

# Livre blanc KVM (Kernel-based Virtual Machine)

créé le 19/04/2010 par Antoine Schellenberger .  
dernière mise à jour le 26/04/2010.

*Ce document présente les commandes de base pour l'administration et l'usage de la solution de virtualisation KVM sous Linux.*

## I - Généralités

---

KVM est une solution de virtualisation libre intégrée aux distributions Linux dotées d'un noyau 2.6.20 ou supérieur.

Elle se compose d'un module noyau et d'un client dérivé du produit Qemu.

Le matériel hôte doit comporter un hyperviseur matériel (technologie VT pour les processeurs intel, ou SVM pour les processeurs AMD).

KVM supporte deux types de virtualisation: la virtualisation « complète » (full virtualization) et sous certaines conditions la para-virtualisation.

L'installation et les tests effectués dans ce document ont été réalisés sur un serveur équipé d'un processeur Intel QuadCore 2.44 Ghz avec support VT - 4G de RAM et un disque dur de 1T. Le système hôte est une distribution Linux Ubuntu 9.04 et le système invité est une Debian Lenny.

Bien que KVM nous permette de virtualiser de nombreux systèmes d'exploitations, nous retenons le système invité Debian Lenny qui nous permettra d'illustrer l'activation de la para-virtualisation avec Virtio.

### a) Conventions typographiques

---

Ce document comporte de nombreuses commandes à saisir dans un terminal, des conventions typographiques sont utilisées pour assister le lecteur dans sa compréhension.

#### Invite de commande

Tout ce qui se situe entre deux chevrons au sein d'une commande doit être saisi sur une seule ligne.

Exemple 1

```
>ce texte doit être saisi dans un terminal sur une seule  
et même ligne même s'il s'étend sur plusieurs lignes  
dans ce document  
>ligne 2
```

#### Nature des utilisateurs

Les commandes devant être saisies en tant que super-utilisateur(root) sont en gras comme dans l'exemple suivant:

```
>sudo apt-get install kvm
```

Celles ne demandant pas les privilèges du super-utilisateur ne seront pas en gras, comme dans l'exemple ci-après:

```
>cd /home/thegeekcorner
```

#### Environnement d'exécution des commandes

Les commandes exécutées au sein du système hôte sont représentées sur un fond clair avec une couleur de police noire comme illustré ci-après:

```
> commande hôte
```

Celles exécutées dans le système invité sont représentées sur un fond foncé avec une couleur de police blanche :

```
>commande invité
```

### b) Remarques sur les commandes shell

---

Les commandes shell qui ne sont pas exécutées avec les privilèges du super-utilisateur (root) seront exécutées à partir de l'utilisateur courant thegeekcorner.

---

Remarque: Evidemment, il vous faudra adapter les commandes en remplaçant l'utilisateur thegeekcorner par celui de votre choix

---

### c) Vocabulaire

---

Le système hôte est le système d'exploitation sur lequel KVM est installé. (dans notre cas, il s'agit de Ubuntu 9.04)

Un système invité est un système d'exploitation virtualisé. (Dans notre cas, il s'agit aussi de Debian Lenny)

Un système hôte peut héberger plusieurs systèmes invités.

## d) Abréviations

VM : Machine virtuelle

# II - Préparation pour l'installation

Ce chapitre décrit les étapes nécessaires à l'installation de KVM sur le système hôte Ubuntu 9.04.

## a) prise en charge de la virtualisation

Avant de démarrer l'installation de KVM, il est important de vérifier la compatibilité matérielle et logicielle du système hôte. Pour cela, saisissez dans un terminal:

```
>egrep '^flags.*(vmx|svm)' /proc/cpuinfo
```

Si rien ne s'affiche c'est que votre processeur n'embarque pas d'hyperviseur matériel (VT/SVM); dans ce cas KVM ne peut pas être installé.

Remarque 1 : Si KVM ne peut pas être installé, vous pouvez tout de même utiliser la solution KQemu qui est un module noyau implantant un hyperviseur logiciel. Cette solution comporte cependant plus de restrictions que KVM (ex: absence du support SMP dans les VM).

Remarque 2 : Il arrive quelquefois que malgré le succès de la commande précédente, la virtualisation soit désactivée au niveau du BIOS, il faudra alors l'activer au démarrage du serveur.

## b) Installation de KVM et Qemu

Comme nous l'avons vu dans les généralités, KVM se compose d'un module noyau et d'un logiciel dérivé de Qemu.

L'installation de ces composants se fait à l'aide de la commande:

```
>sudo apt-get install kvm qemu
```

## c) Chargement du module noyau KVM

Une fois le module noyau installé, il faut alors le charger :

```
>sudo modprobe kvm-intel
```

Remarque : si votre processeur est un AMD il faut charger le module `kvm-amd`

On s'assure du succès du chargement:

```
>lsmod | grep kvm | wc -l
```

Si cette commande renvoie une valeur supérieure à 0 c'est que tout c'est bien passé. Le cas contraire, il faut consulter les journaux systèmes du serveur pour en savoir davantage.

## d) Ajout de l'utilisateur courant au groupe `kvm`

Pour pouvoir exécuter KVM sans les privilèges du super-utilisateur, il faut ajouter l'utilisateur `thegeekcorner` au groupe `kvm`. Pour cela, saisissez dans un terminal:

```
>sudo adduser thegeekcorner kvm
```

# III - Eléments d'administration

Ce chapitre présente les commandes permettant la création et l'installation d'une VM depuis un fichier ISO téléchargé.

## a) Création de la machine virtuelle

Pour notre étude de cas, nous allons créer une VM de 20G que l'on nommera `lenny_thegeekcorner.raw`. Pour cela nous utilisons l'utilitaire `qemu-img` :

```
>qemu-img create ~/lenny_thegeekcorner.raw 20G
```

Cette commande crée un fichier binaire au format brut de 20G.

Remarque : bien que aucun système d'exploitation ne soit encore installé dans le fichier `lenny_thegeekcorner.raw`, celui-ci occupe tout de même 20G d'espace disque.

## b) Installation de l'image virtuelle

### Téléchargement

Nous allons télécharger l'image ISO de la distribution Debian Lenny pour les architectures 64bits.

```
>cd ~->wget http://cdimage.debian.org/debian-cd/5.0.1/amd64/iso-dvd/debian-501-amd64-DVD-1.iso
```

Remarque : l'URL de téléchargement peut changer, veuillez vous référer au site <http://cdimage.debian.org>

## Démarrage de l'installation

Une fois le fichier ISO téléchargé, nous allons nous en servir pour démarrer l'installation de la VM. Pour cela, saisissez dans un terminal du système hôte :

```
>kvm -hda ~/lenny_thegeekcorner.raw -cdrom ~/debian-501-amd64-DVD-1.iso -boot d -vga std -soundhw all -smp 4 -m 1024
```

Quelques explications sur les paramètres :

`-hda ~/lenny_thegeekcorner.raw` : spécifie le chemin absolu de la VM;

`-cdrom ~/debian-501-amd64-DVD-1.iso` : spécifie l'image ISO à monter sur `/dev/cdrom`.

`-boot -d` : indique que le système doit démarrer à partir de `/dev/cdrom`

`-vga std` : spécifie le type de carte graphique à émuler

`-soundhw all` : active la prise en charge du son

`-smp 4` : active le support SMP 4 cores au sein de la VM (évidemment, il faut que le système hôte gère le SMP)

`-m 1024` : spécifie la mémoire vive allouée à la machine virtuelle

remarque : Suivant les distributions linux hôtes, l'erreur suivante peut se produire :

```
open /dev/kvm: Permission deniedCould not initialize KVM, will disable KVM support
```

Si c'est le cas, redémarrez la session graphique du système hôte et essayez à nouveau.

## 1er démarrage de la machine virtuelle installée

Une fois l'installation terminée et afin de s'assurer de son succès, il faut redémarrer la VM en saisissant la commande :

```
>kvm -hda ~/lenny_thegeekcorner.raw -vga std -soundhw all -smp 4 -m 1024
```

Une fenêtre graphique devrait apparaître figurant le lancement du système hôte.

## Mise à jour du système invité Debian Lenny

Une fois la VM redémarrée et pour profiter des dernières corrections d'anomalies et de mises à jour de sécurités, il est nécessaire de mettre à jour le système :

```
>apt-get update
```

## Installation du serveur SSH

Par ailleurs, nous allons voir qu'une fois la VM déployée sur le réseau on pourra la contacter à distance avec SSH. Pour cela, nous allons d'ores et déjà installer un serveur SSH sur le système invité :

```
>apt-get install openssh-server
```

## IV - Administration avancée

Après avoir vu dans la première partie comment installer une machine virtuelle, nous allons durant ce chapitre présenter les commandes nécessaires au déploiement d'une VM sur un réseau ainsi que celles permettant l'activation de la para-virtualisation.

### a) Déploiement sur le réseau local

Par défaut le réseau d'une VM est configuré en NAT, c'est à dire que les paquets réseaux émis par la VM passent obligatoirement par une passerelle (dans ce cas le système hôte) pour arriver à destination.

Cette configuration est confortable dans la majorité des cas quand la VM n'a pas besoin d'être contactée depuis l'extérieur, cependant quand il s'agit de virtualiser des services (serveurs de bases de données, serveurs d'applications, serveurs web, ...), il est nécessaire de déployer la VM sur le réseau afin de lui attribuer une IP.

### Installation des dépendances

Le déploiement d'une VM sur le réseau passe par la création d'une interface virtuelle au sein du système hôte. Pour cela il faut installer le paquetage `uml-utilities`:

```
>sudo apt-get install uml-utilities
```

Nous verrons qu'une fois l'interface virtuelle créée, il faudra la relier à une interface physique (création d'un bridge).

Remarque : hélas, pas toutes les cartes réseaux permettent de réaliser des bridges, c'est le cas notamment des cartes wifi.

La gestion des bridges nécessitent l'installation du paquetage `bridge-utils`:

```
>sudo apt-get install bridge-utils
```

## Préparation du pont réseau (bridge) et de l'interface réseau virtuelle

### Ajout de l'utilisateur courant au groupe **uml-net**:

Pour que l'utilisateur courant (thegeekcorner) puisse utiliser une interface réseau virtuelle il faut le rajouter au groupe uml-net:

```
>sudo adduser thegeekcorner uml-net
```

### Configuration du groupe **tunusers**:

Pour que l'utilisateur courant puisse effectuer des opérations d'entrées/sorties sur l'interface virtuelle, il est nécessaire de créer le groupe tunusers et de lui ajouter l'utilisateur courant:

```
>sudo addgroup tunusers && adduser thegeekcorner tunusers
```

### Edition du fichier **50-udev.rules**

Toujours pour permettre l'usage de l'interface virtuelle par l'utilisateur courant, udev doit être configuré pour intégrer le groupe tunusers. Pour cela, éditez le fichier /etc/udev/rules.d/50-udev.rules en remplaçant:

```
KERNEL=="tun", NAME="net/%k"
```

par

```
KERNEL=="tun", NAME="net/%k", GROUP="tunusers",  
MODE="0660"
```

### Modification de **/lib/udev/devices/net/tun**

Les propriétés du fichier /lib/udev/devices/net/tun doivent être modifiés comme suit:

```
>sudo chown :tunusers /lib/udev/devices/net/tun  
>sudo chmod g+rw /lib/udev/devices/net/tun
```

### Redémarrage du système hôte

Pour que l'ensemble des modifications précédentes prennent effet, il faut redémarrer le système hôte:

```
>sudo reboot
```

### Le fichier **/etc/network/interfaces**

L'interface virtuelle et le bridge sont normalement prêts pour être configurés.

Leurs configurations nécessitent l'édition du fichier /etc/network/interfaces.

### Caractéristiques réseau du système hôte

Les modifications apportées à /etc/network/interfaces s'appuient sur la configuration réseau du système hôte dont voici les caractéristiques principales:

- L'interface réseau active sur le serveur est eth0.
- L'adresse IP de eth0 est fixée manuellement à 192.168.0.2;
- l'adresse IP de la passerelle du réseau local est 192.168.0.254;
- Le masque du réseau local est 255.255.255.0;
- Le nom du bridge est br0;
- Le nom de l'interface virtuelle sera tap10;

### Modification du fichier **/etc/network/interfaces**

A partir des caractéristiques précédentes le fichier /etc/network/interfaces doit comporter le contenu suivant:

```
auto lo eth0 tap10 br0  
iface lo inet loopback  
  
iface eth0 inet manual  
iface tap10 inet manual  
address 192.168.0.2  
netmask 255.255.255.0  
gateway 192.168.0.254  
up ifconfig tap10 up  
down ifconfig tap10 down  
down tunctl -d tap10  
tunctl_user thegeekcorner  
  
iface br0 inet static  
bridge_ports eth0 tap10  
bridge_maxwait 0
```

Remarque : Evidemment, il est de votre ressort d'adapter ce script en fonction de votre configuration.

### Redémarrage du réseau et test de la configuration du système hôte

Pour prendre en compte la nouvelle configuration, il est utile de redémarrer le réseau du système hôte:

```
>sudo /etc/init.d/networking restart
```

### Test du bridge

Afin de s'assurer de la bonne création du bridge, la commande suivante doit retourner 1 :

```
>ifconfig | grep br0 | wc -l
```

### Test de la création de l'interface réseau virtuelle

Afin de s'assurer de la bonne création de l'interface réseau virtuelle, la commande suivante doit aussi retourner 1 :

```
>ifconfig -a | grep tap10 | wc -l
```

## Test de l'association du bridge avec l'interface réseau virtuelle

Pour vérifier que l'interface réseau virtuelle tap10 a bien été associée au bridge br0, la commande suivante doit renvoyer 1 :

```
>brctl show | grep tap10 | wc -l
```

## Redémarrage de la machine virtuelle

Désormais que le système hôte est configuré, il faut relancer la VM en passant quelques arguments supplémentaires à kvm :

```
>kvm -hda ~/lenny_thegeekcorner.raw -vga std -soundhw all -smp 4 -m 1024 -net nic,macaddr=52:54:00:12:34:56 ,model=rtl8139 -net tap,ifname=tap10,script=no
```

Quelques explications sur les nouveaux paramètres :

-net nic,macaddr=52:54:00:12:34:56 attribue l'adresse MAC 52:54:00:12:34:56 à l'interface réseau virtuelle;

,model=rtl8139: permet d'émuler la carte réseau Realtek 8139;

-net tap,ifname=tap10: spécifie le nom de l'interface virtuelle à utiliser;

,noscript: indique que le script /etc/qemu-ifup (ou kvm-ifup) n'est pas nécessaire pour initialiser la configuration réseau.

## Test de la configuration réseau de la machine virtuelle

Une fois la VM démarrée, éditez son fichier /etc/network/interfaces à partir du contenu suivant :

```
auto lo eth0
iface lo inet loopback

#CONFIG MANUELLE
iface eth0 inet static
address 192.168.0.2
netmask 255.255.255.0
gateway 192.168.0.254

##CONFIG DHCP
#iface eth0 inet dhcp
```

Redémarrez le réseau de la VM pour prendre en compte les modifications:

```
>/etc/init.d/networking restart
```

Assurez-vous que la nouvelle IP de la VM appartienne au réseau local (dans notre cas, entre 192.168.0.1 et 192.168.0.254).

pour connaître l'IP actuelle, il suffit de saisir:

```
>/sbin/ifconfig eth0
```

## b) Para-virtualisation

L'activation de la para-virtualisation améliore notablement les performances réseaux et disques mais elle nécessite un système invité modifié.

La para-virtualisation au sein de KVM repose sur le projet virtio. Virtio n'est pas disponible pour tous les systèmes invités. Actuellement il n'existe que pour les systèmes invités windows XP, windows 2000 et les distributions linux comportant un noyau 2.6.25 ou supérieur.

Remarque : Virtio nécessite KVM V.60 ou supérieur (ce qui est le cas de Ubuntu 9.04).

## Tests de performances avant l'activation de la para-virtualisation

Cette partie détaille l'installation et le protocole de test2 pour évaluer les performances induites par la para-virtualisation.

### Installation des outils de test

Afin de nous assurer du bénéfice apportée par la para-virtualisation, nous allons mettre en place une procédure de test de performances. Pour cela, nous allons tester les performances réseau et disque en utilisant les utilitaires iperf et hdparm que nous installons à la fois dans le système hôte et le système invité :

```
>sudo apt-get install iperf hdparm
>apt-get install iperf hdparm
```

### Test sans activation de la para-virtualisation

Sur le système hôte et la VM il est désormais possible de tester la vitesse de lecture du disque

```
>hdparm -t /dev/sda
```

Remarque : l'option -t permet de désactiver le cache du disque

Pour tester la vitesse du réseau entre le système hôte et la VM, il suffit de saisir sur le système hôte :

```
>iperf -s
```

Puis de saisir dans la VM :

```
>iperf -c 192.168.0.2
```

Le résultat est exprimé en Mbits/s.

## Configuration de la machine virtuelle

Une fois le nécessaire des tests installé, nous allons détailler les opérations permettant d'activer la para-virtualisation au sein de la VM.

### Edition du fichier `/etc/initramfs-tools/modules`

Dans la VM, éditez le fichier `/etc/initramfs-tools/modules` et ajoutez-y le contenu suivant:

```
virtio
virtio_balloon
virtio_pci
virtio_rng
virtio_net
virtio_blk
```

Pour prendre en compte les précédentes modifications:

```
>update-initramfs -u
```

### Edition du fichier `/boot/grub/menu.lst`

L'activation de virtio remplace les labels `hd[a-z]` par `vd[a-z]`, il faut de ce fait modifier le fichier `/boot/grub/menu.lst` en remplaçant les occurrences `root=/dev/hda1` par `root=/dev/vda1`.

### Redémarrage de la VM en activant la para-virtualisation

Tout est normalement configuré pour que la para-virtualisation fonctionne: il suffit de relancer KVM avec quelques paramètres supplémentaires:

```
>kvm -drive file=~/  
lenny_thegeekcorner.raw,if=virtio,boot=on  
-vga std -soundhw all -smp 2 -m 1024 -net  
nic,macaddr=52:54:00:12:34:56,model=virtio -net  
tap,ifname=tap10,script=no
```

Remarque : l'usage de `-drive` à la place de `-hda` nous permet de spécifier l'interface virtio pour le disque.

### Tests de performances après activation de la para-virtualisation

En suivant la même procédure de tests que celle établie avant l'activation de la para-virtualisation on peut réaliser un tableau de comparaison :

	Hôte	VM sans virtio	VM avec virtio
Réseau	1GB/s	192MB/s	950MB/s
Disque	87MB/s	40MB/s	71MB/s

## V - Trucs et astuces

### a) Montage d'une image KVM sous linux

Il est possible d'accéder au système de fichier de la VM depuis le système hôte sans avoir à exécuter KVM. Pour cela, il suffit de monter l'image sur un répertoire du système hôte à l'aide de l'interface Loop du noyau Linux:

```
>sudo mkdir /mnt/fs_thegeekcorner  
>sudo mount -o loop,offset=32256 ~/lenny_thegeekcorner.raw /mnt/fs_thegeekcorner
```

Le répertoire `/mnt/fs_thegeekcorner` comporte désormais l'arborescence racine de votre VM.

Pour démonter ce répertoire :

```
>sudo umount /mnt/fs_thegeekcorner
```

Remarque: sur les nouvelles distributions l'interface Loop est active par défaut.

### b) Executer une machine virtuelle sans affichage graphique

La console graphique qui s'affiche et qui présente le démarrage de la machine virtuelle n'est pas systématiquement souhaitable en l'occurrence quand il s'agit de démarrer une VM sur un serveur hôte sans serveur X, ou de démarrer une VM à distance via SSH (évidemment sans tricher (sans `forward X ... -Y` ou `-X`))

La solution consiste à passer en paramètres les options `-nographic` et `-daemonize` à la commande `kvm` comme le figure l'exemple suivant:

```
>kvm -hda ~/lenny_thegeekcorner.raw -nographic -  
daemonize -vga std -soundhw all -smp 4 -m 1024 -net  
nic,macaddr=52:54:00:12:34:56 ,model=rt18139 -net  
tap,ifname=tap10,script=no
```

Quelques explications:

- `-nographic` permet effectivement de ne pas lancer la console graphique
- `-daemonize` permet d'exécuter KVM en démon ce qui nous rend immédiatement la main dans le shell et nous permet de le fermer (sans craindre d'éteindre la VM en cours d'exécution).

# Table des matières

---

I - Généralités.....	p. 1
a) Conventions typographiques.....	p. 1
Invite de commande.....	p. 1
Nature des utilisateurs.....	p. 1
Environnement d'exécution des commandes.....	p. 1
b) Remarques sur les commandes shell.....	p. 1
c) Vocabulaire.....	p. 1
d) Abréviations.....	p. 2
II - Préparation pour l'installation.....	p. 2
a) prise en charge de la virtualisation.....	p. 2
b) Installation de KVM et Qemu.....	p. 2
c) Chargement du module noyau KVM.....	p. 2
d) Ajout de l'utilisateur courant au groupe kvm .....	p. 2
III - Eléments d'administration.....	p. 2
a) Création de la machine virtuelle.....	p. 2
b) Installation de l'image virtuelle.....	p. 2
Téléchargement.....	p. 2
Démarrage de l'installation.....	p. 3
1er démarrage de la machine virtuelle installée.....	p. 3
Mise à jour du système invité Debian Lenny.....	p. 3
Installation du serveur SSH.....	p. 3
IV - Administration avancée.....	p. 3
a) Déploiement sur le réseau local.....	p. 3
Installation des dépendances.....	p. 3
Préparation du pont réseau (bridge) et de l'interface réseau virtuelle.....	p. 4
Ajout de l'utilisateur courant au groupe uml-net:.....	p. 4
Configuration du groupe tunusers:.....	p. 4
Edition du fichier 50-udev.rules .....	p. 4
Modification de /lib/udev/devices/net/tun .....	p. 4
Redémarrage du système hôte.....	p. 4
Le fichier /etc/network/interfaces .....	p. 4
Caractéristiques réseau du système hôte.....	p. 4
Modification du fichier /etc/network/interfaces .....	p. 4
Redémarrage du réseau et test de la configuration du système hôte.....	p. 4
Test du bridge.....	p. 4
Test de la création de l'interface réseau virtuelle.....	p. 4
Test de l'association du bridge avec l'interface réseau virtuelle.....	p. 5
Redémarrage de la machine virtuelle.....	p. 5
Test de la configuration réseau de la machine virtuelle.....	p. 5
b) Para-virtualisation.....	p. 5
Tests de performances avant l'activation de la para-virtualisation.....	p. 5
Installation des outils de test.....	p. 5
Test sans activation de la para-virtualisation.....	p. 5
Configuration de la machine virtuelle.....	p. 6
Edition du fichier /etc/initramf-tools/modules .....	p. 6
Edition du fichier /boot/grub/menu.lst .....	p. 6
Redémarrage de la VM en activant la para-virtualisation.....	p. 6
Tests de performances après activation de la para-virtualisation.....	p. 6
V - Trucs et astuces.....	p. 6
a) Montage d'une image KVM sous linux.....	p. 6
b) Executer une machine virtuelle sans affichage graphique.....	p. 6