

Projet de Virtualisation

Membre

- **KAMSU KOM FLORIAN**
- **NKONGUE BERTHA**
- **WABO ARMAND**
- **GUEFA VALDES**
- **ABESSOLO GEORGES**
- **BOUETOU MBOH**

Objectif du projet

Le but est de permettre la migration de machine virtuelle en se limitant aux features utiles. Notre mission consiste à sélectionner les features utile d'une machine virtuelle

Solution

La solution trouvé a été récupérer la liste des features de la machine virtuelle puis de désactiver un à un ces features et de vérifier à chaque fois l'état de la machine virtuelle, si celle-ci crashe ou les tests de l'application échoue alors ce feature est important donc il est nécessaire de considérer celui-ci lors de la migration.

Application

Nous avons automatiser le processus de teste de feature via des scripts en utilisant comme application de test postgresql (dont nous expliquerons l'installation et comment effectué les tests de postgresql) dans une vm ayant un user florian et notre dom0 a également un user florian.

Le processus se passe comme suit :

- Récupérer la liste des features puis l'écrire dans un fichier ensuite envoyer le fichier au dom0 via une connexion ssh (il est nécessaire de configurer une connexion ssh sans mot de passe entre le domU et le domO)
- Lire le fichier des features ensuite désactiver un à un chaque feature
- Pour chaque feature désactiver vérifier l'état de la vm si celle-ci crashe alors ajouter ce feature dans le fichier contenant les features utile sinon écrire le résultat des tests dans un fichier puis l'envoyé au dom0 via une connexion ssh. Ces résultats de tests sera évaluer plus tard par l'utilisateur

Le procès d'automatisation est composé de plusieurs fichier à savoir :

- Dans le domU on a les fichiers:
 - `exec+`: qui exécute la commande `cpuid` pour récupérer les registres qui indique la présence des features sous forme binaire (c'est l'exécutable d'un fichier `c++` qui a été compilé)
 - `features.py`: qui via `exec+` mappe la chaine binaire des registres avec le nom des features.
 - `Script.sh` : qui exécute les tests au démarrage et également `features.py`. Il permet également d'envoyer les résultats au dom0 via une connexion ssh
- Dans le dom0
 - `main.py` : qui est le code qu'on doit exécuter pour lancer les tests

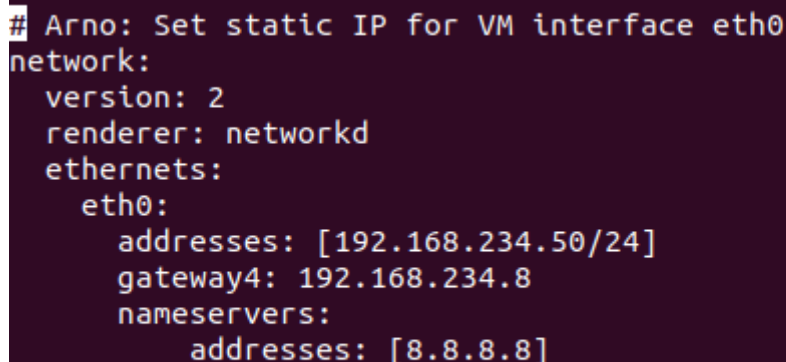
Installation de postgres

- Se positionner dans le dossier «/home/florian» et obtenir les sources de postgres sur le serveur de postgres en utilisant la commande
`wget https://ftp.postgresql.org/pub/source/v12.0/postgresql-12.0.tar.bz2`
- extraire via la commande `tar -xvf postgresql-12.0.tar.bz2`
- `cd postgresql-12.0`
- Puis se connecter en tant que user différent de root. `su - user`
- `./configure`
- Build postgres en utilisant la commande `make`
- Puis faites les tests en utilisant la commande `make check`

Mise en réseau du domU et du dom0

- `sudo brctl addbr xenbr0`
- `sudo ifconfig xenbr0 192.168.234.180`

puis démarrer la vm et modifier l'adresse via le fichier `/etc/netplan/01-netcfg.yaml` comme sur la capture d'écran suivante



```
# Arno: Set static IP for VM interface eth0
network:
  version: 2
  renderer: networkd
  ethernets:
    eth0:
      addresses: [192.168.234.50/24]
      gateway4: 192.168.234.8
      nameservers:
        addresses: [8.8.8.8]
```

1. Récupérer la liste des features

La récupération de la liste des features se fait grâce au fichier « features.py ». Celui-ci est exécuté dans le domU et écrit la liste des features de la VM dans le fichier « features_list.json ». Ce fichier sera par la suite envoyé au dom0 via une connexion ssh dans le repertoire «/home/florian/logFiles ».

2. Lire la liste des features

Le code principal « main.py » lit la liste des features dans le fichier « features_list.json » et boucle sur celle-ci. Pour chaque feature, celle-ci désactive le feature en modifiant le fichier de configuration en utilisant la notation LIBXL (`cpuid= 'host, feature=0'`). Le fichier « main.py » est exécuté dans la machine physique (dom0)

3. Vérification de l'état de la vm après avoir supprimer le features

Un service est exécuté au démarrage (« script.sh ») de la vm celui-ci effectue les tests suites de l'application, écrit les résultats dans un fichier log, elle permet également de récupérer la liste des features dans un fichier en exécutant le fichier « features.py ». Ceux-ci sont envoyés au dom0 sous le format « log_test_suite_featureA » (ceci veut dire résultat du test de featureA) et « features.json ». Si la machine crashe avant le démarrage le nom du feature sera ajouté dans le fichier « some_useful_features ».

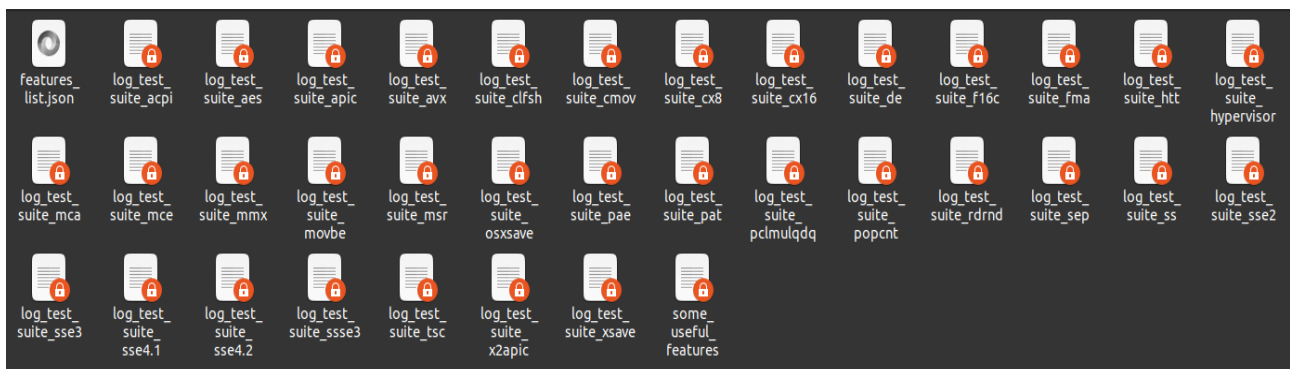
pour créer le service qui exécute script.sh au démarrage il suffit de :

- Aller dans le repertoire *etc/systemd/system* puis y copié dans celui-ci le fichier *check.service*
- Copiez le fichier « *check.sh* » dans le repertoire */usr/local/bin*
- Copier le fichier *script.sh* dans le repertoire « */home/florian* »
- Donner les bonnes permissions aux fichiers
 - *sudo chmod 744 usr/local/bin/check.sh*
 - *sudo chmod 664 etc/systemd/system/check.service*
- Activer le service avec les commandes :
 - *sudo systemctl daemon-reload*
 - *sudo systemctl enable check.service*

En récapitulatif pour utiliser les codes il suffit de :

1. Installer postgresql
2. Mettre en réseau le domU et le dom0
3. configurer une connexion ssh sans mot de passe entre le domU et le dom0
4. Créer le service dans le domU qui exécute script.sh au démarrage en ayant copié script.sh, features.py et exec+ dans le dossier « *home/florian* »
5. Configurer une connexion ssh sans mot de passe entre le user « *florian* » du domU et le user « *florian* » du dom0
6. lancer le fichier *main.py*

Nous avons ci-dessous le résultat des tests effectué sur notre machine virtuelle



dossier contenant le résultat des tests