TP XEN

Mise en œuvre de l'hyperviseur Xen

sur

Debian Lenny

Thierry DOSTES

Sommaire

Objectifs

Ce TP a pour objectif de vous familiariser avec l'outil de virtualisation Xen. Il se déroule en plusieurs étapes :

- la première consiste à installer l'hyperviseur Xen sur la machine hôte ;
- lors de la seconde étape, vous créerez un domaine invité Xen basé sur Debian Lenny grâce à la suite d'outils **xen-tools** ;
- enfin, après vous être familiarisés avec les différentes commandes de gestion des domaines Xen, nous procèderons à l'installation d'un outil graphique de gestion (Virtual Machine Manager ou virt-manager).

1. Présentation de l'environnement de travail

Pour réaliser ce TP, nous mettons à votre disposition une machine installée sous Debian Lenny. Cette installation utilise un noyau Linux « classique », c'est-à-dire qui n'est pas adapté au support de la virtualisation.

Votre machine obtient automatiquement une adresse IP auprès d'un serveur DHCP. Des adresses IP vous seront attribuées ultérieurement (oralement) pour la création de vos domaines invités.

Les paramètres de connexion sont les suivants :

login : root
password : VaX2009 !

Si votre machine n'a pas obtenu une adresse IP auprès du serveur DHCP, il vous est possible de la configurer manuellement. Pour cela, utilisez une adresse IP de la plage qui vous a été attribuée, et éditez le contenu du fichier de configuration **/etc/network/interfaces**. Par exemple, si l'adresse IP qui vous a été attribuée est 10.126.100.190 :

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug eth0
iface eth0 inet static
        address 10.126.100.190
        netmask 255.255.255.0
        network 10.126.100.0
        broadcast 10.126.100.255
        gateway 10.126.100.250
        # dns-* options are implemented by the resolvconf package, if installed
        dns-nameservers 194.57.126.30
        dns-search jtsiars.glm
```

Nous relançons ensuite le service réseau pour tenir compte des nouveaux paramètres :

```
ulysse:~# ifdown eth0
ulysse:~# ifup eth0
```

2. Intégration de Xen sur Debian Lenny

Installation des paquets

Nous allons installer les paquets nécessaires à l'utilisation de Xen sur une distribution Debian Lenny :

- le noyau Debian adapté à la virtualisation ;
- les différents éléments nécessaires au fonctionnement de l'hyperviseur.

Au préalable, vérifions la compatibilité matérielle de notre processeur avec les technologies de virtualisation. Vos machines de TP sont équipées de processeurs Intel. Nous exécutons la commande suivante :

```
zorro:~# grep vmx /proc/cpuinfo
```

Selon le résultat obtenu, vous savez désormais si vous pouvez utiliser Xen en mode paravirtualisation ou virtualisation totale (matérielle).

Revenons à l'installation de notre hyperviseur. Nous installons le paquet **bridge-utils** qui contient tout un ensemble d'outils permettant de configurer les interfaces Ethernet en mode pont. Cela nous permettra de gérer le mode réseau « pont » de Xen.

```
zorro:~# apt-get install bridge-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  bridge-utils
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 31.2kB of archives.
After this operation, 111kB of additional disk space will be used.
Get:1 http://ftp.fr.debian.org lenny/main bridge-utils 1.4-5 [31.2kB]
Fetched 31.2kB in 0s (96.7kB/s)
Selecting previously deselected package bridge-utils.
(Reading database ... 33015 files and directories currently installed.)
Unpacking bridge-utils (from .../bridge-utils_1.4-5_i386.deb) ...
Processing triggers for man-db ...
Setting up bridge-utils (1.4-5) ...
```

Nous procédons ensuite à l'installation des paquets contenant le noyau « xénifié » et les éléments nécessaires au fonctionnement et à la gestion de l'hyperviseur Xen. Le nom du paquet contenant le noyau a changé depuis la version Lenny de Debian. Ainsi, le traditionnel **linux-image-2.6.26-2-xen-686** est remplacé par **xen-linux-system-2.6.26-2-xen-686**.

```
zorro:~# apt-get install xen-linux-system-2.6.26-2-xen-686 libc6-xen xen-
hypervisor-3.2-1-i386 xen-utils-3.2-1 xen-utils-common xen-tools
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  libconfig-inifiles-perl libexpect-perl libio-pty-perl libio-stty-perl
libneon27 librpm4.4 libsdl1.2debian libsdl1.2debian-alsa libsvga1 libterm-
readline-gnu-perl libterm-size-perl
```

```
  libtext-template-perl libx86-1 libxenstore3.0 linux-image-2.6.26-2-xen-
686 linux-modules-2.6.26-2-xen-686 reiserfsprogs rinse rpm screen svgalibg1
vnstat xen-shell xenstore-utils xfsprogs
Suggested packages:
  linux-doc-2.6.26 alien xfsdump attr dvhtool quota
Recommended packages:
  xen-hypervisor-amd64 xen-hypervisor-i386 xen-hypervisor-i386-pae xen-
hypervisor-3.2-1
The following NEW packages will be installed:
  libc6-xen libconfig-inifiles-perl libexpect-perl libio-pty-perl libio-
stty-perl libneon27 librpm4.4 libsdl1.2debian libsdl1.2debian-alsa libsvga1
libterm-readline-gnu-perl libterm-size-perl
  libtext-template-perl libx86-1 libxenstore3.0 linux-image-2.6.26-2-xen-
686 linux-modules-2.6.26-2-xen-686 reiserfsprogs rinse rpm screen svgalibg1
vnstat xen-hypervisor-3.2-1-i386
  xen-linux-system-2.6.26-2-xen-686 xen-shell xen-tools xen-utils-3.2-1
xen-utils-common xenstore-utils xfsprogs
0 upgraded, 31 newly installed, 0 to remove and 0 not upgraded.
Need to get 28.3MB of archives.
After this operation, 82.9MB of additional disk space will be used.
Do you want to continue [Y/n]? Y
```

xen-hypervisor : il s'agit d'un paquet virtuel. Il contient l'hyperviseur Xen proprement dit. Celui-ci est appelé par l'intermédiaire du *boot loader* dès le démarrage de la machine. L'hyperviseur assure la gestion de la mémoire et du processeur et leur répartition entre les différents systèmes invités hébergés par le domaine privilégié (Domain 0). Par défaut, l'hyperviseur fourni par Debian est capable de gérer les modes paravirtualisés et totalement virtualisés (en présence de processeurs compatibles).

xen-utils : ce paquet contient des outils d'administration pour gérer les systèmes virtualisés à travers l'hyperviseur Xen.

xen-tools : ce paquet contient l'ensemble des outils destinés à gérer des machines virtuelles Xen sur système Debian. A partir des scripts fournis, il vous est possible de créer des domaines invités (domU) qui peuvent ensuite être facilement gérés, mis à jour ou copiés (documentation dans le répertoire /usr/share/doc/xen-tools/).

libc6-xen : tout comme son homologue **libc6**, ce paquet contient les librairies standards utilisées par la quasi-totalité des applications du système. Cependant, cette version contient des librairies optimisées pour le fonctionnement avec Xen. En effet, les domaines invités ne doivent pas travailler dans des zones mémoires contigües. En conséquence, ils ne peuvent utiliser efficacement le mode segmenté de la libraire **libc6** classique. Dans ce cas, Xen est obligé d'émuler ce support au prix d'importantes pertes de performance. Si votre distribution ne fournit pas ce paquet (c'était le cas de la Sarge), il est possible de mettre en œuvre un contournement en renommant le fichier **/lib/tls** en **/lib/tls.disabled**. Attention, à chaque mise à jour du paquet **glibc**, vous serez tenu de renouveler cette opération.

Nous installons ensuite les documentations proposées par le projet Xen. Elles seront accessibles dans le répertoire **/usr/share/doc/xen-docs-3.2/**.

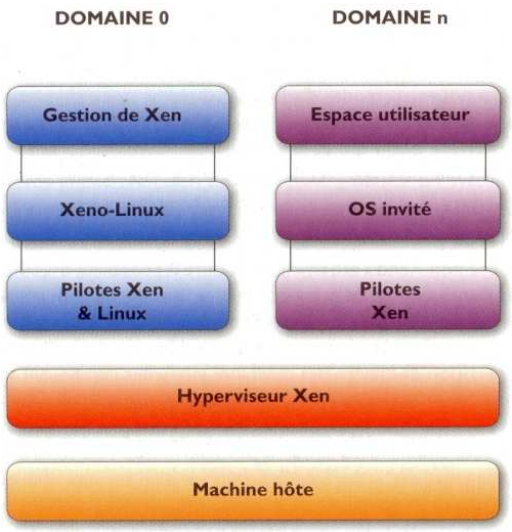
```
zorro:~# apt-get install xen-docs-3.2
```

Avant de redémarrer le serveur, nous nous assurons que le noyau sélectionné (le premier de la liste) dans le fichier **/boot/grub/menu.lst** correspond à celui supportant Xen que nous venons d'installer.

Nous redémarrons ensuite la machine afin de sélectionner le noyau modifié pour permettre l'utilisation des serveurs virtuels. A l'invite de sélection du noyau lors de la séquence de démarrage, nous constatons qu'une nouvelle entrée est disponible et qu'elle est, normalement, sélectionnée par défaut :

```
Xen 3.2-1-i386 / Debian GNU/Linux, kernel 2.6.26-2-xen-686
```

Notre hyperviseur est désormais correctement installé. Avant de créer des domaines invités (càd des machines virtuelles dans la terminologie Xen), nous procédons à la configuration du domaine0 ou domaine privilégié. Pour mémoire, celui-ci exécute les démons et applications permettant de contrôler les domaines invités. L'hyperviseur, quant à lui, gère les temps d'utilisation CPU de chaque domaine et supervise les interruptions.



Configuration du domaine privilégié

A ce stade, le domaine privilégié (domaine 0) a été lancé lors du redémarrage de la machine sur le nouveau noyau. Pour nous en assurer, utilisons la commande **xm** qui permet de gérer les domaines invités. Elle permet de les démarrer, de les arrêter ou de les mettre en pause.

Listons les domaines hébergés par le serveur :

```
zorro:~# xm list
Name                                        ID Mem(MiB) VCPUs State   Time(s)
Domain-0                                     0     7980     8 r-----     46.3
```

De la même façon, il est possible de voir quelles sont les ressources machines consommées par chacun des domaines invités :

```
zorro:~# xm top
```

```
xentop - 12:42:36   Xen 3.0.3-1
1 domains: 1 running, 0 blocked, 0 paused, 0 crashed, 0 dying, 0 shutdown
Mem: 8382412k total, 8252032k used, 130380k free    CPUs: 8 @ 1995MHz
      NAME  STATE   CPU(sec) CPU(%)     MEM(k) MEM(%)  MAXMEM(k) MAXMEM(%) VCPUS
NETS NETTX(k) NETRX(k) VBDS   VBD_OO   VBD_RD   VBD_WR SSID
  Domain-0 -----r         46    0.2    8171520   97.5   no limit       n/a     8
0        0        0    0        0        0        0    0
```

Configuration du réseau en mode “pont”

Par défaut, Xen modifie la configuration réseau de l’hôte physique pour créer un pont qui sera utilisé pour interconnecter les domaines invités. Pour éviter de rendre inutilisable l’interface réseau, le paquet fourni par la distribution Debian ne modifie pas le contenu du fichier de configuration du réseau (/etc/network/interfaces) bien que cela soit requis.

Aussi, nous devons modifier manuellement la configuration réseau en éditant les fichiers **/etc/xen/xend-config.sxp** et, éventuellement **/etc/network/interfaces,** pour définir la manière dont les domaines Xen seront connectés au réseau.

Configurons le réseau du domaine privilégié (domaine 0) pour qu’il utilise le mode pont. Cette configuration se fait par l’intermédiaire de scripts qui sont appelés depuis le fichier **/etc/xen/xend-config.sxp**. Puisque, nous voulons utiliser l’interface réseau en mode pont (bridge), nous décommentons la ligne suivante :

```
(network-script network-bridge)
```

Par défaut, l’installation de Xen sous Debian active la directive ci-dessous qui permet de définir des configurations réseau avancées (exemple : gestion des VLANs). Nous la mettons en commentaires :

```
#(network-script network-dummy)
```

Chacune de ces directives **network-script** appelle son script éponyme :

- **/etc/xen/scripts/network-bridge** qui décrit une configuration en mode pont standard ;
- **/etc/xen/scripts/network-dummy** qui permet de définir sa propre configuration avancée.

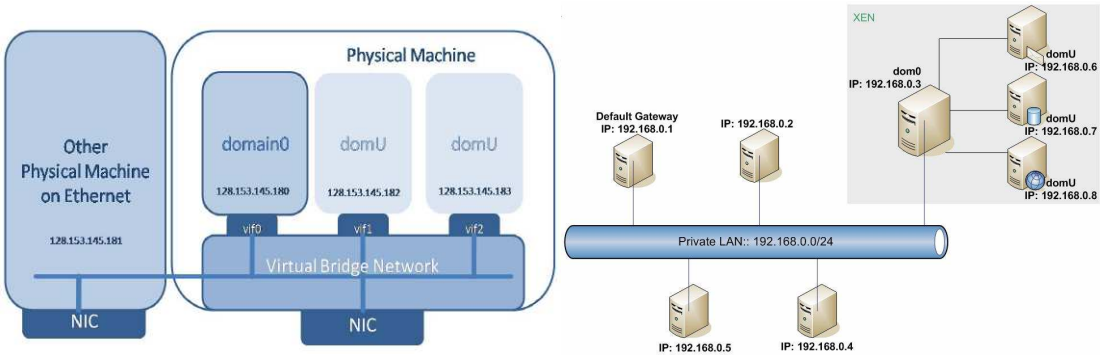
Par défaut, le pont créé par Xen est intitulé **xenBr0**. Il est possible (mais nous le ferons pas dans le cadre de ce TP) de le renommer en utilisant la directive suivante dans le fichier de configuration :

```
(network-script 'network-bridge bridge=<nom_du_pont>')
```

Revenons à la configuration de notre mode “pont”. Déclarons maintenant le script qui sera chargé de contrôler nos interfaces virtuelles. Cette fois encore, nous utilisons le script fourni par défaut lors de l’installation :

```
(vif-script vif-bridge)
```

Rappelons que lorsque l’hyperviseur Xen est configuré en mode pont, chaque domaine invité apparaît sur le réseau comme une machine autonome.



Configuration des paramètres de fonctionnement

Le domaine privilégié (domaine 0) possède la capacité de libérer/réserver de la mémoire au profit d'un domaine invité. Il est possible de définir une valeur minimale de mémoire qui doit toujours lui être réservé grâce à la directive **dom0-min-mem** du fichier de configuration **/etc/xen/xend-config.sxp** :

```
(dom0-min-mem 196)
```

Attention, si cette valeur vaut 0, le domaine privilégié ne libèrera jamais la mémoire qu'il utilise, et ce au détriment des domaines invités hébergés.

De même, si votre machine dispose de plusieurs processeurs, il est possible de préciser le nombre de processeurs utilisés par le domaine privilégié à l'aide de la directive **dm0-cpus**. Par défaut, celle-ci vaut **0**, ce qui signifie que le domaine privilégié pourra utiliser la totalité des unités de calculs mises à sa disposition si cela s'avérait nécessaire :

```
(dom0-cpus 0)
```

Nous pouvons activer le mode débogage de l'hyperviseur Xen pour disposer de traces de fonctionnement plus détaillées :

```
(logfile /var/log/xen/xend.log)
(loglevel DEBUG)
```

Nous redémarrons l'hyperviseur Xen pour prendre en compte ces modifications :

```
zorro:~# /etc/init.d/xend restart
Restarting XEN control daemon: xend.
```

Nous pouvons observer que notre configuration réseau en mode pont est active :

```
calimero:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:15:17:4E:67:CC
          inet addr:10.126.100.190  Bcast:10.126.100.255  Mask:255.255.255.0
          inet6 addr: fe80::215:17ff:fe4e:67cc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24113 errors:0 dropped:0 overruns:0 frame:0
```

```
          TX packets:2502 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3133883 (2.9 MiB)  TX bytes:980470 (957.4 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:9 errors:0 dropped:0 overruns:0 frame:0
          TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:655 (655.0 b)  TX bytes:655 (655.0 b)

peth0     Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
          inet6 addr: fe80::fcff:ffff:feff:ffff/64 Scope:Link
          UP BROADCAST RUNNING NOARP  MTU:1500  Metric:1
          RX packets:24609 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2783 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3284559 (3.1 MiB)  TX bytes:1009358 (985.7 KiB)
          Base address:0x2020 Memory:b8820000-b8840000

vif0.0    Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
          inet6 addr: fe80::fcff:ffff:feff:ffff/64 Scope:Link
          UP BROADCAST RUNNING NOARP  MTU:1500  Metric:1
          RX packets:2502 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24113 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:980470 (957.4 KiB)  TX bytes:3133883 (2.9 MiB)

xenbr0    Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
          inet6 addr: fe80::200:ff:fe00:0/64 Scope:Link
          UP BROADCAST RUNNING NOARP  MTU:1500  Metric:1
          RX packets:21310 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2585834 (2.4 MiB)  TX bytes:0 (0.0 b)
```

```
zorro:~# brctl show
bridge name     bridge id               STP enabled     interfaces
xenbr0          8000.feffffffffff       no              vif0.0
                                                        peth0
```

3. Création d'un domaine invité avec xen-tools

Notre machine hôte est désormais configurée. Nous allons créer un premier domaine invité. Pour simplifier notre tâche, nous utilisons les outils fournis par le paquet **xen-tools**, parmi lesquels des scripts de création de machines virtuelles. Il est ainsi possible de créer un domaine invité Xen, prêt à l'emploi, avec une configuration de base qui comprend les paramètres réseaux et l'installation d'un serveur SSH.

Nous avons vu précédemment qu'il existe plusieurs possibilités pour créer le système de fichiers (l'espace de stockage) d'un domaine invité, que ce soit en local sur la machine hôte ou sur des espaces de stockage réseau :

- à partir d'un fichier image. Si cette solution est de loin la plus rapide à configurer, c'est également celle qui offre les plus mauvaises performances en termes d'E/S. De plus, la machine virtuelle est limitée par la taille initiale de l'image. Cette solution à base de fichier image est cependant facile à implémenter sur un système de secours et il est aisé d'en faire une copie de sauvegarde ;

- sur une partition de type LVM. Il s'agit là de l'implémentation standard retrouvée chez les différents acteurs du monde économique. Après avoir configuré les volumes LVM, il est très facile de les redimensionner en fonction des besoins. Cette flexibilité les rend parfaitement adaptés à l'utilisation des serveurs virtuels Xen. De plus, les performances en E/S sont meilleures que lors du recours à un fichier image ;

- sur des partitions physiques qui présentent les meilleurs performances en E/S mais qu'il est impossible de redimensionner.

Dans le cadre de ce TP, nous avons choisi d'utiliser des fichiers images pour « héberger » les systèmes de fichiers des domaines invités. Nous créons dans la partition **/var** un répertoire dédié au stockage de ces fichiers images :

```
zorro:~# mkdir –p /var/xen-vm/
```

Configuration de xen-tools

Nous commençons par éditer le fichier **/etc/xen-tools/xen-tools.conf** pour paramétrer les éléments qui seront utilisés par **xen-tools** lors de la création des machines virtuelles. Nous définissons tout d'abord le support de stockage utilisé. Plusieurs possibilités s'offrent à nous, mais nous retiendrons uniquement celle qui consiste à déclarer le répertoire qui contiendra des fichiers images :

```
dir = /var/xen-vm
```

Pour information, si vous souhaitez un volume LVM pour stocker les domaines invités, il est alors indispensable de commenter la directive **dir** er de la remplacer par la directive **lvm** :

```
lvm = tests-vg
```

Nous précisons ensuite la méthode d'installation des domaines invités. Comme nous voulons installer un système Debian, nous partirons sur **debootstrap**. Xen-tools nous offre également la possibilité de créer des domaines à partir d'images disque archivées ou en copiant le contenu du répertoire d'une précédente installation.

```
install-method = debootstrap
```

Précisons les paramètres de **debootstrap** en indiquant quel sera le miroir Debian utilisé. Par défaut, xen-tools installera une version 32 bits du système d'exploitation. L'option **arch** permet de demander le téléchargement d'une version 32 ou 64 bits :

```
arch=[i386]
mirror = http://ftp.fr.debian.org/debian/
```

NB : Les images **amd64** de la distribution Debian ne sont pas réservées uniquement aux processeurs AMD. Elles sont également valables pour les puces d'Intel.

Ne le faîtes pas dans le cadre de ce TP, mais il est possible de déclarer explicitement des miroirs pour chaque distribution :

```
mirror_sid=http://ftp.us.debian.org/debian
mirror_sarge=http://ftp.us.debian.org/debian
mirror_etch=http://ftp.us.debian.org/debian
mirror_dapper=http://archive.ubuntu.com/ubuntu
mirror_edgy=http://archive.ubuntu.com/ubuntu
mirror_feisty=http://archive.ubuntu.com/ubuntu
mirror_gutsy=http://archive.ubuntu.com/ubuntu
```

Attention !! Nous gardons les paramètres par défaut pour **arch** et **mirror**. Déclarons maintenant les ressources disque et mémoire affectées à la future machine virtuelle :

```
size   = 6Gb      # Disk image size.
memory = 256Mb    # Memory size
swap   = 256Mb    # Swap size
# noswap = 1      # Don't use swap at all for the new system.
fs     = ext3     # use the EXT3 filesystem for the disk image.
dist   = lenny    # Default distribution to install.
image  = sparse   # Specify sparse vs. full disk images.
kernel = /boot/vmlinuz-`uname -r`
initrd = /boot/initrd.img-`uname -r`
```

La valeur **sparse** pour l'option **image** signifie que la taille du fichier image augmentera en fonction de son remplissage, jusqu'à atteindre la taille définie par l'option **size**. Dans le cas contraire, l'espace sera réservé dès le début sur le système de fichiers de la machine hôte. L'option **image** s'applique seulement lorsque vous utilisez des fichiers images pour le domaine invité (**dir= …**).

Nous définissons ensuite les paramètres réseau : la passerelle, le masque réseau et l'adresse de diffusion. Il est possible de les déclarer de manière statique :

```
gateway   = 10.126.100.250
netmask   = 255.255.255.0
broadcast = 10.126.100.255
```

NB : Si vous disposez d'un serveur DHCP, vous pouvez décommenter l'option **dhcp** en lieu et place des paramètres ci-dessus :

```
dhcp = 1
```

Il est possible de mettre en cache les fichiers téléchargés par debootstrap pour les réutiliser lors de futures installations. Pour cela, il faut activer l'option :

```
cache = yes
```

Pour que le mot de passe root du futur domaine invité vous soit demandé en cours d'installation, positionnez l'option **passwd** à la valeur 1 :

```
passwd = 1
```

Pour information, il est également possible de transférer les comptes utilisateurs de la machine physique qui ne sont pas présents sur les domaines invités vers ces derniers, mais nous ne le ferons pas dans le cadre du présent TP :

```
accounts = 1
```

Enfin, si vous souhaitez que le domaine invité démarre automatiquement après sa création, décommentez l'option **boot** :

```
boot = 1
```

ATTENTION !! Pour la Debian Lenny, lorsque nous utilisons xen-tools pour créer un domaine invité, il est impératif de préciser explicitement ces deux options :

```
serial_device = hvc0
disk_device = xvda
```

La première option permet de préciser le port console utilisé par défaut par le domaine invité. Ce port console est « virtualisé » et mise à disposition par la machine hôte. Cependant, en faisant cela, nous rendons la connexion via ssh impossible car ce service ne trouvera pas de « pseudo terminal ».

```
login as: root
root@10.126.100.192's password:
Server refused to allocate pty
stdin: is not a tty
```

Pour rendre les connexions ssh possibles, il est impératif d'invoquer la commande **xen-create-image** avec l'option -- **role udev**. Le paquet **udev** contient un démon qui crée et enlève dynamiquement des nœuds de périphériques dans le répertoire /dev.

Le fichier de configuration du domaine invité obtenu après l'exécution du script **xen-create-image** sera créé dans le répertoire déclaré ci-dessous :

```
output = /etc/xen
```

```
extension = .cfg
```

ATTENTION !! Un domaine invité ainsi créé ne sera pas lancé automatiquement lors du redémarrage de la machine physique car il ne se trouve pas dans le répertoire **/etc/xen/auto**.

Création du domaine invité

Lors de l'invocation du script **xen-create-image**, il est possible de surcharger certaines valeurs définies dans le fichier de configuration (**/etc/xen-tools/xen-tools.conf**) ou de préciser des paramètres supplémentaires (ex : l'adresse IP du domaine invité ou son adresse Ethernet). Nous invoquons l'option **-- role udev** pour que le script ajoute l'installation du paquet **udev** lors de la création du domaine invité. Grâce à cela, nous pourrons nous connecter par l'intermédiaire du serveur SSH.

Nous vous recommandons de choisir un nom de machine différent de celui utilisé par votre voisin de TP, car vous réaliserez plus tard une migration de votre domaine invité vers son serveur Xen. **Choisissez une adresse IP appartenant à la plage d'adresses qui vous a été attribuée :**

```
zorro:~# xen-create-image --hostname=test2 --ip 10.126.100.192 --role udev

General Information
--------------------
Hostname       :  test2
Distribution   :  lenny
Partitions     :  swap            256Mb (swap)
                  /               6Gb   (ext3)
Image type     :  sparse
Memory size    :  256Mb
Kernel path    :  /boot/vmlinuz-2.6.26-2-xen-686
Initrd path    :  /boot/initrd.img-2.6.26-2-xen-686

Networking Information
----------------------
IP Address 1   : 10.126.100.192 [MAC: 00:16:3E:7A:CB:2B]
Netmask        : 255.255.255.0
Broadcast      : 10.126.100.255
Gateway        : 10.126.100.1


Creating partition image: /var/xen-vm/domains/test2/swap.img
Done

Creating swap on /var/xen-vm/domains/test2/swap.img
Done

Creating partition image: /var/xen-vm/domains/test2/disk.img
Done

Creating ext3 filesystem on /var/xen-vm/domains/test2/disk.img
Done
Installation method: debootstrap

Done

Running hooks
Done

Role: udev
        File: /etc/xen-tools/role.d/udev
```

```
Role script completed.

Creating Xen configuration file
Done
Setting up root password
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
All done
Started new Xen guest: test2 [/etc/xen/test2.cfg]


Logfile produced at:
         /var/log/xen-tools/test2.log
```

Voici le contenu du fichier de configuration que vous avez normalement obtenu pour le nouveau domaine (exemple : /etc/xen/test2.cfg):

```
#
# Configuration file for the Xen instance test2, created
# by xen-tools 3.9 on Tue Aug 25 18:39:52 2009.
#

#
#  Kernel + memory size
#
kernel      = '/boot/vmlinuz-2.6.26-2-xen-686'
ramdisk     = '/boot/initrd.img-2.6.26-2-xen-686'
memory      = '256'

#
#  Disk device(s).
#
root        = '/dev/xvda2 ro'
disk        = [
                  'file:/var/xen-vm/domains/test2/swap.img,xvda1,w',
                  'file:/var/xen-vm/domains/test2/disk.img,xvda2,w',
              ]


#
#  Hostname
#
name        = 'test2'

#
#  Networking
#
vif         = [ 'ip=10.126.100.192,mac=00:16:3E:41:21:6D' ]

#
#  Behaviour
#
on_poweroff = 'destroy'
on_reboot   = 'restart'
on_crash    = 'restart'
```

Pour lister les domaines invités créés grâce au paquet xen-tools, il est possible d'utiliser la commande **xen-list-images** :

```
zorro:~# xen-list-images
Name: test2
Memory: 256
IP: 10.234.224.112
```

Vous pouvez maintenant utiliser les différentes possibilités offertes par la commande **xm** pour superviser, administrer ou vous connecter au domaine invité que vous venez de créer.

Exemple : si vous avez choisi de ne pas démarrer automatiquement le domaine invité après sa création :

```
zorro:~# xm create /etc/xen/test2.cfg
```

NB : Si vous souhaitez installer un système invité Debian avec un serveur X, il est possible d'invoquer le rôle **gdm** en option de la commande **xen-create-image**. Celui est chargé d'ajouter au cours de l'installation les paquets **gdm** et un serveur **vncserver** pour rendre le serveur X accessible à distance, par l'intermédiaire d'une console de gestion par exemple.

```
zorro:~# xen-create-image --hostname=test6 --ip 10.234.224.116 --role udev
--role=gdm –force
(…)
Role: gdm
        File: /etc/xen-tools/role.d/gdm
Role script completed.
```

Quelques commandes

Voici quelques exemples de commandes que vous pouvez taper successivement pour les tester. Commençons par des options permettant de savoir quels domaines invités sont lancés et quelle est la consommation des ressources :

```
zorro:~# xm list
zorro:~# xm list –-long
zorro:~# xm top
```

Pour se connecter en console sur une machine virtuelle :

```
zorro:~# xm console test2
```

Pour sortir de la console d'un domaine invité, il faut utiliser le caractère d'échappement. Sur un clavier Azerty, la séquence est la suivante : **CTRL + SHIFT + touche 5**.

Pour mettre un système invité en hybernation, puis le relancer à partir du contexte sauvegardé :

```
zorro:~# xm save test2 backup.chk
zorro:~# xm restore backup.chk
```

Pour arrêter un domaine invité :

```
zorro:~# xm shutdown
```

4. Virtual Machine Manager (virt-manager)

Virt-manager est un outil graphique de gestion de machines virtuelles Xen. Un paquet Debian officiel est disponible depuis la version Lenny. Nous allons procéder à l'installation de cette application, écrite en Python et utilisant GTK+ pour son interface graphique.

```
zorro:~# apt-get install virt-manager
```

Lors de l'installation, il se peut que vous rencontriez un problème avec le paquet **uswsusp**. Ce paquet contient les programmes exploitant la fonction de mise en veille logicielle en espace utilisateur disponible depuis les noyaux Linux 2.6.17-rc1. Il permet d'enregistrer l'état du système sur disque et de l'éteindre. À la reprise, il sera remis dans l'état où il était avant de quitter (mode *hibernation* ou *veille prolongée*). L'erreur rencontrée lors de son installation indique que le noyau ne supporte pas un espace utilisateur pour les logiciels. En temps normal, il est nécessaire de recompiler le noyau en ajoutant l'option suivante **CONFIG_SOFTWARE_SUSPEND=y**. Debian fournit des scripts de configuration dans le paquet **initramfs-tools** pour assurer une intégration complète.

Bonne nouvelle, nous passerons outre dans le cadre de ce TP. Cependant, avant d'exécuter Virtual Machine Manager, il est indispensable de modifier la configuration du serveur Xen pour autoriser cette application à dialoguer avec l'hyperviseur. Pour cela, il suffit de décommenter la ligne ci-dessous dans le fichier **/etc/xen/xend-config.sxp** et de redémarrer le service Xen.

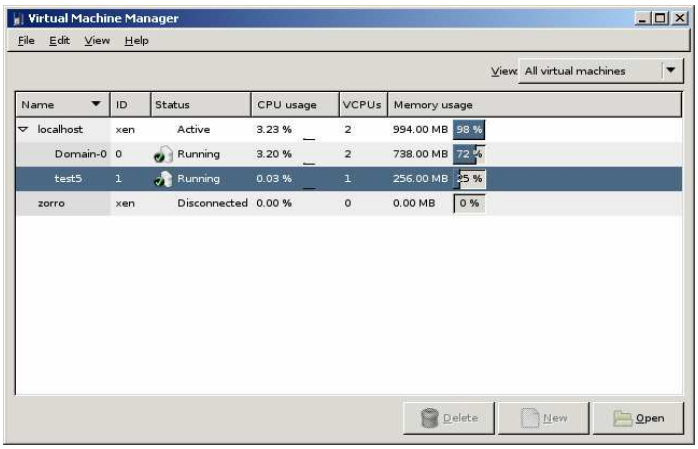
```
(xend-unix-server yes)
```

Dans le fichier de configuration de virt-manager, il est possible de définir l'emplacement des fichiers images des domaines invités créés à l'aide de cet outil graphique (**/usr/share/virt-manager/virtManager/config.py**) :

```
DEFAULT_XEN_IMAGE_DIR = "/var/xen-vm/domains"
```

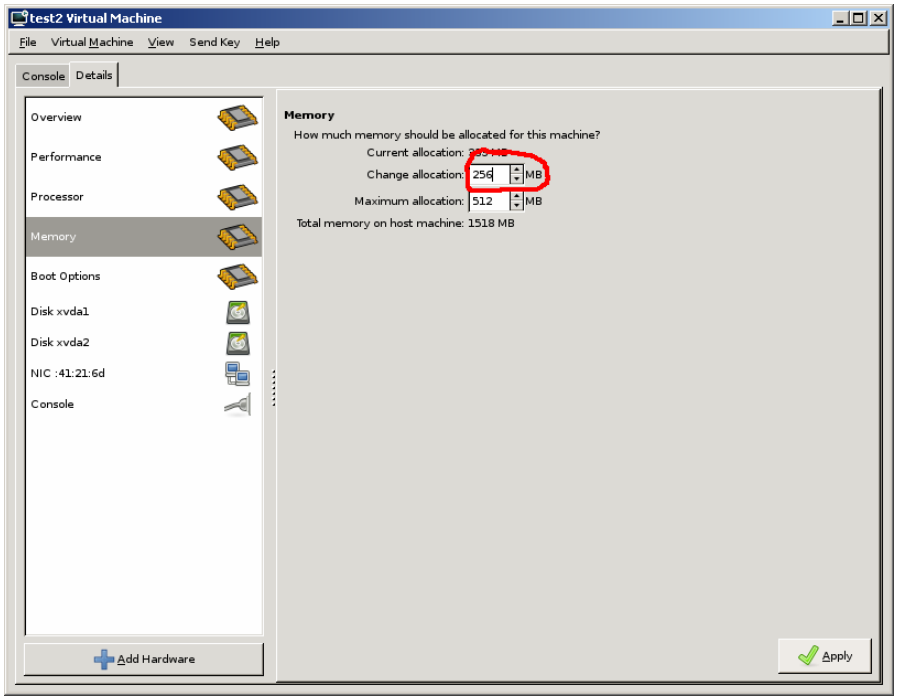
Pour lancer **virt-manager**, après avoir redémarré le démon Xend, exécutez la commande suivante :

```
zorro:~# virt-manager&
```



Pour afficher les différents paramètres de fonctionnement des domaines (ID, Status, CPU usage, VCPUs, Memory usage), sélectionnez le menu **View** et cochez les options souhaitées.

Vous pouvez modifier les paramètres de fonctionnement de votre domaine invité depuis l'interface graphique. Par exemple, diminuez la quantité de mémoire allouée à votre domaine invité.



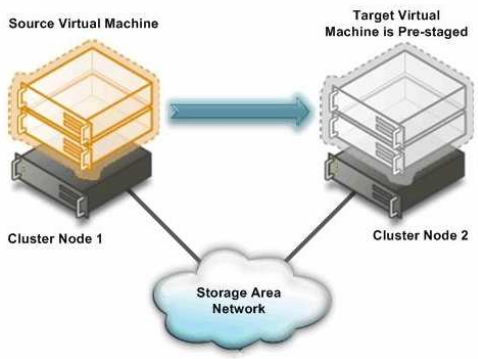
Pour vérifier que la nouvelle valeur est bien prise en compte, connectez-vous sur la console de votre domU et exécutez par exemple la commande **free** :

```
test2:~# free
             total       used       free     shared    buffers     cached
Mem:        262144      27772     234372          0       1736      10644
-/+ buffers/cache:      15392     246752
Swap:       262136          0     262136
```

Rappel : Nous avons vu au cours de la formation que la quantité de mémoire allouée au domaine invité ne peut dépasser la valeur maximale qui a été définie dans le fichier de configuration à l'aide de l'option **memory**.

5. Migration « à chaud » de machines virtuelles Xen

ATTENTION !! Lors d'une migration depuis une machine source vers un serveur de base, Xen n'assure ni le transfert des systèmes de fichiers, ni celui des fichiers de configuration. En effet, Xen a pour vocation d'être utilisé dans des architectures privilégiant des espaces de stockage réseau.



Grâce à ce type d'architecture, il sera seulement nécessaire de transférer manuellement le fichier de configuration de votre domaine invité. **Certains outils graphiques présentés lors de cette formation le feront à votre place, de manière transparente.**

En dépit de cette réserve, la migration des domaines invités reste un point fort de l'architecture Xen. Celle-ci s'effectue en temps réel, de manière totalement transparente pour les utilisateurs de vos services. En effet, la machine virtuelle est migrée vers le serveur destination sans qu'il n'en paraisse rien. Tous les services continuent à fonctionner : les connexions clientes ouvertes restent actives. Par exemple, contrairement à OpenVZ, il n'est pas nécessaire d'arrêter le service Apache pour effectuer une migration « live ».

Le domaine invité migré continue à fonctionner normalement, même en l'absence de son fichier de configuration. Les hyperviseurs Xen de la source et de la destination assurent la continuité de service. L'absence du fichier de configuration pourrait s'avérer critique seulement si la machine devait être relancée, après un redémarrage du serveur hôte par exemple. Cela nous laisse le temps de transférer le fichier sur le nouveau serveur.

Vous allez essayer de migrer votre machine virtuelle Xen sur le serveur de votre voisin :

```
zorro:~# xm migrate test4 10.126.100.160
Error: can't connect: Connection refused
Usage: xm migrate <Domain> <Host>

Migrate a domain to another machine.

Options:

-h, --help           Print this help.
-l, --live           Use live migration.
-p=portnum, --port=portnum
                     Use specified port for migration.
-r=MBIT, --resource=MBIT
                     Set level of resource usage for migration.
```

Le serveur destination refuse la connexion. Cela est normal car nous n'avons pas ouvert au préalable les ports nécessaires à la migration.

Nous activons, sur le serveur destination, le service de relocation et nous précisons le port d'écoute :

```
(xend-relocation-server yes)
(xend-relocation-port 8002)
```

NB : Bien entendu, il est possible de préciser des droits d'accès pour ne pas ouvrir votre serveur Xen au premier venu. Dans le cadre de ce TP, nous ferons confiance à nos collègues. Nous vous invitons toutefois à consulter la documentation ou le fichier de configuration pour sécuriser votre hyperviseur.

```
#(xend-relocation-hosts-allow '^localhost$ ^.*\\.example\\.org$')
```

Nous redémarrons le service Xen. La commande **lsof** nous assure que le port de relocation est est opérationnel :

```
tornado:~# lsof -i |grep 8002
python    2888        root    5u  IPv4   6966       TCP *:8002 (LISTEN)
```

La commande **netstat** nous permet de voir que la machine source essaye de se connecter au serveur destination :

```
tcp      0      1 10.126.100.190:36677     10.126.100.160:8002    SYN_SENT
```

Nous lançons une migration **« regular »**, çàd que nous décidons de mettre le domaine en pause pour réaliser le transfert :

```
zorro:~# xm migrate test4 10.126.100.160
/vm/822825a8-236c-ef12-6935-e12bf3ba68da/name                    :  "migrating-test4"
/local/domain/19/control/shutdown                                :  "suspend"
/local/domain/19/control/shutdown                                :  ""
@releaseDomain                                                   :  "<null>"
/vm/822825a8-236c-ef12-6935-e12bf3ba68da/xend/last_shutdown_reason :  "suspend"
@releaseDomain                                                   :  "<null>"
/local/domain/0/backend/vif/19/0/online                          :  "0"
/local/domain/0/backend/vif/19/0/state                           :  "5"
/vm/822825a8-236c-ef12-6935-e12bf3ba68da/device/vif/0            :  "<deleted>"
/local/domain/0/backend/vbd/19/51713/online                      :  "0"
/local/domain/0/backend/vbd/19/51713/state                       :  "5"
/local/domain/0/backend/vbd/19/51714/online                      :  "0"
/local/domain/0/backend/vbd/19/51714/state                       :  "5"
/local/domain/0/backend/console/19/0/online                      :  "0"
/local/domain/0/backend/console/19/0/state                       :  "5"
/local/domain/0/backend/console/19/0                             :  "<deleted>"
/local/domain/19/device/console/0                                :  "<deleted>"
/local/domain/19                                                 :  "<deleted>"
/local/domain/0/backend/vbd/19/51713/state                       :  "6"
/local/domain/0/backend/vbd/19/51714/state                       :  "6"
/vm/822825a8-236c-ef12-6935-e12bf3ba68da/name                    :  "test4"
/local/domain/0/backend/vif/19/0/state                           :  "6"
/local/domain/0/backend/vbd/19/51714                             :  "<deleted>"
/local/domain/0/backend/console/19                               :  "<deleted>"
/local/domain/0/backend/vbd/19/51713                             :  "<deleted>"
/local/domain/0/backend/vbd/19                                   :  "<deleted>"
/local/domain/0/backend/vif/19/0                                 :  "<deleted>"
/local/domain/0/backend/vif/19                                   :  "<deleted>"
```

Regardons ce qu'il s'est passé sur la machine source : notre domaine invité **test4** n'est plus en cours d'exécution.

```
zorro:~# xm list
Name                                        ID   Mem VCPUs      State   Time(s)
Domain-0                                     0   728     1      r-----   2349.6
test2                                       12   256     1      -b----     22.0
```

Assurons-nous qu'il s'exécute désormais sur le serveur destination :

```
tornardo:~# xm list
Name                                        ID   Mem VCPUs      State   Time(s)
Domain-0                                     0   734     2      r-----     43.4
test4                                        1   256     1      -b----      0.1
```

En revanche, comme nous l'avions vu précédemment, sur la machine destination, nous ne trouvons aucune trace du fichier de configuration du domaine invité. Il suffit de le transférer, via **scp** par exemple.

NB : Parfois, après une migration ou lors d'une montée en charge importante d'un domaine invité, nous observons parfois le message d'erreur suivant dans la console (bug 1098 de Xen) :

```
[  163.006859] clocksource/0: Time went backwards: ret=16e24a67ca38 delta=-
224168733100570 shadow=16e242b2f110 offset=7b5413e
[  163.006859] __ratelimit: 169 messages suppressed
```

Ce message indique que le domaine invité se désynchronise par rapport à l'horloge fournie par Xen. Pour éviter ce problème, il existe un mécanisme de contournement qui consiste à contraindre un domaine invité à utiliser sa propre horloge.

L'opération se déroule en deux étapes. Au niveau du domaine invité (domU), nous ajoutons la ligne suivante dans le fichier **/etc/sysctl.conf** :

```
xen.independent_wallclock=1
```

Ensuite, au niveau de la machine hôte (dom0), nous éditons le fichier de configuration du domaine invité (ex : /etc/xen/test1.cfg) pour ajouter l'option suivante :

```
extra="clocksource=jiffies"
```

Il est possible d'appliquer ces modifications sans redémarrer le domaine invité à la condition de réaliser les deux opérations suivantes sur la machine hôte (dom0) après l'édition des deux fichiers de configuration :

```
sysctl –p
echo "jiffies"> /sys/devices/system/clocksource/clocksource0/current_clocksource
```

Puisque nous avons rendue autonome l'horloge du domaine invité, il sera désormais nécessaire de la synchroniser régulièrement via le protocole NTP.