VMware Virtualization and Software Development

Mark Cloutier
Undergraduate Student, Applied Math and Computer Science

Keywords: Virtualization, VMware, Performance Testing

Abstract

Virtualization technology has recently matured to the point where it can be used dependably on a daily basis by anyone. Computer virtualization is the idea of running virtual computer(s) inside of a physical host computer. The basic idea behind the technology is not difficult to understand, but understanding how the technology is implemented is more complicated. In this paper, the basic ideas behind general virtualization are conveyed through VMware's technology. VMware is a company that is on the cutting edge of this technology and they offer many different virtualization solutions for free and for purchase. Through benchmark testing, it can be shown that current virtualization technology is on par with physical PC technology. This means that the performance decline incurred by using a VM (virtual machine) is very minimal, and it is getting smaller by the day.

Introduction

Computer virtualization technology has now matured to the point where a virtual personal computer (PC) can be just as effective as a real PC for many different reasons and applications. This study of virtualization technology is mainly from the viewpoint of a software developer. The main appeal of virtualization to a developer is the ability to test and develop software on various operating systems (OS) while only using one host computer. This allows for testing software on various platforms easily and transparently. Throughout this paper the following will be presented: an overview of what virtualization is, the common terms associated with virtualization, an overview of VMware applications and the technologies, the advantages to the software developer, performance testing baselines, and the results of performance testing. For the previously mentioned performance testing, the author will be applying his knowledge of the programming language Java on the popular Linux distribution Ubuntu and on Windows XP using both as virtual PCs and as standalone PCs.

Although much of the content in this paper applies to all of the major virtualization products such as Parallels and Microsoft Virtual PC, the author has chosen to focus on VMware, because they control a sizeable part of the virtualization market. According to a study of the market by InfoWorld in 2006, VMware is the clear market leader and will control over half of the market for the near future (Marshall, Survey Suggests Server Virtualization Catching On, 2006). VMware is considered to be the grandfather of the current resurgence of virtualization technologies since releasing the popular VMware Workstation in 1999 (VMware, About Us Home, 2007). VMware is also on the cutting edge of VM (virtual machine) speed because they have developed technology that executes code directly on the processor whenever possible and can dynamically rewrite instructions when direct code execution is not possible (Mittell & Hutchings, 2006). The final reason for choosing VMware is that they offer free virtualization products which allows for easier testing of their products. The main product the author did his testing on is called VMware server, which allows the creation and using of VMs. The software can be downloaded free at http://www.vmware.com/download/server/. They also offer other free

technologies such as VMware converter, which allows users to convert any physical PC into a virtual PC, and VMware player, which is a lightweight application that allows users to boot VMware images created in any other VMware application.

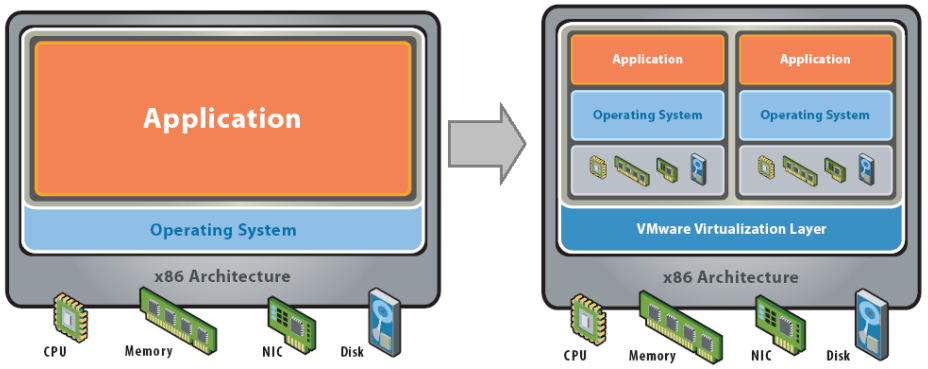

Figure 1. A physical PC (left) and a PC with two virtual PCs within a physical PC (Image Credit: "Virtualization Overview")

What is Virtualization?

The word virtualization generally means "an abstraction of resources" (Hammersley, 2007, p. 2). In the context of this paper, the word virtualization is used in the context of software virtualization. Software virtualization "is a process in which the actual physical hardware of a machine is decoupled, or abstracted … from the underlying operating system by means of software" (Hammersley, 2007, p. 2). The decoupled machine still thinks it is running in a physical environment, but there is actually software in between the decoupled virtual hardware and the physical hardware. Depending on the amount of resources available on the physical PC, there can be multiple virtual PCs within one physical PC. Figure 1 shows a normal physical PC (left) and another PC with two virtual PCs within a physical PC.

There are many advantages to integrating a virtual infrastructure within a physical computer system over using a purely physical infrastructure. According to VMware Inc., in the old system of using only a physical based system, there is "one physical operating system per machine, the software and hardware tightly coupled, running multiple applications on the same machine often creates conflict, resources are underutilized, and the infrastructure becomes inflexible and costly" (VMware, Virtualization Overview, 2006). Conversely, according to VMware Inc., the new virtualized infrastructure offers much more: "hardware-independence of operating system and applications, VMs can be provisioned to any system, and a user can manage OS and application as a single unit by encapsulating them into VMs" (VMware, Virtualization Overview, 2006).

Terms and Definitions

Here is a guide to common terms that will be used in this paper in regards to virtualization.

Abstraction Layer: The process of separating hardware functionality from the underlying hardware is called abstraction (Wolf & Halter, 2005, p. 2). The abstraction layer is responsible for mapping the guest OS's virtualized hardware to the host's physical hardware. This abstraction layer essentially turns the guest computer into a piece of software that can be run on a host PC as a process (Wolf & Halter, 2005, p. 2).This

allows a guest OS to be moved from computer to computer without effecting the new host PC or the moved guest OS.

C++: A general-purpose, high-level programming language with low-level facilities. Since the 1990s, C++ has been one of the most popular commercial programming languages. It was developed in 1983 by Bjarne Stroustrup as an enhancement to the C programming language (AT&T, 2008). In the future the author plans on performance testing C++ in a virtual setting, along with the Java testing the author recently completed.

Guest: This refers to the VM and operating system running inside VMware. Under VMware, each host system may have several guest systems. Guest operating system refers to only the operating system on the guest system, while guest PC refers to the virtual computer as a whole (Ward, 2002, p. 6). Both the guest OS and the host PC are referred to as the guest to simplify the terminology.

Hard Drive: A type of data storage that takes the form of digitally encoded data on rapidly rotating platters (Brain, 2000). Computers use hard drives as the main storage and recovery of data. This will be one of the main focuses of the performance testing, comparing the virtual guest hard drive speed with the physical hard drive speed.

Host: The host machine is the physical machine itself. It is the machine where a virtualization product, such as VMware Server, can be installed. The host operating system is what is installed on the physical host machine (Hammersley, 2007, p. 6). Both the host OS and the host PC are referred to as the host to simplify the terminology.

Java: "An object-oriented programming language developed by Sun Microsystems in the early 1990s. Java applications are compiled to byte code, which at runtime is either interpreted or compiled to native machine code for execution" (Patent Storm, 2000). This is the language in which the author is doing all of his current VMware performance testing.

Network Interface Card (NIC): The NIC is a piece of hardware that allows computers to communicate over a network. This will be one of the main focuses of the performance testing, comparing the virtual guest NIC with the physical NIC.

Physical Hardware: The physical hardware is the hardware on the physical PC that the host OS has direct access to. The main focusing of the performance testing is on the processor, hard drive, and NIC.

Processor: The component in a digital computer that interprets computer program instructions and processes data. This will be one of the main focuses of the performance testing, comparing the virtual guest processor speed with the physical processor speed.

Structured Query Language (SQL): A language designed for retrieving and modifying data in a computer database. SQL data retrieval over a network is what the author is using for testing NIC performance.

Ubuntu: A free and open source operating system for PCs. It is a widely used Linux distribution which is very easy to use and set up compared to other Linux distributions. The author chose Ubuntu as one of the operating systems to run performance testing on along with Windows XP Home Edition because of its ease of installation and popularity.

Virtual Hardware: This is the virtual counterpart of the physical hardware. The guest PC sees what it thinks is physical hardware, but is really virtual hardware translated from the abstraction layer. The guest has direct access to the virtual hardware. The main focus of the author's performance testing on the guest is with the processor, hard drive, and NIC.

VMware Server: Software that creates a virtualized environment between the computer platform and its operating system, so that the end user can operate software on an abstract machine.

Windows XP: An operating system designed to be versatile enough to be intuitive for the home and business user. Windows XP is the most popular operating system in the world, and is relatively easy to set up and use. Along with Ubuntu, the author will be using Windows XP for performance testing virtualization.

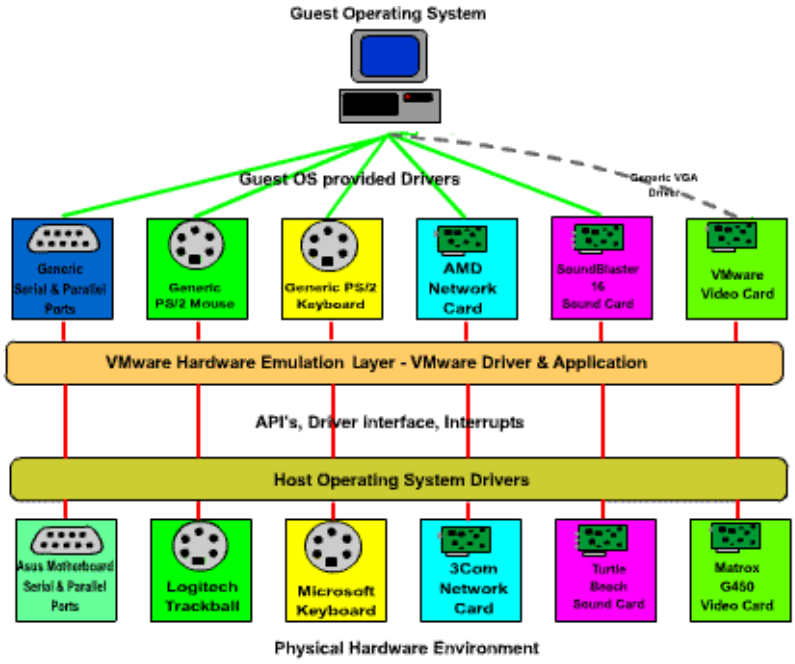

Figure 2. An overview of how the generic guest drivers communicate to the host's physical hardware through the abstraction layer. (Image Credit: Munro, 2001)

Overview of VMware and Virtual Hardware

The basic ideas behind VMware are common to other virtualization products in the market, such as Parallels or Microsoft Virtual PC. The virtualization abstraction layer provides generic drivers to the guest, while the same layer communicates with the host's hardware and passes the hardware input and output to the guest drivers. When a PC is running a VM, the host OS and the guest OS are thought of as two "worlds" (Munro, 2001). The virtual world can communicate directly to the processor or to the host's other hardware through the abstraction layer (Called the Emulation Layer in Figure 2). When a so called "world switch" occurs between the host and guest, a decline in performance can occur (Munro, 2001). "Any time a VM accesses an I/O (input/output) device, it gets a decline in performance from a world switch. For a keyboard or mouse, this is not a major problem because they are very short duration and low processing overhead events. However, for high throughput devices such as a network controller, the overhead can become a problem" (Munro, 2001). The people at VMware have developed very unique methods of trying to combat the overhead a NIC or a USB device can cause. They have developed a unique virtual machine monitor (VMM), which "understands the context in which I/O requests are made, and uses that knowledge to reduce world switches" (Munro, 2001). Figure 2 gives an overview of how the generic guest drivers communicate to the host's physical hardware through the abstraction layer.

The hardware components focused on during testing are the hard drive, the NIC, and the processor. The main reason for focusing on these components is because they are common to every computer and these are three of the four components that determine the performance of

any computer, physical or virtual. The fourth component is the memory (RAM). The author will not be focusing on the RAM outright because the RAM is being used and accessed as a part of the other tests, so the author did not feel that it needed to be singled out. The VM can easily be configured to have more or less RAM, so the author has set a base of 512 MB of RAM for all testing.

Setting up and Running a Virtual Machine in VMware

The process of setting up a VM in VMware is identical to setting up a physical computer after installing the VMware software. The first step is to go to http://www.vmware.com/download/server/, and download the free VMware Server. Once it is downloaded for either Linux or Windows, it should be installed by running the executable in Windows, or by extracting and running the installation file in Linux. The process will install all of the drivers and software to have VMware server run. Once the installation is completed, the user runs the VMware Server Console and connects to the local host.

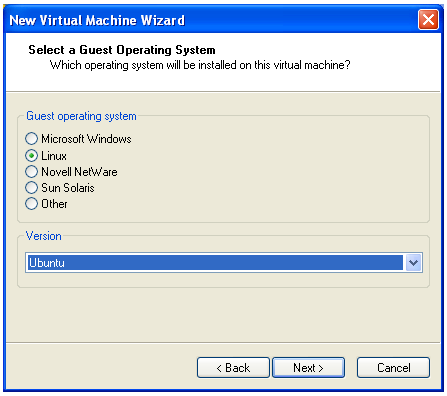

Figure 3. Selecting the operating system to install on the guest

The next step is to create a new VM using the "New Virtual Machine" wizard. This will bring up the wizard where the user will be prompted to select the operating system they want to install in the new VM (Figure 3). Selecting the operating system the user is going to install is important, because VMware will set up the optimal settings for that OS. The user is then given options about how they want the networking to be implemented and how big they want the virtual hard disk to be. One of the best features about the virtual hard disk is that the space can be allocated for virtual hard disk dynamically as more space is needed just by unchecking the "allocate all disk space now" box. Then all the user has to do is place the install disk for the operating system they chose earlier into the physical DVD/CD drive. Then the user clicks "Start this virtual machine" and the virtual PC will boot (Figure 4) just like a regular computer and the operating system can easily be installed on the virtual hard drive. Then once the system is installed, the virtual computer can be booted, suspended, and shut down just like a physical computer. Figure 5 shows an example of Ubuntu running in a VM on Windows XP.

Figure 4. The start of the VMware booting sequence

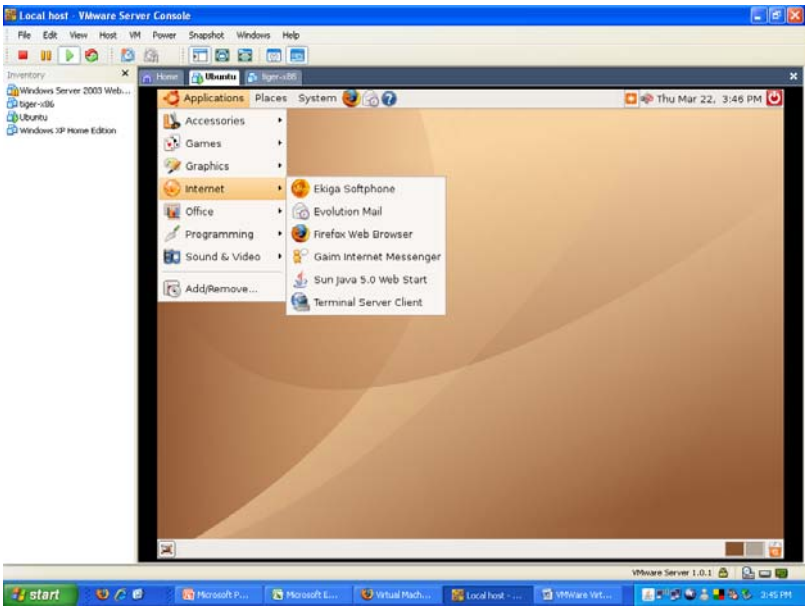

Figure 5. Ubuntu Running in a VM on Windows XP

Virtualization and the Software Developer

As stated in the beginning of this article, the author is using the VMware software from the viewpoint of a software developer. There are many features of VMware that make it a great solution for testing and running software in a real world environment. One of the goals of a software developer is to develop software to run on many different operating systems while giving the user the same experiences no matter the OS in which it is run. By using VMware, the developer can take advantage of running multiple operating systems, as any x86 based operating system can be installed in a VM. The family of x86 operating systems includes any version of Windows, many versions of Linux, Solaris, FreeBSD, and many more. The ease of having any operating system at the developer's disposal can save a lot of time and money. There can be many operating systems on one computer instead of many computers with one operating system. Another feature that is very beneficial to the software developer is the ability to take "snapshots" of the guest operating system. These snapshots save the exact state of everything in the VM. Then, at a later time, the user can revert back to the snapshot whenever they choose. This means that a software developer can do anything to a virtual computer and always go back to a previous state of their choosing. The last feature is the ability to "pause" a VM, so a developer can easily resume the guest operating system at any point in the future without having to shutdown the virtual PC.

A unique thing about the virtual computer is that what the guest sees as a hard drive is actually just a file on the host. A full computer is encapsulated into this one file on the host computer. This means that no partitioning needs to be done to set it up, which is different than if two operating systems were going to be installed on the same physical computer. The encapsulation allows the ability to copy VMs to other hosts. This can be done since the VM is hardware independent, only depending on the generic drivers the abstraction layer provides. This portability allows flexibility in testing different configurations between the guest and the host. The guest will have no idea it has been moved, and will boot exactly the same as it did on the

previous computer. Finally, encapsulation allows isolation, which means that the guest is isolated from other guest operating systems and the host operating system. This means that each guest has its own registry and file system. By having this virtual barrier between the guest and the host, if something such as a virus or other problem is present in the guest, it does not affect the host. The guest can then either be reverted or deleted, and the problem will be completely gone.

One good example of a possible usage of VMware here at UW-Stout is in the Data Structures class. Most of the course is taught in C++ and therefore the professor usually requires the use of Linux, since many Linux distributions have C++ compilers built into them. This also gives the Data Structures student exposure to the Linux OS. Because the laptops that the students use here are university issued, many students are reluctant to partition the hard drive and install another operating system. They are wary of causing some sort of damage to the current Windows XP system and all of their documents and files. By using a Linux distribution installed in a VM, students could use the built in C++ compiler, without having to worry about damaging their laptops in any way. The professor could actually have a preconfigured Linux VM and then distribute the VMware file with the virtual Linux installed on it to the students. Then this file could be booted through VMware just like a physical Linux installation. This would also ensure that each student would have a duplicate experience.

Performance Testing Baselines

The main reason for this performance testing is to show whether or not it is viable to run a VM as a testing/developing environment for the software developer. A software developer needs to test various configurations for their programs, so being able to use VMs seems to be ideal. Through performance testing the author plans to determine whether this is the case.

Before explaining the tests run on the virtual and host hardware, these are the processor speeds and RAM allotments of the host PC and the guest PC that were subjected to tests, as each computer sees: The host PC sees a 1.73 GHz processor and 2.00 GB of RAM and the guest PC sees a 1.73 GHZ processor and 512 MB of RAM. The difference in the RAM is because the guest actually gets allocated 512 MB (or as specified) of the 2.00 GB of the host. There are six total configurations on which tests were conducted. The author ran two controls using only the host; one test was run on Ubuntu and one was run on Windows XP. The other tests were run on various combinations of hosts and guests: Ubuntu (guest) on Ubuntu, Windows (guest) on Ubuntu, Windows (guest) on Windows, and Ubuntu (guest) on Windows. The author did all of the current testing in the Java programming language because of its interoperability between different operating systems. This allowed for a good comparison between the host control PCs and the virtual set ups.

Each test was run 100 times and the results were averaged to get more accurate measurements. All of the tests were also run on clean installs of the operating systems with only the default settings to get the most unaffected and comparable results.

Performance Testing

The first test was to determine the comparative processing speed of the configurations. To test the processing speed the author created a program that tabulates the time it takes to create 500,000 Java objects (Figure 6).

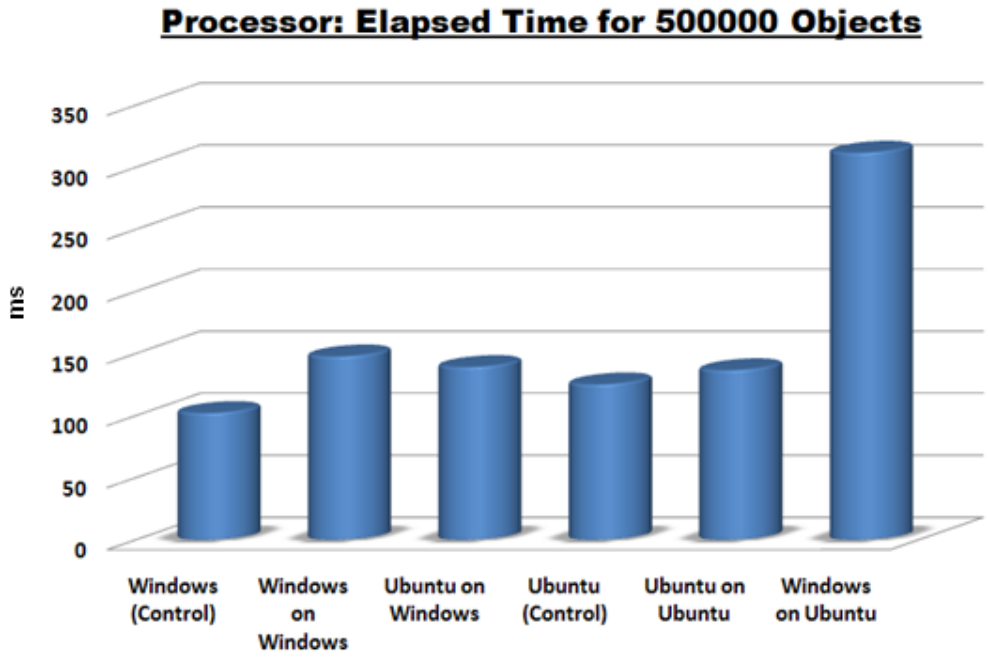

Figure 6. The speed of creating 500,000 Java objects on the various configurations

The results show a minor decrease in performance compared to the controls when the host operating system is the same as the guest operating system. The most surprising thing is the difference between running Ubuntu on Windows and running Windows on Ubuntu. Windows on Ubuntu took more than twice the time in creating the objects compared to the control, while Ubuntu on Windows was actually faster than Windows on Windows. Overall the results show that there are no major differences between the guest and the host, barring Windows on Ubuntu. The author suspects that the large differences here are a result of the way the abstraction layer executes the Windows code when Ubuntu is the host.

The next test was to determine the comparative speed of the NICs on the various configurations. To do this, the author created a program that executes an SQL query through the Java Database Connectivity (JDBC) API (Figure 7). The SQL query result set included 99,900 rows.

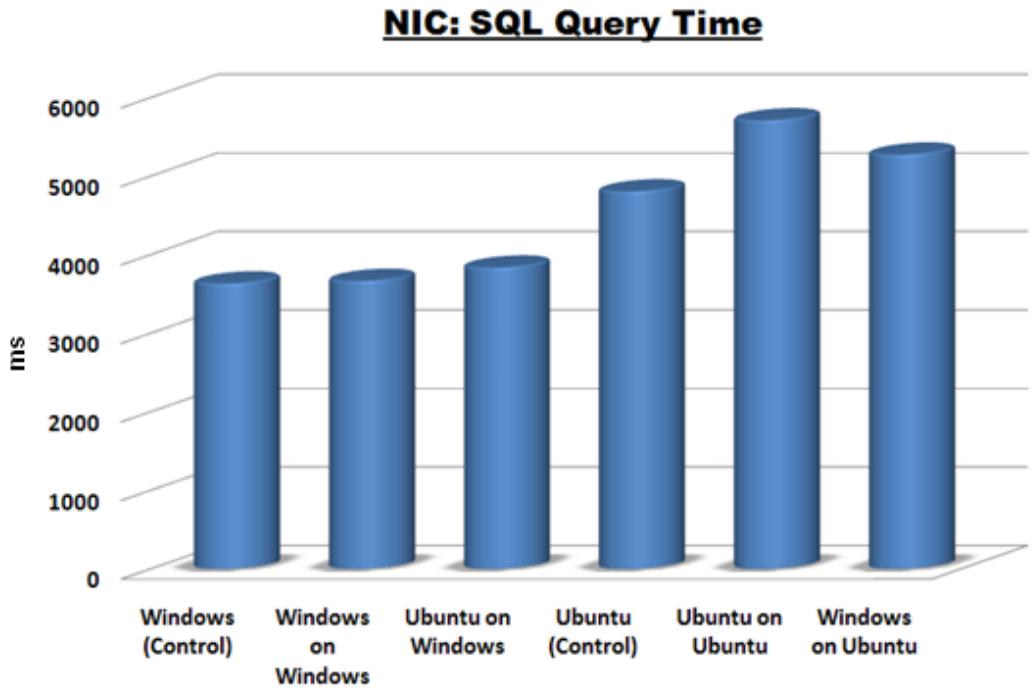

Figure 7. The speed of SQL queries on the various configurations

The results here show a distinct speed difference when using Ubuntu as the host system compared to Windows as the host. The author suspects that the reason is that drivers for things such as NICs are specifically written by the manufacturer for Windows, while the drivers in Linux distributions, such as Ubuntu, are written by the open source community. This may cause a little difference in the implementation of the drivers. There is almost no speed difference between the Windows control and the guests on Windows. On the Ubuntu side there is a little more variation, with the surprise that Ubuntu on Ubuntu is the slowest (although not exceedingly far behind the control). Again the author attributes this to the drivers for Ubuntu being written by the open source community instead of the NIC manufacturer.

The third test program written by the author determines the comparative input and the output speeds of the hard drives on the various configurations. To do this, the program outputs a large text file and then the program reads the file back in and the speeds of each operation are calculated (Figure 8). The key to remember is that the VM's so called hard drive is actually just a file on the host's hard drive.

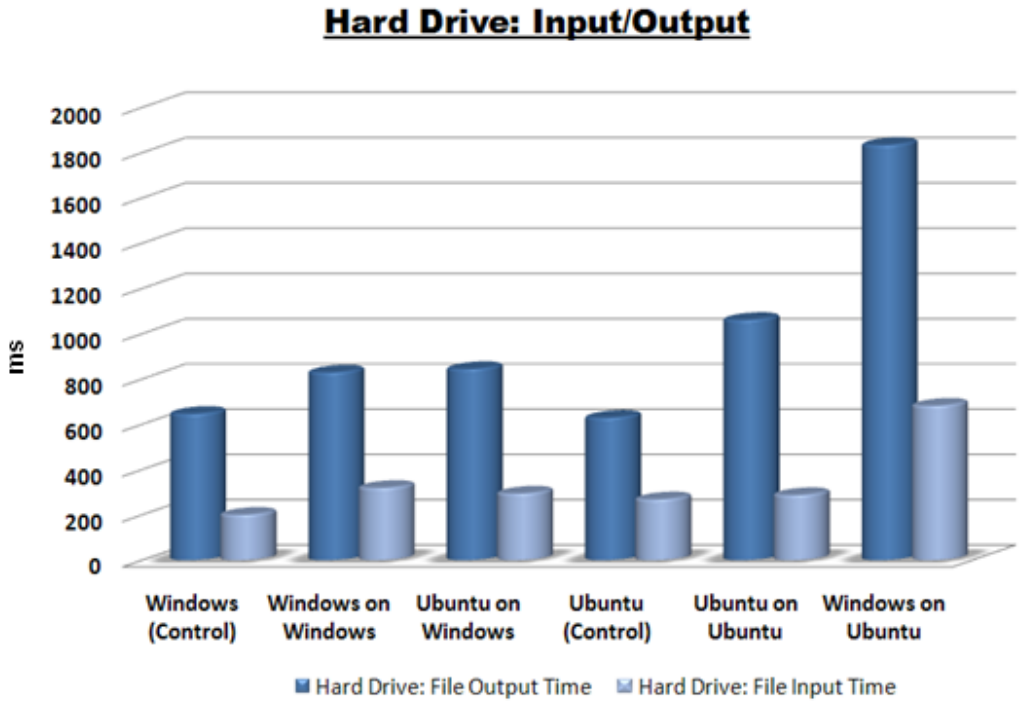

Figure 8. The speed of Hard Drive Input and Output on the Various Configurations

These results show little variation when Windows is the host and much more variation when Ubuntu is the host. With Windows and Ubuntu as the controls, their results are very similar along with the two guest configurations on Windows. The main surprise in this test is Windows on Ubuntu being almost twice as slow on both the input and the output side of things. The author is unsure what factor to attribute that to, although one idea may be code bloat. Code bloat is the fact that Windows continues to run programs developed for DOS, which requires keeping the old code. This may be causing some of the performance issues that have occurred.

The final test the author developed was designed to encompass all of the previous tests to give an overall perspective on performance. The program queries the SQL database and outputs the result set to a file. It then reads the file back in, and creates objects populated with what is in the file (Figure 9).

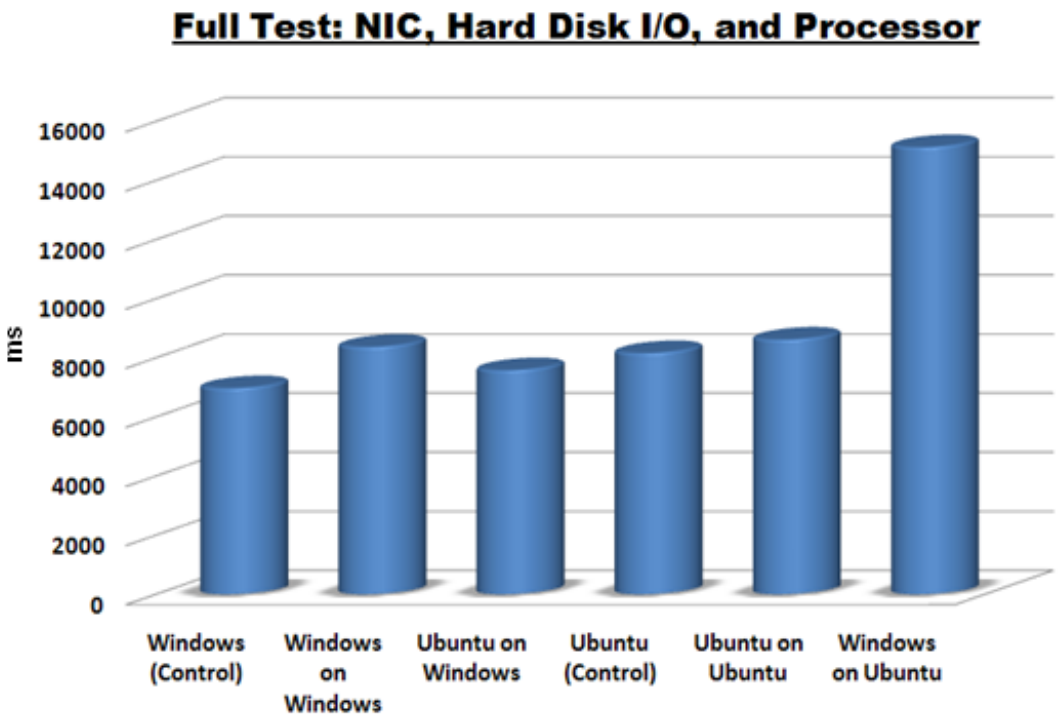

Figure 9. The speed of running the full test, encompassing the NIC speed, hard disk I/O speed, and the processor speed on the various configurations

The tests show that all of the results are very similar across the board, with the exception of Windows on Ubuntu. This tends to go along with all of the previous results. The author was slightly surprised that Windows on Windows did not perform better, but it is still close to the rest and slightly faster than Ubuntu on Ubuntu. This test shows fairly conclusively that, except for the Windows on Ubuntu case, a VM is just as useable as a physical PC.

Conclusion

The main concern about using VMs is the performance cost of using a VM over a physical PC. Through testing, the author is confident that for the most part (and especially when Windows is the host system), virtualization technology can be a great tool for a software developer. The main selling point is the ability to have any operating system at the developer's disposal without the overhead of installing and configuring the OS each time. That is not to say there is no decline in performance, however. There is a slight decline in performance in many cases, but overall the decrease in performance is a very minor one. The only case the author cannot currently recommend is running Windows on Ubuntu, although the author predicts that as both the VMware software and the Ubuntu OS mature, this decline in performance will become smaller. From the point of view of a software developer, the possibilities and applications of virtualization are endless.

References

AT&T. (2008). *Innovation - Technology Timeline - C++.* Retrieved February 13, 2008, from AT&T Corporte Web Site: http://www.corp.att.com/attlabs/reputation/timeline/83cplus.html

Brain, M. (2000, April 1). *How Hard Disks Work.* Retrieved February 13, 2008, from HowStuffWorks.com: http://computer.howstuffworks.com/hard-disk.htm

Hammersley, E. (2007). *Professional VMware Server.* Indianapolis, Indiana: Wiley Publishing, Inc.

Marshall, D. (2006, July 4). *Survey Suggests Server Virtualization Catching On.* Retrieved March 5, 2007, from http://weblog.infoworld.com/virtualization/archives/2006/07/survey_suggests.html

Mittell, A., & Hutchings, J. (2006, November 22). *Virtualization Using VMware Virtual Infrastructure.* Retrieved February 13, 2008, from Oxford University Computing Resources: http://www.oucs.ox.ac.uk/its3/seminar-notes/2006-11-22-VMWare.pdf

Munro, J. (2001, December 21). *Virtual Machines & VMware, Part I.* Retrieved March 21, 2007, from Extreme Tech: http://www.extremetech.com/article2/0,1697,10403,00.asp

Patent Storm. (2000, November 21). *Development system with methods for just-in-time compilation of programs.* Retrieved February 13, 2008, from Patent Storm: http://www.patentstorm.us/patents/6151703-description.html

VMware. (2007). *About Us Home.* Retrieved March 18, 2007, from http://www.vmware.com/company/home.html

VMware. (2006). *Virtualization Overview.* Retrieved March 19, 2007, from http://www.vmware.com/pdf/virtualization.pdf

Ward, B. (2002). *The Book of VMware: The Complete Guide to VMware Workstation.* Berkeley, CA: No Starch Press.

Wolf, C., & Halter, E. M. (2005). *Virtualization: From the Desktop to the Enterprise.* Berkeley, CA: Apress.