La partie du travail accompli

Les images

Nous avons dans un premier temps effectué une réunion pour discuter de l'architecture à mettre en place.

Suite à cette discussion nous avons identifié 5 images à créer : une pour gérer keepalived, une pour pgpool, une pour haproxy, une pour geoserver et une pour postgis. Nous nous sommes donc séparé ce travail, à effectuer chacun de notre côté. Les images devaient tourner sur une architecture ARM, ce qui compliqué la tâche : la majorité des images tournent sur des architectures classiques.

Je devais réaliser une image geoserver et une image haproxy. La réalisation de l'image haproxy n'a pas présenté de difficulté. Cependant, mettre en place l'image geoserver a été plus compliqué.

Effectivement, il faut avoir les bonnes versions de geoserver, tomcat et java pour que tout fonctionne correctement. Dans cette image, nous utilisons geoserver 2.6 avec tomcat 7 et java JDK 1.8. De plus, geoserver est lourd à faire tourner sur une raspberrypi : il faut attendre plus de 1m30 pour que le programme soit lancé. Cela faisait donc perdre beaucoup de temps lors des tests.

Afin de synchroniser les paramètres des différents geoserver, j'ai monté un répertoire partagé sur la Pi maitre. Chaque autre pi monte ce répertoire au démarrage, et le partage avec son image geoserver (si elle en a une).

Le cluster

J'ai aussi effectué l'installation de consul et la configuration du cluster de docker. Cette étape m'avait semblée longue et compliquée pendant le cours, mais finalement elle s'effectue très rapidement.

Dans un deuxième temps, nous nous sommes tous retrouvé pour tester ces images mises ensembles. Nous devions faire des petites modifications pour que tout tourne correctement. Nous avons ensuite fait un docker compose et lancé l'ensemble des images.

Nous avons perdu énormément de temps pour ce débogage : la technologie de docker est beaucoup basé sur internet. Il y avait donc beaucoup de temps de téléchargement. Les Pi ont comme seul stockage des cartes SD, c'était donc très lent de construire de nouvelles images.

Les limitations entrevues sur l'architecture

Notre solution présente un SPOF. Si la Pi qui héberge docker swarm et consul tombe, le système ne sera plus en état correcte de fonctionnement.

Nous avons eu un problème pendant nos tests : consul a mal effacé une donnée en cache. Il nous été impossible de relancer de nouveaux conteneur et tout docker swarm était en mal. Pour contourner ce problème rapidement, nous avons installé consul sur la deuxième Pi. Cela nous donne donc deux SPOF (la Pi qui héberge swarm ou la Pi qui héberge consul). Il faudrait aussi déplacer docker swarm sur la second Pi. Nous n'avons pas voulu perdre de temps à faire cela : notre but était uniquement montrer que notre proposition fonctionne.

Point d'amélioration à traiter

Un point intéressant serait de mettre en place un HAproxy et un keepalived pour consul et de passer consul dans 2 à 3 conteneurs. On éliminerait ainsi un SPOF. Cependant, il y aura toujours docker swarm.

Point non compris ou bloquants

Aucun.