Certaines variables d'environnement peut-être ?
 - ◆ ...
- Nécessaire de copier x fois Deployment/Service/Ingress/... ?
- Template = squelette

Templating

- Plusieurs micro-services
- Globalement identiques à quelques exceptions près
 - ◆ Image docker
 - ◆ Certaines variables d'environnement peut-être ?
 - ◆ ...
- Nécessaire de copier x fois Deployment/Service/Ingress/... ?
- Template = squelette
- Dérivation du template avec les valeurs

Templating

- Plusieurs micro-services
- Globalement identiques à quelques exceptions près
 - ◆ Image docker
 - ◆ Certaines variables d'environnement peut-être ?
 - ◆ ...
- Nécessaire de copier x fois Deployment/Service/Ingress/... ?
- Template = squelette
- Dérivation du template avec les valeurs
- Utilisation d'Helm

Helm

- Plusieurs solutions de templating avec k8s

Helm

- Plusieurs solutions de templating avec k8s
- Helm plus répandue

Helm

- Plusieurs solutions de templating avec k8s
- Helm plus répandue
 - ◆ Facile d'utilisation également

Helm

- Plusieurs solutions de templating avec k8s
- Helm plus répandue
 - ◆ Facile d'utilisation également
- Helm registry

Helm

- Plusieurs solutions de templating avec k8s
- Helm plus répandue
 - ◆ Facile d'utilisation également
- Helm registry
 - ◆ Beaucoup de templates déjà proposés

Helm

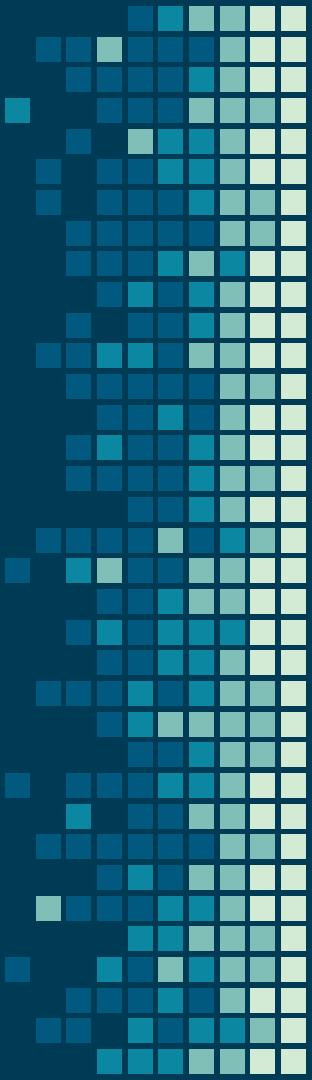
- Plusieurs solutions de templating avec k8s
- Helm plus répandue
 - ◆ Facile d'utilisation également
- Helm registry
 - ◆ Beaucoup de templates déjà proposés
 - ◆ Ex: bitnami's sealed-secrets

Helm

- Plusieurs solutions de templating avec k8s
- Helm plus répandue
 - ◆ Facile d'utilisation également
- Helm registry
 - ◆ Beaucoup de templates déjà proposés
 - ◆ Ex: bitnami's sealed-secrets
- Version 3 actuelle : pas de config dans le cluster

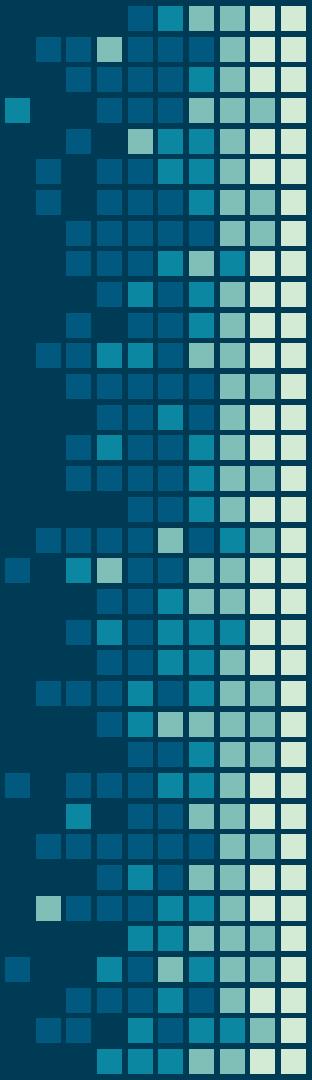
Merci !

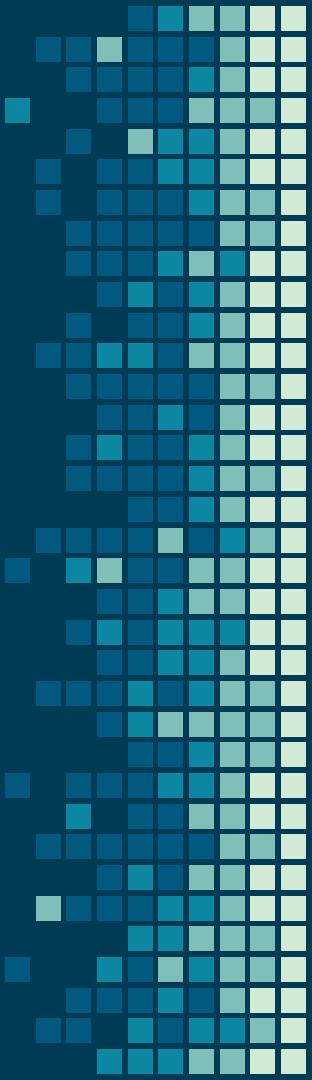
Des questions ?



Merci !

Des questions ?





Disponible sur zarak.fr/