ISSN: 2321-7782 (Online)
Volume 3, Issue 1, January 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Review Paper on Resources Utilization of CPU Using Different Virtualization Applications

Kulwinder Kaur[1]
M-Tech Student,
Computer Science & Engineering
Guru Kashi University, Talwandi Sabo,
Punjab - India

Anand Kumar Mittal[2]
Assistant Professor,
Computer Science & Engineering
Guru Kashi University, Talwandi Sabo,
Punjab - India

Abstract: ***System resources utilization is depending on the work load of the application and pattern of some workloads may be suited for hosting on virtual platform. In this study, we are comparing how different workloads perform on CPU and resources utilization using some virtualization application with different operating system. Performance testing plays an important role in CPU resources utilization. This work gives revelation to the users for proper consumption of resources using virtualization and help to maintaining the performance of the System by utilizing the maximum resources.***

Keywords: Virtual box, PXE, USB, Operating System, Virtualization

I. INTRODUCTION

Within the past decade, the processing power of microprocessors has increased significantly. Especially processors based on IA-32 architecture have become popular since they are inexpensive and supported by various operating systems and applications. In addition to use in desktop computers, IA-32 architecture has also become widely used as server platform [1][5].

Whereas desktop computers can contain a large number of applications and configuration can be complex, server systems have been traditionally built by running a single application on one physical server. This approach has several benefits since configurations can be kept simple and in case of a hardware failure, only one application would be affected. The drawback is that certain applications do not require and are unable to benefit from the increased processing power. Typical examples of this phenomenon are applications where processing power needed to complete a request or transaction and the level of resource utilization have been the same for a long time. After a normal warranty period, the3 maintenance costs of hardware typically increase and at certain point, replacing existing hardware with a new one becomes more cost efficient. Transferring applications directly to new and more powerful hardware usually means that the new system becomes underutilized [5].

Within the past decade the number of actively used applications has increased. Entirely new computing areas such as the Internet have been the main reason for this growth. If the new functionality was impossible to achieve by modifying the existing system, a new system was required. Typically, increment in the number of applications also resulted in an increment in physical servers. Additional needs like a separate test environment has made the situation even worse. When the costs of e.g. maintenance and location facilities are taken account, the overall expenses quickly increase to an intolerable level. In the mainframe environment, a similar phenomenon has not occurred. Main reasons to this have been the price of the mainframe system and available partitioning techniques. Due to the high price, mainframes have been used only in situations where performance and availability of the IA-32 architecture has been insufficient [5].

There are several approaches available to use resources more efficiently. If several servers are using the same application to provide different content, content management and distribution can be centralized to a single server or a smaller number of servers. Also several separate and independent applications can be combined into a single server. Each solution has its benefits and drawbacks. Since changes to the existing infrastructure should be minimized, a solution where administrators and users

would not even know that environment has changed is preferred. Server virtualization can be seen as one solution to these issues [5].

Virtualization is the process of decoupling the hardware from the operating system on a physical machine. It turns what used to be considered purely hardware into software. Put simply, you can think of virtualization as essentially a computer within a computer, implemented in software. This is true all the way down to the emulation of certain types of devices, such as sound cards, CPUs, memory, and physical storage. An instance of an operating system running in a virtualized environment is known as a virtual machine.

II. LITERATURE SURVEY – A REVIEW

2.1 Virtualization

Virtualization is the process of decoupling the hardware from the operating system on a physical machine. It turns what used to be considered purely hardware into software. Put simply, you can think of virtualization as essentially a computer within a computer, implemented in software. This is true all the way down to the emulation of certain types of devices, such as sound cards, CPUs, memory, and physical storage. An instance of an operating system running in a virtualized environment is known as a virtual machine. The main idea of virtualization is to provide computing resources as pools. Depending on the needs, resources are then assigned to different applications either manually or dynamically from different pools. The scope of virtualization can vary from a single device to a large data centre and virtualization can be applied to different areas such as servers, networks, storage systems and applications. Data Centre in virtualized environment is presented in Figure 2.1.

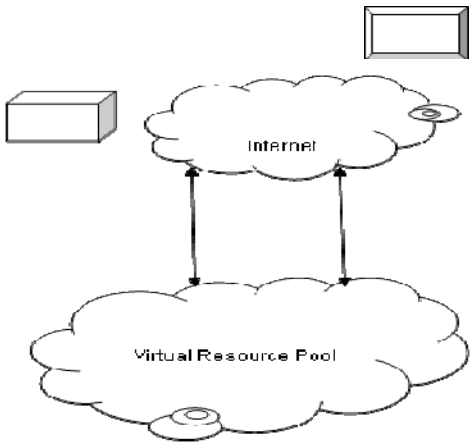

Figure No. 2.1 Data Centre in Virtualzed Environment

Virtualization technology allows multiple virtual machines, with heterogeneous operating systems to run side by side and in isolation on the same physical machine. By emulating a complete hardware system, from processor to network card, each virtual machine can share a common set of hardware unaware that this hardware may also be being used by another virtual machine at the same time. The operating system running in the virtual machine sees a consistent, normalized set of hardware regardless of the actual physical hardware components. There are some other types of Virtualization technology available. For example, computer memory Virtualization is software that allows a program to address a much larger amount of memory than is actually available. To accomplish this, we would generally swap units of address space back and forth as needed between a storage device and virtual memory. In computer storage management, Virtualization is the pooling of physical storage from multiple network storage devices into what appears to be a single storage device that is managed from a central console. In an environment using network Virtualization, the virtual machine implements virtual network adapters on a system with a host network adapter.

The focus of server virtualization is to create virtual machines or virtual environments by using normal server hardware and a virtual machine software. Virtual machine software enables sharing physical hardware among several instances called virtual

machines. Sharing is done by creating a special virtualization layer, which transforms physical hardware into virtual devices seen by virtual machines. The most visible change is the possibility to run different operating systems (OS) within the same hardware concurrently [3][6] Figures 2.2 and Figure 2.3 illustrate the difference between physical servers and virtual machines.

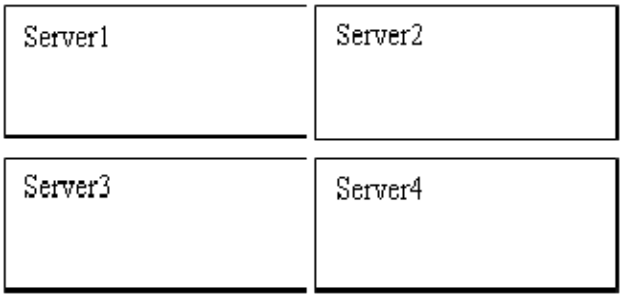

Figure No. 2.2 Physical Servers

Figure 2.2 presents four servers where each server has its own processor, memory, local disk and network connection. Figure 2.3 presents for Virtual Machines (VMs) and a Virtualization Layer.

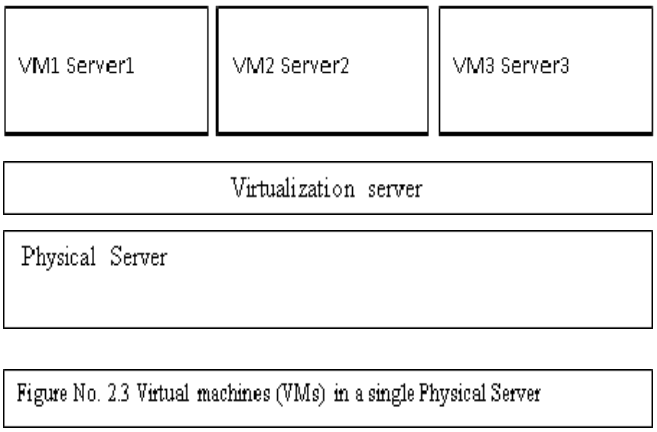

Figure No. 2.3 Virtual machines (VMs) in a single Physical Server

2.1.1 Virtualization compared to system partitioning

The special feature of mainframe systems is different partitioning schemes. By using these schemes hardware resources can be divided into several partitions. Currently there are two main schemes that are widely used: Logical Partitioning (LPAR) and Physical Partitioning (PPAR). Logical partitioning is similar compared to virtualization, since both of these techniques describe hardware resources as pools. The terms logical partitioning and virtualization are therefore often used in the same context. A practical difference is that within mainframes, the partitioning is typically done without sharing a single processor among multiple partitions and different partitions must use the same OS. The term physical partitioning is used when resources are physically divided in the hardware level. Although resource sharing using physical partitioning is not as flexible as in logical partitioning, the partitions are fully isolated and overhead does not exist. [2][5][4].

Mainframe architecture typically consists of separate CPU/memory cards, I/O cards and interconnection bus. Combinations of these cards are used to create a building block. An example of a server hardware that contains 4 building blocks is presented in Figure 2.4. The physical building blocks are the limiting factor when physical partition is created. Logical partition does not have similar limitations [2]. Figures 2.5 and Figure 2.6 present the difference between physical and logical partitioning. In Figure 2.5 there are two physical partitions: Physical partition 1 (3 building blocks) and Physical partition 2 (1 building block).

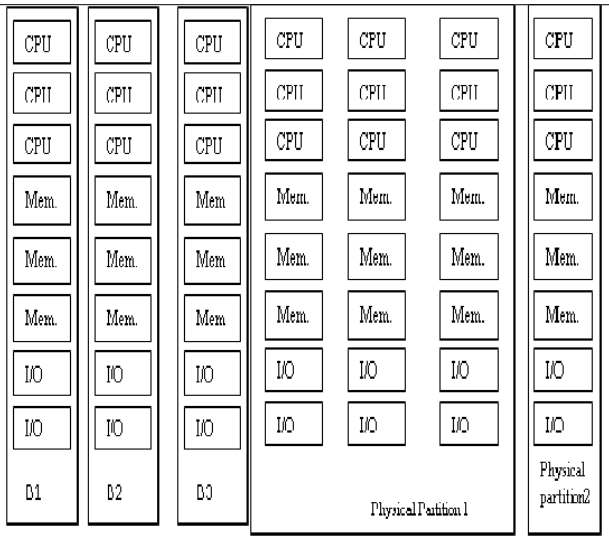

Figure 2.4 Three Building blocks. **Figure 2.5** Physical Partitioning

Logical partitioning with three separate partitions is presented in Figure 6. This partitioning example presents the typical scheme of production and testing environments within a single physical system: both environments are separated and the production environment has more resources than the testing environment. Due to the flexibility of logical partitioning, partitions can be configured to support CPU, memory and I/O sensitive applications.

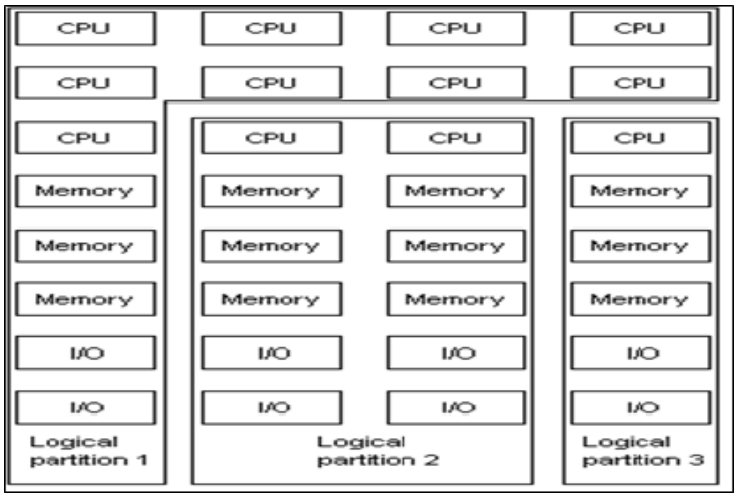

Figure 2.6 Logical partitioning.

2.1.2 Server virtualization compared to workload management

Server virtualization and workload management both point to the same target of using resources more efficiently and describing them as resource pools. The practical approach, however, is different. The main idea of the workload management is to provide resources to different tasks as efficiently as possible. A common solution to providing resources is to create a pool by using several physical servers and workload management software.

Instead of sharing or partitioning resources of a single physical server, the most suitable system from the resource pool is selected [8]. The common problem with workload management is its limited usage. All systems in the pool must be hardware compatible with each other and they must use the same OS. Besides hardware limitations, there are also restrictions in the software side: only those tasks can be used that the workload management software is capable of distributing. Also the whole concept of separate virtual machines and its benefits does not exist. In order to enable the distribution of tasks, a tight cooperation between the hardware, OS and workload management software is required. Usually this means that the whole

system must be obtained from a single vendor. Due to these restrictions, the workload management software is commonly used only in a single vendor UNIX environment [8].

Server virtualization compared to consolidation

The term consolidation is typically used to describe a process that aims at providing existing services more efficiently and thus save costs. One part of consolidation is server consolidation, which can be divided into three different categories:

- Location consolidation. The number of physical locations containing hardware is reduced.
- Physical consolidation. The number of physical hardware is reduced. [7][13].

Server virtualization is often mentioned as a consolidation scheme or in the same context as consolidation since virtualization provides similar benefits. When applications are running in underutilized servers, the efficiency can be increased by moving applications to virtual machines and shared hardware. Virtualization software creates a hardware standard, since every virtual machine runs in an identical environment. The benefits of physical consolidation can be achieved by a migration of several physical servers into virtual machines running on single server. Virtualization makes the practical part of consolidation process easier, but location consolidation cannot be done using it. Virtual machines can be transferred as files over network instead of transferring physical hardware [9].

Server virtualization can be seen in three major roles in consolidation:

» Multiple existing systems are combined into a single system, whose resources are used efficiently (physical consolidation).

» Instead of replacing aged hardware with new, old systems are migrated to virtual machines.

» Virtualization is used to provide a platform where creating a single system or an entire environment can be done in a short time span without purchasing new hardware.

Physical consolidation using server virtualization is presented in Figure 2.7. In this scenario, selected targets are assumed to be underutilized with an average load of 1-10%. Half of the servers are assumed to be production servers, the rest of them are either development or test servers. In the current setup, each application is installed to a separate server to make systems as simple as possible. Therefore, hardware for six separate servers is required.

Using virtualization to perform consolidation, separate physical servers are replaced by a single server and software that enables virtualization. Each server is then replaced by a virtual machine. After virtualization is performed, each virtual machine can be administered, backed up and used as if it was an independent machine. The result of the process is that the number of physical servers has reduced. Transferring systems from old hardware to virtual machine has certain benefits even though obtaining new server hardware would be required. Having underutilized servers is avoided since the new hardware is shared, systems are transferred to a more standardized environment and the number of physical servers is reduced. The workload of migration to virtual machines and of replacing aged hardware is usually the same, since both operations contain similar tasks (e.g. transferring disk partitions). A shift to the virtualized environment is the most suitable solution in situations where changes to OS and application are not needed and virtualization as a technology is acceptable.

During planning, development and testing phases of a new system, a number of different environments are needed. While creating these environments requires hardware, the obtained hardware does not necessarily satisfy the requirements of the final production environment. If a separate production and development environment is needed, both environments also

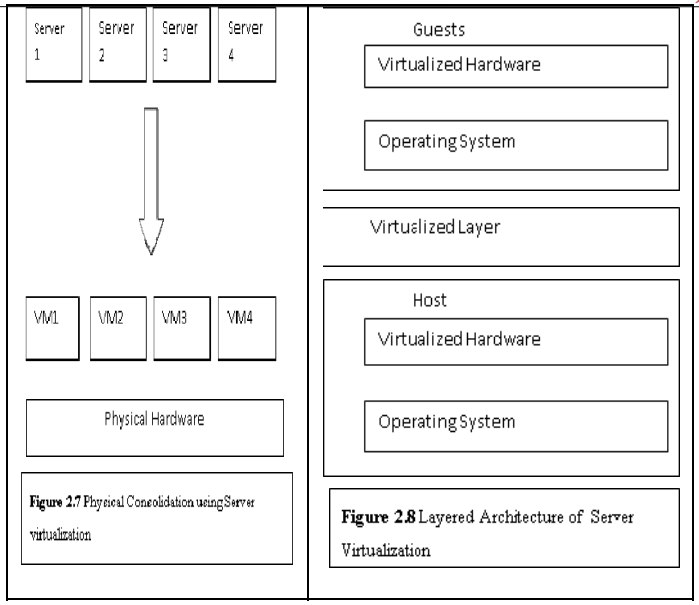

Figure 2.7 Physical Consolidation using Server virtualization

Figure 2.8 Layered Architecture of Server Virtualization

Require separate hardware. In the production environment, performance is one of the main criteria while development and testing can be done in more modest environments. With virtualization, the entire environment can be built quickly by using virtual machines instead of separate physical machines. The solution is also cost effective since several virtual machines can run on a single server.

Server virtualization can be divided into three separate layers:

» Host (Physical hardware and operating system)

» Virtualization layer

» Guests (Virtual machines)

Figure 8 presents these three layers as a hierarchical model. At the lowest level, the host contains all procedures that are close to the hardware. On the top, the guest is fully implemented in software.

The host contains the physical hardware that is being virtualized and OS that is used to allocate hardware resources. The difference between a normal server and a host is in the use: the host focuses only on providing a virtual layer while the normal server is typically used to provide one or more services (e.g. e-mail and database).

The virtualization layer is created with virtualization software. The purpose of the virtualization layer is to share the host's hardware resources among guests. Therefore, virtualization does not change the hardware architecture that can be done with emulation. Besides resource sharing, the virtualization layer can also be used to provide other features such as isolation. Although the host's OS can provide isolation between processes and secure memory management, the isolation between guests is typically handled by the virtualization layer [12].

Guests are either virtual machines or virtual environments that can only see resources that are provided to them by the virtualization layer. The term virtual machine is used if all physical components of the hardware are virtualized and the guests and the host do not have to possess the same OS. The term virtual environment is used when the same OS is used in both host and guest systems. [14][12].

Different virtualization approaches

Normally all instructions issued by the OS are executed directly on hardware. If the hardware is shared among multiple operating systems, a portion of instructions may be required to be executed using the software instead of the hardware. The difference between hardware and software execution portions can be used to determine the VMM type. The following classification is typically used to create a distinction between emulation, real machine and different VMM types:

» Real machine. Everything is executed directly on hardware.

» Virtual Machine Monitor (VMM). A large part of instructions is executed directly on hardware. The rest of the instructions are executed on software.

» Hybrid Virtual Machine (HVM). All privileged instructions are emulated using software.

» Complete Software Interpreter Machine (CSIM). Software is being used to emulate every processor instruction [16].

Hardware virtualization can be provided in two different ways: by a replication of the host ISA or by modifying the guest OS. The replication of the host ISA provides full virtual environment, including basic input/output system (BIOS) that is used in hardware detection. The advantage of the ISA replication is that the guest OS sees shared resources as if they were physical devices. Modifying the guest OS consists of changing hardware specific calls to normal system calls and recompiling the OS. The result is a modified OS that can run as a normal process without the need of direct hardware access. Although only the kernel part of OS would require modification, obtaining the source code of the kernel and creating modifications are not always possible [15][6][23].

There are currently two vendors that use the ISA replication in commercial server virtualization products. The Server, on the other hand, uses hardware resources directly and it contains a minimal OS to start the virtualization. Microsoft Virtual Server uses the same approach as VMware Server. [26][24].

Modifying the kernel of OS is possible when the source code of the kernel is available. User-mode Linux and Plex86 are both based on this approach and their underlying principle is the same: A modified Linux kernel is used as a user process on a system that runs Linux kernel. Both User-mode Linux and Plex86 are being distributed as patches to normal Linux kernel. Although Linux kernel is available to a number of different ISA architectures, kernel modifications are only available to the IA-32 architecture. [15][28].

2.2 Hardware virtualization

The focus of hardware virtualization is to enable virtualization in three areas: processor, memory and input/output (I/O) system. Hardware virtualization is required due to restrictions in the IA-32 architecture: it does not support virtualization in the hardware level. Processor, memory and I/O system of a single system are designed to be used only with one OS at a time. Removing this limitation is possible with special software that enables virtualization, thus sharing hardware safely. As a result, the physical hardware\ does not need changes and multiple OS can be used simultaneously[25]. The following chapter presents different areas of hardware virtualization in a more detailed level.

2.2.1 Processor

The processor provides resources for program execution in the form of instructions and registers. Instructions define single operations while registers are used to store code, data and state information. Besides executing commands and storing information, the processor also provides protection in the form of operating modes and levels. Most of the instructions and registers are used in normal program execution. The protection mechanisms in cooperation with OS ensure that the environment where programs are executed is safe. To enable e.g. changing the operation mode of a processor, registers are used to share information and instructions to perform the actual change. Due to the design of the IA-32 architecture, protection and sharing mechanisms work flawlessly only when a single OS is used at the same time[1].

The purpose of processor virtualization is to enable the use of execution resources and protection mechanisms of the processor without any limitations. A large portion of processor resources already support virtualization: their behaviour and the result of operation is always the same regardless of the number of OSs running simultaneously.

There are a total number of 20 processor instructions that cause problems. These instructions can be divided into sensitive register instructions (8 instructions) and protection system references (12 instructions). Sensitive register instructions contain instructions that either read registers, change register values or change memory locations. Examples of altered registers are clock register and interrupt register. Protection system references, on the other hand, contain instructions that refer to storage protection system, memory or address relocation system. A common feature of sensitive register instructions is that they modify or read values containing information about operating mode, status and the state of the processor. These instructions are normally used only by the OS under a privileged mode. In addition, due to the register sizes, only values for one processor can be stored at a time. Facing problems is inevitable since sensitive register instructions can read register values also in an unprivileged mode.

In a virtualized environment, certain registers are shared among several OSs. Without any additional protection, each virtual machine and the host OS itself would be capable of changing e.g. operating modes at the same time. Instead of executing these instructions directly on processor, virtualization software must emulate the execution and, for each virtual machine, create separate registers to prevent sharing [1][16].

Protection system references are related to instructions that either require certain protection level to enable execution or verification of the protection level during execution. In the IA- 32 architecture, the protection is implemented using four privilege levels. Accessing a higher privilege level can be only done using a tightly controlled and protected gate interface. The highest privilege level (level 0) is normally used only by the kernel of the OS. The lowest level (level 3) is mostly used by normal applications, in other words user processes or applications running in a non-privileged mode. In the virtualized environment the OS of the virtual machine expects to have the same privilege levels available as if it would be the only OS in the whole system. The virtual machine, on the other hand, is usually executed as a normal user process with a privilege level 3. Virtualization software must therefore provide an illusion to the virtual machine that it can use protection levels without exceptions. The virtual machine itself is running at the same time as an user-mode process in the host OS. If a process running in a non-privilege mode performs a privileged call, a special trap is always generated. The VMM detects these traps and manages them by using software emulation to execute instructions[1][16].

2.2.2 Physical memory

Physical memory of the computer is used in cooperation with the processor. The IA-32 architecture contains various memory management features including segmentation and paging. Although using more sophisticated memory management techniques disables the possibility to use direct memory addressing, additional features enable more flexible use of memory and reliability for program execution. When the program allocates the memory and memory management features are enabled, the memory is not addressed directly, but using one of the three different memory models. Depending on the used memory model, essential features such as address spaces and addressing model differ from another. After the operation mode of the processor and memory model are selected, the OS will know how the memory is handled[1].

The OS expects that a specific memory area is available for use and the area begins from a certain address. Firmware (BIOS) provides information to OS about the size of total memory available. Running multiple OSs at the same time causes errors since each of them is trying to use the same memory area. Memory management in the protected mode also uses several registers that are called Local Descriptor Table Register (LDTR), Interrupt Descriptor Table Register (IDTR) and Global Descriptor Table Register (GDTR). Using these registers is problematic since in a single physical processor, there are only one register of each type. Using multiple OS at the same time means that these registers and their content are shared between different operating systems[1][16].

Memory virtualization is done by address translation, since an additional level of translation is needed to provide memory mapping. The mapping is being done between addresses of VMM memory and virtual machine OS. While this solution creates

an additional layer to memory management, the process itself is quite simple and the implementation requires only little overhead [27][25][21].

2.2.3 Input/output System

The communication of I/O peripherals is typically done using I/O ports that are provided by the system hardware. A processor can transfer data from I/O ports in a same way as in using memory based on addresses and instructions. I/O ports can be accessed in two different ways: Using separate I/O address space or memory-mapped I/O. The difference between these two alternatives is that memory-mapped I/O can be used as if it would be normal memory (e.g. same instructions as in normal memory operations). Separate I/O address space contains 65536 individual 8-bit ports that reside in a special area separated from physical address space. Transmitting data by separate address space is done using IN/INS and OUT/OUTS instructions and special registers. Likewise memory management, additional protection mechanisms are available when processor is used in a protected mode. Mechanisms include privilege level and permission bit map for control access. In memory-mapped I/O, additional memory management features such as segmentation and paging also affect I/O ports [1].

In addition to I/O ports, Direct Memory Access (DMA) and interrupts are common parts of the system. DMA enables transferring data from peripherals directly into memory without using a processor. DMA transfer as an operation is fairly simple, since only the starting addresses, block length and operation type are needed. Interrupt is one of the two ways to stop execution of a currently running program on processor. Once the interrupt event is triggered, the processor halts the execution and switches over to handling the interrupt by using information in interrupt descriptor table (IDT) [1][18].

In a virtualized environment, the guest OS expects to have the same I/O ports, DMA and interrupts available as in a real machine. Since the number of IRQ and DMA values is limited, this has caused problems already before virtualization. The most common solution to this problem has been using devices that can share e.g. IRQ value with another device.

Virtualization uses the same technique by accepting I/O calls from the virtual machine, translating I/O call to suitable system call for underlying physical hardware and then performing the operation. Similarly, the results of the operation must be converted back to I/O calls and relayed to the virtual machine. Addition to sharing, instruction SIDT used in interrupts is a part of sensitive register instructions. SIDT instruction is used to obtain information from IDT register that contains values for address and the size of the register. SIDT instruction can be called from a normal program without an exception and therefore e.g. reading IDT register values of the wrong virtual machine is possible if additional protection is not available. By creating separate IDT register for each virtual machine, this issue can be avoided[1][20][18][16].

2.3 Process and thread management

Virtualization does not basically change the behaviour and usage of processes and threads. Virtual machines can be seen as normal processes in the host OS or sub processes under the VMM. The ISA replication typically hides single processes from virtual machines and the host OS only sees the VMM process. Kernel modification requires some additions to normal process creation. Since trap generation and monitoring is required for signalling, it is useful to create a new process as a child of the process that handles trapping. Since trap generation and monitoring is usually performed in the host OS, creating a process in virtual machine means that the new process is seen also in the host OS. This approach also simplifies context switching because processes of virtual machines can directly notify the process to the host OS that manages trap monitoring[15][12].

2.4 Memory management

In virtualization, no additional modification is needed apart from address translation. The VMM allocates memory from the host OS as a normal application and manages address translation. The guest OS sees the memory area that the VMM provides as if it would be physical memory. Figure 2.9 presents a layered structure of memory management. Since all memory handling of the virtual machines is managed by the VMM, the VMM can also see the content of the memory that the guest OS as well as

its applications use. Therefore, the security and isolation between virtual machines is provided by the VMM if ISA replication is used. If kernel modification is used, it must contain the required security and isolation features [17][12].

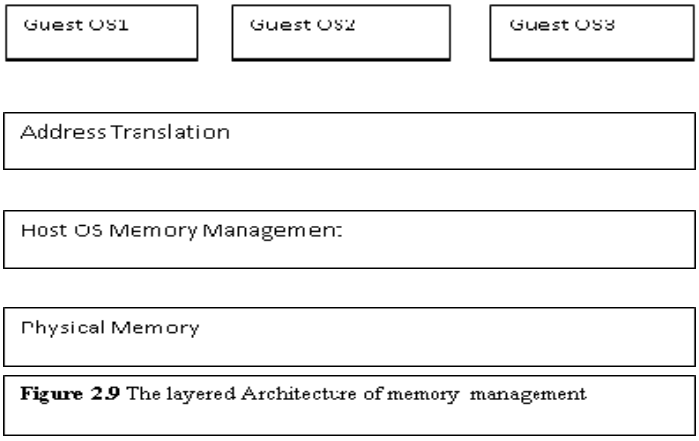

Figure 2.9 The layered Architecture of memory management

The OS typically includes memory optimizations to improve memory management and overall performance. Since all memory management of the virtual machines is performed in a single place, separate optimizations can be implemented to VMM.

2.5 Disk management

While disk virtualization requires similar translation compared to memory virtualization, it provides some additional flexibility. Usually the operating system is installed to a physical partition in the hard disk. Besides partitions, virtual machines can also be installed to raw devices or as files in the host OS. The host OS can therefore see the virtual machine disk as a normal file in its own file system and e.g. the replication of virtual machine can be done by creating a copy of this file.

Disk virtualization is performed by the VMM, which provides physical disk partition or a file from the host system as a virtual device to the guest. The guest OS then sees the virtual device as a normal device with I/O addresses and interrupts. If a guest OS has drivers for the device, it uses them to install and configure hardware. Every time the guest OS performs a reading or writing operation to disk, the VMM receives the I/O request from the\ virtual device. A request is then translated to matching I/O action for the hardware based on a virtual machine disk configuration (raw device, physical partition or file in a host file system). After the operation is performed in the host system, the results are sent back to the guest using a virtual device.

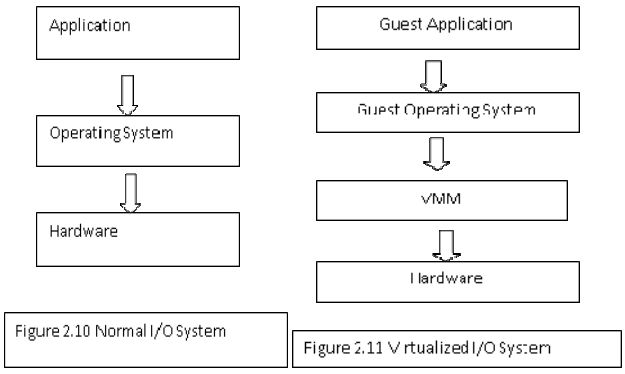

Figure 2.10 Normal I/O System

Figure 2.11 Virtualized I/O System

Figure 2.10 and Figure 2.11 illustrates the difference between a Normal I/O System and a virtualized I/O System [25].

2.6 Network management

In addition to processor, memory and disk, a network connection is also one of the main components of server hardware. As with disks, the typical requirement of a network card is low latency and high throughput. Since additional configurations such as load balancing and fail-over are often used in servers, the same functionality should be also available to virtual machines. These configurations might require that e.g. a single virtual machine can use several physical NICs of the host.

In the ISA replication, providing a network connection to the virtual machine is basically done the same way as with a disk apart from one exception. Creating a network can be accomplished without a physical network interface card (NIC). A connection to the existing network is not necessarily required either, since the VMM manages all requests. The VMM provides virtual I/O addresses and a virtual IRQ that are used with the virtual network adapter. The guest OS then sees the virtual network adapter as a normal device and uses device drivers to enable communication. Each time a virtual machine sends a packet to the network, all I/O operations are first performed with the VMM and then with the host OS and actual hardware. Receiving a packet is done in reverse order.

In kernel modification, networking can be arranged by creating a special device in the host that enables communication between the kernel of the host OS and the virtual machines. The modified kernel of guest OS then knows that the network connection is provided by a special device instead of the normal NIC. The host OS sees the created device as a normal network interface, which can be configured to send and receive packets to network using NIC. Communication between virtual machines only can be arranged e.g. by creating a separate instance that routes packets between virtual machines [15].

Communication directly to the network requires additional changes to the host system. Each NIC has a special hardware address called Media Access Control (MAC) that has been assigned to it during manufacturing process. MAC address is used in physical data transmission to identify the sender and receiver. For each manufacturer, separate address spaces have been assigned in order to avoid overlapping. Since a virtual machine does not have physical hardware, the host system's NIC is used to communicate with the network. Therefore the host system and VMM must have the capability to send and receive packets that contain a MAC address different from the actual physical hardware. In addition, VMM must provide unique MAC addresses to virtual machines [11].

2.7 Device and hardware access

In addition to disk and network, there are several other devices that are used during normal computer operation. Most common ones are display adapter and various input devices such as keyboard and mice. If the host OS can detect the hardware and provide a suitable driver, sharing devices among virtual machines is possible. The VMM must then either know how to convert and monitor device calls or allow direct execution ("pass through") using hardware. If the shared device uses e.g. privileged instructions, additional monitoring and conversion is needed [22].

2.8 Isolation and security

One of the most important issues in virtualization is the isolation between separate virtual machines and between a virtual machine and the host OS. While the host OS can provide efficient isolation between processes, hardware virtualization in the ISA replication can create certain problems. The same processor instructions, for example, are available to virtual machines as well as to the host OS, since the ISA will not change. Problems can occur if the virtual machine uses undocumented or undefined features of the hardware. If the software that provides virtualization does not detect instructions that allow reading register values without hardware protection, isolation and therefore security are not guaranteed. There is no good solution available to this problem, since new operating systems and modifications to existing ones are continuously introduced [12].

While the kernel modification enables the operating system to run as a user process that has limitations, additional security provided by the kernel mode with an assistance of the hardware is lost. Since the OS runs as a process, the kernel memory resides within the same address space of the process and can therefore be changed by a user space program. Security can be arranged by creating a write protection to the kernel memory when the user process is running and releasing it in the kernel mode [15].

Security issues must be considered due to the basic nature of virtualization. Since resources are shared among multiple instances, separating critical applications to separate physical machines is no longer possible. If the server uses e.g. confidential information during operation, virtualization should be carefully considered, since providing the same security to virtual machine

than physical server is very difficult. The following example illustrates this point: when virtual machines share NIC, it means that the same physical hardware is used to send and receive packets among virtual machines. Since it is possible for a virtual machine to receive packets destined to a different system, additional protection mechanisms are needed. Similar problems appear also in the disks of virtual machines regardless of the disk type. In addition, the host OS has access to all hardware so it can also see the disks of virtual machines and possibly read and modify their contents [16].

While security features such as encryption of network traffic and disk storage would provide necessary security level compared to a physical machine, these operations often consume a large amount of processing power and thus the overhead caused by virtualization will increase. Generally, kernel modification can be considered as a better solution than ISA replication. The main reason is that in the ISA replication, virtualization is done by additional drivers and there are no guarantees that the driver can handle all requests issued by the OS. Through kernel modification, also harmful instructions can be handled. They are disabled and converted to a more appropriate form without risking the host OS.

2.9 Optimizations for performance

Although the idea of server virtualization is to use all resources efficiently, optimizing the virtualization software can enable running a larger number of instances at the same time. The most common optimization methods are reducing the overhead caused by virtualization and sharing similar resources among virtual machines. The virtualization overhead is caused by operations that cannot be executed directly on hardware and by additional mappings that are used to provide the virtual machine a normal environment.

Using the full emulation of the IA-32 architecture causes significant overhead. This is because every hardware call must be verified separately and converted to a suitable system call format for the host OS. Even though kernel modification enables the reduction of hardware related operations, the tracking of system calls generate overhead. Additional overhead is further caused by the need to forward normal signals from the host OS back to the virtual machines. An alternative approach is to analyze the code and execute insecure instructions as emulation by the VMM. After a large portion of the code is analyzed and marked safe while the number of insecure instructions is low, overhead, too, is relatively low. To provide an optimal performance, the code of virtual machines should be executed as much as possible on a physical processor [15][16].

While virtualization creates the possibility to use resources more efficiently, it does not reduce the amount of required processing power, memory, network or storage capacity to perform a specific task. Depending on the number of virtual machines running concurrently and their similarity of tasks and OSs, resources can be reduced by sharing. This is especially true with disks and memory. If multiple virtual machines are running exactly the same OS, they are also likely to run the same services and use the same libraries. Typically an application uses some memory, whose content is not being changed during execution. Running five copies of the same application means that there are five possible memory areas, whose content will remain intact until the execution of application has ended. Memory usage can be reduced by creating only one copy of that memory and sharing it among applications [21].

Sharing memory and disk requires different approaches but both of them benefit the most of sharing common operating system components. For example, several virtual machines can be started by using the same system image. When differences occur among the execution of virtual machines (e.g. time stamps on log files), changes compared to the original system image can be created. Disk sharing can thus be eliminated by creating a separate copy of the system image and applying the changes that a single virtual machine had created.

Sharing memory between virtual machines cannot be done directly, since there are no guarantees that a certain memory area contains similar data in each of the virtual machines. Memory content must be identified e.g. by creating hash values. Similarities can be then found by comparing these values of the memory areas between separate virtual machines. While in disk sharing changes are tracked using a separate file, a change in memory requires the creation of a separate copy of the shared area

and applying changes. After changes are applied, the hash value can be recalculated and sharing can occur if areas with similar content are found. In addition to sharing memory between the virtual machines, passing information between applications, drivers, host OS and guest OS using shared memory is generally efficient [19][17].

While resource management does not directly provide better performance or decrease overhead, it has become one of the main features especially in commercial products. Traditionally, different operating systems contain resource management that enables fluent execution of the user program and efficient use of hardware resources. A modern OS typically contains a scheduler that shares the execution time of the processor among different processes. Scheduler also changes the order of processes through prioritization. By changing the rank order of processes, the amount of execution time can be adjusted.

Even though changing priorities and scheduling processes requires additional resources, the benefits usually exceed the overhead. Virtual machine resource management is a similar concept where virtual machines can have different priorities. Resources that can be prioritized typically contain those parts of the server hardware that contribute to performance most: memory, processor, disk and network. Prioritizing is easy to implement in the VMM since it handles all hardware related calls. Most commonly used sharing schemes are proportional shares, percentages and explicit values. When virtual machine resources are underutilized, resource sharing does not basically need to interact at all. However, if e.g. managing overall peak load requires more processing power that the hardware can provide, resource management can be used to ensure that the most important virtual machines obtain adequate resources [11].

Besides saving memory and disk resources, reducing the usage of privileged instructions is the main target in optimization. Other instructions can be run natively with small overhead. Processes that require or produce heavy I/O load create bottlenecks in performance due to context switching. Instead of the I/O performance being the bottleneck, the processor power becomes inadequate to manage the overhead [16]. Practical optimization approaches are fairly simple. For example, it is possible to gather multiple packets of network traffic together and perform a send or receive process during one switch. While optimizations can reduce the overhead of virtualization, sharing resources always lead to isolation and security issues. Providing the same performance by virtualization compared to a native environment is very difficult [22][11].

2.10 Features of Virtualization:

Running multiple operating systems simultaneously- Virtual Box allow us to run more than one operating system at a time. This way, you can run software written for one operating system on another without having to reboot to use it. Since we can configure what kinds of “virtual” hardware should be presented to each such operating system, we can install an old operating system such as DOS or OS/2 even if our real computer’s hardware is no longer supported by that operating system.

Easier software installations- Software vendors can use virtual machines to ship entire software configurations. For example, installing a complete mail server solution on a real machine can be a tedious task. With Virtual Box, such a complex setup can be packed into a virtual machine. Installing and running a mail server becomes as easy as importing such an appliance into Virtual Box.

Testing and disaster recovery- Once installed, a virtual machine and its virtual hard disks can be considered a “container” that can be arbitrarily frozen, woken up, copied, backed up, and transported between hosts. On top of that, with the use of another Virtual Box feature called “snapshots”, one can save a particular state of a virtual machine and revert back to that state, if necessary. This way, one can freely experiment with a computing environment. If something goes wrong, one can easily switch back to a previous snapshot and avoid the need of frequent backups and restores. Any number of snapshots can be created, allowing you to travel back and forward in virtual machine time. You can delete snapshots while a VM is running to reclaim disk space.

Infrastructure consolidation- Virtualization can significantly reduce hardware and electricity costs. Most of the time, computers today only use a fraction of their potential power and run with low average system loads. A lot of hardware resources

as well as electricity is thereby wasted. So, instead of running many such physical computers that are only partially used, one can pack.

III. Technique Used

Virtual Box is a cross-platform virtualization application. It installs on our existing Intel or AMD-based computers, whether they are running Windows, Mac, Linux or Solaris operating systems. Secondly, it extends the capabilities of your existing computer so that it can run multiple operating systems (inside multiple virtual machines) at the same time. It can run everywhere from small embedded systems or desktop class machines all the way up to datacenter deployments and even Cloud environments.

IV. Conclusion And Future Work

Though virtualization we have improved resources and memory utilization and also optimised the performances of the system with different security level.

It has been proved that, a single physical system cannot utilize their 100% resources and memory, so by using virtualization we have improved the resources and memory utilization but also each virtual operating system can also use as the server.

As the technology increases, the desktop computer which are uses at home or small office purpose are very advance in configuration. So by using virtualization those desktop systems are not only works as server but also working as the High performance gaining system in which the many number of operating systems are working in parallel which required different hardware configuration and that different hardware configuration can we provided virtually.

References

1. Intel Corporation. IA-32 Intel Architecture Software Developer's Manual.Volume 1: Basic Architecture. [pdf-document]. 2003. [retrieved July 10,2003].
2. International Business Machines. White paper: Partitioning for the IBM@server pSeries 690 System. [pdf-document]. October, 2001. [retrieved June 9, 2003].
3. Hewlett-Packard Company. HP Virtualization. [pdf-document]. 2003.[retrievedJune6,2003].
4. Sun Microsystems Inc. White paper: Sun Fire [tm] 12K and 15K Servers High Availability Whitepaper in Data Center. [pdf-document]. February,2003. [retrieved June12,2003.]
5. McIsaac, Kevin. Intel Server Consolidation: Part 1 – Virtualization. MetaGroup. February 20, 2003.
6. Smith, J.E. An Overview of Virtual Machine Architectures. [pdfdocument].October27,20001.[retrievedJune6,2003]..
7. Cognizant Technology Solutions. SOS Solutions. Server Consolidation.[pdfdocument].2002.[retrievedJune13,2003]..
8. Day, Brad. Server Workload Consolidation, Part 2 – Evaluating UnixWorkload Management. Giga Information Group, Inc. [pdf-document].August 21, 2002. [retrievedJune26,2003]..
9. Roach, Steven. Making the move to Windows Server 2003: Migration,Integration, & Consolidation (Part 2). Microsoft USA Presentations.[Powerpoint presentation]. May8,2003.[retrievedJune21,2003].
10. McIsaac, Kevin. Intel Server Virtualization: Part 2 – Picking the Low-Hanging Fruit. Meta Group. February 20, 2003
11. VMware, Inc. VMware ESX Server. User's Manual. Version 1.5. 2002.
12. VMware, Inc. Technical White Paper. [pdf-document]. February, 1999. [retrievedJune24,2003].
13. International Business Machines, Server Consolidation with the IBM@server xSeries 440 and VMware ESX Server. November 2002. ISBN0738427330
14. SWsoft. Virtual Environments. [www-document]. 2003. [retrieved June 24,2003].
15. Dike, Jeff. A user-mode port of the Linux kernel. [www-document]. 2000.[retrievedJune24,2003]. Html
16. Robin, John Scott. Irvine, Cynthia. Analysis of the Intel Pentium's Abilityto Support a Secure Virtual Machine Monitor. Proceedings of the 9thUSERNIX Security Symposium.pdfdocument].August2000.[retrievedJune30,2003].
17. Dike, Jeff. User-mode Linux. [www-document]. 2001. [retrieved June 25,2003].
18. Kozierok, Charles M. The PC Guide – System Resouces. [www-document].April 17,2001[retrievedJuly15,2003].
19. Bugnion, Edouard. Devine, Scott. Rosenblum, Mendel. Disco: RunningCommodity Operating Systems on Scalable Multiprocessors. Proceedingsof the 16th Symposium on Operating Systems Principles (SOPS). [pdfdocument].October1997.retrievedJuly14,2003].
20. Intel Corporation. IA-32 Intel Architecture Software Developer's Manual.Volume 2: Instruction Set Preference. [pdf-document]. 2003. [retrieved July10, 2003].
21. Waldspurger, Carl A. Memory Resource Management in VMware ESXServer. Proceedings of the Fifth Symposium on Operating Systems Designand Implementation (OSDI'02). [pdf-document]. December 2002.[retrieved June 26, 2003].

22. Sugerman, Jeremy. Venkitachalam, Ganesh. Lim, Beng-Hong. Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor.Proceedings of the 2001 USERNIX Annual Technical Conference. [pdfdocument].June 2001. [retrievedJune30,2003].

23. Whitaker, Andrew. Shaw, Marianne. Gibble, Steven D. Denali: LightweightVirtual Machines for Distributed and Networked Applications. Universityof Washington Technical Report. [pdf-document]. February 2, 2001.[retrievedJune25,2003].

24. VMware, Inc. VMware Server Products: Enterprise-Class Virtual MachineSoftware for Intel Servers. 2003. [retrieved June 25, 2003].

25. Smith, J.E. ECE 902: Special Topics in Computers. Virtual MachineArchitectures, Implementations, and Applications. Operating System VMs.[pdf-document]. October 1, 2001. [retrieved June 24, 2003].

26. Connectix Corporation. Virtual Server: Product Overview. [wwwdocument].2003. [retrieved August 4, 2003].

27. Rosenblum, Mendel. Operating Systems and Systems Programming. VirtualMachine Monitors. [pdf-document]. 2003. [retrieved June 26, 2003].

28. Lawton, Kevin. The Plex86 x86 Virtual Machine Project. [www-document].2003. [retrieved June 24, 2003]

29. Kato, Ken. Virtual Disk Driver Version 2. [www-document]. July 30, 2003.[retrievedSeptember15,2003].

30. I. Krsul, A. Ganguly, J. Zhang, J. A. B. Fortes, and R. J. Figueiredo. Vmplants: Providing and managing virtual machine execution environments for grid computing. In SC '04: Proceedings of the2004 ACM/IEEE conference on Supercomputing, Washington, DC, USA, 2004. IEEE Computer Society.

31. D. Reed, I. Pratt, P. Menage, S. Early, and N. Stratford. Xenoservers: Accountable execution of untrusted programs. In In Workshop on Hot Topics in Operating Systems, pages 136–141, 1999.

