International Journal of Cloud Computing and Services Science (IJ-CLOSER)
Vol.3, No.4, August 2014, pp. 218~232
ISSN: 2089-3337

Server Virtulization and Network Virtulization in Cloud Computing

Vedula Venkateswara Rao*, Mandapati Venkateswara Rao**
* Departement of Computer Science Engineering, Sri Vasavi Engineering College, Tadepalligudem, India
** Departement of Information Technology, Gitam Institute of Technology, Gitam University, Visakhapatnam, India

Article Info

Article history:

Received Jun 20th, 2014
Revised Jul 22nd, 2014
Accepted Aug 19th, 2014

Keywords:

Virtualization,
Cloud Computing,
Virtual Machine,
Data Center,
Performance,
Scheduling,
Hyper visor,
Green Computing.

ABSTRACT

Data Centers of today are rapidly moving towards the use of server virtualization as a preferred way of sharing pool of server hardware resources between multiple gust domains that hosts different applications. Due to trends like Cloud Computing and Green IT, virtualization technologies are gaining increasing importance. They promise energy and cost savings by sharing physical resources, thus making resource usage more efficient. However, resource sharing and other factors have direct effects on system performance, which are not yet well-understood. Hence, performance prediction and performance management of services deployed in virtualized environments like public and private Clouds is a challenging task. One benefit of virtual machines in these environments is the ability to multiplex several operating systems on the hardware based on dynamically changing system characteristics. However such multiplexing can be done while observing per VM performance guarantees.

Corresponding Author:

Vedula Venkateswara Rao,
Research Scholar (Dept of CSE,Gitam Institute Of Technology, Gitam Univewrsity,Visakhapatnam,India)
Departement of Computer ScienceEngineering, Sri Vasavi Engineering College,
Tadepalligudem-534101, India.
Email: venkatvedula2012@gmail.com

1. INTRODUCTION

Cloud computing is one of the most explosively expanding technologies in the computing industry today. A Cloud computing implementation typically enables users to migrate their data and computation to a remote location with some varying impact on system performance [2]. This provides a number of benefits which could not otherwise be achieved. Such benefits include: (i) Scalability - Clouds are designed to deliver as much computing power as any user needs. While in practice the underlying infrastructure is not infinite, the cloud resources are projected to ease the developer's dependence on any specific hardware [2]. (ii) Quality of Service (QoS) - Unlike standard data centers and advanced computing resources, a well-designed Cloud can project a much higher QoS than traditionally possible. This is due to the lack of dependence on specific hardware, so any physical machine failures can be mitigated without the prerequisite user awareness [5]. (iii) Customization - Within a Cloud, the user can utilize customized tools and services to meet their needs. This can be to utilize the latest library, toolkit, or to support legacy code within new infrastructure.Cost Effectiveness - Users finds only the hardware required for each project. This reduces the risk for institutions potentially want build a scalable system, thus providing greater flexibility, since the user is only paying for needed infrastructure while maintaining the option to increase services as needed in the future.

Simplified Access Interfaces - Whether using a specific application, a set of tools or Web services, Clouds provide access to a potentially vast amount of computing resources in an easy and user-centric way

[2]. While Cloud computing has been driven from the start predominantly by the industry through Amazon, Google and Microsoft, a shift is also occurring within the academic setting as well. Due to the many benefits, Cloud computing is becoming immersed in the area of High Performance Computing (HPC), specifically with the deployment of scientific clouds and virtualized clusters. There are a number of underlying technologies, services, and infrastructure-level configurations that make Cloud computing possible. One of the most important technologies is virtualization. Virtualization, in its simplest form, is a mechanism to abstract the hardware and system resources from a given Operating System.

This is typically performed within a Cloud environment across a large set of servers using a Hypervisor [7] or Virtual Machin1e Monitor (VMM), which lies in between the hardware and the OS. From the hypervisor [7], one or more virtualized OSs can be started, leading to one of the key advantages of Cloud computing. This, along with the advent of multi-core processors, allows for a consolidation of resources within any data center [2]. From the hypervisor level, Cloud computing middleware is deployed atop the virtualization technologies to exploit this capability to its maximum potential while still maintaining a given QoS and utility to users [5]. The physical server transition to virtualized infrastructure server have encountered crucial problems such as server consolidation, virtualization performance, virtual machine density, total cost of ownership (TCO), and return on investments (ROI).This paper introduces five distinct virtualized cloud computing servers (VCCS), and gives appropriate assessment to five well-known hypervisors in VCCS[10].

In this work there are some issues to cover (i) to monitor and study various performances issues in Server Virtualization and Network Virtualization., (ii) to Study the optimization process and design a controller for increasing resource capability and efficiency in Server with the help of virtualization and (iii) the goal is to consolidate the Data Center and increase its performance.

2. LITERATURE REVIEW

Virtualization has profoundly changed the information technology (IT) industry in different areas such as network, operating systems, applications or storage. Virtualization is no longer a subject only IT people know about it [1]. It has gained space on the administrators and directors vocabulary. Companies have realized that most of their systems were running at ratios of 10 percent or less of utilization, yet these systems continue to require space, power and cooling system as any other machine. Reducing these requirements would have a direct impact in reducing the IT budget an environment cares as the carbon footprint. Virtualization technology was the solution found by many companies, moving this technology into the mainstream. According to a recent IDC survey, companies that have deployed virtualization could see a return of investment of 472 percent in less than a year. The increased utilization and consolidation of x86 architectures had an important role for this as well. Many companies use this architecture because it has lower cost compared with others in the market. However, this architecture had historically hardware support issues for virtualization which significant degrade the performance of the virtual machine (VM) comparing with the same system running on a physical host. In order to solve this problem, Intel and AMD implemented architectural extensions to directly support virtualization in hardware.

This overcomes the classical virtualization limitations of the x86 architecture, improving important aspects such as performance and scalability making x86 server virtualization a keystone of most IT consolidation projects. The increasing investment and implementation of virtualization is comparable to the implementation of internet in companies at the end of the last decade. Recent advances in software and architectural support for server virtualization have created interest in using this technology in the design of consolidated hosting platforms. Since virtualization enables easier and faster application migration as well as secure colocation of antagonistic applications, higher degrees of server consolidation are likely to result in such virtualization-based hosting platforms (VHPs). There are two shortcomings in existing virtual machine monitors (VMMs) that prove to be obstacles in operating hosting platforms, such as Internet data centers, under conditions of such high consolidation: 1) CPU schedulers that are agnostic to the communication behavior of modern, multitier applications and 2) inadequate or inaccurate mechanisms for accounting the CPU overheads of I/O virtualization [15]. Traditionally, VMM schedulers have focused on fairly sharing the processor resources among domains while leaving the scheduling of I/O resources as a secondary concern. However, this can result in poor and/or unpredictable application performance, making virtualization less desirable for applications that require efficient and consistent I/O behavior [13,14,26].

2.1 Background

"Virtual machines have finally arrived"[4], said Robert P. Goldberg in 1974. Although this is, in fact, our current reality, it seems to have been the reality of the last 35 years with the slow adoption of virtual machines. Back in 1960s, virtualization was better known as time-sharing. Christopher Strachey, Professor of

Computation at Oxford University and the first director of the Programming Research Group published a paper titled “Time Sharing in Large Fast Computers”. His paper, as he refers later in a letter, “was mainly about multi-programming (to avoid waiting for peripherals) although it did envisage this going on at the same time as a programmer was debugging his program at a console. I did not envisage the sort of console system which is now so confusingly called time sharing. The multi-programming technique allows different users to execute jobs simultaneously. This was possible for one job to take advantage of the CPU, while another is not using the CPU, because it is waiting for an I/O device to store or retrieve data. From among some computers that took advantage of this technique, two are considered part of the evolutionary lineage of virtualization: Atlas and IBM’s M44/44X. The Atlas computer project was run by the Department of Electrical Engineering at Manchester University and funded by Ferranti Limited. It became operational in 1962 and it was considered the fastest computer in the world, until the release of the CDC 6600 in 1964. The Atlas Computer was the first supercomputer to take advantage of time sharing, multi-programming and shared peripheral control. It introduced a component called supervisor, which managed important resources, such as the processing time of the computer and passed special instructions called extra codes, which would help it to manage the computer environment for the user program’s instruction.

The name supervisor remembers the actual name hypervisor, and in fact, we can consider supervisor as its roots. Trying to compete with Atlas Computer, IBM created the M44/44X at the IBM Thomas J. Watson Research Center in Yorktown, New York. The central principle of its architecture was a set of virtual machines, one for each user. The M44, the real machine, was a modified version of IBM 7044 and the 44X was each virtual machine with an experimental image of the 7044. However, the incomplete implementation of the underlying hardware simulation by the M44/44X virtual machine made this project to fail, but soon IBM released its System/360 mainframe. Virtualization is best known to have been started with the development of the System/360 mainframe, by IBM Corporation. One of the problems presented at the time was the high cost of the machines, which were inefficiently used by people. The main operations were made by using key punches and submitting batch jobs.

Engineers were trying to let multiple users to come into the system, by making these batches more interactive. Implementing a time-sharing system at the time for multiple users was not an easy thing to do. For that very reason, IBM’s engineering team in Cambridge, Massachusetts presented an idea that would provide each user a virtual machine (VM), with a simple operating system, which only has to support one user. Robert Creasy and Les Comeau from IBM started to developed CP- 40 in 1964. This operating system was designed for the System/360 mainframe and was the first step to create virtual machines on these systems. It could support up to fourteen simultaneous virtual machines. Each virtual machine ran in a mode called “problem state”, where privileged instructions (e.g. I/O operations) would cause exceptions, which were intercepted by the control program and simulated. It was replaced in 1965 by CP-67 with the System/360 model 67. This new hypervisor was the first fully virtualized virtual machine operating system and because of this, it is referred in many documentation as the beginning of virtualization. It provided CMS to each mainframe user. CMS stands initially for Cambridge Monitor System, then it was designed as Console Monitor System, but at the end it was renamed to Conversational Monitor System. This CSM was a lightweight single-user operating system supporting time-sharing capabilities. The S/360-67 had a new component called the "Blaauw Box" (designed by Gerrit Blaauw) which implemented virtual memory. CP-67 had the functionality of memory sharing across VMs while providing each user with his own virtual memory space.

The advantages were impressive. It was possible to implement test platforms for software development and testing in a much more efficient way. In addition, it increased the debugging efficiency, since it was possible to analyze the virtual memory when the application failed. VM technology stayed as an internal project inside IBM until 1972, when it became a commercial product. One year after, Madnick and Donovan released the first security analysis about virtual machines. Even so, during these years, VM technology was an important technology in the mainframe world. IBM continued its VM technology on the System/360 and System/370, which appeared in 1970. Nowadays, it continues on the IBM’s 64-bit z/Architecture with their z/VM. Until late 90’s, some companies released their Virtual Machine, but none with continuous success. In 1998, in California, it was founded VMWare. Their first product was released one year after and has remained one of the most used products on the market, VMWare Workstation. In 2001 their first server edition of VMware ESX 1.0 (Elastic Sky X) was released. Beside VMware, some other companies, such as Sun, Microsoft, Parallels and Citrix, have released their virtualization products that have wide acceptance, some of them as a commercial product, other as an open source solution.

2.2 Virtualization

Defining virtualization is not an easy task because as we will see later, there are different types of virtualization and a definition that would be adequate for all is not easy to achieve. Singh describes virtualization as "framework or methodology of dividing the resources of a computer into multiple execution environments[2], by applying one or more concepts or technologies such as hardware and software partitioning, time-sharing, partial or complete machine simulation, emulation, quality of service, and many others". However, this definition leaves out cases as network virtualization, application virtualization or storage virtualization. Kiyanclar describes virtualization as "the faithful reproduction of an entire architecture in software, which provides the illusion of a real machine to all software running above it". Most of the definitions are correct if we only consider server virtualization; nevertheless, I adapted Singh's definition saying that Definition: Virtualization is as a framework dividing the resources of the device from the execution environment, allowing environment plurality by using one or more techniques such as time-sharing, emulation, partitioning.

2.3 CPU Virtualization

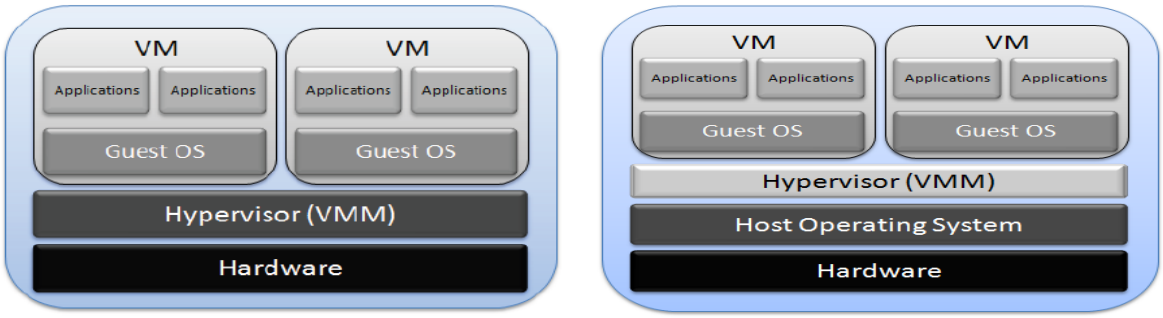
The x86 architecture is the most used CPU architecture in enterprise datacenters today, and virtualization can take benefits of that. The Intel 80286 chipset, introduced on February 1982, was the first of the x86 family to provide two main methods of addressing memory: real mode and protected mode. Later, in 1985, with the 80386 chipset, a third mode was introduced called virtual 8086 mode (also called virtual real mode, V86-mode or VM86). The VM86 allowed multiple real mode processes to be run simultaneously while taking full advantage of the 80386 protection mechanism. Real mode soon became obsolete because it had some disadvantages, such as it was limited to a one megabyte of memory and only one program can be run at a time. The same way, virtual mode was locked in at 16-bit and became obsolete with the high use of 32-bit operating system. Protected mode, by the other hand, is the natural 32-bit environment of the 80386 processor providing many features in order to support multitasking, such as hardware support for virtual memory and segmenting processor.

Protected mode in the x86 family uses 4 privilege levels, numbered from 0 to 3. Sometimes these levels are designated as rings, and the term comes from the MULTICS system [17], in which privilege levels were illustrated as a set of concentric rings. We are going to use the term "ring" as level, because it is a terminology more used. System memory is divided into segments and each segment is assigned and dedicated to a particular ring. The processor uses the privilege ring to decide what actions can be done with the code or data within a segment. As it shows in the Figure 1.1, Ring 0 is considered the innermost ring, which has total control of the hardware while Ring 3 is the outermost ring and has restricted access.

Figure 2.1: Privilege rings of the x86 architecture. High privilege:0; Low privilege: 3.

The supervisor mode is the execution mode on an x86 processor with unrestricted access, which enables the executions of all instruction, including I/O and memory management operations, which are privileged instructions. Operating system runs on this supervisor mode, normally on the ring 0. However if this ring is compromised, it will have direct impacts on the ring 3 (user mode). The idea of having isolated ring 0 for each virtualized guest is that if one of the ring 0 of a virtualized guest is affected by, for instance, a failure it will not have impact the ring 0 of others virtualized guest. In order to do this, it is necessary to make this ring 0 closer to the guest, residing in either ring 1 or ring 2 for x86 architectures. However, the further it goes from the real ring 0, the more distant is from executing direct hardware operations, resulting in a loss of performance and independence. Virtualization moves ring 0 up one level in the privilege rings model and places the virtual machine monitor in the next higher privilege ring. This will be the ring 0 and it is upon this the guest operating systems runs, while the VMM handles the interaction with the underlying hardware platform. VMMs can be classified in two types:

- Type 1: This type is also called as native or bare-metal because the hypervisor software runs on top of the host's hardware on the real ring 0 (Figure 1.2a). A guest operating system thus runs on another level above the hypervisor, allowing for true isolation of each virtual machine [12]. This is the classic VM architecture. An example of this implementation is the VMware [11] and Xen.
- Type 2: This type is also called hosted VMM because the hypervisor software runs within a normal host operating system already installed, usually in ring 3 (Figure 1.2b). This type of VMM has a lower performance than the other type because factors as calls to the hardware must traverse many diverse layers before the operations are returned to the guest operating system. Examples of this implementation include VMware Workstation, Sun Virtual Box and Parallels Workstations.



(a) Type 1 Hypervisor/VMM (b) Type 2 Hypervisor/VMM

Figure 1.2: The two types of Hypervisors/VMMs.

It is important to clarify that the term Hypervisor has the same meaning as VMM. Privileged instructions trap when they are called from user mode. When called from supervisor mode (kernel mode), they do not trap. A trap passes control to a trap-handler, located in the kernel, so that processor mode changes. A sensitive instruction is an instruction that reads from or writes to memory locations or sensitive registers. All sensitive instructions have to trap on a virtualizable architecture and therefore, in this context, sensitive instructions can be seen as subset of privileged instructions. A VM cannot access hardware directly without passing the hypervisor, which is responsible for maintaining control over sensitive instruction and the hardware. When a VM tries to access sensitive data, the instruction is trapped and control is passed to the hypervisor. This happens because VMs runs in user mode but if the guest OS attempts to access sensitive data, a trap will occur. Then, the hypervisor, which is running in supervisor mode, it will catch this trap, inspect the state of the guest OS that cause it and emulate the behavior that would occur if the guest OS was running on a real machine. The hypervisor will then resume the VM, allowing the executing to continue. This method is called "trap and emulate". The Popek & Goldberg requirements are satisfied by this approach as long as the processor is guaranteed to trap whenever any privileged operation is attempted in user mode. However x86 architectures does not guarantee this, since there are 17 sensitive non privileged instructions that disables "trap and emulate".

The control is not passed to the hypervisor when these sensitive instructions are called and so, the hypervisor cannot emulate the expected behavior. According to Smith and Nair, the 17 sensitive non privileged instructions fall into two categories: (i) protection and (ii) sensitive.

Protection system references: These instructions reference address relocation system, memory system or storage protection system. The problem is the possibility of a virtual machine to access locations outside its virtual memory. An example presented by the authors is the MOVE instruction, which moves a value from general-purpose register to the CS register, the control register that specified the current privilege ring number in bits. An instruction such as move ax, cs, when executed in the user mode disallows the CS register to be loaded. This happens to offer some protection, but which makes it not well virtualizable is that instead of generate a trap, the instruction generates a no-op. The instructions in this category are:

```
CALL: Call procedure
JMP: Jump
INT n: Software Interrupt
LAR: Load access rights
LSL: Load segment limit
```

MOV: Move data between general-purpose registers or between memory and general-purpose/segment registers. It can also move immediate to general purpose registers

```
POP: Pop off of stack
PUSH: Push onto stack
RET: Return
STR: Store task register
VERR: Verify segment for reading
VERW: Verify segment for writing
```

Sensitive register instructions: These instructions read or change resource related registers and/or memory locations, such as a clock register or interrupt registers. The authors detail the example of the POPF instruction. POPF pops the flag registers from a stack held in memory. One of the flag registers is the interrupt-enable flag (IF), which can only be modified in privilege mode. The problem happens when the guest OS requires that the IF bit be changed and since it is running in the user mode under the VM, then the IF bit cannot be changed. This can lead the guest OS to take erroneous action because the flag bit was not set as expected. These instructions are:

```
PUSHF: Push EFLAGS onto stack
POPF: Pop EFLAGS from stack
SGDT: Store global descriptor table register
SIDT: Store interrupt descriptor table register
SLDT: Store local descriptor table register
Intel 286 processor.
```

On the x86 architecture, the operating system normally runs in ring 0, because it is the When we add virtualization on this scheme, what normally happens is that VMM runs in ring 0, since it must have privileged control of platform resources and the guest operating system goes to ring 1 or ring 3. In order to overcome the limitation of implement CPU virtualization on x86 architectures, some techniques such as direct execution combined with fast binary translation and Para virtualization. Binary translation (BT) is not a new technique and can be use for various purposes such as migrations between different architectures. The combination of direct execution with BT was an idea developed by VMware to be used for CPU virtualization. This technique allows running supervisor mode code controlled by the binary translator. The translator replaces the privileged code into a similar block, patching the sensitive, unprivileged instructions. This translated block can then run directly on the CPU and they are cached by the BT system in a trace cache so they can be used on subsequent executions.

Using BT, only the sensitive instructions like POPF are replaced while the normal instructions are executed unchanged. This binary translation is only applied when the code first executes. Para virtualization uses a different approach to overcome the x86 virtualization issue. With Para virtualization, the non virtualizable instructions are replaces with virtualizable equivalent ones. This requires the guest OS to be changed although most of the normal applications remain unchanged. One difference with the BT approach is that in para virtualization, the guest OS knows that it is running in a virtual environment, while using BT the guest OS have the illusion that is running on a real machine. The para virtual hypervisor is smaller and easier to implement containing only a small interface for the 17 sensitive, unprivileged instructions and then is more trustworthy than one using BT and similar to a VMM using “trap” and emulate”. The second generation of hardware virtualization (Intel VT and AMD-V) was designed with the goal of eliminate the need for BT and para virtualization on the x86 architecture. The CPU creates containers and introduces new modes of operations that can distinguish if the CPU is real or virtual.

VMM runs the highest privilege container which is commonly designed as ring -1. Guests run within a lower privileged container, although conceptually is considered a ring 0. This allows guest OS to run at their normal ring and only leaving when the guest tries to execute a privilege or sensitive instructions which will trap to the VMM. This can be seen as “trap and emulate” and therefore, all security issues related with it can be also applied to hardware virtualization.

2.4 Memory Virtualization:

Normal operating system use page tables to translate virtual addresses into physical addresses. Virtual machines brought new challenges regarding memory virtualization since memory is going to be shared although isolation as to be guaranteed. We can consider three classes of addresses on a virtualized system:

- Virtual addresses, which are the same as the ones used by a conventional OS
- Guest physical address
- Machine memory

Guest operating systems maintain page tables that translate from virtual to pseudo-physical addresses, and hypervisor maintains separate shadow page tables that translate from virtual addresses to machine addresses [24]. The recent x86 CPUs support memory in hardware. Translation from virtual to physical addresses is performed by the memory management unit (MMU) and the most used parts of the page tables are cached in the translation look aside buffer (TLB). Guest OS sees page tables, which run on an emulated MMU. These tables provide the guest OS with the illusion that it can translate the virtual guest OS addresses into real physical addresses, but it is the hypervisor that deals with it. The real page table is the shadow page table used to translate the virtual addresses of the guest OS into the real physical pages.

The classic implementation of hypervisors maintains a shadow page table, which allows to control what page of the machine's memory is available to a virtual machine. Just like in a traditional operating system's virtual memory subsystem, when the memory allocated to VMs exceed the host physical memory size, the hypervisor can page the VM to the disk. This way, the hypervisor can dynamically control how much memory each VM receives. However, the hypervisor's virtual memory system does not have the perception of which pages are good for paging out but the guest OS should have because it can identify, for instance, that the process that created a page has exited and so nothing will access to that page. This page would not be a good candidate for paging out, but since the hypervisor's virtual memory system has no clue about that, it might page out that page. To deal with this problem, VMware's ESX Server created a mechanism which allows the hypervisor to ask to the guest OS for the pages it can swap out, using a balloon process [11,24]. A balloon process runs inside a guest OS and communicates with the hypervisor.

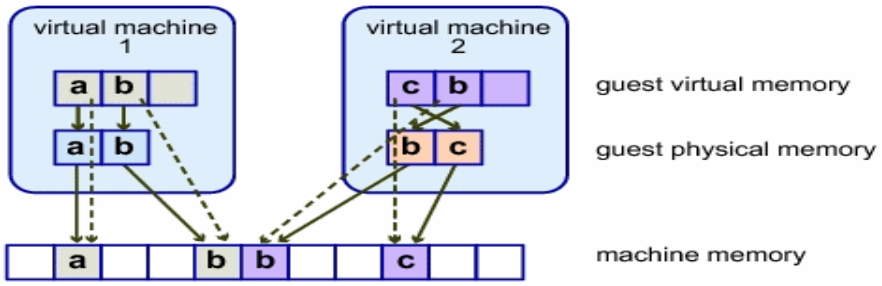


Figure 2.3: ESX server memory mapping .

When the hypervisor needs to take memory away from the VM, it communicates with the balloon process to "inflate" the process, allocating more memory. This will force the guest OS to select the pages to give to the balloon process, which will be passed to the hypervisor for reallocation, but also it will force the guest OS to page memory to the virtual disk. VMware engineers also develop a mechanism that would decrease the memory used by several VMs running the same version of an operating system. They identify that different virtual machines running the same operating system will produce redundant copies of code and data stored in memory that could be shared among them. To address this, they have designed a content-based page sharing system, analyzing if the contents of physical pages are identical. When such content is identified, the hypervisor modifies the VM's shadow page table to point to only a single copy and freeing the redundant copy. If the content is changed, then the hypervisor provides the VM with its own copy of the page, in a copy on-write page-sharing scheme.

However, this can have some drawbacks. In order to detect duplicate pages, it is necessary to periodically scan the memory and build a list of page fingerprints, which will be used to compare page contents. VMware ESX scans frequency is set by default to once an hour (with the maximum of six times per hour), but this means that short-lived sharing opportunities will be missed. Without shadow pages, it would be necessary to translate guest virtual memory into guest physical memory and then translate this one into the real machine memory. The shadow page table avoids the double bookkeeping by making the MMU work with the guest virtual memory to real physical memory page table. Figure 1.3 illustrates the VMware ESX server implementation of memory virtualization. In this figure, the boxes represent the pages and the arrows show the different memory mappings.

As we can see, there are arrows from the guest virtual memory to the guest physical memory which represents the mapping maintained by the page tables in the guest OS. The arrows from guest physical to machine memory represent the mapping maintained by the hypervisor, while the dashed arrows show the mapping from guest virtual memory to machine memory in the shadow page tables which is also maintained by the hypervisor. Nevertheless, there is a performance issue because each update of the guest OS page tables forces a shadow page table bookkeeping. The second generation of hardware virtualization (Intel VT and

AMD-V) partly solves this problem with their AMD's Nested Page Tables (NPT) and Intel's Extended Page Tables (EPT). When using nested paging, the CPU caches both guest virtual and physical memory as the guest physical memory to real physical memory transition in the TLB. The TLB has a new tag called Address Space Identifier (ASID) which allows to keep track of which TLB entry belongs to which VM. This way, entries of different virtual machines can coexist in the TLB at the same time. Using nested paging has an increment importance if there is being used multiple virtual CPU per VM, because they have to sync the page tables many times with direct impact on the shadow page table update. With NPT, the CPU only has to synchronize TLBs as it would happen in a non-virtualized environment.

2.5 Device and I/O Virtualization:

The VMM virtualizes the physical hardware and allows each virtual machine a set of customizable virtual devices. Most of this virtualized I/O requires software drivers that run on the host operating system to access the real hardware. If it is a type 2 hypervisor, then it will use the device drivers already in the host OS, otherwise it may be necessary to develop its own device drivers for the hardware on the machine, like in the case of VMware ESX. Emulation is normally used for a VMM to handle I/O devices, and it is the VMM the responsible to implement a software model of the I/O device, making believe the guest OS that it is communicating to a hardware device, when is communicating with a software model. The I/O virtualization may provide to the guest, virtual hardware that does not exist in the real hardware, for instance, emulating an IDE hard disk when the real hardware is SATA. The direct memory access (DMA) has problems when used with virtual machines. The DMA controller can write to the entire physical memory instead of only the memory assigned to the guest OS. In order to deal with this problem, Intel and AMD added I/O Memory Management Unit (IOMMU). With IOMMU it is possible to restrict which physical address a device may access.

3. SERVER VIRTULIZATION AND ARCHITECTURE

There are many different implementations of server virtualization on, and for a big range of CPU platforms and architectures. Informally, server virtualization can be seen as creating many virtual systems within a single physical system. To accomplish this, we can take three approaches: physical layer, virtualization layer and OS layer. Hardware partitioning divides a single physical server into partitions where each partition is able to run an operating system while hypervisor places a layer of software between the physical hardware and the multiple operating systems that will share the same physical hardware.

Physical layer: (i) Hardware partitioning: The server is physically segmented into distinct smaller systems that will act as a physically independent and self-contained server. Normally each of these smaller systems has their own CPUs, OS, boot area, memory and net- work resources. (ii) Virtualization layer: Hypervisor technology can be organized in some distinct categories: Full virtualization: Allows virtual infrastructures to run un modified operating systems in isolation. The operating system running inside the virtual machine is called guest operating system. This approach was pioneered in 1967 with IBM CP-40 and CP-67, predecessors of VM family. In order to implement full virtualization, it is necessaryafullcombinationofhardwareandsoftware. It was not possible on IBM System/370until 1972 and it was not natively possible in the x86 architecture until 2005 when Intel and AMD added the hardware virtualization extensions (Intel VT and AMD-V respectively).

Nevertheless, many companies tried to accomplish full virtualization. on x86 architecture even before Intel VT and AMD-V additions. VMware uses a combination of direct execution with binary translation techniques to accomplish full virtualization of an x86 system. This provides full disassociation of the guest OS from the underlying hardware by the virtualization layer. As depicted in Figure 2.4, the OS requests that needs to interact with the hardware are needs to be translated by the VMM, replacing non virtualizable instructions with new sequences of instructions, which have the same result on the virtual hardware , while the user's applications are directly executed on the processor for high performance virtualization.

Para virtualization: modifies the guest kernel system in order to purge the necessity of binary translation. It has the advantage of higher performance but has the drawback of needing a modified operating system kernel. The fact the virtual platform is not identical to the real hardware, it makes necessary for the operating system to be ported to the abstracted machine interface. This could be seen as a violation of the Goldberg's equivalence requirements, because the architecture-dependent part of the operating system kernel needs to be changed. Then on virtualizable instructions are replaced with hyper calls that communicate directly with the virtualization layer hypervisor. The architecture independent part and the entire user mode software stack stay un modified.

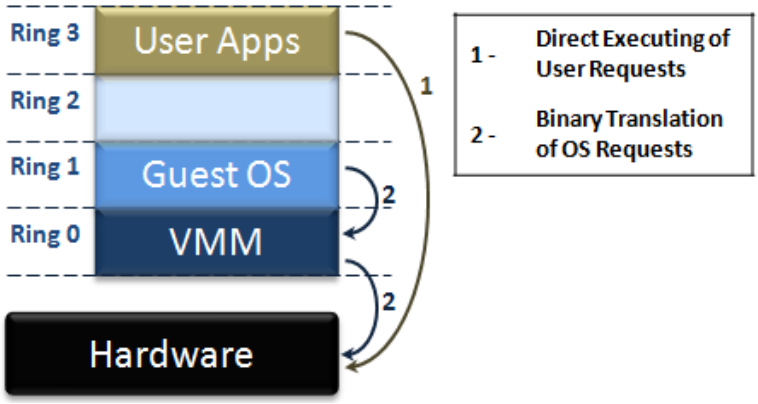


Figure 3.1: The binary translation approach to x86 virtualization used by VMware.

Emulation: Sometimes people confuse emulation with full virtualization. Although both run un modified guest operating systems, they are both very different. In emulation, the virtual machine simulates the entire hardware set needed to run the un- modified guest OS normally for a completely different hardware architecture. There are some utilities for this technique. For instance, it allows developing programs and operating systems for new hardware design before the hardware is physical available. It also allows, as we are going to see later, to run an un modified version of Server Virtualization provides following things. (i) Servers associated to virtual machines instead of physical machines, (ii) Virtual machine: operating system + software apps.

We adopt Server Virtualization for following reasons: (i) Allows for hosting multiple applications and services in a seamlessly way. (ii) On-demand server allocation and dynamic capacity dimensioning, (iii) Server consolidation: smaller number of servers for hosting different apps, (iv) Efficient use of computer resources: increase of server utilization and energy efficiency [1].

4. NETWORK VIRTULIZATION AND ARCHITECTURE

When people talk about network virtualization, probably the first thing that comes to their minds is Virtual Private Network (VPN)or perhaps Virtual Local Area Networks (VLAN). However there is more when we talk about network virtualization[3]. The most used network virtualizations are:

1. Virtual LAN (VLAN): Defined in the IEEE802.1Q standard, is a method of creating independent networks using a shared physical network. They are used to logically segment broadcast domains and control the interaction between different network segments. VLANS is a common feature in all modern Ethernet switches, allowing creating multiple virtual networks, which isolates each segment from the others. All the available resources are segments and allocated to each of these segments. Therefore, VLAN is a safe method of creating independent or isolate logical networks within a shared physical network.
2. VirtualIP (VIP): A VIP is an IP address that is not associated to a specific computer or network interface, but is normally assigned to a network device that is in-path of the network traffic. Incoming packets are sent to the VIP but are redirected to the actual network interface of the receiving host or hosts. It is used in solutions like High-Available and Load-Balancing, where multiple systems have a common application, and they are able to receiving the traffic as redirected by the network device.
3. Virtual Private Network (VPN): It is a private communication network that uses public network, such as Internet. Its purpose is to guarantee confidentiality on an un- secured network channel, from one site to another. It is normally used as a means of extending remote employee home networks to the company network. This is normally done by using special software(as Cisco VPN Client),but after the connection beingestablished,alltheinteractionwiththeotherresourcesonthenetworkishand-led as if the computer was physically connected to the same network, although this depends of the way security policies are applied.

5. BENEFITS OF VIRTULIZATION

Nowadays, virtualization is in the vanguard, helping companies to take advantage of two important properties of virtualization: scalability and management. It can bring many benefits and there are many reasons for its application. We are going to see some of the Key benefits of virtualization.

One of the benefits of virtualization has historical reasons and is related with the misused of servers, mainly because of Microsoft Windows NT Server. This operating system started to be used in datacenters back in 1990 although it was a hard battle in the beginning to be accepted as a good operating system for enterprise datacenters. Today we know it was a battle won and Microsoft has it share on most datacenters. However, Windows NT was a monolithic OS and when an application freeze it would often freeze the OS causing the well known Blue Screen of Death. Administrator started to apply the philosophy of single-purpose servers, running a single application per server.

This would prevent that if one application fails, causing the operating system to fail, it would not disrupt any other application running on another server. For these reason, each time it was necessary a new business application, it was also necessary another server to be used. Over time, Microsoft solved the monolithic problem but administrator's habits were already created. However, it is not only Microsoft's fault. Many software vendors required their application to be isolated so they can support them. In addition, security taught us that the less application we have installed in a machine, the smaller will be the attack surface. For this reason, companies have realized that most of their systems were running at ratios of 10 percent or less of utilization, yet these systems continue to require space, power and cooling system as any other machine. Instead of having some servers for their principal services (e.g. email, stock programs), they can invest in a better server and consolidate all those services in separated virtual machines. With that, they gain scalability, since they can upgrade their virtual machine without necessary upgrade the physical machine. They gain security, since they have a more control environment and easier to backup and restore, which also is related with management. However there is a drawback which has to be balanced. Since all the main services of the company are installed on one only server (or few), this is considered a single point of failure.

High availability is essential for corporate environment where availability of their services is crucial to the success of their business. One of the advantages of using virtual machines is that it can be restored very easily (one or very few files) and so, if updated backups exists, then the IT manager can simply restore that file on another machine. Normally, it is recommended hot standby virtual machines with at least two servers. Taking the example of email server and stock programs, on the server A would be installed the email server running on a VM (VM.A.1.ori) and an updated offline copy of the VM (VM.B.1.bck) with the stock program running on the server B. And the opposite on server B, which would be an update offline backup of the email server that is running on server A (VM.A.1.bck) and the stock program running on a VM (VM.B.1.ori). If any problem happens to one of the servers, there would be another server that could temporary support both VMs.

This example shows also another benefit of virtual machines, which is disaster recovery (DR). Some companies have their disaster recovery center in another geographic location and applying a hot standby allows them to easily replicate the VM to their DR center and when needed, be able to quickly make the services available. Another advantage of virtualization on disaster recovery is when facing an exploit that can compromise a server, be able to use a VM trustworthy (i.e. not infected) baseline installation of the affected system, patch them and turn into production, leaving the infected system for analysis and evaluation. Virtual machines offer a perfect environment for development and research. According to Silberschatz et al. , changing an operating system is a difficult task because they are complex programs and since they executes in kernel mode, the impact of changing a pointer can destroy the entire file system. Therefore, it is necessary to test all changes to the operating system. With virtualization, the system programmer can have their own virtual machine and system development or test is made on those virtual machines instead of on a physical machine.

This reduce the system development time and cost, increasing the productivity. Another benefit of virtualization is the possibility of having multiple operating systems running, even those systems that are obsolete and cannot be utilized by the newer hardware resources, may be supported to run on a virtual machine. It is also useful for testing software solutions. Using virtualization, a company can try some solutions without the necessity of using many real servers. The same way it is useful for software developers to simulate the production environment the best they can using virtual machines, and that way, debugging their application in an almost real environment. In addition to testing new solutions, virtual machines are useful to test new patches before applying them into production systems. Virtual Machines are also very useful for forensic team research. It is possible to clone a potentially compromised host into a VM and do further investigation without the need of the physical machine. The investigation team can also take advantage of snapshots to return to a previous state. The same can be said for malware investigation team. It

can be very useful to use a VM since it guarantees isolation and to have the ability to use the snapshot function. However, malware does not always have the same behavior inside a VM as on a real machine. There are other benefits of using virtualization. Honeypots or honeynets are intrinsically related with server virtualizations and are normally used by organizations to attract possible attackers. Honeypots can be described as servers with fake information in an isolated network, simulating a real DMZ or Intranet in order to analyze the attackers' behavior. The logs generated on these systems is much less than the generated every day on other security systems as firewalls, IDS and IPS alerts or even system logs, but their value is normally high, because it include most likely scans, probes and attacks. Detect intrusions using hypervisor is purposed on some papers as a way to alert for root kits running on the guest OS.

6. VMWARE

Founded in 1998 by Diane Greene and Dr. Mendel Rosenblum along with two students from Stanford University and a colleague from Berkley, VMware is a well known company on the x 86 virtualization markets. In October of the same year, these five founders filed for a patent regarding new virtualization techniques. These techniques were based on a project called SimOS conducted at Stanford University. The U.S. Patent 6,397,242 was awarded on May 28, 2002. Their first product was VMware Workstation with the first version being released on February 8, 1999 for Windows and Linux. This is one of the most successful products from VMware for desktop and stays as a commercial product with its current version 6.5.x. VMware Workstation is a type 2 hypervisor, supported on top of a Host OS – either Windows or Linux – and able to create virtual machines for a variety of guests OS, such as Solaris x86, Netware, FreeBSD, Windows and Linux. In late 2000, they released their first version of the server virtualization platform called VMware GSX Server. As in the case of VMware Workstation, this GSX Server needs to be installed on top of an existing Windows or Linux operating system. In 2006, VMware GSX Server was renamed to VMware Server and it is now released as freeware. In 2001, VMware release their first version of Elastic Sky X (ESX).

This is their first server product with an approach different to that of the workstation version. In this case, VMware ESX does not require a host OS, but instead it has its own native hypervisor on a bare-metal system. This had the drawback to support less hardware but had the advantage of requiring less overhead to host each virtual system. On May 2009, it was released VMware ESX v.4.0. Figure 5.1 shows the many releases of VMware ESX Server [11]. The interval between the major releases and the minor release has been growing, from 14 months between v.1.0 and v.1.5 to 26 months between 3.0 and 3.5. This is normally due the fact minor releases are the accumulation of many patches released since the major release, but also with added features.

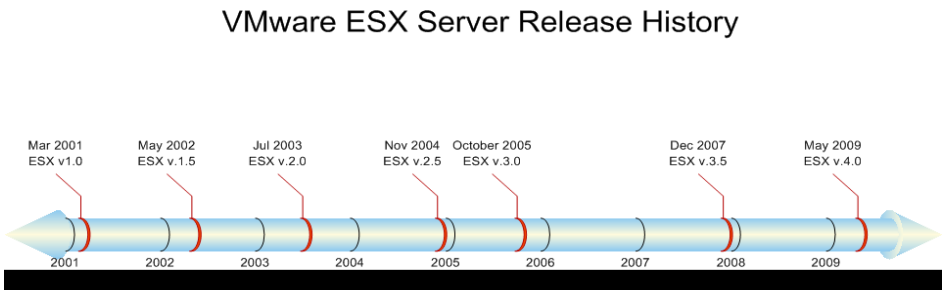


Figure 6.1: VMware ESX server release history

6.1.1 The ESX Platform

The core of VMware ESX has three main modules capable of regulating CPU affinity, memory allocation and oversubscription, network bandwidth throttling and I/O bandwidth control. Along with these, Virtual Machine File System completes the VMware ESX base platform.

The three primary components of VMware ESX are:

1. **Physical Host Server**: This is related with the physical host server where VMware ESX runs on. VMware had a Hardware Compatibility List (HCL) that includes some of the main servers' brands on the market, such as Dell, Hewlett-Packard, IBM among others. For support reasons, it is convenient to use ESX only on the servers reported on HCL.
2. **VMkernel**: The VMkernel is the center of the VMware ESX hypervisor, and it is a high performance operating system developed by VMware to run on the ESX host server. Although it has some similarities, it is not a Linux kernel. It does not share the Linux data structures or symbols neither depend on the Linux kernel for any services. The VMware ESX has a modular approach,

loading various system modules. However, the version 3.x presents some innovations regarding this modular approach, such as allowing new devices to be added without the need of recompile the VMkernel. The Console Operating System (COS): The service console has been upgrade from being based on a variant of Red Hat version 7.2 with ESX 2.x to Red Hat Enterprise Linux 3, Update 6 for ESX 3.0 and Update 8 for ESX 3.5. The COS does not interact with system hardware that is a job made by VMkernel. The COS function is to provide the executing environment for monitoring and administrating ESX. On versions before ESX 3.0, VMkernel would only load after COS was fully booted. In ESX 3.0 this was changed and now VMkernel runs before than COS. In fact, COS runs within a specialized VM with more privilege than the normal VMs.

3. **Virtual Machine File System (VMFS):** The VMFS is a high performance cluster file system created by VMware. VMFS has many advantages compared to conventional file system. One of those is the fact that up to 32 ESX Servers can concurrently read and write to the same storage by using per-file lock. It has some other important security features such as the fact that it allows live migration of powered-on virtual machines from a host server to another and by using distributed journaling, it is possible to recover VMs faster and more reliable in a case of a server failure. However, according to Scott Davis from VMware, the choice between VMFS and NFS will depend of what type of storage infrastructure the organization is familiar with. If it uses block based storage, then it is recommended to use VMFS, but if it is already using a Network-attached storage (NAS), then it would be recommended to use NFS instead.
4. **VirtualCenter**: The VMware VirtualCenter is the management console used to control the virtualized enterprise environments. Is provides services such as access control, performance monitoring, and configuration. On December 28, 2004, About if he thinks this is an approximated number.

7. XEN

The Xen project was first described in the paper "Xen and the Art of Virtualization" presented at SOSP in 2003. It was a project originally developed by the System Research Group at the University of Cambridge Computer Laboratory and was part of the XenoServers projects which had the goal of build a public infrastructure for global-scale service deployment. The first public release of Xen 1.0 was made in October of 2003 and the project had a good evolution all over the years, getting maturity in virtualizing resources like CPU, memory, disk and network[3]. For that, Xen had many project contributors, including AMD, HP, Intel, Novell, RedHat and XenSource.

XenSource, Inc. was a company founded by Ian Pratt, senior lecturer at Cambridge and lead of the Xen project, with the goal of supporting and developing the open source Xen project and to create a commercial enterprise version of the software. In 2005, XenSource release Xen 3.0, which was the first enterprise-class of Xen, supporting up to 32 processors. It was also the first version with built-in support for Intel's VT and with support for Physical Address Extensions (PAE) to support 32-bit host servers with more than 4GB of memory. At the time, ADM-V was not released yet, but eventually it was supported too. In order to compete more directly with VMware, XenSource released XenOptimizer, an integrated virtual infrastructure management platform competing with VMware's VirtualCenter and VMotion technologies. However, XenSource was still missing the point, providing separated products when VMware was taking the market with their consolidated product ESX. In order to change this, XenSource released their first version of XenEnterprise 3.0, a product based on Xen v.3.0.3 and a directly compete with VMware ESX Server. This version had two important features:

It included a new management and monitoring console based on XenOptimizer and it was the first Xen product supporting Windows guest operating systems. To this contributed the partnership made between XenSource and Microsoft. Xen is an open-source hypervisor for both 32 and 64 bit process architecture that runs on top of the bare-metal. It allows to securely and efficiently run several virtual guest OS on the same host at the same time. Xen as many features as:

1. Near native performance on the virtual machines
2. Full support on x86 (32-bit) with and without Physical Address Extension (PAE)
3. Full support on x86 with 64 bit extensions
4. Support for almost all hardware with Linux drivers available
5. Live migration of running virtual machines between two physical hosts with zero downtime
6. Support of Hardware Virtualization extensions from Intel (Intel-VT) andAMD(AMDV), allowing unmodified guest operating systems.

Initially, Xen would only support one mode of virtualization called para virtualization. In this mode, the guest OS must be modified and the kernel recompiled to support proper interaction with the Xen hypervisor. This had the drawback of limiting the choice of OS, since it would have to be open source, but had the advantage of improving its performance. Since version 3.0 of Xen, a new mode was introduced called full virtualization. This mode was only possible with the addition of hardware virtualization extensions and so, the physical hosts must have an Intel-VT or AMD-V processors.

Unmodified guest OS as Microsoft Windows can now run in full virtualization mode on Xen, with a minor performance penalty. On the Xen technology, there are two important and distinct domains: Dom0 and DomU. The Dom0 is a special privileged domain also designated as Domain0. It is the first domain launched when the Xen's system is booted. This domain can be used to create and configure all the guest domains. These guest domains are called DomU because they are unprivileged domain.

8. QEMU

The terms virtualization and emulation are occasionally used to describe virtualization products. Sometimes people say, by the fact that they are running something that was supposed to run in another platform, then that is considered virtualization. For instance, it is possible to run ancient arcade games in a normal computer, using images of their ROMs. In this case, what we are running is an emulator program and not a virtualization program.

Virtualization techniques normally run software that was compiled for the native instruction set of the physical hardware on where the virtual machine is running, while emulation techniques normally emulates all the physical environment, including processor. Microsoft Virtual PC and QEMU are two examples of emulation product. QEMU stands for Quick Emulator and it is an open source emulator, which uses dynamic recompilation with the purpose to reduce the overburden caused by the emulation.

It was developed by Fabrice Bellard and it can emulate a multiplicity of architectures, processors and related peripheral hardware. It can run OS and software that was compiled for platforms such as 32 - and 64 - bit x86, 32 - and 64 - bit PowerPC, Motorola 68000, 32 – and 64 - bit SPARC, SH, MIPS, and ARM. It can virtualize a complete hardware environment for each of these processors and architectures, enabling the possibility to run unmodified guest OS for any of those. It is used as the base of various virtualization products, like KVM, Virtualbox, KQEMU and even Xen.

9. OPEN SWITCH

Open vSwitch is an open source switching stack for virtualization[5]. The most powerful piece of real estate in a network is the edge and the hypervisor is the new edge. Open vSwitch is a production quality, multilayer virtual switch licensed under the open source Apache 2.0 license. It is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols (e.g. NetFlow, sFlow, SPAN, RSPAN, CLI, LACP, 802.1ag). Two ways to view OVS: (i) Gaining back visibility and control that usually comes from the features of a hardware switch, and (ii) An opportunity to exploit the flexibility that comes from software and virtualization.

9.1 Open vSwitch Architecture

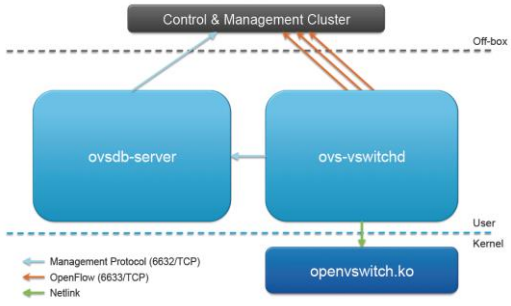


Fig 8.1(a) Open vSwitch Architecture.

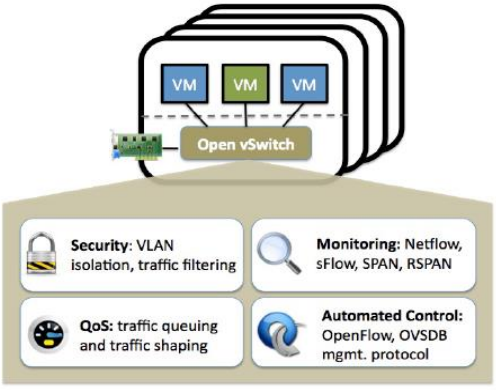


Fig 8.1(b) Open vSwitch Architecture

Basic Features
Open vSwitch brings many features standard in hardware devices to virtualized environments: VLANs, LACP and other bonding modes, STP, QoS shaping and policing, ACLs over a range of L2-L4 protocol, NetFlow, sFlow, IPFIX, mirroring, A variety of tunneling protocols

Remote programmability and management features: OpenFlow 1.0 and experimental support for versions 1.1-1.3, All features and status remotely configurable and viewable, Many extensions for supporting high availability control clusters.

Advanced Capabilities
Programmability requires primitives more similar to a CPU than a network ASIC. Over time, the flow table in Open vSwitch has slowly changed from a list of policies to a nearly general purpose processing pipeline.

Examples:
Resubmit: Move between multiple independent flow tables, similar to subroutines.
Registers: Storage for intermediate metadata, including manipulation functions such as a stack.
Learning: Dynamically generate new flows based on packet traffic patterns.
Hashing and Perform actions based on deterministic or probabilistic
Sampling: properties of the traffic.

10. CONCLUSION

Server virtualization and Network virtualization is always necessary for optimizing the IT Infrastructure. In this paper we have described about the architecture and several important tools for Server Virtualization and Network Virtualization. This technology will become increasingly important in the future and be introduced for widespread as a common technology for building the IT infrastructure.

REFERENCES

[1] ”Energy–aware resource allocation heuristics for efficient management of Data Centers for Cloud Computing”, Anton Beloglazov, Jemal Abawajy, Rajkumar Buyya, Future Beneration Computer Systems(2012), Elsevier, also available at Science Direct, Sponsored by Cloud Computing and Distributed Systems(CLOUDS).

[2] ” Linear Scheduling Strategy for Resource Allocation in Cloud Environment “,Abirami S.P. and Shalini Ramanathan, International Journal on Cloud Computing: Services and Architecture(IJCCSA),Vol.2, No.1,February 2012.

[3] Performance Measurements and Analysis of Network I/O Applications in Virtualized Cloud, 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD),5-10 July 2010 Page(s): 59 – 66, E-ISBN : 978-0-7695-4130-3, Print ISBN: 978-1-4244-8207-8, INSPEC Accession Number: 11499480.

[4] ”Performance evaluation of mapreduce using full virtualization on a departmental cloud”, Horacio GonAlez Velez, Maryam Kontagora, Int. J. Appl. Math. Comput. Sci., 2011, Vol. 21, No. 2, 275–284 DOI: 10.2478/v10006-011-0020-3(AMCS).

[5] ” Research and Evaluation of Network Virtualization in Cloud Computing environment”, Zongjian He, Guanqing Liang, IEEE Third International Conference on Networking and Distributed Computing (ICNDC), 2012 at Hangzhou,40-45, ISSN :2165-5006,Print ISBN:978-1-4673-2858-6,INSPEC Accession Number:13263829,Digital Object Identifier :10.1109/ICNDC.2012.18.

[6] ”Evaluating and modeling virtualization performance overhead for cloud environments”, nikolaus huber, marcel von quast, Michael Hauck, Samuel Kounev, CLOSER 2011 - Proceedings of the 1st International Conference on Cloud Computing and Services Science, Noordwijkerhout, Netherlands, 7-9 May, 2011. SciTePress 2011 ISBN 978-989-8425-52-2.

[7] ” Evaluation of Virtual Machine Performance and Virtualized Consolidation Ratio in Cloud Computing System”, Bao Rong Chang, Hsiu-Fen Tsai, Chi-Ming Chen, Journal of Information Hiding and Multimedia Signal Processing©2013 ISSN 2073-4212 Ubiquitous International Volume 4, Number 3, July 2013.

[8] “Monitoring-as-a-Service in The Cloud”, Shicong Meng, Ling Liu, ICPE’13, April 21–24, 2013, Prague, Czech Republic. ACM 978-1-4503-1636-1/13/04.

[9] ”Analysis of Virtualization Technologies for High Performance Computing Environments”, Andrew J. Younge, Robert Henschel, James T. Brown, Gregor von Laszewski, Judy Qiu, Geoffrey C. Fox, Pervasive Technology Institute, Indiana University 2729 E 10th St., Bloomington, IN 47408, U.S.A. {ajyounge,henschel,jatbrown,gvonlasz,xqiu,gcf}@indiana.edu. Proceedings of the IEEE 4th International Conference on Cloud Computing (CLOUD. 9-16.

[10] ” Evaluating and modeling virtualization Performance overhead for cloud environments”, Nikolaus Huber, Marcel von Quast, Michael Hauck, Samuel Kounev. *Journal of Information Hiding and Multimedia Signal Processing*© 2013 ISSN 2073-4212, Ubiquitous International Volume 4, Number 3, July 2013.

[11] " Memory Resource Management in VMware ESX Server", Carl A. Waldspurger VMware, Inc. Palo Alto, CA 94304 USA carl@vmware.com, Proceedings of the 5th Symposium on Operating Systems Design and Implementation.
[12] " Resource Management for Isolation Enhanced Cloud Services", Himanshu Raj, Ripal Nathuji, CCSW'09, November 13, 2009, Chicago, Illinois, USA. Copyright 2009 ACM 978-1-60558-784-4/09/11 ...$10.00.
[13] " Impact of Cloud Computing Virtualization Strategies on Workloads 'Performance" Qingling Wang, Carlos A. Varela Department of Computer Science Rensselaer Polytechnic Institute, Troy, NY, USA http://wcl.cs.rpi.edu/ {wangq9, cvarela}@cs.rpi.edu, *UCC, page 130-137. IEEE Computer Society,* (*2011*).
[14] " Scheduling I/O in Virtual Machine Monitors", Diego Ongaro Alan L. Cox Scott Rixner, VEE'08, March 5–7, 2008, Seattle, Washington, USA. Copyright c 2008 ACM 978-1-59593-796-4/08/03...$5.00.
[15] " Xen and Co.: Communication-Aware CPU Management in Consolidated Xen-Based Hosting Platforms", Sriram Govindan, Jeonghwan Choi, Arjun R. Nath, Amitayu Das, 0018-9340/09/$25.00 2009 IEEE Published by the IEEE Computer Society.
[16] "Performance Issues of Heterogeneous Hadoop Clusters in Cloud Computing ",B.Thirumala Rao, N.V.Sridevi, V.Krishna Reddy, L.S.S.Reddy, Global Journal of Computer Science and Technology Volume XI Issue VIII May 2011 .
[17] " Virtual Network Performance Evaluation for Future Internet Architectures" Diogo M. F. Mattos, Lyno Henrique G. Ferraz, JOURNAL OF EMERGING TECHNOLOGIES IN WEB INTELLIGENCE, VOL. 4, NO. 4, NOVEMBER 2012."
[18] Application Performance Management in a Virtualized Environment Growing ", WHITE PAPER: APPLICATION PERFORMANCE MANAGEMENT.
[19] " Better virtualization of XenApp and XenDesktop with XenServer", XenApp and XenDesktop with XenServer White Paper.
[20] " Performance Impact of Virtual Machine Placement in a Datacenter", Indrani Paul, Sudhakar Yalamanchili, Lizy K. John.
[21] " Application Performance Modeling in a Virtualized Environment", Sajib Kundu, Raju Rangaswami, Kaushik Dutta, Ming Zhao, School of Computing & Information Sciences, College of Business Administration Florida International University {skund001, raju}@cs.fiu.edu kaushik.dutta@business.fiu.edu,zhaom@cs.fiu.edu,
[22] " CPI^2: CPU performance isolation for shared compute clusters", Xiao Zhang Eric Tune Robert Hagmann Rohit Jnagal Vrigo Gokhale John Wilkes Google, Inc, 2013 ACM 978-1-4503-1994-2/13/04. $15.00
[23] Enforcing Performance Isolation Across Virtual Machines in Xen", Diwaker Gupta1, Ludmila Cherkasova, Rob Gardner, Amin Vahdat1 Enterprise Software and Systems Laboratory HP Laboratories Palo Alto HPL-2006-77 May 4, 2006*.
[24] " Diagnosing Performance Overheads in the Xen Virtual Machine Environment", Aravind Menon, Jose Renato, Yoshio Turner,G. (John) Janakiraman, Palo Alto john,Willy Zwaenepoel, VEE'05, June 11-12, 2005, Chicago, Illinois, USA. Copyright2005ACM1-59593-047- 7/05/0006...$5.00.
[25] " Scheduling I/O in Virtual Machine Monitors", Diego Ongaro Alan L. Cox Scott Rixner, VEE'08, March 5–7, 2008, Seattle, Washington, USA. Copyright c 2008 ACM 978-1-59593-796-4/08/03...$5.00.
[26] "Xen and the Art of Virtualization", Paul Barham∗, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer†, Ian Pratt, Andrew Warfield, SOSP'03, October 19–22, 2003, Bolton Landing, New York, USA. Copyright 2003 ACM 1-58113-757-5/03/0010 ...$5.00.
[27] "IO Performance Prediction in Consolidated Virtualized Environments", Bhukya, D.P. ; Ramachandram, S. ; Reeta Sony, A.L.
[28] "Experimental Evaluation of the Performance-Influencing Factors of Virtualized Storage Systems", Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on Digital Object Identifier: 10.1109/ICCIC.2010.5705753, Publication Year: 2010 , Page(s): 1 - 4
[29] " Increasing memory density by using KSM", Andrea Arcangeli, Izik Eidus, Chris Wright Red Hat, Inc. aarcange@redhat.com, ieidus@redhat.com, chrisw@redhat.com.

BIOGRAPHY OF AUTHORS

Vedula Venkateswara Rao is a Ph.D Candidate in the Department of Computer Science Engineering at Gitam Institute of Technology, Gitam University, Vishakapatnam, and AndhraPradesh, India. He received Matsers Degree in Computer Science Engineering from JawaharLalNehru Technological University Kakinada, Masters Degreen In Information Technology from Punjabi University, Patiayala, India. His research interests include Cloud Computing and Distributed Systems, Data Mining, Big Data and Image Processing.He published several papers in International conferences and journals.

Dr. Mandapati Venkateswara Rao is Professor in Department of Information Technology at Gitam Institute of Technology, Gitam University, and Vishakapatnam, India. He Has received M.Tech in CST and Ph.D in Robotics from Andhra University. His Research Interests includes Robotics, Cloud Computing and Image processing. He published several papers in International conferences and journals.