Computer Virtualization in Practice *  
[ life between virtual and physical ] †

A. Németh  
University of Applied Sciences, Oulu, Finland  
andras.nemeth@students.oamk.fi

ABSTRACT
This paper provides an overview of the currently available open source virtualization tools and practice examples of virtual to physical (V2P) and physical to virtual (P2V) transformations. There are several commercial tools available on the market for various business models for server virtualization, desktop virtualization, private cloud computing and virtualization management. The focus of this paper is not on dealing with any of those, but merely open source solutions for x86 processor architecture.

Keywords  
virtualization, V2P, P2V, virtual disks, virtualbox, OS migration, FLOSS

1. INTRODUCTION
The term “virtualization” was coined in the 1960s, to refer to a virtual machine (sometimes called pseudo machine). There are several approaches to platform virtualization in which, the most common virtualization scenarios are:

1. Running one or more applications that are not supported by the host OS
2. Evaluating an alternate operating system
3. Server virtualization
4. Duplicating specific environments
5. Creating a protected environment

This work presents a few example operations on a desktop scale personal computer for scenarios: 2 and 4.

*using free/libre and open source software  
†virtual and physical instances of OS installations

2. VIRTUAL MACHINES
Virtualization techniques create multiple isolated partitions –Virtual Machines (VM) or Virtual Environments (VEs)– on a single physical (or virtual) hardware. Virtual machines emulate some real or fictional hardware which allows to run an arbitrary guest operating system without modifications because guest OS is not aware that it is not running on real hardware. With this approach some CPU instructions may not be executed in user space thus requiring a virtual machines monitor (VMM) to analyze executed code and make it safe on-the-fly. This approach is used by VirtualBox and QEMU.

2.1 Paravirtualization
This technique also requires a VMM, but most of its work is performed in the modified guest OS software code. The paravirtualization technique requires them to be ported, i.e. they should know that they are running under the hypervisor. This approach is used by products such as Xen and User Mode Linux (UML). [4] Paravirtualized VM can securely execute virtual machines with near native performance.

2.2 Hypervisors
Hypervisor allows multiple OSes to run concurrently on a host computer. [15] This layer of software running directly on computer hardware replacing the operating system. The hypervisor is the interface for all hardware request such as CPU, I/O, and disk for the guest operating systems. [9] It is so named because it is conceptually one level higher than a supervisory program or operating system. An example place of hypervisor in the virtual platform can be seen in figure 1.

2.3 Towards Super Computing Power
Naturally the question arises: how to serve multiple VMs on a single HW without compromising performance? There will be a need for a higher performance machine with plenty of memory, disk space and preferably multiple CPUs. With the help of currently available SW environments, -such as XtreemOS [14]. - it is not necessarily needed to own a server class computer, a powerful virtual cluster can be built out of several desktop class PCs.

Parallel Virtual Machine (PVM) system is designed to enable concurrent or parallel computation by using a collection of computers. The heart of the system consists of message-passing libraries that is responsible for inter-process communication. This can be viewed as a single large distributed computer. [17] The virtual machine can be composed of
hosts of varying types of architectures and OSes. (PC, Mac, SUN, SPARC, HP9000, etc) [8]. Nodes are often placed in physically remote locations. The PVM project started in 1989 and gained widespread attention in between 1991 and 1995. The SW releases are rather infrequent today.

Message Passing Interface (MPI) is another widely used software system (set of libraries) for parallel computing that is similar to PVM. The MPI standard is based on the consensus of the MPI Forum, which has over 40 participating organizations. MPI-1 and MPI-2 was designed later when PVM was already existing [18] While PVM is supporting C, C++ and Fortran programming languages, MPI has programming interface implementation of wider range of languages such as: C, C++, Python, Perl, Java, etc. This makes it more available for hobbyists rather than just for scientists with supercomputer background.

A distributed operating system is also referred as single system image (SSI) cluster that appears to be one single system. The SSI designed for limited purposes only - job scheduling for instance- over a conventional operating system on each node. [16]

Parallel processing capable SW using MPI requires parallel design and compilation.

An example of parallel design using a master and multiple slaves may be written as follows:

```c
/* Algorithm for the master program */
initialize the array 'items'.

/* send data to the slaves */
for i = 0 to 3
    Send items[25*i] to items[25*(i+1)-1] to slave Pi
end for

/* collect the results from the slaves */
for i = 0 to 3
    Receive the result from slave Pi in result[i]
end for

/* calculate the final result */
sum = 0
for i = 0 to 3
    sum = sum + result[i]
end for

print sum
```

Many of the MPI and PVM based implementations are using this kind of algorithms for parallel processing in distributed VM clusters. [6] The source code linking and compilation have to be made by using MPI capable compiler, like mpicc, mpiCC, mpicxx, mpic++. [7]

3. VIRTUALIZATION IN PRACTICE

In the following chapters it is explained how different virtual machines can be operated on a single core HW without using cluster solutions. During the experiments, various operating systems were tested as guests on Ubuntu Linux 10.10.

3.1 32-bit vs. 64-bit operating systems

Virtual Machines may support 32 or 64-bit guest OSes regardless of running VM on 32 or 64-bit hosts. In order to run 64-bit operating system, the host processor must support hardware virtualization. The conventional x86 architecture was not designed to be virtualized, but since 2006 Intel and AMD processors have hardware virtualization capability [11]. These hardware features differ between Intel and AMD processors. Intel named its technology VT-x; AMD calls theirs AMD-V. On many systems, the hardware virtualization features first need to be enabled in the BIOS before VM (such as VirtualBox) can use them.

3.2 Data exchange

One may ask how data can be transferred between host and guest? This might be the first issue the new user faces with. Virtual machines allow for declaring certain host directories as "shared folders", which can then be accessed from within virtual machines. [12] With the VM installation, multiple virtual network interfaces are being installed. Each of them can be separately configured to operate in bridged or network address translation (NAT) modes. These modes allow
hosts in private network to transparently communicate with destinations on an external network or vice versa. [5]

Once the network is transparent and sshd installed on the host, remote file systems can be mounted through secured shell (SSH) simply by using sshfs unix command.

```
andras@vesta:~$ sshfs -h
usage: sshfs [user@]host:[dir] mountpoint [options]
```

### 3.3 Conversion from physical to virtual

An often discussed scenario is when a working and well configured physical infrastructure is transformed to virtual machine. In this process we need to take a copy of the affected partition(s) of the hard disk where the system is installed to.

During the experiments, the open source Clonezilla [2] v1.2.8 disk imaging and cloning program was used with Virtualbox [13] v1.3.6.

The following steps were performed to transform the source to virtual machine disk format (VMDK):

1. Prepare bootable USB media with Clonezilla software
2. Connect source and target SSH server in the same network
3. Boot source machine using the USB media
4. Clone the source partition and save to server machine
5. Create target VMDK disk and partition in VirtualBox
6. Create new virtual machine in VirtualBox with Clonezilla bootable image and newly created VMDK disk access
7. Boot the virtual machine and select `restore partition` option to VMDK disk
8. Release and remove Clonezilla image from the virtual machine

At this point the new P2V virtual image should be bootable that is the clone of the source machine.

### 3.4 Conversion of Microsoft OSes

Microsoft OSes store theirs system configuration database (registry) in a binary file on the disk and hardly accessible in case if boot failure. However, NTFS file systems can be mounted and the system registry can be modified off-line by using chntpw utility from Linux.

Copying and sharing Microsoft Windows® operating systems on multiple PCs are not allowed according to its software’s end user license agreement (EULA). [10] There are various methods used in different MS OSes and for different license agreement types for the validation. Windows upgrade licenses acquired through Microsoft’s volume licensing agreements will not require activation. Installations of Windows XP made using volume licensing media and volume license product keys (VLKs) will have no activation, hardware checking, or limitations on installation or imaging. The P2V transformation of these OSes are relatively simple.

The majority of customers acquire Windows with the purchase of a new computer, and most new computers pre-loaded with Windows. These installations are protected by basic input/output system (BIOS) locking mechanism with product activation that is called “System Locked Preinstallation” or SLP.

During log-on process the operating system compares the PC’s BIOS to the SLP information. If it matches, no activation is required. In order to avoid triggering the reactivation mechanism, the BIOS information should also be transferred separately to the virtual instance.

Question arises if one or two valid licenses are required in case of transferring a physical installation to virtual? Technically, all the same set of files running on the very same original hardware on the same CPU, HDD, RAM, NIC, etc. During the experiment, P2V conversion triggered undesired Windows activation.

### 3.5 Conversion from virtual to physical

The following steps were performed to transform the source to physical partition (V2P):

1. Prepare bootable USB media with Clonezilla software
2. Connect source and target SSH server in the same network
3. Add and Clonezilla bootable ISO to virtual machine
4. Boot virtual machine with Clonezilla bootable ISO and source VMDK disk mounted
5. Clone the source virtual partition and save to server machine
6. Create physical partition on target machine
7. Boot target physical machine using the USB media with Clonezilla and select `restore partition` option
8. Remove USB media and boot physical OS

One V2P transformation was executed successfully using Xubuntu linux 10.10 installation. The same image as it was originally virtualized from physical.

In the case of different from-to partition number, there is a need to rename the compressed archive on the server side otherwise the restoration process fails.

```
root@juno:/home/partimag/2011-05-10-05-img#
mv sda1.ext4-ptcl-img.gz aa sda8.ext4-ptcl-img.gz.aa
root@juno:/home/partimag/2011-05-10-05-img#
```
After performing V2P restore operation, the GRUB (GRand Unified Bootloader) [3] entries have to be updated. GRUB2 has its automated update script called: update-grub.

```
andras@vesta:~$ sudo update-grub
[sudo] password for andras:
Generating grub.cfg ...
Found linux image: /boot/vmlinuz-2.6.35-28-generic
Found initrd image: /boot/initrd.img-2.6.35-28-generic
Found memtest86+ image: /boot/memtest86+.bin
Found Microsoft Windows XP Professional on /dev/sda1
Found Ubuntu 10.10 (10.10) on /dev/sda8
done
andras@vesta:~$
```

In the example above, the partition was restored to /dev/sda8 and found automatically by update-grub.

3.6 Security and privacy
During the partition cloning process all the content of the filesystem is transferred through the local network including private files, folders and passwords. SSH secured mount points were used, so the transfer is considered to be safe and secure against threats.

In the experiment, the user’s private directory (/home) was included in the transferred partition. In order to respect user’s privacy, it is recommended to threat the user home differently and not cloned this way. It can be made by defining a separate (/home) partition for user files and exclude from the process.

4. RESULTS AND DISCUSSION
Both V2P and P2V transformations have been carried out by using Virtualbox through ssh secured network channel. After each conversion, there was a need to do minor OS level modifications.

The summary of the results can be seen in table 1 below. As it is stated, some of the Microsoft Windows transformations were not successful. “Blue Screen of Death ” BSOD is observed and could not be recovered without having Microsoft Installation disk. During another trial, the activation mechanism did not allow to log in without entering a new product identification number for the virtual instance.

Ubuntu Linux 10.10 operating system is cloned and restored without observing any problem.

The partition saving and restoration process may take several hours depending on the size of data, speed of the computers and the network load.

5. CONCLUSIONS
By decoupling the physical hardware from the operating system, virtualization provides more operational flexibility and increases the utilization rate of the underlying physical hardware. [1] The described technique allows quick pre-configured OS deployment on multiple target machines or can be utilized as an efficient archiving / backup mechanism.

6. REFERENCES
<table>
<thead>
<tr>
<th>Guest Operating system</th>
<th>Conversion</th>
<th>Result</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows XP Professional</td>
<td>P2V</td>
<td>Success</td>
<td>intelppm.sys disabled by modifying registry</td>
</tr>
<tr>
<td>Windows XP Professional laptop</td>
<td>P2V</td>
<td>Partial success</td>
<td>License activation triggered, could not log on</td>
</tr>
<tr>
<td>Windows 2000 Professional</td>
<td>P2V</td>
<td>Failed</td>
<td>unrecoverable error BSOD 0x7B</td>
</tr>
<tr>
<td>Xubuntu linux 10.10</td>
<td>P2V</td>
<td>Success</td>
<td>GRUB bootloader updated manually</td>
</tr>
<tr>
<td>Xubuntu linux 10.10</td>
<td>V2P</td>
<td>Success</td>
<td>GRUB bootloader updated manually</td>
</tr>
</tbody>
</table>

Table 1: Outcome of executed transformations during experiment