Architectures distribuées

Master 1 génie de l'informatique logicielle

Déploiement et kubernetes

Présenté par : Lydia KAHOUADJI & Luc Perin PANTA

20 mars 2024

Sommaire

Problématique

Face à la complexité croissante des architectures logicielles, les équipes de développement et d'exploitation sont confrontées à des défis majeurs. Comment assurer un déploiement efficace, une scalabilité transparente et une gestion cohérente des microservices dans un environnement aussi dynamique que Kubernetes ? Cette problématique constitue le point de départ de notre exploration.

Contexte

Contexte

L'évolution rapide des applications modernes nécessite des solutions robustes de déploiement et de gestion. Kubernetes, un orchestrateur de conteneurs open-source, s'est imposé comme une technologie clé pour automatiser le déploiement, la mise à l'échelle et la gestion des applications conteneurisées.

Définition de Kubernetes

Kubernetes

Est un système open-source d'orchestration de conteneurs, initialement développé par Google et maintenant géré par la Cloud Native Computing Foundation (CNCF). Il offre une plateforme extensible pour automatiser le déploiement, la mise à l'échelle et la gestion des applications conteneurisées. Kubernetes simplifie la gestion des applications distribuées, offrant une solution robuste pour les défis du déploiement moderne.

Caractéristiques principales

- **Orchestration :** Coordination automatisée du déploiement des conteneurs.
- **Scalabilité :** Possibilité d'ajuster dynamiquement le nombre de répliques d'une application.
- **Gestion des Services :** Services, réseaux et stockage définis de manière déclarative.
- **Portabilité :** Fonctionne sur diverses infrastructures, locales ou cloud.

Concepts de base de Kubernetes

1. Nœud (Node)

- Un serveur physique ou virtuel sur lequel Kubernetes s'exécute.
- Exemple : Un serveur physique ou une machine virtuelle dans un centre de données.

2. Cluster

- Un ensemble de nœuds qui travaillent ensemble.
- Exemple : Un cluster Kubernetes composé de plusieurs nœuds.

3. Pod

- La plus petite unité déployable dans Kubernetes.
- Exemple : Un Pod contenant un conteneur web et un conteneur de base de données.

Concepts de base de Kubernetes (suite)

4. Service

- Abstraction qui définit un ensemble de Pods et une politique d'accès à ces Pods.
- Exemple : Un service exposant une application web aux utilisateurs.

5. Déploiement (Deployment)

- Définit l'état souhaité de l'application et permet à Kubernetes de le maintenir.
- Exemple : Un déploiement gérant plusieurs répliques d'une application.

Concepts avancés de Kubernetes

1. RéplicaSet

- Garantit un nombre spécifié de réplicas (instances) de Pods en cours d'exécution.
- Exemple : Un RéplicaSet assurant trois instances d'un service backend.

2. StatefulSet

- Maintient un état persistant pour chaque Pod.
- Exemple : Un StatefulSet pour une base de données distribuée.

3. ConfigMap et Secret

- Stockent la configuration et les secrets séparément des Pods.
- Exemple : Utilisation de ConfigMap pour injecter des paramètres de configuration dans un Pod.

Helm et Kubernetes

Helm

- Helm est un gestionnaire de packages pour Kubernetes.
- Facilite le déploiement et la gestion d'applications Kubernetes complexes.
- Utilise des charts (paquets) pour définir, installer et mettre à jour des ressources Kubernetes.

Utilisation de Helm

- **Création d'un Chart :** Structuration des ressources Kubernetes dans un package réutilisable.
- **Installation :** Helm install my-chart pour déployer une application avec ses dépendances.
- **Mise à jour :** Helm upgrade pour mettre à jour une application en cours d'exécution.
- **Suppression :** Helm uninstall pour supprimer une application et ses ressources.

Cas Réel de Déploiement avec Kubernetes

Étape 1 : Analyse des Besoins

- Application web avec un service web Node.js.
- Base de données Redis pour le stockage des données.

Cas Réel de Déploiement avec Kubernetes

Étape 2 : Conteneurisation

- Conteneuriser l'application Node.js et la base de données Redis.
- Utiliser Docker pour créer les images des conteneurs.
  Exemple Dockerfile pour l'application Node.js :

```
FROM node :14
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
CMD ["npm", "start"]
```

Cas Réel de Déploiement avec Kubernetes (suite)

Étape 3 : Configuration YAML

- Créer des fichiers de configuration YAML pour décrire les ressources Kubernetes.
  Exemple YAML pour le Pod de l'application Node.js :
  apiVersion : v1
  kind : Pod
  metadata :
  name : nodejs-app
  spec :
  containers :
  - name : nodejs-container
  image : your-registry/nodejs-app :latest

Cas Réel de Déploiement avec Kubernetes

Étape 4 : Déploiement Initial

- Utiliser kubectl pour déployer les Pods sur un cluster Kubernetes.
  Commande pour créer le Pod :
  kubectl apply -f nodejs-pod.yaml
- Vérifier que les Pods sont en cours d'exécution avec
  kubectl get pods

Cas Réel de Déploiement avec Kubernetes (suite)

Étape 5 : Orchestration

- Utiliser un Deployment pour assurer que l'application Node.js a le nombre requis de répliques.

Exemple YAML pour le Deployment

```
apiVersion : apps/v1
kind : Deployment
metadata :
    name : nodejs-deployment
spec :
    replicas : 3
    selector :
        matchLabels :
            app : nodejs
template :
     metadata :
     labels :
        app : nodejs
spec :
    containers :
        - name : nodejs-container
        image : your-registry/nodejs-app :latest
```

Cas Réel de Déploiement avec Kubernetes

Étape 6 : Surveillance et Ajustement

- Configurer Prometheus pour surveiller les performances de l'application et de la base de données.
- Ajuster dynamiquement le nombre de répliques en fonction du trafic avec kubectl scale.
  Commande pour ajuster le nombre de répliques :
  kubectl scale deployment nodejs-deployment –replicas=5

Cas Réel de Déploiement avec Kubernetes (suite)

Étape 7 : Mises à Jour sans Interruption de Service

- Utiliser Rolling Updates avec un Deployment pour mettre à jour l'application Node.js.
  Commande pour effectuer une mise à jour :
  kubectl set image deployment/nodejs-deployment nodejs-container=your-registry/new-nodejs-app :latest
- Appliquer les mises à jour progressivement pour éviter les temps d'arrêt.