Using kvm, or kqemu, to speed up qemu

Posted by Azerthoth on Fri 24 Aug 2007 at 09:33

Have you ever wanted to play with a new distro without having to burn and then reboot into a liveCD or do an install into a spare partition that you may or may not have? QEMU has been an option for awhile but it is slow. There are several options available to run up a virtual machine, i.e. a second operating system running inside and seperate from your already running operating system. Here we will be focusing on kqemu and kvm.

If your kernel is already 2.6.20 or higher you can run `kvm`, if not then you will have to use `kqemu`.

`kvm` is in sid/unstable so you will have to add that to your apt sources if you're using someting different.

Become root and edit your sources list:

```
nano /etc/apt/sources.list
```

add this:

```
deb http://ftp.debian.org/debian unstable
```

Now update by running :

```
apt-get update
aptitude install kvm
```

After `kvm` and associated packages are installed it would be best to edit your sources.list again and comment out the line where you added in unstable with a "#" character at the beginning of the line.

`kqemu`, which is technically a qemu accelerator instead of kernel level virtualization, like `kvm`, is available in Debian stable and therefor as easy to install as:

```
aptitude install kqemu
```

You should have noticed if it wasn't already installed that `qemu` was installed regardless of whether you went with `kvm` or `kqemu`. Thats because both packages use `qemu`. So no worries, we need the `qemu` tools anyway.

If you installed kqemu then we need to load that module when we boot. Become root again and then run:

```
 echo kqemu >> /etc/modules
```

For kvm you need to know something whether you have an Intel or an AMD processor. You'll need to load a processor-specific module to take advantage of the system.

If you have an Intel processor you will need to load `kvm-intel`, for AMD processors you'll need to load `kvm-amd`. The module can again be added to `/etc/modules`.

Now that we have the modules ready to load, we need to make them accessible to you as a user.

```
adduser $USER kvm
adduser $USER kqemu
```

(Only one of thse commands needs to be run; it depends which program you installed.)

Congratulations, right now you should have everything just about ready to go. The problem is, even if you modprobe the modules active right now they still wont be usable to you, they need to be loaded against the kernel when you boot, so bookmark this page and reboot your computer, I'll wait here while you do.

OK so now we need to create a virtual drive to install our test distro to. I'll explain the parts of it after. From here on out everything is applicable to QEMU, Kqemu, and KVM. It is done as a normal user, so no more need to be root.

```
qemu-img create debian.img -f qcow 5G
```

The first bit is self explanatory, create an image (virtual drive) named `debian.img`. The next bit "`-f qcow`" tells it to format it in an inflatable structure. The 5G means a maximum physical size of 5 gig real hard drive space. The nice thing about this format is, if your VM installed only takes up 2.5 gig, then the virtual drive only takes up 2.5 gig of space on your real hard drive.

I guess the next thing that you'll be wanting to do is actually spin this up and try it out. I should note that I have had problems on occasion with KVM locking up during the install process, if this should happen to you drop back and use `qemu` with the same command line arguments. If you are using `kqemu` instead of `kvm` use `qemu` in the command line instead of `kvm`, `kqemu` is a module called by `qemu` when it starts. I'll be using `kvm` for the command line, you use what works for you.

I'm not ready yet to tell you how to start. A few things first if you don't mind. I usually make a seperate directory for my VM's because there are usually one or two files other than just the .img file. Such as an overlay file or two, which I will cover shortly and a script that simplifies launching the VM after its made. There are tons of command line arguments that can be added to the basic ones I am using here to get you started, the script is a huge time saver.

There are two basic ways to start this off, either with an .iso image or a CD/DVD. Lets start with an .iso image, the debian net-inst image in this case. We will assume for the sake of argument that the .iso is in the same folder as the virtual drive that you created is, and that it is also our working directory.

```
kvm -cdrom net-inst.iso -hda debian.img -m 512 -boot d
```

First this calls `kvm` and tells it that the .iso image is actually a CD-drive, then the "`-hda debian.img`" is its hard drive. The "`-m 512`" tells it that its a computer with 512 meg of memory. Careful here because this is the amount of physical memory that its going to block out for itself. Rule of thumb is no more than 1/2 of your actual physical memory. The final bit is -boot d, it tells it to boot from the cdrom drive.

```
kvm -cdrom /dev/cdrom -hda debian.img -m 512 -boot d
```

The only difference here is that you're pointing it to your real cd drive here. If `/dev/cdrom` doesn't work for you then you can "`cat /etc/fstab`" and look there to see what you cdrom drive really is.

Now I mentioned that you might have problems doing an install using `kvm`, if this happens you have to explicitly tell `qemu` to not use the `kvm` module in the command line like this:

```
qemu -no-kvm -cdrom /dev/cdrom -hda debian.img -m 512 -boot d
```

Lets move on with the thought that you have installed your new virtual operating system. You're going to want to get in and play with it. The command for this is simply:

```
kvm -hda debian.img -m 512
```

With that you will be up and running in your new OS running inside your existing. However there

is another trick that is really handy that will let you do whatever you want without permanently breaking this new creation, overlay files. Which basically takes a snapshot of your virtual drive and then run it from the overlay instead of the virtual drive. Really handy if you want to have several versions accessible but only have to do the install process once. Say like having a version of stable a version of testing and a version of unstable all available from the same install. To do this its as simple as:

```
qemu-img create -b debian.img -f qcow stable.ovl
```

To boot into this you just change the command line a little bit and tell it to use the overlay file you just made.

```
kvm -hda stable.ovl -m 512
```

As you can see the easy way to run multiple versions off the same install would be to do a base install of stable and then make your overlay file for it. Next you would make an overlay file named something like testing.ovl and another for unstable.ovl all from the debian.img that we made to start.

Then simply fire each up in turn via the overlay files, edit your `/etc/sources.list` to what ever you want and update yourself into debian nirvana.

My thanks to Scott Ruecker over at lxer.com for asking the question that started the process for this how-to. As I said though there are tons of switches that will add functionality to your virtual machine. More than I can adequately explain as I haven't managed to figure them all out yet either. This how to was written with the whole intent to get someone armed and dangerous before kicking them out the door and isn't intended to be all inclusive.

For more information on this topic you can start at the qemu homepage.

Have fun.
~Az
