Module 4 : Les Architectures Conteneurisées

16 & 17 mars 2023

Thomas Lecler
Software Engineer
Devops

Amaury Bloch
Software Engineer
Devops

QUI SUIS-JE ?

Thomas
LECLER
Software Engineer - DevOps

Mon expertise

4 Projets (Ministère des Armées / Ministère de l'Intérieur / Ministère de l'Agriculture)

Mon parcours

- A Capgemini depuis le Stage de Fin d'Étude
- Avant Capgemini :
  - École d'ingénieur en Informatique, Réseau et Cybersécurité
  - CPGE : Mathématiques et Physique

Mes technos

Java, Terraform, Ansible, docker, RabbitMQ, Vue.js, JS, python

QUI SUIS-JE ?

Amaury BLOCH

Software Engineer - DevOps

Mon expertise

6 Projets (Ministère des Armées / Ministère de l'Agriculture)

Mon parcours

- A Capgemini depuis le Stage de Fin d'Étude
- Avant Capgemini :
  - École d'ingénieur généraliste et spécialité informatique
  - CPGE : Physique, Technologie et Sciences de l'ingénieur

Mes technos

kubernetes docker ANSIBLE CI/CD RabbitMQ Vue.js spring Falco

AGENDA

- Module 1 – Le Cloud Computing
  Comprendre de manière critique le Cloud
- Module 2 – Les Architectures Cloud
  Instancier une architecture reposant sur les services présentés
- Module 3 – Scalabilité
- Module 4 – Les Architectures Conteneurisées
  Réduire l'adhérence vis-à-vis des clouders

• Module 4 – Les Architectures Conteneurisées Réduire l'adhérence vis-à-vis des clouders

Conteneuriser ses applications

- Concepts & bénéfices
- Création d'une image
- Lancement d'un conteneur

Orchestrer ses conteneurs via Kubernetes

- Concepts & bénéfices
- Déployer un cluster Kubernetes
- Les ressources Kubernetes
- Déployer des ressources sur le cluster

Industrialiser ses déploiements

- Du code au cluster
- Automatiser la construction et le déploiement

Exploiter & superviser ses applications en production

- Exploiter ses applications
- Superviser ses applications
- Collecter des métriques

CONTENEURISER SES APPLICATIONS

POUR LES RENDRE PORTABLES

Conteneuriser ses applications

- Concepts & bénéfices
- Création d'une image
- Lancement d'un conteneur

Orchestrer ses conteneurs via Kubernetes

- Concepts & bénéfices
- Déployer un cluster Kubernetes
- Les ressources Kubernetes
- Déployer des ressources sur le cluster

Industrialiser ses déploiements

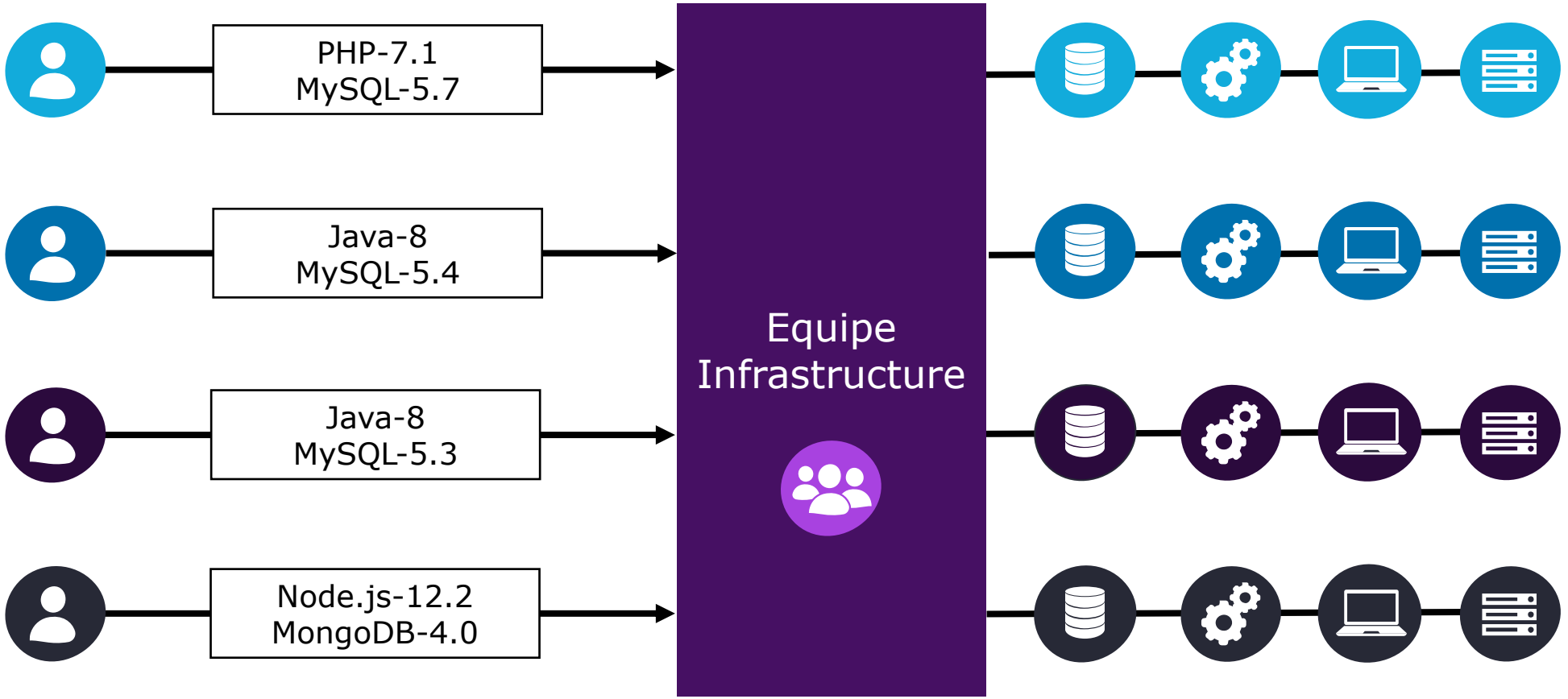
- Du code au cluster
- Automatiser la construction et le déploiement

Exploiter & superviser ses applications en production

- Exploiter ses applications
- Superviser ses applications
- Collecter des métriques

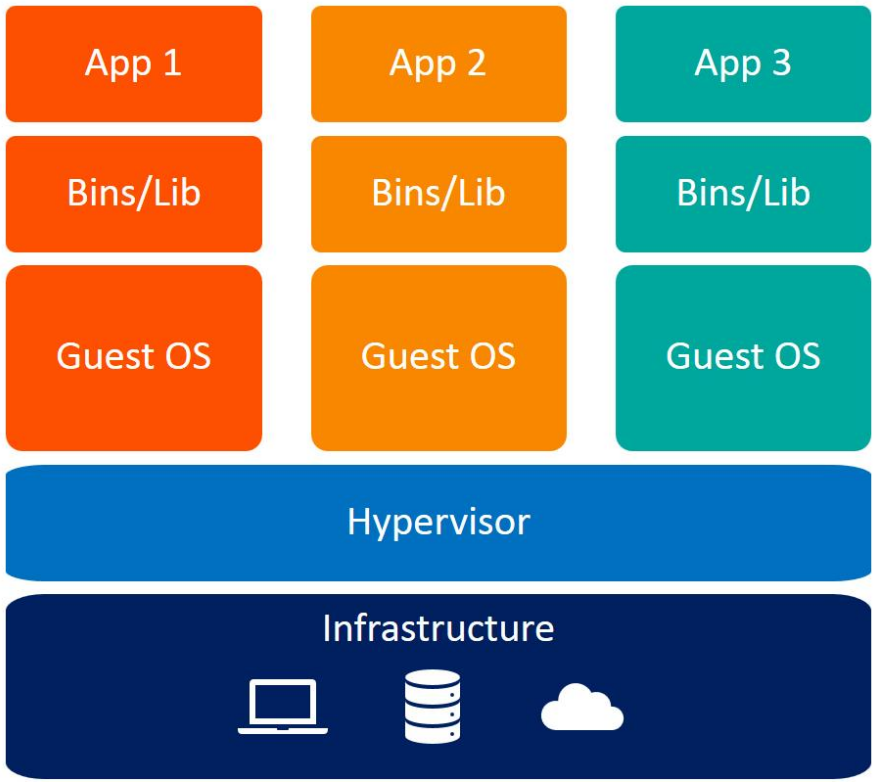
CONCEPTS

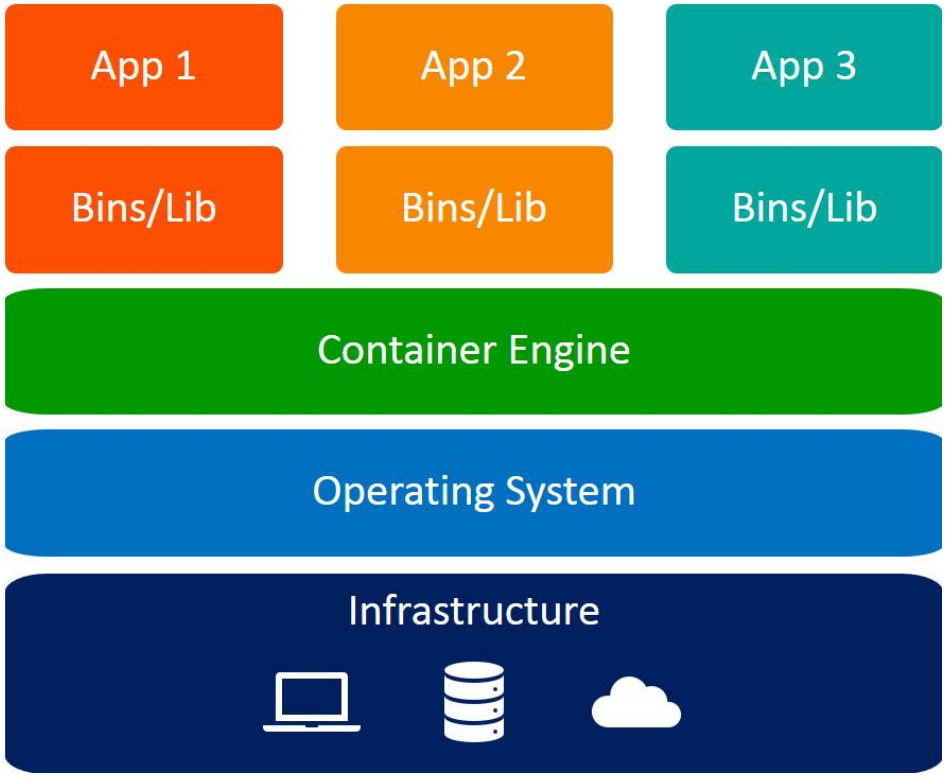
Déployer une application en entreprise



CONCEPTS

Virtualisation vs Conteneurisation

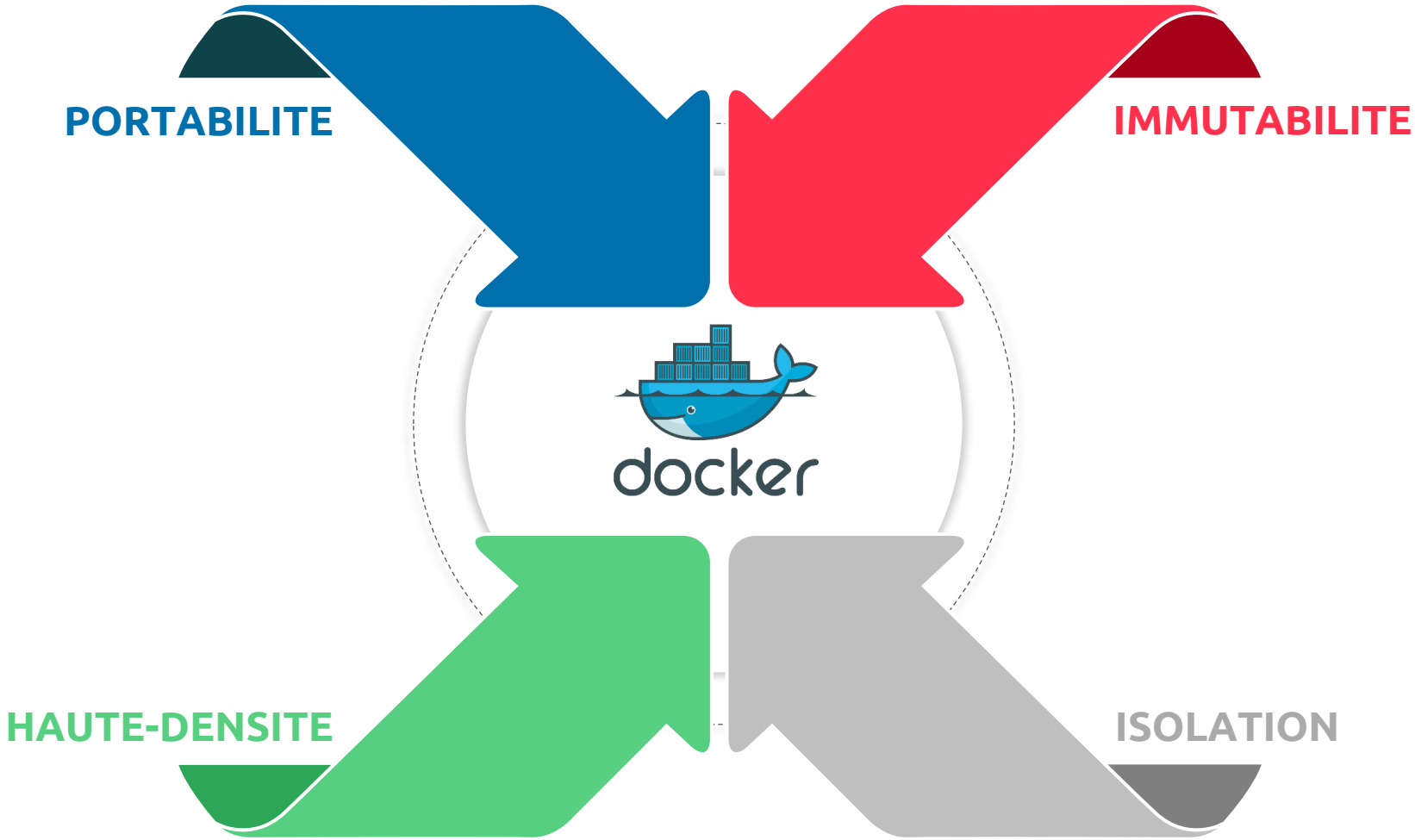




https://www.bmc.com/blogs/containers-vs-virtual-machines

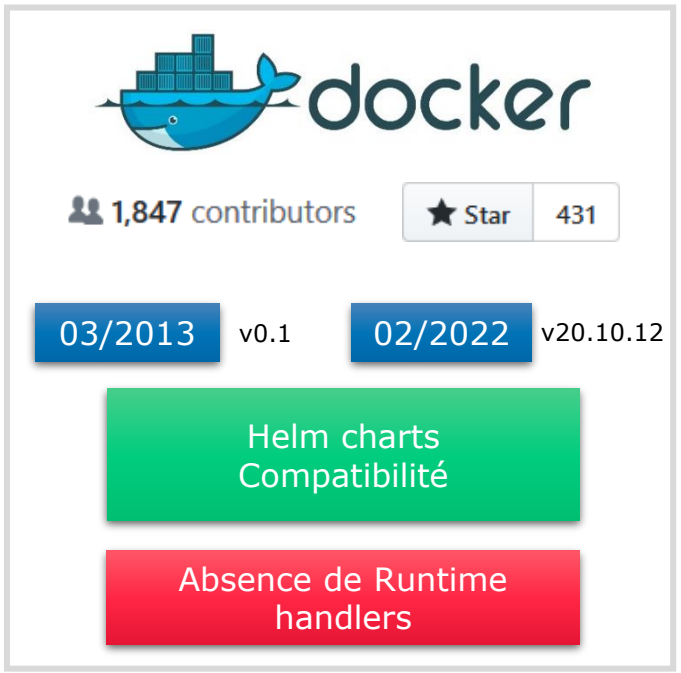
CONCEPTS

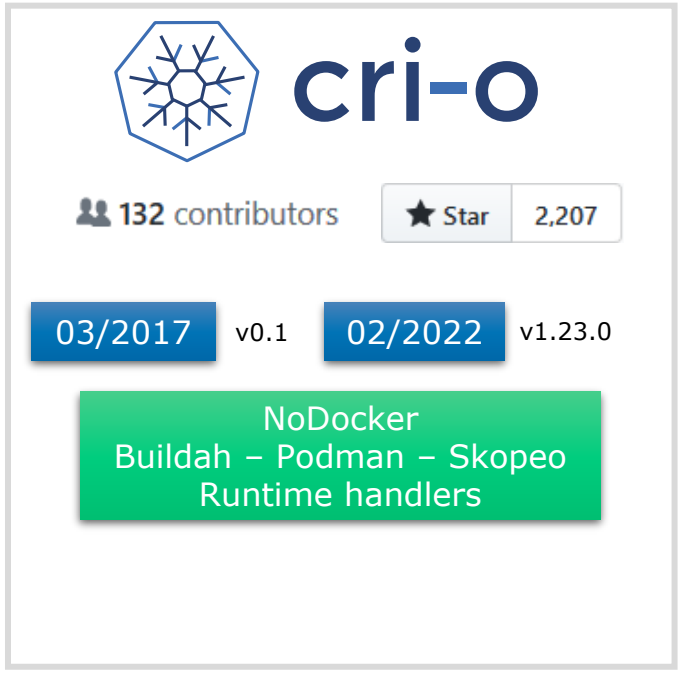
Bénéfices de la conteneurisation

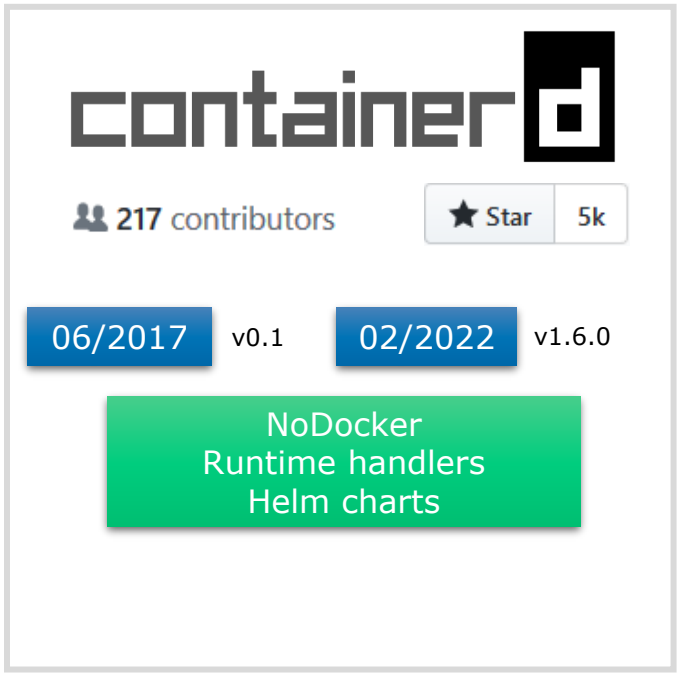


CONTENEURISATION

Choix du Container Engine







Opportunités

Sandboxing

Multi Container Runtime

LES STANDARDS DE LA CONTENEURISATION

Open Container Initiative et Container Runtime Initiative

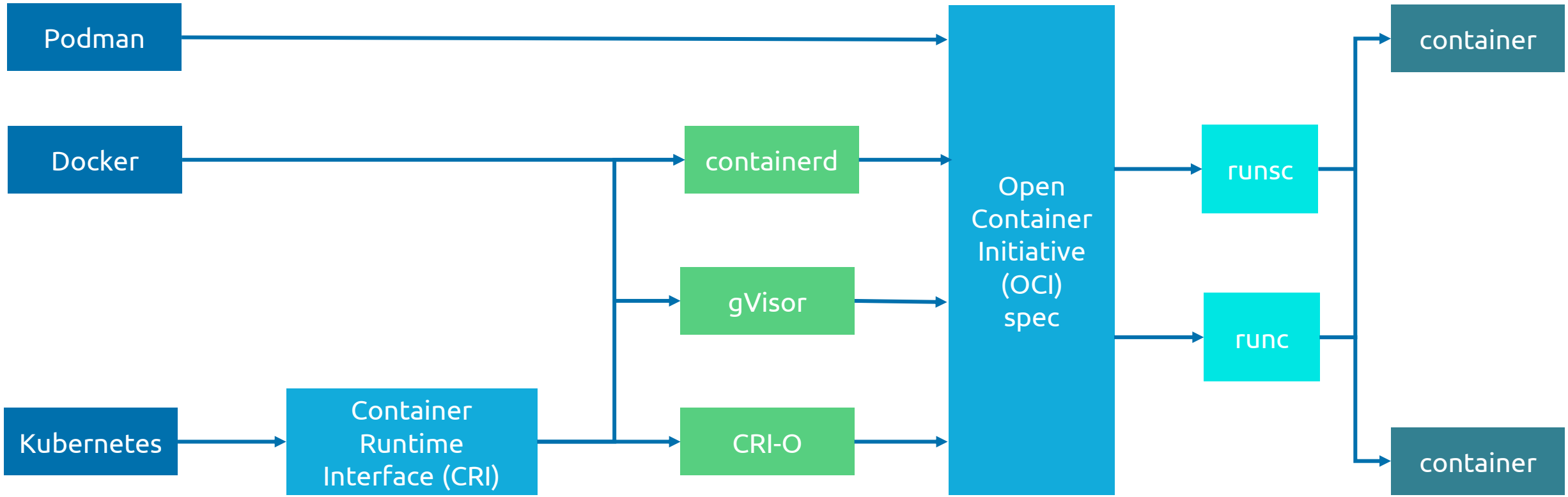
L'**Open Container Initiative** (**OCI**) publie des spécifications pour les conteneurs et leurs images.

La Kubernetes **Container Runtime Interface** (**CRI**) définit une API entre Kubernetes et un conteneur runtime sous-jacent.

OCI permet de choisir la chaîne d'outils pour conteneurs la plus adaptée au contexte (e.g. Docker, CRI-O, Podman, LXC, etc).

LES ACTEURS DU DÉPLOIEMENT D'UN CONTENEUR

Construire et exécuter des conteneurs



CRÉATION D'UNE IMAGE

L'image : version statique de l'application

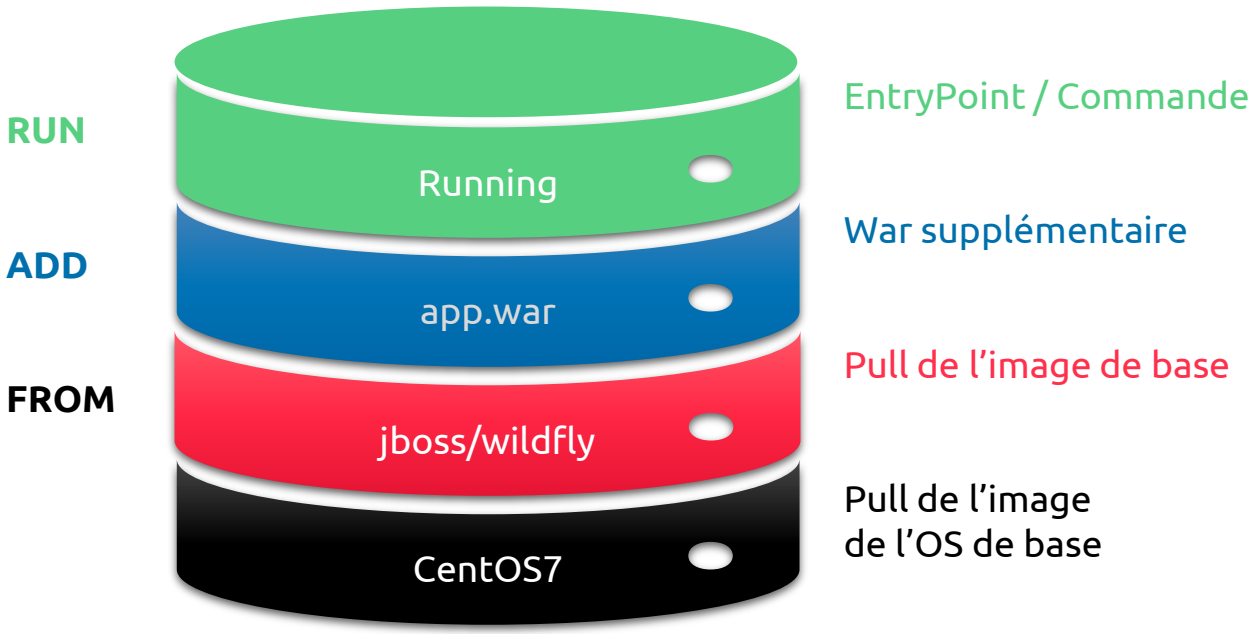
Dockerfile

```
FROM jboss/wildfly:11

ADD /target/*.war
/opt/jboss/wildfly/standalone/deployments/

RUN echo 'we are running some # of cool things'

EXPOSE 8080
USER 9000
```



HÉBERGER SON APPLICATION

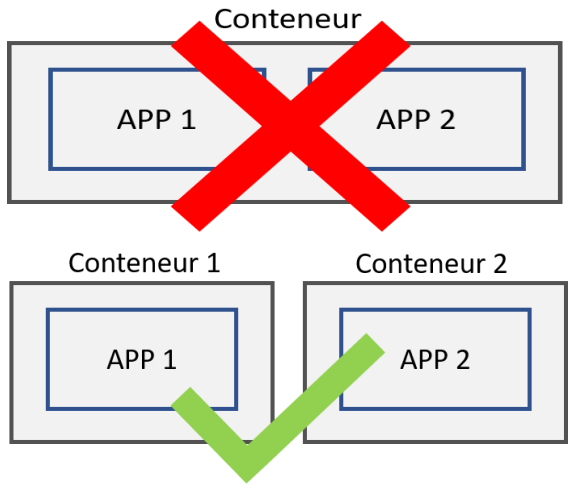
Utiliser des registres publics ou on-premise

Registres publics

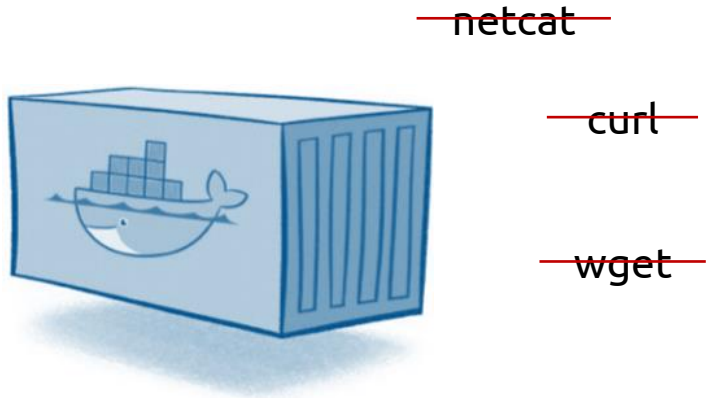
Registres privés

CRÉATION D'UNE IMAGE

Bonne pratique : conteneuriser le minimum



Un seul composant par conteneur
e.g. application et BDD dans des conteneurs différents

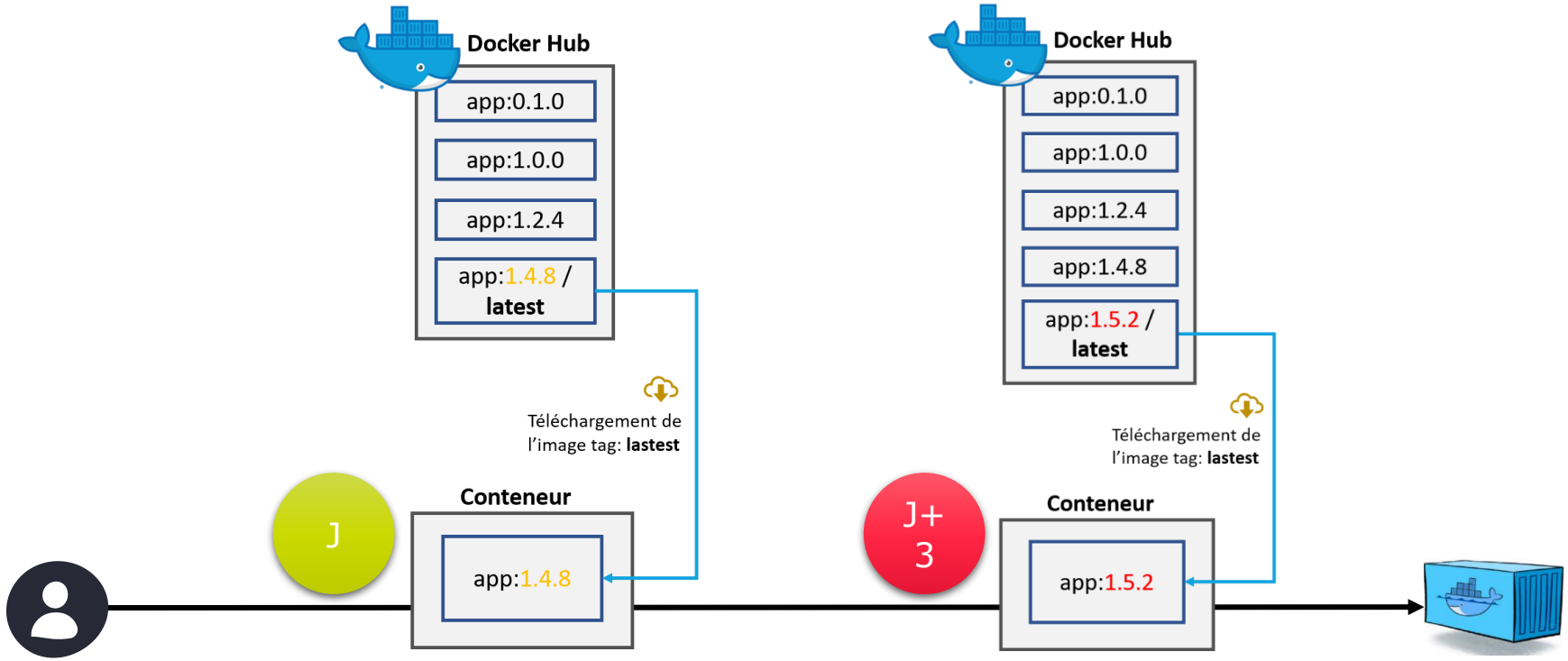


Supprimer les outils inutilisés
e.g. netcat, curl, wget

Réduire poids et temps de démarrage
Réduire la portée d'une action malveillante

CRÉATION D'UNE IMAGE

Bonne pratique : utiliser des images taguées

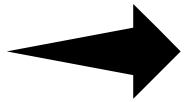


La version de l'image latest dépend de la date de pull
→ Améliorer la stabilité en réduisant les effets de bords

CRÉATION D'UNE IMAGE

Bonne pratique : utiliser des images taguées

```
FROM ubuntu:latest
RUN apt-get update --fix-missing && \
  apt-get install -y redis-server && \
  rm -rf /var/lib/apt/lists/*
USER 9000
EXPOSE 6379
CMD redis-server
```



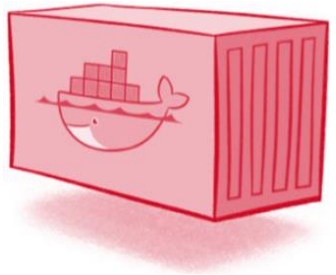
```
FROM ubuntu:xenial
RUN apt-get update --fix-missing && \
  apt-get install -y redis-server && \
  rm -rf /var/lib/apt/lists/*
USER 9000
EXPOSE 6379
CMD redis-server
```

✗ **Image non taguée**

✓ **Image taguée**

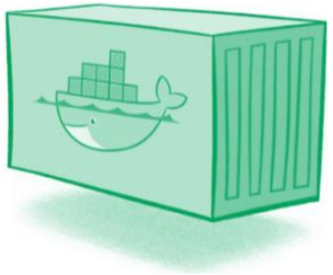
CRÉATION D'UNE IMAGE

Bonne pratique : utiliser des images non root



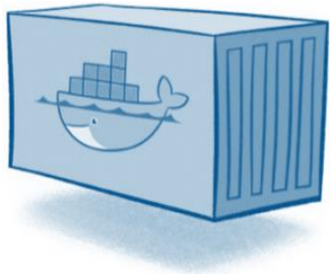
Root

Accès illimité au conteneur
Accès au système de fichiers de l'hôte (volume)



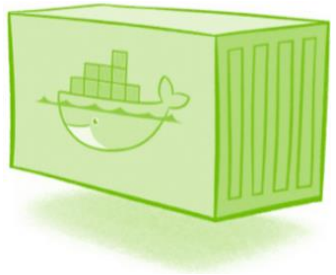
Non-Root

Accès limité au conteneur



Privileged

Accès aux devices (/dev) de l'hôte



Un-privileged

Accès limité aux devices (/dev) de l'hôte

Construire des images non-root
Capitaliser sur des images non-root existantes
→ Réduire la surface d'attaque

CRÉATION D'UNE IMAGE

Bonne pratique : utiliser des images non root

```
FROM ubuntu:latest
RUN apt-get update --fix-missing && \
  apt-get install -y redis-server && \
  rm -rf /var/lib/apt/lists/*
EXPOSE 6379
CMD redis-server
```

```
$ docker run --rm example whoami
root
```

❌ **Utilisateur root (par défaut)**

```
FROM ubuntu:xenial
RUN apt-get update --fix-missing && \
  apt-get install -y redis-server && \
  rm -rf /var/lib/apt/lists/*
USER 9000
EXPOSE 6379
CMD redis-server
```

```
$ docker run --rm example whoami
whoami: cannot find name for user ID 9000
```

✓ **Utilisateur non root**

LANCEMENT D'UN CONTENEUR

Le conteneur : version dynamique de l'application

Aides

```
docker --help
```

```
docker inspect <id>
```

Listings

```
docker images
```

```
docker ps
```

```
docker rm(i) <id>
```

Inspections

```
docker logs <id>
```

```
docker exec –it <id> <cmd>
```

Instanciations

```
docker build <Dockerfile>
```

```
docker create <img>
```

```
docker run <img>
```


Orchestrer ses conteneurs via Kubernetes

Pour passer à l'échelle

Conteneuriser ses applications

- Concepts & bénéfices
- Création d'une image
- Lancement d'un conteneur

Orchestrer ses conteneurs via Kubernetes

- Concepts & bénéfices
- Déployer un cluster Kubernetes
- Les ressources Kubernetes
- Déployer des ressources sur le cluster

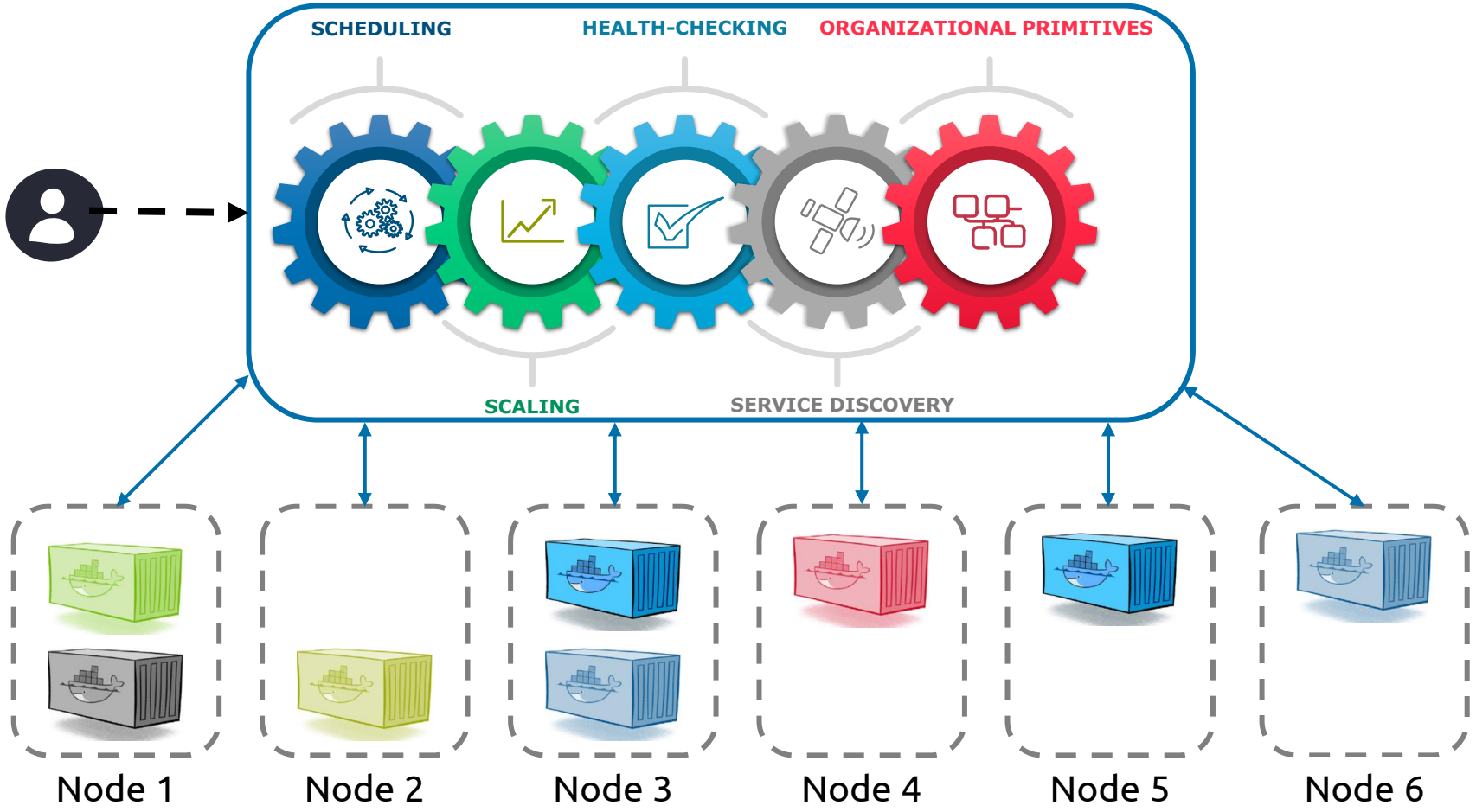
Industrialiser ses déploiements

- Du code au cluster
- Automatiser la construction et le déploiement

Exploiter & superviser ses applications en production

- Exploiter ses applications
- Superviser ses applications
- Collecter des métriques

DÉPLOYER DES CONTENEURS SUR DES MACHINES



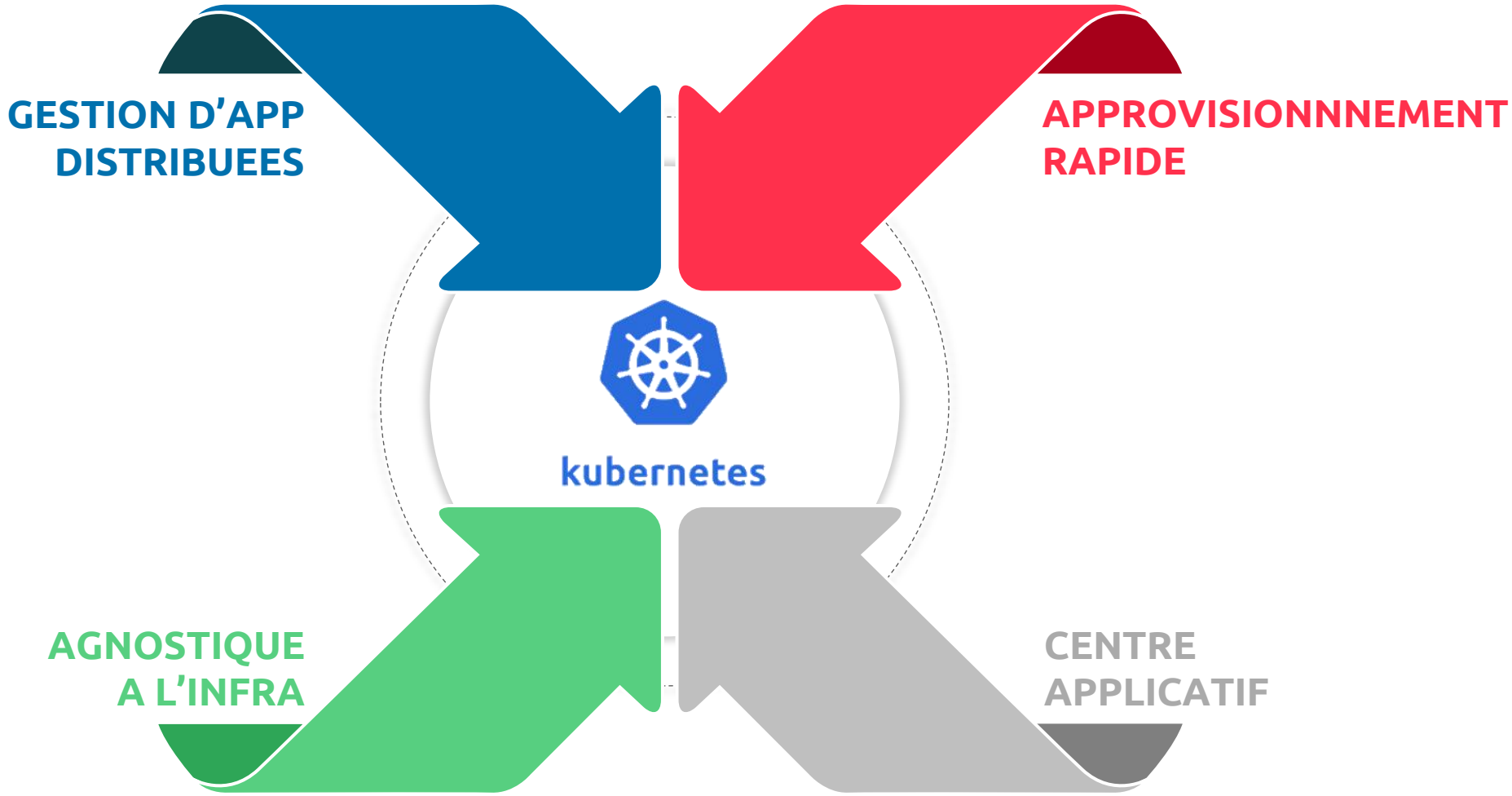
kubernetes

docker SWARM

Docker Compose

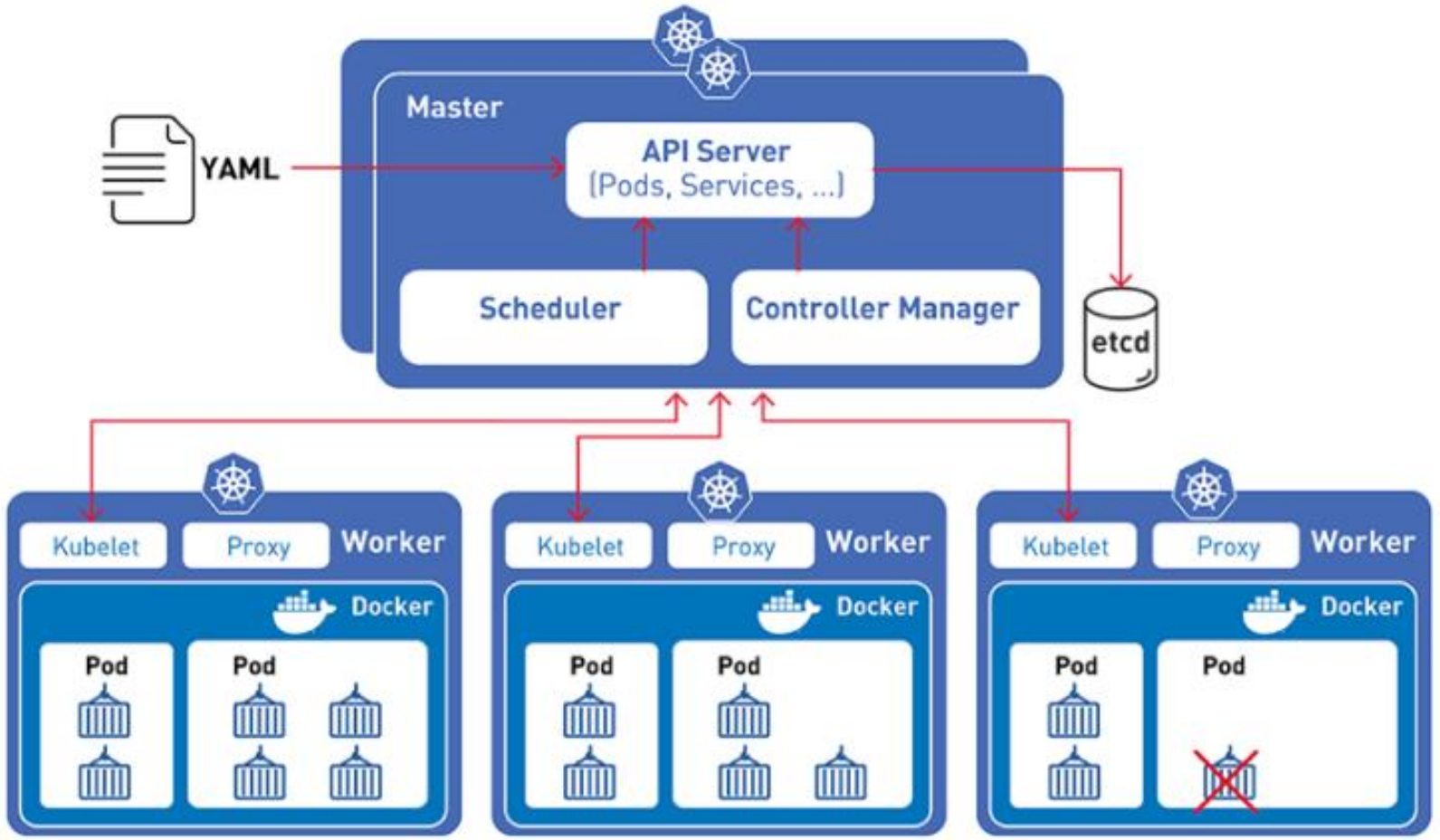
MESOS MARATHON

BÉNÉFICES DE L'ORCHESTRATION



ARCHITECTURE D'UN CLUSTER KUBERNETES

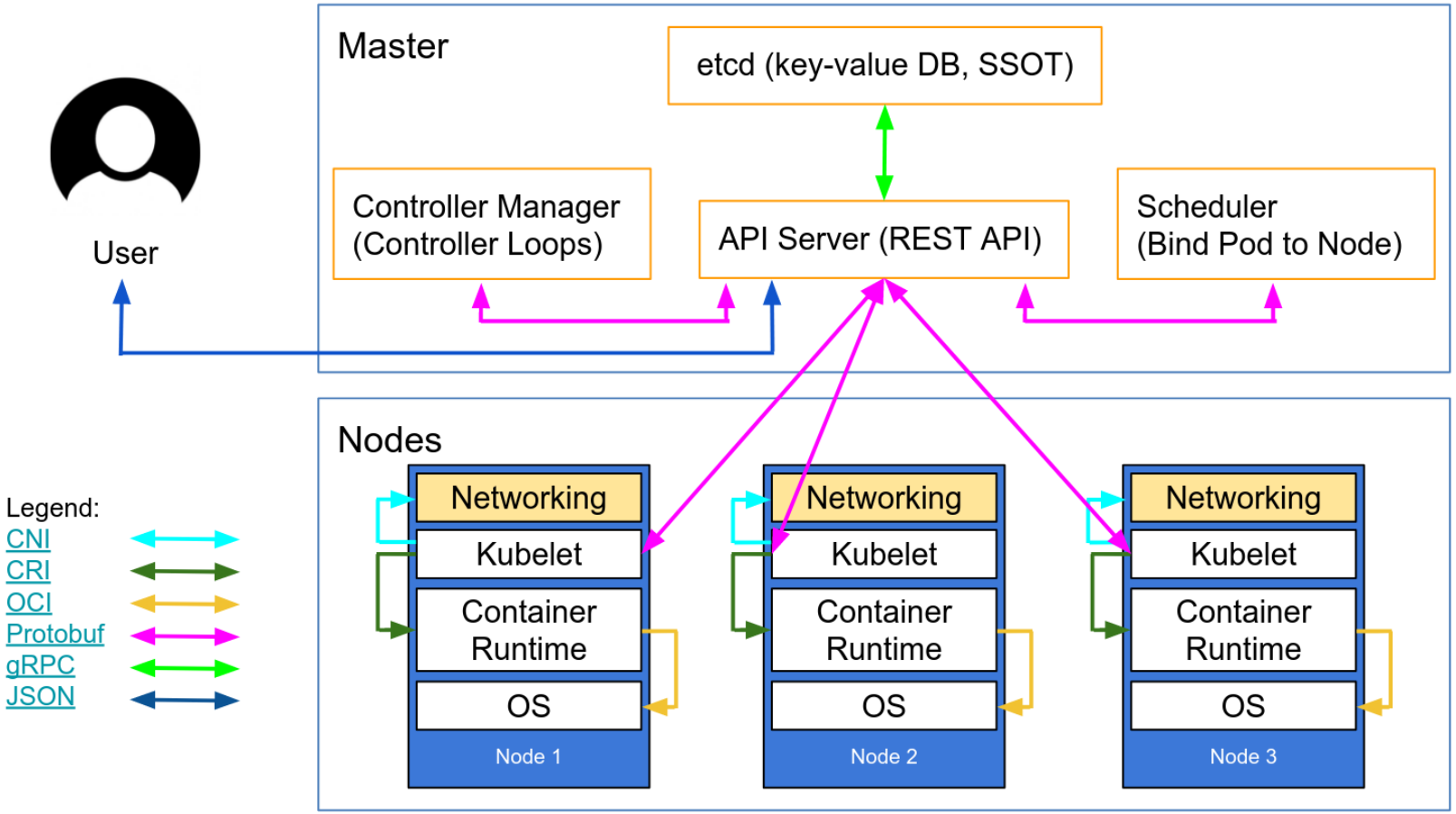
Kubernetes, un orchestrateur modulaire



https://devtron.ai/blog/why-kubernetes/

ARCHITECTURE D'UN CLUSTER KUBERNETES

Flux d'informations au sein d'un cluster



Composants Master

- kube-apiserver
- kube-scheduler
- etcd
- controller-manager

Composants nœuds

- kubelet
- kube-proxy ou CN (CNI)
- CR (CRI)

DÉPLOYER UN CLUSTER KUBERNETES

Choix de l'installer / manager

Installers

Kops / Kubespray

- Déploie l'infra & installe K8s
- Nombre limité de providers
- Debug difficile

Cloud-Hosted K8 Services

GKE / EKS / AKS

- Facilité de mise en place
- Composants K8s managés

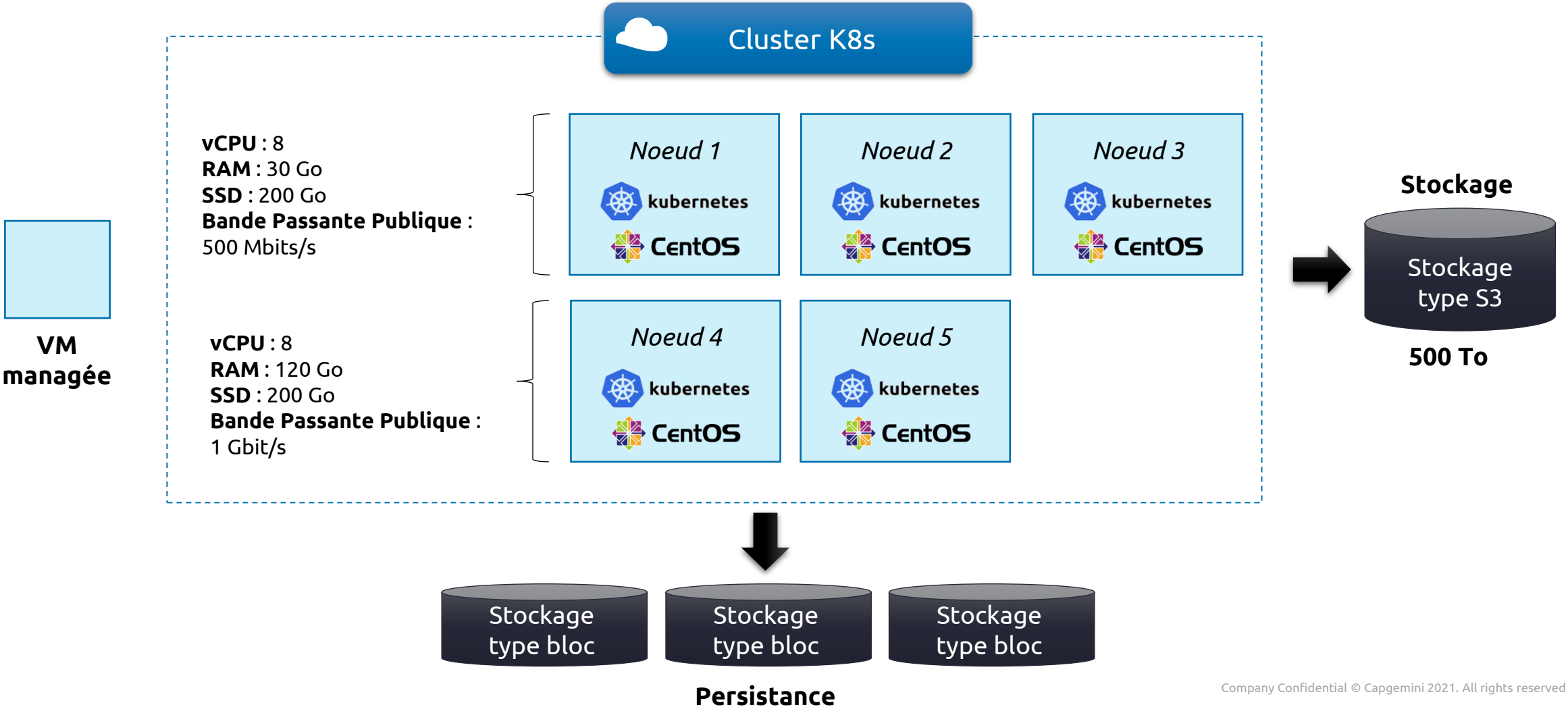
Red Hat Kubernetes Installer

Red Hat OpenShift

- Installeur automatisé
- Fonctionnalités additionnelles
- Payant

DÉPLOYER UN CLUSTER KUBERNETES

Exemple d'architecture physique



Conteneuriser ses applications

- Concepts & bénéfices
- Création d'une image
- Lancement d'un conteneur

Orchestrer ses conteneurs via Kubernetes

- Concepts & bénéfices
- Déployer un cluster Kubernetes
- Les ressources Kubernetes
- Déployer des ressources sur le cluster

Industrialiser ses déploiements

- Du code au cluster
- Automatiser la construction et le déploiement

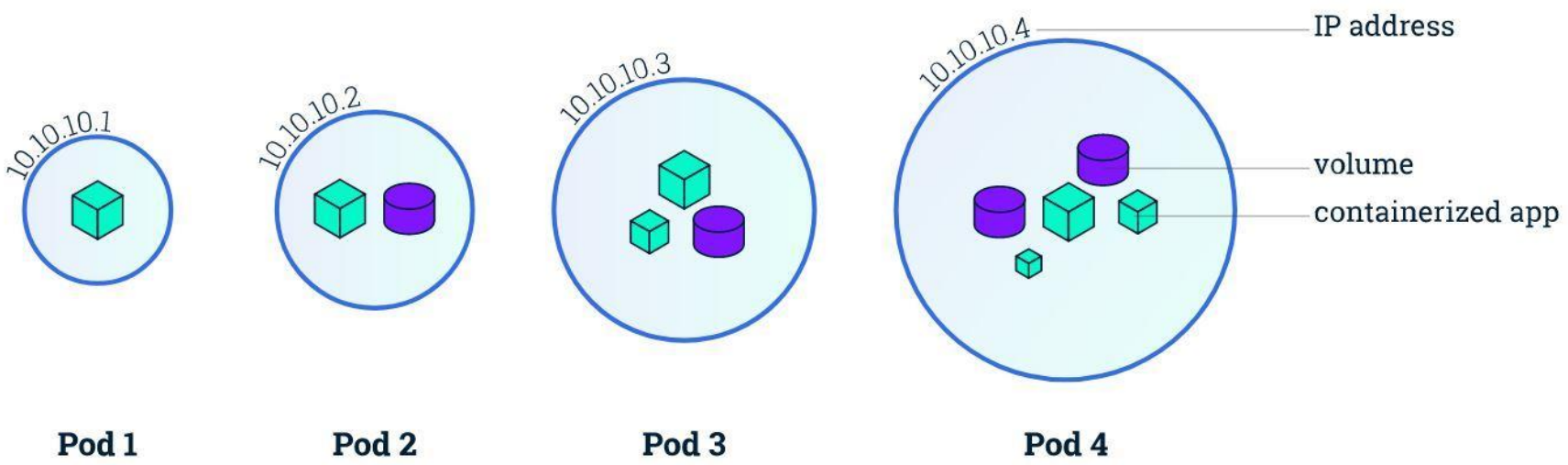
Exploiter & superviser ses applications en production

- Exploiter ses applications
- Superviser ses applications
- Collecter des métriques

POD

Unité élémentaire de Kubernetes

***Ressource** Kubernetes : objet déployé par un **manifest** yaml*

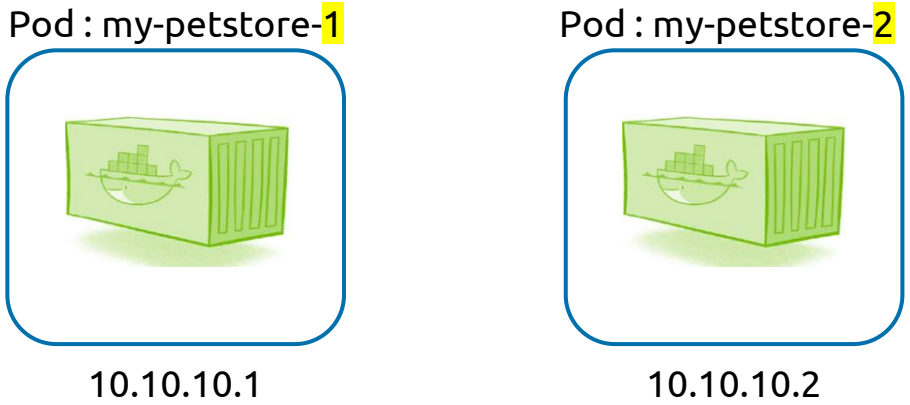


DEPLOYMENT

Déployer son application sur un cluster Kubernetes

deployment.yaml

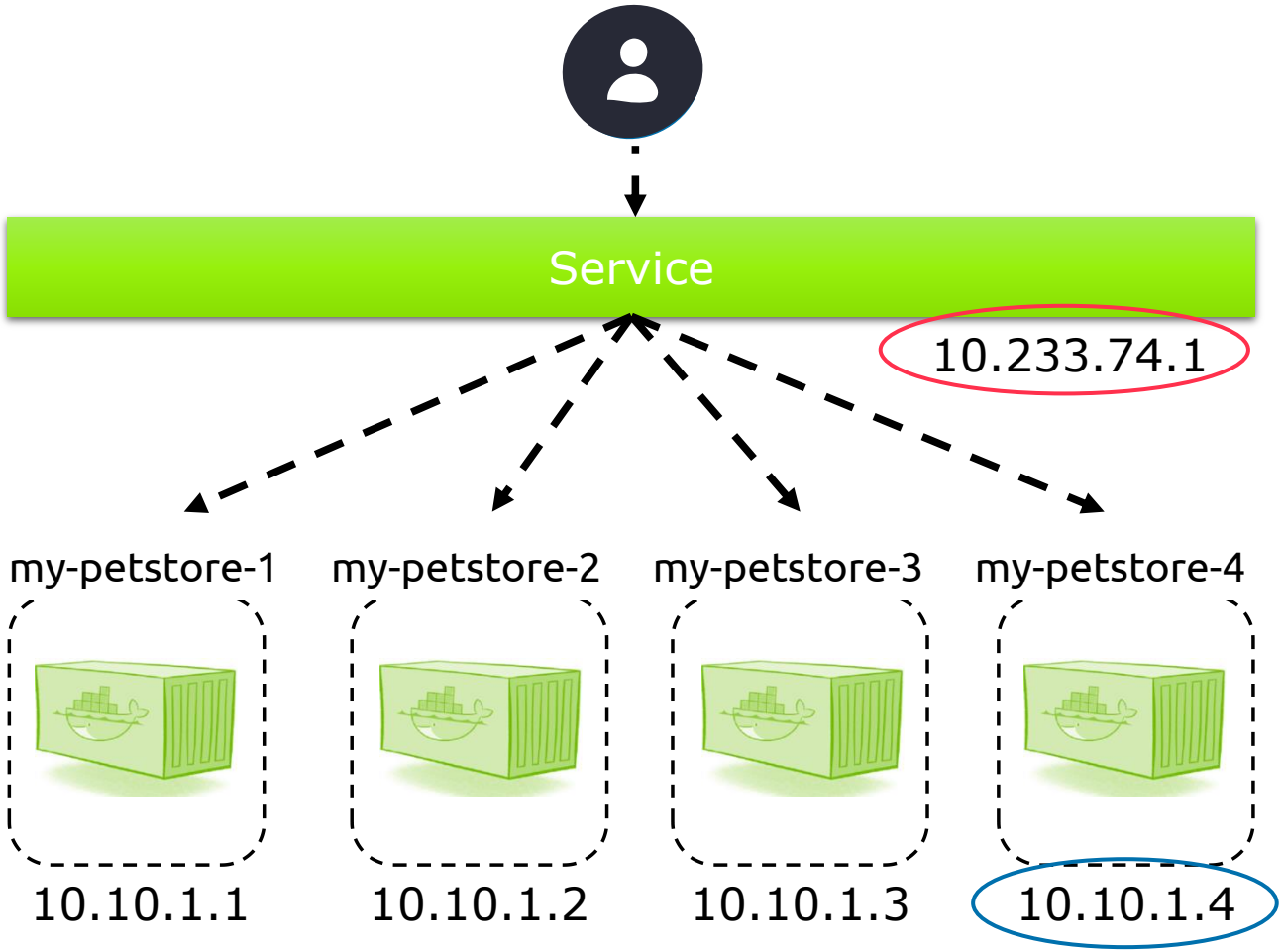
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-petstore
  namespace: java
spec:
  selector:
    matchLabels:
      run: my-petstore
  replicas: 2
  template:
    metadata:
      labels:
        run: my-petstore
    spec:
      containers:
        - name: petstore
          image: <registre>/java-petstore-test:latest
```



- Stabilité de l'application
- Continuité du service
- Facilité de déploiement

SERVICE

Loadbalancer intégré à Kubernetes



IP du Service stable

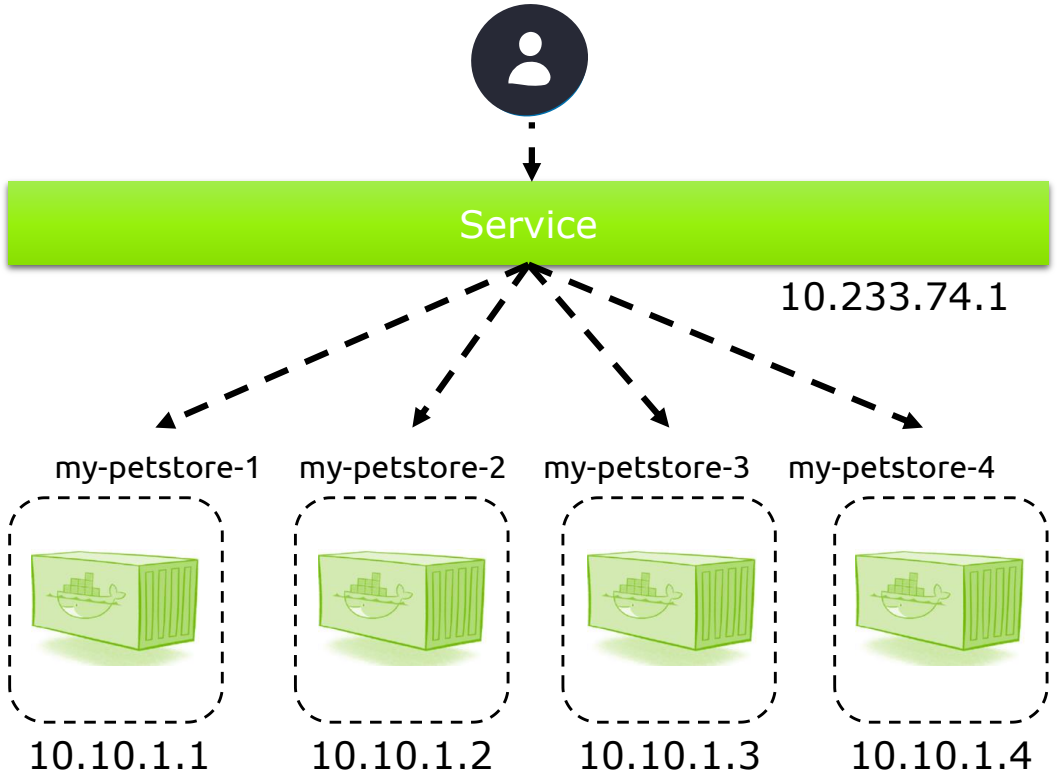
IP du Pod dynamique,
réattribuée à chaque démarrage

SERVICE

Déployer un Service

service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: my-petstore
  namespace: java
  labels:
    run: my-petstore
spec:
  ports:
    - port: 8080
      protocol: TCP
  selector:
    run: my-petstore
```

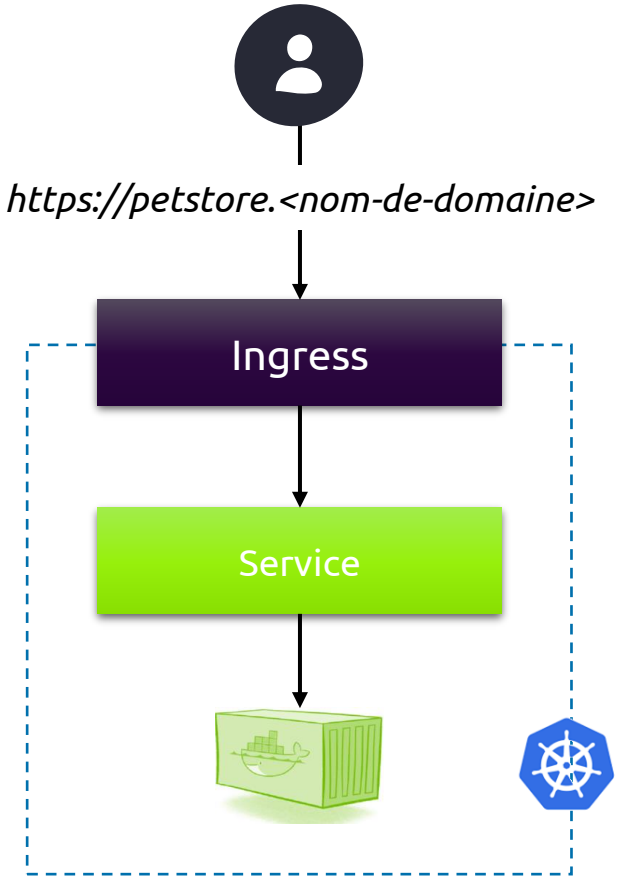


INGRESS

Configurer le trafic entrant avec l'Ingress

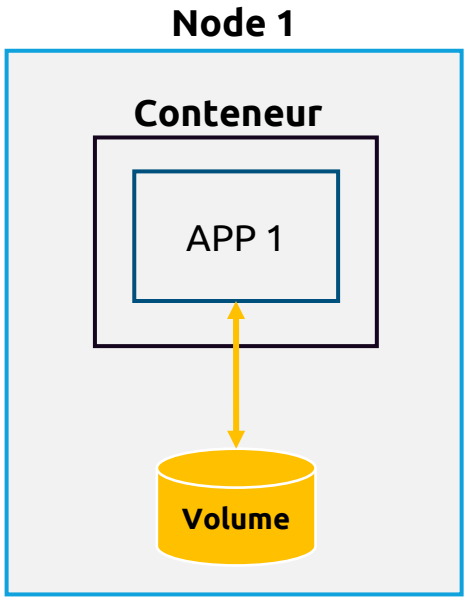
ingress.yaml

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  annotations:
    nginx.ingress.kubernetes.io/proxy-connect-timeout: "30"
    nginx.ingress.kubernetes.io/ssl-redirect: "false"
    nginx.ingress.kubernetes.io/app-root: /applicationPetstore
  labels:
    app: my-petstore
  name: my-petstore
  namespace: java
spec:
  rules:
  - host: petstore. <nom-de-domaine>
    http:
      paths:
      - backend:
          serviceName: my-petstore
          servicePort: 8080
```



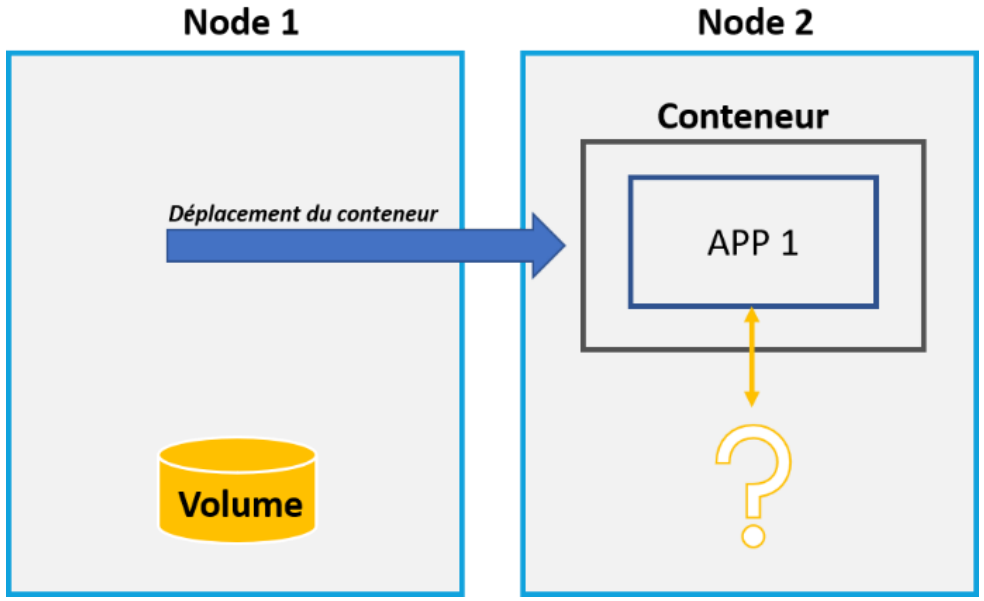
VOLUMES

Persister la donnée sur un stockage bloc

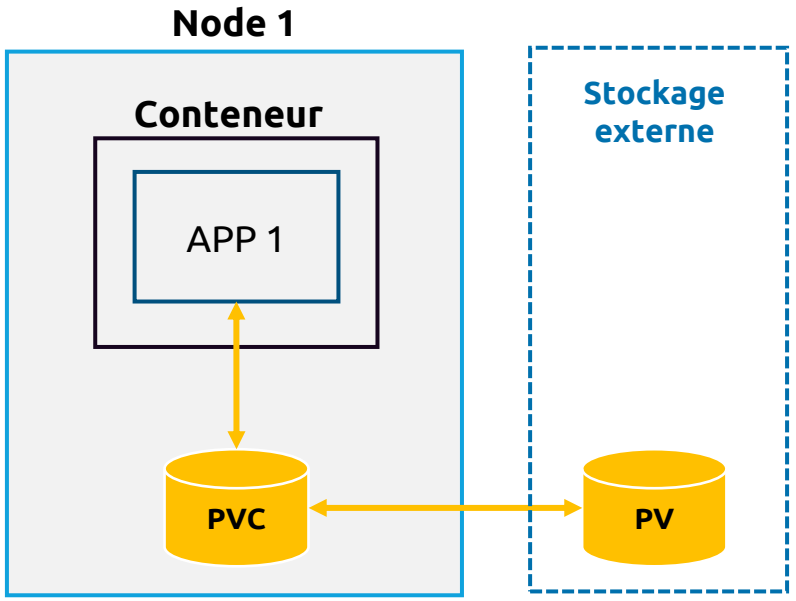


VOLUMES

Persister la donnée sur un stockage bloc externe



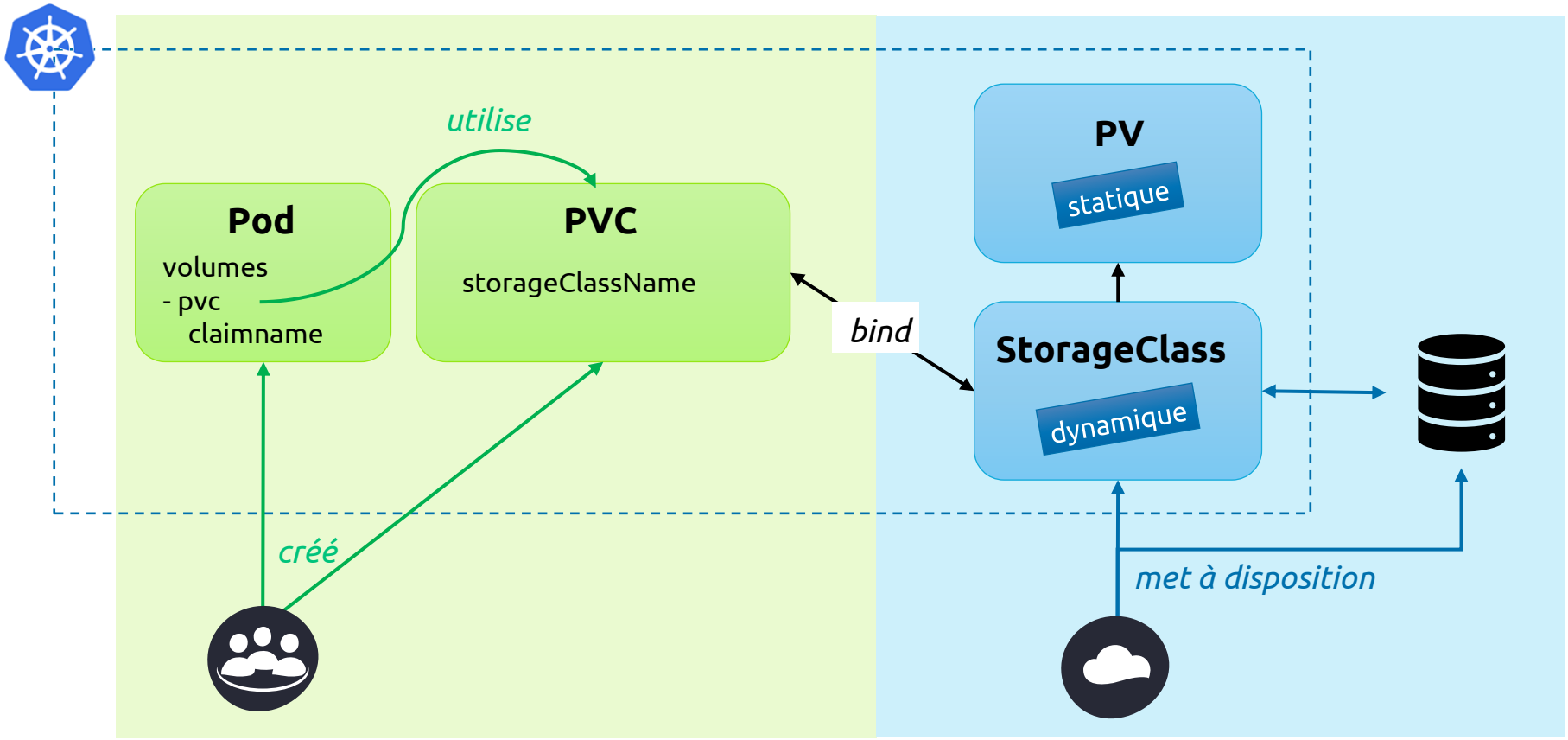
❌ **Volume intra cluster**



✔ **Volume externe**

VOLUMES

PersistentVolume & PersistentVolumeClaim



VOLUMES

PersistentVolumeClaim : demander un volume via une storageClass

pvc.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: bdd-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: csi-cinder-high-speed
```

Utiliser une storageClass dynamique

→ Pas de gestion des volumes

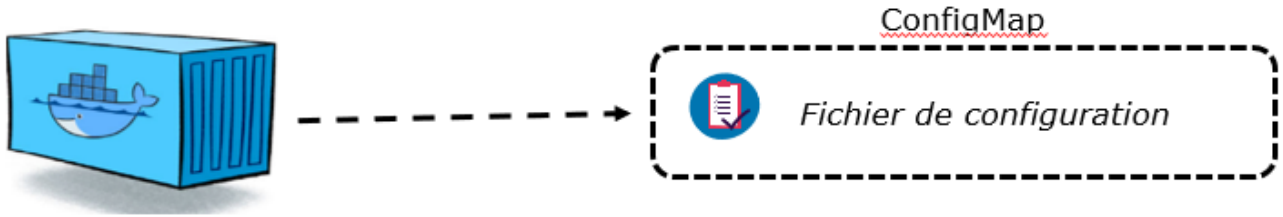
→ Approvisionnement dynamique

Stabilité de l'application

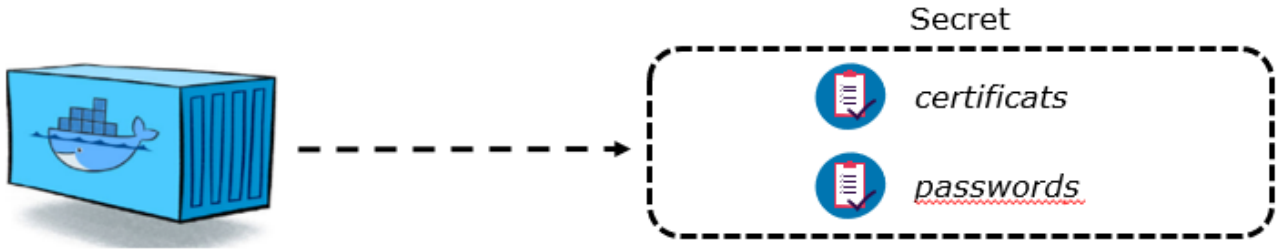
CONFIGMAPS & SECRETS

Stocker la configuration et les credentials dans les Pods

- Stocker des configurations avec la ConfigMap



- Stocker les credentials ou les certificats avec les Secrets



CONFIGMAP

Stocker la configuration des applications dans les Pods

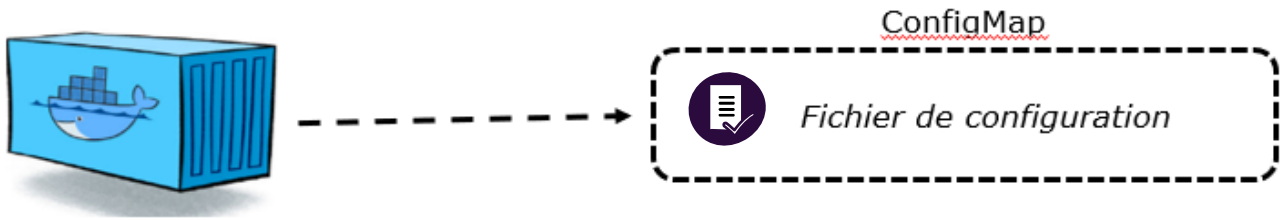
configmap.yaml

```
apiVersion: v1
kind: ConfigMap
data:
  demo: |
    hello world
metadata:
  name: demo
  namespace: java
```

+

deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
spec:
  template:
    spec:
      volumes:
      - name: demo
        configMap:
          name: demo
```



SECRET

Stocker les secret encodés des applications dans les Pods

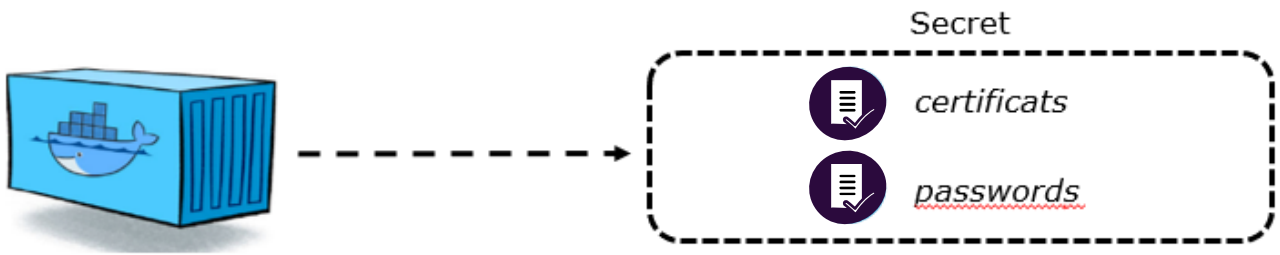
secret.yaml

```
apiVersion: v1
kind: Secret
data:
  username: YWRtaW4=
metadata:
  name: demo
```

+

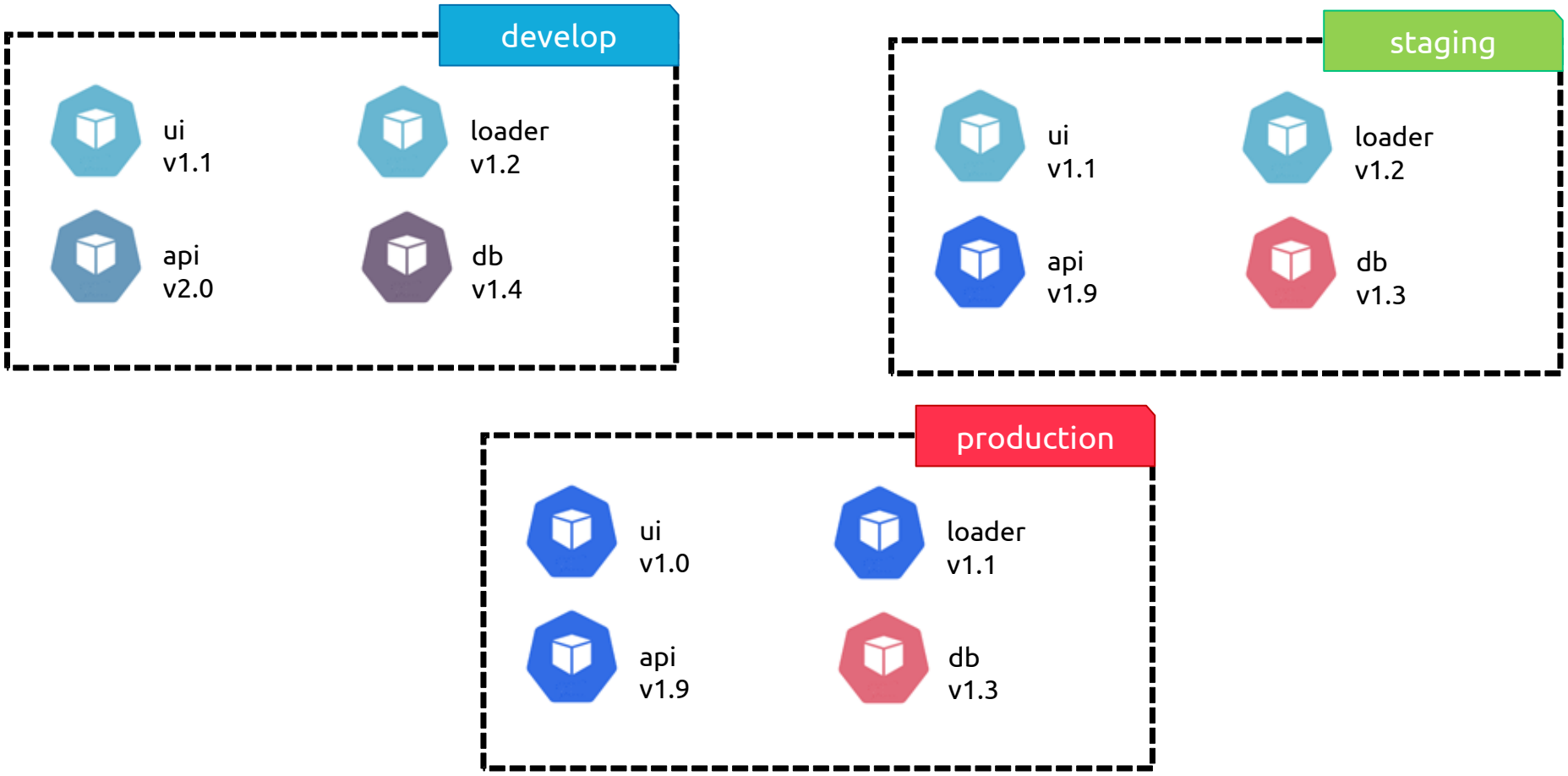
deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
spec:
  template:
    spec:
      volumes:
      - name: demo
        secret:
secretname: demo
```



NAMESPACE

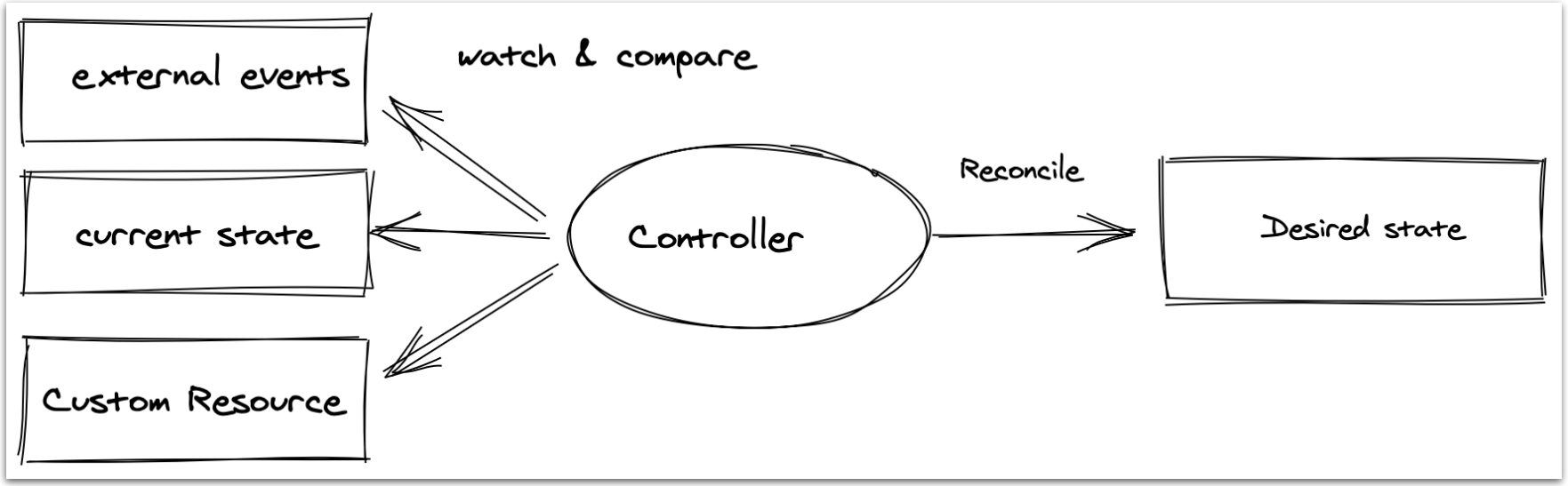
Isoler les Pods via des Namespaces



CUSTOM RESOURCE DEFINITIONS

Extension de l'API Kubernetes par les Operators

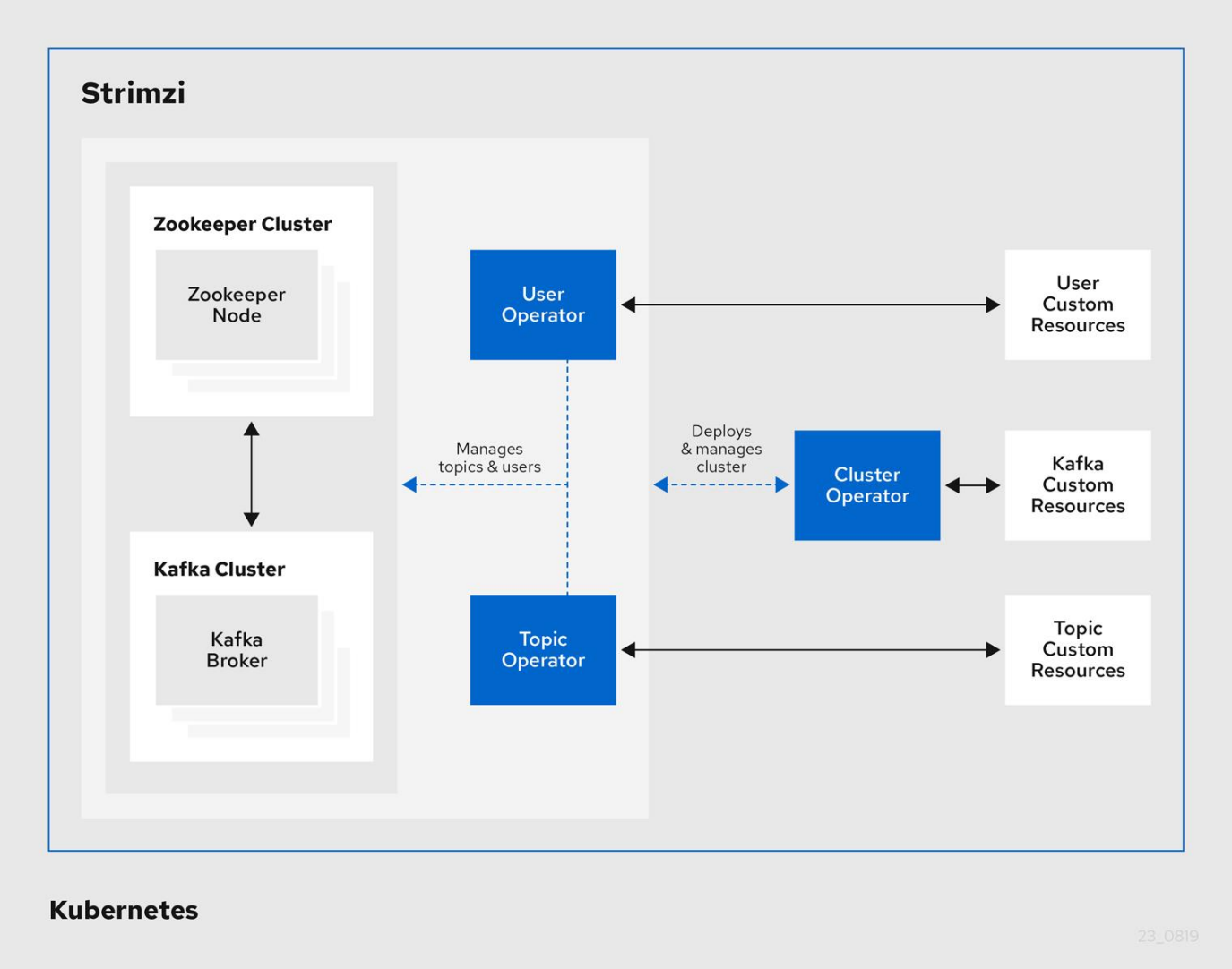
"An operator is a Kubernetes controller that understands 2 domains: Kubernetes and something else. By combining knowledge of both domains, it can automate tasks that usually require a human operator that understands both domains." Jimmy Zelinskie



https://github.com/cncf/tag-app-delivery/blob/eece8f7307f2970f46f100f51932db106db46968/operator-wg/whitepaper/Operator-WhitePaper_v1-0.md

CUSTOM RESOURCE DEFINITIONS

Exemples de Kubernetes Operators



DÉPLOYER UN WORKLOAD

CLI kubectl : commandes utiles

Aides

```
kubectl --help
```

```
kubectl explain <resource>
```

Listings

```
kubectl get nodes
```

```
kubectl get pods
```

```
kubectl get svc
```

Inspections

```
kubectl describe pod <pod-id>
```

```
kubectl logs <pod-id>
```

Instanciations

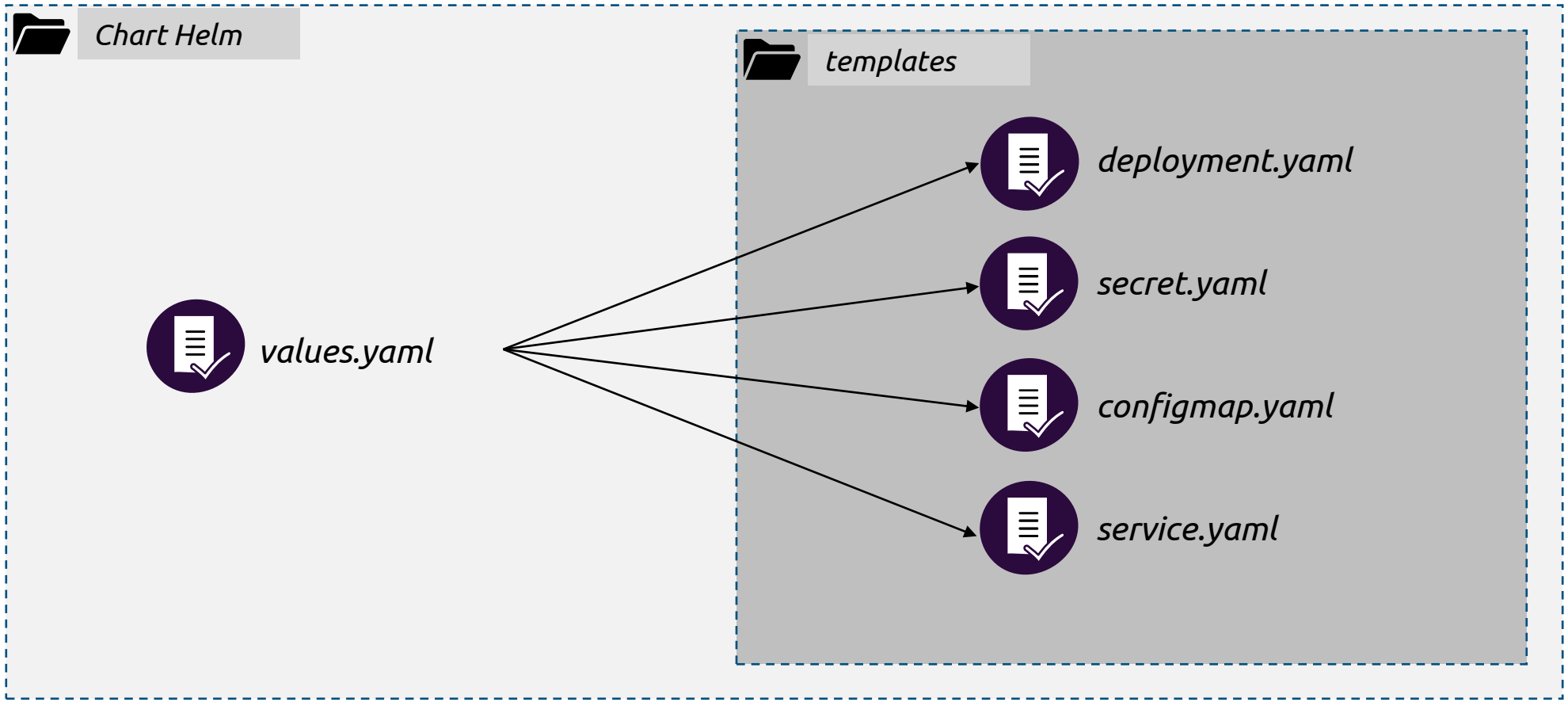
```
kubectl create –f <file>
```

```
kubectl apply –f <file>
```

```
kubectl delete –f <file>
```


DÉPLOYER UN WORKLOAD

Via le manager de paquet Helm & des ressources variabilisées



DÉPLOYER UN WORKLOAD

Commandes utiles de l'utilitaire helm

Aides

```
helm --help
```

Listings

```
helm ls
```

Installation

```
helm install <chart> --name <nom-chart> --namespace <nom-namespace>
```

Désinstallation

```
helm uninstall <nom-chart>
```

Télécharger la Helm Chart

```
helm fetch <chart>
```


Du code au cluster : industrialiser ses déploiements

Conteneuriser ses applications

- Concepts & bénéfices
- Création d'une image
- Lancement d'un conteneur

Orchestrer ses conteneurs via Kubernetes

- Concepts & bénéfices
- Déployer un cluster Kubernetes
- Les ressources Kubernetes
- Déployer des ressources sur le cluster

Industrialiser ses déploiements

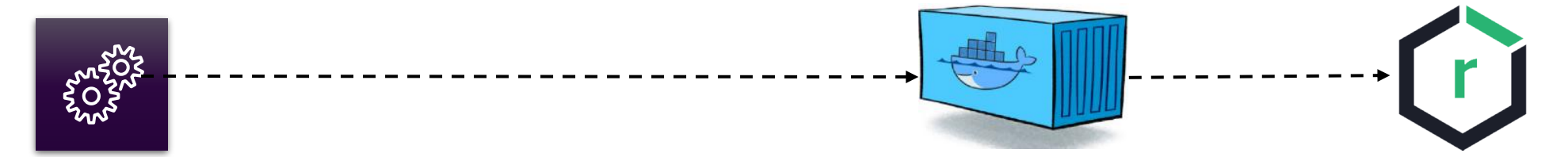
- Du code au cluster
- Automatiser la construction et le déploiement

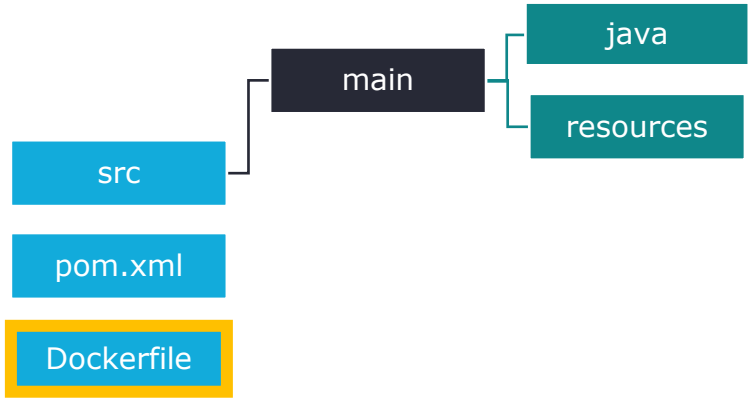
Exploiter & superviser ses applications en production

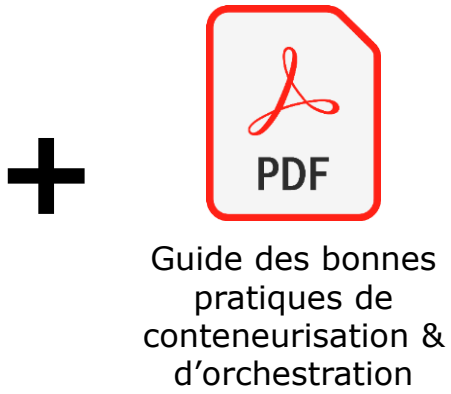
- Exploiter ses applications
- Superviser ses applications
- Collecter des métriques

DU CODE AU CLUSTER

Packager son application au sein d'un conteneur

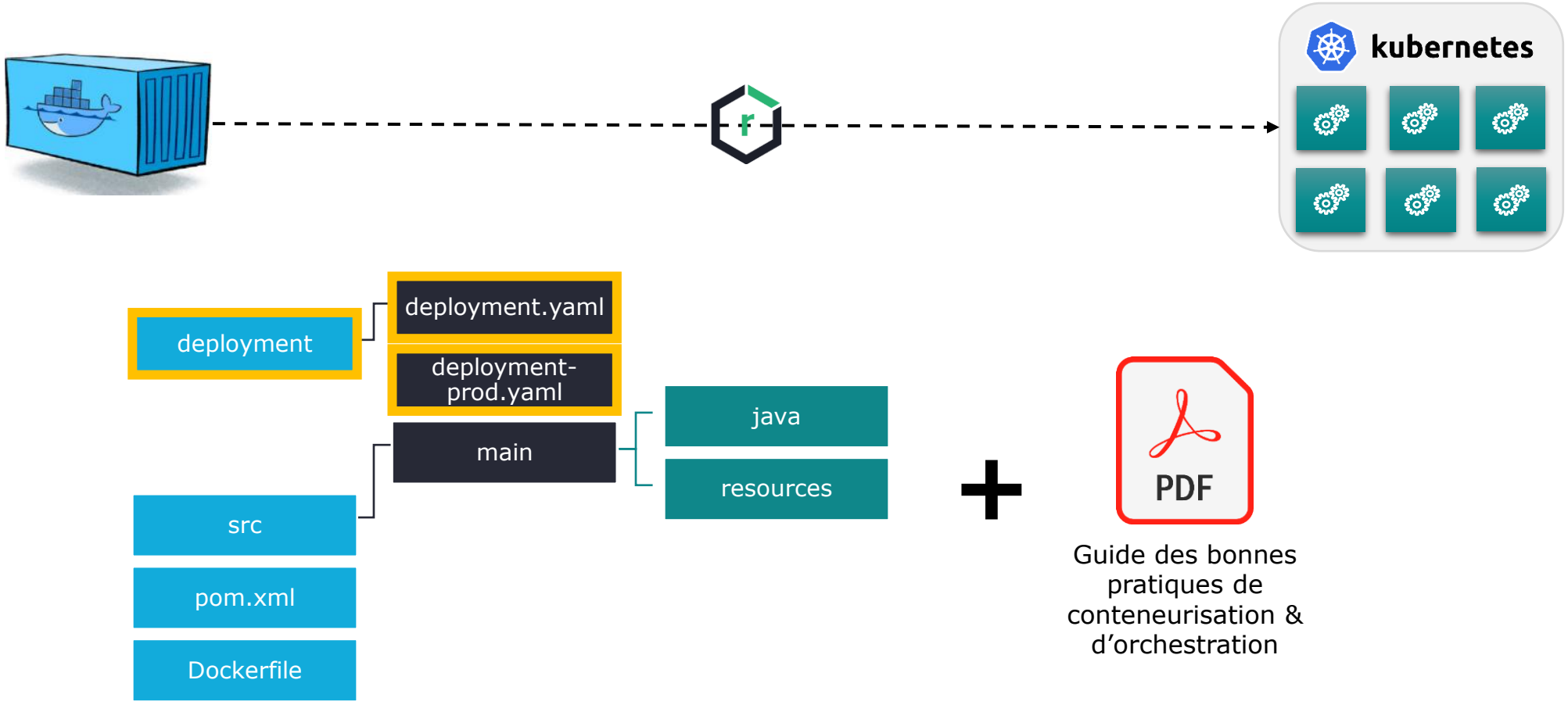






DU CODE AU CLUSTER

Héberger son conteneur sur des machines



DU CODE AU CLUSTER

Adapter son application pour l'orchestrer dans Kubernetes

Deployment

Fichier de déploiement de l'application
Appelle l'image
Contient replicas, volumes, probes, requests / limits

Ingress

Expose l'application vers l'extérieur
Contient nom de domaine

PersistentVolumeClaim

Demande un espace de stockage pour l'app
Contient StorageClass

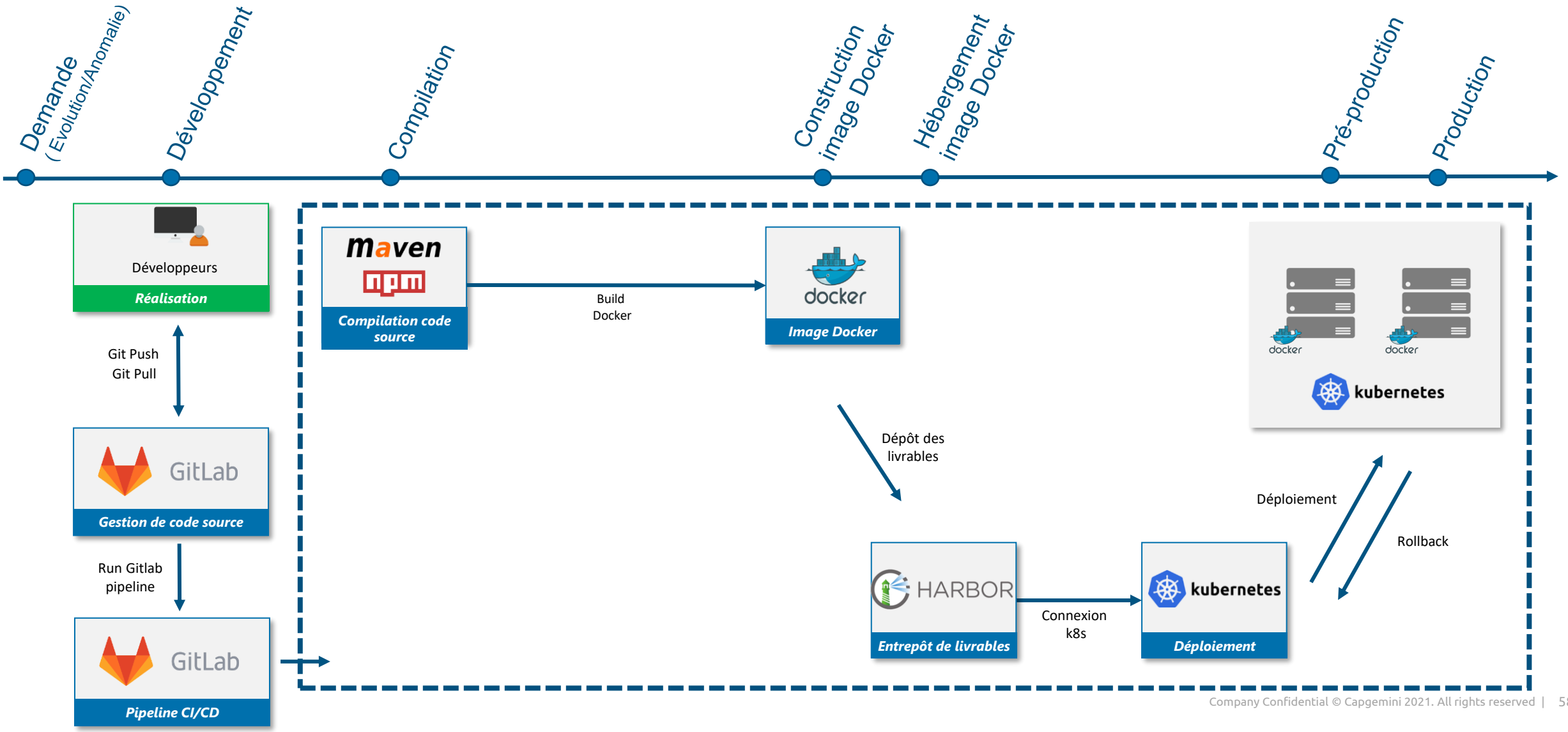
Service

Expose l'application dans le cluster
Contient port, protocole

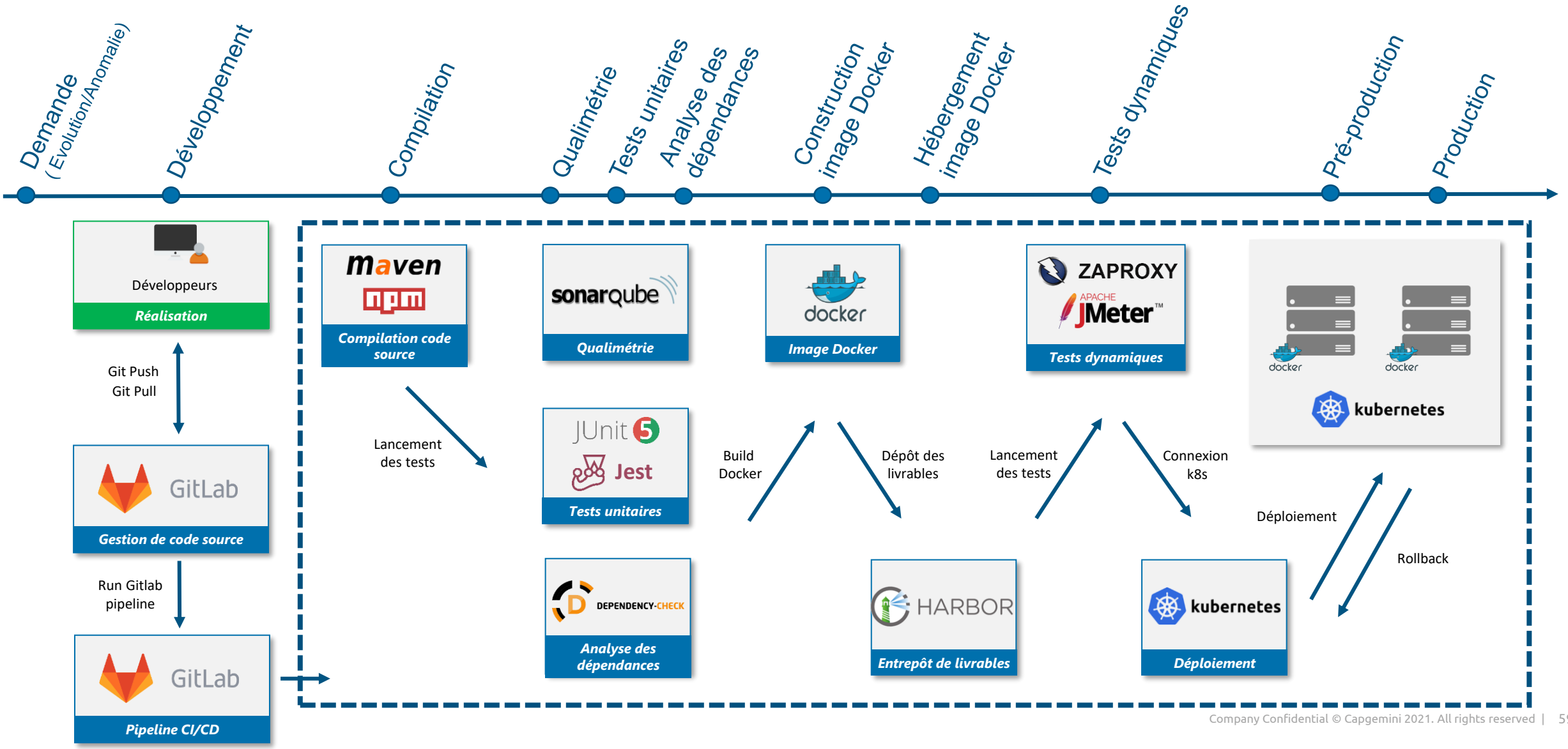
ConfigMap & Secrets

Monte la configuration dans l'app
Contient configuration, identifiants

AUTOMATISER LA CONSTRUCTION ET LE DÉPLOIEMENT

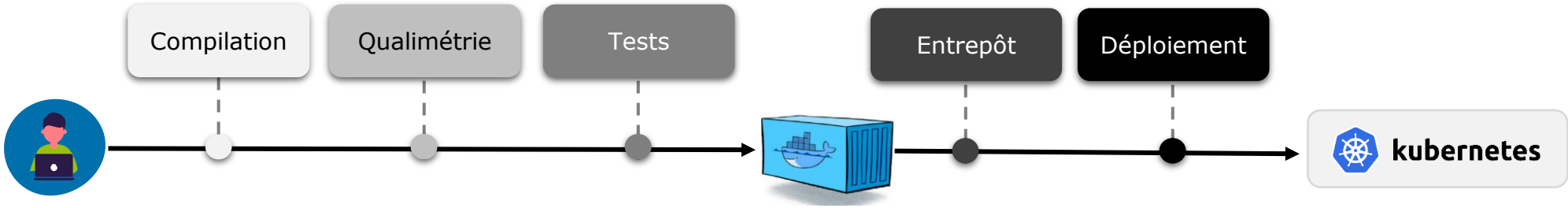


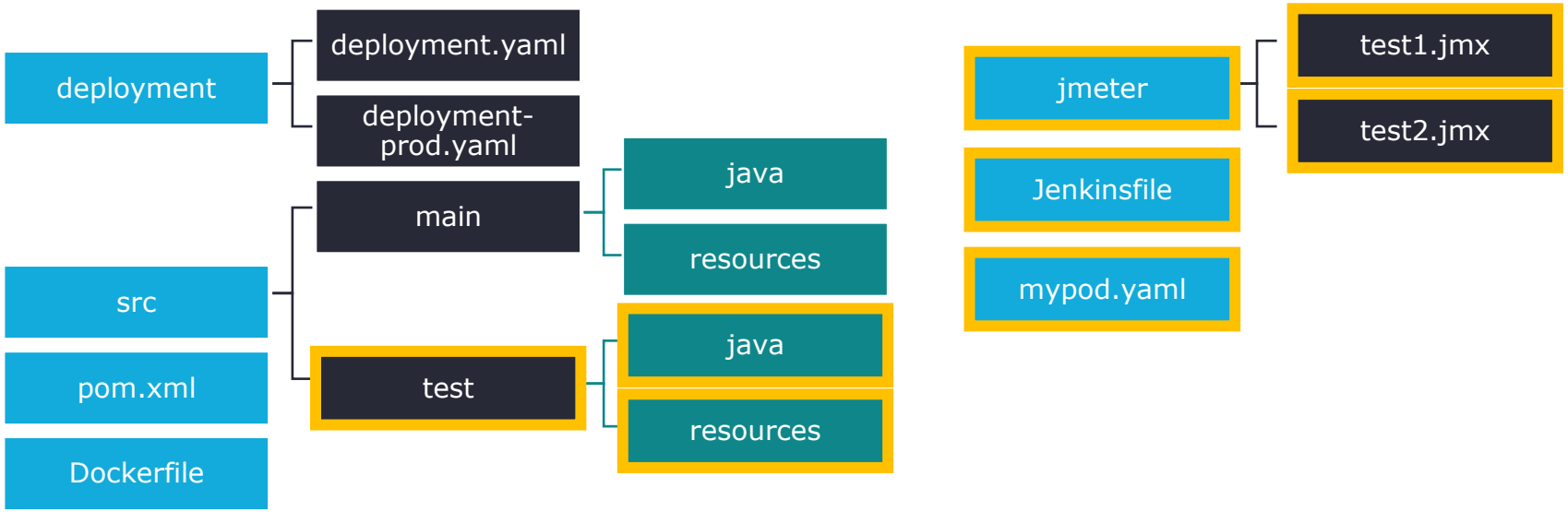
AUTOMATISER LA CONSTRUCTION ANALYTIQUE ET LE DÉPLOIEMENT



AUTOMATISER LA CONSTRUCTION ANALYTIQUE ET LE DÉPLOIEMENT

Adapter le contenu du dépôt de code





EXPLOITER SES APPLICATIONS EN PRODUCTION

Conteneuriser ses applications

- Concepts & bénéfices
- Création d'une image
- Lancement d'un conteneur

Orchestrer ses conteneurs via Kubernetes

- Concepts & bénéfices
- Déployer un cluster Kubernetes
- Les ressources Kubernetes
- Déployer des ressources sur le cluster

Industrialiser ses déploiements

- Du code au cluster
- Automatiser la construction et le déploiement

Exploiter & superviser ses applications en production

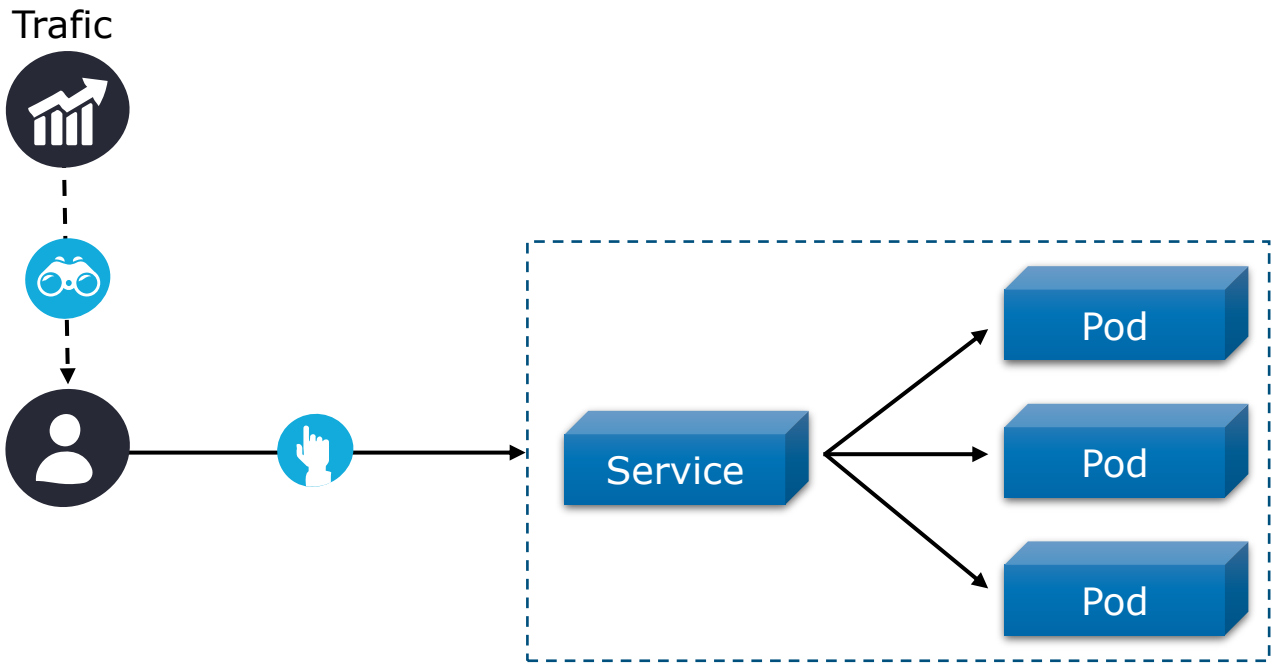
- Exploiter ses applications
- Superviser ses applications
- Collecter des métriques

EXPLOITER SES APPLICATIONS

Gérer le cycle de vie des applications en production

- ✓ **Résilience**
  - Redémarrage en cas d'échec
  - MAJ de la configuration
  - Suppression de l'instance

- ✓ **Passage à l'échelle**
  - Upscaling
  - Downscaling



SUPERVISER SES APPLICATIONS

Intérêts de la supervision

Visualiser l'état du système

- Vérifier le fonctionnement
- Suivre les exigences
- Détecter les incidents

Diagnostiquer & Investiguer

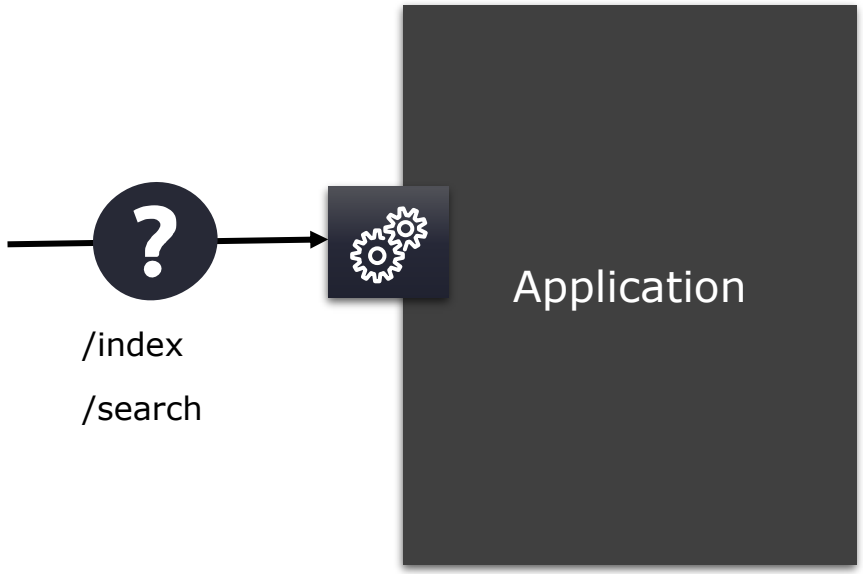
- Debugger
- Analyser les incidents (post-mortem)
- Analyser les tendances

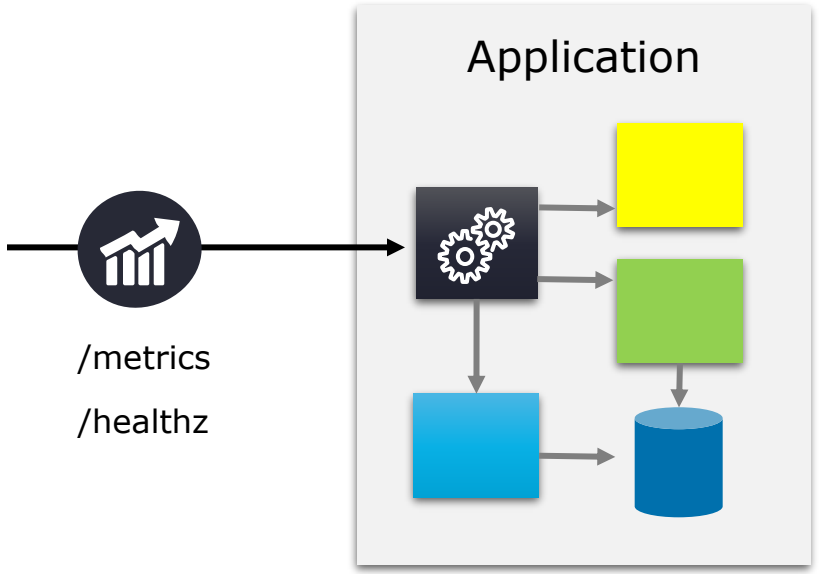
Corriger le système

- Implémenter des correctifs
- Anticiper les pannes

SUPERVISER SES APPLICATIONS

Black-box vs white-box monitoring





Le service fonctionne-t-il ?

Le nombre important de requêtes risque de saturer le service.

SUPERVISER SES APPLICATIONS

Supervision vs observabilité

> Observability is a measure of how well **internal states of a system** can be inferred from knowledge of its **external outputs**.

Wikipedia, *https://en.wikipedia.org/wiki/Observability*

> Monitoring tells you **whether the system works**. Observability lets you ask **why it's not working**.

Baron Schwarz

OBSERVABILITÉ EN PRODUCTION

Métriques, logging et tracing : la supervision n'est plus une option !

Healthchecks

Logs

Metrics

Distributed Tracing

kubernetes

https://blog.twitter.com/engineering/en_us/a/2013/observability-at-twitter.html

COLLECTER DES MÉTRIQUES

Métrique versus log

Métriques

Mesure numérique d'un fait pour un **instant donné**.

```
http_requests_total{code="200",handler="/",method="get"} 1
```

Logs

timestamp	2019-11-18T16:40:23+00:00
level	ERROR
log	Read time out
service	orders
team	alpha
commit	548a8c9a
build	548a8c9a.3
runtime	java-1.8.0_164
region	Paris-3
node	node_ea522aF28
userId	389
requestId	ef524aC42
method	GET
endpoint	/orders

COLLECTER DES MÉTRIQUES

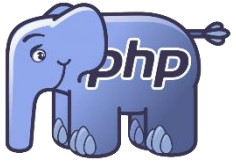
Instrumenter ses développements

Coder des métriques customs

Instrumenter à l'aide d'une librairie

https://prometheus.io/docs/instrumenting/clientlibs





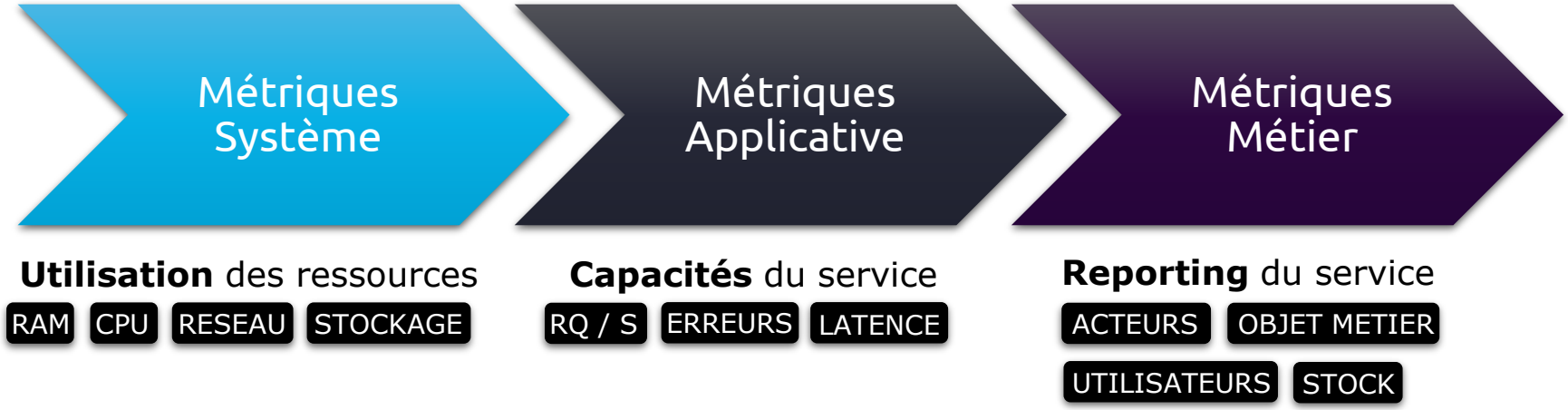

Utiliser un exporter / module existant

https://prometheus.io/docs/instrumenting/exporters



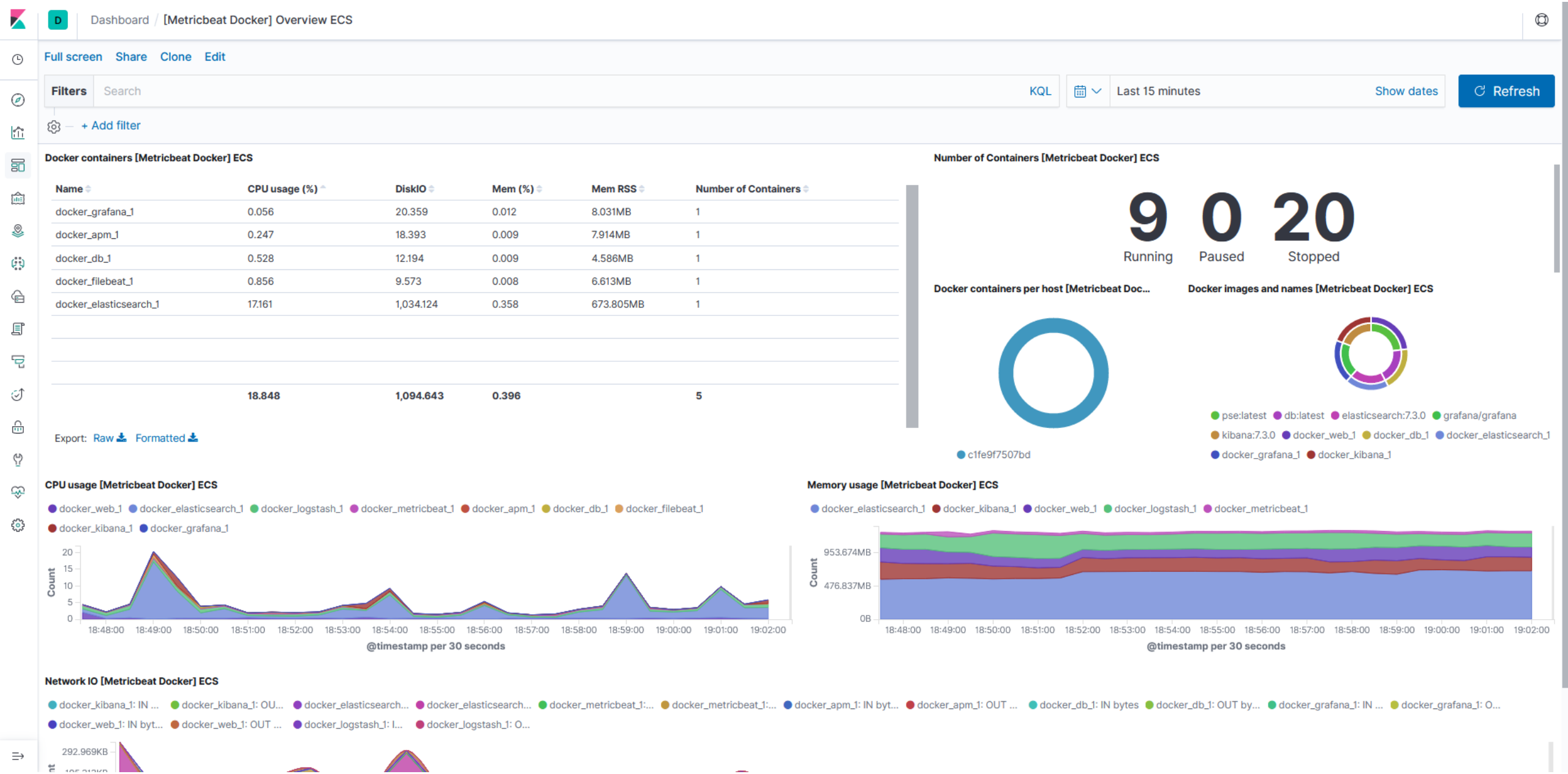

COLLECTER DES MÉTRIQUES

Typologie de métriques

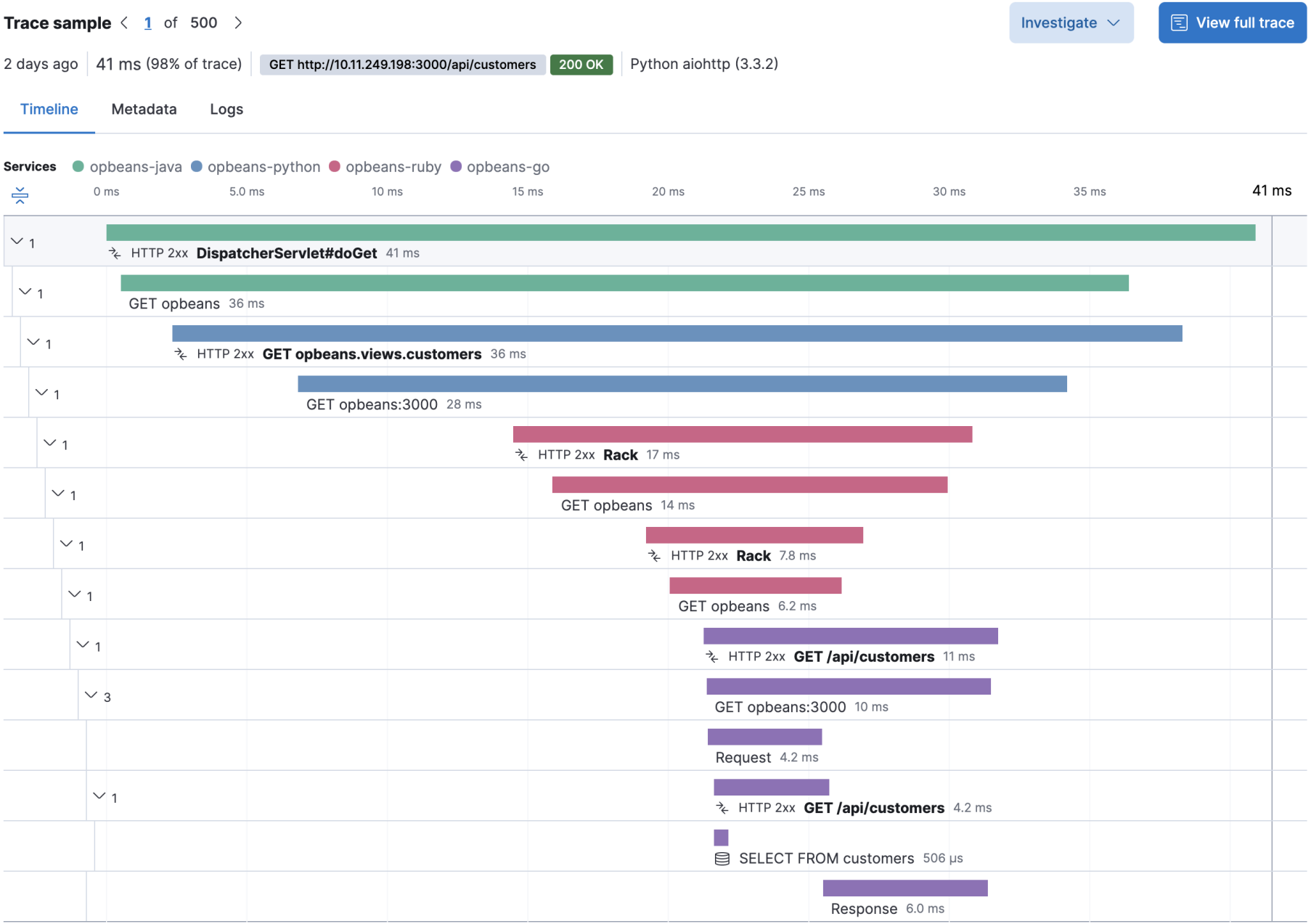


DASHBOARD GRAFANA

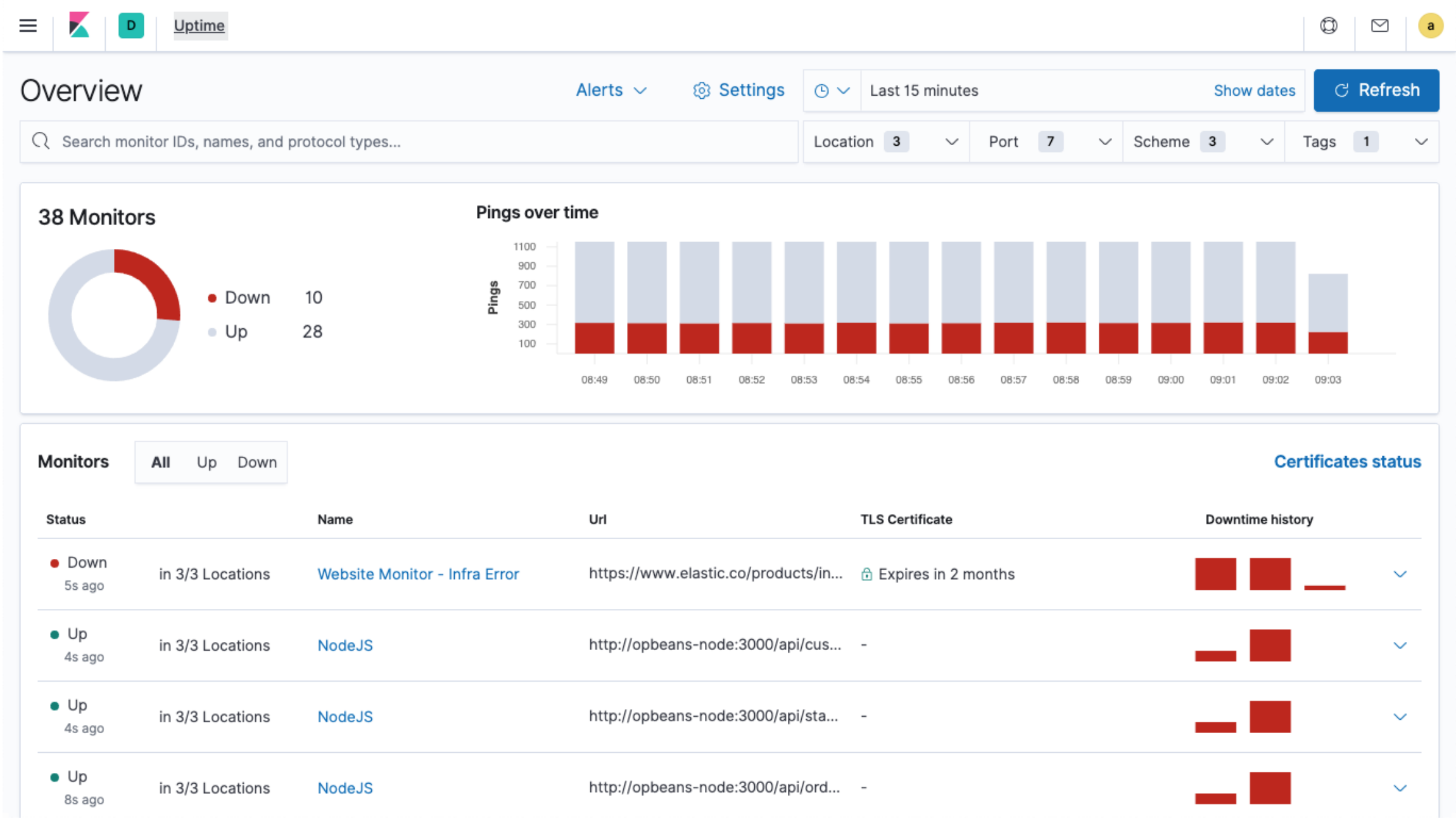
DASHBOARD KIBANA



TRACES KIBANA

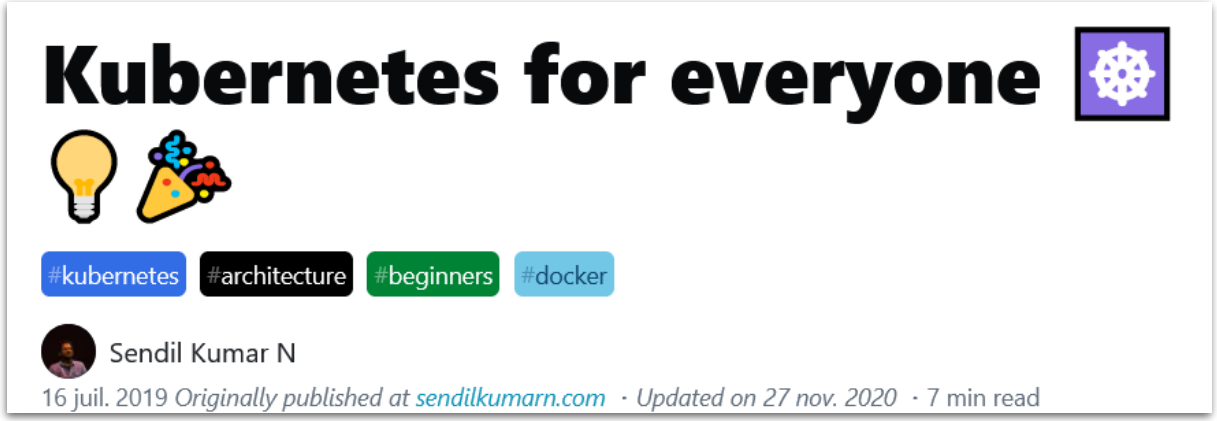


MONITORING KIBANA

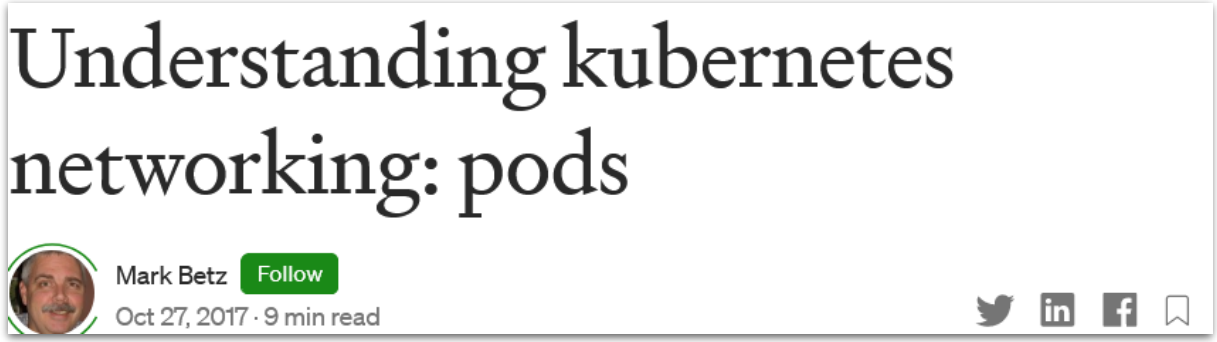


RECOMMANDATIONS

Comprendre le Kube



https://dev.to/sendilkumarn/kubernetes-for-everyone-opb



https://medium.com/google-cloud/understanding-kubernetes-networking-pods-7117dd28727

https://blog.kubernauts.io/

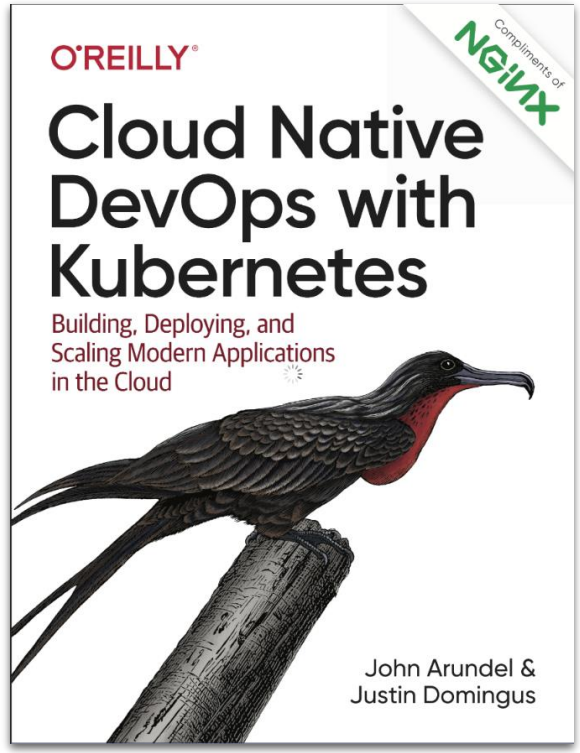
RECOMMANDATIONS

Implémenter le Kube

https://www.ssi.gouv.fr/guide/recommandations-de-securite-relatives-au-deploiement-de-conteneurs-docker/

learnk8s

https://learnk8s.io/production-best-practices



https://www.oreilly.com/library/view/cloud-native-devops/9781492040750/

Capgemini

MERCI POUR VOTRE ATTENTION

About Capgemini

Capgemini is a global leader in partnering with companies to transform and manage their business by harnessing the power of technology. The Group is guided everyday by its purpose of unleashing human energy through technology for an inclusive and sustainable future. It is a responsible and diverse organization of 270,000 team members in nearly 50 countries. With its strong 50 year heritage and deep industry expertise, Capgemini is trusted by its clients to address the entire breadth of their business needs, from strategy and design to operations, fuelled by the fast evolving and innovative world of cloud, data, AI, connectivity, software, digital engineering and platforms. The Group reported in 2020 global revenues of €16 billion.

Get the Future You Want | www.capgemini.com