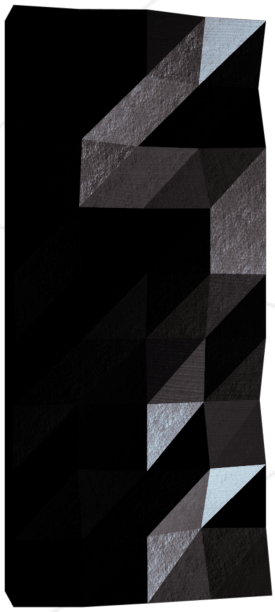

Ft_services

Résumé: Ce document est un sujet d'Administration Système.

Table des matières

Chapitre I

Introduction

Ce sujet a pour but d'approfondir vos connaissances en vous faisant utiliser Kubernetes afin de vous faire virtualiser un réseau et de vous faire découvrir à quoi ressemble réellement un environnement de production. Vous allez faire ce qu'on appelle du "clusturing".

Chapitre II

Consignes générales

- Vous devez rendre tous les fichiers nécessaires à la configuration de votre application dans un dossier `srcs`.
- Votre fichier `setup.sh` devra se trouver à la racine de votre répertoire et devra permettre de mettre en place toute votre application.
- Ce sujet requiert de mettre en pratique pas mal de notions que vous avez pour certains déjà abordés, pour d'autres non. Nous vous conseillons donc de ne pas avoir peur de lire beaucoup de docs sur l'utilisation de kubernetes, ainsi que sur tout ce dont vous aurez besoin pour résoudre le projet.

Chapitre III

Partie obligatoire

Le projet consistera à vous faire mettre en place une infrastructure de différents services, avec ses propres règles. Pour ce faire, vous devrez obligatoirement utiliser `Kubernetes`.
Vous devrez donc mettre en place un `cluster` regroupant plusieurs containers de services.
`Chaque service devra tourner dans un container dédié`.
Chaque container devra `obligatoirement` porter le même nom que le service concerné.
Pour des raisons de performances, les containers `devront` être build sous Alpine Linux.
Aussi, ils devront tous posséder un `Dockerfile` écrit par vos soins qui sera appelé dans le setup.sh.
Vous devrez donc build vous mêmes les images que vous utiliserez et il est bien entendu interdit d'en prendre des toutes faites, tout comme utiliser des services tel que DockerHub.

Vous allez aussi devoir mettre en place :

- Le dashboard web de Kubernetes. Celui-ci est utile pour gérer votre cluster.
- Un `Load Balancer` qui gère l'accès externe à vos services dans un cluster. C'est uniquement lui qui vous servira pour exposer vos services. Vous `devez` garder les ports propres aux services (IP :3000 pour grafana etc).
- Un serveur `Nginx` ouvert sur les ports 80 et 443. Le port 80 sera en http et devra faire une redirection systématique de type 301 vers le 443, qui sera lui en https. La page affichée n'a pas d'importance.
- Un serveur `FTPS` ouvert sur le port 21.
- Un `WordPress` ouvert sur le port 5050, fonctionnant avec une base de données `MySQL`. Les deux devront être dans deux containers distincts. Le wordpress devra comporter plusieurs utilisateurs et un administrateur.
- `PhpMyAdmin`, tournant sur le port 5000 et relié à la base de données MySQL.
- Un `Grafana`, accessible sur le port 3000, fonctionnant avec une base de données `InfluxDB`. Celui-ci devra vous permettre de monitorer `tous` vos containers. Les deux devront aussi être dans deux containers distincts. Vous devrez créer un dashboard par container.
- En cas de crash ou d'arrêt d'un des deux containers de base de données, vous devrez vous assurer que celles-ci puissent persister et ne soient pas perdues. En cas de suppression, les volumes où la data est sauvegardée doivent persister.

- Vous devrez vous assurer de pouvoir accéder à votre Nginx en connexion SSH.
- Chacun de vos containers devra pouvoir redémarrer automatiquement en cas de crash ou d'arrêt d'un des éléments le composant.

L'utilisation de services de type Node Port, de l'objet Ingress Controller ou de la commande kubectl port-forward est interdite. Votre Load Balancer doit être le seul point d'entrée du Cluster.

Il est inutile d'essayer d'utiliser un Load Balancer proposé par les Cloud Provider. Regardez plutot du côté de MetalLB.

La securité est importante, n'exposez pas de services qui ne devraient pas l'être...