Xen to KVM
virtual machine conversion guide

Ettore Trevisiol - ettore@trevisiol.net

1 Abstract

This article show a fast view about the existing open source tehcnologies for the developmet of computer systems that grant quality of service. In the first two sections are presented the virtualization architectures that allow the drastic reduction of infrastructures costs and work optimization. In the end there is a guide about the convertion of a virtual machine from the old, but still good Xen, to the new KVM without reinstalling anything on a new VM[1].

2 Xen

Xen[2] is an hypervisor born from a research project of the University of Cambridge, it's opensource so it's free available. Ian Pratt, the chief professor of the project, found Xensource, the society that sell and support an enterprise version of Xen. The only Xen core is not sufficient to build a fully virtualized solution for enterprise level, in fact it's usually included on GNU/Linux distribution (like Red Hate, Suse or Debian) in which it's together with lot of tools to allow an efficient management, or it's included in full commercial solution, like Xensource or for example Virtual Iron.

The first public version, in 2003, was essentially a software for interface virtual machine with the host's hardware. Till 2.0 version, in 2005, this hypervisor didn't have a great success in the business world because it didn't support multiprocessor systems and the deploy was very complex. With the release of the 3.0, in 2006, it gained new important feature that have led Xen to become competitive in the enterprise sector, becoming one of the best competitor of the most successful VMWare.

Since the first version, Xen used only paravirtualization technologies for the VM creation. It only allowed, so, the execution of duly modified operating systems, like the most famous distribution of Linux, like NetBSD, OpenBSD, FreeBSD or OpenSolaris.

With the introduction from Intel and AMD of the virtualization libraries into the new processors, Xen project gain big interest on this, contributing at the integration into the hypervirsor the support to the new technology. Thereafter from the 3.0 version,

[1] VM = virtual machine.

[2] `http://www.xen.org`

the paravirtualization of Xen is flanked with the native virtualization, allowing to Xen the execution of every kind of operating system without any sort of modify.

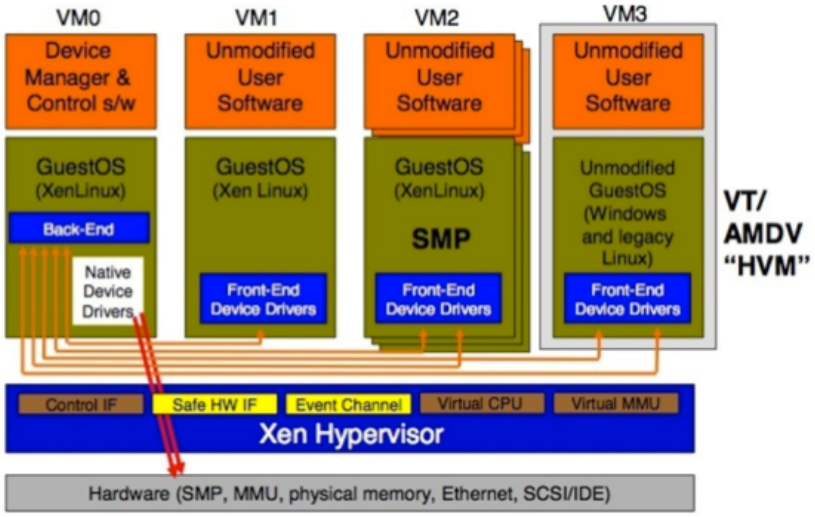

Figure 1: Xen architecture

The architecture, as show in figure 1, provides different levels with decrescent privileges. In Xen's terminology, every VM is called domain. Every domain manage applications of its own users, that makes the scheduling easier in the time granted from the hypervisor.

3 KVM

The Kernel-based virtual machine (KVM) project[3] is a last generation technology of opensource virtualization. The object of the project is to create a new and modern hypervisor that combine the experience earned from previous virtualization technologies (e.g. Xen) and new feature provided from the new hardware.

KVM, as show in figure 2, is implemented as a module, that insert into the kernel, that perfectly integrate in the operating system and makes it a native hypervisor. Two fondamental object guided the development of this project:

- As it's been born after the develop of hardware technologies that help the virtualization, like Intel-VT and AMD-V, KVM is been written around these and leverages their potential.
- In many ways an hypervisor is like and operating system, in fact, in addition to virtualize resource as CPU and memory, it need a scheduler, a device driver, a

[3] `http://www.linux-kvm.org`

memory manager, I/O manager, security policy,... As the Linux kernel already have these features, it's more efficient build KVM around these bases, rather than rebuild them from zero.

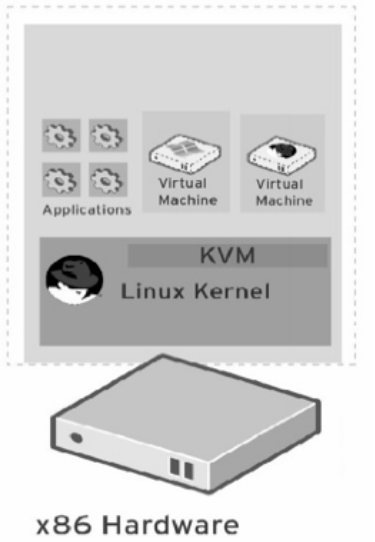

Figure 2: KVM architecture

In the KVM architecture, the virtual machines behave exactly like a normal Linux task, so they are managed as they are.

The resource emulation is managed from a modified version of QEMU, that provides the emulation of IDE or SCSI hard drives, network interfaces, BIOS, PCI bus and USB. As shown in figure 3, KVM provides the memory sharing functionality, and a kernel feature, called Kernel Same-page Merging (KSM), that manage, scanning the allocated memory, if two ore more VM load the same page. If this happen only one copy of these two pages is been shared on every VM. In this way different VM can be consolidated in a single host minimizing hardware costs and optimizing server utilization.

In addition to the normal offline migration, KVM support the live migration, that consists in a migration of a VM from an host to another without stopping the execution, and totally transparent to the user.

4 VM conversion

Despite there are some different projects, including Fedora with virt-v2v or Red Hat, that are working on a software that convert Xen images into KVM images, nothing has been released nowadays, so we found an our method to do this.

Main things to do are the Linux kernel installation and the GRUB configuration on the new KVM images. But this create some problems, because this two part of the

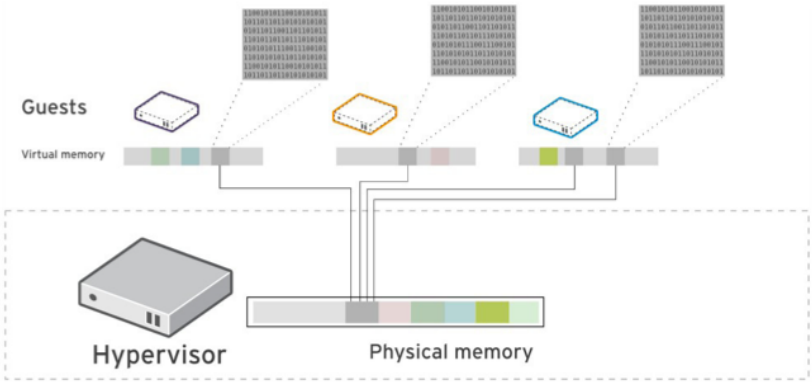

Figure 3: KVM memory management

image require a phisycal device onto work, so it's necessary to use a "system rescue" that see the KVM image that we're going to create as a device.

The way to follow is this:

1. Create a raw image (.img) of appropiate dimensions to host the Xen image that will be converted. Divide it in two parts, one dedicated to primary logic and one to the swap.

   ```
   dd if=/dev/zero of=kvm.img bs=1M count=6000
   parted kvm.img mklabel msdos
   parted kvm.img mkpart primary ext2 0 5000
   parted kvm.img mkpart primary linux-swap 5000 6000
   parted kvm.img set 1 boot on
   ```

2. Mount the just created image and copy the Xen image's files into.

   ```
   losetup /dev/loop0 kvm.img
   kpartx -a /dev/loop0 dd if=xen.img of=/dev/mapper/loop0p1 bs=1M
   fsck.ext3 -f /dev/mapper/loop0p1
   ```

3. Convert the KVM image from raw format with VBoxManage into .vdi, the unique format usable with VirtualBox[4] virtualization ambient.

   ```
   VBoxManage convertdd kvm.img kvm.vdi
   ```

4. Start the "system rescue" with VirtualBox that see as device the just created .vdi image, and mount it properly (figure 4). As best solution we have found, we used the live version of the distribution used from the Xen image. In our case, it was a live version of Ubuntu 8.04.

[4] `http://www.virtualbox.org`

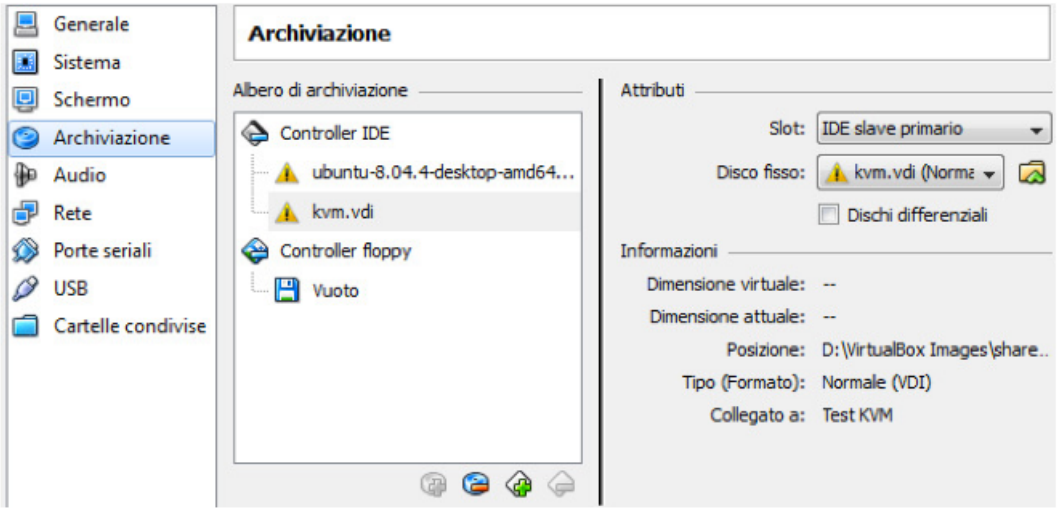

Figure 4: VirtualBox configuration

5. Do chroot into the mounted device and modify the network configuration files (backup the old ones), just to have internet access.

```
mount /dev/sdb1 /mnt
mount --bind /dev /mnt/dev
mount --bind /proc /mnt/proc
mount --bind /sys /mnt/sys
chroot /mnt
```

6. Download, install and configure the kernel and grub.

```
apt-get install kernel-generic
apt-get install grub-pc
update-grub
grub-install /dev/sda
grub-install --recheck /dev/sda
```

7. Modify the system boot to allow the correct visualization of the console after the boot.

   Change `xcv0` with `tty1` into the `/etc/event.d/tty1` file.

8. Finally, still with VirtualBox, reconvert the .vid image in .img, and restore the old network configuration.

In this way we will have a copy of the Xen image but perfectly compatible with KVM. This process is particularly useful to who need to migrate from a hypervisor to another without be forced to reinstall and reconfigure the operating system and applications, modify option or preferences and without any lose of files or content.