



# Kubernetes vs OpenShift

## Architecture et Comparaison

Présentation technique destinée aux experts DevOps

Auteur : Ahmed ZIAD

Date : 21/07/2025

# Sommaire

---

- 1 Introduction à OpenShift
- 2 Pourquoi choisir OpenShift ?
- 3 OpenShift dans l'écosystème DevOps
- 4 Architecture générale d'OpenShift
- 5 Composants principaux d'OpenShift
- 6 Flux de déploiement sur OpenShift
- 7 Qu'est-ce que Kubernetes ?
- 8 Architecture de Kubernetes
- 9 OpenShift vs Kubernetes : Principales différences
- 10 Gestion de la sécurité
- 11 Gestion des mises à jour et du cycle de vie
- 12 Avantages d'OpenShift par rapport à Kubernetes natif
- 13 Limites et contraintes d'OpenShift
- 14 Cas d'usage et retours d'expérience
- 15 Conclusion et perspectives

# Introduction à OpenShift

## Qu'est-ce qu'OpenShift ?

OpenShift est une plateforme d'orchestration de conteneurs enterprise-ready développée par Red Hat, construite sur Kubernetes et optimisée pour le développement et le déploiement d'applications.

## Caractéristiques principales

- ☰ Distribution Kubernetes entreprise avec support commercial
- 🛡️ Sécurité renforcée et authentification intégrée
- 🔗 CI/CD intégré et gestion du cycle de vie applicatif
- 🔧 Interface utilisateur riche et outils de développement



### Développé par Red Hat

Leader du marché des solutions open source pour l'entreprise

### Positionnement

- ✓ Solution complète PaaS/CaaS
- ✓ Orientée entreprise et production
- ✓ Déploiement multi-environnements: on-premise, cloud public, hybride

# Pourquoi choisir OpenShift ?

## Défis de l'entreprise moderne

Les organisations font face à des exigences croissantes pour délivrer rapidement des applications fiables et sécurisées tout en optimisant les ressources IT.

## Réponses apportées par OpenShift



### Automatisation avancée

Automatisation des builds, déploiements et scaling des applications



### Opérationnalisation du DevOps

Intégration native des pipelines CI/CD et des workflows d'équipe



### Standardisation

Environnements cohérents du développement à la production



### Sécurité Enterprise

Sécurité renforcée à tous les niveaux de la pile applicative



### Accélération du TTM

Réduction significative du temps de mise sur le marché des nouvelles applications et fonctionnalités



### ROI démontré

Amélioration de la productivité des développeurs de 35% et réduction des coûts opérationnels de 20%



### Support Enterprise

Assistance 24/7 par Red Hat, SLA garantis et roadmap prévisible pour une tranquillité d'esprit



# OpenShift dans l'écosystème DevOps

## Intégration DevOps

OpenShift s'intègre au cœur de l'écosystème DevOps comme plateforme de déploiement et d'orchestration, servant de lien entre le développement et les opérations.

## Chaînes CI/CD

- 🔗 Pipeline intégré pour gérer le cycle complet : build, test, deploy, monitor
- 🔄 Déploiement continu avec rollback automatisé
- ⚙️ Workflows automatisés pour une livraison rapide et fiable

## Intégration avec les outils DevOps



### Jenkins

Intégration native avec les pipelines Jenkins pour l'automatisation CI/CD



### Git

Webhook et déclencheurs automatisés depuis GitHub, GitLab, Bitbucket



### Tekton

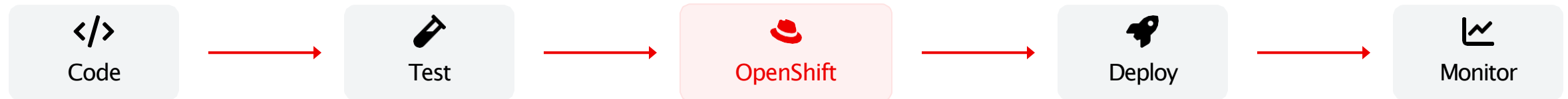
Pipelines cloud-native pour Kubernetes



### Sécurité

Intégration avec outils de sécurité et scanning (Quay, SonarQube)

## Flux de travail DevOps avec OpenShift



# Architecture générale d'OpenShift

## Une plateforme bâtie sur Kubernetes

OpenShift étend Kubernetes avec des composants additionnels pour fournir une plateforme complète d'applications conteneurisées pour l'entreprise.

### Master Nodes

-  **API Server**  
Point d'entrée pour toutes les requêtes
-  **etcd**  
Stockage distribué des données du cluster
-  **Controller Manager**  
Gestion des ressources et états
-  **Scheduler**  
Placement des charges de travail

### **Console Web**

Interface utilisateur




### **Registry**

Stockage d'images

### **Router**

Gestion du trafic

### Worker Nodes

-  **Kubelet**  
Agent d'exécution sur chaque nœud
-  **Kube-proxy**  
Règles de réseau et services
-  **CRI-O**  
Runtime de conteneurs
-  **Pods**  
Unités d'exécution

### Valeur ajoutée d'OpenShift

- ◆ Gestion intégrée des utilisateurs et de l'authentification
- ◆ Mécanismes avancés de routage et d'exposition de services

### Modèle en couches

- ◆ Infrastructure (bare metal, virtualisation, cloud)
- ◆ Kubernetes (orchestration de base)

# Composants principaux d'OpenShift



## Kube-apiserver

Point d'entrée pour l'API REST qui valide et configure les données des objets API (pods, services, etc.)



## Control Plane

Ensemble des composants de gestion du cluster, incluant le master et ses services associés



## Scheduler

Attribue les pods aux nœuds de travail selon les contraintes (ressources, affinité, etc.)



## Etcd

Base de données clé-valeur distribuée qui stocke toutes les données du cluster (état, configuration)



## Routers

Gèrent le trafic entrant dans le cluster et acheminent les requêtes vers les services appropriés



## Registry

Registre d'images conteneurs intégré pour stocker, récupérer et partager des images au sein du cluster



## Console Web

Interface utilisateur web avancée pour gérer applications, projets et ressources du cluster



## Operators

Extensions qui automatisent la gestion des applications et services complexes sur Kubernetes

## Architecture intégrée

Ces composants fonctionnent ensemble pour fournir une plateforme complète et intégrée qui étend les fonctionnalités de base de Kubernetes avec des outils orientés entreprise et développeurs.

# Flux de déploiement sur OpenShift

Cycle de vie du déploiement d'une application - du code source à la production



## Code Source

Git, SVN, Archives



## Build

S2I, Docker Build,  
Jenkins



## Image

Container Image  
Registry



## Déploiement

Pods, Services, Routes



## Monitoring

Prometheus, Grafana



## Production

Scaling, HA, Rollbacks

## Caractéristiques du flux de déploiement OpenShift

### Déploiement continu

Intégration CI/CD automatisée avec triggers de build et déploiement

### Routage intégré

Exposition automatique des services via Routes et HAProxy

### Rollbacks simplifiés

Retour rapide à une version antérieure stable

### Sécurité native

Contrôle d'accès, scanning d'images et politiques de sécurité







# Qu'est-ce que Kubernetes ?

## Définition

Kubernetes (K8s) est une plateforme open-source d'orchestration de conteneurs conçue pour automatiser le déploiement, la mise à l'échelle et la gestion des applications conteneurisées.

## Principes fondamentaux

-  Architecture déclarative basée sur l'état souhaité
-  Orchestration distribuée et auto-guérison
-  Abstraction de l'infrastructure sous-jacente
-  Scaling automatique et équilibrage de charge







## Origine & Historique

Développé par Google en 2014





Donné à la Cloud Native Computing Foundation (CNCF) en 2015

## Écosystème & Communauté




-  Communauté mondiale très active
-  Large écosystème d'extensions et d'outils
-  Nombreuses distributions disponibles
-  Support natif par tous les grands clouds

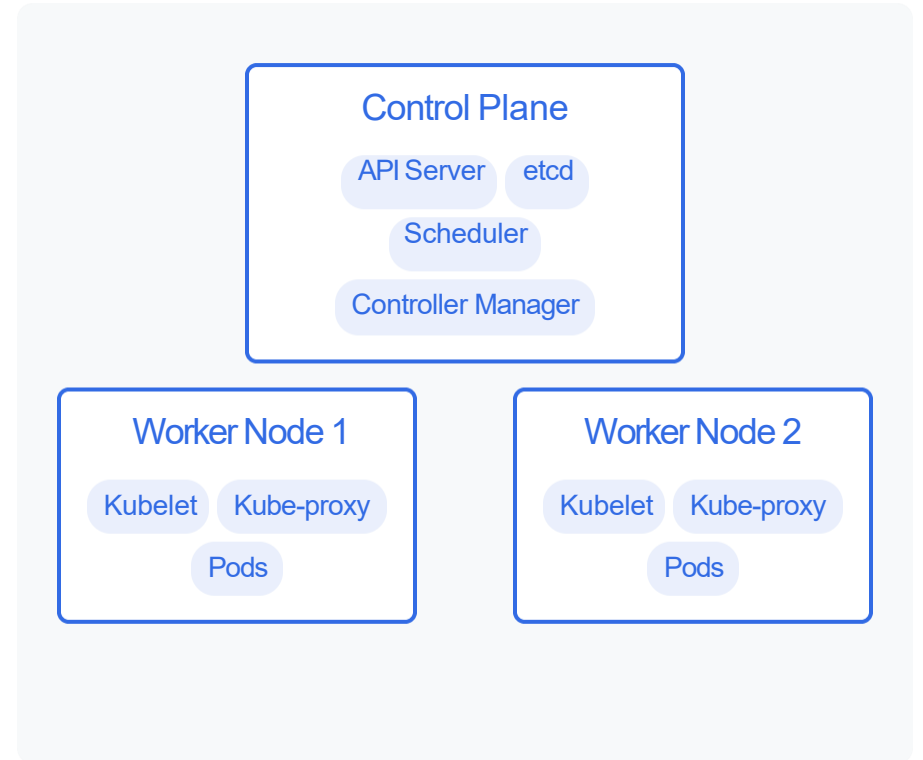
# Architecture de Kubernetes

## Control Plane (Master Node)

-  **API Server**  
Point d'entrée REST pour toutes les opérations
-  **etcd**  
Stockage clé-valeur des données du cluster
-  **Scheduler**  
Assigne les pods aux nœuds
-  **Controller Manager**  
Gère les contrôleurs de ressources

## Worker Nodes

-  **Kubelet**  
Agent qui assure l'exécution des pods sur le nœud
-  **Kube-proxy**  
Gère les règles réseau et le routage du trafic
-  **Container Runtime**  
Exécution des conteneurs (Docker, containerd, CRI-O)















## Principes clés

**Déclaratif** : Vous déclarez l'état souhaité, Kubernetes s'occupe de sa réalisation

**Auto-guérison** : Surveillance et remplacement automatique des composants défectueux

**Scalabilité** : Mise à l'échelle horizontale des applications

# OpenShift vs Kubernetes : Principales différences

Caractéristiques	 OpenShift	 Kubernetes
Modèle commercial	 Solution commerciale avec support Red Hat	 Projet open source géré par la CNCF
Fonctionnalités intégrées	✓ CI/CD, registry, monitoring, logging, routage intégrés	✓ Fonctionnalités de base pour l'orchestration de conteneurs
Gestion des builds	✓ Builds intégrés (Source-to-Image, Dockerfile)	✗ Pas de fonctionnalité native (solutions tierces requises)
Sécurité	 Modèle de sécurité renforcé (SCC, RBAC), authentification intégrée	 RBAC basique, sans authentification intégrée
Interface utilisateur	 Console web riche, CLI dédiée (oc)	 Dashboard basique, CLI kubectl
Gestion du cycle de vie	 Opérateurs, mises à jour simplifiées, versions stables	 Mises à jour manuelles, rythme de releases rapide
Coût et licences	 Licence commerciale avec support	 Gratuit, support communautaire ou distributions payantes

# Gestion de la sécurité

## OpenShift

Approche de sécurité renforcée et orientée entreprise

### Gestion des identités

- Intégration LDAP/Active Directory native
- OAuth et SSO intégrés
- Authentification multifacteur

### Contrôle d'accès

- RBAC étendu avec modèle de permissions hiérarchiques
- Security Context Constraints (SCC)
- Isolation et séparation des projets

### Sécurité conteneurs

- Scanner de vulnérabilités intégré
- Politiques d'admission avancées
- Signature et validation d'images

## Kubernetes

Sécurité modulaire nécessitant une configuration avancée

### Gestion des identités

- Pas d'authentification intégrée
- Nécessite des solutions tierces
- Flexible mais complexe à configurer

### Contrôle d'accès

- RBAC standard
- Pod Security Policies (deprecated)
- Pod Security Standards

### Sécurité conteneurs

- Requiert des outils externes pour scanning
- Admission Controllers configurables
- OPA/Gatekeeper pour validation

# Gestion des mises à jour et du cycle de vie

## OpenShift

Approche entreprise avec cycle de vie prévisible et stabilité renforcée.

### Cycle de mise à jour

- Versions majeures tous les ~6 mois
- Support étendu jusqu'à 18 mois
- Mises à jour incrémentales simplifiées (z-stream)

### Operators Framework

- Automatisation des opérations day-2
- OperatorHub intégré
- Gestion déclarative des applications

## Kubernetes natif

Évolution rapide guidée par la communauté CNCF avec innovations fréquentes.

### Cycle de mise à jour

- 3-4 versions majeures par an
- Support de ~9 mois par version
- Mises à jour manuelles plus fréquentes

### Gestion du cycle de vie

- Dépend des distributions ou solutions tierces
- Outils externes pour l'automatisation
- Responsabilité de l'équipe d'exploitation

## Avantages de l'approche OpenShift

- ✓ Prévisibilité pour l'environnement production
- ✓ Simplification des mises à niveau complexes
- ✓ Support Red Hat pour la résolution des problèmes

# Avantages d'OpenShift par rapport à Kubernetes natif

OpenShift enrichit Kubernetes avec des fonctionnalités essentielles pour les entreprises, offrant une valeur ajoutée significative pour les environnements de production.



## Sécurité renforcée

- Authentification et autorisation intégrées (LDAP, SSO)
- Security Context Constraints (SCC)
- Scanning automatique des images



## Productivité améliorée

- Fonctionnalités CI/CD prêtes à l'emploi
- Source-to-Image (S2I)
- Templates et Helm Charts intégrés



## Support entreprise

- SLA de niveau entreprise par Red Hat
- Documentation complète et formation
- Roadmap claire et prévisible



## Intégration facilitée

- Intégration avec écosystème Red Hat
- Connecteurs prêts à l'emploi (middleware, BDD)
- Support multi-cloud et hybride



## Interface utilisateur complète

- Console web intuitive et riche
- Tableaux de bord de monitoring intégrés
- Visualisation des ressources et métriques



## Écosystème enrichi

- OperatorHub et catalogue de services
- Registry intégré
- Extensions spécifiques (serverless, service mesh)

# Limites, Cas d'usage et Conclusion

## ⚠ Limites d'OpenShift

- Coût des licences et support plus élevé
- Complexité accrue pour certaines configurations
- Ressources matérielles plus importantes
- Courbe d'apprentissage pour les équipes
- Personnalisation parfois contrainte

## 🏠 Cas d'usage

### Finance

Déploiement sécurisé d'applications critiques

### Industrie

Transformation numérique et edge computing

### Secteur public

Modernisation des infrastructures et des services

### Télécom

Infrastructure 5G et services distribués

## 🚩 Conclusion

OpenShift représente une valeur ajoutée significative pour les entreprises cherchant une plateforme conteneurisée robuste et supportée.

### Quand choisir OpenShift ?

- ✓ Projets d'entreprise nécessitant support et SLA
- ✓ Besoins de sécurité et conformité renforcés

### Perspectives

- ➔ Développement du multi-cloud et hybrid-cloud
- ➔ Intégration croissante avec l'IA/ML

"La plateforme adaptée dépend de votre contexte et de vos priorités."

L'essentiel est de choisir une solution alignée avec votre stratégie technique et vos compétences internes.