Guide simplifié pour QEMU

Emmanuel Saracco

esaracco@free.fr
esaracco@users.labs.libre-entreprise.org
esaracco@easter-eggs.com

Guide simplifié pour QEMU

par Emmanuel Saracco

Historique des versions

Version 1.1 2006-04-22 Revu par : es
Ajout d'une partie sur le partage d'un même réseau virtuel entre plusieurs images. Remplacement de l'exemple de création d'image po
Version 1.0 2006-04-17 Revu par : es
Version initiale.

Table des matières

Liste des tableaux

Introduction

Malgré toutes les documentations déjà écrites à propos de QEMU, beaucoup d'utilisateurs ont encore du mal à le manipuler correctement, notamment lorsqu'il s'agit d'en configurer la couche réseau. C'est pourquoi j'ai décidé d'écrire ce tutoriel, en espérant qu'il éclairera un peu plus l'utilisation de ce fabuleux outil.

Définition Wikipédia (http://fr.wikipedia.org/wiki/QEMU)

> *QEMU est un « émulateur de système » : c'est un logiciel qui permet de faire tourner un ou plusieurs systèmes d'exploitation (ou seulement des processus) sur un système d'exploitation déjà installé sur la machine.*

QEMU peut être utilisé un peu à toutes les sauces. Que ce soit pour tester un système, une distribution, expérimenter la programmation noyau, faire des tests de sécurité sans mettre en péril votre environnement de travail, adapter vos développements à d'autres environnements etc.

Ce tutoriel se veut être un guide *pratique*, donc si vous cherchez des explications techniques et théoriques sur QEMU, le mieux est dans un premier temps de vous référer aux documentations rédigées par Fabrice Bellard, son auteur. Reportez vous également vers Annexe B pour plus d'informations.

Nous traiterons ici de la création et de la gestion de l'image d'une distribution Debian de GNU/Linux.

Chapitre 1. Configuration du noyau

Afin de pouvoir configurer correctement le réseau pour les images QEMU nous avons besoin d'une configuration noyau particulière. En plus de votre configuration habituelle, les éléments importants sont les suivants [1]:

```
Networking  --->
  Networking options  --->
    <*> Packet socket
    [*] TCP/IP networking
    [*]   IP: advanced router
    [*] Network packet filtering (replaces ipchains)  --->
      Core Netfilter Configuration  --->
        <*> Netfilter netlink interface
        <M>   "conntrack" connection tracking match support
      IP: Netfilter Configuration  --->
        <*> Connection tracking (required for masq/NAT)
        <M>   Connection tracking netlink interface (EXPERIMENTAL)
        <M>   FTP protocol support
        <M>   IRC protocol support
        <M>   NetBIOS name service protocol support (EXPERIMENTAL)
        <M> IP tables support (required for filtering/masq/NAT)
Network device support  --->
  [*] Network device support
  <M> Universal TUN/TAP device driver support
```

Une fois la configuration modifiée, recompilez le noyau et redémarrez dessus.

Notes

1. Nous utilisons ici un noyau 2.6.16.5.

Chapitre 2. Installation

QEMU est un logiciel sous licence Libre (GPL and LGPL (http://www.gnu.org/licenses/licenses.html)). Il peut très bien être utilisé tel quel. Néanmoins la vitesse d'exécution des systèmes et applications sur les composants émulés est lente et vous en viendrez rapidement à vous demander s'il n'existe pas un moyen d'accélérer un peu tout ça.

Il existe effectivement un moyen, mais il faut en passer par un brique propriétaire. Il s'agit d'un module noyau à charger avant le lancement de QEMU. Cet accélérateur est développé par Fabrice Bellard lui-même et permet un réel gain au niveau des performances. Nous l'utiliserons donc dans ce tutoriel.

2.1. KQEMU - L'accélérateur

Avertissement

Si vous décidez d'utiliser l'accélérateur, toutes les opérations de compilation doivent se faire avec une version de GCC inférieure à 4. Ceci vaut également pour votre noyau.

L'émulateur peut être téléchargé à partir de cette page (http://fabrice.bellard.free.fr/qemu/download.html). N'hésitez pas à prendre la toute dernière version disponible.

L'installation de se module est on ne peut plus simple. Il suffit de télécharger l'archive et de la décompresser sous le répertoire de QEMU. Pour la suite, reportez vous vers Section 2.2.

```
$ cd /usr/src/qemu-0.8.0/
$ wget http://fabrice.bellard.free.fr/qemu/kqemu-0.7.2.tar.gz
$ tar zxvf kqemu-0.7.2.tar.gz
```

Une fois que le module a été compilé et installé avec QEMU il est nécessaire d'effectuer quelques changements dans la configuration de `udev` afin que le device `/dev/kqemu` soit créé avec les droits nécessaires. Sur un système Debian, éditez le fichier `/etc/udev/permissions.rules` et y ajouter les lignes suivantes:

```
# kqemu
KERNEL=="kqemu",                MODE="0666"
```

Suite à cette modification, redémarrez `udev` avec **`/etc/init.d/udev restart`**.

A présent vous pouvez charger le module manuellement, en tant qu'utilisateur `root`:

```
# depmod -ae
# modprobe kqemu major=0
# dmesg | tail -4
QEMU Accelerator Module version 1.2.0, Copyright (c) 2005 Fabrice Bellard
This is a proprietary product. Read the LICENSE file for more information
Redistribution of this module is prohibited without authorization
KQEMU installed, max_instances=4 max_locked_mem=129300kB.
```

Pour éviter le chargement manuel de ce module par la suite, vous pouvez demander au système de le charger automatiquement. Sur une Debian, il faudra ajouter une ligne **`kqemu`** dans le fichier `/etc/modules` et créer un fichier `kqemu` sous `/etc/modprobe.d/` [1]:

```
# echo "qkemu" >> /etc/modules
# echo "options kqemu major=0" > /etc/modprobe.d/kqemu
# update-modules
```

2.2. QEMU - L'émulateur

L'émulateur peut être téléchargé à partir de cette page (http://fabrice.bellard.free.fr/qemu/download.html). N'hésitez pas à prendre la toute dernière version disponible.

```
$ cd /usr/src/
$ wget http://fabrice.bellard.free.fr/qemu/qemu-0.8.0.tar.gz
$ tar zxvf qemu-0.8.0.tar.gz
```

Une fois l'archive décompressée, allez dans le répertoire de qemu et lancez un **`./configure --help | less`** pour visualiser les différentes options de compilation disponibles. Les options intéressantes sont les suivantes:

```
  --prefix=PREFIX          install in PREFIX []
  --disable-kqemu          disable kqemu build
  --enable-adlib           enable Adlib emulation
  --enable-coreaudio       enable Coreaudio audio driver
  --enable-alsa            enable ALSA audio driver
  --enable-fmod            enable FMOD audio driver
  --enabled-dsound         enable DirectSound audio driver
```

Nous n'utiliserons dans ce tutoriel que les options `--prefix` (indication de la base du répertoire d'installation) et `--enable-alsa` (activation de la couche ALSA).

Ensuite, lancez la commande **`configure`** avec les options qui vous semblent les plus appropriées à votre configuration et à ce que vous voulez faire avec QEMU:

```
grinch@badiou:/usr/src/qemu-0.8.0$ ./configure --prefix=/usr --enable-alsa
Install prefix    /usr
BIOS directory    /usr/share/qemu
binary directory  /usr/bin
Manual directory  /usr/share/man
ELF interp prefix /usr/gnemul/qemu-%M
Source path       /usr/src/qemu-0.8.0
C compiler        gcc
Host C compiler   gcc
make              make
host CPU          i386
host big endian   no
target list       i386-user arm-user armeb-user sparc-user ppc-user mips-user mipsel-user i386-softmm
gprof enabled     no
static build      no
SDL support       yes
SDL static link   no
mingw32 support   no
Adlib support     no
CoreAudio support no
ALSA support      yes
DSound support    no
FMOD support      no
kqemu support     yes

KQEMU Linux module configuration:
kernel sources    /lib/modules/2.6.16.5/build
```

```
kbuild type       2.6
grinch@badiou:/usr/src/qemu-0.8.0$
```

N'oubliez pas de vous reporter vers Section 2.1 avant de compiler QEMU au cas ou vous voudriez utiliser l'accélérateur KQEMU.

Ensuite, lancer **`make`** puis **`su -c "make install"`** pour compiler et installer QEMU.

2.3. Arborescence locale

Je vous conseille de créer une arborescence dédiée à toutes vos images et scripts QEMU. L'arborescence décrite dans ce tutoriel correspond à celle attendue par le script de gestion qui l'accompagne: qemuctl (http://esaracco.free.fr/downloads/qemuctl).

Voici l'arborescence à construire:

```
$ mkdir -p ~/qemu/{debian,fedora,freebsd}/src
```

Nous avons là les répertoires `debian/`, `fedora/` et `freebsd/` qui contiendront les différents fichiers nécessaires à la gestion de systèmes spécifiques (dans le cas présent il s'agit respectivement d'une distribution GNU/Linux Debian (http://www.debian.org), d'une distribution GNU/Linux Fedora (http://fedora.redhat.com), et d'un système FreeBSD (http://www.freebsd.org)). Chacun de ces 3 répertoires contient un répertoire `src/` qui n'est pas vraiment nécessaire mais permet l'archivage des ISO utilisées pour la création des images QEMU.

Notes

1. Sur d'autres distributions, ou bien si vous utilisez un noyau 2.4.x vous devrez peut-être plutôt créer ce fichier sous `/etc/modutils/` et lancer un utilitaire comme **`update-modules`**.

Chapitre 3. Création de l'image

> **Avertissement**
>
> QEMU ne sait pas installer un système à partir de plusieurs images ISO.

QEMU ne travaille pas directement sur votre système de fichier, ou plutôt il a besoin d'un fichier contenant l'image d'un système. Il nous faut donc tout d'abord créer cette image et y installer un système.

QEMU peut installer des images à partir du lecteur de CDROM ou bien directement à partir d'un fichier ISO de CD ou de DVD. Lorsque vous installez une distribution fedora par exemple, pensez à le faire à partir de l'ISO DVD et pas à partir des ISO CD, sinon vous ne pourrez pas dépasser la première demande de changement de CD de l'installeur.

La façon la plus simple de créer une image et de lancer l'installation d'un système GNU/Linux Debian est de procéder comme suit [1]:

```
$ cd ~/qemu/debian/
$ qemu-img create debian.img 5G
$ cd src/
$ wget http://cdimage.debian.org/debian-cd/3.1_r1/i386/iso-cd/debian-31r1a-i386-netinst.iso
$ cd ../
$ qemu -localtime -m 256 -k fr -net nic -boot d -hda debian.img -cdrom src/debian-31r1a-i386-netinst.
```

Durant l'installation, n'hésitez pas à choisir de configurer le réseau via DHCP, nous reviendrons par la suite sur cette configuration pour forcer le système de l'image à utiliser un réseau local spécifique et nous lui allouerons une adresse IP fixe.

Notes

1. Il n'est pas nécessaire d'utiliser un DVD pour installer GNU/Linux Debian. Il suffit de télécharger une ISO minimale (http://www.debian.org/CD/netinst/) et de continuer l'installation via le réseau.

Chapitre 4. Configuration du réseau

Comme vous avez déjà dû le remarquer si vous avez déjà fait quelques essais avec QEMU, le réseau se configure rapidement à l'aide de l'option `-net user` [1]. Cependant cette configuration simple, si elle permet bien au système QEMU d'accéder au réseau, ne permet pas au système hôte d'y accéder.

Pour que le système QEMU s'intègre pleinement au réseau local il faut lui créer son propre réseau virtuel et router le traffic à l'aide d'une interface réseau virtuelle.

Assurez-vous d'avoir bien suivi les conseils de configuration du noyau Chapitre 1 avant de commencer.

Nous choisirons dans ce tutoriel de configurer un réseau `192.168.4.0/24`. Nous aurons donc la configuration suivante:

Tableau 4-1. Réseau virtuel

Nom	Valeur
Interface	tap0
Réseau	192.168.4.0/24
Adresse hôte	192.168.4.1
Adresse image	192.168.4.2

Dans un premier temps il nous faut configurer le système pour qu'il puisse charger correctement le module qui nous permettra de mettre en place un pont. Sur un système Debian, éditez le fichier `/etc/udev/permissions.rules` et y ajouter les lignes suivantes:

```
# tun
KERNEL=="tun",                  MODE="0666"
```

Suite à cette modification, redémarrez `udev` avec **`/etc/init.d/udev restart`**.

4.1. Configuration de l'hôte

QEMU va automatiquement chercher à exécuter un fichier `/etc/qemu-ifup` au démarrage. Cela va

nous permettre de configurer le réseau de l'image.

Nous allons donc créer un fichier `/etc/qemu-ifup` (téléchargeable ici (http://esaracco.free.fr/downloads/qemu-ifup)), dont le contenu sera le suivant:

```
#!/bin/sh

NET=192.168.4.0/24
GATEWAY=192.168.4.1

/sbin/ifconfig $1 $GATEWAY

/sbin/modprobe iptable_nat
/sbin/modprobe ip_nat_irc
/sbin/modprobe ip_nat_ftp
/sbin/modprobe iptable_filter
/sbin/modprobe ip_conntrack_ftp
/sbin/modprobe ip_conntrack_irc
/sbin/modprobe ip_conntrack_netbios_ns
/sbin/modprobe ip_conntrack_netlink

/sbin/iptables -D POSTROUTING -t nat -s $NET -d ! $NET -j MASQUERADE
/sbin/iptables -t nat -s $NET -d ! $NET -A POSTROUTING -j MASQUERADE
/bin/echo 1 > /proc/sys/net/ipv4/ip_forward
```

Assurez-vous que tous les modules noyau chargés ici sont bien présents dans le répertoire `/lib/modules/2.6.16.5/kernel/net/ipv4/netfilter/` de l'hôte. N'hésitez pas à commenter les lignes qui ne vous seront pas nécessaires (comme celles concernant le chargement des modules `ip_nat_irc`, `ip_conntrack_irc`, `ip_conntrack_netbios_ns` etc., par exemple).

Le rôle de ce script est de configurer l'interface virtuelle nouvellement créée pour QEMU. Celui-ci passe son nom en premier paramètre lors de l'appel. Ensuite on charge les modules nécessaires, puis on crée les règles qui permettront à l'hôte et à l'image QEMU de communiquer via le réseau `$NET`.

4.2. Configuration de l'image

Lors de la première installation du système sur l'image vous avez certainement configuré le réseau pour qu'il se configure automatiquement via le protocole DHCP. A présent nous allons revoir la configuration pour l'adapter à notre réseau virtuel.

Lancez votre image avec la commande suivante:

```
$ qemu -localtime -m 256 -k fr -net nic -boot c -hda ~/qemu/debian/debian.img
```

Identifiez vous en tant que root et éditez le fichier `/etc/network/interfaces` pour modifier la section de configuration de l'interface `eth0` comme suit:

```
auto eth0
iface eth0 inet static
        address 192.168.4.2
        netmask 255.255.255.0
        network 192.168.0.0
        broadcast 192.168.0.255
        gateway 192.168.4.1
```

Ensuite arrêtez l'image QEMU.

A présent tout devrait être prêt. Pour tester votre nouveau réseau, lancez l'image QEMU en tant qu'utilisateur `root` [2] avec les options suivantes [3]:

```
# qemu -localtime -m 256 -k fr -net nic -net tap -boot c -hda /home/grinch/qemu/debian/debian.img
```

4.3. Partager un réseau virtuel

Il est possible de faire tourner plusieurs instances de QEMU sur le même réseau virtuel. Il faut pour cela créer et configurer une première instance comme expliqué dans les chapitres précédents. Ensuite les différences se trouvent principalement dans les options de lancement de chaque image.

4.3.1. Serveur de réseau

Le serveur de réseau doit être unique. Si vous avez configuré une première image comme indiqué dans ce tutoriel, celle-ci mettra à disposition des autres instances un réseau `192.168.4.0/24`.

Cette première instance doit être lancée avec les options suivantes:

```
# qemu -localtime -m 256 -k fr -net nic -net tap -net socket,listen=127.0.0.1:1234 -boot c -hda /home/
```

A présent l'image fait office de serveur de réseau virtuel. Elle écoute en local sur le port `1234`.

> **Note :** Le choix de l'adresse IP et du port est libre. A vous de faire attention à ce que le port choisi ne soit pas déjà utilisé.

4.3.2. Clients du réseau

Avant de faire partager le réseau à d'autres instances il faut s'assurer qu'il n'y a pas de conflit au niveau de leur adresse IP. Il faudra donc attribuer à chacune une adresse unique sur le réseau virtuel.

Toutes les autres instances doivent être lancées avec les options suivantes:

```
# qemu -localtime -m 256 -k fr -net nic -net tap -net socket,connect=127.0.0.1:1234 -boot c -hda /home
```

Avertissement

Pensez bien à indiquer ici le même couple adresse IP/port que celui spécifié au lancement du serveur de réseau.

Notes

1. Cette option est utilisée par défaut pour la configuration du réseau.
2. Nous avons besoin des droits superutilisateur pour manipuler `ifconfig` et **iptables**. Nous verrons Section 5.1 comment utiliser la table `sudoers` à cette fin.
3. Assurez-vous d'avoir bien installé et configuré le script `/etc/qemu-ifup` auparavant (voir Section 4.1).

Chapitre 5. Automatisation

La gestion des images QEMU devient vite fastidieuse. Une automatisation s'impose donc.

5.1. Utilisation de sudo

Pour manipuler les modules et utiliser **iptables** nous avons besoin d'avoir les droits de l'utilisateur `root`. Donc plutôt que d'utiliser directement ce compte pour gérer les images QEMU nous allons passer par l'utilitaire **sudo**, qui permet de donner à un utilisateur ou un groupe les droits d'autres utilisateurs ou d'autres groupes.

sudo se configure dans le fichier `/etc/sudoers`. Vous pouvez éditer directement ce fichier, néanmoins il est préférable d'utiliser la commande **visudo** si elle est disponible sur votre système. En tant qu'utilisateur root, ajoutez les lignes suivantes:

```
grinch  ALL=NOPASSWD:/usr/bin/qemuctl
```

Pensez à remplacer `grinch` par le nom de l'utilisateur système de votre choix. Nous verrons Section 5.2 à quoi correspond l'utilitaire **qemuctl**, et comment l'utiliser pour la gestion des images QEMU.

Pour vérifier que la configuration de **sudo** est correcte, utilisez la commande **`sudo -l`**, qui devrait sortir quelque chose comme:

```
grinch@badiou:~$ sudo -l
User grinch may run the following commands on this host:
    (root) NOPASSWD: /usr/bin/qemuctl
grinch@badiou:~$
```

A présent l'utilisateur système `grinch` pourra lancer le script `/usr/bin/qemuctl` avcec les droits de l'utilisateur `root` sans fournir de mot de passe, en utilisant la commande **`sudo qemuctl`**.

5.2. Utilisation de qemuctl

> **Avertissement**
>
> Pensez bien à lancer la commande **qemuctl** en la faisant toujours précéder de **sudo**. Dans le cas contraire le script s'écuterait sans les droit de l'utilisateur `root` et s'arrêterait en affichant un message d'erreur.

Afin d'automatiser les tâches de création, de démarrage et d'arrêt des images QEMU j'ai mis au point un script nommé **qemuctl** (téléchargeable ici (http://esaracco.free.fr/downloads/qemuctl)). Une fois le script téléchargé, déposez-le en tant que `root` dans `/usr/bin/` et rendez-le exécutable à l'aide de **`chmod +x /usr/bin/qemuctl`**. Ensuite vérifiez que vous avez bien suivi ce que nous avons dit Section 5.1 pour pouvoir utiliser ce script avec les droits de l'utilisateur `root`.

Avant toute chose, pensez à adapter la configuration du script à votre système. Les constantes à modifier se trouvent en début de script:

```
use constant QEMU_PATH => "/home/grinch/qemu";
use constant QEMU_RAM_SIZE => 256;
use constant QEMU_GRAPHIC => '-nographic';
use constant QEMU_VLAN_IP => '127.0.0.1';
use constant QEMU_VLAN_PORT => '1234';
```

QEMU_PATH

La constante `QEMU_PATH` pointe sur le répertoire dans lequel se trouve vos répertoires d'images QEMU.

QEMU_RAM_SIZE

La constante `QEMU_RAM_SIZE` indique la taille maximum de la RAM à allouer pour le système contenu dans l'image QEMU.

QEMU_GRAPHIC

La constante `QEMU_GRAPHIC` contient l'option de contrôle de l'affichage. Si elle contient `-nographic`, aucun display X ne sera utilisé lors du lancement. Si elle ne contient rien (une chaîne vide), QEMU utilisera un display X.

QEMU_VLAN_IP

La constante `QEMU_VLAN_IP` contient l'adresse IP sur laquelle est géré le VLAN. Il ne s'agit pas obligatoirement de la boucle locale. Elle est utilisée lorsque l'option `vls` (VLAN serveur) ou `vlc`

(VLAN client) est passée à **qemuctl**.

QEMU_VLAN_PORT

La constante `QEMU_VLAN_PORT` contient le numéro de port sur lequel est géré le VLAN. Il est utilisé lorsque l'option `vls` (VLAN serveur) ou `vlc` (VLAN client) est passé à **qemuctl**.

Note : Une fois que le système est installé et configuré sur l'image je vous conseille de ne plus utiliser de display X et d'accéder à votre image via SSH depuis la machine hôte.

Le fonctionnement du script est très simple. Dans un premier temps, lancez-le sans arguments afin d'afficher l'aide:

```
grinch@badiou:~/qemu$ qemuctl

    Version: 2006042300

    Usage  : qemuctl imgname [install|start|stop] [vls|vlc]
    Example: qemuctl freebsd start

    Available QEMU images:

        - fedora
        - freebsd
        - debian
        - gentoo

    Report bugs to <esaracco@users.labs.libre-entreprise.org>

grinch@badiou:~/qemu$
```

Par défaut, **qemuctl** affiche une aide ainsi que la liste des différentes images détectées.

Le synopsis est le suivant:

qemuctl *imgname* [install | start | stop] [vls | vlc]

5.2.1. Création d'une image

qemuctl permet de simplifier la création d'une image QEMU. Il faut néanmoins lui préparer un peu le terrain. Le script s'attend à trouver l'arborescence décrite Section 2.3. Si nous décidions par exemple

d'installer une image du système FreeBSD (http://www.freebsd.org) nous préparerions le terrain comme suit:

```
$ cd ~/qemu/
$ mkdir -p freebsd/src
$ cd freebsd/src/
$ wget "ftp://ftp.freebsd.org/pub/FreeBSD/ISO-IMAGES-i386/6.1/6.1-RC1-i386-disc1.iso"
$ ln -s 6.1-RC1-i386-disc1.iso freebsd.iso
$ cd ../
$ qemu-img create freebsd.img 5G
```

Ce qui nous donnerait au final:

```
grinch@badiou:~/qemu/freebsd$ pwd
/home/grinch/qemu/freebsd
grinch@badiou:~/qemu/freebsd$ ls -lR
.:
total 4
-rw-r--r-- 1 grinch grinch 5368709120 2006-03-13 18:43 freebsd.img
drwxr-xr-x 2 grinch grinch       4096 2006-04-17 19:47 src

./src:
total 49728
lrwxrwxrwx 1 grinch grinch       30 2006-04-17 19:47 freebsd.iso -> 6.1-RC1-i386-disc1.iso
-rw-r--r-- 1 grinch grinch 50862080 2006-03-13 18:25 6.1-RC1-i386-disc1.iso
grinch@badiou:~/qemu/freebsd$
```

Une fois le terrain préparé il suffirait de lancer **`sudo qemuctl freebsd install`**.

5.2.2. Exécution d'une image

Pour exécuter une image (`debian` dans notre cas), utilisez **`sudo qemuctl debian start`**.

Un fichier de log est automatiquement créé directement sous le répertoire d'hébergement de l'image. Ici sous `~/qemu/debian/`. Ce fichier porte le nom de l'image: `debian.log`. Le script crée également un fichier `debian.pid` dans lequel est stocké l'identifiant du processus correspondant à l'image QEMU démarrée.

Pour lancer une image en tant que serveur de réseau virtuel (voir Section 4.3.1) il suffit de préciser l'argument `vls` après l'action: **`sudo qemuctl debian start vls`**.

Pour lancer une image en tant que client d'un réseau virtuel (voir Section 4.3.1) il suffit de préciser l'argument `vlc` après l'action: **`sudo qemuctl debian start vlc`**.

5.2.3. Arrêt d'une image

> **Avertissement**
>
> Il est recommandé de ne jamais arrêter manuellement une image lancée avec **qemuctl**. Les images lancées avec **qemuctl** doivent être arrêtées avec **qemuctl**.

L'arrêt d'une image est aisé lorsqu'on l'a démarrée en ne lui interdisant pas d'utiliser un display graphique. Néanmoins l'utilisation d'un display pouvant causer des problèmes au sein de X (instabilité du comportement de la souris, instabilité de la console graphique etc.) il est plus simple de passer l'option `-nographic` à QEMU via la constante `QEMU_GRAPHIC` de **qemuctl** (voir Section 5.2 pour les détails).

Pour arrêter une image (`debian` dans notre cas), pensez à stopper son système proprement (via la command `halt` par exemple), puis après avoir attendu un peu le temps que ses processus se terminent, utilisez **`sudo qemuctl debian stop`**.

Chapitre 6. Conclusion

Les scripts *compagnons* présentés dans ce tutoriels sont susceptibles d'évoluer en fonction de vos remarques. L'Annexe A sera tenue à jour en fonction de ces évolutions. N'hésitez pas à la consulter régulièrement.

Annexe A. Fichiers compagnons

Vous trouverez dans cette section tous les fichiers *maison* dont il est question dans ce document.

Script de gestion des images QEMU (http://esaracco.free.fr/downloads/qemuctl) (voir Section 5.2) - Dernière version 2006042300.

Script de configuration du réseau virtuel (http://esaracco.free.fr/downloads/qemu-ifup) (voir Section 4.1) - Dernière version 2006041700.

Annexe B. Liens utiles

Site officiel de QEMU (utilisateurs) (http://fabrice.bellard.free.fr/qemu/).

Site officiel de QEMU (développeurs) (http://savannah.nongnu.org/projects/qemu).

QEMU sur Wikipédia (http://fr.wikipedia.org/wiki/QEMU).

Images QEMU prêtes à l'emploi (http://free.oszoo.org/).

Interface graphique en GTK pour QEMU (http://gtk-qemu.sourceforge.net/).

Un module noyau d'accélération Libre (http://savannah.nongnu.org/projects/qvm86/).

B.1. Autres tutoriels et Howto

Installer et utiliser QEMU [Léa-Linux]
(http://lea-linux.org/cached/index/Software-soft_emul-qemu.html).

Utiliser QEMU avec l'accélération KQEMU
(http://genibel.org/blog/index.php/2005/11/08/45-utiliser-qemu-avec-l-acceleration-kqemu).

Installation de QEMU sur Ubuntu
(http://www.codepoets.co.uk/docs/qemu_windows2000_on_ubuntu_linux_howto).

Installation de QEMU sur gentoo (http://gentoo-wiki.com/HOWTO:_Qemu).