

Projet Kubernetes

Déploiement et orchestration de l'application Dockercoins

Brahim Hamdi

Générer les bitcoins

Le minage c'est le procédé par lequel les transactions Bitcoin sont sécurisées. A cette fin les mineurs effectuent avec leur matériel informatique des calculs mathématiques pour le réseau Bitcoin. Comme récompense pour leurs services, ils collectent les bitcoins nouvellement créés ainsi que les frais des transactions qu'ils confirment.

Les mineurs (ou les coopératives de mineurs) sont en concurrence et leurs revenus sont proportionnels à la puissance de calcul déployée.

— minage de bitcoins, <https://bitcoin.fr/minage/>

Dockercoins

Notre application exemple est **Dockercoins**, une application de minage de Dockercoins! (simulation de bitcoins).

Toutes les images de cette application (fictive et complètement inutile) sont disponibles sur DockerHub.

Elle est composée de 5 composants :

- **rng** = un service web (Python) générant en sortie des nombres aléatoires
image docker : *brahimhamdi/rng:latest*
- **hasher** = un service web(Ruby) générant en sortie un hash des données qui lui sont envoyées par HTTP POST
image docker : *brahimhamdi/hasher:latest*
- **worker** = processus(Python) utilisant rng et hasher
image docker : *brahimhamdi/worker:latest*

- **webui** = web interface (JS)
image docker : *brahimhamdi/webui:latest*
- **redis** = base de données NoSQL
image docker : *redis:latest*

Principe

- worker demande à rng de lui fournir des données aléatoires
- worker injecte ces données dans hasher , hasher génère un hash, récupéré par worker,
- Pour chaque hash commençant par 0, worker génère un DockerCoin
- Le worker stocke les DockerCoins générés dans une base de données Redis,
- La webui affiche les données ajoutées à la base Redis en temps réel.

Chaque service est appelé par un nom DNS simple (hasher, rng, etc..). Ce sera notre rôle de démarrer ces différents services avec ce nom pour que le code soit valide.

Travail demandé

- Créez un nouveau Namespace nommé « dockercoins », vous allez appliquer tout le travail dans ce namespace.
- En se basant sur le principe de fonctionnement de l'application décrit ci-dessus, définir la liste des objets à créer pour déployer l'application (deployments, services, ...)
- Créez et appliquez le manifest yaml permettant de déployer cette application.
- Vérifiez le fonctionnement de l'application via son interface web (en utilisant le navigateur web). Quel est le taux de hachage .
- Réservez les ressources suivantes à *hasher* :
 - CPU : 0.1
 - Mémoire : 300 Mi
 La limite de chaque ressource est deux fois la valeur réservée.
- Configurer l'autoscaling du hasher avec les paramètres suivants :

- scale out si la charge moyenne du CPU dépasse 50%
- Nombre min de replicas : 1
- Nombre max de replicas: 5
- Modifiez le fichier de manifest pour avoir :
 - exactement un rng par nœud.
 - 20 rélicas de worker

Appliquez le fichier yaml. Quel est le taux d'hashage ?