INF8480 - Systèmes répartis et infonuagique

TP3 - Systèmes de fichiers distribués

Chargés de laboratoire :
Pierre-Frederick Denys pierre-frederick.denys@polymtl.ca
Daniel Capelo Borges daniel.capelo@polymtl.ca

Automne 2019 - V2.4

1 Introduction

1.1 Prérequis

— **Services de fichiers répartis et poste à poste :** composantes et interfaces, mécanismes pour l'implémentation. Exemples de Sun NFS, AFS, GFS, Ceph, Napster, Gnutella et BitTorrent

1.2 But du TP

— Découverte de docker et déploiement de containers
— Mise en place d'une architecture de stockage distribuée avec GlusterFS

2 Système de fichiers distribués GlusterFS

2.1 Introduction

Le but du TP est de mettre en place une architecture de stockage haute disponibilité avec glusterfs. L'architecture est composée de deux serveurs de stockage simulés par deux containers, et un poste client simulé par un troisième container. Ensuite, un volume répliqué et un volume distribué seront déployés sur les deux noeuds.

2.2 Procédure

Attention
Pensez à noter les étapes et commandes d'installation, car vous en aurez besoin pour la partie 2.

1. Connectez vous sur Openstack (`http://canari.info.polymtl.ca`), et supprimer les instances et volumes des TP précédents (bien regarder dans *Volumes/volumes*).
2. Créer une nouvelle machine virtuelle sur Openstack selon la procédure habituelle, en faisant attention à bien utiliser l'image `xenial-server-cloudimg-amd64-disk1` avec le gabarit `3072-20-1-1` (<5 minutes). Se connecter à la machine en ssh.
3. Installer docker avec la documentation : `https://docs.docker.com/install/linux/docker-ce/ubuntu/` et suivre **Install Docker Engine - Community** (la procédure n'est pas détaillée içi, car il est important de savoir installer docker sur n'importe quelle machine).
4. Créer les trois containers avec les commandes suivantes : (les ports ouverts sont documentés ici : `https://www.jamescoyle.net/how-to/457-glusterfs-firewall-rules`)

```
sudo docker run -t -d --privileged=true --name gluster1 --
   hostname gluster1 -p 24007 -p 49152-49160  ubuntu
sudo docker run -t -d --privileged=true --name gluster2 --
   hostname gluster2 -p 24007 -p 49152-49160  ubuntu
sudo docker run -t -d --privileged=true --name client --hostname
    client -p 24007 -p 49152-49160  ubuntu
```

5. Connectez vous au premier container

```
sudo docker exec -it gluster1 /bin/bash
```

   Récupérer l'IP du container avec la commande `ifconfig` (installer le paquet `net-tools`).

6. Dans une autre console, connectez vous au second container. Editer le fichier `/etc/hosts/` afin d'y ajouter le nom et l'IP du premier container. Faire de même sur le premier container. Au final, les fichiers `/etc/hosts/` devront contenir les noms et IP des deux containers.

Conseil

Le logiciel `terminator` vous permet de lancer des commandes identiques dans plusieurs terminaux avec les groupes.

7. Sur les **deux** containers, installer gluster-fs avec les commandes suivantes :

```
apt-get install software-properties-common
add-apt-repository ppa:gluster/glusterfs-6
apt-get update
apt-get install glusterfs-server
/usr/sbin/glusterd -p /var/run/glusterd.pid
mkdir /home/disk1/
```

8. Sur le container **gluster1**, ajouter le second container au *trusted pool*, et créer un volume répliqué.

```
gluster peer probe gluster2
gluster volume create replicated1 replica 2 gluster1:/home/disk1
   / gluster2:/home/disk1/ force
gluster volume start replicated1
```

9. Vous connecter au container `client` et exécuter les commandes suivantes pour installer glusterfs client, et monter le volume réseau. N'oubliez pas de modifier le fichier `/etc/hosts` afin d'y ajouter les noms et adresses des deux serveurs.

```
apt-get install software-properties-common
add-apt-repository ppa:gluster/glusterfs-6
apt-get update
apt-get install glusterfs-client
mkdir /mnt/replicated1
mount.glusterfs gluster1:/replicated1 /mnt/replicated1
```

10. Sur les **deux serveurs**, lancez la commande qui permet de voir le contenu du disque du volume en temps réel.

```
watch ls /home/disk1/
```

11. Sur le **client**, créer un fichier `bob.txt` et copiez le dans le dossier `/mnt/replicated1/` Il devrait apparaître sur les deux serveurs. Créer un fichier de grande taille avec la commande suivante, et copier le aussi.

```
head -c 200M </dev/urandom >bigfile1
```

12. Comparer le temps de copie en local et sur le réseau et noter le rapport entre les deux temps.

```
time cp bigfile1 /mnt/replicated1
time cp bigfile1 copie
```

13. Maintenant que vous avez un volume de stockage répliqué, mettez en place un volume de stockage distribué `distributed1` selon la même démarche. Monter le sur le client dans le dossier `/mnt/distributed1`, et observer la répartition des fichier avec la commande `watch` sur les deux serveurs :la commande suivante crée 20 fichiers et vous pouvez observer leur répartition sur les deux serveurs.

```
for (( i=1; i <= 20; i++ )); do touch $i; done
```

2.3 Remise

Répondez aux questions sur moodle et déposez le résultat de la commande exécutée sur l'un des noeuds.

```
gluster volume status
```

3 Partie 2 : Création d'un Dockerfile

Avant de passer à la partie 2, complétez le quiz disponible sur Moodle. Cela permet de vous assurer que vous disposez des bases pour la création d'images docker.

3.1 Introduction

Pour "containeriser" une application, une bonne méthode est de la déployée manuellement (comme dans la partie 1) dans un container pour tester, puis d'écrire un *Dockerfile* pour créer une image générique. Ensuite, on utilise `docker-compose` ou Kubernetes ou Docker swarm pour orchestrer les containers.

Dans cette seconde partie, on va se contenter de créer une image d'un serveur de stockage glusterfs, decréer un container avec et de l'ajouter à notre grappe (*trusted pool*) de la partie 1.

3.2 Procédure

— Créer un dockerfile à partir de la documentation `https://docs.docker.com/develop/develop-images/dockerfile_best-practices/`. Le fichier devra contenir :
  — Image de base `ubuntu`
  — Installation de GlusterFS
  — Exposer les bon ports
  — Pour simplifier, l'édition du fichier `/etc/hosts/` sera effectuée à la main hors du Dockerfile.

— L'image finale `glusterfs_tp3` devra être construite, et lancée avec :

```
sudo docker run -t -d --privileged=true --name gluster3 --
   hostname gluster3 -p 24007 -p 49152-49160  glusterfs_tp3
```

— Se connecter à l'intérieur de ce nouveau container, et demarrer Glusterfs. N'oubliez pas de modifier le fichier `/etc/hosts/` des quatre containers.

```
usr/sbin/glusterd -p /var/run/glusterd.pid
```

— Sur le container `gluster2`, ajouter le noeud et un réplica au premier volume.

```
gluster peer probe gluster3
gluster volume add-brick replicated1 replica 3 gluster3:/home/
   disk1/ force
```

— Tester la copie de fichiers comme pour la partie 1.

3.3 Remise

Répondez aux questions du devoir Moodle, et déposez le résultat de la commande suivante exécutée sur l'un des noeuds.

```
gluster vol info
```

Déposez également votre Dockerfile.

4 Conclusion

On remarque que l'ajout de noeuds devient fastidieuse avec cette méthode, car il faut déclarer le nom du noeud sur tous les autres noeuds. Cela est du au fait que l'on utilise une résolution de nom locale (le fichier `/etc/hosts/`). Dans un système réel, on utilise un DNS (domain name server), qui permet de faire la résolution de nom sur le réseau. Ainsi, pour l'ajout d'un noeud, il suffit d'ajouter une entrée sur un seul fichier. Ce type de service sera abordé dans le prochain TP.

Contrairement à certains systèmes de stockage distribué, GlusterFS n'utilise pas de serveur de métadonnées, il n' y a donc pas de serveur central. GlusterFS est une solution très fiable et opensource, utilisée massivement en IA et Big Data et par de grandes entreprises comme Amazon et Red Hat.

♢