Qemu

- Objet : Qemu
- Niveau requis :
  avisé
- Commentaires : *logiciel libre de machine virtuelle*
- Suivi :
  à-tester
  - Création par captnfab le 17/08/2009
  - Testé par arpinux le 02/10/2021
- Commentaires sur le forum : C'est ici[1)]

Résumé

Ce logiciel permet de créer une machine virtuelle avec un large choix d'architecture (i.e. de type d'ordinateurs) sur sa propre machine, permettant par exemple d'installer une Debian pour tester la future version, ou d'installer un android version arm64, ou tester un autre système d'exploitation comme

- une autre distribution Linux,
- un os à base de noyau BSD,
- un système propriétaire,
- etc.

Installation

Classiquement, on commence par mettre à jour la liste des paquets :

```
apt-get update
```

puis on installe qemu :

```
apt-get install qemu-system qemu-utils qemu-system-gui
```

Prérequis

Pour gagner en fluidité, Qemu utilise KVM.

- KVM est un hyperviseur de type 1 (“bare metal”) qui permet à Qemu la virtualisation simultanée de différents systèmes d'exploitation sur la même machine hôte.
- KVM est un module du noyau Linux permettant aux applications de l'espace utilisateur d'utiliser les fonctionnalités de virtualisation matérielle.
- KVM permet à Qemu de virtualiser une architecture sur un hôte de même architecture et

exploitera alors pleinement l'accélération matérielle que KVM pourra lui apporter.
  - L'accélération matérielle par KVM ne sera possible que pour les microprocesseurs pourvu de la technologie `Intel VT-x` et `AMD-V`.
  - L'accès direct au matériel (bus PCI, Carte Graphique, etc..) depuis les machines virtuelles nécessitera aussi que la carte mère soit équipée d'un *chipset* permettant d'exploiter la technologie `IOMMU` (**I**nput/**O**utput **M**emory **M**anagement **U**nit) de ces microprocesseur. ⇒ `Intel Vt-d` et `AMD-Vi`.

Pour vérifier que le microprocesseur de votre machine permet la virtualisation avec KVM

```
grep -E 'vmx|svm' /proc/cpuinfo &>/dev/null && echo "La virtualisation est possible sur cette machine." || echo "Le microprocesseur de cette machine ne permet pas d'utiliser la virtualisation avec KVM."
```

Et assurez-vous ensuite que les option du BIOS concernant ces technologies de virtualisation sont activées.

Merci à **bendia** *pour ses remarques sur ce tuto dans le forum*

N'hésitez pas chacun à intervenir dans le forum pour signaler des compléments à mettre ou à modifier dans les tutos que vous visitez ! 😉

Configuration

Les configurations de machines virtuelles peuvent vite devenir très complexes. En effet, si l'on veut préciser les paramètres de chaque périphériques virtuel associé, on n'en fini pas. C'est pourquoi à moins de savoir exactement ce que l'on veut, on n'utilise en général pas directement qemu, mais on l'utilise en passant par une outil plus haut niveau tel libvirt et son gestionnaire de machines virtuelles virt-manager.

De quoi a-t-on besoin pour faire une VM ?

Un microprocesseur

En fait, on parle plus généralement d'architecture.

- Si l'on est sur un ordinateur fixe, il y a de grandes chances pour que l'architecture soit x86_64, c'est ce que debian appelle `amd64` et correspond aux processeurs 64bits communs.
- Si l'on est sur un ordinosaure fixe, alors il est possible que l'architecture soit x86, c'est ce que debian appelle `i386` et correspond aux processeurs 32bits du début du siècle.
- Si l'on est sur un smartphone, alors on est très probablement sur une architecture arm64.

Là, la question que l'on doit se poser est: « Quelle est l'architecture de la machine que je veux simuler ? » Donc si l'on veut installer un Android, c'est probablement arm64, mais si l'on veut installer une debian sid pour tester, il y a des chances que ça reste x86_64.

Pour lister les commandes qemu associées à chaque architecture, lancer la commande:

```
ls /usr/bin/qemu-system-*
```

On indique ensuite le nombre de cœurs que l'on souhaite dédier à la machine via l'option `-smp`

Exemple:

```
qemu-system-x86_64 -smp 2
```

Un disque de stockage

C'est a priori là où sera installé le bootloader et système d'exploitation que l'on veut lancer. Ça peut être un lecteur CD, une clef USB, mais ça peut aussi être une image disque le représentant.

Attention, utiliser une partition comme image disque est possible mais vous ne pourrez pas monter la partition en tant que système de fichier sur l'hôte car elle ne contiendra pas un système de fichier mais un disque entier.

Souvent, on crée des images disques avec `qemu-img`. Il permet en particulier de manipuler les images au format `.qcow2` ou de convertir des images faites pour d'autres logiciels de VM.

Exemple:

```
qemu-img create -f qcow2 hdd.qcow2 20G
```

Où :

- `hdd.qcow2` est le nom du fichier *image disque* qui va être créé.
- `10G` sera la taille **virtuelle** de l'espace disponible dans ce **disque virtuel**.

Le format **qcow2**[2)] est un format d'espace de stockage *optimisé*, c'est à dire que l'espace occupé par le fichier *image disque* sera très nettement inférieur à l'espace disponible sur le support de données, mais à mesure que des données seront écrites sur le *disque* la taille du fichier va augmenter.

ATTENTION ! qemu utilise l'unité Gio, mais certains installateurs tels celui de Debian utilise l'unité Go :

- 1 Go = 10^9 octets
- 1 Gio = 2^{30} octets = 1,073741824 Go

et donc 10 Gio correspondront à 10,7 Go.

Un grand merci à **vince06fr**, *sur le forum là ! Lucide, le gars* 😃

Il est important de choisir un support de stockage dont la taille et la vitesse correspondent à ce que

l'on veut faire. On n'utilisera pas une clef USB pour installer une VM destinée aux jeux vidéo.

On utilise `-hda`, `-hdb`, etc. pour indiquer les supports de stockages de type disque dur, et `-cdrom` pour indiquer des media en lecture seule tels les cdroms.

Exemple:

```
qemu-system-x86_64 -hda /srv/VM/archlinux/hdd.qcow2 -cdrom
/home/captnfab/Téléchargements/archlinux.iso
```

Mémoire vive

On veut souvent spécifier la taille de la RAM disponible, cela se fait en utilisant l'option `-m` et en précisant l'unité (`M` pour des Mio, `G` pour des Gio').

Exemple:

```
qemu-system-x86_64 -hda /srv/VM/archlinux/hdd.qcow2 -cdrom
/home/captnfab/Téléchargements/archlinux.iso -m 2G
```

Les périphériques USB

Pour rajouter un contrôleur USB virtuel, on rajoute l'option `-usb`, ensuite on peut rajouter

- des ports USB via un paramètre du type `-usbdevice host:bus.addr`
- ou des périphériques USB via un paramètre du type `-usbdevice host:vendor_id:product_id` pour un périphérique.

Exemple :

Si la commande `lsusb` retourne la ligne suivante :

retour de la commande

```
Bus 007 Device 003: ID 0b97:7772 Blahhhhhhhhhh
```

Si je veux monter le port USB sur lequel est branché mon périphérique Blahhhhhhhhhh, qui restera connecté même si je change le périphérique branché dessus:

```
qemu-system-x86_64 -hda /srv/VM/archlinux/hdd.qcow2 -cdrom
/home/captnfab/Téléchargements/archlinux.iso -m 2G -usb -usbdevice
"host:007.003"
```

Si je veux monter Blahhhhhhhhhh où qu'il soit branché:

```
qemu-system-x86_64 -hda /srv/VM/archlinux/hdd.qcow2 -cdrom
/home/captnfab/Téléchargements/archlinux.iso -m 2G -usb -usbdevice
```

```
"host:0b97:7772"
```

Les autres périphériques

Une carte son, une carte graphique, des périphériques USB, le nombre de cœurs du microprocesseur, la carte réseau...

Il serait trop long de tout détailler ici, et ce n'est pas l'objet. Chaque cas d'utilisation est unique et nécessite ses propres options.

Pour en savoir plus, consulter le manuel:

```
man qemu-system
```

Enregistrement des options

Il est fastidieux de se souvenir des options à chaque lancement, on peut donc écrire un fichier contenant la commande complète, exemple:

```
touch "launch_vm.sh"
```

On édite le fichier de sorte à ce qu'il ressemble à ça :

launch_vm.sh

```
#!/bin/sh
qemu-system-x86_64 -hda /srv/VM/archlinux/hdd.qcow2 -m 2G -usb -usbdevice "host:0b97:7772" "$@"
```

On le rend exécutable :

```
chmod a+x "launch_vm.sh"
```

Lancement de la machine la première fois

On veut booter sur le CDRom d'installation, donc

```
./launch_vm.sh -cdrom /home/captnfab/Téléchargements/archlinux.iso -boot d
```

Lancement de la machine les autres fois

```
./launch_vm.sh
```

Aller plus loin

Faire des tests sans pourrir sa config

Si on utilise sa vm pour tester des logiciels non stables, on n'a certainement pas envie de ~~foutre en l'air, détruire,~~ perdre sa jolie installation vierge toute belle toute neuve sur sa machine virtuelle.

Faire une copie de sauvegarde de l'image prendrait beaucoup de place. Heureusement, il y a mieux ! Il y a la possibilité de créer une image disque basée sur une autre image disque en lecture seule avec qemu-img.

Exemple: Une image disque `hdd.qcow2` contient un système fragile que l'on a peur de casser. On va renommer ce disque en `hdd-base.qcow2`, le passer en lecture seule, puis créer une image `hdd.qcow2` qui sera basée sur la première ne n'enregistrera que les modifications par rapport à celle-ci.

```
mv hdd.qcow2 hdd-base.qcow2
chmod a-w hdd-base.qcow2
qemu-img create -b hdd-base.qcow2 -f qcow2 hdd.qcow2
```

Et voilà, maintenant, `launch_vm.sh` utilisera `hdd.qcow2` sans modifier `hdd-base.qcow2`

Quelques raccourcis indispensables

RACCOURCIS CLAVIER	***ACTION***
Ctrl+Alt+F	passe en plein-écran
Ctrl+Alt+1	affichage graphique de l'invité
Ctrl+Alt+2	passage au moniteur Qemu; attention, dans ces deux options, le 1 et le 2 sont à taper sur les chiffres hauts du clavier, pas sur le pavé numérique
Ctrl+Alt	pour arrêter le contrôle de la souris dans l'invité, alors que pour l'activer, on avait cliqué dans la fenêtre de Qemu

voir le très bon tuto sur lea-linux là :

http://lea-linux.org/documentations/index.php/Software-soft_emul-qemu

Utiliser lecteur CDRom de l'hôte

Au moment de lancer votre machine virtuelle, ajoutez `-cdrom /dev/sr0` dans la ligne de commande.

Merci à MicP pour le tuyau. 😎

Utiliser la clef USB de l'hôte comme périphérique USB

Insérez sur et identifiez la partition d'une clé usb avec la commmande lsblk, mettons par exemple que

ce soit `/dev/sdg`

Et pour lancer votre machine virtuelle sous qemu rajoutez le paramètre suivant, en spécifiant le périphérique trouvé dans lsblk (à la place de `/dev/sdg` :

```
-usbdevice disk:/dev/sdg
```

Merci pour cette information à ce cher **adrien** *sur le salon #slackware-fr qu'est un vrai geek sympa !* 😀

Faire une installation à distance

Si vous procédez à une installation distante, rajouter à la fin de la ligne:

```
--vnc :0 -k fr
```

Ainsi, vous devriez pouvoir vous connecter via un client VNC tout en gardant une disposition clavier AZERTY ;)

Changer de CDROM en cours d'installation d'un logiciel sur la machine virtuelle

Imaginons un instant qu'à partir d'un support CDROM avec grande souffrance morale vous ayez lancé l'installation d'un logiciel privateur avec “wine” dans votre machine virtuelle qemu (pour éviter que de potentiels vilains malwares viennent titiller votre OS vénéré).
Tout se déroule correctement et soudain l'installateur vous demande d'insérer le CDROM N°2 !!!

La tuile...
Comment faire, car on a beau changer le CDROM, qemu reste incapable de le lire ?

Ne cherchez pas non plus à faire de pénibles opérations de démontage-remontage du périphérique, c'est l'échec assuré !

Pas de panique, il y a un moyen simple d'y arriver en utilisant le moniteur Qemu ! 😀

Toujours dans la machine virtuelle commencez par appuyer sur le raccourci clavier CTRL+ALT+2 (dans la VM qemu) vous accéderez ainsi au moniteur qemu (qui ressemble à un terminal...).

Première chose à faire, il faut identifier les lecteurs vus par Qemu en entrant la commande suivante :

séquence interactive

```
(qemu) info block
```

retour de la commande

```
ide0-hd0: /home/<user>/<RépertoireImage>/image.qcow2 (qcow2)

ide1-cd0: /dev/sr0 (raw,read-only)
    Removable device: locked, tray closed

floppy0: [not inserted]
    Removable device: not locked, tray closed

sd0: [not inserted]
    Removable device: not locked, tray closed
```

La ligne **ide1-cd0: /dev/sr0 (raw,read-only)** désigne le lecteur cdrom monté par le système virtuel. Il est actuellement verrouillé (locked).

Il faut maintenant éjecter le cdrom.

Commencez par appuyer sur le bouton (physique) d'éjection du tiroir cdrom de votre PC et **laissez-le ouvert**.
Puis entrez les deux commandes suivantes dans le moniteur Qemu :

séquence interactive

```
(qemu) eject ide1-cd0
```

séquence interactive

```
(qemu) info block
```

retour de la commande

```
ide0-hd0: /home/<user>/<RépertoireImage>/image.qcow2 (qcow2)

ide1-cd0: /dev/sr0 (raw,read-only)
    Removable device: not locked, tray closed

floppy0: [not inserted]
    Removable device: not locked, tray closed

sd0: [not inserted]
    Removable device: not locked, tray closed
```

Removable device: not locked = Le lecteur est déverrouillé.

Insérez **le nouveau** cd dans le lecteur cdrom, fermez le tiroir puis entrez les deux commandes suivantes dans le moniteur Qemu :

séquence interactive

```
(qemu) change ide1-cd0 /dev/sr0
```

séquence interactive

```
(qemu) info block
```

retour de la commande

```
ide0-hd0: /home/<user>/<RépertoireImage>/image.qcow2 (qcow2)

ide1-cd0: /dev/sr0 (raw,read-only)
    Removable device: locked, tray closed

floppy0: [not inserted]
    Removable device: not locked, tray closed

sd0: [not inserted]
    Removable device: not locked, tray closed
```

C'est bon le nouveau CD est correctement reconnu par Qemu. 😎

Appuyez sur les touches de raccourci CTRL+ALT+1 pour retourner dans le mode graphique de l'invité et poursuivez votre installation.

C'est le moment de verser une larme de joie et de chanter un psaume car un miracle s'est accompli.

On ne peut être qu'ému n'est-ce pas ? 😀

Ajouter une interface réseau de type "bridge" (pont)

Afin de bénéficier de toute la commodité d'un réseau où les machines virtuelles créées sont bien intégrées au réseau des machines physiques déjà présentes, suivre d'abord ce tuto :

- Configurer une interface réseau virtuelle

Activez l'interface réseau et démarrez la machine virtuelle

Une fois l'installation terminée, démarrez votre machine virtuelle et son interface réseau.

Activez l'interface réseau `tap0` est activé :

```
ifup tap0
```

Démarrez votre machine virtuelle par cette commande :

```
kvm -net nic,model=rtl8139,vlan=0,macaddr=00:11:22:33:44:55 -net
```

```
tap,vlan=0,ifname=tap0,script=no -enable-kvm -m 512 image.qcow2
```

Pare feu et Qemu

Un post dans le forum à donné une solution pour configurer le firewall avec KVM : [Résolu] Virt-Manager QEMU/KVM Aucune conexion réseau.

Problèmes rencontrés

Problème de carte graphique

Le 27/02/2012 Après l'installation d'une squeeze sur une VM Qemu/KVM sous fedora, lors du boot j'ai eu le message :

```
unaligned pointer 0x8ef10002
```

Le problème a été résolu en changeant le modèle de carte graphique virtuelle pour `-vga std` ou `-vga cirrus` à la place de `-vga qxl` ou `-vga vmvga`.

Voir sur le forum :

- http://debian-facile.org/forum/viewtopic.php?pid=42774#p42774

Merci **chiwawa** *t'est pas chien de nous l'avoir indiqué, sûr !* 😀

Tutos

1. Installation de Fenêtre XP sur une VM
2. Installation de Slackware sur une VM

Références

1. `man qemu` (anglais)
2. http://www.esaracco.fr/documentation/qemu/qemu/ (ou en pdf) sur le site http://www.esaracco.fr/
3. http://virt.kernelnewbies.org/KVM_Multiuser_Usage?highlight=%28kvm%29%7C%28usb%29 *Offert par* **arthefact** *sur irc*
4. http://doc.ubuntu-fr.org/kvm *Nos habituels amis utilisant la distribution Debian.*
5. http://fr.wikipedia.org/wiki/Adresse_MAC

1) N'hésitez pas à y faire part de vos remarques, succès, améliorations ou échecs !

2) **Q**emu **C**opy **O**n **W**rite version **2**

From:
http://debian-facile.org/ - **Documentation - Wiki**

Permanent link:
http://debian-facile.org/doc:systeme:vm:qemu

Last update: **14/08/2023 19:45**