Evaluation of Different Hypervisors Performance in the Private Cloud with SIGAR Framework

P. Vijaya Vardhan Reddy
Department of Computer Science & Engineering
University College of Engineering, Osmania University
Hyderabad, India

Dr. Lakshmi Rajamani
Department of Computer Science & Engineering
University College of Engineering, Osmania University
Hyderabad, India

***Abstract*— To make cloud computing model Practical and to have essential characters like rapid elasticity, resource pooling, on demand access and measured service, two prominent technologies are required. One is internet and second important one is virtualization technology. Virtualization Technology plays major role in the success of cloud computing. A virtualization layer which provides an infrastructural support to multiple virtual machines above it by virtualizing hardware resources such as CPU, Memory, Disk and NIC is called a Hypervisor. It is interesting to study how different Hypervisors perform in the Private Cloud. Hypervisors do come in Paravirtualized, Full Virtualized and Hybrid flavors. It is novel idea to compare them in the private cloud environment. This paper conducts different performance tests on three hypervisors XenServer, ESXi and KVM and results are gathered using SIGAR API (System Information Gatherer and Reporter) along with Passmark benchmark suite. In the experiment, CloudStack 4.0.2 (open source cloud computing software) is used to create a private cloud, in which management server is installed on Ubuntu 12.04 – 64 bit operating system. Hypervisors XenServer 6.0, ESXi 4.1 and KVM (Ubuntu 12.04) are installed as hosts in the respective clusters and their performances have been evaluated in detail by using SIGAR Framework, Passmark and NetPerf.**

Keywords—CloudStack; Hypervisor; Management Server; Private Cloud; Virtualization Technology; SIGAR; Passmark

I. INTRODUCTION

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources such as networks, servers, storage, applications, and services that can be rapidly provisioned and released with minimal management effort or service provider interaction [1].

Virtualization, in computing, refers to the act of creating a virtual version of something, including but not limited to a virtual computer hardware platform, operating system, storage device, or computer network resources. Storage virtualization is amalgamation of multiple network storage devices into what appears to be a single storage unit. Server virtualization is partitioning of a physical server into smaller virtual servers. Operating system-level virtualization is a type of server virtualization technology which works at the operating system (kernel) layer. Network virtualization is using network resources through a logical segmentation of a single physical network. Virtualization is the technology which increases the utilization of physical servers and enables portability of virtual servers between physical servers. Virtualization Technology gives the benefit of work load isolation, work load migration and work load consolidation.

For being able to reduce hardware cost, cloud computing uses virtualization. Virtualization technology has evolved really quickly during past few years. Also it is particularly due to hardware progresses made by AMD and Intel. Virtualization is a technology that combines or divides computing resources to present one or many operating environments using methodologies like hardware and software partitioning or aggregation, partial or complete machine simulation, emulation, timesharing, and many others [2]. A virtualization layer provides an infrastructural support using the lower-level resources to create multiple virtual machines that are independent and isolated from each other. Such a virtualization layer is also called Hypervisor. [2].

Cloud computing allows customers to reduce the cost of the hardware by allowing resources on demand. Also customers of the service need to have guaranty of the good functioning of the service provided by the cloud. The Service Level Agreement brokered between the providers of cloud and the customers is the guarantees from the provider that the service will be delivered properly [3].

This paper provides a quantitative comparison of three hypervisors Xen Server 6.0, VMware ESXi Server 4.1 and KVM (Ubuntu 12.04) in the private cloud environment. Microsoft Windows 2008 R2 server is installed on three hypervisors as a guest operating system and a series of performance experiments are conducted on the respective guest OS and results are gathered using SIGAR [36], Passmark [16] and NetPerf [35]. This technical paper presents and analyses the results of these experiments. The discussion in this paper should help both IT decision makers and end users to choose the right virtualization hypervisor for their respective private cloud environments. The experimental results indicate that both XenServer and VMware ESXi Server deliver almost equal and near native performance in all the tests except in CPU test ESXi is performing marginally better than XenServer and in Memory test XenServer performing slightly better than that of ESXi Server. Furthermore, KVM performance is noticeably lower than that of XenServer and ESXi Server, hence it needs to improve in all the performance aspects.

II. VIRTUALIZATION TECHNIQUES

This section describes the different virtualization techniques namely, Full virtualization and Paravirtualization used by different hypervisors.

X86 operating systems are designed to run directly on the bare-metal hardware, so they naturally assume they fully 'own' the computer hardware. The x86 architecture offers four levels of privilege known as Ring 0, 1, 2 and 3 to operating systems and applications to manage access to the computer hardware. While user level applications typically run in Ring 3, the operating system needs to have direct access to the memory and hardware and must execute its privileged instructions in Ring 0. Virtualizing the x86 architecture requires placing a virtualization layer under the operating system (which expects to be in the most privileged Ring 0) to create and manage the virtual machines that deliver shared resources. Three alternative techniques now exist for handling sensitive and privileged instructions to virtualize the x86 Architecture. Full virtualization [17] approach, translates kernel code to replace non-virtualizable instructions with new sequences of instructions that have the intended effect on the virtual hardware. This combination of binary translation and direct execution provides Full virtualization as the guest OS is fully abstracted (completely decoupled) from the underlying hardware by the virtualization layer. The full virtualization approach allows datacenters to run an unmodified guest operating system, thus maintaining the existing investments in operating systems and applications and providing a non-disruptive migration to virtualized environments. VMware ESXi server uses a combination of direct execution and binary translation techniques [4] to achieve full virtualization of an x86 system. Paravirtualization [17], involves modifying the OS kernel to replace non-virtualizable instructions with hyper-calls that communicate directly with the virtualization layer hypervisor. The hypervisor also provides hyper-call interfaces for other critical kernel operations such as memory management, interrupt handling and time keeping. The paravirtualization approach modifies the guest operating system to eliminate the need for binary translation. Therefore it offers potential performance advantages for certain workloads but requires using specially modified operating system kernels [4]. The Xen open source project was designed initially to support paravirtualized operating systems. While it is possible to modify open source operating systems, such as Linux and OpenBSD, it is not possible to modify "closed" source operating systems such as Microsoft Windows. Hardware vendors are rapidly embracing virtualization and developing new features to simplify virtualization techniques. First generation enhancements include Intel Virtualization Technology (VT-x) and AMD's AMD-V which both target privileged instructions with a new CPU execution mode feature that allows the VMM to run in a new root mode below ring 0. The hardware virtualization [17] support enabled by AMD-V and Intel VT technologies introduces virtualization in the x86 processor architecture itself.

III. HYPERVISOR MODELS

All three hypervisors which used in the experiment are discussed from viewpoint of their virtualization technique.

A. Paravirtualized Hypervisor

XenServer - Citrix XenServer is an open-source, complete, managed server virtualization platform built on the powerful Xen Hypervisor. Xen [21] uses para-virtualization. Para-virtualization modifies the guest operating system so that it is aware of being virtualized on a single physical machine with less performance loss. XenServer is a complete virtual infrastructure solution that includes a 64-bit Hypervisor with live migration, full management console, and the tools needed to move applications, desktops, and servers from a physical to a virtual environment [8]. Based on the open source design of Xen, XenServer is a highly reliable, available, and secure virtualization platform that provides near native application performance [8]. Xen usually runs in higher privilege level than the kernels of guest operating systems. It is guaranteed by running Xen in ring 0 and migrating guest operating systems to ring 1. When a guest operating system tries to execute a sensitive privilege instruction (e.g., installing a new page table), the processor will stop and trap it into Xen [9]. In Xen, guest operating systems are responsible for allocating the hardware page table, but they only have the privilege of direct read, and Xen [9] must validate updating the hardware page table. Additionally, guest operating systems can access hardware memory with only non-continuous way because Xen occupies the top 64MB section of every address space to avoid a TLB flush when entering and leaving the Hypervisor [9]. XenServer is a complete virtual infrastructure solution that includes a 64-bit Hypervisor [8].

B. Full virtualized Hypervisor

ESXi Server - VMware ESXi is a Hypervisor aimed at server virtualization environments capable of live migration using VM motion and booting VMs from network attached devices. VMware ESXi supports full virtualization [7]. The Hypervisor handles all the I/O instructions, which necessitates the installation of all the hardware drivers and related software. It implements shadow versions of system structures such as page tables and maintains consistency with the virtual tables by trapping every instruction that attempts to update these structures. Hence, an extra level of mapping is in the page table. The virtual pages are mapped to physical pages throughout the guest operating system's page table [6]. The Hypervisor then translates the physical page (often-called frame) to the machine page, which eventually is the correct page in physical memory.

This helps the ESXi server better manage the overall memory and improve the overall system performance [19]. VMware's proprietary ESXi Hypervisor, in the vSphere cloud-computing platform, provides a host of capabilities not currently available with any other Hypervisors. These capabilities include High Availability (the ability to recover virtual machines quickly in the event of a physical server failure), Distributed Resource Scheduling (automated load balancing across a cluster of ESXi servers), Distributed Power Management (automated decommissioning of unneeded servers during non-peak periods), Fault Tolerance (zero downtime services even in the event of hardware failure), and Site Recovery Manager (the ability to automatically recover virtual environments in a different physical location if an entire datacenter outage occurs) [7].

C. Hybrid methods

KVM - KVM (Kernel-based Virtual Machine) is another open-source Hypervisor using full virtualization apart from VMware. And also as a kernel driver added into Linux, KVM enjoys all advantages of the standard Linux kernel and hardware-assisted virtualization thus depicting hybrid model. KVM introduces virtualization capability by augmenting the traditional kernel and user modes of Linux with a new process mode named guest, which has its own kernel and user modes and answers for code execution of guest operating systems [9]. KVM comprises two components: one is the kernel module and another one is userspace. Kernel module (namely kvm.ko) is a device driver that presents the ability to manage virtual hardware and see the virtualization of memory through a character device /dev/kvm. With /dev/kvm, every virtual machine can have its own address space allocated by the Linux scheduler when being instantiated [9]. The memory mapped for a virtual machine is actually virtual memory mapped into the corresponding process. Translation of memory address from guest to host is supported by a set of page tables. KVM can easily manage guest Operating systems with kill command and /dev/kvm. User-space takes charge of I/O operation‘s virtualization. KVM also provides a mechanism for user-space to inject interrupts into guest operating systems. User-space is a lightly modified QEMU, which exposes a platform virtualization solution to an entire PC environment including disks, graphic adapters and network devices [9]. Any I/O requests of guest operating systems are intercepted and routed into user mode to be emulated by QEMU [9].

IV. Related Work

The following papers are studied to understand about the relevant work which had happened in the selected research area.

Benchmark Overview - vServCon a white paper by FUJITSU [10], scalability measurements of virtualized environments at Fujitsu Technology Solutions are currently accomplished by means of the internal benchmark "vServCon" (based on ideas from Intel‘s "vConsolidate"). The abbreviation "vServCon" stands for: "virtualization enables SERVer CONsolidation. A representative group of application scenarios is selected in the benchmark. It is started simultaneously as a group of VMs on a virtualization host when making a measurement. Each of these VMs is operated with a suitable load tool at a defined lower load level. All known virtualization benchmarks are thus based on a mixed approach of operating system and applications plus an "idle" or "standby" VM, which represents the inactive phases of a virtualization environment and simultaneously increases the number of VMs to be managed by the Hypervisor [10].

The virtualization overhead involves performances depreciation rather to native performances. Research have been made to measure the overhead of the virtualization for different hypervisor such as XEN, KVM and VMware ESX [11]; [12]; [13]; [14]; [15]. For their researches Menon used a toolkit called Xenoprof which is a system wide statistical tool implemented specially for Xen [13]. Due to this toolkit they have managed to analyse the performances of the overhead of network I/O devices. Their study has been performed within uniprocessor as well as multiprocessor. A part of their research has been dedicated to performance debugging of Xen using Xenoprof. Those researches have permitted to correct bugs and improve by that the network performances significantly. After the debugging part it has been focused on the network performances. It has been observed that the performance seems to be almost the same between Xen Domain0 and native performances. However if the number of interfaces increase, the receive throughput of the domain0 is significantly smaller than the native performances. This degradation of network performances is cause by an increasing CPU utilisation. Because of the overhead caused by the virtualization there are more instructions that need to be managed by the CPU. This involves more information to treat and bufferization by the CPU which cause a degradation of receive throughput compared to native performances. More recent studies try to compare the differences between hypervisors and especially the performances of each one according to their overhead [12];[15]. They are using three different benchmark tools to measure the performances: LINPACK, LMbench and Iozone. Their experiment is divided in three parts according to the specific utilisation of each tool. With LINPACK Jianhua had tested the processing efficiency on floating point. Different pick value has been observed over the different systems tested which are native performance, Xen and KVM. The result of this show that the processing efficiency of Xen on floating point is better than KVM because Fedora 8 virtualized with Xen have performances which represent 97.28% of the native rather than Fedora 8 virtualized with KVM represent only 83.46% of the native performances. The virtualization of Windows XP comes up with better performances than with the virtualization of fedora 8 on Xen. This is explained by the authors by the fact that Xen own fewer enhancement packages for windows XP than for fedora 8because of that the performances of virtualized windows XP are slightly better than virtualized fedora 8.

After having testing the processing efficiency with LINPACK, Jianhua have analysed memory virtualization of Xen and KVM compared to native memory performances with LMbench. It has been observed that the memory bandwidth in reading and writing of Xen are really close to native performances. However the performances of KVM are slightly slower for reading but significantly slower concerning the writing performances. The last tool used by Jianhua is IOzone which is used to perform file system benchmark. Once again the native performances are compared to the virtualization performances of Xen and KVM. Without Intel-VT processor the performances of either Xen or KVM are around 6 or 7 times slower than the native performances. However within the Intel-VT processor the performances of Xen increase significantly because the performances are even better than native performances. However KVM does not exploit the functionalities of the Intel-VT processors and because of that does not improve its performances.

After analysing the relevant work on hypervisors performance we have chosen the below experimentation to compare the respective hypervisors in the private cloud environment with CloudStack using SIGAR framework which is a novel idea.

V. Test Methodology - Private Cloud: CloudStack With Hypervisors

In our experiment, the proposed test environment contains following infrastructure using open source cloud computing software. CloudStack is an Infrastructure as a service (IaaS) cloud based software which is able to rapidly build and provide private cloud environments or public cloud services. Supporting KVM, XenServer and Vmware ESXi, CloudStack is able to build cloud environments with a mix of multiple different hypervisors. With rich web interface for users and administrators with operations of cloud use and operation being performed on a browser. Additionally, the architecture is made to be scalable for large-scale environments [22]. CloudStack is open source software written in java that is designed to deploy and manage large networks of virtual machines, as a highly available, scalable cloud computing platform. CloudStack offers three ways to manage cloud computing environments: an easy-to-use web interface, command line and a full-featured RESTful API [22]. Private clouds are deployed behind the firewall of a company where as public cloud is usually deployed over the internet. It is always ideal to use open source solutions to perform any experiment related to cloud computing.

In our test environment XenServer, ESXi and KVM are used as hypervisors (Hosts) in the CloudStack (private cloud). One machine is Management Server, runs on a dedicated server. It controls allocation of virtual machines to hosts and assigns storage and IP addresses to the virtual machine instances. The Management Server runs in a Tomcat container and requires a MySQL database for persistence. In the experiment, Management Server is installed on Ubuntu (12.04 64-bit). On the host servers XenServer 6.0, ESXi 4.1 and KVM (Ubuntu 12.04) [31] hypervisors are installed as depicted in Fig. 1. Front end will be any base machine to launch CloudStack UI using web interface (with any browser software IE, Firefox, Safari) to provision the cloud infrastructure by creating zone, pod, cluster and host in the sequential order. After respective hypervisors are in place, guest OS Windows 2008 R2 64-bit [33] installed on them to carry out all performance tests.

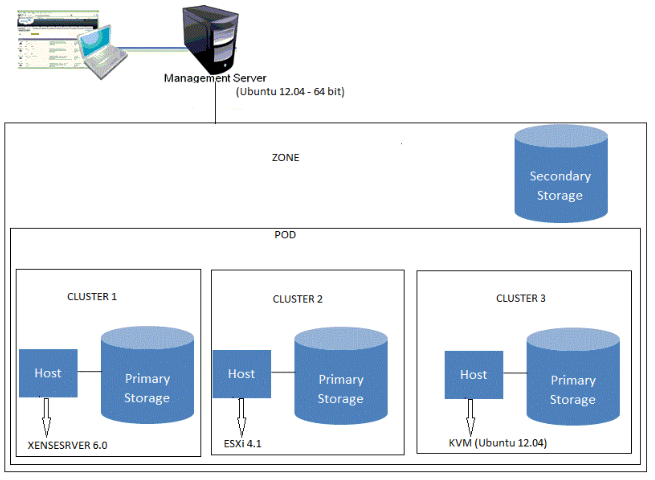

Fig. 1. Test Environment Architecture – Private Cloud (CloudStack with Multiple hypervisors)

A typical enterprise datacenter runs a mix of CPU, memory, and I/O-intensive applications. Hence the test workloads chosen for these experiments comprise several well-known standard benchmark tests. Passmark, a synthetic suite of benchmarks intended to isolate various aspects of workstation performance, was selected to represent desktop-oriented workloads. Disk I/O performance is measured using Passmark. CPU and Memory performance on the guest OS are measured using SIGAR Framework. SIGAR (System Information Gatherer and Reporter) is a cross-platform, cross-language library and command-line tool for accessing operating system and hardware level information in Java, Perl and .Net. In the experiment, Java program has written to gather system information using SIGAR API by deploying sigar-amd64-winnt.dll for Windows. And for network performance Netperf is used in the experiment. Netperf was used to simulate the network usage in a datacenter. The objective of these experiments was to test the performance of the three virtualization hypervisors. The tests were performed using a Windows 2008 R2 64-bit as guest operating system. The benchmark test suites are used in these experiments only to illustrate performance of the three hypervisors.

VI. Results

This section provides the detailed results for each of the benchmarks run. Disk I/O and Network Performance results have been normalized to native performance measures. Native performance is normalized at 1.0 and all other various benchmark results are shown relative to that number. Hence benchmark results of 90% of the native performance would be shown as 0.9 on the scale in the graph. Higher numbers indicate better performance of the particular virtualization platform, unless indicated otherwise. Near-native performance also indicates that more virtual machines can be deployed on a single physical server, resulting in higher consolidation ratios. This can help even if an enterprise plans to standardize on virtual infrastructure for server consolidation alone. CPU utilization tests indicate lower CPU utilization is better for a hypervisor, which is evaluated by using SIGAR API. In case of Memory tests, High available memory indicated better performance of a hypervisor which gathered using SIGAR.

A. SIGAR

CPU utilization on the guest Operating System is captured when it is running on the respective Hypervisor. CPU utilization details are captured through java program using SIGAR API on the guest OS for each hypervisor. As shown in Fig. 2, ESXi for its guest OS shows less utilization of CPU as compared to other hypervisors. Lower utilization CPU indicates the better performance for a hypervisor. XenServer also shows low utilization of CPU for its guest OS but little higher than ESXi hypervisor. On the other hand KVM's CPU utilization is slightly high for its guest OS as compared to other two hypervisors.

Memory performance is evaluated by considering the available memory in the respective hypervisor when the single guest Operating Systems is given full available memory.

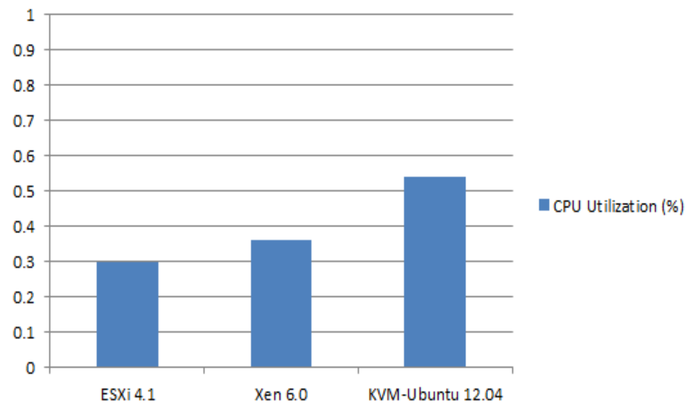

Fig. 2. CPU Utilization captured using SIGAR (Lower value is better)

Fig. 3 shows Available memory on the respective hypervisor when guest OS is running. Memory details are captured using Java program with SIGAR API on the guest OS. XenServer for its guest OS shows maximum available memory as compared to other hypervisors. Higher available memory indicates better performance for a hypervisor. ESXi also exhibits higher available memory only but slightly less compared to XenServer. KVM indicates marginally less available memory compare to other hypervisors.

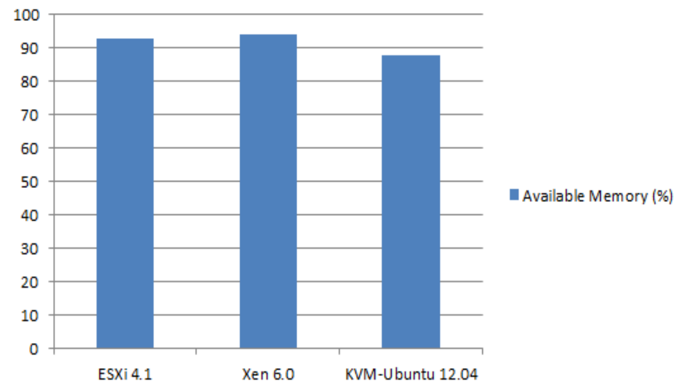

Fig. 3. Available Memory captured using SIGAR (Higher Value is better)

B. PASSMARK

The following Fig. 4 shows benchmark results for Passmark Disk I/O read write tests. Sequential Read and Sequential Write are the disk mark tests which were conducted on the three hypervisors in the private cloud environment. Both XenServer and ESXi perform almost equal to native performance.

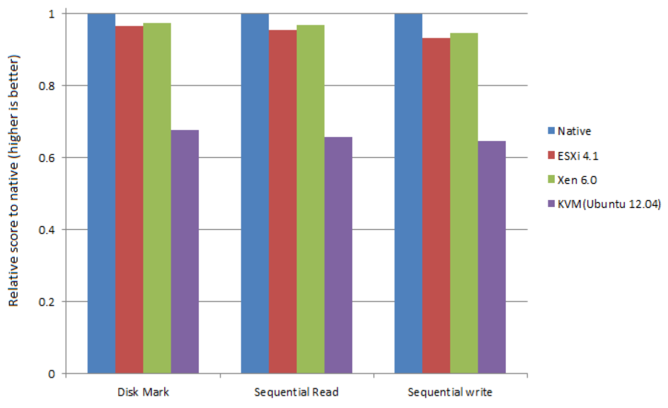

Fig. 4. Passmark – Disk I/O Read Write results compared to native (Higher values are better)

In Sequential Read and Sequential Write XenServer slightly shows better performance than that of VMWare ESXi Server. In overall disk mark performance XenServer shows 2.7% overhead vs native whereas ESXi shows 3.4% overhead vs native. KVM significantly falls behind other two hypervisors and native as well.

C. NETPERF

For experiment, in the private cloud for all the three hypervisors, Netperf test involved running single client communicating with single virtual machine through a dedicated physical Ethernet adapter and port. All tests are based on the Netperf TCP_STREAM test. Fig. 5 shows the Netperf results for send and receive tests. XenServer and ESXi demonstrated near native performance in Netperf test, while KVM lags behind other hypervisors and native.

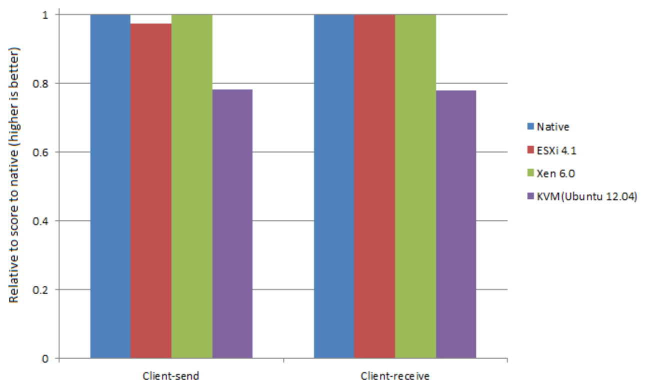

Fig. 5. Netperf results compared to native (higher values are better)

VII. Discussion On Results

Performance results show convincingly that XenServer and ESXi Server both perform equally well in all experiments close to near native performance without showing the signs of any virtualization overhead except KVM falling behind other two hypervisors and native as well.

In CPU utilization tests ESXi CPU utilization is 0.06% less than that of XenServer and 0.24% less than that of KVM thus exhibiting better performance in CPU utilization. In memory tests XenServer available memory is 1% more than that of ESXi Server and 6% more than that of KVM hence showing better memory performance among two other hypervisors. In I/O tests XenServer scores over ESXi and KVM, where XenServer shows 4% overhead in sequential read and 6% overhead in sequential write as compared to native. ESXi shows 5% overhead in sequential read and 7% overhead in sequential write as compared to native. And KVM shows 35% overhead in sequential read and 36% overhead in sequential write as compared to native. In Network performance tests both XenServer and ESXi gives near native performance and KVM falls marginally behind other two hypervisors. In Client-Receive tests both XenServer and ESXi gives performance equal to native and in Client-Send tests XenServer gives equal to native performance but ESXi shows 3% overhead as compared to native. In Client-Send and Client-Receive tests KVM shows 22% overhead as compared to native.

On overall XenServer and ESXi two hypervisors are reliable, affordable and offer the windows or any other guest operating system IT professional a high performance platform for server consolidation for production workloads. KVM needs to improve up on almost all fronts if it has to become on par with other two hypervisors. ESXi and XenServer are matured hypervisors as compare to KVM and their Reliability, Availability and Serviceability (RAS) is significantly higher than that of KVM.

VIII. Conclusion And Future Work

The objective of this experiment is to evaluate the performance of VMWare ESXi Server, XenServer and KVM Hypervisors in the private cloud environment. After evaluation results indicate that XenServer and ESXi hypervisors exhibit impressive performance in comparison with KVM. Virtualization infrastructure should offer certain enterprise readiness capabilities such as maturity, ease of deployment, performance, and reliability. From the test results VMware ESXi Server and XenServer are better equipped to meet the demands of an enterprise datacenter than the KVM hypervisor. And KVM needs significant improvement to become an enterprise ready hypervisor. The series of tests conducted for this paper proves that VMware ESXi Server and XenServer delivers the production-ready performance needed to implement an efficient and responsive datacentre in the private cloud environment.

The performance tests are conducted in the private cloud with 64-bit Windows guest operating system. While evaluating network performance, one client send and receive tests are performed on three hypervisors which are supported by CloudStack private cloud platform. The future work can include multiple client send and receive network tests for hypervisors. Experiments can also be carried out with paravirtualized Linux guest operating system as well. With more workloads scalability tests can be performed with other hypervisors which are not covered in the present experiment. And future work can also consider public cloud environment for experimentation.

References

[1] Mell, P. & Grance, T. (2009) The NIST Definition of Cloud Computing. Version 15, 10-7-09. National Institute of Standards and Technology, Information Technology Laboratory.

[2] Nanda, S., T. Chiueh, ―A Survey on Virtualization Technologies, Technical report, Department of Computer Science, SUNY at Stony Brook, New York, 11794-4400, 2005.

[3] Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I. (2009) "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility". In: Future Generation Computer Systems, Elsevier B. V.

[4] Adams K. and Agesen O. A Comparison of Software and Hardware Techniques for x86 Virtualization. *ASPLOS* October 2006.

[5] AMD. (2005) Amd secure virtual machine architecture reference manual.

[6] Barham, P., B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, Xen and the art of virtualization, Proceedings of the Nineteenth ACM Symposium on Operating systems Principles. ACM Press, New York, 2003, pp. 164–177.

[7] Hostway UK VMware ESXi Cloud Simplified, Comprehensive explanation of the features and benefits of VMware ESXi Hypervisor.

[8] Fujitsu Technology Solutions, DataSheet Citrix XenServer,

[9] Che, J., Q. He, Q. Gao, D. Huang, ―Performance Measuring and Comparing of Virtual Machine Monitors,College of Computer Science, Zhejiang University, Hangzhou 310027, China, IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, 2008.

[10] FUJITSU, Benchmark Overview-vServCon, white paper, March 2010.

[11] Apparao, P. & Makineni, S. & Newell, D.Virtualization (2006) Characterization of network processing overheads in Xen. Technology in Distributed Computing, 2006. VTDC 2006.

[12] Jianhua, C. & Qinming, H. & Qinghua, G. & Dawei, H. (2008) Performance Measuring and Comparing of Virtual Machine Monitors. Embedded and Ubiquitous Computing, 2008. EUC '08.

[13] Menon, A. et Al. (2005) Diagnosing Performance Overheads in the Xen Virtual Machine Environment. Conference on Virtual Execution Environments (VEE'05).

[14] Shan, Z. & Qinfen, H. (2009) Network I/O Path Analysis in the Kernel-based Virtual Machine Environment through Tracing. Information Science and Engineering (ICISE).

[15] VMware (2007) A Performance Comparison of Hypervisors VMware. White paper feb 1, 2007.

[16] Passmark. Performance Test – PC Benchmarking

[17] VMware (2007) Understanding Full Virtualization, Paravirtualization, and Hardware Assist. VMware, white paper nov 10, 2007.

[18] Vallee, G. & Naughton, T. & Engelmann, C. & Ong, H. &Scott, S.L. (2008) System- Level Virtualization for High Performance Computing. Parallel, Distributed and Network-Based Processing, 2008. PDP 2008. 16th Euromicro Conference on. Varia, J. (2008) Cloud Architectures. White Paper of Amazon.

[19] VMware, ―The Architecture of VMware ESXi, white paper, 2007.

[20] Xu, X., F. Zhou, J. W. Y. Jiang, ―Quantifying Performance Properties of Virtual Machines, School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China, International Symposium on Information Science and Engineering, 2008.

[21] Xen,―How does Xen work, Xen Orgnaization

[22] CloudStack – OpenSource Cloud Computing

[23] Greenberg, A & Hamilton, J & Maltz, D. A. & Patel, P. (2009) The Cost of a Cloud: Research Problems in Data Center Networks.

[24] He, Q. & Zhou, S. & Kobler, B. & Duffy, D. & McGlynn, T. (2010) Case study for running HPC applications in public clouds. HPDC '10 Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing.

[25] Karger, P.A. & Safford, D.R. (2008) I/O for Virtual Machine Monitors: Security and Performance Issues. Security & Privacy, IEEE.

[26] King, R. (2008) How Cloud Computing Is Changing the World. CEO guide to technology.

[27] Kloster, J. F. & Kristensen, J. & Mejlholm, A. (2007) A Comparison of Hardware Virtual Machines Versus Native Performance in Xen.

[28] Lui, J. & Huang, W. & Abali, B. & Panda, K. D. (2006) High performance VMM Bypass I/O in virtual machines. USENIX 2006 Annual technical conference refereed paper.

[29] Mollick, E. (2006) Establishing Moore's Law. Annals of the History of Computing, IEEE.

[30] XenSource (2007) A Performance Comparison of Commercial Hypervisors. XenEnterprise vs. ESX Benchmark Results. 2007 XenSource.

[31] Ubuntu 12.04 – Free Operating System

[32] Padala, P. & Zhu, X. & Wang, Z. & Singhal, S. & Shin, K. G. (2007). Performance Evaluation of Virtualization Technologies for Server Consolidation. Enterprise Systems and Software Laboratory, HP Laboratories Palo Alto.

[33] Windows 2008 R2 Server Operating sytesm

[34] Zhao, T. & Ding, Y. & March, V. & Dong, S & See, S. (2009) Research on the Performance of xVM Virtual Machine Based on HPCC. ChinaGrid Annual Conference, 2009. ChinaGrid '09. Fourth.

[35] Network Performance Benchmark Netperf.

[36] Hyperic's System Information Gatherer (SIGAR)

AUTHOR'S PROFILE

Vijaya Vardhan Reddy (First Author): He received his M.Tech degree in Computer Science and Engineering from Osmania University in 2000. For the past 14 years he has been working in the IT industry. He had worked in Tokyo (2001) for CSFB Project and in London (2005-06) for ADP Freedom Payroll Project in Java / J2EE Technologies. Currently he is working as an Assistant Vice President in GE Capital. His research areas include distributed computing, grid computing and cloud computing.

Dr. Lakshmi Rajamani (Second Author): She is a retired professor from Osmania University. She has many papers published in journals across the world. She had served as a HOD for computer science department from 2010 to 2012.