Qemu, un émulateur de processeur

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 15 mai 2006

https://www.bortzmeyer.org/qemu.html

Dans la grande famille des solutions de virtualisation, permettant de faire tourner un système d'exploitation sur un autre, Qemu appartient à la sous-famille des émulateurs de processeur, comme Bochs.

Cela veut dire qu'il lit les instructions machine en mémoire et les évalue, comme ce que fait le CPU (mais bien plus lentement).

Cela permet de faire tourner des systèmes d'exploitation non modifiés, dont on n'a qu'un binaire, comme par exemple Microsoft Windows. Contrairement à User-Mode-Linux <https://www.bortzmeyer.org/user-mode-linux.html>, il n'est donc pas spécifique de Linux et, contrairement à Xen <https://www.bortzmeyer.org/xen.html>, il ne nécessite pas de modifier le système d'exploitation invité.

En outre, Qemu peut tourner sur divers processeurs et émuler divers processeurs donc on pourrait (je n'ai pas testé) faire fonctionner une Sparc virtuelle sur un PC.

J'ai testé Qemu sur ma Debian et tout s'est bien passé, y compris le réseau (contrairement à ce qu'on lit souvent).

Je lance Qemu avec `qemu -hda images/freebsd_6.0/freebsd_6_0.img -boot c` et Qemu émule un disque dur IDE (Qemu n'émule pas seulement le processeur mais aussi un PC complet) avec le fichier `freebsd_6_0.img` et démarre dessus. Ainsi, j'ai un FreeBSD complet sur ma Debian. On trouve énormément d'images toutes prêtes, pour les principaux systèmes libres, sur Free OS Zoo <http://free.oszoo.org/>.

Si on préfère installer un système soi-même, on crée l'image vide :

```
% qemu-img create -f qcow  tmp/fenetres.img 100G
```

Puis on lance l'émulateur, ici in indiquant de démarrer sur le CD-ROM, qui contient un CD d'installation :

```
% qemu -cdrom /dev/cdrom -hda tmp/fenetres.img -boot d
```

Pour le réseau, Qemu lance par défaut `/etc/qemu-ifup` qui, chez moi, contient :

```
#!/bin/sh

iface=$1

# The Qemu host
myipv4addr=192.134.4.69

# The guest virtual machine
remipv4addr=192.134.4.79

# IPv4
sudo /sbin/ifconfig $iface $myipv4addr  netmask 255.255.255.255
sudo route add -host $remipv4addr dev $iface

# IPv6
# My local network is fd14:6941:e887::/48
sudo /sbin/ifconfig $iface add fd14:6941:e887::1
sudo route add  -Ainet6  fd14:6941:e887::/48 dev $iface

# activate ip forwarding
sudo sh -c 'echo 1 > /proc/sys/net/ipv4/ip_forward'

# activate ARP proxy to "spoof" arp address
sudo sh -c 'echo 1 > /proc/sys/net/ipv4/conf/eth0/proxy_arp'
sudo sh -c 'echo 1 > /proc/sys/net/ipv4/conf/${iface}/proxy_arp'

# set "spoofed" arp address
sudo arp -Ds $remipv4addr eth0 pub
```

Je n'ai pas mesuré de façon rigoureuse les performances mais, au pifomètre, elles sont très inférieures à celles de Xen, qui exécute le code machine directement sur le processeur physique. Un module à l'origine non libre existait pour Qemu pour faire à peu près la même chose mais il n'est plus nécessaire depuis l'apparition de processeurs ayant des fonctions de virtualisation, et du système kvm <https://www.bortzmeyer.org/kvm.html> qui peut les utiliser.