



Vaal University of Technology
Your world to a better future

Virtualization performance in private cloud computing



By

KHATHUTSHELO NICHOLAS THOVHEYI

Student Number: 209106093

A research dissertation submitted in fulfilment for the Degree

MAGISTER TECHNOLOGIAE

In

Information Communication Technology

Faculty of Applied and Computer Sciences

VAAL UNIVERSITY OF TECHNOLOGY

Supervisor: Prof Tranos Zuva

Co-Supervisor: Prof Kazeem Okosun

04 October 2019


Mama, Papa, Sesi, Buti, Makhulu Kuku, Thendo, Bethu, Robane and Ouma

Thank you

Always find a Way

DECLARATION

This dissertation is the result of my own work. It has not been previously submitted, in part or whole, to any university of the institution for any degree, diploma, or other qualification.

Signed: _____

Date: 04 October 2019_____

Khathutshelo Nicholas Thovheyi (209106093)

Vaal University of Technology

ABSTRACT

Virtualization is the main technology that powers today's cloud computing systems. Virtualization provides isolation as well as resource control that enable multiple workloads to run efficiently on a single shared machine and thus allows servers that traditionally require multiple physical machines to be consolidated to a single, cost-effective physical machine using virtual machines or containers. Due to virtual machine techniques, the strategies that improve performance like hardware acceleration, running concurrent virtual machines without the correct proper resource controls not used and correctly configured, the problems of scalability as well as service provisioning (crashing response time, resource contention and functionality or usability) for cloud computing, emanate from the configurations of the virtualized system. Virtualization performance is a critical factor in datacentre and cloud computing service delivery. To evaluate virtualization performance as well as to determine which virtual machine configuration provides effective performance, how to allocate and distribute resources for virtual machine performance equally is critical in this research study. In this study, datacentre purposed servers together with Type 1 (bare metal hypervisors), VMware ESXi 5.5, and Proxmox 5.3 were used to evaluate virtualization performance. The experimental environment was conducted on server Cisco UCS B200 M4 which was the host machine and the virtual environment that is encapsulated within the physical layer which hosts the guest virtual machines consisting of virtual hardware, Guest OSs, and third-party applications. The host server consists of virtual machines with one operating system, CentOS 7 64 bit. For performance evaluation purposes, each guest operating system was configured and allocated the same amount of virtual system resources. Various Workload/benchmarking tools were used for Network, CPU, Memory as well as Disk performance, namely; Iperf, Unibench, Ramspeed, and IOzone, respectively. In the case of IOzone, VMware was more than twice as fast as Proxmox. Although CPU utilization in Proxmox was not noticeably affected, considerably less CPU utilization was observed in VMware. While testing the memory performance with ramspeed, VMware performs 16 to 26% better than Proxmox. In the case of writing, VMware observed 31 to 51% better than Proxmox. In a network, it was observed that the performance on Proxmox was very close to the level of bare metal setup. The results of the performance tests show that the additional operations required by virtualization can be confirmed utilizing test programs. The number of additional operations and their type influence specifically to performance as overhead. In memory and disk areas, where the virtualization

procedure was clear, the test outcomes demonstrate that the measure of overhead is little. Processor and network virtualization, then again, was more perplexing. Hence the overhead is more significant. At the point when the overall performance of a virtual machine running in VMware ESXi Server is contrasted with a conventional system, the virtualization causes approximately an increase of 33% in performance. Because of the difficulty in providing optimal real system configurations, workload/benchmarks could provide close to real application systems for better results. The tests demonstrate that virtualization depends immensely on the host system and the virtualization software. Given the tests, both VMware ESXi Server and Proxmox are capable of providing Optimal performance.

ACKNOWLEDGEMENTS

I thank Professor Tranos Zuva for this dissertation, for the advice and guidance that I have received during the writing process. Mr David Ramasodi, Mr Braam van der Walt, Mr Attie Naude, Mr Carel Bosman and Mr Makalo Motsamai, thank you for all the support and upbringing, I am what I am because of you all.

CONTENTS

1	CHAPTER 1: INTRODUCTION	1
1.1	Background	1
1.2	Research problem	2
1.3	Purpose of the study.....	3
1.4	Research question	3
1.5	Objectives.....	3
1.6	Thesis structure graph	4
1.7	The structure of the dissertation	4
1.8	Chapter summary	5
2	CHAPTER 2: LITERATURE REVIEW	6
2.1	Virtualization.....	6
2.2	Concepts of virtualization	8
2.2.1	<i>Types of virtualization</i>	<i>9</i>
2.2.2	<i>Virtualization benefits.....</i>	<i>9</i>
2.3	Types of hypervisors.....	10
2.3.1	<i>VMware ESXi virtualization technology.....</i>	<i>11</i>
2.3.2	<i>Proxmox</i>	<i>11</i>
2.4	Cloud computing	11
2.4.1	<i>Types of clouds</i>	<i>13</i>
2.5	Cloud computing benefits	14
2.6	Performance of virtualization in private clouds	14
2.7	A combination of methods/framework	34
2.8	Chapter Summary	36
3	CHAPTER 3: METHODOLOGY	37
3.1	Measuring virtualization effects by tests	37
3.1.1	<i>Test types of performance test</i>	<i>38</i>
3.2	Experimental environment	39
3.2.1	<i>Host system.....</i>	<i>39</i>
3.3	Performance.....	41
3.3.1	<i>Network performance.....</i>	<i>41</i>

3.3.2	<i>Disk performance</i>	44
3.3.3	<i>Memory performance</i>	47
3.3.4	<i>CPU performance</i>	49
3.3.5	<i>Measurement procedure</i>	50
3.3.6	<i>Validation</i>	51
3.3.7	<i>Test cases in a virtualized scenario (CPU)</i>	51
3.3.8	<i>VMware</i>	52
3.3.9	<i>Proxmox</i>	53
3.3.10	<i>Summary</i>	54
4	CHAPTER 4: EXPERIMENTS, RESULTS AND ANALYSYS	55
4.1	Configurations for hardware and software used.....	55
4.2	Network.....	55
4.2.1	<i>Analysis of network performance</i>	55
4.2.2	<i>Network conclusion</i>	61
4.3	Disk	62
4.3.1	<i>Analysis of disk performance</i>	63
4.3.2	<i>Disk conclusion</i>	69
4.4	Cpu.....	69
4.4.1	<i>Vmware</i>	70
4.4.2	<i>Proxmox</i>	71
4.4.3	<i>Result summary</i>	72
4.4.4	<i>CPU conclusion</i>	74
4.5	Memory.....	75
4.5.1	<i>Ram speed</i>	75
4.5.2	<i>Integer and writing</i>	75
4.5.3	<i>Integer and reading</i>	76
4.5.4	<i>Float and writing</i>	76
4.5.5	<i>Float and reading</i>	77
4.5.6	<i>RamSpeed test results in discussion</i>	77
4.5.7	<i>Overall ram speed results</i>	77
4.5.8	<i>Memory conclusion</i>	79
4.6	Chapter summary.....	80
5	CHAPTER 5: CONCLUSION AND RECOMMENDATIONS	81
5.1	Discussion	81

5.2	Future work	82
5.3	Conclusion.....	83
5.4	Recommendations.....	87

LIST OF TABLES

TABLE 1: METHODS / FRAMEWORK	35
TABLE 2: RESOURCE DISTRIBUTION PLAN	41
TABLE 3: AVAILABLE TOOLS FOR NETWORK PERFORMANCE.....	42
TABLE 4: TOOLS AVAILABLE FOR DISK PERFORMANCE.....	44
TABLE 5: MEMORY PERFORMANCE TOOLS	47
TABLE 6: LIST OF AVAILABLE CPU PERFORMANCE TOOLS	50
TABLE 7: VMWARE - TEST CASES FOR THE 16 CORE SETUP.....	52
TABLE 8: VMWARE- TEST CASES FOR THE 12 CORE SETUP.....	52
TABLE 9: VMWARE- TEST CASES FOR THE 6 CORE SETUP	52
TABLE 10: PROXMOX- TEST CASES FOR THE 16 CORE SETUP	53
TABLE 11: PROXMOX- TEST CASES FOR THE 12 CORE SETUP	53
TABLE 12: PROXMOX- TEST CASES FOR 6 CORE SETUP.....	53
TABLE 13: HARDWARE CONFIGURATIONS.....	55
TABLE 14: SOFTWARE CONFIGURATIONS.....	55
TABLE 15: WRITE PERFORMANCE TESTS	67
TABLE 16: READ PERFORMANCE TESTS	68
TABLE 17: VMWARE- CPU UTILIZATION OVERVIEW	73
TABLE 18:PROXMOX- CPU UTILIZATION OVERVIEW	73
TABLE 19: VMWARE- RESPONSE TIME OVERVIEW.....	73
TABLE 20: PROXMOX- RESPONSE TIME OVERVIEW	74
TABLE 21: INTEGER AND FLOAT WRITING RESULTS	78
TABLE 22: INTEGER AND FLOAT READING RESULTS.....	79

LIST OF FIGURES

FIGURE 1: THESIS STRUCTURE.....	4
FIGURE 2:VMWARE ESXI SERVER ARCHITECTURE	40
FIGURE 3:PROXMOX SERVER ARCHITECTURE.....	40
FIGURE 4: THE COMPARISON OF TCP BANDWIDTH BETWEEN VMWARE AND PROXMOX	56
FIGURE 5: THE COMPARISON OF TCP THROUGHPUT AMONG VMWARE AND PROXMOX.	57
FIGURE 6: THE COMPARISON OF UDP BANDWIDTH AMONG VMWARE AND PROXMOX.....	57
FIGURE 7: THE COMPARISON OF UDP THROUGHPUT AMONG VMWARE AND PROXMOX.....	58
FIGURE 8: THE COMPARISON OF DATAGRAM LOSS BETWEEN PROXMOX AND VMWARE.	59
FIGURE 9: THE COMPARISON OF JITTER BETWEEN VMWARE AND PROXMOX	60
FIGURE 10: THE COMPARISON OF A MAXIMUM NUMBER OF THE REQUESTS SENT BY CLIENTS BETWEEN VMWARE AND PROXMOX.....	60
FIGURE 11: IOZONE AVERAGE WRITE	63
FIGURE 12: IOZONE AVERAGE RE-WRITE	63
FIGURE 13: IOZONE AVERAGE READ.....	64
FIGURE 14: IOZONE AVERAGE RE-READ	64
FIGURE 15: IOZONE AVERAGE RANDOM READ	65
FIGURE 16: IOZONE AVERAGE RANDOM WRITE	66
FIGURE 17: WRITE PERFORMANCE.....	67
FIGURE 18: CONSOLIDATED READ PERFORMANCE	69
FIGURE 19: VMWARE - CPU UTILIZATION	70
FIGURE 20: VMWARE - AVERAGE RESPONSE TIME	71
FIGURE 21:PROXMOX - CPU UTILIZATION	72
FIGURE 22: PROXMOX - AVERAGE RESPONSE TIME.....	72
FIGURE 23: RAM SPEED AVERAGE INTEGER AND WRITING.....	75
FIGURE 24: RAMSPEED AVERAGE FLOAT AND WRITING.....	76

FIGURE 25:: FLOAT AND READING AVERAGE FOR RAMSPEED	77
FIGURE 26: INTEGER AND FLOAT WRITING	78
FIGURE 27: INTEGER AND FLOAT READING	79

LIST OF ABBREVIATIONS AND ACRONYMS

I/O	Input/output
OS	Operating System
CPU	Central Processing Unit
RAM	Random Access Memory
VM	Virtual Machine
PC	Personal Computer
P2V	Physical-to-Virtual
VMM	Virtual Machine Monitor
IT	Information Technology
SAAS	Software as a Service
PAAS	Platform as a Service
IAAS	Infrastructure as a Service
HPC	High Performance Computing
QoS	Quality of Service
VT	Virtualization Technology
ARM	Advanced RISC Machine
KVM	Kernel-based Virtual Machine

LIST OF DEFINITIONS

Virtual Machine - Is a software computer like a real physical machine that runs applications and operating systems, referred to as a guest machine.

Hypervisor - A thin kernel layer that abstracts the physical hardware and presents virtual hardware to the guests.

Virtual Machine Monitor - a layer of software that runs between a hypervisor or host operating system and one or more virtual machines that provide the virtual machine abstraction, This becomes the management portal of the virtualized machine.

Guest Operating System - An operating system is running in a virtual machine environment that normally runs directly on a physical system.

Virtual Resource - A physical resource, (for example, Disk, CPU, or memory) that is overseen by a hypervisor and allocated to a guest. Virtualized Changing a physical system to a virtual guest system.

Overcommit - when more resources are assigned than are physically available. System-wide profiling Both the guest and VMM are profiled.

Paravirtualization - A virtualization system where the native instruction set is not implemented. Generally, the Guest OS should be altered to realize it is virtualized. This is the procedure utilized by Xen.

Overhead - The extra cost (time) required for virtualization. For every task, there might be extra resources required to virtualize the activity as operation t instead of interacting directly with the hardware.

Snapshot- A complete state of the entire virtual machine saved to non-volatile disk for later

Virtualization – “Virtualization is a technology that joins or partitions computing resources to exhibit one or many operating environments utilizing methodologies like hardware and partitioning or aggregation, partial or complete machine simulation, emulation, timesharing, and numerous others.”

Cloud computing - is shared pools of configurable computer system resources and more elevated services that can be quickly provisioned with minimal administration effort, regularly over the Internet. Cloud computing depends on sharing of resources to accomplish cognizance and economies of scale, like a public utility.

Jitter - is the amount of variation in latency/response time, in milliseconds.

CHAPTER 1

INTRODUCTION

1 INTRODUCTION

1.1 BACKGROUND

Virtualization is the main technology that powers today's cloud computing systems. Virtualization provides isolation and resource control that enables multiple workloads to run efficiently on a single shared machine and thus allows servers that traditionally require multiple physical machines to be consolidated to a single, cost-effective physical machine using virtual machines or containers (Enberg, 2016).

Server virtualization opens the present traditional balanced design of x86 servers by abstracting the operating system and applications from the physical hardware, empowering a more cost-effective, coordinated, and improved server condition. Utilizing server virtualization, numerous operating systems can keep running on a single physical server as virtual machines, each with access to the fundamental server's computing resources. Most servers work at less than 15 percent capacity; not only is this highly inefficient, it also introduces server sprawl and complexity. Server virtualization addresses these inefficiencies(Sligh and Owusu, 2014).

With virtualization, all software, drivers, and the operating system are put away on servers, as opposed to being installed on each client 's machine. This implies that extensive quantities of clients can be managed centrally, with all client data and information kept up in the datacentre. This reduces the time and costs necessary for administering a massive infrastructure. The users' environment remains unchanged; they get the same experience as using a standard PC. The Information Technology(IT) manager, however, experiences greater efficiency. For instance, if new software is required for 50 clients, the IT administrator just needs to install the software once centrally and afterwards activate it for the different virtual machines, as opposed to installing it on multiple machines on an individual basis. Henceforth, the desktop virtualization model enables organizations to accomplish significant savings of both time and money while dealing with their IT infrastructure(A Vouk, 2008).

Virtualization is likewise intended to improve the manageability of the enterprise infrastructure. As virtual servers and desktops can be live-migrated with no downtime, organizing hardware upgrades with clients or arranging work windows is required so that in

future, vital upgrades can occur whenever necessary, with no effect to the client. Additionally, high availability and dynamic load-balancing solutions given by virtualization product families can monitor and optimize the virtualized environment with a minimal manual contribution. Supporting similar capacities in a non-virtualized world would require much operational effort. Moreover, enterprises utilize virtualization to give Infrastructure as a Service(IaaS) cloud offerings that give clients access to computing resources on demand in the form of virtual machines. This can enhance developer productivity and lessen the time to showcase that is key in today's fast-moving business environment. Since rolling out an application sooner can give a first-mover advantage, virtualization can help support the business(Morabito, 2017).

Creating an efficient, responsive IT environment by virtualizing, the datacentres will be able to reduce datacentre footprint by consolidating physical servers, storage, and networking hardware. You can also improve asset utilization, lower capital and power, and cooling costs, reduce management touch points, accelerate IT service delivery, increase scalability and flexibility, increase redundancy and reliability, extend hardware lifecycles and improve support efficiency(Li et al., 2017).

In this research, we focused on virtualized computers as well as server infrastructure, which is the foundation of a private cloud. We also compared resource utilization through computer systems performance stress tools in order to measure how the virtual machines handled workloads.

1.2 RESEARCH PROBLEM

More industries have moved to virtualization technology, and this has become the most dominant way in which datacentres and cloud computing are built(Babu et al., 2014). Due to the high demand for computerized infrastructure, virtualization performance is a critical factor in datacentre and cloud computing service delivery(Deshane et al., 2008). Virtual machines consist of many components and techniques that can hinder the performance of private clouds. If virtualized systems are not configured correctly such as hardware acceleration, running concurrent virtual machines without the correct and proper resource controls used, private clouds will have problems of scalability and service provisioning (crashing response time, resource contention and functionality or usability)(Matthews et al., 2007). This impacts on user experience(Somani and Chaudhary, 2009b).

1.3 PURPOSE OF THE STUDY

The purpose of this study is to evaluate the performance of virtualization and to determine which virtual machine configuration provides effective performance.

1.4 RESEARCH QUESTION

- What configuration of virtualized systems provides an effective performance?

In order to answer the main question, the following sub-questions will be answered:

- What has been done in the literature to effectively measure the performance of a virtual environment?
- How to set a virtualized environment in order to test different performance configurations?
- How to measure the performance of different configurations in a virtualized environment?

1.5 OBJECTIVES

- To study existing literature to measure the performance of a virtual environment effectively.
- To develop a virtualized environment in order to test different configurations.
- To evaluate the performance of different configurations in a virtualized environment.

1.6 THESIS STRUCTURE GRAPH

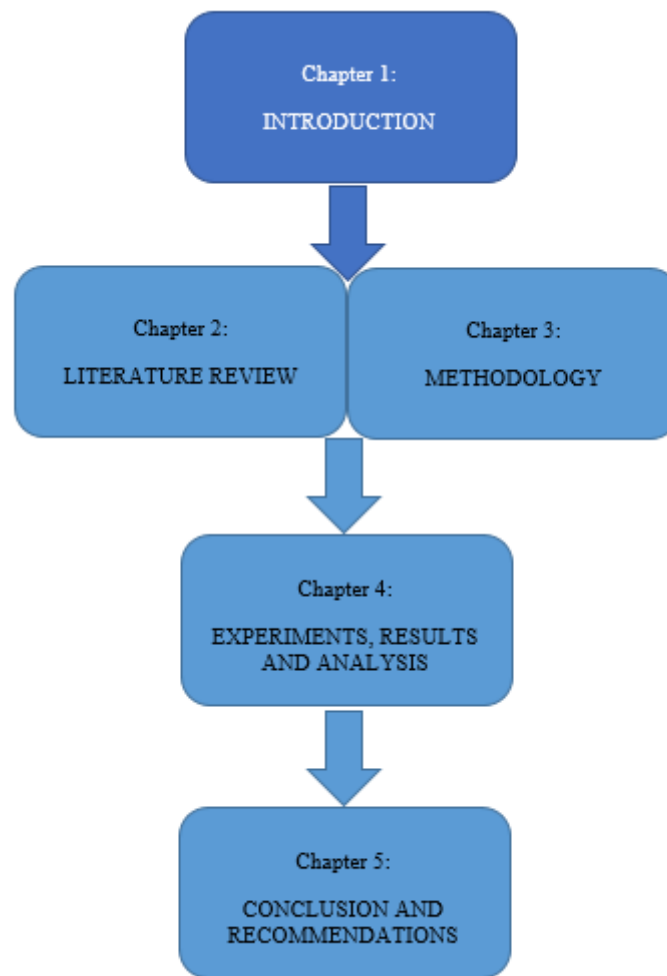


Figure 1: Thesis Structure

1.7 THE STRUCTURE OF THE DISSERTATION:

Chapter 1 Introduction: This is the chapter where the Project is introduced, the research problem, the purpose of the study, research question, and objectives; of the research are outlined. The focus area of the study is explained.

Chapter 2 Literature Review: This chapter focuses on the background of virtualization, cloud computing, and related work done by other researches.

Chapter 3 Methodology: This chapter focuses on related techniques that are relevant to tackling this problem.

Chapter 4 Experiments, Results, and Analysis: This chapter focuses on experimentation and results for the research

Chapter 5 Conclusion and Recommendations: This chapter focuses on other alternatives that can be used and a summary of the entire research as well as the outcomes of the research.

1.8 CHAPTER SUMMARY

Chapter 1 serves as a basis for this research. The research problem, the purpose of the study, the research question, objectives, and thesis structure are described in this chapter. A brief discussion of what is virtualization is explained and how it is vital for cloud computing. The research problem elaborates how virtualization performance is a critical factor in datacentre and cloud computing service delivery. In the following chapter, virtualization technologies, their usage, and current research in the field of virtualization and cloud computing are thoroughly covered and explained. Review on the previous work by other authors is outlined from the leading concept that associate to virtualization and cloud computing in the technical review.

CHAPTER 2

LITERATURE REVIEW

2 INTRODUCTION

We will look into previous work conducted on virtualization performance on clouds.

Previous work done by other authors will provide a clear explanation of virtualization and cloud computing.

Performance investigations of virtualization methods in cloud computing environments are done for various reasons. To begin with, numerous parts of the performance should be looked at, for example, networking, Central Processing Unit(CPU) use, and the disc I/O speeds, and that is just the beginning. Secondly, there is seldom an ideal approach to workload/benchmark these computing assignments. Thirdly, the different variety of software, for example, operating systems and core applications, lends many outcomes flawed or uncertain. There are a wide range of hypervisors, cloud suppliers, operating systems, and workload/benchmark software suites to look over. Dependent on which hypervisor is considered, there are specific hardware and software requirements that must be adhered to. Without considering the more significant part of the potential choices, a performance study may feel deficient. Regardless of these difficulties, it is vital, both for the advance of cloud computing and for the validation of procedures, that these tests exist(Rao and Rao, 2015).

2.1 VIRTUALIZATION

Chiueh and Brook (2005) define Virtualization “as a technology that combines or divides computing resources to present one or many operating environments utilizing methodologies like hardware and software partitioning or aggregation, partial or complete machine simulation, emulation, timesharing, and many others.”

Isolated environments over hardware for application operating system applications are provided by virtualization. Virtualization has opportunity advantages from hardware resources that are closed to physical machines as users want to get maximum utilization from the hardware. (Smith and Nair, 2005) explain that virtualization refers to “abstraction of logical resources away from their underlying physical resources”. Virtualization plays a significant

role in cloud computing since it provides many advantages in sharing, management, and isolation of the resources in the cloud (Rimal et al., 2009).

A system that has physical hardware capable of running many virtual machines concurrently is known as a virtualization host, whereas virtual machines running on it are called guests. This virtualization system usually consists of underlying hardware like network cards, CPUs, hard disk drives, and memory (Ali and Meghanathan, 2011).

Virtual Machine Monitor (VMM) is software that manages the usage of hardware resources and the concurrent running of virtual machines (Lee and Brooks, 2006). The hypervisor makes it possible to run multiple virtual machines at the same time on the hardware resources (Hauswirth et al., 2005). The hypervisor, which works between different operating systems and system resources is responsible for managing multiple virtual machines that are competing for resources such as memory, CPU, network, and data.

Type 1 hypervisors, like Xen, include a different hypervisor software component, which runs correctly on the hardware and gives a virtual machine abstraction to VMs running on the hypervisor. Type 2 hypervisors, like Kernel-based Virtual Machine (KVM), run a current OS on the hardware and run both VMs and applications over the OS. Type 2 hypervisors regularly alter the current OS to facilitate the running of VMs, either by incorporating the Virtual Machine Monitor (VMM) into the current OS source code base or by installing the VMM as drivers into the OS. KVM incorporates explicitly with Linux where other solutions, for example, VMware Workstation, utilize a loadable driver in the current OS kernel to monitor virtual machines. The OS incorporated with a Type 2 hypervisor is usually referred to as the host OS, rather than the guest OS which keeps running in a Virtual Machine M. One advantage of Type 2 hypervisors over Type 1 hypervisors is the reuse of existing OS code, particularly device drivers for an extensive variety of accessible hardware. This is particularly valid for server systems with PCI where any commercially accessible PCI connector can be utilized. A Type 1 hypervisor experiences are having to re-implement device drivers for all upheld hardware (Morabito et al., 2015).

Notwithstanding, Xen, a Type 1 hypervisor, maintains a strategic distance from this by just executing a negligible measure of hardware support directly in the hypervisor and running a particular privileged VM, Dom0, which runs a current OS, for example, Linux and uses all the existing device drivers for that OS. Xen at that point utilizes Dom0 to perform I/O utilizing

existing device drivers in the interest of typical VMs, otherwise called DomUs (Dall et al., 2016)

Nevertheless, to move applications from physical machines to virtualized consolidated platforms, one should have the capacity to assess the performance these applications will accomplish in the new environment. Will moved applications keep running with competitive performance as they keep running on their immediate environment? What number of servers will be expected to make a virtual environment ready to help the execution of the services given, with acceptable performance? What is the best configuration of resources in the virtual environment for a specific application? In this way, there is a current requirement for new tools for anticipating performance, giving data to resource allocation, and determining optimal system configuration (Benevenuto et al., 2006).

Performance models (Benevenuto et al., 2006) help foresee the values of performance measures of a system from an arrangement of values of workload, operating system, and hardware parameters. Performance expectation is the way towards assessing performance measures of a computer system for a given arrangement of parameters. Typical performance measures incorporate reaction time, throughput, resource use, and resource queue length. The input parameters to such a model fall into one of three categories: workload, necessary software, and hardware parameters. The workload parameters describe the load imposed on the system of interest by the applications, i.e., the transactions submitted to it. The software parameters describe features of the necessary software, such as the Xen virtual machine monitor overhead. Examples of such parameters are virtualization overhead, CPU dispatching priority, etc. Examples of hardware performance parameters include the components of the servers that support a Xen system, for example, processor speeds, disk latencies, and transfer rates, and local area network speed. The output of a performance model is a set of performance measures, for example, reaction times, throughput, and resource utilization.

2.2 CONCEPTS OF VIRTUALIZATION

The software abstraction among hardware and the operating system is called virtualization. Every one of the applications is kept running in the operating system that is running over the abstraction layer, additionally called the virtual machine monitor or hypervisor (Marinescu and Kröger, 2007). The hypervisor is used to hide the hardware system resources from the operating system that allows running different operating systems at the same time as the hardware is not directly accessible by the operating system. Further, (Marinescu and Kröger, 2007) explains

that available hardware is logically divided into some logical units that each called virtual machine.

2.2.1 Types of Virtualization

- Full virtualization: Full virtualization is based on the emulation of the hardware. In this approach, the guest operating systems do not require any modification since they are not aware of being virtualized. It provides security and isolation for virtual machines and also facilitates migration and portability (Mahjoub et al., 2011).
- Paravirtualization: In paravirtualization, the guest OSes are aware of the hypervisor, and the OS kernel is modified to provide an interface for communication between the hypervisor and the OS kernel, which improves performance and efficiency. Compatibility and portability of para-virtualization are weak since it does not support unmodified OSes (Li et al., 2010).
- Hardware-assisted virtualization: Hardware-assisted virtualization uses virtualization hardware extensions, mainly host CPU, to provide full virtualization. Consequently, it needs explicit support in the host CPU, which is not available in all processors. Intel VT and AMD-V processors include virtualization technology support. Guests that are using this technique are usually slower than the guests who are using para-virtualization due to a high CPU overhead caused by emulation. However, hardware-assisted virtualization does not require modification of the guest OS (Matthews et al., 2007).

2.2.2 Virtualization Benefits

Some of the significant benefits of using virtualization are discussed below (Younge et al., 2011; Che et al., 2008; Lombardi and Di Pietro, 2011; (Yaqub, 2012) :

- One can support multiple operating systems and virtual machines on a single pane of glass.
- Performance measurement and debugging of virtual machines are possible using the hypervisor management console.
- Through environment isolation, Research academics can run experiments without the fear of breaking the system. This creates great testing and safe environment.

- It provides a safe test environment before taking application servers to the production environment.
- One can create multiple virtual machines from one physical machine through the resource sharing capabilities of virtualization.
- Through resource consolidation, virtualization becomes cost effective as it reduces the requirements for hardware.
- For secure computing, one can isolate virtual machines which have untrusted applications and keeping them separate from the rest of the virtual machines in the same cluster.
- Consolidating workloads is possible with virtual machines .on a single server Virtualization can be utilized to consolidate workload, which has advantages in terms of hardware and software costs as well as management and server infrastructure administration.
- Operating system new feature testing on a virtual machine before actual implementation.
- It is faster and easier to perform backup and recovery as well as the migration of virtual machines.
- Dynamic resource provisioning makes it feasible due to virtualized resources utilization in clustered environments.
- The pay-per-use model that is used in cloud computing allows users to pay for the resources they use, which makes it virtualization flexible and scalable.
- virtual machine migration from one host or cluster to another is effortless.

2.3 TYPES OF HYPERVISORS

Type 1: Native or Bare-Metal Hypervisor: This type of hypervisor runs directly on the host's hardware. Guest OS can be installed on top of this hypervisor. Such hypervisors have lesser memory footprint as compared to Type 2 hypervisor. Examples of the bare metal hypervisor are Citrix XenServer, VMware ESX, and Hyper-V.

Type 2: Hosted Hypervisor: This type of hypervisor requires a base OS that acts as a host. Such hypervisors abstract the presence of host from the guest OS. They may use hardware support for virtualization or can emulate the sensitive instructions using binary translation. Examples of the hosted hypervisor are VMware Workstation, VirtualBox, and KVM.

2.3.1 VMware ESXi Virtualization Technology

Virtualization is an innovation that is expanding by each passing day in the IT industry due to its number of advantages, higher usage of costly hardware, enhanced security, ease of administration, and enhanced data integrity. World's biggest provider of virtualization software, platforms, and tools are provided by VMware, and their products are widely utilized across numerous industries. The ESXi is the most advanced hypervisor design of VMware. VMware Inc. ESXi is a Bare Metal hypervisor and installed over the physical machine. ESXi was introduced in 2007 by VMware with conveying industry-driving performance and adaptability while setting another bar for reliability, security, and hypervisor administration effectiveness (Elsayed and Abdelbaki, 2013).

2.3.2 Proxmox

Proxmox is a Linux distribution based on Debian (64 bits) that carries OpenVZ and KVM. Proxmox allows performing centralized management of many physical servers. Proxmox at least consists of a single master and node (Ali, 2015).

2.4 CLOUD COMPUTING

Cloud computing is a service-oriented model, and abstraction and accessibility are two crucial factors in this model. The underlying cloud architecture is abstracted and hidden from the user as a result of virtualization and consolidation. Concurrently, the key components of underlying cloud architecture can be easily accessed. In general, cloud computing transparently delivers the following services (Wang et al., 2010); (Gong et al., 2010):

- Software-as-a-service;
- Hardware-as-a-service;
- Data-as-a-service; and
- Platform-as-a-service.

Cloud Computing technology is an on-demand service which provides optimal resources allocation and dynamic computing infrastructure which has the hardware, network, storage,

and interfaces that enable the delivery of computing as a service. These services in the cloud include the software as a service, infrastructure as a service, and storage as service over the internet based on user demand (Kumar and Singh, 2015). 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 (Younge et al., 2011).

For Cloud computing to be possible, underlying services, technologies, and configurations exist, and one of that technology is virtualization. Virtualization is a mechanism of hardware and system resource abstraction. This is typically performed in Cloud environments running on clustered hypervisors servers. In this environment, multiple virtual machines can concurrently run, which is one of the critical advantages of Cloud computing. This allows resources consolidation within data centres. From the hypervisor level, Cloud computing systems depend on the virtualization technologies to maintaining QoS and utility to users while reaching optimal performance (Younge et al., 2011).

The internet and virtualization technology is required to make cloud computing have such capabilities such as on-demand access and server or service provisioning. The software that does virtualize in hardware resources such as CPU, Memory, Disk, and NIC and by providing infrastructural support to multiple virtual machines is called a Hypervisor. It is essential to understand different hypervisor performance in private Clouds. Hypervisors come in 3 primary forms, which are Full Virtualization Paravirtualization and Hybrid virtualization. Comparing these in private clouds for performance measurements is essential (Reddy and Rajamani, 2014).

Through resources on demand, this allows customers to have cost-effective, high-quality servers, and applications and services with excellent high performance that they need in the cloud (Reddy and Rajamani, 2014).

Virtualization is fundamental to cloud computing. It allows abstraction centred on services and isolation of lower level functionalities and underlying hardware. Modelling, analyzing, and verifying cloud systems necessarily involve virtualization and services. However, there exist few efforts to effectively formalizing virtualization in cloud computing (Li et al., 2013). Cloud computing is becoming a prominent distributed computing framework as the number of

systems and servers moving to the cloud increase. It is essential to test the performance effects of virtualization in cloud computing to provide quality service to customers. Services such as storage, private, and hybrid clouds are offered in cloud computing but they are typically offered as a service (PaaS) in the forms of infrastructure (IaaS), platform (PaaS), and software (SaaS). Comprehensive computing capabilities by virtualizing processing, storage, and network resources are provided in cloud environments (Kumar and Singh, 2015).

Online interactive systems like Facebook, Wikipedia, Twitter, and many others employ cloud infrastructures to meet user demands. Large scale scientific simulations and high-performance computing (HPC) Computing draw attention by the benefits of Cloud outsourced. It is enticing to have computing resources that are endless, scalable, and compatible with different systems and applications. The term cloud computing often refers to computing distribution, hardware, and software encapsulating in an overall system. Thus, the cloud represents the fuzzy notion of networked computing resources. Users can provision computing resources depending on the demand or task. Multi-node computing infrastructures such as clusters and grids have been around for many years (Overby, 2014).

2.4.1 Types of clouds

- Private cloud: Private cloud service model is hosted within the organization. It can be hosted internally or externally. This type of cloud service is more expensive since it requires more involvement for the organization to virtualize the business environment(Luo et al.).
- Public cloud: Public cloud is a cloud which is made available to the general public. Its customers share the same infrastructure pool, which makes this type of cloud very vulnerable and insecure. Public clouds operate on a low-cost or pay-per-use model. Public clouds can be accessed only via the Internet(Gong et al., 2010).
- Community cloud: Community cloud service model is shared between several organizations within the same community. Its primary purpose is to bring the benefits of the public cloud with an added level of security of the private cloud. Community clouds can be either on-premise (local) or off-premise (remote)(Zhang et al., 2010).

- Hybrid cloud: Hybrid cloud is a combination of two or more clouds (private, community, or public) that are handled as unique entities but are bound together (Mell and Grance, 2009).

2.5 CLOUD COMPUTING BENEFITS

In Cloud computing, users are able to migrate their data and servers to remote locations and used as a disaster recovery site. This provides some benefits which could not otherwise be achieved (Younge et al., 2011).

Such benefits include:

- Cost-Effectiveness - only pay for the needed infrastructure while maintaining the option to increase services as needed in the future.
- Scalability - Clouds enough computing power as required by the user and lowers dependence on specialized hardware.
- Simplified Access Interfaces - a vast amount of computing are easy to access through options like web portals and client consoles.
- Quality of Service (QoS) - A well-designed Cloud provides higher QoS than traditionally possible with advanced computing.
- Customization - Within a Cloud, a user can customize their environment to need their requirements and needs, such as backward compatibility.

2.6 PERFORMANCE OF VIRTUALIZATION IN PRIVATE CLOUDS

Reddy and Rajamani (2014) evaluated and provided quantitative comparison regarding the performance of three hypervisors ESXi, XenServer, and KVM, utilizing SIGAR structure for framework information and Passmark for system workloads in the private cloud condition. (Reddy and Rajamani, 2014) designed a private cloud utilizing open source cloud computing software CloudStack. Hypervisors are conveyed as hosts in the CloudStack. They suggested best-suited hypervisors for respective workloads in the private cloud based on the performance of system information and system workloads. In the test, CloudStack 4.0.2 (open source cloud computing programming) is utilized to make a private cloud, in which administration server was introduced on Ubuntu 12.04 – 64-bit operating system. Hypervisors as XenServer 6.0,

ESXi 4.1 and KVM (Ubuntu 12.04) were installed as hosts in the separate bunches, and their exhibitions have been assessed in detail by utilizing SIGAR Framework, Passmark and NetPerf.

The tests were performed utilizing a Windows 2008 R2 64-bit as a guest operating system. Moreover, the workload/benchmark mentioned above test suits were used in the experiments. On general XenServer and ESXi, two hypervisors are dependable, reasonable and offer the windows or some other guest operating system IT proficient an elite stage for server solidification for production workloads (Reddy and Rajamani, 2014). KVM needs to enhance on all fronts if it needs to wind up keeping pace with the other two hypervisors. ESXi and XenServer have developed hypervisors as a contrast with KVM, and their Reliability, Availability, and Serviceability (RAS) is altogether higher than that of KVM. The series of tests conducted on CPU, Memory and Network performance for the paper demonstrates that VMware ESXi Server and XenServer conveys the production-ready performance expected to actualize a proficient and responsive datacentre in the private cloud condition. (Reddy and Rajamani, 2014), advised that for future work, one can include multiple clients send and receive network tests for hypervisors. It was furthermore suggested that the experiment could likewise be done with para-virtualized Linux guest operating system. With more workloads, adaptability tests can be performed with different hypervisors which are not canvassed in the present trial. Furthermore, future work can likewise consider the open public cloud for experimentation.

The primary target of this investigation is to workload/benchmark the performance of 32bit Debian 6.0 virtual machines running on Xen and VMware ESXi. The workload/benchmark tests will endeavour to quantify the performance of virtual machines concerning network activity, file system I/O, CPU, and memory performance. Its outcomes were utilized as a pattern when contrasting the two hypervisors. Tests were completed on network activity, file system I/O, CPU, and memory performance (Perera and Keppitiyagama, 2011).

Memory workload/benchmarking was finished utilizing Read, Write, and Number-crunching operations on Integers and Floating point numbers. An open source device RAMSpeed/SMP was utilized. Network activity workload/benchmark was finished by measuring Unidirectional TCP/UDP information exchange throughput between the test machines. Netperf, an open source instrument, was used. Amid the parameter, the CPU usage of the sending virtual machine was observed. The information exchange latency likewise measured and looked at between two hypervisors. To workload/benchmark the document arrangement of ESXi and

Xen, IOzone workload/benchmark instrument was utilized. Read, Write, Re-read, Re-compose, Random read, Random compose, backward read, Backwards compose, and Strided read tests on the document framework were done on shifting document and record sizes. To workload/benchmark the CPU performance, a progression of tests were completed speaking to various sorts of uses that may keep running on a PC framework, which may use the CPU vigorously. A Linux kernel compiles, a data compression test, a data encoding test, an application build test, and a graphics manipulation test were used. By analyzing the workload/benchmark comes about in light of memory operations; as a rule, (Perera and Keppitiyagama, 2011) watched that both hypervisors perform similarly well when the guests are conducting activities that need high memory transmission capacity (exchanging information amongst CPU and RAM). Quantitatively Xen is marginally quicker than ESXi. In network-based exercises too, we watched that the two stages are reasonable, however, ESXi is marginally superior to Xen. Utilization of fully virtualized guests (HVM) on Xen for network intensive deployments is not suggested as HVM guest displays inferior system performance on Xen. For file system based activities (disk110), ESXi performs superior to Xen, particularly Xen displays a performance degradation in writing to the file system. For CPU severe applications, both hypervisors perform similarly well, yet Xen is somewhat better. At the point when wholly virtualized guest operating systems are running on Xen, an immense performance disintegration was seen because of the emulation by the hypervisor on all test cases (Perera and Keppitiyagama, 2011).

Bhukya et al. (2010) presented a comprehensive evaluation methodology to workload/benchmark the performance of sequential programs by running them in several virtualized environments by the technique called virtualization and checking the throughput in those environments using a method called experimental design or design of the experiment. The workload/benchmark in this paper is running on Linux guests on virtualization technology (VT-x) enabled platforms. The results show that the how the performance of the following program in both Xen and VMware changes concerning the factors viz. type of hypervisor, size of RAM, and the number of virtual CPUs affecting it. In their experiment, the hardware platform is Intel® Core™ 2 DUO with an Intel processor at 2.93 GHz. It has got an L1 data cache of 32KB for private data of each core. It also has an L1 instruction cache of 32KB for instructions. It has an L2 cache of 3MB that is used for fast data access; it includes a SCSI disk of size 300GB with the RAM of size 4GB with DMA enabled. Their experiment is conducted by repeatedly executing the workload/benchmarks in NPB3.3 Serial and collecting the

performance data. (Bhukya et al., 2010) use the DOE (Design Of Experiment) methodology and formulate the results.

The experimenter should select a suitable design type considering the experiment. In this analysis, a Full Factorial design is selected. The design will provide an option for experimental runs. In this task, add up to quantities of runs are 36. The experimenter will be furnished with an alternative of choosing methods like replication, randomization, and hindering in the experiment to decrease the test blunders. Analysis Of Variance (ANOVA) is a customarily utilized actual procedure for inspecting the information by looking at the methods for subsets of the data. ANOVA is also used to assess the important impact of variables at various levels and their interaction impacts. The diagrams are plotted by considering their mean esteems. The exploratory outcomes give an understanding of how the productivity of the successive program impacts in both virtualized environments. (Bhukya et al., 2010) run CFD applications in virtualized environments and inspect their performance. These assist the clients to pick which hypervisor is useful for their successive application. Xen gives better performance when contrasted with VMware concerning the factors that we considered for our experiment. In the case of Xen, (Bhukya et al., 2010) explored that efficiency may always not be increased with increase in the size of RAM.

Kumar and M (2015) demonstrated that present measurements for the performance of offerings by cloud suppliers are liable to imprecision and changeability. The thesis tried to elucidate worries about performance in cloud computing, breaking down the variables that make the performance of clouds unpredictable and recommending approaches to tackle this issue. The performance degradation because of virtualization and the absence of isolation between virtual machines were observationally assessed in a eucalyptus testbed given the KVM virtualizer. Drawing upon past research, every one of the parts of the issue, from the conduct of particular application types when facilitated in clouds to a proposition for another age of SLAs with performance ensures, will be talked about.

This segment gives the outcomes along acquainting the system utilized to test the performance seclusion capacity of a KVM-fueled Eucalyptus private cloud. The fundamental issue is the effect on the performance of a specific virtual machine case when another VM is making escalated utilization of at least one physical, and therefore shared resources. The physical resources being considered are the CPU, memory, disk, and network interface. For each of these, the consequences of a trial will be displayed trailed by an exchange of the possible

reasons for these outcomes, bolstered by additional references to past investigations when fundamental. As it has been said before, a tweaked EMI was made to run these tests. This EMI was made out of a spotless establishment of Ubuntu Server 10.04 and different workload/benchmarking and testing programs. In the accompanying subsections, a short clarification of how each of these testing programs functions will be given. If not communicated unique, each experiment was performed with two running virtual machine occurrences of the modified EMI, with 15 GB disk, 512 MB RAM, and 1 CPU core. The approach depended on finding the components which the performance of cloud-facilitated applications relies upon. Right off the bat, the effect on the performance of a virtualized and shared physical server was tried all through the usage of a private cloud. This investigation, as opposed to comparable workload/benchmarks found in literature, was executed in a controlled domain as opposed to in a public cloud, where setup alternatives are constrained, and foundation stack is by, and massive hard to know and uncontrollable and like this permitted making some reasonable determinations (Kumar and M, 2015).

The main conclusion of this proposition is that cloud computing is, as a rule, arranged to host most typical web applications effectively and with incredible cost funds, yet those applications with strict inertness prerequisites or other network performance necessities, those that require working with substantial datasets, or those whose requirements for accessibility are basic should be considered precisely. In these cases, thought of particular performance necessities, distinctive for each sort, and remuneration models for infringement of the SLA are critical. Indeed, even with the development of SLAs, generally useful clouds will not will to different certifications for the most requesting applications, in this way, there will be an open door for more research in this area (Kumar and M, 2015).

Hypervisors are utilized as a part of cloud environments, and their effect on application performance has been a point of critical research and viable interest. The current development in cloud environments has quickened the progression of virtualization through hypervisors; be that as it may, with such a large number of various virtualization advancements, it is hard to discover how different hypervisors affect application performance and whether a similar performance can be accomplished for each hypervisor. (Li et al., 2013), conducted experimental estimations of a few workload/benchmarks utilizing Hadoop MapReduce to assess and look at the performance effect of three prominent hypervisors: a commercial hypervisor and open source Xen and KVM. (Li et al., 2013) found that distinctions in the

workload sort (CPU or I/O intensive), workload size and VM situation yielded excellent performance contrasts among the hypervisors. Many different hypervisors (both opensource and commercial) exist today, each with their advantages and disadvantages. This introduces a large number of new and challenging research questions.

Some past work has focused on virtualization overhead of a single hypervisor on a particular application or micro workload/benchmark. Other work has been aimed to provide a quantitative performance comparison between different hypervisors using microbenchmarks. The authors utilized the three hypervisors to run a few MapReduce workload/benchmarks, for example, Word Count, TestDSFIO, and TeraSort and further approved (Li et al., 2013) observed theories utilizing microbenchmarks. In our observation for CPU-bound workload/benchmark, the performance distinction between the three hypervisors was negligible; be that as it may, substantial performance varieties were seen for I/O-bound workload/benchmarks. Also, including more virtual machines, the same physical host debased the performance on each of the three hypervisors, yet they watched distinctive patterns among them. Solidly, the commercial hypervisor is 46% quicker at TestDFSIO Write than KVM, however 49% slower in the TeraSort workload/benchmark. What is more, expanding the workload estimate for TeraSort yielded completion times for CVM that was two times that of Xen and KVM. The performance contrasts shown between the hypervisors proposes that further examination and consideration of hypervisors are required later in future deploying applications to cloud environments (Li et al., 2013).

As cloud computing rises as a prevailing worldview in dispersed systems, it is vital to comprehend the fundamental advances that make clouds conceivable entirely. One innovation, and maybe the most essential, is virtualization. As of late, virtualization, using hypervisors, has turned out to be broadly utilized and surely understood by numerous. There is an expansive spread of various hypervisors, each with their advantages and disadvantages (Younge et al., 2011).

In late history, there have been multiple comparisons identified with virtualization innovations and Clouds. The primary performance analysis of different hypervisors began with, obviously, the hypervisor merchants themselves. VMware has cheerfully put out its take on performance as well as an original Xen article, which analyzes Xen, XenLinux, and VMware over various SPEC and standardized workload/benchmarks, bringing about contention between the two works. From here, various more impartial reports began, focusing on server consolidation and

web application performance with fruitful yet sometimes incompatible results. A feature-based survey on virtualization technologies likewise shows the vast assortment of hypervisors that as of now exist. Besides, there has been some investigation concerning the performance inside HPC, particularly with InfiniBand performance of Xen, and of late, a detailed take at the practicality of Amazon Elastic Compute cloud for HPC applications. Nonetheless, the two works focus on a solitary deployment as opposed to a genuine comparison of advantages (Younge et al., 2011). As these underlying hypervisors and virtualization implementations have developed quickly alongside virtualization, sustained specifically by standard x86 hardware, it is essential to painstakingly and precisely assess the performance ramifications of every system. Consequently, we directed an investigation of a few virtualization advances, specifically Xen, KVM, VirtualBox, and to a limited extent, VMware. Each hypervisor is contrasted closely with each other and (with the particular case of VMware) run through some High-Performance workload/benchmarking tools (Younge et al., 2011).

Younge et al. (2011) indicated that the goal of their manuscript was to viably thoroughly analyze the different virtualization advancements, mainly to support HPC-based Clouds. The first set of results speak to the performance of HPCC workload/benchmarks. Every workload/benchmark was run a total of 20 times, and the mean brought with error bars represented to indicate the standard deviation over the 20 runs. The workload/benchmarking suite was fabricated utilizing the Intel 11.1 compiler, utilizing the Intel MPI and MKL runtime libraries, set with defaults and no enhancements whatsoever. We open first with High-Performance Linpack (HPL), the accepted standard for comparing resources. They could see the comparison of Xen, KVM, and Virtual Box contrasted with native bare-metal performance.

To begin with, we see that the native bare-metal system is equipped for around 73.5 Gflops which, without any improvements, accomplish 75% of the hypothetical peak performance. KVM, Xen, and VirtualBox perform at 49.1, 51.8 and 51.3 Gflops, respectively, at the point when finding the average value of more than 20 runs. However, Xen, not at all like KVM and VirtualBox, has a high level of change between runs. This is a fascinating phenomenon for two reasons (Ali, 2015).

To begin with, this may affect performance measurements for other HPC applications and cause errors and postponements between even pleasingly-parallel applications and add to reducer work delays. Second, this vast difference breaks a key segment of Cloud computing, giving a particular and predefined nature of service. On the off chance that performance can

influence as broadly as what happened for Linpack, at that point, this may negatively affect clients. Next, they swing to another key workload/benchmark inside the HPC community, Fast Fourier Transforms (FFT). Dissimilar to the synthetic Linpack workload/benchmark, FFT is a particular, deliberate, workload/benchmark which provides results about which are regularly viewed as more concerning a client's useful application than HPL. (Younge et al., 2011) saw rather particular outcomes from what was already given by HPL. Taking a look at Star and Single FFT, its consistent performance overall hypervisors is equivalent to bare-metal performance, a great sign that HPC applications might be appropriate for use on VMs. The outcomes for MPI FFT also demonstrate similar outcomes except for Xen, which has a diminished performance and high variance as found in the HPL workload/benchmark. Their present speculation is that there is an adverse effect of using Intel's MPI runtime on Xen. However, the investigation is still ongoing.

Taking everything into account, the authors project that KVM is the best general solution for use inside HPC Cloud environments. KVM's component rich experience and near-native performance make it a natural fit for deployment in an environment where ease of use and performance are central. Inside the FutureGrid venture particularly, they would like to send the KVM hypervisor over our Cloud stages shortly, as it offers clear advantages over the current Xen deployment. Besides, we anticipate that these discoveries will be of remarkable significance to other public and private Cloud deployments, as system usage, Quality of Service, working expense, and computational effectiveness could all be enhanced through the careful assessment of major virtualization advancements(Younge et al., 2011).

(Ha et al., 2016) studied the I/O performance of long, consecutive workloads that copy those of Big Data applications, to comprehend the ramifications of framework virtualization on information-intensive structures, for example, Apache Hadoop and Spark, which are as often as possible keep running in groups of Virtual Machines (VMs). They do experimental measurement campaign that collects low-level traces and metrics, to show the role played by essential parameters such as the I/O schedulers and caching mechanisms involved in the I/O path, and the VM configuration regarding dedicated resources. Our findings are significant, especially for determining appropriate deployment strategies for today's emerging Analytics Services host both on a public and private cloud.

Ha et al. (2016) used FIO, which is a flexible tool that allows designing a variety of workloads, and that provides detailed statistics for computing our metrics. FIO is widely used in academia

and industry for standard workload/benchmarking, stress testing, and I/O verification purposes. They run FIO on the physical host with different numbers of concurrent threads. Then with the same configuration. Finally, they also study the case of multiple active VMs being instantiated on the same physical host, each with one FIO thread performing I/O operations. (Ha et al., 2016) used the same OS and settings for consistency across different measurement scenarios. The same fixed amount of data, namely 4 GBs, was used and distributed evenly across all the threads in our experiments. Their reported performance figures are the result of 10 runs with error bars to verify results variability.

The goals of their work were to understand the implications and overheads of virtualization on I/O performance, focusing on the storage subsystem. Indeed, many Big Data applications are I/O bound in nature, and a proper assessment and understanding of I/O performance in Cloud environments are essential. To answer our questions, we used an in-depth, low-level measurement study and analyzed the behaviour of several configurations supporting the specific workloads that characterize analytics applications, that is, extended sequential operations (Ha et al., 2016). Findings are instrumental in defining how to configure cloud computing environments to meet high I/O performance demands by modern Big Data applications and to indicate areas requiring further research efforts. They showed that current best practices for Big Data application deployments are not reaping the benefits of decades of research and engineering done at the OS level.

The present virtualization arrangement in the Cloud broadly depends on hypervisor-based technologies. Alongside the current prominence of Docker, the container-based virtualization begins accepting more consideration for being a promising option. Since both of the virtualization arrangements are not resource-free, their performance overheads would prompt adverse effects on the nature of Cloud services. To help fundamentally comprehend the performance difference between these two sorts of virtualization solutions, we utilize a physical machine with "simple enough" assets as a gauge to examine the performance overhead of an independent Docker holder against an independent virtual machine (VM). With discoveries as opposed to the related work, results demonstrate that the virtualization's performance overhead could shift not only on a feature-by-feature basis but also on a job-to-job basis (Li et al., 2017). Even though the container-based arrangement is without a doubt lightweight, the hypervisor-based innovation does not accompany higher performance overhead for each situation. For

instance, Docker containers especially display lower QoS as far as storage transaction speed (Li et al., 2017).

Currently, several streaming servers are available to provide a variety of multimedia applications such as VoD (Video on- Demand), IP-phone, and IP-TV. As a result, the provision of multiple streaming servers on a single machine using the virtualization technology has become essential to save the operational/management costs while enhancing the performance and the reliability of the system. The authors(Sritrusta et al., 2009), evaluated the performance of two representative open source software for the virtualization technology, *Xen*, and *OpenVZ*, in various configurations of applications on three open source streaming servers, *Red5*, *Darwin*, and *VLC*. Their experimental results indicated that OpenVZ provided better performance for streaming applications with Darwin and VLC, whereas Red5 can run only on Xen. They compared the application-level performance such as the throughput and the response time when they run three open source software for multimedia applications, Red5, Darwin Streaming Server, and VLC, on a virtualization software by preparing three scenarios for the platform, namely on the Linux native system, on Xen, and on OpenVZ for comparisons. Their results showed that OpenVZ achieved higher performance for Darwin and VLC, whereas Red5 could only run on Xen. Siege was used for a stress test on the streaming servers and Unibench for performance evaluation. For future studies, they suggest to incorporate performance assessments of Internet servers, for example, Radius, DHCP, and LDAP, and different applications on the virtualization innovation (Sritrusta et al., 2009).

Virtual machines (VMs) have as of late risen as the reason for allocating resources in enterprise settings and hosting centres. One advantage of VMs in these environments is the capacity to multiplex a few operating systems on hardware based in light of progressively changing system characteristics. Be that as it may, such multiplexing must frequently be done while observing per-VM performance assurances or service level agreements. Accordingly, one vital prerequisite in this condition is robust performance isolation among VMs. Virtual machines empower fault isolation - "encapsulating" diverse applications in independent execution environments, so a failure in one virtual machine does not influence different VMs facilitated on the same physical equipment. Performance isolation is another vital objective Individual VMs are often configured with performance guarantees furthermore, desires, e.g., in light of service level agreements. Along these lines, the resource utilization of one virtual machine ought not to affect the guaranteed assurances to different VMs on the same hardware.

Virtualization is quickly turning into a commercially suitable option for expanding system utilization. However, from a customer perspective, virtualization cannot succeed without providing appropriate resource and performance isolation guarantees. The Authors (Gupta et al., 2006) proposed two mechanisms – SEDF-DC and ShareGuard – that improve CPU and network resource isolation in Xen. They demonstrated how these mechanisms enable new policies to ensure performance isolation under a variety of configurations and workloads. They trust that performance isolation requires suitable resource distribution arrangements. In this way, another region for future examination is strategies for proficient capacity planning and workload administration and also plan to extend these mechanisms to support other resources such as disk I/O and memory (Gupta et al., 2006).

Dall et al. (2016), presented the first study of ARM virtualization performance on server hardware, including multi-core measurements of the two major ARM hypervisors, KVM, and Xen. They acquaint a suite of microbenchmarks with measure regular hypervisor operations on multi-core systems. The two major ARM hypervisors, KVM and Xen, acquaint a suite of microbenchmarks with measuring normal hypervisor operations on multi-core systems. Utilizing this suite, they demonstrate that ARM empowers Type 1 hypervisors, for example, Xen to transition between a VM and the hypervisor significantly faster than on x86, however, this low change cost does not stretch out to Type 2 hypervisors, for example, KVM, in light of the fact that they cannot run entirely in the EL2 CPU mode ARM intended for running hypervisors. While this quick change cost is valuable for supporting virtual interferes with, it doesn't help with I/O performance in light of the fact that a Type 1 hypervisor like Xen needs to communicate with I/O backends in an extraordinary Dom0 VM, requiring more complex interactions than basically progressing to and from the EL2 CPU mode. (Dall et al., 2016) demonstrate that present hypervisor outlines cannot use ARM's conceivably quick VM-to-hypervisor transition cost in practice for genuine application workloads. KVM ARM surpasses the performance of Xen ARM for generally genuine application workloads including I/O. This is because of contrasts in hypervisor software design and usage that assume a more significant part than how the hardware underpins low-level hypervisor operations. For instance, KVM ARM effectively gives zero duplicate I/O since its host OS has full access to the more significant part of the VM's memory, where Xen authorizes a strict I/O isolation approach bringing about poor performance despite Xen's considerably quicker VM-to-hypervisor transition mechanism. They demonstrate that ARM hypervisors have comparable overhead to their x86 counterparts on certain applications. At long last, (Dall et al., 2016) indicate how

changes to the ARM engineering may permit Type 2 hypervisors to bring ARM's quick VM-to-hypervisor transition cost to genuine application workloads involving I/O.

Benevenuto et al. (2006) proposed queuing models for predicting the performance that applications running on a Linux system, will accomplish if migrated to a Xen virtual system, with the same hardware arrangement. To exhibit the reasons for virtualization overhead on the Xen VMM, the authors give a performance assessment of three workload/benchmarks running on Xen and Linux. They use a case study of an application Apache as a Web server to validate the performance models by sending requests from HTTPERF as clients to the Web server, measuring throughput and server response time of the requests. They wanted to predict the performance of a Web server application, running on a physical system, will accomplish whenever relocated to a Xen virtual machine. Their research strategy consisted of a performance study of a Web server which provided static content. They presented some results to discuss which components of the Xen environment need to be considered in a model. To develop performance models (Benevenuto et al., 2006) needed to be able to measure the virtualized system. By developing an application called *Xencpu* to measure CPU busy time on Xen. This tool depends on the source code of XM tool, furnished with Xen, and was designed aiming at the automatic execution of scripts. The CPU busy time on the Linux system was acquired based on data from `/proc` directory. Disk busy time, on both Linux and Xen, was likewise acquired from the `/proc` directory. Different parameters, for example, experiment duration and some processed requests, are acquired with scripts or from the workload/benchmarks utilized based on data from `/proc` directory. Disk busy time, on both Xen and Linux, was also from the `/proc` directory. Different parameters, for example, experiment duration and some processed requests, are obtained with scripts or from the workload/benchmarks used.

To harvest high-performance systems, cluster operating environment has pressed on additional abstraction using virtualization technology. Specific problem-solving environments are isolated at the operating system level, where real executions are performed in the virtualization domain. Virtualization technology helps not only to increase the utilization of computing resources but also reduce configuration workload, administrative cost, application porting, and energy saving. Numerous product operating systems are permitted to share ordinary hardware in a safe environment. (Prueksaaron et al., 2009) investigated the implementation of a virtualized computing cluster. Performance evaluation results of the virtualized cluster based

on HPL is shown where the maximum of 20% performance degradation from virtualization overhead is observed. (Prueksaaron et al., 2009) describe a straightforward approach to deploy virtualization onto the cluster. The management of virtualization images is discussed and the process of considering computational cluster resources in the virtualization environment is described. The objective of this work is to the virtualized resources running with a production of the virtual cluster environment.

Padala et al. (2007) evaluate two representative virtualization innovations, Xen and OpenVZ, in different arrangements. They merge at least one multi-layered system onto a couple of hubs and drive the system with a bartering workload called RUBiS. They contrast the two advancements and a base system as far as application performance, resource utilization, versatility, low-level system measurements like cache misses, and virtualization-specific measurements like Domain-0 utilization in Xen. Their analyses demonstrate that the average reaction time can increment by more than 400% in Xen and just an unassuming 100% in OpenVZ as the number of application instances occurrences develops from one to four. The higher virtualization overhead causes this expansive error in Xen, which is likely because of higher L2 cache misses and misses per instruction. A similar pattern is seen in CPU utilization of virtual containers. (Padala et al., 2007) give an overhead examination with kernel-symbol-specific information created by Oprofile.

Another way to deal with building broad scale computing systems by virtualizing existing resources utilizing system virtual machine (VM) innovations (e.g., VMware and Xen) to help adaptable resource offering to solid disconnection and advantageous application deployment on modified execution environments, is considered. VMs are ending up unavoidably utilized, driven by the quick development and broad accessibility of VM products, and additionally, the fast development of computing energy of present-day computers. Their deployments can be found from big data centres for resource consolidation to PCs for multi-OS hosting. In their proposed framework, (Martinez et al., 2009), mentioned that VMs can be progressively conveyed to encourage the combination of uses and co-designation of the available resources of existing PCs both scattered crosswise over associations and owned by people. Subsequently, resource-requesting applications can be disseminated and executed alongside the VMs in a considerably parallel manner. Remembering the actual objective to investigate the attainability of building a vast scale virtualized computing framework and to recognize the potential research challenges, (Martinez et al., 2009) have built up an extensive VM-based system

comprising of more than 100 VMs facilitated on 30+ shared existing physical servers at FIU. Two delegate enormously parallel applications are tried on this condition, and an examination of how the performance is influenced is introduced and analyzed accordingly taking into consideration the nature of each application. VM advancements give a deep layer of abstraction for resource sharing.

System-level VMs are considered in this thesis, which depends on the virtualization of whole physical hosts' resources, including memory, CPU, and I/O devices, and introducing virtual resources to the guest operating systems and applications. Even though the procedures proposed in this paper can likewise be connected to a portion of other sorts of virtualization (e.g., OS-extension based VMs), those are not the concentration of this paper. System VMs incorporate the accompanying two sorts: full-virtualized VMs and paravirtualized VMs. Full-virtualized VMs (e.g., VMware ESX) exhibit a similar hardware interface to guest OSs as the physical machines and in this way bolster unmodified OSs in the VMs. Paravirtualized VMs (e.g., Xen) introduce an altered hardware interface which is advanced to decrease the overhead of virtualization, yet they require the guests OSs to be adjusted too with a specific end goal to oblige these progressions. System virtualization is actualized by the layer of software called virtual machine screen (VMM, a.k.a. hypervisor). VMM can be either hosted on a current OS or run correctly on the hardware. Hosted VMs use the local OS to get to resources and in this manner, ordinarily brings about more overhead, yet they can be helpfully sent on existing resources and straightforwardly work with their OS installations. Illustrations incorporate VMware Server on Windows and Linux, Parallels Desktop on Mac OS. Non-hosted VMs require existing OSs to be evacuated so VMM can have coordinate control of the resources; however, they can commonly convey better performance contrasted with hosted VMs. Cases of non-hosted VM items incorporate Xen and VMware ESX Server. Accordingly, non-hosted VMs are gradually picking up predominance in server virtualization environments, while hosted VMs are all the more broadly utilized as a part of systems where VMM needs to exist together with traditional OSs without upsetting the ordinary operation of those systems (Martinez et al., 2009).

Voorsluys et al. (2009) Virtualization technology have become commonplace in modern datacenters furthermore, cluster systems, regularly alluded as computing clouds". Specifically, the capacity of the virtual machine (VM) movement brings various advantages, for example, higher performance, enhanced sensibility, and adaptation to internal failure. Virtual machine

(VM) innovation has of late developed as a fundamental building block for data centres. Furthermore, cluster systems, fundamentally because of its abilities to isolate, merging, and relocating workload. Altogether, these features allow a data centre to serve multiple users in a secure, flexible, and efficient way. Subsequently, these virtualized infrastructures are considered a crucial part of driving the developing Cloud Computing worldview. Migration of virtual machines tries to enhance manageability, performance, and fault tolerance of systems. More specially, the reasons that legitimize VM relocation in a production system include the need to adjust system load, which can be the expert by moving VMs out of overburden/overheated servers; and the need of explicitly bringing servers down for maintenance in the wake of relocating their workload to different servers. The capacity to move a whole operating system beats most troubles that traditionally have influenced process-level migration a complex operation. Applications themselves and their comparing processes should not know that relocation is happening. Hypervisors, for example, Xen and VMware, permit migrating an OS as it keeps on running. Such method is named as "live" or "hot" movement, rather than "pure stop-and-duplicate" or "cold" migration, which includes stopping the VM, replicating all its memory pages to the destination host and after that restarting the new VM. The preferred primary standpoint of live migration is the likelihood to migrate an OS with close to zero downtime, a critical component when live services are being served (Voorsluys et al., 2009).

Hypervisors utilizing virtualization technology empower numerous operating systems to keep running on one physical server. Cloud computing model is more affordable because it streamlines the conveyance of services by giving a phase to improving sophisticated IT resources in a flexible way with the assistance of virtualization technology and hypervisors. Choosing a reasonable hypervisor for their organization's private cloud is a gigantic task for current CIOs. Hypervisor merchants do guarantee that they have refuted virtualization overhead totally contrast with native system, yet at the same time there exists minute virtualization overhead on the grounds that virtual machines need to communicate with the centre layer hypervisor to get to the underlying physical hardware and moreover there is an impact of other virtual machines running on the equivalent hypervisor. Hypervisors are created utilizing various virtualization procedures like full virtualization, para-virtualization, and hybrid model virtualization. This paper assesses the performance of three hypervisors ESXi, XenServer, and KVM utilizing SIGARframework for system data as well as Passmark for system workloads in private cloud environments. The private cloud has been composed utilizing open source cloud computing software CloudStack. Hypervisors are conveyed as

hosts in the CloudStack. This paper suggests best-suited hypervisors for respective workloads in the private cloud based on the performance of system information and system workloads (Reddy and Rajamani, 2014).

Cloud computing as a model empowers on-request access to servers, networks, applications and gives the alternative to pay as you utilize way. The real advantages of cloud computing are adaptable and versatile infrastructures, decreased execution and support costs, IT division transformation, and expanded accessibility of high-performance applications. Cloud computing model encourages availability and is made out of four deployment models. In which, Private Clouds are sent behind the firewall of an organization, and the cloud infrastructure is worked exclusively for an organization. Private cloud deployment shows model exclusive computing design behind a firewall with full control over the infrastructure. This paper utilizes a private cloud model for the experiment (Reddy and Rajamani, 2014).

Virtualization is an innovation that joins or partitions computing resources to exhibit many operating environments utilizing procedures like hardware and software partitioning, machine reproduction, emulation, and timesharing (Ali, 2015).

From the 1990s until the mid-2000s, the pattern in the data centre was to help the organization by giving cheap and powerful x86 server setups isolated and committed to particular applications. A decentralization administration approach hosted these applications due to the utilization of a software development lifecycle (SDLC) as a systematic way to deal with application improvement that regularly required a devoted server design to host isolated and unmistakable iterations of an application (i.e., advancement, test, and production). The commoditization of servers and PC hardware and the simplicity of server upkeep fixes or upgrades could be connected without compromising or influencing other applications or operating systems, giving extra purposes behind decentralizing the administration of the application server-hosting environment also in a similar period, the late 1990s, associations immensely expanded use of a blend of uses, for example, database access, Web systems, decision support systems, dispersed file services, transaction processing systems, and high-performance computing to help their business needs. This expansion of utilization use caused an exponential development in application hosting servers, which made the requirement for organizations to combine their server platforms inside an incorporated server farm. This is ordinarily alluded to as server sprawl, which is a circumstance or the pattern in server development in which various, under-utilized servers consume up more space and expend a

more significant number of resources that can be justified by their workload. In the 2000s, associations saw this heightened server sprawl caused an expansion in power utilization prerequisites, wasteful operational aspects, and upkeep overhead. IT organizations looked for approaches to lessen the server sprawl affect since it has brought forth the accompanying issues, underutilization of hardware, lack of space in server farms and expanded operational expenses. Thusly, numerous IT organizations were using server virtualization advancements to merge applications inside servers to avoid server sprawl that would lessen the prerequisites for extra server hardware, power spending, and data center space (Sligh and Owusu, 2014).

The virtualization innovation has been created quickly with the development of the hardware supported virtualization advances and the presence of the different services. Numerous analysts are focusing on building up the virtualization advances which are perceived as the most significant core technology of the IT applications, for example, green IT and Cloud Computing. The virtualization advancements have been considered for expanding usage of centralized computer servers since the 1960s. The fundamental ideas of the virtualization initially originated from the circumstance that loads of servers had low use rate around 10~20% around then. For improvement, various servers worked on the virtualized machine which exists in one actual physical machine to build server use and additionally support security. The conventional virtualization advances have been fundamentally produced for the centralized computer servers when desktop PC did not have enough performance. The exploration of the virtualization innovation for desktop PC has been begun effectively since the 1990s through the improvement of desktop PC. At present there are a few server virtual machines for desktop PC, for example, VMware Corporation has distributed Xen which has been produced and maintained by open source community and VMware ESX Server. The three parts of virtualization technologies are processed virtualization, device virtualization, and memory virtualization (Kim et al., 2010).

Cloud Computing is a developing innovation which gives on-demand service, and it offers dynamic computing infrastructure and allocation of resources optimally. Cloud is a set of hardware, network, storage, and an interface that empower the conveyance of computing as a service (Kumar and Singh, 2015). Cloud services incorporate conveyance of software, infrastructure, and capacity over the internet based on user demand. The infrastructure might be virtualized across the globe, means you may not know where your computing resources, application, or even data reside. These service providers are designing their infrastructure for scale. The leading Cloud deployment models are a private cloud, public cloud, hybrid cloud,

and community cloud. A particular entity controls private cloud and customarily used only by that entity or one of its customers. The underlying technology may reside on-site or off-site. A private cloud offers increased security at a more significant cost. A public cloud which is available for use by the general public may be controlled by a substantial company or organization providing cloud services. In a community cloud, the cloud is shared by two or more organizations typically with shared concerns (such as schools within a university). A hybrid cloud is a cloud that consists of two or more private, public, or community cloud. Service providers offer economically efficient services using virtualization of resources (Kumar and Singh, 2015).

Virtualization was developed over thirty years before enabling substantial, costly mainframes to be effortlessly shared among various application environments. As hardware costs went down, the requirement for virtualization blurred away. All the more as of late, virtualization at all levels (system, storage, and network) wound up essential again as an approach to enhance system security, reliability, decrease costs, and give more prominent adaptability. Virtualization is being utilized to help server consolidation endeavours. Many virtual machines running diverse application environments share the same hardware resources. System virtualization includes a hardware abstraction layer, called Virtual Machine Monitor (VMM), over the bare hardware. This layer gives an interface that is practically proportionate to the real hardware to various virtual machines. These virtual machines may then run standard operating systems, which would regularly run correctly over the actual hardware. There are different virtualization methods and in additional requirements for architectures to be virtualizable. The principle inspiration for virtualization in the mid-seventies was to build the level of sharing and use of costly computing resources, for example, the mainframes. The eighties saw a decline in hardware costs that caused a considerable part of the computing needs of an organization to be moved far from extensive incorporated centralized servers to an accumulation of departmental minicomputers. The primary inspiration for virtualization vanished what is more, with it their commercial exemplifications. The approach of microcomputers in the late eighties and their boundless appropriation amid the nineties alongside omnipresent networking brought the dissemination of computing to new grounds. The expansive number of customer machines connected to many servers of various types gave rise to new computational paradigms such as client-server and peer-to-peer systems. These new environments carried with them a few difficulties and issues including, security, reliability, expanded administration cost and complexity, expanded floor space, energy consumption, and thermal dissemination necessities.

The current resurrection of the utilization of virtualization methods in commodity, economical servers, and client machines is ready to address these issues in a prosperous manner. Virtualization might be utilized for server consolidation. Each Virtual Machine underpins the operating system and application environments of a server being consolidated in the virtualized environment (Jiang et al., 2014).

The computational humankind is complimenting to a high degree, cumbersome and multifaceted. Cloud Computing is getting to be a standout amongst the most growing methodologies in the computing business. It is a novel approach for its deliverance services on the World Wide Web. This model gives computing resources in the puddle for customers, entirely through the Internet. In cloud computing, resource designation and scheduling of various total web services are a goal and this paper gauges the different network resource allocation methodologies and their applications in the Cloud Computing environment. A short depiction for network resource allocation in Cloud Computing, given differentially adjusted dynamic extents, has additionally been finished. This research addresses and orders the preeminent difficulties ordinary to the resource allocation progress of Cloud Computing as far as different sorts of resource allocation techniques (Mohan and Raj, 2012). Cloud Computing is a computing model that keeps up measurements and applications, utilizing web and focal detached servers. This approach licenses end clients and organizations to utilize applications without putting in and entrée their private records at any PC with web entrée.

Cloud computing grants for considerably more capable computing by centralizing storage, memory, dispensation, and data transfer capacity. A few cases of cloud computing are Yahoo email, Google, Gmail, or Hotmail. Cloud Computing goes about as an administration modestly than stock, whereby common resources, software, and data are given to PCs and different procedures. Cloud computing can be arranged into three services, namely, i) SaaS (software-as-a-service), ii) PaaS (platform-as-a-service), and iii) IaaS (infrastructure-as-a-service) respectively. Designation of Cloud resources ought not just to ensure Quality of Service (QoS) requirements specified by customers using Service Level Agreements (SLAs), yet additionally to consolidate energy consumption. Resource allocation is one of the urgent issues in cloud computing, where rare resources are distributed. From a buyer's perspective, resource allocation identifies with how products and services are dispersed amidst clients. Capable resource distribution brings about a more enterprising economy (Mohan and Raj, 2012).

The virtualization technology (VT) can be classified into three major approaches: full-virtualization, para-virtualization, and OS-level virtualization. Differences among these approaches are considered regarding performance, ease of installation and administration, level of security between each virtualization images, and supported hardware platforms. Several virtualization platforms are mainly accomplished using the software, so-called virtualization software. The virtualization software creates and manages virtual tools, such as a processor unit, a memory unit, and I/O units for the virtual operating system (Prueksaaron et al., 2009).

Virtual machine innovations are a vital and necessary part of Cloud Computing. They lessen administration complexity by permitting multiple operating systems, separated process environments, and adaptation to non-critical failure. Workloads can be all the more effortlessly merged, and keeping software refreshed is not anymore a tedious undertaking. As cloud infrastructure gets more modern, the quantity of utilizations moving to the cloud develops. Virtual machines give many advantages to these applications. Presently, like never before, it fundamentally critical to look at and audit the performance impacts of virtualization in Cloud Computing infrastructure. Virtual machines give many advantages and are regularly used to better use the more significant part of the hardware resources accessible powerful servers and hardware such as workload consolidation, updated applications, simultaneous operating systems, and machine isolation (Overby, 2014).

There are two primary sorts of virtualization innovations today — hypervisor-based technology such as VMware vSphere, Microsoft Hyper-V Virtual Server, KVM and Xen, and operating system (OS) level virtualization such as VMware Workstation, Microsoft Virtual Server, Oracle VirtualBox, OpenVZ, Linux VServer and Solaris Zones (Padala et al., 2007).

The Authors(Hwang et al., 2013) extensively compared four hypervisors: Hyper-V, KVM, vSphere, and Xen. They show their performance differences and similarities in a variety of situations. Their results indicate that there is no perfect hypervisor and that different workloads may be best suited for different hypervisors. They believe that the results of the study demonstrate the benefits of building a highly heterogeneous data centre and cloud environments that support a variety of virtualization and hardware platforms. While this has the potential to improve efficiency, it also will introduce some new management challenges so that system administrators and automated systems can adequately make use of this diversity. (Hwang et al., 2013). Results also illustrate how competing VMs can have a high degree of performance interference. They noted that correctly determining how to place and allocate

resources to virtual servers will remain a vital management challenge due to the shared nature of virtualization environments. Moreover, more research should be done in the future to solve those problems.

To utilize one physical server with the ability to convey the performance of multiple servers. Virtualization enables IT Managers to boost resources by consolidating them on a single server. With virtualization, new applications will be made accessible inside a couple of minutes and without the cost of extra equipment (Elsayed and Abdelbaki, 2013).

The enthusiasm for virtualization has been developing quickly in the IT business on account of essential advantages like better resource use and simplicity of system manageability. The experimentation and utilization of virtualization, and also the concurrent deployment of virtual software, are progressively getting mainstream and being used by educational institutions for research and educating (Ali and Meghanathan, 2011).

With the advent of cloud computing and virtualization, modern distributed applications run on virtualized environments for hardware resource utilization and flexibility of operations in an infrastructure. However, when it comes to virtualization, resource overhead is involved. Linux containers can be an alternative to traditional virtualization technologies because of their high resource utilization and less overhead. (Joy, 2015) provided a comparison between Linux containers and virtual machines regarding performance and scalability. Containers have outperformed virtual machines regarding performance and scalability. Because of their better scalability and resource utilization, containers can be used for application deployments to reduce resource overhead. However, there are use cases where virtual machines would be a better fit than Linux containers. One of the use cases is running applications with business-critical data (Joy, 2015).

The expression "virtualization" portrays the production of a Virtual Computer System (VCS) or Virtual Machine (Strobl et al., 2013).

2.7 A COMBINATION OF METHODS/Framework

A combination of methods/framework is used in this thesis; the table below summarizes the methods that are was taken into consideration as well as their strengths and weaknesses:

Table 1: Methods / Framework

Work	Model/Framework	Strengths	Weaknesses
(Buyya et al., 2011)	A framework for SLA management with particular reference to managing QoS requirements	Successfully integrates the market-based resource provisioning with virtualization technologies for flexible resource allocations.	Does not integrate IaaS, PaaS and SaaS in a combined manner.
(Chen and Zhang, 2012)	A set-based PSO approach scheduling problem in cloud computing	Multiple parameter optimizations are possible.	However, no monitoring mechanism is implemented- ed for catching violations.
(Emeakaroha et al., 2011)	A scheduling heuristic that takes multiple SLA parameters when deploying applications in the Cloud	Considers deployment attributes such as CPU time, network bandwidth, storage capacity, etc., before installation of applications in the cloud system.	Does not consider performance parameters such as response time, performance time, etc.,
(Li et al., 2012)	Profit-Based Analysis of Resource Allocation on QoS	An innovative method for analyzing the impact of resource provisioning.	No discussion on how to optimally allocate resources.
(Stoicuta et al., 2012)	A monitoring application for QoS parameters in iOS5.	Can be used by clients to monitor the performance of service providers.	Minimal application due to focusing only on available transfer rate and one-way delay as QoS parameters.

2.8 CHAPTER SUMMARY

Virtualization is an essential aspect of Cloud Computing. It allows abstraction centred on services and isolation of lower level functionalities and underlying hardware. However, few efforts exist to configure virtualization in cloud computing optimally. Cloud Computing provides a new way of computing resource distributing based on virtualization (Jiang et al., 2014). Virtualization techniques are essential to Cloud Computing since it improves service delivery by giving a platform to optimizing sophisticated IT resources in a versatile way. Virtualization's three attributes; partitioning, isolation, and encapsulation makes it perfect for cloud computing. In cloud computing, virtualization has been used in all computing services that include memory, storage, operating systems, networks, applications, and hardware.

CHAPTER 3

METHODOLOGY

3 INTRODUCTION

This chapter defines the experimental environment used. This section states the experiments used to complete this study. We also distinguish data required from these layers, and a method to gather the information. Finally, we depict a method to break down extra information and decide whether the virtual machine is encountering performance costs or not.

We first need to set up a known baseline for the resources by calculating overhead in that setup. When running different systems in production, we can think about throughput and latency of the resources at the virtual machine and host layer.

Even though server virtualization can utilize all resources productively, optimizing virtualization software can empower a more significant number of guest machines to run concurrently. Popular optimization methods are lessening the overhead caused by virtualization and comparable resources sharing between guest machines. Virtualization overhead is caused by operations which directly cannot be executed on the hardware and by extra mappings which are used to give the guest VM a typical environment.

3.1 MEASURING VIRTUALIZATION EFFECTS BY TESTS

Notwithstanding the theoretical perspective, tests are used to measure the impacts of virtualization. Measurements are partitioned into three primary classes, given their character (Smith and Nair, 2005):

- Operation under various circumstances;
- Pure performance; and
- Environmental isolation and security.

Performance tests in a virtualized domain are comparatively contrasted with conventional one; the objective is discovering the ideal configuration for performance. For a theoretical perspective in virtualization overheads, comparative operations are performed in virtualized and traditional conditions (Benevenuto et al., 2006).

Albeit unadulterated performance influences the after effects of operation tests, it likewise considers other essential features, for example, system administration and joining to existing infrastructure. If re-establishing a virtual machine from a backup requires special tools or additional phases contrasted with restoring a conventional system, the distinction can be found by utilizing these operation tests.

Unadulterated performance tests are utilized to measure conceivable overhead that virtualization causes contrasted with performance utilizing the native hardware. Operation tests give a better general picture of the contrasts between a traditional and virtualized environment. Furthermore, performance tests can be utilized to check diverse improvement and resource sharing plans. Environment security tests are utilized to guarantee that virtualization software is equipped for giving comparative security and isolation features than independent physical systems in a familiar environment. Performing tests in a virtualized environment do not fundamentally vary from testing a traditional environment. The main significant contrast is that rather than utilizing a few physical systems, an essential piece of testing should be possible in an isolated physical system(Li et al., 2013).

3.1.1 Test types of performance test

Performance tests have three classifications in light of what is the primary focus of the test. The classifications are the following (Jiang et al., 2014):

- Hardware. The impact of the operating system is reduced so that only device drivers and essential OS parts are utilized to run the tests;
- Software. Operating system testing and different parts of various software; and
- Overall performance: incorporates software and hardware particular capacities.

While the hardware has a specific performance as per details and models, a poor software execution of, e.g., device drivers can decrease this performance altogether. A case of a standard hardware test is estimating the disk's reading and writing speed, while conventional software test can be, e.g., testing memory administration or the scheduler of OS. General tests more often than not contain performing a common task that utilizations both hardware and software resources intensively (Sligh and Owusu, 2014).

Performance testing between virtualized and physical environments is easy if the host and virtual machine have the same operating systems. Performance tests joined with factual

information about utilizing resources can be utilized to give an estimate of what number of virtual machines they chose hardware is competent to run fluidly. In the arranging phase of server virtualization, this data is valuable since choosing appropriate hardware arrangement winds up less demanding (Ali, 2015).

3.2 EXPERIMENTAL ENVIRONMENT

As seen in the literature review, authors evaluated virtualization performance either using Type 1 or Type 2 hypervisor on desktop computers to present cloud computing or data centre (private or public) (Morabito et al., 2015). In this study, datacentre purposed servers together with Type 1 (bare metal hypervisors) were used to evaluate virtualization performance. To create a private cloud environment, the methodology and resources used the same design template and outlook as how many other researchers have done with variation in the type of hardware and software specifications (Morabito et al., 2015). Section 3.2.1 below shows the hardware and software specifications for the physical machine that was used to build the private cloud environment and sub-section 3.2.1.1 shows the specifications for the virtual machine which will be evaluated for performance in line with objectives that have to be met. (Reddy and Rajamani, 2014).

3.2.1 Host system

The experimental environment was conducted on server Cisco UCS B200 M4, which was the host machine with the following hardware and software specifications below. Cisco UCS was chosen because many experiments of this nature have not used this model before, and it would provide a new variable in the experiment setup (Elsayed and Abdelbaki, 2013):

Hardware:

- Processor type: Intel® Xeon CPU E5620 @ 2.40GHz
- Cores: 8 CPUs x 2,393 GHz
- Memory: 47,99 GB
- Storage: 131 GB local and 5,87 TB from SAN (NetApp Storage Area Network)
- Processor sockets: 2
- Cores per Socket: 4
- Logical Processors: 16
- NICs: 4

Software:

- VMware ESXi 5.5, as seen in Figure 2, is managed through VMware vSphere was the chosen hypervisor because VMware can provide secure and stable virtual environments. Is it the world leading hypervisor for virtualization with some benefits? VMware has a significantly advanced Graphic User Interface as well as a Web-based interface that users make use of to manage VMware hosts remotely and centrally easily (Younge et al., 2011). VMware is commercial.

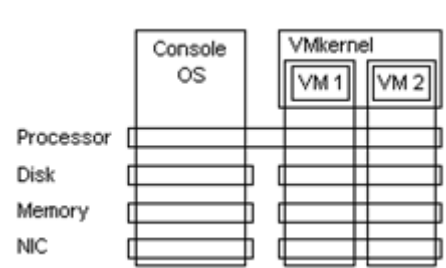


Figure 2: VMware ESXi Server architecture

- Proxmox 5.3 with an illustration on Figure 3 was chosen as the second hypervisor as it a data centre virtualization software which is freely available because it is open source (Kovari and Dukan, 2012). This will provide not only results for virtual machine performance only holistically but compare between open source and commercial as the first hypervisor,

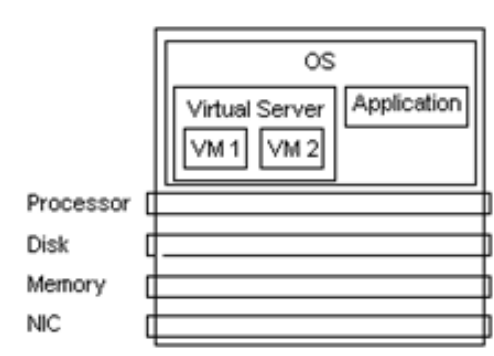


Figure 3: Proxmox Server architecture

3.2.1.1 Virtual machines

The virtual environment that is encapsulated within the physical layer, which hosts the guest virtual machines have of Guest OSs, virtual hardware, as well as applications. The host server consists of one operating system, CentOS 7 64 bit. The same amount of virtual resources are allocated for performance evaluation for each guest operating system. Table 2 virtual setup for the guest operating system.

Table 2: Resource distribution plan

Operating system	CPU	Memory	Storage
CentOS 7 64 bit	4vCPU ,2,40Ghz	4GB	40GB

3.3 PERFORMANCE

We discuss the performance tools used to measure various parameters. These workload/benchmark measurements help to recognize the utilization of network, CPU, disk, and memory resources that can be allocated for virtual machines. Information about the tools or procedures that are utilized for estimating the performance of different workload/benchmark parameters like Network, CPU, Memory, Disk, and are given here.

Workload/benchmarking tools measure performance characteristics. Testing hardware feature performance is vital because it makes performance measurements evaluation in the thesis meaningful. From the literature review, it can be seen that performance evaluation can be determined with workload/benchmark results. Performance results can be the same when the performance characteristics of the two machines are similar. These tools were chosen from a list of options in table form that can be seen in between sections 3.3.1 and 3.3.4

3.3.1 Network performance

Network performance can be a noteworthy estimation in virtualized systems. The effect of virtualization on network performance experienced by clients can be calculated to portray virtual machine instances networking performance. Network transmission capacity, as well as network trace, are observed and estimated by utilizing network performance workload/benchmark. Measurement of network performance can be experimented using various workload/benchmarking tools such as the ones in table 3.

Table 3: Available Tools for Network Performance

S No.	Software tools	Operating System	Version
1	Netmeter	Linux and windows	3.6.0.437
2	Netperf	Linux and windows	2.7.0
3	LAN speed test	Windows and Mac OS	4.3.1
4	Sockperf	Linux and Windows	3.5.1
5	Uperf	Linux and windows	1.0.6
6	Iperf	Linux and windows	3.6
7	LANBench	Windows	1.1.0

This study selected the Iperf tool for estimating TCP and UDP data transfer capacity performance. The **Iperf** workload/benchmarking tool is utilized for measuring TCP and UDP transmission capacity performance (Somani and Chaudhary, 2009a). This tool quantifies the network connections of virtual machines that send data bundles and receive data packets. Iperf instrument measures this sort of network connections (Wang and Varela, 2011). Written in C++, it makes TCP and UDP data to quantify the throughput of a network that conveys them. It measures throughput between two points that can either be unidirectional or bi-directional. The Iperf instrument runs on Linux and Windows platforms (Walters et al., 2008a). This specific tool is used to assess the bandwidth capacity over a test period, size of data transferred.

To implement Iperf, it needs to be with installing on the server and then on the client computer. This tool is used to measure TCP and UDP bandwidth and throughput performance. After running TCP and UDP test via Iperf on both the server and the client to measure network performance is described in detail as following.

3.3.1.1 TCP Testing

The Iperf command was run in server mode in the server to simulate TCP tests.

root@localhost:/: Iperf -S

```
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
```

The second Iperf command to send the TCP data to the server was run in the client computer in client mode.

```
root@zartec:/home/pratique1# Iperf -c 192.168.0.1 -i 10
```

```
-----  
Client connecting to 192.168.0.1, TCP port 5001  
TCP window size: 86.8 KByte (default)  
-----  
[ 3] local 192.168.0.2 port 33064 connected with 192.168.0.1 port 5001  
[ID] Interval  Transfer  Bandwidth  
[ 3] 0.0-10.0 sec 1.09 GBytes 939 Mbits/sec  
[ 3] 0.0-10.0 sec 1.09 GBytes 939 Mbits/sec
```

A script was written to run the Iperf command multiple times, and the data were copied in an array every time the command ran the average results of the data were saved in an output file. The command was run ten times in each reiteration. The task was repeated up to 10th reiterations to get more data so to confirm the reliability of the data. To test TCP, bandwidth and throughput were measured for the network.

3.3.1.2 Testing UDP

Iperf command was run in the server to start the UDP test.

```
root@localhost: ~ Iperf -s -u
```

The server waited for connection and data from the client computer when the above command was run.

The second Iperf command to send the UDP data to the server was run in the client computer in client mode.

```
root@zartec:# Iperf 192.168.0.1 -I 10 -u -b 900M
```

The command gave results for bandwidth, jitter, throughput, and packet loss on the network, as shown below:

```

Client connecting to 192.168.0.1, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 208 KByte (default)
-----
[ 3] local 192.168.0.2 port 33525 connected with 192.168.0.1 port 5001
[ ID] Interval  Transfer  Bandwidth
[ 3] 0.0-10.0 sec  970 MBytes  814 Mbits/sec
[ 3] 0.0-10.0 sec  970 MBytes  814 Mbits/sec
[ 3] Sent 691784 datagrams
[ 3] Server Report:
[ 3] 0.0-10.0 sec  969 MBytes  813 Mbits/sec  0.018 ms  279/691783 (0.04%)
[ 3] 0.0-10.0 sec  1 datagrams received out-of-order

```

For the experiment, the command was running several times. The same scripts were implemented for both TCP and UDP tests. In the UDP test, more network metrics were measured. The bandwidth throughput, jitter, and packet loss were measured in the UDP mode. The data was saved in an output file to create a table and graphs later on. The average bandwidth was measured for various clients. Results were discussed and analyzed in the next chapter.

3.3.2 Disk performance

Disk performance identifies with the maximum values at which the read and write computational operations are completed by the disk. Workload/benchmark tools are utilized to evaluate disk speed. There are various workload/benchmarking tools available for Disk performance, as seen in Table 4.

Table 4: Tools Available for Disk Performance

S No.	Software tool	Operating System	Version
1	Datamarck	Windows Xp/Vista	0.0.4
2	IOzoneFilesystem	Linux and Windows	3.405
3	Crystal DiskMark	Windows XP/7/8/10	6.0.2
4	DiskBench	Windows all	4.0.0.0

In this study, IOzone Filesystem Workload/benchmark is used (Somani and Chaudhary, 2009). IOzone is used to measure are read and write performance of the disk. IOzone provides a broader system performance that it is portable to any machine. This tool works on any operating system like Windows and Linux. The I/O operations primary workloads for measuring disks performance. Read, write, re-read, re-write are some of the essential operations. IOzone is Written in ANSI C and; it measures performance for both single and multiple streams (Walters et al., 2008a).

IOzone is used to test I/O performance on CentOS 7. IOzone is a known a tool for disk I/O operations in workload/benchmarking and gives the different helpful outcomes to decide the I/O of the disk. To get effective results is to test the diverse file sizes and will use 1 Megabyte, 64 Megabytes, 128 Megabytes, 256 Megabytes, 512 Megabytes until 1 Gigabyte. Each file was made utilizing a record size that is the measure of information composed into a file while a single IO operation is 4 KB. For each size, the standard test was repeated at regular intervals a few times by utilizing a shell script. For more positive outcomes, the test is repeated multiple times, and information is gathered. The average data of the repeated tests were used to present the results graphically. The outcomes were copied to a file that was moved to another shell script to write the data to another file that can be imported to an excel file. IOzone uses options to change file sizes and recordset. These options are described below:

- -s option: measured file size
- -r record size: Kilobytes
- -i: performance type
- I performance kinds of measurements
- -R Excel
- -c time calculation

IOzone tests the several types of properties for the I/O operation that includes: write, rewrite, read, reread, random read, and random write. I/O performance is tested against all the variables of the disk. IOzone results are in kilobytes but are changed into megabytes. An Iozone command ran :

iozone -s 1024M -r 4k -Rac >> Result1.txt

```

1 Iozone: Performance Test of File I/O
2 Version $Revision: 3.394 $
3 Compiled for 64-bit mode.
4 Build: Linux
5 File size set to 1048576 KB
6 Record Size 4 KB
7 Excel chart generation enabled
8 Auto Mode
9 Include close in write timing
10 Command line used: iozone -s 1024M -r 4k -Rac
11 Output is in Kbytes/sec
12 Time Resolution = 0.000001 seconds.
13 Processor cache size set to 1024 Kbytes.
14 Processor cache line size set to 32 bytes.
15 File stride size set to 17 * record size.
16 KB reclen write rewrite read reread random read random write bkwd read
17 1048576 4 538044 835397 3616565 3622749 2348734 748127 3230143
18 record rewrite stride read fwrite frewrite fread fread
19 1068210 2986758 828436 830210 3509842 3517329
20 iozone test complete.
21 Excel output is below:
22 "Writer report"
28 "4"
29 "1048576" 538044
30 "Re-writer report"
31 "4"
32 "1048576" 835397
33 "Reader report"
34 "4"
35 "1048576" 3616565
36 "Re-Reader report"
40 "4"
41 "1048576" 3622749
42 "Random read report"
43 "4"
44 "1048576" 2348734
45 "Random write report"
46 "4"
47 "1048576" 748127
48 "Backward read report"
49 "4"
50 "1048576" 3230143
51 "Record rewrite report"
52 "4"
53 "1048576" 1068210
54 "Stride read report"
55 "4"
56 "1048576" 2986758
57 "Fwrite report"
58 "4"
59 "1048576" 828436
60 "Re-Fwrite report"
61 "4"
62 "1048576" 830210
63 "Fread report"
64 "4"
65 "1048576" 3509842
66 "Re-Fread report"
67 "4"
68 "1048576" 3517329

```

This command gives the results for Iozone tests. On the file, it shows the command, size of the

file, size of the record, time is taken, and kilobytes/second. Secondly, the result shows the 13 operations that Iozone performs during disk I/O operation and the performance for each. The data per second shown in kilobytes is larger and changed to megabytes to be able to interpret it easily. Different formats for Iozone test is used for data collection. Iozone test.sh will repeatedly run the test 70 times and copy results in a text file using the command below.

/iozone.sh

/file separator. Sh iozone.txt

Graphs and the summary that will include meaning, medians, minimum, and maximum values will be generated and discussed in the next chapter.

3.3.3 Memory performance

Memory performance refers to RAM bandwidth and throughput measurements and the capacity to store data. **Ramspeed** is a ram speed testing tool. Ramspeed is good to use as it has the capability of utilizing memory at a lower capacity. It has read-write operations on the memory to run error checks (Hwang et al., 2013). There are various workload/benchmarking tools available for Memory performance, as seen in table 5.

Table 5: Memory Performance Tools

S No.	Software Tools	Operating Systems	Version
1	BenchMem	Windows all/Redhat	0.1.5
2	Memtest86	Windows all/Redhat	V5.0 Beta
3	Memtest	Windows all/Redhat	4.0
4	Memtach	Windows all/Redhat	0.93 Alpha
5	Ramspeed	Linux	3.5.8

Ramspeed has different options to test various memory levels in a computer system. The system which is under tests has more significant physical memory; it is a large block size that is 2 GB and is the maximum allocated size in Ramspeed. Option -b determines the type of test in a particular scenario that can be changed with an integer. Four options are used to test memory performance, which includes integer read and write and float read and write, hence

used from 1 to 4 with -b option. The ramsmc command below is used to test memory with -b1 which is integer writing with a block size of maximum 2 GB.

```
[root@zartec ramsmc-3.5.8]# ./ramsmc -b1 -m 2048
```

The command above puts workloads on memory with Integer write operations by using block sizes from 2 KB up until 2 GB and exponentially incrementing. After every block size, it calculates the bandwidth of memory per second.

```
1 8GB per pass mode, 2 processes
2
3 INTEGER & WRITING 1 Kb block: 37841.87 MB/s
4 INTEGER & WRITING 2 Kb block: 38115.23 MB/s
5 INTEGER & WRITING 4 Kb block: 38123.18 MB/s
6 INTEGER & WRITING 8 Kb block: 38060.18 MB/s
7 INTEGER & WRITING 16 Kb block: 38113.54 MB/s
8 INTEGER & WRITING 32 Kb block: 37852.34 MB/s
9 INTEGER & WRITING 64 Kb block: 33881.47 MB/s
10 INTEGER & WRITING 128 Kb block: 33804.33 MB/s
11 INTEGER & WRITING 256 Kb block: 32621.90 MB/s
12 INTEGER & WRITING 512 Kb block: 26705.57 MB/s
13 INTEGER & WRITING 1024 Kb block: 26019.54 MB/s
14 INTEGER & WRITING 2048 Kb block: 25993.22 MB/s
15 INTEGER & WRITING 4096 Kb block: 26005.46 MB/s
16 INTEGER & WRITING 8192 Kb block: 25998.06 MB/s
17 INTEGER & WRITING 16384 Kb block: 12981.12 MB/s
18 INTEGER & WRITING 32768 Kb block: 11676.65 MB/s
19 INTEGER & WRITING 65536 Kb block: 11558.14 MB/s
20 INTEGER & WRITING 131072 Kb block: 11498.66 MB/s
21 INTEGER & WRITING 262144 Kb block: 11372.35 MB/s
22 INTEGER & WRITING 524288 Kb block: 11195.43 MB/s
23 INTEGER & WRITING 1048576 Kb block: 10753.74
MB/s
24 INTEGER & WRITING 2097152 Kb block: 9966.57 MB/s
```

The primary section in the above outcome is INTEGER, and WRITING is given the - b1 alternative. In the second segment, block size is specifies with exponential increment and maximum of 2 GB on account of -m 2048 alternative utilized in the ram speed command. While in the last segment, memory bandwidth in MB per second is created after the test.

As we watched while the block size expanded the speed of writing diminished. For block size 1 KB to 32 KB, the average writing speed is over 38000 MB for each second. That is presumably L1 cache memory type which is quickest in the framework. From 64 KB to 256 KB, average bandwidth is over 33000 MB and could be an L2 cache of the framework that is second most elevated speed after L1. Further, from block size 512 KB to 8 MB with average bandwidth is over 26000 MB for every second and could be L3. From 16 MB to 2 GB block size the average bandwidth is over 11000 MB for each second, and that could be the smash with the slowest rate. The test is rehashed multiple times, and average outcomes are utilized to ensure the information flawlessness in every one of the four cases that are Integer and Float reading and writing.

To the procedure computerized, a Perl script was written in three documents. The principal record slam speed. Sh runs the `ramsm` command for multiple times with various choices of integer and float with writing and readings and store the information in a text file. The `ramspeed.sh` is run utilizing following command.

```
[root@zartec ramsm-3.5.8]# ./ramspeed.sh
```

In the wake of completing the main test with 25-time iteration, the content record containing the information is passed to another scripting document called `information generator.sh`. This script will isolate the content record as indicated by their unique nature and block. Another script called `ramsingle-record`. Sh then takes this separate file and convert the information in an arrangement that will be later utilized in R for graphical presentation and examination. The accompanying command will be utilized for this reason.

```
[root@zartec ramsm-3.5.8]# ./ramsinglefile.sh ramdata.txt
```

3.3.4 CPU performance

CPU performance measurements appraise the speed at which the CPU (processor) works. For this, unique measurements instruments are used. Unibench is a workload/benchmark tool is utilized in this thesis to measure CPU performance (Babu et al., 2014). Unibench gives better performance when utilized in VMware. Unibench is a reliable tool as VMware ESXi is one of the hypervisors utilized for implementation. It does distinctive operations on the system such as CPU speed tests, floating point operations, etc. It compares speeds of various processors like

AMD Opteron, Intel Core2 Quad, Intel Core i7, and mobile CPUs (Sritrusta et al., 2009). There are various workload/benchmarking tools available for CPU performance, as seen in table 6.

Table 6: List of Available CPU Performance Tools

S No.	Software tools	Operating systems	Version
1	Passmark	Windows all/Redhat	8.0
2	SPECvirt-sc2013	Windows all/Redhat	1.00
3	Former CPU mark	Windows all/Redhat	2.2
4	OCB(openCPU	Windows all/Redhat	0.1.01.07
5	CPU-M workload/benchmark	Windows all/Redhat	1.4.0.0
6	Unibench	Windows all /Linux	3.04
7	Linpack	Windows all,linux	11.1.0.00

Unibench workload/benchmark is used for the CPU test. CPU measurements in VMware and Proxmox were performed using UNIBENCH. UNIBENCH is a tool that performs measurements of CPU utilization in VMware and Proxmox. The application provides statistical data about response time and throughput. This approach was suitable thanks to the reliability and simplicity of the statistical data sets, and it was used to identify the behaviour of telecommunication services. MATLAB software was used to visualize the raw data and for calculation of mean, variance, standard deviation, and margin of error.

3.3.5 Measurement procedure

The maximum traffic load that can be handled by the application had to be determined for the setup in a non-virtual environment. Traffic generated by simulator was being gradually increased until failed requests were detected to determine traffic load limits. Various test cases for low, moderate, and high traffic loads were defined based on the maximum traffic load. In the virtualized scenario, the application was not able to handle all test cases due to the virtualization overhead.

Consequently, the test cases for each setup in a virtual environment were redefined to reflect the traffic limitation. The whole traffic was sent to the active node. In case of the six core setup, the traffic was split into two equal traffic loads which were sent to the active nodes. Each test

case ran for twenty minutes with a constant traffic. The CPU utilization and average response time were measured in ten-second intervals. The test cases were repeated two times.

3.3.6 Validation

The validity of the results was ensured by performing the following steps:

1. All measurements ran for 20 minutes in order to check that the results are stable;
2. All measurements were repeated twice in order to check the repeatability and stability of the results; and
3. Statistical analysis of results was carried out in order to validate the results.

3.3.7 Test cases in a virtualized scenario (CPU)

The same test cases were performed for different configurations of CPU cores and memory resources. Lack of CPU resources is the bottleneck in this experiment, and thus assessment of the impact of using fewer CPU cores in VMs was critical.

In the first test case, one VM was created on each server with dedicated 16 cores of CPU and 24 GB of RAM as in the non-virtualized system. Therefore, a fair performance comparison of these two scenarios was achieved.

In the previous test case, no CPU core resources were allotted to the hypervisor itself. Hence, in the second test case, 12 CPU cores were dedicated to VMs, and four cores were retained for the hypervisor's internal use.

The primary objective of virtualization is to share resources among several instances to better utilize the resources. In the next test case, 12 CPU cores were equally divided between two VMs on each server (6 cores per each VM) with dedicated 14 GB of RAM. The traffic load was also split between two VMs to achieve a fair comparison with the 12 core setup test case. This setup is called the six core setup in this report.

All test cases are defined in the sections below.

3.3.8 VMware

Table 7: VMware - Test cases for the 16 core setup

Test case number	Load [req/s]
1	750
2	2250
3	4500
4	6450
5	7950
6	9450

Table 8: VMware- Test cases for the 12 core setup

Test case number	Load [req/s]
1	750
2	2250
3	4500
4	6450
5	7350
6	7950
7	9450

Table 9: VMware- Test cases for the 6 core setup

Test case number	Load [req/s]	Total Load [req/s]
1	375	750
2	1125	2250
3	2250	4500
4	3225	6450
5	3675	7350
6	4425	8850
7	4725	9450

3.3.9 Proxmox

Table 10: Proxmox- Test cases for the 16 core setup

Test case number	Load [req/s]
1	75
2	2250
3	4500
4	6450
5	7950
6	9450

Table 11: Proxmox- Test cases for the 12 core setup

Test case number	Load [req/s]
1	750
2	2250
3	4500
4	5700
5	6450
6	7950
7	9450

Table 12: Proxmox- Test cases for 6 core setup

Test case number	Load [req/s]	Total Load [req/s]
1	375	750
2	1125	2250
3	2250	4500
4	3225	6450
5	3975	7950
6	4725	9450

3.3.10 Summary

The tools which can be utilized for performance workload/benchmarks are indicated. Different tools can be utilized for performance measurements.

The ideal approach to keep the virtual machine from expending the majority of the resources is to limit the resources allocated to the guest machines. One conceivable strategy to limit the resources is to seclude the resources like network, memory, CPU. Isolating network interfaces are considered here. In like manner, the performance of the virtual machines is contrasted with VMware and other virtualization technologies. For this, resources like CPU use of virtual machines and are considered to indicate performance.

Proxmox and VMware ESXi Hypervisors are used upon which guest Os's are run. Proxmox is a portal for creating, operating and managing and Virtual Machines. Proxmox Virtual Environment (Proxmox VE; short PVE) is an open-source server virtualization environment. It is a Debian-based Linux distribution with a modified Ubuntu LTS kernel and allows deployment and management of virtual machines and containers. Proxmox VE includes a Web console and command-line tools and provides a REST API for third-party tools (Kovari and Dukan, 2012). Two types of virtualization are supported: container-based with LXC (starting from version 4.0 replacing OpenVZ used in version up to 3.4, included), and full virtualization with KVM. It comes with a bare-metal installer and includes a Web-based management interface.

In this section, the research methodology chosen for this exploration think about was clarified. Unique methodologies and methods for performing this study were explained, and reasonable methods were featured. This part additionally portrayed the hypervisors utilized for implementing performance tests on the VMs, and additionally, performance tools were depicted and decided for this theory. The following section covers data on making the experimental environment and the usage of performance tools that are used in this dissertation.

CHAPTER 4

EXPERIMENTS, RESULTS, AND ANALYSIS

4 INTRODUCTION

This section deals with experiments performed as described in the previous chapter. The set-up is shown in the figure below. Sections below show results by utilizing the graphs for each workload/benchmarking tool.

4.1 CONFIGURATIONS FOR HARDWARE AND SOFTWARE USED

Table 13: Hardware Configurations

CPU speed	2.50GHz
Host Memory	24 Gigabytes
Guest Memory	4 Gigabytes
Processor	Intel(R) Xeon(R) CPU E5-26400
Host Disk	2 Terabytes
Guest Disk	40 Gigabytes

Table 14: Software Configurations

Operating System used	CentOS 7
Hypervisors	Proxmox, VMware ESXi
Tool for Network performance	Iperf
Tool for CPU performance	Unibench
Tool for Memory performance	Ramspeed
Tool for Disk performance	IOzone Filesystem Workload/benchmark

4.2 NETWORK

4.2.1 Analysis of Network Performance

The implementation of Iperf was installed on both the server and client. TCP Test and UDP test were done, and the following results were obtained.

4.2.1.1 Average bandwidth (TCP)

Figure 4 demonstrates the comparison of TCP bandwidth among VMware and Proxmox. The bandwidth is nearly the equivalent for Proxmox though VMware platform recorded a lower bandwidth in comparison with Proxmox. Notwithstanding, there is a particular case when all the five clients were sending data, the bandwidth in VMware was somewhat higher than the Proxmox. The trend of the chart demonstrates that when there is a single client, at that point it is connected with most extreme bandwidth however when there are numerous clients simultaneously active the bandwidth is divided between them resulting in less bandwidth per client on the network. The work of (Younge et al., 2011) indicates that there is agreement in terms of results obtained.

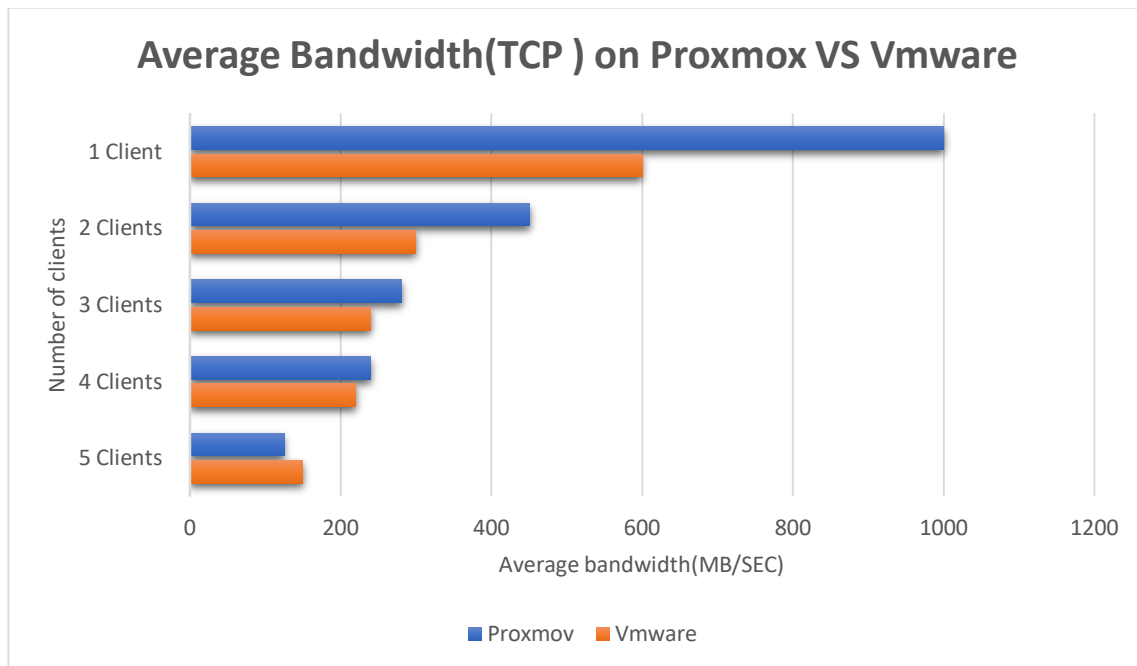


Figure 4: The comparison of TCP bandwidth between VMware and Proxmox

4.2.1.2 Average throughput(TCP)

Figure 5 is merely a comparison between TCP throughput in VMware and Proxmox. Proxmox performance level is almost the same value while VMware showed a lower throughput because of lower bandwidth. The work of (Overby, 2014) indicates that there is an agreement in terms of results obtained.

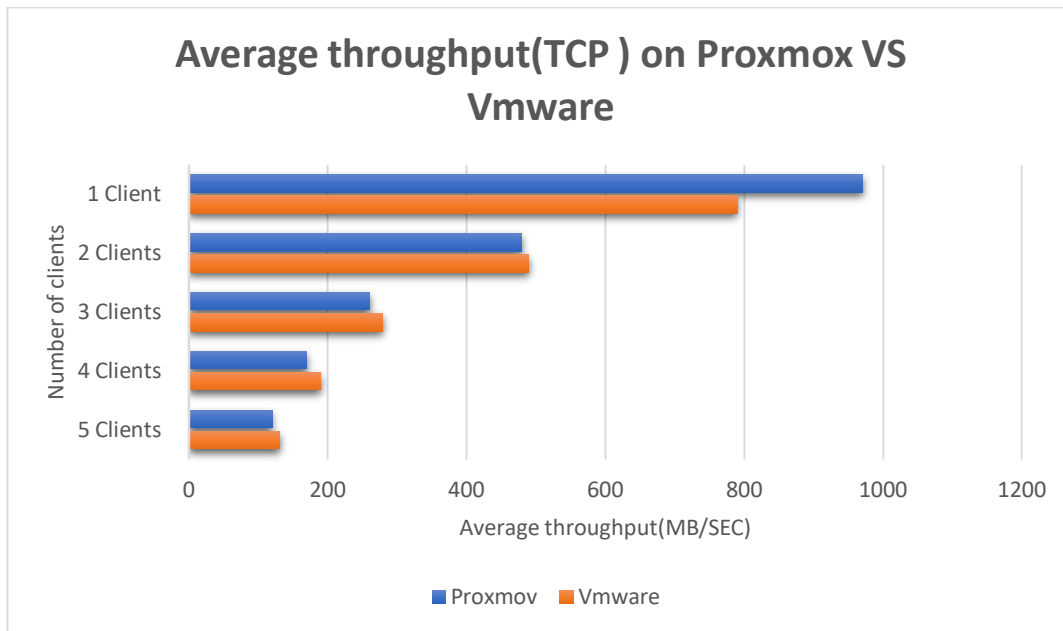


Figure 5: The comparison of TCP throughput among VMware and proxmox.

4.2.1.3 Average bandwidth(UDP)

Figure 6 shows the correlation of UDP bandwidth among Proxmox and VMware. Not at all like in TCP mode, VMware recorded somewhat higher bandwidth than Proxmox. This can be viewed as an outlier. The bandwidth is most extreme for a single client, and it is appropriated to different clients when they have connected effectively on the network. The work of (Overby, 2014) indicates that there is an agreement in terms of results obtained.

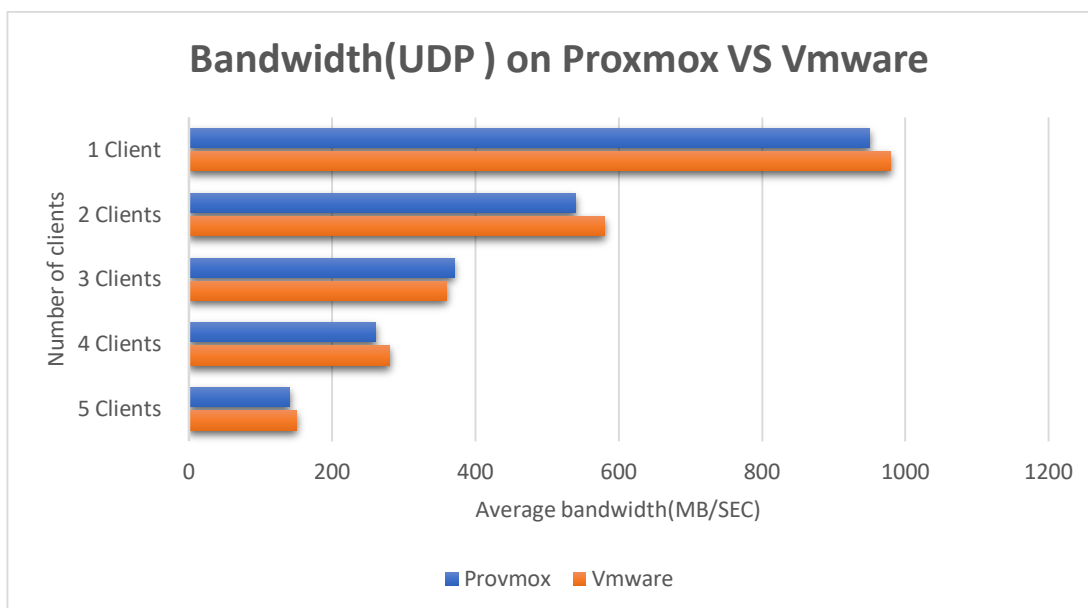


Figure 6: The comparison of UDP bandwidth among VMware and Proxmox

4.2.1.4 Average throughput(UDP)

Figure 7 depicts the comparison between VMware UDP throughput and Proxmox UDP throughput. Throughput trend followed the trend of bandwidth, that is why VMware has a slightly higher throughput than Proxmox. In client 3; VMware had less bandwidth than Proxmox, which is an exception or some abnormal behaviour in client 3. Reference to (Overby, 2014) reveals that the results obtained below are similar and follow the same protocols in terms of measuring UDP throughput.

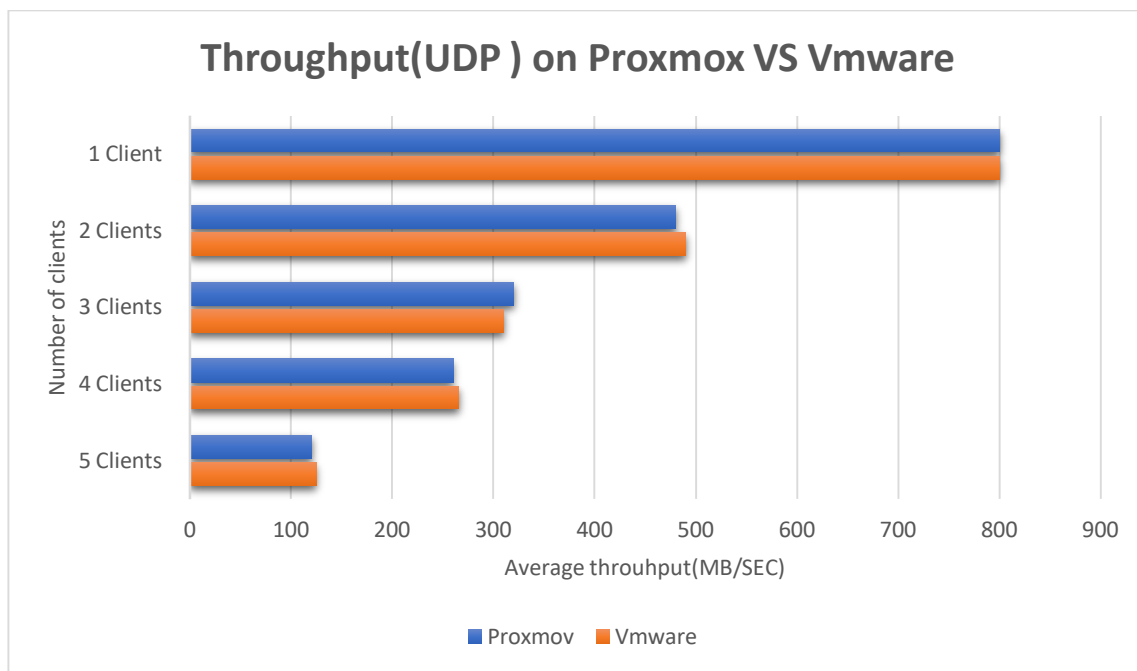


Figure 7: The comparison of UDP throughput among VMware and Proxmox

4.2.1.5 Datagram loss in UDP

Figure 8 depicts the comparison of datagram loss between Proxmox and VMware. The graph shows enormous datagram loss in VMware. Datagram loss is negligible for bare metal, and slight datagram loss is recorded in Proxmox when transferring large amounts of data. Loss in the datagram is inversely proportional to the number of active clients on the network. As the number of client increases, then the datagram loss lowers down. Reference to (Tsugawa et al., 2009) reveals that the results obtained here are similar and follow the same protocols in terms of measuring UDP loss.

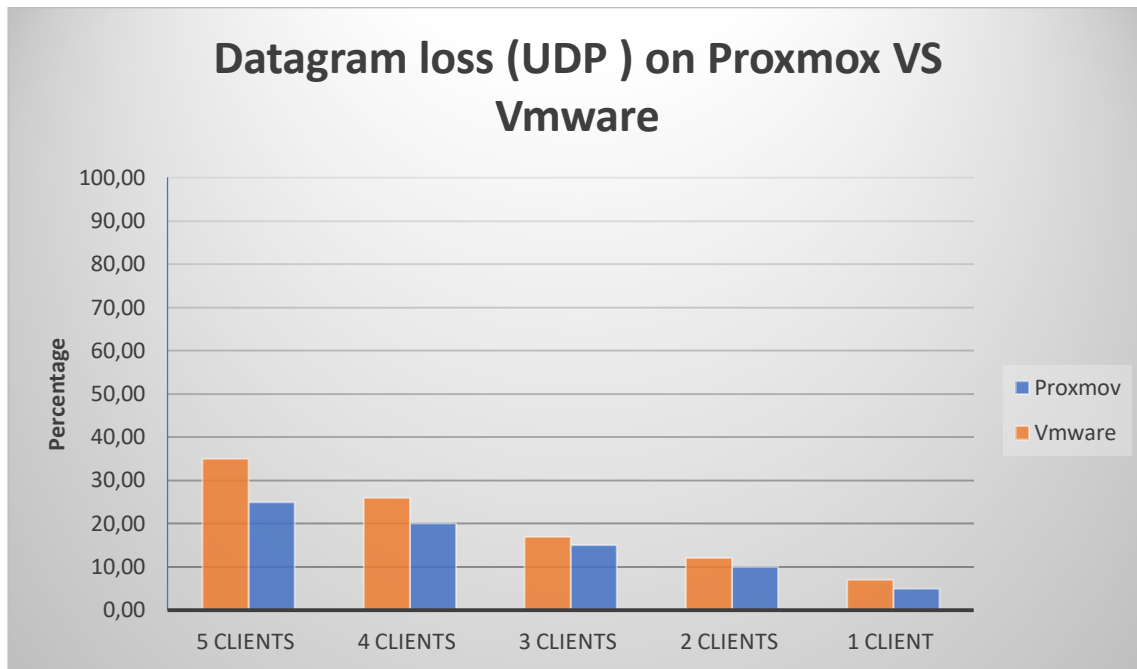


Figure 8: The comparison of datagram loss between Proxmox and VMware.

4.2.1.6 Average jitter Test

Figure 9 compares the jitter between Proxmox and VMware. Jitter on VMware was higher than Proxmox. Proxmox recorded some jitter but almost negligible in comparison with VMware. Jitter increases with the increase of some clients on the network, as you can see on the graph. generally, this is correct, but sometimes a little bit deviation is also observed in Proxmox. Reference to(Cheng et al., 2011) reveals that the results obtained here are similar and follow the same protocols in terms of measuring jitter.

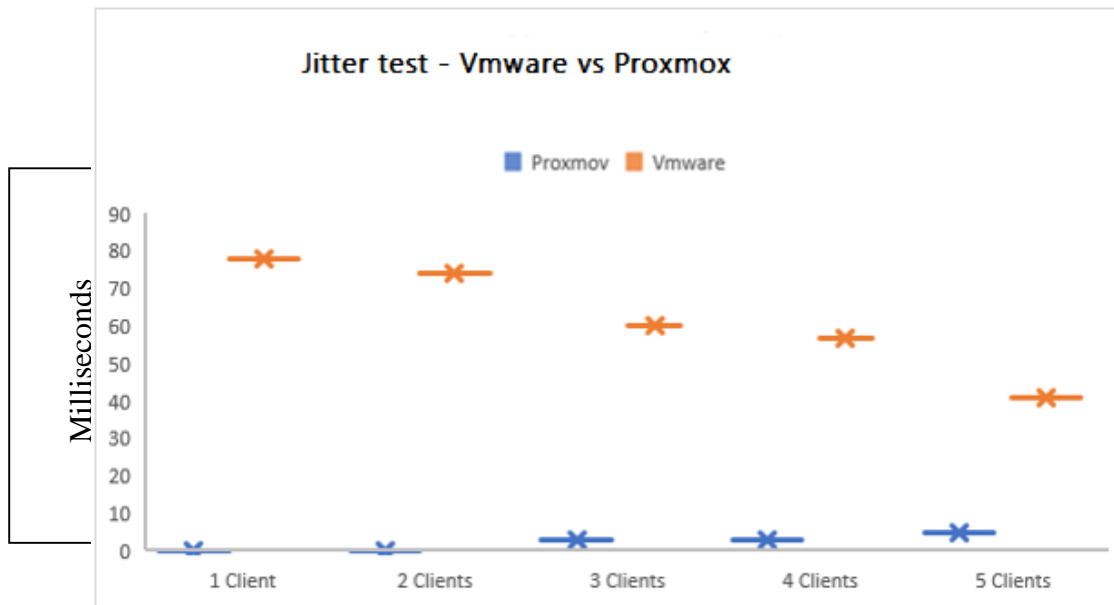


Figure 9: The comparison of jitter between VMware and Proxmox

4.2.1.7 Maximum average request before saturation

Figure 10 compares the maximum number of requests sent by the client before the server reaches saturation; this is between VMware and Proxmox. The lowest request sent by the client is observed in Proxmox. VMware performance is slightly better than Proxmox. More the number of clients the less is the maximum number of requests sent by clients to the server(Ahmed et al., 2008).

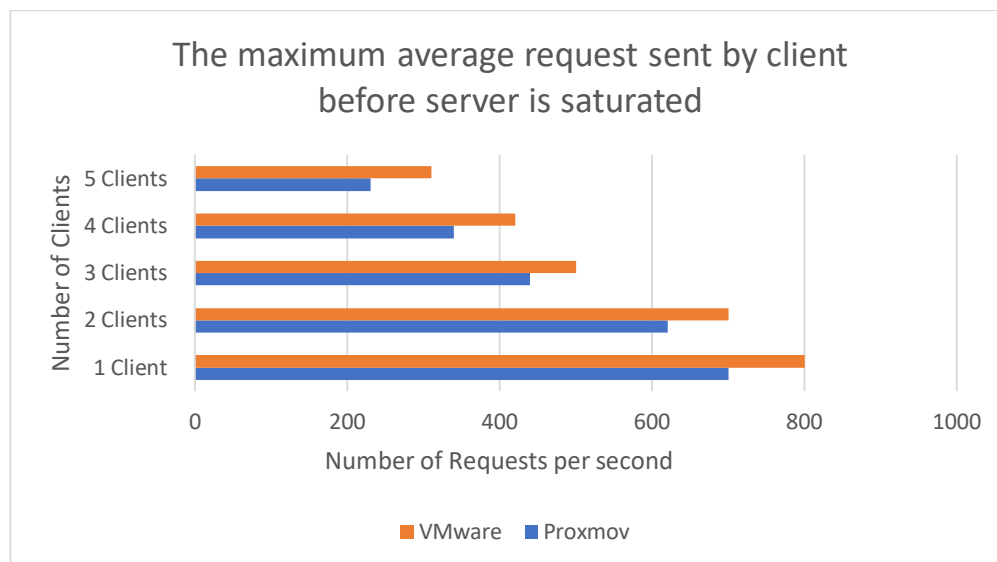


Figure 10: The comparison of a maximum number of the requests sent by clients between VMware and Proxmox

4.2.2 Network Conclusion

In accordance to other results from other authors (Gulati et al., 2010, Guo et al., 2016), it very well may be seen that the outcomes above represent the comparison of average TCP bandwidth and UDP bandwidth on Proxmox and VMware virtualization. Bandwidth for an isolated client in TCP test is higher than UDP. At the point when more clients are available on the network sending data, the bandwidth is distributed between clients. After adding a second client machine, UDP has marginally higher bandwidth than TCP. This is because when less data is transferred, UDP is superior to TCP because UDP does not perform three-handshaking or any acknowledgement process. There is a comparison between the average TCP and average UDP throughput. At the point when there is more than one client active on the network and sending data, the throughput of UDP is higher than TCP throughput. It very well may be anticipated that the higher bandwidth prompted higher throughput in UDP data transfer.

The essential result of this task is the investigation of the impact of the virtualization on the performance of the computer network. The idea behind this experimental setup is to observe the performance of the network in the virtualized environment. After a reasonable vision performance level, it is significantly more advantageous to make sense of the impact of the virtualized server on the network. The measurement of the performance of a server is a beneficial undertaking achieved in this task. To measure virtual computer network performance, Iperf was the tool that was used. By implementing Iperf, the maximum bandwidth with which the clients were connected on the network and maximum throughput were measured in the TCP data transfer.

Jitter and datagram loss on the network along with maximum bandwidth and throughput was measured in UDP data transfer mode.

Proxmox and VMware ESXi were the deployed hypervisors. There were five virtual clients, and a virtualized server was provisioned for a small private datacenter. The performance was close to the physical machine level. However, there was some deviation on VMware experimental setup. There was Datagram with the abnormal loss, which showed that data transfer was not as reliable for VMware in UDP mode. Jitter, which is the amount of variation in latency/response time on a network, was found to be higher on VMware in comparison to Proxmox.

On VMware during TCP data transfer, low bandwidth and throughput were recorded. Jitter and datagram loss was higher in the virtual environment compared to the physical machine. To measure the performance of the network, the response transferred by the server and the clients were stored in a text file for graphs and analysis. VMware had a slightly better result over Proxmox.

When the number of active virtual machine increases, less bandwidth becomes available for the virtual machines resulting in fewer throughput. The graph shows that datagram loss is high when there is only one client. By increasing virtual machines, the datagram reduces, and when it reaches the fifth virtual machine, it is an almost negligible amount of datagram. When small amounts of data are being sent by the virtual machine, then the datagram loss is fewer. We can conclude that when the virtual clients are supposed to spend fewer amounts of data, then UDP is efficient, but TCP can be better than the UDP when a bulk of data is required.

4.3 DISK

The implementation of IOzone was done in the previous chapter, and the following results were obtained. The review of the data in graphical format for discussion and analysis and is seen below. In the last section, the analysis of the data was made using various statistical methods.

After 70 running Iozone tests 70 times, the average data was obtained. To generate the combined graphs for Proxmox and VMware, the average values were used. IOzone was used to put the workload on various guests to measure disk I/O performance. To calculate the performance with different workloads, file sizes ranging from 1 Megabyte, 64 MegaBytes, 128 MegaBytes, 256 MegaBytes, 512 MegaBytes, to 1 Gigabyte were used. The graphs below show the performance of Proxmox and VMware virtual machines. In accordance with other results from (Gschwandtner et al., 2011, Ha et al., 2016), it can be seen that the results are measured the same way and have a similar outcome.

4.3.1 Analysis of Disk performance

4.3.1.1 Write



Figure 11: Iops average write

4.3.1.2 Re-Write

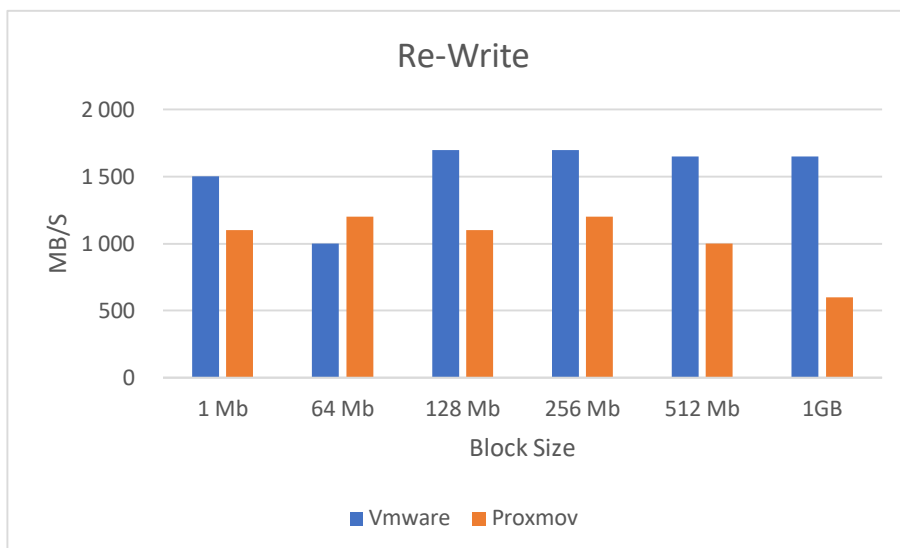


Figure 12: Iops average re-write

Writing and re-writing performance Iops tests was shown in figure 11 and figure 12. Performance of Virtualization remained high between VMware and Proxmox. The performance for VMware was similar to that of Bare Metal systems. It was observed that both

the two graphs in figure 11 and 12 show an increase in performance while the file size increased from 1 MB to 1 GB. Re-writing in fine is usually faster than file writing. Evidence from the graphs shows that re-writing was almost twice faster than writing. For the 64 MB file size, Proxmox shows 12% better performance than VMware.

4.3.1.3 Read

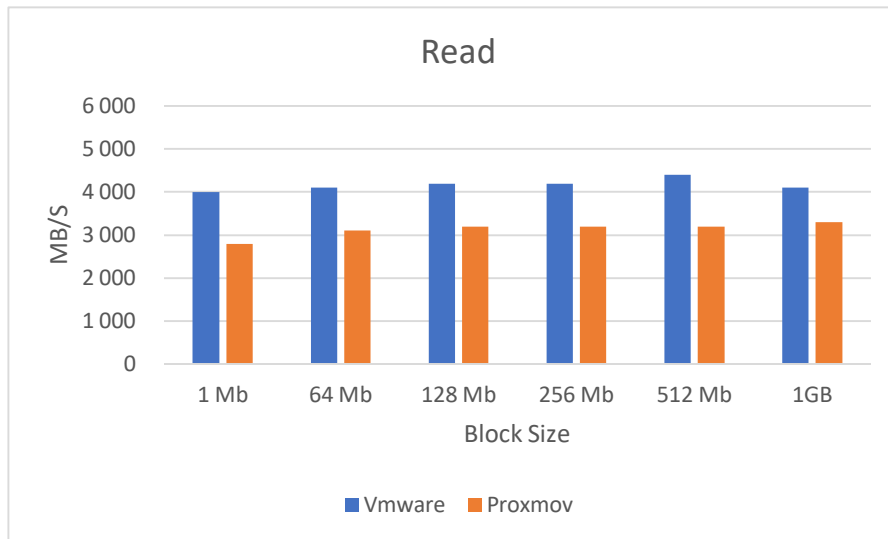


Figure 13: Iozone average read

4.3.1.4 Re-Read

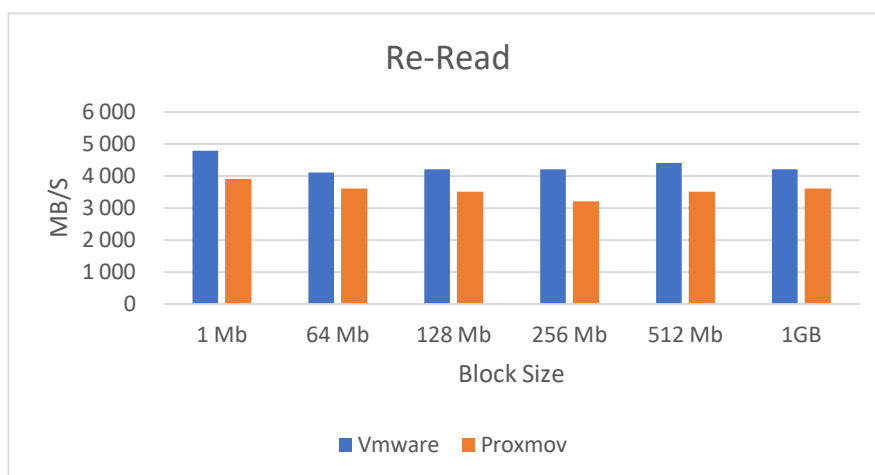


Figure 14: Iozone average re-read

Re-read is faster than reading because the file is cached in its system. In figure 13 and figure 14, it showed that the performance of re-read was more efficient than reading. VMware's performance increased in the case of reading from file size 1 MB to 1 GB. In re-read, there was

increment from 64 Megabytes to 1 Gigabyte file size. Proxmox continuously increments for reading and decrements for re-read when the file size is increasing from 1 Megabyte to 1 Gigabyte. Read performance for VMware remained constant while the file size increased. In smaller file sizes for reading, VMware gave an outstanding performance, which was in line with Bare Metal performance.

4.3.1.5 Random Read

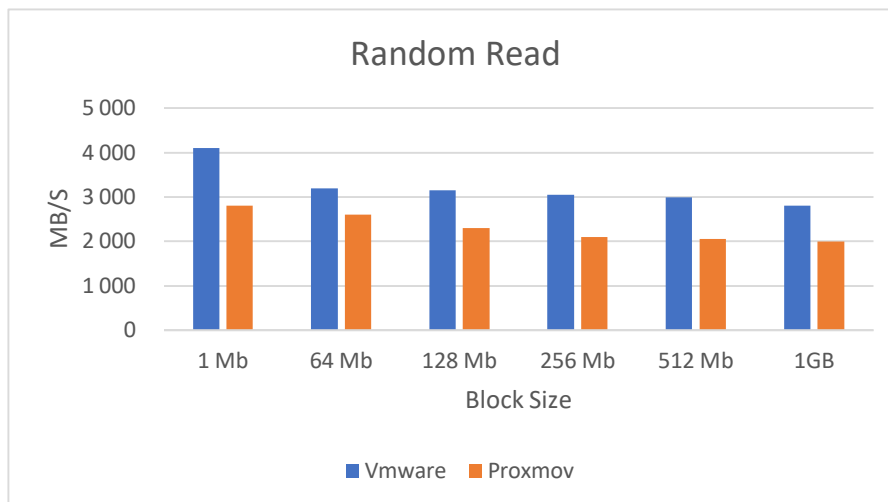


Figure 15: Iozone average random read

Random read in figure 15, shows Proxmox and VMware had a continuous decrease in performance while the size of the file increased. VMware performed better than Proxmox with the difference of 28%, 37%, 36%, 42% and 37% for file sizes 64 MB, 128 MB, 256 MB, 512 MB, and 1 GB respectively.

4.3.1.6 Random Write

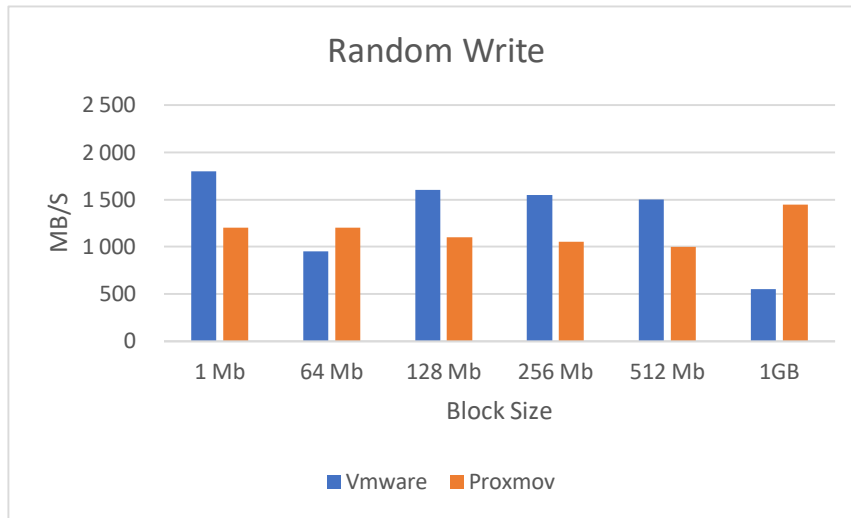


Figure 16: Iozone average random write

Figure 16 shows that Proxmox outperformed VMware with an average of 46% in all file sizes. Proxmox performs better than VMware for 64 MB file size.

4.3.1.7 Discussion of Iozone test results

With 13 different kinds of tests, Iozone measures disk I/O performance. Virtualization guest performance is usually lower as compared with Bare Metal because it adds overhead due to layer abstraction. There is room for the improvement of virtualization technologies to reduce the overhead for better performance. As illustrated in the graphs above, VMware had better performance than Proxmox. Some instances were unusual especially in the case of 64 MB with write, rewrite and random write where proxmox performed better than VMware

In the case of large file sizes, the Proxmox had the worst performance. In most of the cases for a 1 GB file size, the proxmox performance was three times poorer than VMware. For smaller file sizes, Proxmox performed better, but VMware had less overhead. For all kinds of reading Proxmox had better performance.

4.3.1.8 Consolidated Iozone results

Iozone tests performed are either write or read. Write tests include write, re-write, random write, record rewrite, forward write and re-forward write whereas read tests include reading,

re-read, random read, backward read, stride read, forward read, re-forward read. All the write and read test results are tabulated in different file sizes.

4.3.1.9 Write performance (consolidated)

Iozone writes performance tests were consolidated, and the percentage of Proxmox and VMware was calculated. The sum of all Iozone write tests was added converted into a percentage. Bare Metal was a base to calculate the percentage of write for Proxmox and VMware. The values are given in table 15 in percentages.

Table 15: Consolidated Write performance of Iozone test

Table 15: Write performance tests

Size of File	Proxmox	VMware
1 MB	62.0 %	92.4 %
64 MB	78.1 %	57.3 %
128MB	75.1 %	95.5 %
256MB	68.5 %	88.5 %
512MB	66.4 %	85.7 %
1GB	36.2 %	83.8 %

The consolidated graph for Iozone writes test presented regarding percentage.

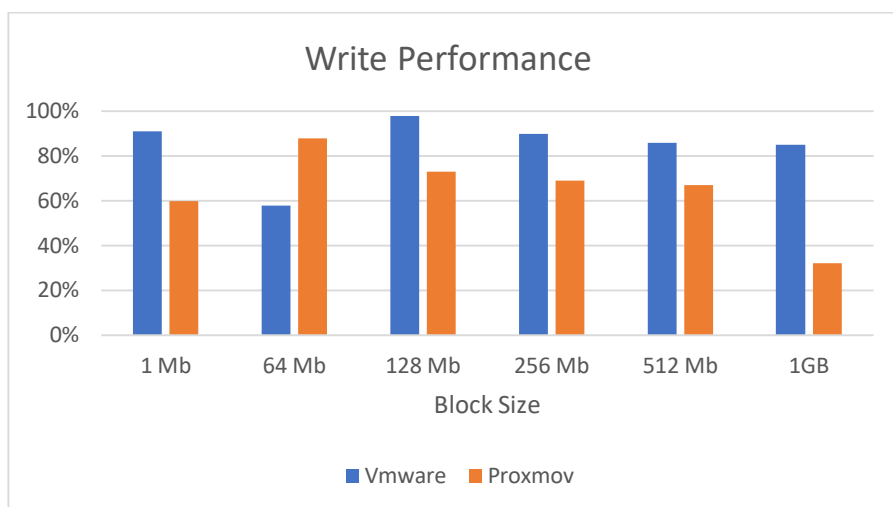


Figure 17: write performance

In almost all the file size, VMware has had better performance compared to Proxmox. For a file size of 64 MB, Proxmox had 89% performance, and VMware was at 68%. In case of VMware with a file size of 1 GB, the performance was 86% and was more than twice as fast compared to Proxmox. VMware shows the best performance for 128 MB file size.

4.3.1.10 Read performance (Consolidated)

Read performance for Iozone tests for Proxmox and VMware were consolidated and analyzed. Iozone read tests were added, and the calculation for Proxmox and VMware percentage was obtained using Bare Metal as a base. Table 16 shown below with figures in percentages.

Table 16: Read performance tests

Sizes of file	Proxmox	VMware
1 MB	66.5 %	92.1 %
64 MB	83.2 %	100 %
128MB	78.4 %	100 %
256MB	68.8 %	86.8 %
512MB	68.9 %	84.5 %
1GB	68.5 %	82.5 %

Consolidated Iozone read test are in graph form below:

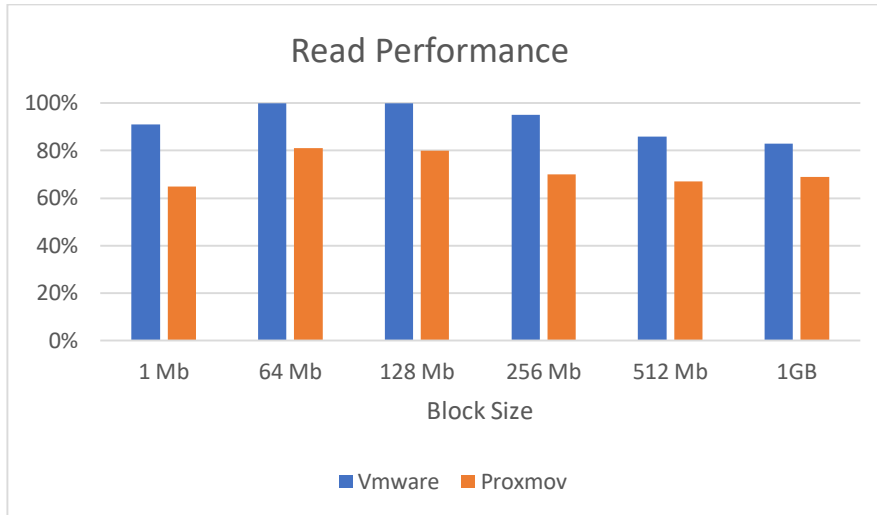


Figure 18: Consolidated read performance

4.3.2 Disk conclusion

VMware outperformed Proxmox in all Iozone file sizes for consolidated read performance. 64 MB and 128 MB files sizes for VMware performance were identical to the Bare Metal system. In a 1 MB file size, Proxmox had a 66% performance rate compared to Bare Metal. Reference to (Ha et al., 2016, Xavier et al., 2015, Chen et al., 2016, Li et al., 2013) reveals that the results obtained here are similar and follow the same protocols in terms of measuring disk performance.

4.4 CPU

While increasing the traffic, the CPU utilization is growing almost linearly. It means that the system behaviour is predictable regarding CPU utilization. On the contrary, the slopes vary for different core setups. System behaviour can be divided into low and high traffic scenarios based on the intersection point. The 16 core setup has poor overall performance. For the low traffic environment, the 12 core setup would be the best replacement for the non-virtualized environment. As regards the high traffic scenario, the six core setup has the lowest CPU utilization, even lower than the non-virtualized system.

Moreover, it is capable of handling more traffic. Therefore, it seems better to have several instances with fewer CPU cores for high traffic loads. The results below are in accordance to

results from other authors, (Gupta et al., 2006, Hwang et al., 2013, Wang and Varela, 2011) and it can be seen that they follow the same format.

4.4.1 Vmware

4.4.1.1 CPU utilization

Figure 21 shows CPU utilization for different core setups. As can be seen, CPU utilizations differ depending on the number of cores.

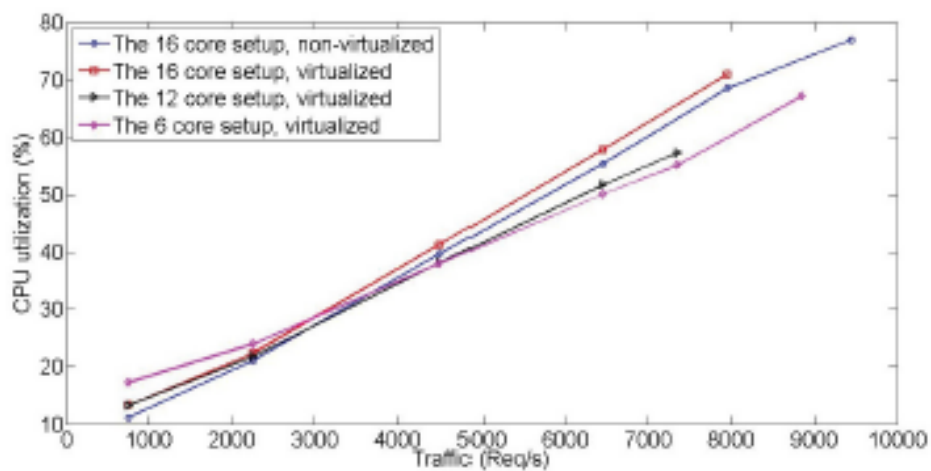


Figure 19: VMware - CPU utilization

4.4.1.2 Average response time

Figure 22 shows the average response times for different core setups. While increasing the traffic, the average response times are growing exponentially. As it can be observed, the six core setup performed the best among virtualized scenarios regarding average response time and handling the maximum traffic load.

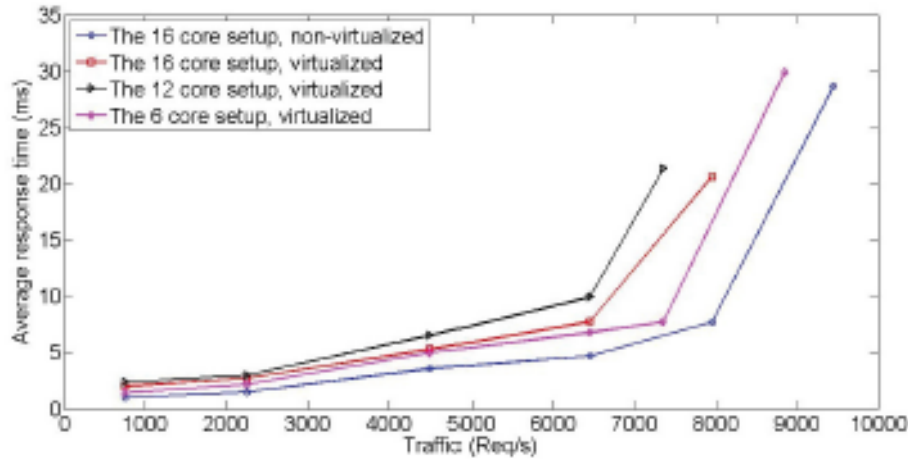


Figure 20: VMware - Average response time

4.4.2 Proxmox

CPU utilization for Proxmox also shows an almost linear pattern. The values are much higher than in the non-virtual scenario. This fact affects the maximum traffic which could be handled by the system. Among the practical scenarios, there is also an intersection point in CPU utilization, but in a higher traffic range than for VMware.

The 12 and the 16 core setups had quite similar behaviour. However, the 16 core setup would be a better replacement for the non-virtualized environment since it performed better regarding maximum traffic, and the CPU utilization difference is insignificant. The six core setup has a high CPU utilization, and it would only be a good option for high traffic loads because it shows less CPU utilization in comparison with the 16 core setup.

4.4.2.1 CPU utilization

Figure 23 shows CPU utilization for different core setups.

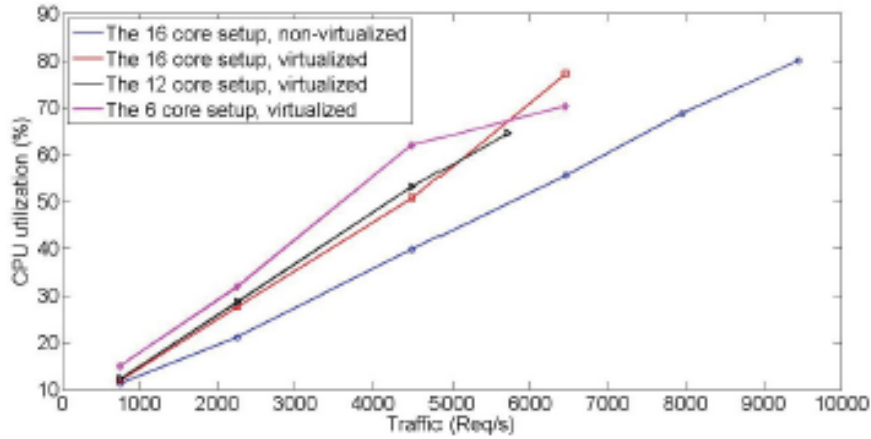


Figure 21: Proxmox - CPU utilization

4.4.2.2 Average response time

Figure 22 shows the average response times measured for different core setups. Average response times follow the exponential pattern as the traffic increases. The 16 core setup has the best performance among virtualized scenarios regarding average response time. The 12 core setup has a better response time in low traffic, while the six-core setup performed better in a high traffic. However, the non-virtualized system had always lower response time.

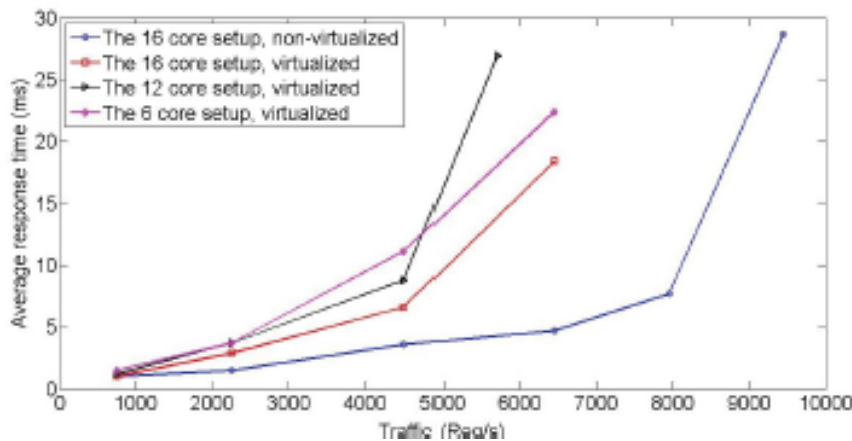


Figure 22: Proxmox - Average response time

4.4.3 Result summary

Since the application is CPU-dependent, availability of CPU resources becomes more vital for higher traffic loads. Moreover, the system also must perform the handling and scheduling of the CPU resources simultaneously. The experimental measurements have shown that the six

core setup provides the best performance for high traffic. This is most probably caused by less fight over the CPU resources (Ramalho and Neto, 2016).

Differences in performance behaviour between the two hypervisors were noticeable. The significance of this finding lies in identifying the strengths and weaknesses of each hypervisor. Thus telecommunication companies can use this information to select and employ the most suitable hypervisor based on the requirements.

Results of the experiments are summarized below. Three traffic loads were selected from the middle of the traffic spectrum since the system performs better and is more stable in that range.

The following tables provide the comparison of results

4.4.3.1 Various CPU core setups

Table 17: VMware- CPU utilization overview

	2000 [req/s]	3500 [req/s]	5000 [req/s]
The 16 core setup	22%	35%	48%
The 12 core setup	21%	31%	43%
The six core setup	24%	33%	42%

Table 18: Proxmox- CPU utilization overview

	2000 [req/s]	3500 [req/s]	5000 [req/s]
the 16 core setup	25%	41%	58%
the 12 core setup	25%	42%	58%
the six core setup	29%	50%	65%

Table 19: VMware- Response time overview

	2000 [req/s]	3500 [req/s]	5000 [req/s]
the 16 core setup	3ms	4ms	7ms
the 12 core setup	3ms	5ms	8ms
the six core setup	2ms	4ms	6ms

Table 20: Proxmox- Response time overview

	2000 [req/s]	3500 [req/s]	5000 [req/s]
the 16 core setup	3ms	6ms	10ms
the 12 core setup	4ms	8ms	17ms
the six core setup	4ms	9ms	14ms

4.4.4 CPU Conclusion

This master thesis presents a comparison study of different virtualization technologies and their impacts on performance. Performance tests were conducted in virtualized environments in order to investigate the effects of virtualization. Moreover, various testbed configurations were used to clearly distinguish which of the hypervisors better complies with the requirements of cloud computing.

Two questions were formulated to get a clear answer on the CPU configuration performance:

How does virtualization impact the response time and CPU utilization of telecommunication services?

In order to obtain comparable statistical data from both virtualized and non-virtualized systems, the same number of CPU cores and RAM was used during the tests. As can be seen from the tables above, virtualization adds overheads to both CPU utilization and response time.

Which hypervisor has better performance regarding migration, response time, and CPU utilization?

VMware demonstrated better performance regarding CPU utilization and response time. Therefore, VMware would be a better choice for telecommunication services sensitive to CPU resources and response time. On the contrary, Proxmox has shown less downtime in comparison to VMware, which makes it more suitable for large environments where maintenance, fault-tolerance, and manageability are essential. The work of (Li et al., 2013) indicates that there is an agreement in terms of results obtained. Many other authors have obtained similar results, and outcomes follow the same principle (Morabito, 2017, Ramalho and Neto, 2016).

4.5 MEMORY

4.5.1 Ram speed

Results for ramspeed are shown below. Ramspeed test was repeated 25 times using a script for data reliability. The test was done using exponential of 2 KB block size with a maximum of 2 GB. Test results were calculated into average and converted in the following graphs. The results below are in accordance to results from other authors (Wang et al., 2015, Wu et al., 2016), they agree with the type of data obtained for memory performance measurement.

4.5.2 Integer and Writing

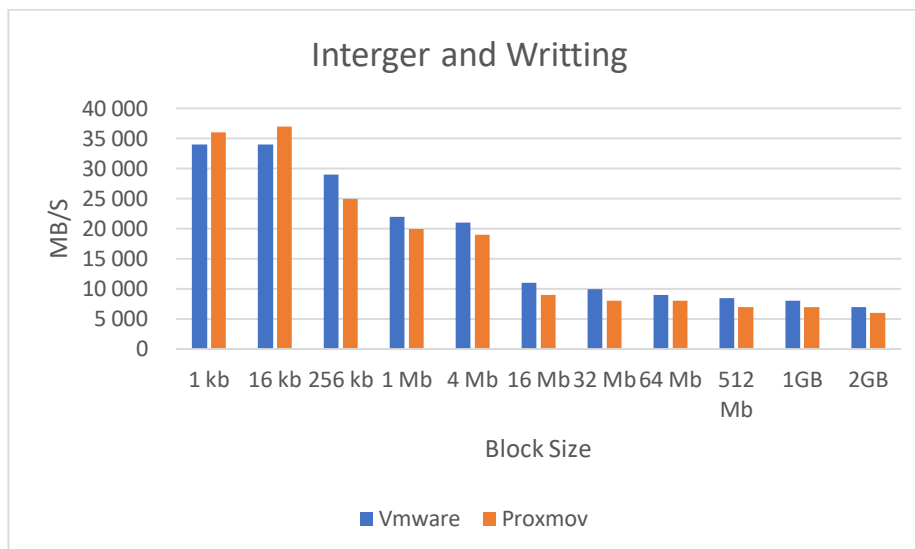


Figure 23: Ram speed average integer and writing

4.5.3 Integer and Reading

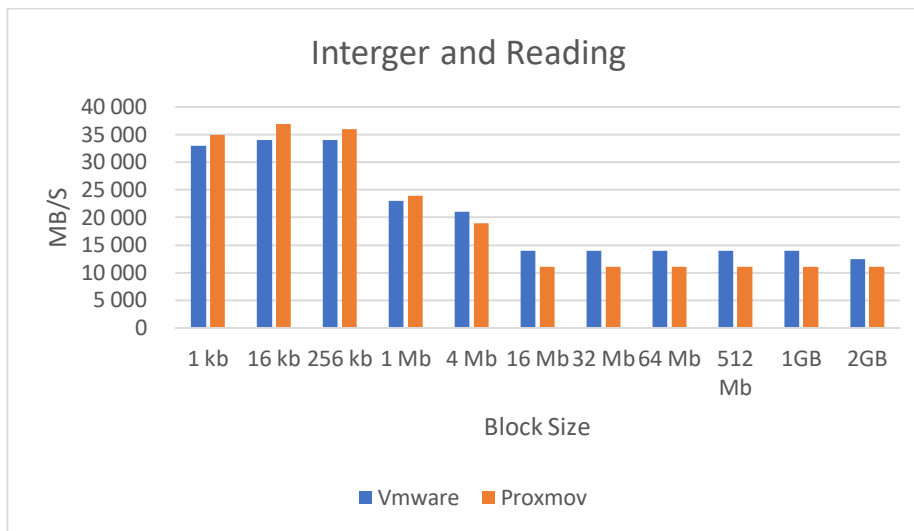


Figure 28: Ramspeed average integer and reading

4.5.4 Float and Writing

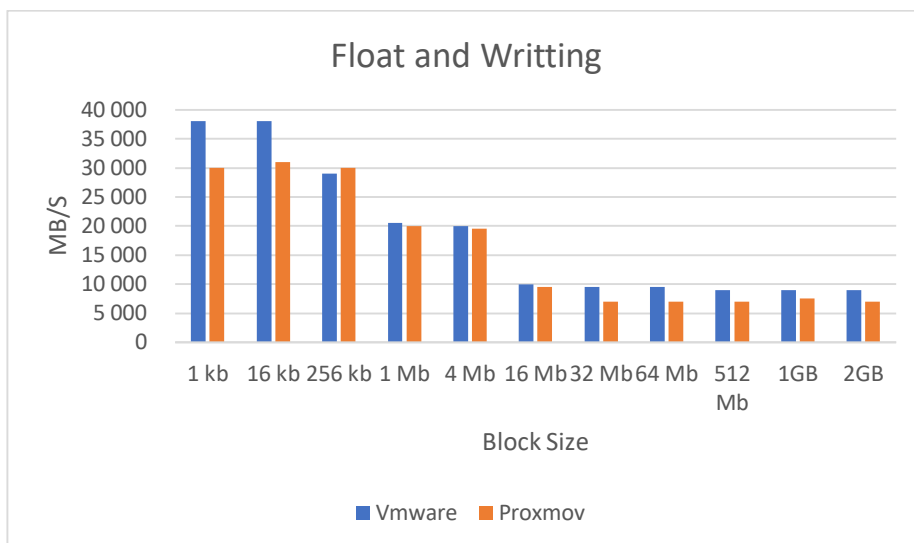


Figure 24: Ramspeed average float and writing

4.5.5 Float and Reading

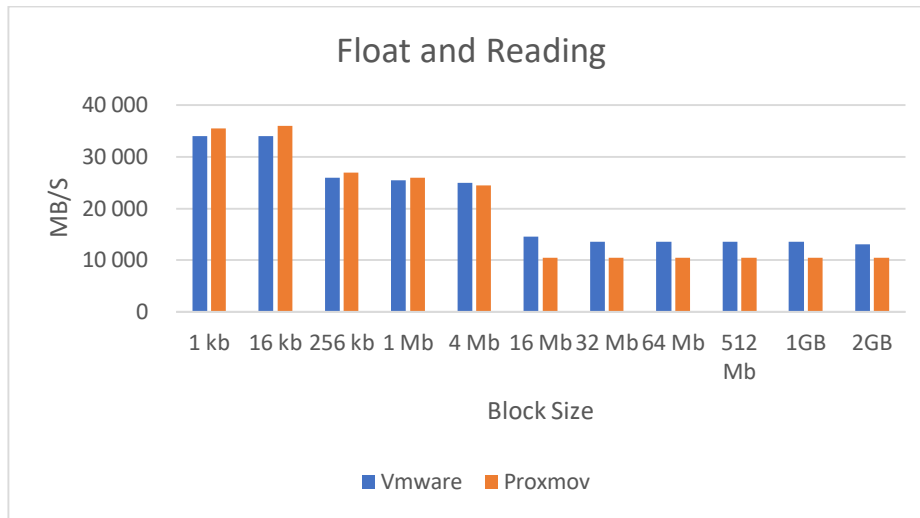


Figure 25:: float and reading average for ramspeed

4.5.6 RamSpeed test results in Discussion

Ramspeed uses four various kinds of tests that include Integer reading and writing, Float reading and writing to measure memory performance. Virtualization technologies for Proxmox and VMware resembled almost the same performance as compared to physical servers.

The performance of both was different with different memory block sizes. With a block size smaller than 1 MB, Proxmox performs better than VMware. Block size larger than 1 MB, VMware performance was better than Proxmox. In the case of float and writing with 4 MB block size, Proxmox and VMware performance were almost the same.

4.5.7 Overall Ram speed results

Writing and reading results for Proxmox and VMware were calculated into a consolidated percentage.

4.5.7.1 Integer and float writing performance (Consolidated)

Table 21 shows the Proxmox and VMware performance for writing.

Table 21: integer and float writing results

Size of File	Proxmox	VMware
1 KB	93.0 %	87.6 %
16 KB	94.6 %	89.3 %
256 KB	90.8 %	87.6 %
1 MB	76.6 %	80.0 %
4 MB	75.4 %	78.7 %
16 MB	71.5 %	81.6 %
32 MB	70.6 %	84.6 %
64 MB	69.8 %	83.7 %
512 MB	70.3 %	86.3 %
1 GB	68.5 %	86.2 %
2 GB	67.5 %	84.8 %

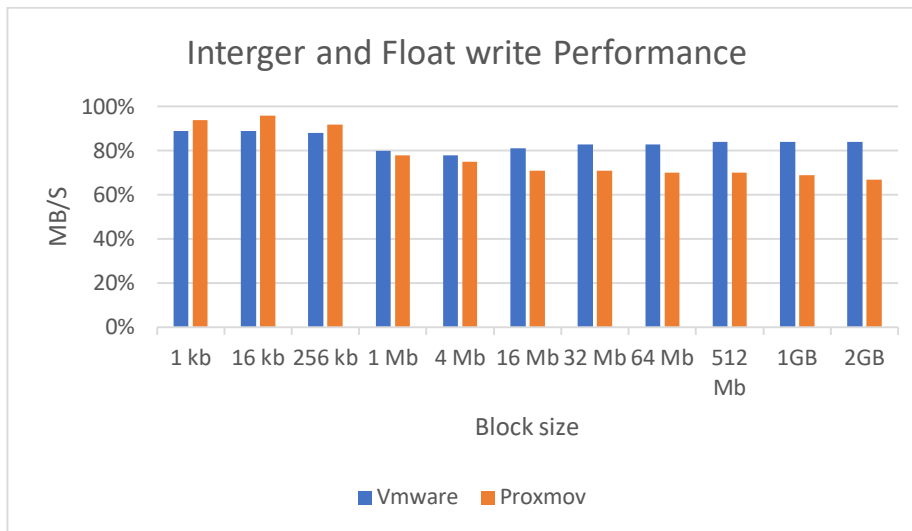


Figure 26: Integer and Float writing

Proxmox writing performance in small block sizes was better than VMware and remained at 93%. Whereas the performance of the Proxmox with block size larger than 1 MB, remained constant with an average of 71%. The performance of VMware with a block size smaller than 1 MB was under 91%. Continuous rise in VMware performance was observed from 1 MB block size.

4.5.7.2 The consolidated performance of integer and float reading

Table 22 shows Proxmox and VMware performance for reading.

Table 22: integer and float reading results

File size	Proxmox	VMware
1 KB	93.8 %	88.6 %
16 KB	95.8 %	89.1 %
256 KB	94.3 %	89.2 %
1 MB	93.6 %	90.4 %
4 MB	85.9 %	88.9 %
16 MB	67.1 %	84.7 %
32 MB	69.4 %	85.3 %
64 MB	71.6 %	85.7 %
512 MB	71.9 %	85.6 %
1 GB	72.3 %	85.7 %
2 GB	71.4 %	84.4 %

The performance of Proxmox and VMware is shown in Figure 29

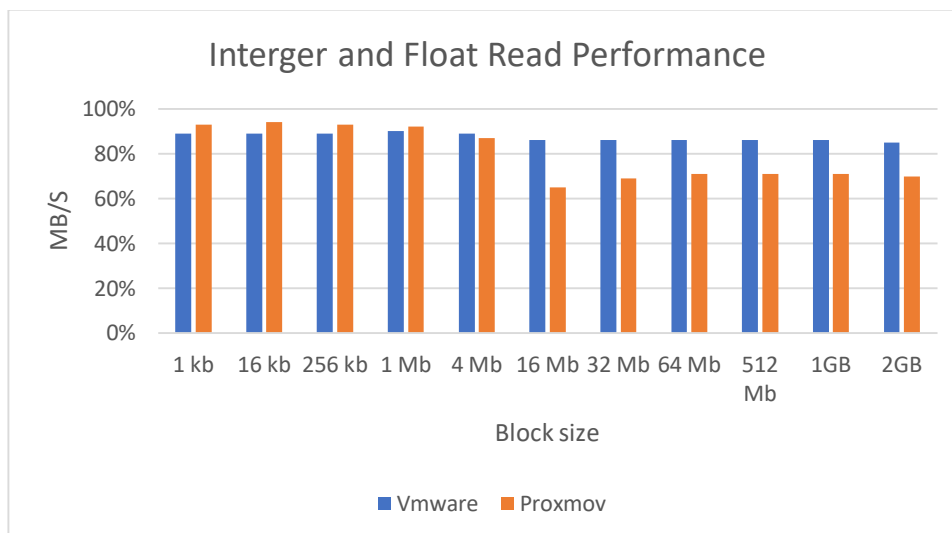


Figure 27: Integer and Float reading

4.5.8 Memory conclusion

Consolidated Proxmox performance was better when compared with VMware in smaller block size for reading in a ram speed test. VMware performance remained 86% in larger block sizes. In larger block sizes, Proxmox remained at 71%. Reference to (Huber et al., 2011, Wang and Varela, 2011) reveals that the results obtained here are similar and follow the same protocols in terms of measuring memory performance.

4.6 CHAPTER SUMMARY

Chapter 4 presented data in graphical view as well as analysis and discussion. Because of variation in data, results from Iozone and Ram Speed were collected after a large number of runs. For UnixBench, the data was collected after three runs as it had a small number in variation. Perl scripts helped to collect data (Wu et al., 2016). The Above gives the data transfer capacity of the network in every one of the virtual machines. Iperf provided the measure of data transferred at specific interims of time. Iperf yielded results of data transferred, data transfer capacity rate, and time intervals. The results above in comparison with other authors, it can be seen that most virtual machine workloads and results behalf in the same manner with variations in terms of hardware and software configurations (Hwang et al., 2013, Somani and Chaudhary, 2009b).

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5 INTRODUCTION

If the optimum resource utilization were taken as a factor, the virtual servers would be in the winning situation provided that the total demands regarding resource and capacity have been correctly outlined before the virtualization infrastructure implementation. This critical aspect was also supported by the CPU capacity and processing load demanded by the physical and the virtual servers where the virtual servers always demanded more resources compared to those of the physical servers. Different approaches to virtualization resource allocation and configuration created the difference in performance.

5.1 DISCUSSION

It was observed that the Bare Metal resembles Proxmox and VMware in most of the tests. Comparing Proxmox and VMware, impressive results were observed. For Iozone, writing large files, VMware was more than twice as fast as Proxmox. It was observed that Proxmox was more than 31% better than VMware in writing 64 MB files. VMware performs better than Proxmox in reading. VMware was 21 to 26% better than Proxmox for smaller file sizes. VMware was 41% better than Proxmox for 1 MB file.

With ramspeed memory performance with a block size smaller than 5 MB, Proxmox performs 5 to 8% better than VMware. Block size larger than 5 MB, VMware performs 16 to 26% better than Proxmox. VMware was 31 to 51% better than Proxmox In the case of writing. UnixBench was used to measure CPU performance. VMware performed better than Proxmox In overall performance. VMware performed twice better than Proxmox in some cases. While Proxmox had better results than VMware in some instances.

In general, high overheads were measured in Proxmox for all setups, while overhead values observed in VMware were lower. An obvious difference between VMware and Proxmox behaviour was observed during the six core setup tests. Overall, the six core setup had the best performance among all setups using VMware, while it has the worst performance in Proxmox. Two scenarios were used for the comparison of different test cases. The first scenario is a comparison of the 16 and 12 core setups, which reflects the impact of a different number of

allocated CPU cores in VMs. Second is the comparison of the 12 and six-core setups, which shows the effect of using multiple VMs with an even number of allocated CPU resources.

Analysis of the test results has shown that a decreasing number of allocated CPU cores causes a higher response time in both hypervisors. Although CPU utilization in Proxmox was not noticeably affected, considerably less CPU utilization was observed in VMware. Using multiple VMs in VMware decreased the response time and CPU overhead. However, running the same tests in Proxmox caused increased CPU utilization. Additionally, response time was shorter for high traffic loads, and it was increased for low traffic loads.

Iperf was implemented to measure the performance of the computer network. The maximum bandwidth clients were connected on the network, and maximum throughput was measured in TCP data transfer. Jitter and datagram loss on the network along with maximum bandwidth and throughput was measured in UDP data transfer mode. The data for the two protocols were also compared.

A small private network was built with virtual clients and a single virtualized server. It was seen that the performance was very close to the level of the physical environment. On VMware experimental setup, some deviation was noticed. Abnormal datagram loss occurred on VMware, which signified that data transfer was not reliable on the network in UDP. Jitter, which is the amount of variation in latency/response time on the network, was found to be higher on VMware than in Proxmox.

Transmitting data over network connections requires special connections in a virtualized environment since the communications to the network utilize physical NIC of the host system. The overhead caused more data transmissions can be seen obviously in the results of the network performance tests. Network performances in virtual machines running under Proxmox Server are about 5 to 56% higher than a similar design in a physical setup. With VMware ESXi, the performance is around 36 to 71 % higher contrasted with the physical environment.

5.2 FUTURE WORK

It would be very compelling to continue this research for other hypervisors such as Citrix and Hyper-V. The results of the additional tests would show how different CPU resource scheduling and network configurations are implemented in the hypervisors. Particular implementation can have a direct effect on the performance of telecommunication services.

Since the performance of the services is considerably improved by using multiple VMs in VMware, it would be exciting to increase the number of VMs on each server. By repeating the tests, it would be possible to investigate potential virtualization benefits regarding CPU utilization. Another exciting experiment could be focused on identification of the maximum number of simultaneously running VMs that would allow keeping the system stable from CPU and network resources point of view. It would be nice to see how other various tests performed using different hardware and operating systems.

5.3 CONCLUSION

This section concludes by answering the research question. Each chapter is also summarized.

- What has been done in the literature to measure the performance of a virtual environment effectively?

To facilitate this, a comprehensive literature review was done in chapter 2, and the following parameters were identified to measure virtualization performance which was namely CPU, Disk, Memory, and Network (Reddy and Rajamani, 2014, Perera and Keppitiyagama, 2011)

- How to set a virtualized environment in order to test different performance configurations?

In this study, datacentre purposed servers together with Type 1 (bare metal hypervisors), VMware ESXi 5.5, and Proxmox 5.3 were used to evaluate virtualization performance. The experimental environment was conducted on server Cisco UCS B200 M4 which was the host machine and the virtual environment that is encapsulated within the physical layer which hosts the guest virtual machines consists of virtual hardware, Guest OSs, and third-party applications. The host server consists of virtual machines with one operating system, CentOS 7 64 bit. For performance evaluation purposes, each guest operating system was configured and allocated the same amount of virtual system resources. This study developed a virtual environment and experimental design to conduct tests on different configuration (Ali, 2015, Martinez et al., 2009)

- How to measure the performance of different configurations in a virtualized environment?

This study was able to use Different Workload/benchmarking tools for Network, CPU, Memory and Disk performance, namely; Iperf, Unibench, Ramspeed, and IOzone, respectively to simulate the workload as well as measure the various configurations on the virtual environment(Hwang et al., 2013, Somani and Chaudhary, 2009a, Babu et al., 2014, Walters et al., 2008b).

It is easier to figure out the performance of virtualized environments on private clouds. Differentiating tests between real hardware and software is challenging because measuring hardware performance means measuring virtualization overheads. Virtualization overhead depends on the software implementation of the hypervisor, meaning measuring software performance as well. The primary objective of this project was to evaluate virtualization performance as well as to determine which virtual machine configuration provides effective and optimal performance.

It can be seen that is vital to have dynamic hardware configuration, and the configuration described in Chapter 3, section 3.3.1 proves to provide virtual machines with optimal resources for performance in a private cloud and as well as capacity growth and this can be proven from the various performance tests performed. The resources allocation to the virtual machines proves, and the full performance capabilities of a virtual machine are only as good as the hardware it sits on with adequate resources. VMware ESXi provided optimal performance throughout the tests, which can be recommended when providing a private cloud solution.

Chapter 1 introduced us to Virtualization and the cloud. The research problem, the purpose of the study, the research question, objectives, and the structure of the thesis are described in this chapter. Chapter 2 discussed various literature in cloud computing and virtualization computing and were thoroughly covered and explained. It was understood that the basis if the cloud is the use of virtualization technology in information technology infrastructure. After considering factors such as compatibility, cost, and features and performance from the findings on the literature review, VMware EXSI 5.5 and Proxmox were selected for study and experimentation. The first objective has been to review the previous work which has been done on that subject. To be able to review on the previous work, it has been outlined at first the primary concept associate to virtualization and cloud computing in the technical review. After the technical review, the literature review has permitted to critic the previous work on virtualization technology and cloud computing. The different articles review was showing that

much progress has been made to try to uniform cloud computing and improve virtualization performances.

The second objective was to design and build and develop a virtualized environment to be able to test the performances of different performances. This setup can be seen in the chapter. The methodology used in the study for Chapter 3 was discussed. The mixed research methodology was adopted has both quantitative and qualitative research methodologies so to obtain accurate and relevant results from the study. Qualitative data from the literature was obtained using qualitative research methodology and made qualitative comparisons. Mixed research methodology allowed us to study and analyze the performance for VMware ESXi 5.5 and Proxmox on CentOS 7. Chapter 4 presented detailed experimental results and analysis on the performance of the different configuration on the operating system running on the different hypervisors (VMware and Proxmox).

Chapter 4 also met the third objective, which was to evaluate the performances of different configurations. The different instances have been tested for CPU performances, memory, network performances, as well as hard drive performances. Also, it has been evaluating the difference of performances between VMware and Proxmox.

Satisfactory results were obtained in this study by answering the research question. There were Performance differences in different virtualization systems through different configurations.

For memory and disk, the test outcomes demonstrated measurements of overhead is little. For Processor and network, was more perplexing and hence the overhead is more significant. At the point of the overall performance of a virtual machine running in VMware ESXi Server is contrasted with a conventional system, virtualization increased by 33% in terms of performance.

It is not easy to provide a real system configuration. In such cases, workload/benchmarks could provide close to real application systems for better results. The tests demonstrate that virtualization relies on the host system and the hypervisor. Given the tests, both VMware ESXi and Proxmox servers can provide Optimal performance.

There are various sections that overall performance depends on. One section that influences the performance is the drivers utilized by the OS. On the off chance that a virtual device contrasts in two virtualization products, additionally the driver utilized by the virtual machine

OS is unique. This distinction, at that point, influences the performance of the virtual machine directly. An important region where the impact of lower performance ought to be inspected is the production environment. After virtualization overhead is expelled from the overall system resources, the number of virtual machines in a single host system can be chosen. These machines will then keep running on the rest of the resources. The more there are several virtual machines, the more valuable virtualization can be. In zones, for example, testing environments where the requirements need high-performance server virtualization is the most optimal option. These results are consistent with what has been done by other authors (Walters et al., 2008a, Varrette et al., 2013, Perera and Keppitiyagama, 2011).

The prerequisite of this conclusion is that all 16 logical processors are occupied by vCPUs. For tightly coupled CPU-intensive workloads, the total number of VMs, vCPUs per VM, and memory allocated per VM become critical for performance. We obtained the best performance when the ratio of the total number of vCPUs to processors is 2. The experimental measurements have shown that the six core setup provides the best performance for high traffic. Doubling the memory size on each VM, for example from 1024MB to 2048MB, gave us at most 15% improvement of performance when the ratio of total vCPUs to logical processors is 2. As regards the high traffic scenario, the six core setup has the lowest CPU utilization, even lower than the non-virtualized system.

Moreover, it is capable of handling more traffic. Therefore, it seems better to have several instances with fewer CPU cores for high traffic loads. The total virtual memory of all VMs allocated to a physical machine has to be less than the physical memory of the machine to avoid poor performance due to swapping. From the Network results, there is less bandwidth available allocated for the clients resulting in fewer throughputs when the numbers of active Clients increase. When there is more than one client active on the network and sending data, the throughput of UDP is slightly higher than TCP throughput. It can be predicted that the higher bandwidth led to higher throughput in UDP data transfer. From the results, we can see that UDP is efficient when the clients spend fewer amounts of data, but when data is sent in bulk, then TCP can be better than UDP. VMware results comparing them to with Proxmox. The optimal requests sent by the client before server saturation is when there are three clients. The consolidated write performance for Iozone test has shown the comparison of Proxmox and VMware with Bare Metal. VMware has shown better performance in almost all the file size as compared with Proxmox. Proxmox has shown 79% performance, and VMware was at 60%

when compared to Bare Metal with 64 MB file size. The performance of VMware in case of 1 GB file size was 86%, which was more than twice fast while compared with Proxmox. In the case of 128 MB file size, VMware outperformed the Proxmox in all file sizes, showing it has the best performance. VMware performance was identical to the Bare Metal system in consolidated read performance of for file size 64 MB and 128 MB. While the rest of the cases, VMware performance was better than Proxmox has shown 66% performance while compared with Bare Metal in 1 MB file size.

5.4 RECOMMENDATIONS

Cloud computing brings benefits for service providers and users because of its characteristics: such as pay for use, on-demand, and scalable computing. Managing Virtualization configurations is a critical task to accomplish effective sharing of physical resources and scalability (Ha et al., 2016). The scale of the workloads submitted to a cloud environment is much larger than the benchmarks in our experiments; the difference of performance, resource consumption, or cost on different virtual configurations is essential. Therefore, it is crucial to know the impact of different virtual configurations in a cloud environment for users, service providers, and private cloud owners. Our findings help us decide appropriate methods to deploy services hosted both on public and private clouds as well as the VM configuration in terms of dedicated resources (Jiang et al., 2014). Virtualization is an essential factor in cloud computing because it provides a way to analyze, verify, and configure computing resources from clouds and dynamically to assign or to reassign virtual resources. This research will help private cloud administrators, owners, and users decide how to configure virtual resources for given workloads to optimize performance.

REFERENCES

- A VOUK, M. 2008. Cloud computing—issues, research and implementations. CIT. Journal of Computing and Information Technology, 16, 235-246.
- AHMED, M., ZAHDA, S. & ABBAS, M. Server consolidation using OpenVZ: Performance evaluation. 2008 11th International Conference on Computer and Information Technology, 2008. IEEE, 341-346.
- ALI, E. 2015. Optimizing Server Resource by Using Virtualization Technology. Procedia Computer Science, 59, 320-325.
- ALI, I. & MEGHANATHAN, N. 2011. Virtual Machines and Networks-Installation, Performance Study, Advantages and Virtualization Options. arXiv preprint arXiv:1105.0061.
- BABU, A., HAREESH, M., MARTIN, J. P., CHERIAN, S. & SASTRI, Y. System performance evaluation of para virtualization, container virtualization, and full virtualization using xen, openvz, and xenserver. Advances in Computing and Communications (ICACC), Fourth International Conference on, 2014. IEEE, 247-250.
- BENEVENUTO, F., FERNANDES, C., SANTOS, M., ALMEIDA, V., ALMEIDA, J., JANAKIRAMAN, G. J. & SANTOS, J. R. Performance models for virtualized applications. International Symposium on Parallel and Distributed Processing and Applications, 2006. Springer, 427-439.
- BHUKYA, D. P., RAMACHANDRAM, S. & SONY, A. R. Evaluating performance of sequential programs in virtual machine environments using design of experiment. Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on, 2010. IEEE, 1-4.
- BUYYA, R., GARG, S. K. & CALHEIROS, R. N. SLA-oriented resource provisioning for cloud computing: Challenges, architecture, and solutions. Cloud and Service Computing (CSC), 2011 International Conference on, 2011. IEEE, 1-10.

- CHEN, Q., LIANG, L., XIA, Y. & CHEN, H. Mitigating sync amplification for copy-on-write virtual disk. 14th {USENIX} Conference on File and Storage Technologies ({FAST} 16), 2016. 241-247.
- CHEN, W.-N. & ZHANG, J. A set-based discrete PSO for cloud workflow scheduling with user-defined QoS constraints. Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on, 2012. IEEE, 773-778.
- CHENG, L., WANG, C. & DI, S. Defeating Network Jitter for Virtual Machines. 2011 Fourth IEEE International Conference on Utility and Cloud Computing, 5-8 Dec. 2011 2011. 65-72.
- CHIUEH, S. N. T.-C. & BROOK, S. 2005. A survey on virtualization technologies. Rpe Report, 142.
- DALL, C., LI, S.-W., LIM, J. T., NIEH, J. & KOLOVENTZOS, G. ARM virtualization: performance and architectural implications. Proceedings of the 43rd International Symposium on Computer Architecture, 2016. IEEE Press, 304-316.
- DESHANE, T., SHEPHERD, Z., MATTHEWS, J., BEN-YEHUDA, M., SHAH, A. & RAO, B. 2008. Quantitative comparison of Xen and KVM. Xen Summit, Boston, MA, USA, 1-2.
- ELSAIED, A. & ABDELBAKI, N. Performance evaluation and comparison of the top market virtualization hypervisors. 2013 8th International Conference on Computer Engineering & Systems (ICCES), 26-28 Nov. 2013 2013. 45-50.
- EMEAKAROHA, V. C., BRANDIC, I., MAURER, M. & BRESKOVIC, I. SLA-aware application deployment and resource allocation in clouds. Computer Software and Applications Conference Workshops (COMPSACW), 2011 IEEE 35th Annual, 2011. IEEE, 298-303.
- ENBERG, P. 2016. A Performance Evaluation of Hypervisor, Unikernel, and Container Network I/O Virtualization.

- GONG, C., LIU, J., ZHANG, Q., CHEN, H. & GONG, Z. The characteristics of cloud computing. Parallel Processing Workshops (ICPPW), 2010 39th International Conference on, 2010. IEEE, 275-279.
- GSCHWANDTNER, P., FAHRINGER, T. & PRODAN, R. Performance analysis and benchmarking of the intel scc. 2011 IEEE International Conference on Cluster Computing, 2011. IEEE, 139-149.
- GULATI, A., MERCHANT, A. & VARMAN, P. J. mClock: Handling Throughput Variability for Hypervisor IO Scheduling. OSDI, 2010. 1-7.
- GUO, J., LIU, F., LUI, J. C. & JIN, H. 2016. Fair network bandwidth allocation in IaaS datacenters via a cooperative game approach. IEEE/ACM Transactions on Networking, 24, 873-886.
- GUPTA, D., CHERKASOVA, L., GARDNER, R. & VAHDAT, A. Enforcing performance isolation across virtual machines in Xen. Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware, 2006. Springer-Verlag New York, Inc., 342-362.
- HA, S.-H., VENZANO, D., BROWN, P. & MICHIARDI, P. On the impact of virtualization on the I/O performance of analytic workloads. Cloud Computing Technologies and Applications (CloudTech), 2016 2nd International Conference on, 2016. IEEE, 31-38.
- HAUSWIRTH, M., DIWAN, A., SWEENEY, P. F. & MOZER, M. C. Automating vertical profiling. ACM SIGPLAN Notices, 2005. ACM, 281-296.
- HUBER, N., VON QUAST, M., HAUCK, M. & KOUNEV, S. Evaluating and Modeling Virtualization Performance Overhead for Cloud Environments. CLOSER, 2011. 563-573.
- HWANG, J., ZENG, S., WU, F. Y. & WOOD, T. A component-based performance comparison of four hypervisors. 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), 27-31 May 2013 2013. 269-276.

- JIANG, J. M., ZHU, H., LI, Q., ZHANG, S., GONG, P. & HONG, Z. Configuration of Services Based on Virtualization. 2014 Theoretical Aspects of Software Engineering Conference, 1-3 Sept. 2014. 177-184.
- JOY, A. M. Performance comparison between Linux containers and virtual machines. 2015 International Conference on Advances in Computer Engineering and Applications, 19-20 March 2015. 342-346.
- KIM, I., KIM, T. & EOM, Y. I. NHVM: Design and Implementation of Linux Server Virtual Machine Using Hybrid Virtualization Technology. 2010 International Conference on Computational Science and Its Applications, 23-26 March 2010. 171-175.
- KOVARI, A. & DUKAN, P. KVM & OpenVZ virtualization based IaaS open source cloud virtualization platforms: OpenNode, Proxmox VE. 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics, 20-22 Sept. 2012. 335-339.
- KUMAR, D. & SINGH, A. S. A survey on resource allocation techniques in cloud computing. International Conference on Computing, Communication & Automation, 15-16 May 2015. 655-660.
- KUMAR, S. & M, V. 2015. Cloud Computing: Performance Study in an Eucalyptus Private Cloud. International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com Certified Journal, 5, 9.
- LEE, B. C. & BROOKS, D. M. Accurate and efficient regression modeling for microarchitectural performance and power prediction. ACM SIGOPS Operating Systems Review, 2006. ACM, 185-194.
- LI, J., WANG, Q., JAYASINGHE, D., MALKOWSKI, S., XIONG, P., PU, C., KANEMASA, Y. & KAWABA, M. Profit-based experimental analysis of IaaS cloud performance: impact of software resource allocation. Services Computing (SCC), 2012 IEEE Ninth International Conference on, 2012. IEEE, 344-351.
- LI, J., WANG, Q., JAYASINGHE, D., PARK, J., ZHU, T. & PU, C. Performance Overhead among Three Hypervisors: An Experimental Study Using Hadoop Benchmarks. 2013 IEEE International Congress on Big Data, June 27 2013-July 2 2013. 9-16.

- LI, Y., LI, W. & JIANG, C. A survey of virtual machine system: Current technology and future trends. *Electronic Commerce and Security (ISECS)*, 2010 Third International Symposium on, 2010. IEEE, 332-336.
- LI, Z., KIHLE, M., LU, Q. & ANDERSSON, J. A. Performance Overhead Comparison between Hypervisor and Container Based Virtualization. *Advanced Information Networking and Applications (AINA)*, 2017 IEEE 31st International Conference on, 2017. IEEE, 955-962.
- LUO, H., EGBERT, A. & STAHLHUT, T. 2012 IEEE 3rd International Conference on Software Engineering and Service Science (ICSESS).
- MAHJOUB, M., MDHAFFAR, A., HALIMA, R. B. & JMAIEL, M. A comparative study of the current cloud computing technologies and offers. *Network Cloud Computing and Applications (NCCA)*, 2011 First International Symposium on, 2011. IEEE, 131-134.
- MARINESCU, D. & KRÖGER, R. 2007. State of the art in autonomic computing and virtualization. *Distributed Systems Lab, Wiesbaden University of Applied Sciences*, 1-24.
- MARTINEZ, J. C., WANG, L., ZHAO, M. & SADJADI, S. M. Experimental study of large-scale computing on virtualized resources. *Proceedings of the 3rd international workshop on Virtualization technologies in distributed computing*, 2009. ACM, 35-42.
- MATTHEWS, J. N., HU, W., HAPUARACHCHI, M., DESHANE, T., DIMATOS, D., HAMILTON, G., MCCABE, M. & OWENS, J. Quantifying the performance isolation properties of virtualization systems. *Proceedings of the 2007 workshop on Experimental computer science*, 2007. ACM, 6.
- MELL, P. & GRANCE, T. 2009. The NIST definition of cloud computing. *National institute of standards and technology*, 53, 50.
- MOHAN, N. R. R. & RAJ, E. B. Resource Allocation Techniques in Cloud Computing -- Research Challenges for Applications. *2012 Fourth International Conference on Computational Intelligence and Communication Networks*, 3-5 Nov. 2012 2012. 556-560.

- MORABITO, R. 2017. Virtualization on internet of things edge devices with container technologies: a performance evaluation. *IEEE Access*, 5, 8835-8850.
- MORABITO, R., KJÄLLMAN, J. & KOMU, M. Hypervisors vs. Lightweight Virtualization: A Performance Comparison. 2015 IEEE International Conference on Cloud Engineering, 9-13 March 2015 2015. 386-393.
- OVERBY, M. 2014. A Survey of Virtualization Performance in Cloud Computing. University of Minnesota, Duluth, MN, USA.
- PADALA, P., ZHU, X., WANG, Z., SINGHAL, S. & SHIN, K. G. 2007. Performance evaluation of virtualization technologies for server consolidation. HP Labs Tec. Report.
- PERERA, P. M. & KEPPITIYAGAMA, C. A performance comparison of hypervisors. *Advances in ICT for Emerging Regions (ICTer)*, 2011 International Conference on, 2011. IEEE, 120-120.
- PRUEKSAAROON, S., VARAVITHYA, V. & VANNARAT, S. An implementation of virtualization cluster: Extending Beowulf cluster using virtualization cluster management and image storage. 2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 6-9 May 2009 2009. 700-703.
- RAMALHO, F. & NETO, A. Virtualization at the network edge: A performance comparison. 2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 21-24 June 2016 2016. 1-6.
- RAO, V. V. & RAO, M. V. 2015. A survey on performance metrics in server virtualization with cloud environment. *Journal of Cloud Computing*, 2015.
- REDDY, P. V. V. & RAJAMANI, L. 2014. Evaluation of different hypervisors performance in the private cloud with SIGAR framework. *International Journal of Advanced Computer Science and Applications*, 5.
- RIMAL, B. P., CHOI, E. & LUMB, I. A taxonomy and survey of cloud computing systems. *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*, 2009. Ieee, 44-51.

- SLIGH, D. & OWUSU, T. D. 2014. CONSIDERATIONS FOR EMPLOYING SERVER VIRTUALIZATION TECHNOLOGIES. *Issues in Information Systems*, 15.
- SMITH, J. E. & NAIR, R. 2005. The architecture of virtual machines. *Computer*, 38, 32-38.
- SOMANI, G. & CHAUDHARY, S. Application performance isolation in virtualization. *Cloud Computing. CLOUD'09. IEEE International Conference on*, 2009a. IEEE, 41-48.
- SOMANI, G. & CHAUDHARY, S. Application Performance Isolation in Virtualization. 2009 IEEE International Conference on Cloud Computing, 21-25 Sept. 2009 2009b. 41-48.
- SRITRUSTA, S., NOBUO, F., TORU, N. & PRAMADIHANTO, D. A comparative study of open source softwares for virtualization with streaming server applications. 2009 IEEE 13th International Symposium on Consumer Electronics, 25-28 May 2009 2009. 577-581.
- STOICUTA, F., IVANCIU, I., MINZAT, E., RUS, A. B. & DOBROTA, V. An OpenNetInf-based cloud computing solution for cross-layer QoS: Monitoring part using iOS terminals. *Electronics and Telecommunications (ISETC)*, 2012 10th International Symposium on, 2012. IEEE, 167-170.
- STROBL, M., KUCERA, M., FOELDI, A., WAAS, T., BALBIERER, N. & HILBERT, C. Towards automotive virtualization. 2013 International Conference on Applied Electronics, 10-12 Sept. 2013 2013. 1-6.
- TSUGAWA, M., MATSUNAGA, A. & FORTES, J. User-level virtual network support for sky computing. 2009 Fifth IEEE International Conference on e-Science, 2009. IEEE, 72-79.
- VARRETTE, S., GUZEK, M., PLUGARU, V., BESSERON, X. & BOUVRY, P. HPC Performance and Energy-Efficiency of Xen, KVM and VMware Hypervisors. 2013 25th International Symposium on Computer Architecture and High Performance Computing, 23-26 Oct. 2013 2013. 89-96.
- VOORSLUYS, W., BROBERG, J., VENUGOPAL, S. & BUYYA, R. 2009. Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation. *CloudCom*, 9, 254-265.

- WALTERS, J. P., CHAUDHARY, V., CHA, M., GUERCIO JR, S. & GALLO, S. A comparison of virtualization technologies for HPC. Advanced Information Networking and Applications. AINA. 22nd International Conference on, 2008a. IEEE, 861-868.
- WALTERS, J. P., CHAUDHARY, V., CHA, M., S, G., JR. & GALLO, S. A Comparison of Virtualization Technologies for HPC. 22nd International Conference on Advanced Information Networking and Applications (aina 2008), 25-28 March 2008 2008b. 861-868.
- WANG, L., VON LASZEWSKI, G., YOUNGE, A., HE, X., KUNZE, M., TAO, J. & FU, C. 2010. Cloud computing: a perspective study. New Generation Computing, 28, 137-146.
- WANG, P., GAO, R. X. & FAN, Z. 2015. Cloud computing for cloud manufacturing: benefits and limitations. Journal of Manufacturing Science and Engineering, 137, 040901.
- WANG, Q. & VARELA, C. A. Impact of Cloud Computing Virtualization Strategies on Workloads' Performance. 2011 Fourth IEEE International Conference on Utility and Cloud Computing, 5-8 Dec. 2011 2011. 130-137.
- WU, H., REN, S., GARZOGLIO, G., TIMM, S., BERNABEU, G., CHADWICK, K. & NOH, S. 2016. A Reference Model for Virtual Machine Launching Overhead. IEEE Transactions on Cloud Computing, 4, 250-264.
- XAVIER, M. G., DE OLIVEIRA, I. C., ROSSI, F. D., DOS PASSOS, R. D., MATTEUSSI, K. J. & DE ROSE, C. A. A performance isolation analysis of disk-intensive workloads on container-based clouds. 2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, 2015. IEEE, 253-260.
- YAQUB, N. 2012. Comparison of Virtualization Performance: VMWare and KVM.
- YOUNGE, A. J., HENSCHER, R., BROWN, J. T., LASZEWSKI, G. V., QIU, J. & FOX, G. C. Analysis of Virtualization Technologies for High Performance Computing Environments. 2011 IEEE 4th International Conference on Cloud Computing, 4-9 July 2011 2011. 9-16.
- ZHANG, Q., CHENG, L. & BOUTABA, R. 2010. Cloud computing: state-of-the-art and research challenges. Journal of internet services and applications, 1, 7-18.

