Année Universitaire 2013-2014
3ième année IMAC

Mardi 11 février 2014

Cloud computing
Travaux Pratiques

Objectif

Dans un premier temps, on utilisera libvirt : une librairie d'accès aux principaux hyperviseurs du marché ; l'API libvirt est écrite en C et est sous licence libre (http://libvirt.org/). Nous allons utiliser le programme `virsh`, une interface en ligne de commande pour la librairie libvirt pour manipuler les machines virtuelles : création, démarrage, arrêt, destruction, modification des ressources, snapshot, recopie et déplacement.

Dans un second temps, nous installerons dans une machine virtuelle, un service de stockage de fichiers en ligne. On profitera de ces dispositifs pour analyser les traces sur le réseau, ce qui nous conduira à augmenter le niveau de sécurité du service. Enfin nous ferons une analyse de traces de serveur.

En début de travaux pratiques, deux étudiants peuvent se regrouper autour d'un poste de travail et pour la partie de déplacement des machines virtuelles, il faut travailler avec deux postes et donc par groupe de quatre étudiants.

Installations préalables

Se logguer avec l'utilisateur `tpreseau`, mot de passe `tpreseau`, ouvrir un terminal de commande.
Vérifier les possibilités de virtualisation du CPU, la commande

```
egrep '(vmx|svm)' --color=always /proc/cpuinfo
```

doit afficher les caractères `vmx` ou `svm` en couleur.
Vérifier la configuration BIOS, la commande

```
dmesg | grep -i kvm
```

ne doit rien afficher.
Pour avoir les droits root avec transfert des droits sur le serveur X, exécuter la commande,

```
sux
```

le mot de passe est `tpreseau.`
Récupérer les scripts, le domaine `squeeze.qcow2` et l'image ISO du cd-rom d'installation `FAN-2.4-i386.iso` dans le répertoire `/mnt/nfs` qu'il faut créer :

```
mkdir /mnt/nfs
rsync 172.17.4.ZZ:/mnt/nfs/* /mnt/nfs/
```

Ranger les scripts et fichiers de configuration dans /root

```
mv /mnt/nfs/squeeze.xml /mnt/nfs/*.sh /mnt/nfs/*.php /root
```

Faire tourner le script

```
./generic-init.sh
```

Obtenir de l'aide sur l'interface `virsh` :

```
virsh help|less
```

Obtenir de l'aide sur la commande `start` :

```
virsh help start
```

Installer une machine virtuelle à partir d'une image ISO

Éditer le fichier `install-fan.sh` pour lui attribuer votre adresse mac en remplaçant les xx par votre numéro et exécuter le script d'installation :

```
./install-fan.sh
```

dérouler l'installation et pendant les temps morts passer aux taches suivantes.

Le cycle de vie des domaines

Éditer le fichier `squeeze.xml` pour lui attribuer une adresse mac en remplaçant les xx par votre numéro puis démarrer le domaine squeeze qui n'est pas encore enregistré dans la liste des domaines gérés par libvirt

```
virsh create squeeze.xml
```

Accéder à la console du domaine squeeze

```
virt-viewer squeeze &
```

Vérifier le démarrage

```
virsh list --all
```

Arrêter le domaine

```
virsh shutdown squeeze
```

Vérifier l'arrêt

```
virsh list --all
```

Enregistrer le domaine squeeze dans la liste des domaines gérés par libvirt

```
virsh define squeeze.xml
```

Vérifier cet enregistrement

```
virsh list --all
```

Supprimer cet enregistrement du domaine squeeze

```
virsh undefine squeeze.xml
```

Vérifier la suppression

```
virsh list --all
```

Définir à nouveau le domaine squeeze

```
virsh define squeeze.xml
```

Démarrer le domaine squeeze qui est défini (noter la différence avec la commande « virsh create squeeze.xml » qui permet de démarrer un domaine qui n'a pas été défini)

```
virsh start squeeze
```

Vérifier le démarrage

```
virsh list
```

Vérifications

Accéder à la console de la machine invité

```
virt-viewer squeeze &
```

Se logguer en tant `root`, mot de passe `rootroot` et lancer la commande

```
ifconfig
```

pour récupérer l'adresse IP du domaine invité.

Afin de simplifier les commandes et en même temps de montrer l'utilisation des commandes ssh et pipe, nous allons permettre à l'utilisateur root de la machine hôte de se connecter en ssh sur le compte root de la machine invité sans avoir à rentrer de mot de passe.

Réaliser une première connexion ssh

```
ssh ip_inv
```

afin de stocker une empreinte (le fingerprint) sur votre machine. Cette empreinte permet de se protéger d'une usurpation d'identité de la machine cible, ip_inv dans ce cas.

Ensuite, il faut créer un couple de clé privé, clé public sur la machine hôte :

```
ssh-keygen
```

- le choix proposé d'emplacement de la clé peut être validé, appuyer sur entrée pour le valider

- la passphrase peut être vide, appuyer sur entrée
- pour confirmer, appuyer à nouveau sur entrée

Il faut maintenant envoyer la clé public de root de la machine A dans le compte root de la machine B
```
cat /root/.ssh/id_rsa.pub |ssh ip_inv "cat >>/root/.ssh/authorized_keys"
```
Vous pouvez maintenant vérifier que la commande
```
ssh ip_inv
```
ne demande pas de mot de passe. Puis sortir du terminal invité avec
```
exit
```

Installer un script php qui donne l'heure sur la machine invité, depuis l'hôte :
```
cat /root/heure.php | ssh ip_inv "cat >/var/www/heure.php"
```

Enfin, dans un navigateur de la machine hôte, aller à l'adresse
http://ip_inv/heure.php

Sauvegarde à chaud

Pour réaliser une sauvegarde à chaud des domaines, on exécute les trois opérations suivantes :
1) figer le domaine
2) prendre un snapshot (une photo) de la partition support du domaine
3) reprendre l'exécution normale du domaine.
Nous n'avons pas la possibilité ici de montrer le snapshot de la partition, nous ne montrons donc que les étapes 1) et 3), pendant ces deux opérations, conserver un regard sur http://adresse_ip_du_domaine_squeeze/heure.php

Suspendre l'exécution du domaine
```
virsh suspend squeeze
```

Reprendre l'exécution du domaine
```
virsh resume squeeze
```

Modifier les ressources

Dans le domaine squeeze, exécuter la commande top afin de visualiser les ressources du domaine
```
top
```

Puis, sur l'hyperviseur, augmenter la mémoire de la façon suivante
```
virsh setmem squeeze 1048576
```
Vous devez constater une augmentation immédiate de la ressource Mem, il est aussi possible de diminuer la mémoire, à exécuter avec la plus grande précaution, car si un processus utilise la partie de la mémoire qui disparaît du système, ce processus risque fort de disparaître !

Augmenter le nombre de CPU :
```
virsh setvcpus squeeze 2
```

Dans ce cas, il est nécessaire de redémarrer le domaine pour visualiser l'apparition d'un second CPU, exécuter la commande top, puis appuyer sur la touche 1.

Pour ne pas surcharger le système inutilement, refaire l'opération avec
```
virsh setvcpus squeeze 1
```

Migration de domaine

Il faut maintenant travailler avec deux postes et donc à quatre étudiants, soient les postes A et B. L'objectif de cet exercice est de montrer que l'image (le disque virtuel) d'un domaine peut résider sur un système de fichiers et qu'il peut être exécuté sur une autre machine hôte ; on va montrer que la migration du domaine peut se faire sans arrêt du domaine. Cet exercice illustre les propos d'Eben Moglen : « Cloud means servers have gained freedom. Freedom to move. Freedom to dance; to combine and to separate, re-aggregate, and do all sorts of tricks. » Ici, l'image du domaine réside sur la machine A. Au début de l'exercice, le domaine est exécuté sur la machine A et nous montrons que tout en résidant sur la machine A son exécution peut se déplacer sur la machine B.

Situation initiale :

Sur la machine A
Le domaine squeeze est défini est il est en cours d'exécution, vérifier avec
```
virsh list
```

Il faut installer un serveur de fichier pour permettre à la machine B d'accéder à l'image du domaine. Avec le script suivant, nous installons le serveur NFS (Network File System)
```
./serveur-nfs.sh
```

Sur la machine B
Le domaine squeeze n'est pas défini, exécuter
```
virsh shutdown squeeze
virsh undefine squeeze
```

Pour que le poste B accède au serveur de fichier de la machine A, il faut éditer le script `client-nfs.sh` et indiquer l'adresse ip de la machine A, puis faire tourner le script
```
./client-nfs.sh
```

La commande
```
mount
```
vous permettra de vérifier le montage de la partition

Maintenant les deux machines sont prêtes pour la migration du domaine, pendant l'opération observez bien la continuité d'exécution du domaine http://adresse_ip_du_domaine_squeeze/heure.php
sur la machine A, exécuter la commande
```
virsh migrate --live squeeze qemu+ssh://ip_de_B/system
```

Sur les machines A et B, vérifier la liste des domaines
```
virsh list --all
```

A la fin de cet exercice, pour continuer dans de bonnes conditions, il est nécessaire de démonter le volume NFS avec la commande
```
umount /mnt/nfs
```

Recopie des domaines

Une fonctionnalité très appréciée de ces domaines est de pouvoir se répliquer très facilement. On distinguera trois modes de recopies.

1) Le clonage d'un domaine, il s'agit d'une recopie de l'image dans l'état, l'outil de création prend soin de ne pas recopier les informations qui doivent être uniques pour un domaine.

2) Le snapshot n'est pas une véritable recopie, il s'agit d'une photo du système à un instant T. Il permet de conserver l'état d'un domaine à un instant T.
3) Le fichier de débordement (Copy on Write)

Le clonage d'un domaine

Attention, la recopie ne peut pas se faire à chaud

```
virsh shutdown squeeze
```

définissez votre adresse mac en remplaçant les xx par votre numéro

```
virt-clone --original=squeeze --mac 54:52:00:03:xx:00 --auto-clone
```

Vous venez d'installer une machine complète en un temps record ! Vérifiez que les domaines démarrent bien

```
virsh start squeeze
virsh start squeeze-clone
```

Pour visualiser les différences des deux domaines, vous pouvez exécuter la séquence de commande suivante :

```
virsh dumpxml squeeze > squeeze.xml
virsh dumpxml squeeze-clone > squeeze-clone.xml
diff squeeze.xml squeeze-clone.xml
```

Snapshot d'un domaine

Sur la machine invitée, écrire un fichier

```
echo "avant snapshot" >> journal
```

Voir la liste des snapshot d'un domaine

```
virsh snapshot-list squeeze
```

Créer un snapshot

```
virsh snapshot-create squeeze
```

vérifier la présence du snapshot

```
virsh snapshot-list squeeze
```

afficher le contenu du fichier /root/journal dans le domaine

```
cat journal
```

effectuer une modification dans le domaine

```
echo "apres snapshot" >> journal
```

vérifier le contenu du fichier :

```
cat journal
```

revenir au point du snapshot

```
virsh snapshot-revert squeeze xxxxxx
```

vérifier dans la VM

```
cat journal
```

supprimer le snapshot pour récupérer de la place disque

```
virsh snapshot-delete squeeze xxxxxx
```

vérifier la disparition du snapshot

```
virsh snapshot-list squeeze
```

Installation de Owncloud

Owncloud est un logiciel libre de service de stockage et de partage de fichiers en ligne qui possède des clients intégrés pour les principaux systèmes d'exploitation windows, mac et linux et pour les mobiles android et ios ; on peut comparer son périmètre fonctionnel à Dropbox. Il existe d'autres logiciels libres sur ce sujet : sparkleshare, seafile, cozycloud et pydio.

Installer owncloud sur la machine invité en exécutant le code expliqué en début de cours

```
cat install_owncloud.sh | ssh root@ip_inv "cat | sh"
```

Vérifier l'installation

http://ip_inv/owncloud/index.php

créer le compte administrateur, par exemple : utilisateur = admin , mot de passe = admin

créer un compte toto et un autre titi, sortir de la session admin, rentrer avec la session toto, uploader un document, le partager avec titi, sortir de la session, rentrer avec titi et vérifier que le document uploadé est bien disponible, puis sortir de l'application.

Le site protège son accès par mot de passe sans chiffrement, c'est comme fermer une porte sans la fermer à clé, n'importe qui peut rentrer sans casser la serrure. Pour le montrer, il suffit de se placer sur une machine qui se trouve sur le chemin réseau de la requête (en l'occurrence la machine hôte ou la machine invité) et observer le flux avec la commande sur la machine invité :

```
tcpdump -i eth0 port 80 -A | grep -i password=
```

et sur la machine hôte

```
tcpdump -i br0 port 80 -A | grep -i password=
```

si la commande tcpdump n'est pas installée, vous pouvez l'installer avec la commande

```
apt-get install tcpdump
```

À la suite de cette observation, il devient évident de la nécessité de chiffrer le flux réseau. Il faut dans un premier temps installer le module SSL du serveur Apache, le paramétrer et installer un certificat. Dans un second temps, aller dans l'interface d'administration d'Owncloud pour cocher la case « Forcer HTTPS ».

Le script install_ssl-owncloud.sh vous permettra de réaliser cette installation.

À la suite de cette installation, vérifier que le chiffrement est bien opérationnel :

```
tcpdump -i eth0 port 80 or port 443 -A | grep -i password=
```

Analyse des traces

Donner la liste des adresses ip qui ont vu la page truc.html

```
grep truc.html access.log
grep truc.html access.log|cut -d" " -f1|
grep truc.html access.log|cut -d" " -f1|sort
grep truc.html access.log|cut -d" " -f1|sort|uniq -c
```

utiliser man tail, man grep, man cut, man sort, man uniq pour bien comprendre chacune des étapes

Obtenir la liste des adresses ip qui se sont connectées le 8 février entre 7h et 7h15 et le nombre de fois où chacune d'elle s'est connectée. Avec la suite des commandes suivantes, expliquer comment on arrive au résultat.

```
grep "08/Feb/2014:07:0" access.log
```

```
grep "08/Feb/2014:07:1[0-5]" access.log
grep "08/Feb/2014:07:0\|08/Feb/2014:07:1[0-5]" access.log
grep "08/Feb/2014:07:0\|08/Feb/2014:07:1[0-5]" access.log | cut -d" " -f1 |sort
|uniq -c
```

Obtenir la liste des adresses ip qui se sont connectées le 9 février entre 7h05 et 7h15 et qui ont vu la page bidule.html et le nombre de fois où ces deux conditions se sont produites.

Donner tous les jours où l'adresse IP 192.168.11.53 s'est connecté. Utiliser la commande man sed

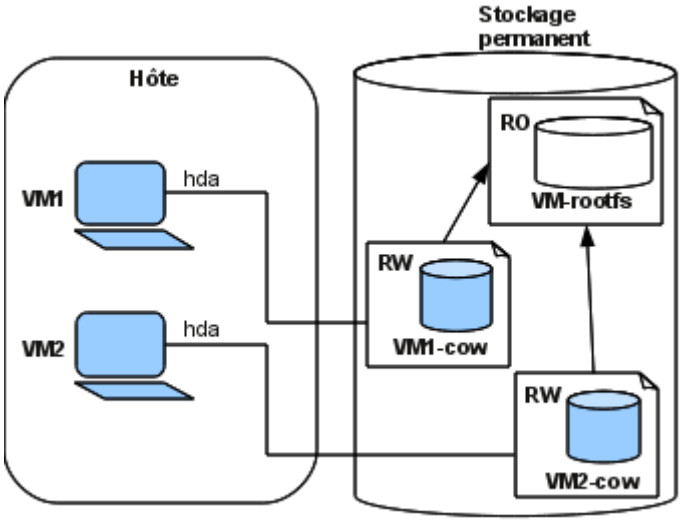
```
grep 192.168.11.53 access.log
grep 192.168.11.53 access.log| sed -e 's/.*\[//'
grep 192.168.11.53 access.log| sed -e 's/.*\[//' -e 's/:.*//'
grep 192.168.11.53 access.log| sed -e 's/.*\[//' -e 's/:.*//'|uniq -c
```

Copy On Write

Cet exercice a été élaboré par Jacques Landru de l'Institut TELECOM pour un TP sur la virtualisation. Les paragraphes d'explication sont conservés dans le texte, mais l'exercice a été modifié pour manipuler les résultats avec libvirt. Dans la suite du texte le *rootfs* désigne le système de fichier racine (root file system) d'une machine virtuelle, qui peut être vu comme son disque dur.

Il existe la possibilité à plusieurs machines virtuelles de se partager un *rootfs* en lecture seule. Les opérations d'écriture de chacune des machines virtuelles sont déportées dans des fichiers séparés propres à chaque machine virtuelle. Les machines virtuelles disposent donc d'un espace de débordement (overlay) dans lequel sont stockées les surcharges apportées au système de fichiers de référence. Ces fichiers de débordement ne contenant que les différences avec le *rootfs*, sont dénommés fichiers COW (abréviation de Copy On Write). Ils sont liés au fichier *rootfs* commun et sont de taille modeste. L'utilisation des fichiers COW permet donc de personnaliser un ensemble de machines virtuelles, partageant par ailleurs une base système commune. Le clonage d'un système de base de référence devient une opération très facile. Les machines virtuelles VM1 et VM2 de la figure disposent ainsi de leur propre système de fichiers ne contenant que les différences avec le système de fichiers de référence qu'est le *rootfs*.

« Vachement bien » les fichiers COW : Ce système COW introduit une certaine souplesse de gestion. Le premier avantage de ce système COW est l'optimisation de taille. L'unique et volumineux *rootfs* est partagé pour un ensemble de machines virtuelles clonées. La différentiation de ces VM est déportée dans de petits fichiers. Les environnements de virtualisation libres XEN et KVM/QEMU peuvent exploiter les fichiers COW. QEMU a même amélioré le format en créant les formats QCOW et QCOW2 qui peuvent être chiffrés et compressés. Toutefois, il convient de noter que les fichiers COW sont liés au fichier de référence par un ensemble de pointeurs internes. Ceux-ci référencent les blocs de données modifiés par rapport au *rootfs* de référence. Ce lien fort entre le fichier COW et le *rootfs* de référence interdit toute modification de ce dernier.



Arrêter le domaine squeeze

```
virsh shutdown squeeze
```

Créer le disque squeeze-copie01.qcow2 qui fait référence au disque squeeze.qcow2

```
qemu-img create -f qcow2 \
  -o backing_file=/mnt/nfs/squeeze.qcow2 /mnt/nfs/squeeze-copie01.qcow2
```

observez la taille des différents disques

```
ls -al /mnt/nfs
```

lire les informations sur les disques

```
qemu-img info /mnt/nfs/squeeze.qcow2
qemu-img info /mnt/nfs/squeeze-copie01.qcow2
```

Maintenant que le disque est crée, il faut inclure de domaine dans le gestionnaire libvirt : éditer le fichier `import-copie.sh` pour lui affecter une bonne adresse mac, puis exécuter le fichier

```
./import-copie.sh
```

à ce stade, si vous rencontrez une erreur de type « no bootable disk » dans la console du domaine, il s'agit d'un bug répertorié qui se corrige de la façon suivante :

```
virsh dumpxml squeeze-copie01 > squeeze-copie01.xml
virsh destroy squeeze-copie01
virsh undefine squeeze-copie01
```

éditer le fichier `squeeze-copie01.xml` et changer type='**raw**' en type='**qcow2**'

```
virsh define squeeze-copie01.xml
virsh start squeeze-copie01
ls -al
```

créer un fichier de 50Mo dans le domaine squeeze-copie01

```
virt-viewer squeeze-copie01 &
dd if=/dev/zero of=/tmp/my-50mo-file count=50 bs=1M
```

Continuer l'observation sur l'hyperviseur de la commande

```
ls -al
```

Interface graphique d'accès à libvirt

lancer la commande

```
virt-manager &
```

et recommencer le TP !