Fondamentaux de Kubernetes

Benoit Lange

benoit@lange.xyz

Objectifs pédagogiques

- Les distributions
- Installation
- kubectl
- Autre outils

Des distributions plus ou moins avancés

Red Hat OpenShift

SUSE Rancher

Canonical Kubernetes

Mirantis Kubernetes Engine (anciennement Docker Enterprise)

VMware Tanzu (anciennement Pivotal)

Platform9 Managed Kubernetes

Giant Swarm

Portainer

Trois exemples

Kubernetes (K8s)

- Orchestrateur de conteneurs **open source** (CNCF).
- Gère déploiement, scalabilité, auto-réparation, supervision.
- “Vanilla” = version upstream, très personnalisable mais brut.

OpenShift (Red Hat)

- Distribution Kubernetes enrichie.
- Ajoute sécurité (SELinux, SCC), CI/CD (Tekton), registry intégrée, console web.
- Orientée **entreprise** + support commercial Red Hat.

Rancher (SUSE)

- Plateforme de **gestion multi-clusters Kubernetes**.
- S’appuie sur Kubernetes (vanilla ou k3s).
- Apporte : gouvernance, RBAC centralisé, monitoring, marketplace d’apps.

Comparatif

Critère	Kubernetes (Vanilla)	OpenShift (Red Hat)	Rancher (SUSE)
Nature	Orchestrateur CNCF	Distribution Kubernetes enrichie	Plateforme de gestion K8s
Installation	kubeadm, kOps, k3s, etc.	Automatisée (installateurs Red Hat)	Déploie et administre K8s
Sécurité	Basique (RBAC, PSP/OPA)	Renforcée (SELinux, SCC)	Centralisation RBAC multi-clusters
CI/CD	Non intégré	Tekton, ArgoCD intégrés	Marketplace d'outils
Registry	Externe (Harbor, Docker Hub)	Intégrée	À configurer
Console	CLI `kubectl` / outils tiers	Console web complète + CLI `oc`	UI web Rancher (multi-clusters)
Cas d'usage	Flexible, modulaire	Entreprise, sécurité, support SLA	Multi-cluster, hybride, edge

Installation avec kubeadm

- Installer un runtime de conteneurs (ex : containerd).
- Installer kubeadm, kubelet, kubectl :

```
sudo apt-get update && sudo apt-get install -y kubelet kubeadm kubectl
```

- Initialiser le cluster (sur le master) :

```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
```

- Configurer kubectl :

```
mkdir -p $HOME/.kube
cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
```

- Déployer un plugin réseau (CNI) :

```
kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
```

Des distribution légère

Problèmes de Kubernetes “complet”

- Installation **complexe** (kubeadm, CNI, certificats…)
- **Consommation importante** en ressources (CPU/RAM/disque)
- Non adapté pour des environnements contraints (IoT, edge, petites VM)
- Pas toujours nécessaire d’avoir toute la stack “enterprise”

Objectif des distributions légères

- Fournir un **Kubernetes conforme CNCF** mais simplifié
- Réduire l’empreinte mémoire et CPU
- Installation rapide (souvent en **1 ligne**)
- Parfait pour **apprendre, tester, prototyper** ou déployer en **environnements limités**

Example

k3s (Rancher/SUSE)

- Kubernetes allégé (<100 Mo binaire)
- Intègre etcd simplifié (SQLite par défaut)
- Idéal pour **IoT, edge, dev rapide**

microk8s (Canonical)

- Installation via snap install microk8s
- Modulaire (activer/désactiver DNS, ingress, registry, etc.)
- Bon choix pour du **dev local ou petites équipes**

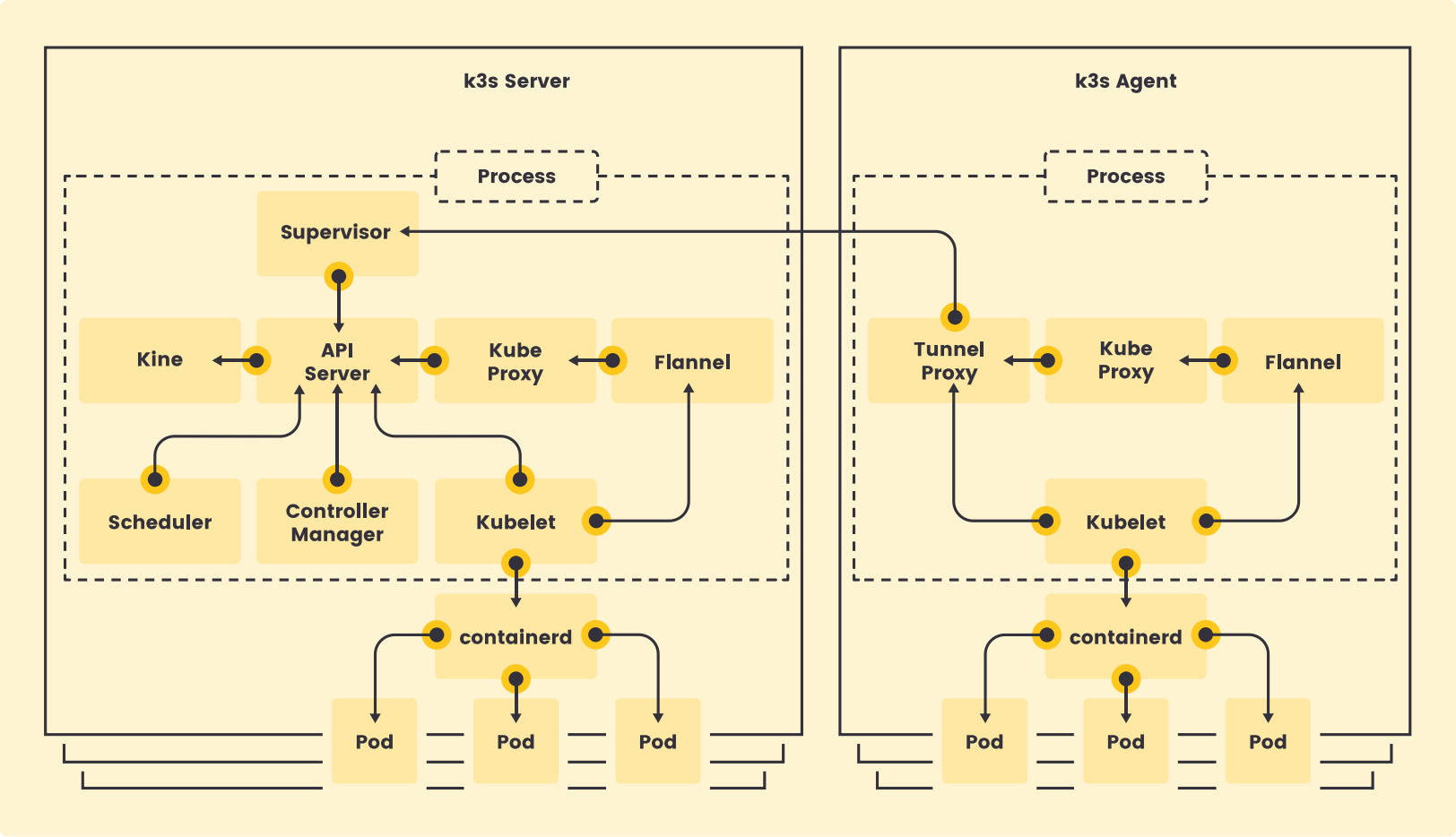
minikube / kind

- Pour apprendre Kubernetes **en local**
- Fonctionne dans une VM ou dans Docker
- Très utile en **formation** et **CI/CD pipelines**

Le cas de k3s

curl -sfL https://get.k3s.io | sh -

sudo k3s kubectl get node



Qu’est-ce que kubectl

Définition

- **kubectl** est la **CLI officielle** pour interagir avec un cluster Kubernetes.
- Communique avec l’**API Server** du Control Plane.
- Utilise le fichier **kubeconfig** pour savoir à quel cluster se connecter.

Rôle

- Déploiement et gestion des ressources.
- Consultation de l’état du cluster.
- Exécution de commandes/debug dans les Pods.

Syntaxe et commandes de base

Structure générale

- kubectl <resource> <action> [options]

Exemples

- kubectl get nodes # Liste les nœuds du cluster
- kubectl get pods # Liste les pods
- kubectl describe pod X # Détails d’un pod
- kubectl logs X # Voir les logs
- kubectl exec -it X -- sh # Entrer dans un conteneur

Configuration et contextes

kubeconfig

- Fichier de config (par défaut : ~/.kube/config).
- Contient : clusters, utilisateurs, contextes.

Contextes

- Permettent de gérer plusieurs clusters/environnements (dev, prod…).
  - kubectl config get-contexts
  - kubectl config use-context dev-cluster

Namespaces

- kubectl agit par défaut sur default.
- On peut cibler un namespace :
  - kubectl get pods -n kube-system