Master of Science in Computer Science
February 2017

Performance Evaluation of Cassandra in a Virtualized environment

Mohit Vellanki

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:
Author:
Mohit Vellanki
E-mail: move15@student.bth.se

University advisor:
Sogand Shirinbab
Department of Computer Science and Engineering (DIDD)

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

ABSTRACT

Context. Apache Cassandra is an open-source, scalable, NoSQL database that distributes the data over many commodity servers. It provides no single point of failure by copying and storing the data in different locations. Cassandra uses a ring design rather than the traditional master-slave design.
Virtualization is the technique using which physical resources of a machine are divided and utilized by various virtual machines. It is the fundamental technology, which allows cloud computing to provide resource sharing among the users.

Objectives. Through this research, the effects of virtualization on Cassandra are observed by comparing the virtual machine arrangement to physical machine arrangement along with the overhead caused by virtualization.

Methods. An experiment is conducted in this study to identify the aforementioned effects of virtualization on Cassandra compared to the physical machines. Cassandra runs on physical machines with Ubuntu 14.04 LTS arranged in a multi node cluster. Results are obtained by executing the mixed, read only and write only operations in the Cassandra stress tool on the data populated in this cluster. This procedure is repeated for 100% and 66% workload. The same procedure is repeated in virtual machines cluster and the results are compared.

Results. Virtualization overhead has been identified in terms of CPU utilization and the effects of virtualization on Cassandra are found out in terms of Disk utilization, throughput and latency.

Conclusions. The overhead caused due to virtualization is observed and the effect of this overhead on the performance of Cassandra has been identified. The consequence of the virtualization overhead has been related to the change in performance of Cassandra.

Keywords: Cassandra, Virtualization, performance evaluation, KVM

ACKNOWLEDGMENTS

I would like to acknowledge my supervisor, **Sogand Shirinbab** for her support and valuable feedback in completing this thesis work. I'd like to express my profound gratitude to my **friends** and **family** for giving me the continuous support and encouragement and for being there at the time of need.

I am also thankful to **Emiliano Casalicchio**, **Christine Niyizamwiyitira**, **Kurt Tutschku** for their inputs. I would like to specially thank **Bengt Olsson** and **Björn Mattsson** from the IT helpdesk for their patience and timely responses in helping me solve the practical issues I faced during the experiments.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

This research intends to compare the performance of Cassandra, a NoSQL database in physical (or non-virtualized) and virtualized environments. The method used in this thesis to collect the results is an experiment, in which we measure the metrics using the tools available. The metrics used to compare the performances are CPU, disk utilization and throughput. These metrics are measured and compared in two similar experiment setups, one being non-virtualized and another in a virtualized environment. Based on the results collected, we observe the overhead caused due to virtualization and its effect on Cassandra performance is measured.

1.1 Research Gap

Cassandra is a NoSQL database management system used for managing large amounts of structured data distributed on many servers, which provides high availability with no single point of failure [1]. NoSQL, which is interpreted as Not Only SQL, emerged as an alternative to the Relational database systems. It supports distributed data management and scalability of applications. NoSQL database systems leave behind the ACID (Atomicity, Consistency, Isolation, Durability) transactions primarily found in relational databases [2]. Virtualization is a technique which allows the efficient use of existing hardware and helps with costs and work reduction with improved performance [3]. Hypervisors or virtual machine monitors (VMM) are the extra layer of software that implement virtualization. Research has been done on multiple occasions to evaluate the performance of Cassandra and also compare the various hypervisors in terms of performance [4], [5], [6] [7], [8], [9]. This leads to the gap where less or no work has been done before, the evaluation of Cassandra in virtualization. Some questions unanswered because of lack of research are how well does Cassandra perform when used under Virtualization? How much overhead does a hypervisor cause when used to virtualize? Answers to such questions can be found out from this research where we compare the performance of Cassandra in a physical server and virtual server arrangement.

1.2 Aim and Objectives

1.2.1 Aim

The main aim of this thesis is to compare the performance of Cassandra in terms of CPU Utilization and disk utilization along with the throughput and latency in a physical server arrangement and virtual server arrangement.

1.2.2 Objectives

- To measure the CPU and disk utilization in physical machines and virtual machines arrangements when operations are performed on a Cassandra database populated using the stress tool.
- Measure the throughput and latency of Cassandra in physical and virtual machine arrangements.
- Compare the CPU utilization, throughput, latency and disk utilization of Cassandra in physical machine and virtual machine arrangements.

1.3 Research Questions

In this research, the gap mentioned above is filled, by answering the following research questions. Two research questions have been framed to fill the gap and they are mentioned below along with the motivation for choosing them.

Research Question 1: What is the overhead caused by virtualization in terms of CPU utilization?

Motivation: Virtualization is implemented by running a layer of software called the hypervisor or the virtual machine monitor (VMM). This usually causes overhead as it requires some resources to run the hypervisor. The overhead is measured as the amount of CPU utilized to virtualize under this research question.

It can be found out by measuring the difference between CPU utilization when Cassandra is run on physical server arrangement and a virtual server arrangement.

Research Question 2: How does the virtualization affect the performance of Cassandra in terms of disk utilization, throughput and latency?

Motivation: The overhead caused by the virtualization can have effects on the performance of Cassandra. These effects can be measured in terms of disk utilization, throughput and latency.

Measuring the effect of overhead caused by virtualization on Cassandra can explain how well it works when virtualized.

1.4 Method

The research method used in finding solutions for the research questions mentioned above is an experiment. Four Ubuntu machines with same specifications are provided by the Blekinge Institute of Technology for running this experiment. Cassandra was installed in these four machines and the experiments were performed in these machines. Later virtual machines with Cassandra were deployed on these physical machines and the experiments were repeated. The results from both set of experiments were compared. The independent variable in this experiment is the thread count whereas the dependent variables are CPU and Disk utilization, latency and throughput.

1.5 Contribution

The major contribution of this research is to find out the effect of virtualization on Cassandra when compared to the Cassandra run on physical machines. It can be inferred that the overhead caused by virtualization is affecting the performance of the Cassandra in the virtual machines arrangement and causing a decrease in the throughput.

1.6 Thesis Outline

The rest of the document is organized as below.

Chapter 2 explains the necessary background regarding the NoSQL database Cassandra used in this research along with some important concepts regarding the virtualization and the hypervisor used Kernel-based Virtual Machine (KVM).

Chapter 3 contains the synopsis of related work done in the related fields of Cassandra, Virtualization and hypervisors.

Chapter 4 discusses the research method used in this thesis. It explains about the design of the experiment, setup of the experiment and how the experiment is run with the criteria to be followed.

Chapter 5 is the results and analysis section. It explains the results gathered from the experiment and their analysis along with the Cohen's d effect size which finds if the difference between the results is small, medium, large.

Chapter 6 is the discussion related to the experiment. It presents the validity threats, and the answers to the research questions.

Chapter 7 contains the conclusions of the research and the possible options for future work.

2 BACKGROUND

In the following subsections, the various key concepts that are necessary to understand and follow the research are discussed. These concepts summarize about cloud computing, virtualization, hypervisors, KVM, NoSQL databases, and working and architecture of Cassandra.

2.1 Cloud Computing

Cloud computing is defined as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable resources (e.g. Networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" by the National Institute of Science and Technology [10]. The access of services, as per requirements and without any additional information on how the services are hosted or provided, is made easy for the users by the virtue of cloud computing. Several techniques like grid computing, P2P computing have earlier offered such utility computing and now more recently cloud computing, where the infrastructure is referred to as cloud, consists of services that can be accessed by the users and businesses. These services, which maybe infrastructure or a platform or a software can be accessed as a subscription or pay as you use model [11]. The technology using which the cloud computing enables the users to access services as they need or the underlying technology which enables resource sharing is the virtualization technique.

2.2 Virtualization

Virtualization, as an enabler technology to the cloud computing, allows the user to use a physical server and share it among more than one user by dividing it into virtual resources called virtual machines [12]. It is implemented by adding an extra layer of software called Virtual Machine Monitor (VMM) which is also known as a hypervisor. A VMM acts in between the operating system (OS) and the virtual machines. It helps in mediating the physical resources between the physical server, also known as host machine and virtual machines known as guest machines [13]. Virtual machine monitors are classified into two types, type 1 known as native or bare metal and type 2 known as hosted hypervisors.

Native hypervisors run directly on the host machine's hardware controlling the hardware and allocating it to the guest machines running on the host machines at a different level above hypervisor. Hosted hypervisors run on a layer above the host operating system and manage the guest operating systems running at another level [14].

Virtualization is implemented using three different techniques, namely full virtualization, para virtualization and hardware assisted virtualization [15]. When the virtual machines created are not aware that they are virtualized and work in the same way as an actual physical machine, it is known as full virtualization. This is possible by the translation of non-virtualized instructions using the binary translations and direct execution of user level instructions. Whereas in para virtualization the virtual machine is aware of the virtualization and interacts with the host operating system instead of directly with the hardware [16]. In the hardware assisted virtualization [4], the hardware is provided with additional features by the hardware vendors. Microprocessors are provided with features that allow them to facilitate special requests from the hypervisors.

2.3 KVM

KVM is a Linux based open-source solution for virtualization. It contains extensions that implement a full virtualization technique. KVM also effectively uses the hardware-assisted virtualization technique [17]. A virtual machine created using KVM is seen as a normal Linux process working with the rest of the processes. KVM creates virtual machines by opening a device node and having its own memory. KVM is based on the open source QEMU emulator [18].

2.4 Cassandra

Apache Cassandra is an open source Not only SQL (NoSQL) database management system. A NoSQL database system is a non-relational database that can handle large volumes of data, unlike a traditional relational database management system (RDBMS) [1]. Cassandra ensures high data reliability by providing highly available service with no single point of failure. Cassandra has been designed in such a way that it can run with austere resources or on cheap commodity hardware and handle large volumes of data without the loss of efficiency [19].

Cassandra is also being widely used because of its ease in handling huge amounts of data by providing continuous availability, linear scalability and operational simplicity across many servers [20].

2.4.1 Working of Cassandra explained

Apache Cassandra is built in a way to handle petabytes of data and thousands of simultaneous users/operations per second. Various aspects of the working are explained below from the documentation of Cassandra [20].

- Partitioned row store database

  Cassandra enables the authorized user to access the data in any node at any data center using the Cassandra Query Language (CQL). CQL has a similar syntax to that of Structured Query Language (SQL). CQL can be used in the CQL shell known as cqlsh. This shell can be used to perform various database operations like creating, editing or removing keyspaces and tables or inserting and querying from the tables etc.

- Data distribution

  Cassandra distributes the data across all the nodes that are a part of a ring or a database cluster. This is done automatically without any intervention from the user and the data is partitioned across all the nodes of a cluster.

- Replication

  Cassandra provides replication as a built in feature which creates redundant copies of data across all the nodes in a cluster. By doing this, Cassandra makes sure that in case one of the nodes goes down, the data in that node is available as a backup in another node of the same ring. Various replication arrangements can be made like: replicate on a single data center or across various centers etc.

- Scalability

  Adding nodes for changing the capacity to scale up linearly is a simple process in Cassandra. Capacity can be increased by simply adding new nodes online.

2.4.2 Architecture

As discussed earlier, Cassandra is designed to handle large volumes of data across multiple nodes and with no single point of failure. It implements a node to node distribution system which creates copies of data in a cluster [20].

- All the nodes exchange data over the cluster ring every second.

- To ensure data durability a commit log is written sequentially for each node of a cluster to capture all the write activities.
- Data is then indexed and written into a memTable which is an in memory structure.
- After the memory structure becomes full, data is written in the form of a SSTable data file to the disk.
- All the writes are partitioned and replicated across the cluster.
- Cassandra combines the SSTables occasionally discarding the disused data and tombstones which indicate that data was deleted. This process is called Compaction.

3 RELATED WORK

This chapter contains the details of research conducted till date and related to performance evaluation of Cassandra in virtualized and non-virtualized environments.

Various research works have been conducted in the past to improve or measure the performance of Cassandra. The metrics measured in the previous studies discussed below, have tried to improve were execution times, overall performance, rate of execution requests etc.

Martins et al. have discussed how different virtualization techniques: full and para virtualization generate overhead in NoSQL databases like MongoDB and Cassandra. They have used Yahoo! Cloud Serving Benchmark (YCSB) to measure the performance. The authors concluded that the virtualization technique caused a difference in the performance of a NoSQL database [21]. Authors of [22] have used YCSB to compare the performance of various NoSQL databases. They reported the performance based on execution time for various operations like INSERT, UPDATE and READ. Elsayed has compared various hypervisors performance by using a custom SQL instance as a workload which serves as a simulation of real life situations [23].

3.1 Cassandra performance

In [5], the authors have proposed a measure to improve the response time of Cassandra by query scheduling. Implementing query scheduling has reduced the average response time. Feng et al. have mentioned the feasibility of using CCIndex in Cassandra which also includes recovery mechanisms along with the pros and cons of CCIndex for different Distributed Ordered Tables (DOTs). Though the study shows that the Cassandra is optimized to be used over hashtables rather than ordered tables, experiments conducted prove that there has been an increase in efficiency of CCIndex [6].

In [7], the authors have implemented MapReduce to improve the overall efficiency of Cassandra. They have listed various use cases, so that the developers can decide based on the use cases listed in the study and their applications. Authors of [8] have designed a monitoring tool which can generate the statistics and graphs based on the performance of Cassandra. This tool can be used to observe Cassandra when some kind of performance tuning is done and determine if the tuning done is good or bad. In [9], Cassandra has been scaled up by adding more nodes and the performance has been measured using the YCSB – Yahoo! Cloud Servicing Benchmark. The results have shown that adding more nodes has improved the performance of Cassandra.

3.2 Virtualization

In [4], the authors have measured the performance and compared the three famous hypervisors: KVM, XenServer and VMware. These hypervisors have been running a large scale telecommunication application during which the CPU utilization, disk utilization and response time are measured. The hypervisors have been compared based on the downtime and total migration time during live migration. The authors in [24] have measured the CPU overhead in a Xen hypervisor for processing a I/O request by a virtual machine. Authors have used the SIGAR framework in [17] to compare four hypervisors virtualization overhead in a private cloud created using CloudStack.

4 METHODOLOGY – EXPERIMENT

According to the authors in [25], a method used in a research project, sometimes referred as the scientific method is the approach in which a selected problem is solved. It consists of various steps, including but not limited to formulating a problem or a question, data collection, testing the data with respect to the problem and analyzing the results. After the collection and analysis of results, conclusions can be formed according to the problem. As choosing a method is a significant step in a research project, a four-step process has been described where we initially identify and describe the goals of the project after which we choose a suitable method and achieve the goal of the project [25].

Authors in [26] have defined performance evaluation which is a sub discipline of experimentation as an activity which can be used to understand how a hardware or a software entity performs. In [25] an experiment is defined as the examination of variables under different experimental procedures. So the performance of Cassandra in physical machine and virtual machine arrangements is compared by performing an experiment which is the chosen method for this research.

There exist other research methods used in computer science to carry out research. Some of them are case study, surveys or interviews, literature review. A case study was not considered because in this research, we are dealing with the general performance of Cassandra in different environments and not a specific operation of Cassandra or a specific application. Interviews and surveys can be used to find out the opinion of respondents or users in this case. But this does not give a specific answer to our research questions.

There are many kinds of experiments out of which controlled experiment is selected for this research as it involves comparison of one experimental condition or method vs another experimental condition.

4.1 Experiment Setup

4.1.1 Physical Servers arrangement

The experimental setup in the first part consists of four physical server machines provided by the university as arranged in Figure 1. These four machines run on Ubuntu 14.04 LTS each with Intel Xeon Processor E5-2420 v2, a physical memory of 23GB and 12 cores, as a result of hyper threading making it 24 virtual cores [13].

These four machines are accessed through Secure Shell (SSH) connection from another server with an IP address 194.47.131.201 known as Diptprisrv01.

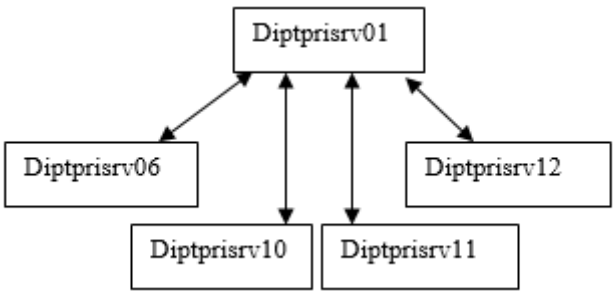

Figure 1 - Node Arrangement

The four machines with the respective IP addresses are

- Diptprisrv06 – 194.47.131.207 – Stress node
- Diptprisrv10 – 194.47.131.211 – Node 1
- Diptprisrv11 – 194.47.131.212 – Node 2

- Diptprisrv12 – 194.47.131.213 – Node 3

Cassandra is installed in all the four nodes using the tarball installation. After the installation is completed using the tarball file, the following changes are made to the configuration file of Cassandra (***install_location/conf/Cassandra.yaml***) on each node other than the stress node [19]:

- listen_address

  The IP address of the node itself is provided here

- rpc_address

  RPC or remote procedure call address is the IP address of the node itself. It is used by the seed node to initiate the Cassandra process and work distribution.

- seed_provider

  -seeds:

  One of the three nodes is selected as a seed node and the IP address of the seed node is provided in this field for all the three nodes. This seed node is the first node to be communicating with the stress node during the execution of stress command. This node distributes the work to the other nodes and replicates data to maintain consistency.

Adding the same seed address to all the three nodes forms, what is referred in Cassandra as, a multi node cluster. Multi node cluster setup aids in huge performance boosts and is the reason behind how Cassandra is highly scalable. The Cassandra-stress tool is used for benchmarking and load testing a Cassandra cluster. It is a Java based testing utility that helps in creating better data models or understanding how a database performs or scales up in a multi node cluster.

Before the actual experimentation begins, using the stress tool we find out the number of threads required to obtain the maximum performance. This is done by using the write command of stress tool by which we identify at which thread count the operations performed in a second is maximum. In the same procedure we also find out the thread count at which the performance was 66% maximum. These thread counts are used in the actual experiment to generate 66% and 100% load.

The Cassandra-stress command is used in the following format.

```
$ cassandra-stress command [options]
```

Three types of loads generated using the following commands:

- `$ cassandra-stress write`

  The write command in the stress tool generates load with only write requests and populates the database. It is required to write some data before attempting to perform a read or mixed operation

- `$ cassandra-stress read`

  The read command in the stress tool generates load by performing read operations on a database already populated with write requests.

- `$ cassandra-stress mixed`

  Mixed command of the stress tool utility can be used to perform a write and read operations together in a specified ratio.

4.1.2 Virtual Servers arrangement

To achieve the virtual server arrangement, a virtual machine is created in each one of the four nodes and these virtual machines are used to repeat the exact process again.

These virtual machines are created using Kernel-based Virtual Machine (KVM). The virtual machines are created using the virt-install tool, which is a command line based tool for creating new KVM. The virtual machines created are of exact same specifications as that of the physical machines. They were created with a physical memory of 23GB and 12 physical

processors and running Ubuntu 14.04 LTS. These four virtual machines were assigned a public IP address after creating a network bridge. This bridge acts as a communication medium between the network of a virtual machine and a physical machine. Bridging the network allows the virtual machine to be seen as a physical machine in the main network and hence a public IP address can be assigned to it.

Cassandra is installed on the virtual machines and the changes required to create a multi node cluster are made in the main configuration file. Once the multi node cluster is created, the experimentation process begins.

The metrics- CPU and disk utilization are measured using the Dstat tool in both the cases. It is a resource statistics tool. Throughput is given at the end of the stress tool with a bunch of other statistics.

4.2 Experiment Design

The experiment consists of two parts. In the first part we evaluate the performance of Cassandra in a physical server arrangement and in the second part we evaluate the performance in a virtual server arrangement. The metrics used for evaluation are CPU and disk utilization and throughput.

Steps involved in both the parts are similar. A super step consists of the following sub steps:

1. From the root node (Node 1), initially a write operation is performed using the Cassandra-stress tool to populate the database on which other operations are performed in the following steps
2. From the Node 1, the following operations are performed for twenty minutes on to the nodes 2,3,4 using the Cassandra-stress tool. It took around twenty minutes to perform an operation on the initially populated database and hence twenty minutes has been used as the duration for the further steps.
    a. Firstly, 100% mixed load is generated in the ratio of three reads for every write.
    b. Next, 100% read load is applied by the stress tool.
    c. Finally, a 100% write load is applied.
3. While the load is being generated in all the steps in step 2, the CPU and disk utilization are measured continuously averaging for every 30 seconds using the Dstat tool.
4. At the end of the load generation, the Cassandra-stress tool generates statistics which includes the throughput of the total operation
5. Same procedure is repeated with a 66% load because maximum load (100%) and two thirds of the maximum (~66%) are the frequently used workloads to test the databases in telecommunication systems.

After performing the above steps in a physical server arrangement, the same sub steps are replicated in the virtual server arrangement as well. Each super step has been performed ten times in both physical and virtual server arrangements. These iterations were performed to ensure that the results obtained were consistent. The metrics measured in the both the parts are compared and analyzed later to find out the overhead of virtualization in terms of CPU utilization and also the difference in throughput of Cassandra in physical and virtual server arrangement.

4.3 Metrics

The following metrics are measured during the experiment:

- CPU Utilization
    - It is measured using the Dstat tool at an interval of 30 seconds for twenty minutes i.e., forty values.
    - CPU Idle time is measured out of which the CPU utilization is calculated
    - Difference in CPU utilization in both the cases is the overhead of virtualization

- Disk Utilization
    - Disk utilization is measured using the Dstat tool at an interval of 30 seconds for twenty minutes i.e., forty values.
    - Total write operations(kB) made are displayed by the Dstat tool
    - Difference in disk operations in both the cases is caused due to the overhead of virtualization and it affects the performance of Cassandra
- Cassandra metrics
    - Total operation rate – Throughput

      The average number of operations made in a second are measured and displayed at the end of each stress tool run.
    - Latency mean

      Mean of the latency is measured in milliseconds and displayed after each stress tool run.

4.4 Tools used

4.4.1 Cassandra-stress

Cassandra stress tool is a testing tool which is based on Java. It is primarily used to evaluate or benchmark the performance of a database model. This tool has been chosen over the other tools because it is provided in the default Cassandra package and hence has been used. While using the stress tool, tables are created within a new keyspace. In this research, these are the tables and keyspaces on which we load the test and measure the metrics [19].

4.4.2 Nodetool utility

Nodetool utility is a command line based tool in Cassandra which is used to manage a single node as well as a multi node cluster. This utility provides with various commands which provide information ranging from various statistics about tables to name, details, status of a cluster to various partition and keyspace details [19]. There were no other known alternatives to perform the same tasks as Nodetool utility.

4.4.3 Virt-install

Virt-install is a command line based tool in Linux that can be used to create and install kernel-based virtual machines. This tool comes with various options which can be used to provide details about the virtual machine. These details range from the memory specifications, architecture types, graphic and network configurations.

4.4.4 Dstat

Dstat is an integrated and unique tool which can perform the functions of various commands combined together and display the information about various system resources in a detailed and organised manner in columns. Dstat has been chosen over other tools like Iostat and sar owing to its many robust features some of which are mentioned next. Dstat could measure both the CPU and Disk utilization together. Tools like Iostat or sar could only measure CPU or disk utilization but not both together. Dstat is also capable of writing the results directly to a .csv file [27].

4.5 Constraints

The following are some rules or constraints mentioned below that were followed while performing the experiment:

- Write operation before any other

The write operation of Cassandra-stress tool as to be used before performing a read or mixed operation because executing a write operation initially creates and populates tables and keyspaces which can be used for other operations later.

- Clearing data and logs after every iteration

  Tables and keyspaces created were deleted after every iteration to ensure that all the iterations were performed on new data tables and keyspaces every time.

- Same configurations for virtual machine

  The virtual machines were created with the same configuration as that of the physical machines to replicate the same exact conditions and perform the experiment. This was to make sure that the performance was not affected by any changes of configurations.

5 RESULTS AND ANALYSIS

In this chapter, we discuss the results obtained from the experiment and analyze them in the following sections. The results from the different iterations and super steps of both physical and virtual arrangements are analyzed and compared to come to draw conclusions regarding the research questions.

To find the maximum operations per second or the maximum performance, we used the Cassandra-stress tool with the write command. Write command of the stress tool is executed by only specifying to write a large number of requests (~50 million) and without specifying the duration. This operation writes the requests and increases the thread count slowly till the maximum operations per second is obtained. The thread count where maximum operations per second was obtained is at 450 threads. While the same experiment was running, the thread count at which the performance was 66% of the maximum load was noted to be 150. These values as shown in Table 1 were used in the further steps to measure the metrics at 100% and 66% load. The increase in thread count and the corresponding maximum load were having a non-linear relation.

Table 1 - Workloads and corresponding thread counts

Load amount	Thread Count
100% workload	450
66% workload	150

5.1 Overhead of Virtualization

Virtualization overhead is measured in this research in terms of CPU utilization. Cassandra database is run on physical server arrangement and virtual server arrangement. The difference observed in the CPU utilization in both the cases can be attributed to the hypervisor, in this case, KVM because all other experimental conditions remain exactly the same.

Initially before the operations mentioned below are performed, the tables and keyspaces are populated using the write command of the stress tool. These tables and keyspaces are used to execute the listed operations.

5.1.1 100% Workload

5.1.1.1 Mixed Operation

The mixed operation was performed using the Cassandra-stress tool and the mixed command. Read and write operations were performed in the ratio of 3:1 which executes a write operation and three read operations consecutively. The mixed load was chosen in 3:1 ratio because this research was conducted with telecommunication systems in mind and in these systems, a write request is usually followed by three read requests. It was observed that the mean CPU utilization for a 100% workload in the physical server arrangement – where Cassandra was installed on all the nodes and stress was distributed from the stress node to the nodes 2,3 and 4 was 63.4%. This utilization is the average of over 10 iterations performed independently and in succession.

In the virtual server arrangement, where Cassandra was similarly installed and stress tool was operated from the stress node to the multi node cluster consisting of nodes 2,3 and 4, the CPU utilization was 69.42%. The utilization mentioned was the mean of the values obtained continuously over 10 iterations. The average utilization in both the cases is shown in Figure 2.

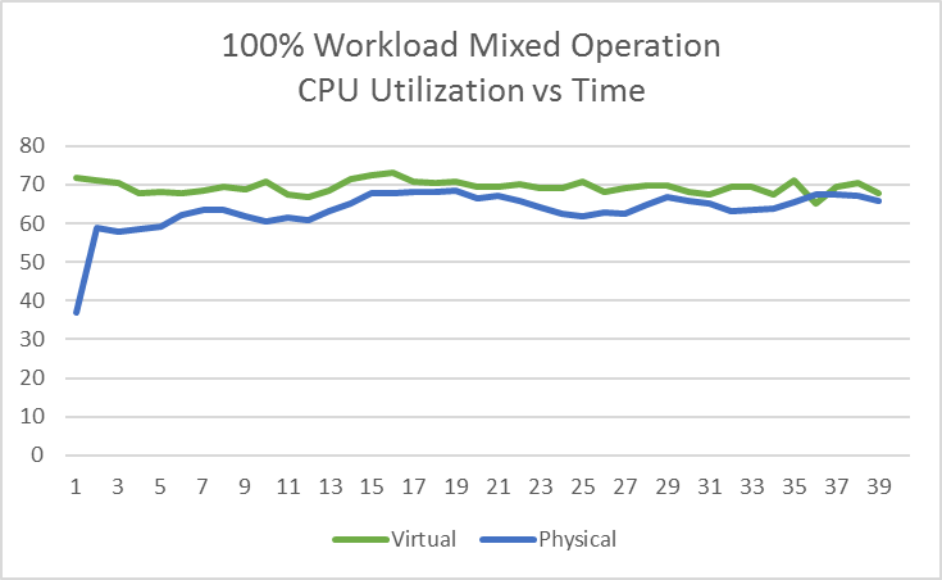

Figure 2 - 100% workload mixed operation (CPU (%) vs Time (30sec interval))

5.1.1.2 Read Operation

Using the Cassandra-stress tool, the read command is run for twenty minutes' duration at 450 thread count, i.e., maximum performance. While the command was executing, CPU utilization of the nodes as shown in Figure 3 was measured using the Dstat tool. In the physical machines arrangement, the average of the CPU utilization continuously for each sub step and for ten iterations is 57.48%, whereas the same metric measured similarly in a virtual machine configuration is 72.32%.

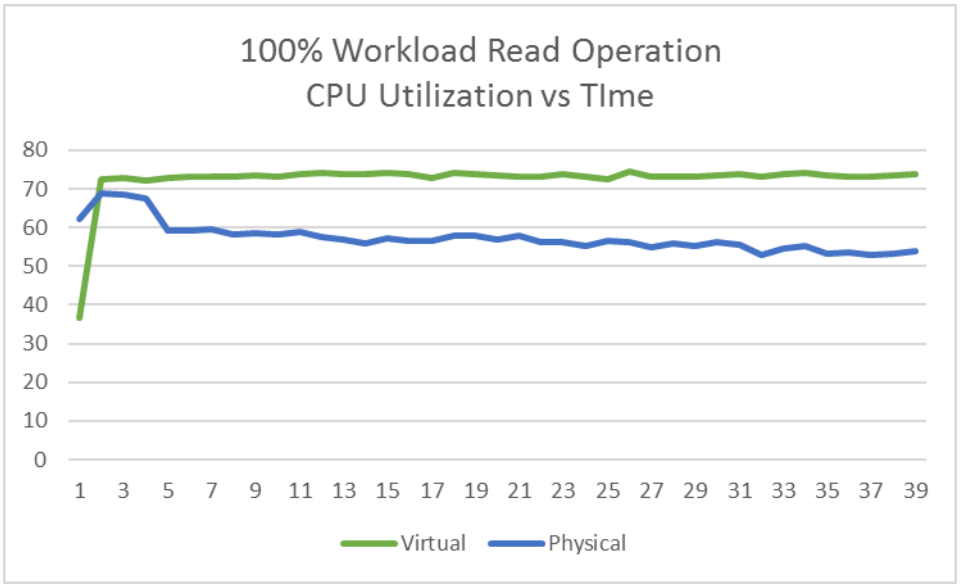

Figure 3 - 100% workload read operation (CPU (%) vs Time (30sec interval))

5.1.1.3 Write Operation

A write operation at the end of a sub step is performed for twenty minutes unlike the one at the beginning of every iteration which is performed to populate the database. The write command is executed with the thread count specified to 450 and the command is distributed to the multi node cluster consisting of nodes 2,3 and 4. The average value for the CPU utilization in the physical server arrangement is 48.82%, whereas the mean value of the virtual

server arrangement is 58.61%. The average utilization in both the cases are shown in Figure 4.

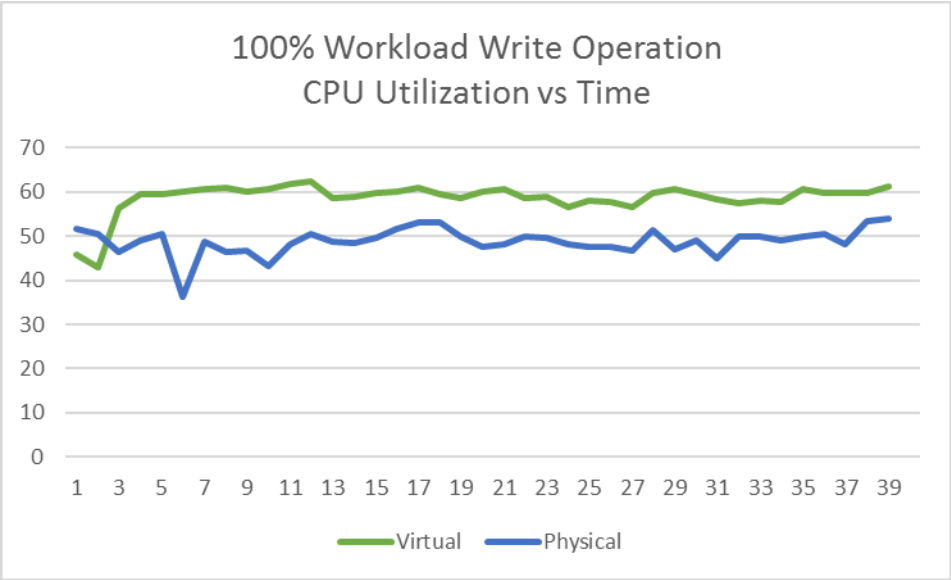

Figure 4 – 100% workload write operation (CPU (%) vs Time (30sec interval))

The average utilization for mixed, read and write operations are tabulated in Table 2 for the 100% workload along with the standard deviation values.

Table 2 - 100% workload operations results

Type of Operation	Physical Machines	Standard Deviation	Virtual Machines	Standard Deviation
Mixed	63.4%	5.13	69.42%	1.62
Read	57.48	3.69	72.32	5.81
Write	48.82%	3.01	58.61%	3.6

5.1.2 66% Workload

5.1.2.1 Mixed Operation

For the 66% workload, the thread count is set to 150 while executing the mixed operation. The experiment is repeated in a similar manner like the 100% workload and the metrics are measured during the sub step for the ten iterations using the Dstat tool. The mean utilization for both the cases is shown in Figure 5, the average CPU utilization for physical machine arrangement with a 66% load is 56.98%, whereas the average utilization over ten iterations in case of virtual machine arrangement is noted to be 62.05%.

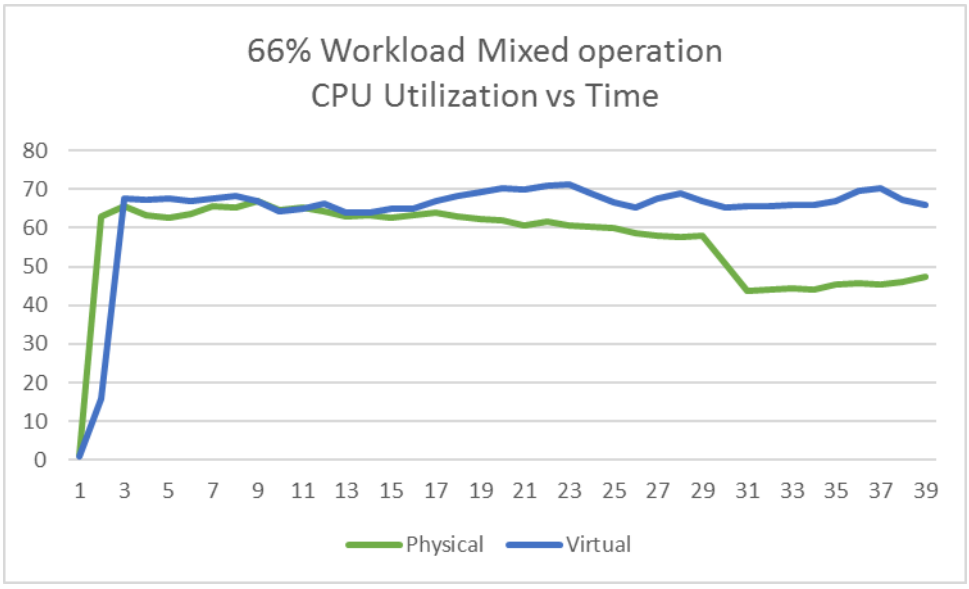

Figure 5 - 66% workload mixed operation (CPU (%) vs Time (30sec interval))

5.1.2.2 Read Operation

The mean values of CPU utilization are measured continuously during the sub step and averaged over ten iterations during the execution of a stress tools read operation for twenty minutes. The average utilization for physical machine arrangement is 42.26% and the mean value for the virtual machine arrangement is 67.36% as shown in Figure 6.

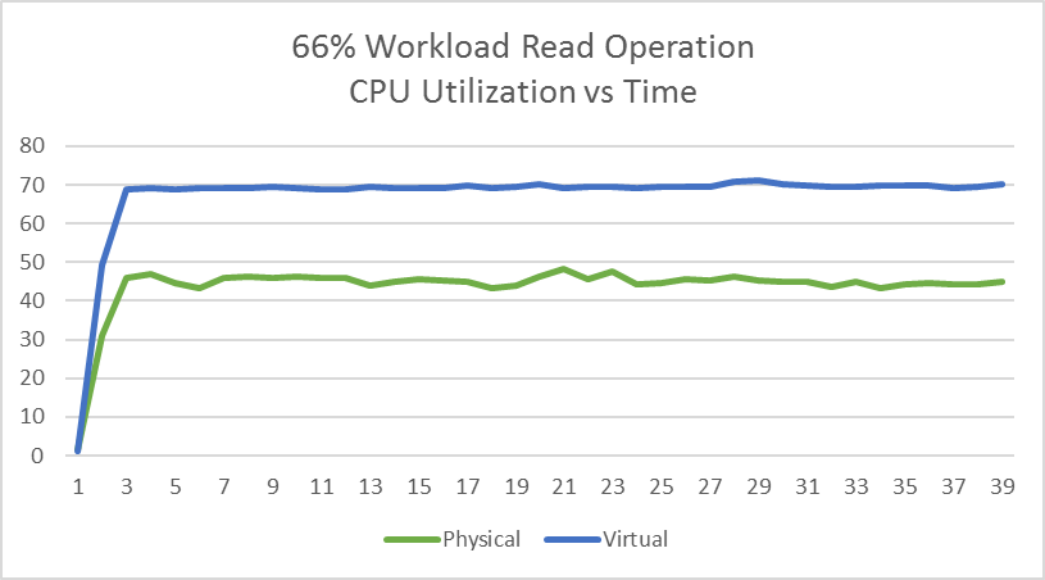

Figure 6 - 66% workload read operation (CPU (%) vs Time (30sec interval))

5.1.2.3 Write Operation

The write operation performed for twenty minutes at the thread count of 150 or 66% performance rate has given the mean utilization of CPU for the physical machine arrangement to be 41.74% and the average value of CPU utilization for the virtual machine arrangement is 58.64%. The average utilizations in both cases are as shown in Figure 7.

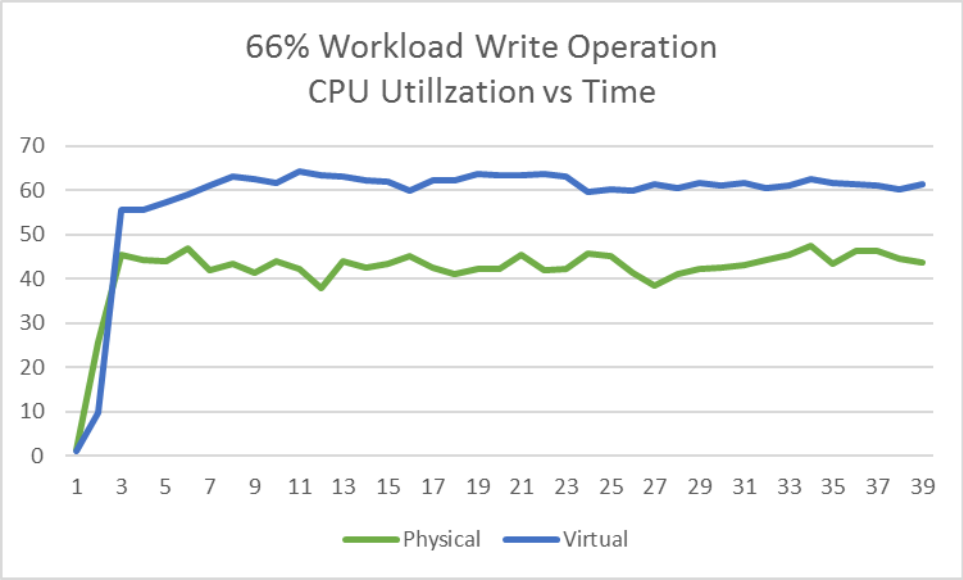

Figure 7 - 66% workload write operation (CPU (%) vs Time (30sec interval))

The results for mixed, read and write operations are shown in Table 3 below for the 66% workload.

Table 3 - 66% workload operations results

Type of Operation	Physical Machines	Standard Deviation	Virtual Machines	Standard Deviation
Mixed	56.98%	3.23	62.05%	2.39
Read	42.26	2.51	67.36%	3.24
Write	41.74%	3.48	58.64%	1.94

5.2 Performance of Cassandra

In this section, we discuss the metrics that demonstrate the effect of virtualization on Cassandra. Throughput of the Cassandra is measured in terms of operations performed per second when a command is executed using the stress tool. The total operations per second displayed at the end of stress tool operation is the average of the values of operations per second measured continuously during the stress tool operation. The latency of a database operation is the response time taken to respond to the operation and begin the database transaction. It is measured in milliseconds in the experiment. Throughput and latency are measured and displayed by the Cassandra-stress tool at the end of every operation. Disk utilization is measured using the Dstat tool.

5.2.1 Throughput and Latency

5.2.1.1 100% Workload

Latency values measured during the stress tool operation for both physical and virtual machine arrangements are shown in the Figure 8 below.

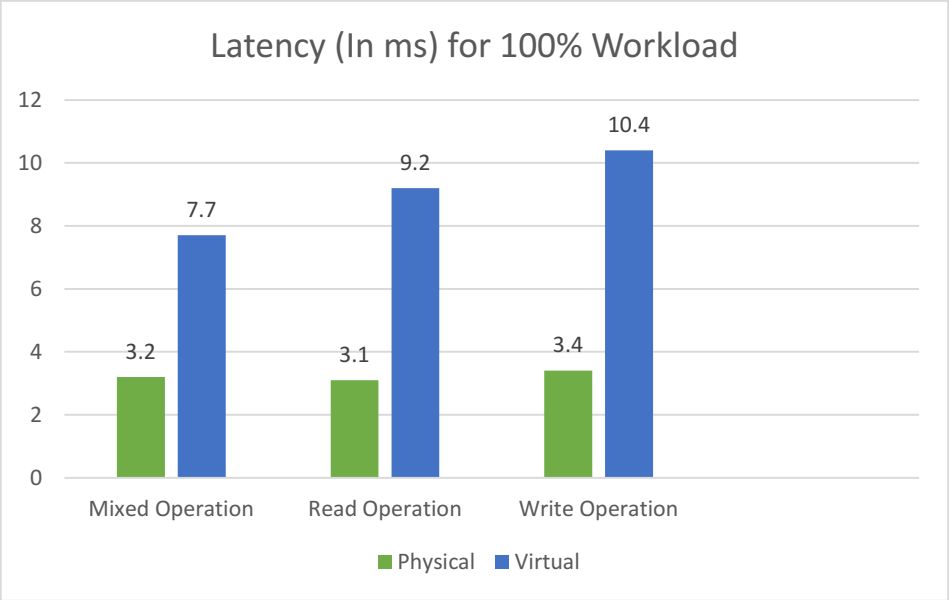

Figure 8 - 100% workload latency

The latency values distribution with 100% workload and the three operations, i.e., mixed, read and write operations is shown below in the form of a box plot for physical machines in the Figure 9 and for virtual machines in the Figure 10.

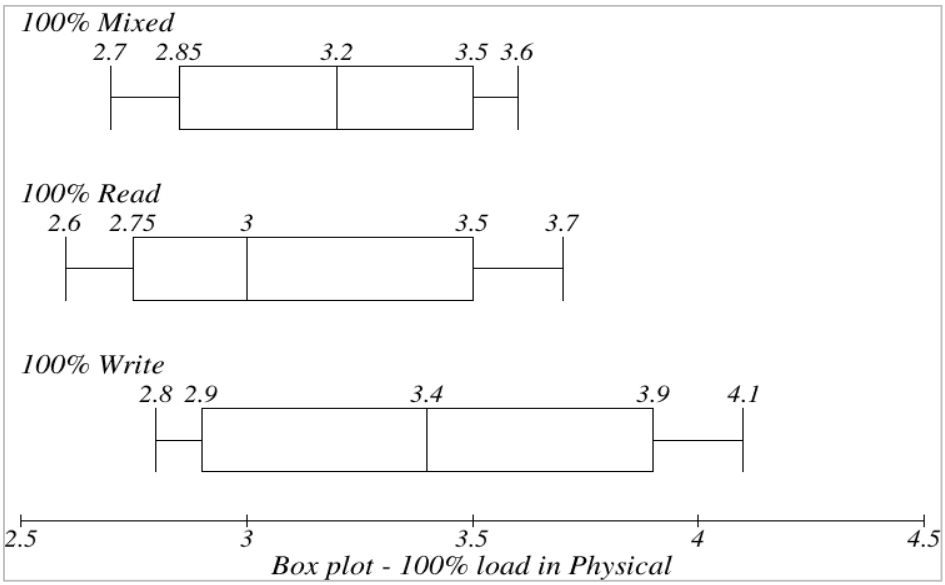

Figure 9 - Box plot for Physical machines with 100% workload

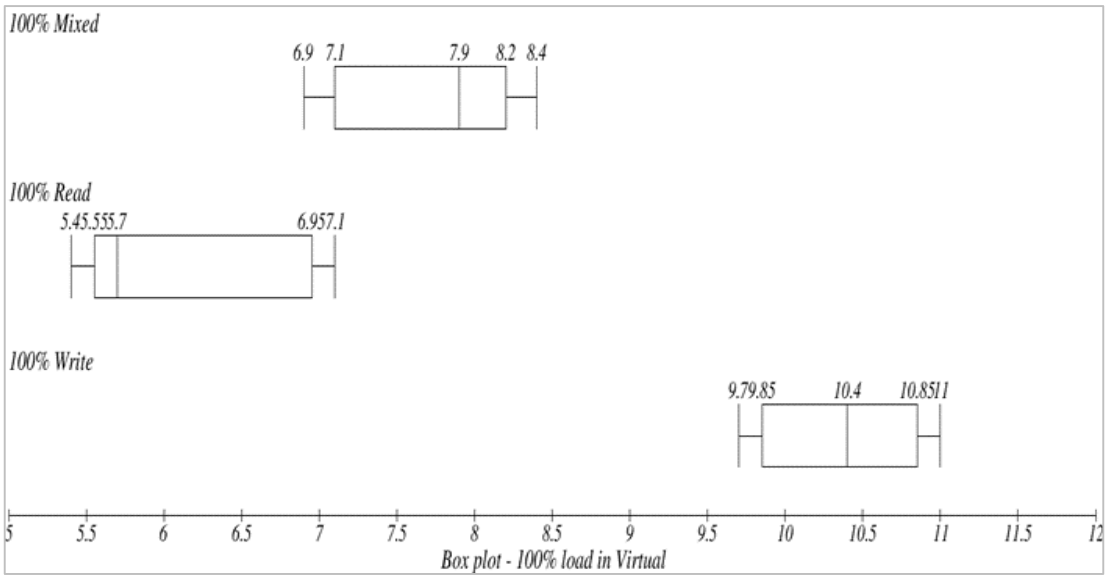

Figure 10 - Box plot for Virtual machines with 100% workload

The throughput or the average of total operations per second for the mixed operation performed for a duration of twenty minutes on a physical machine operation is 139833. The same measurement in a virtual machine arrangement is 58304. Both these values are total operations per second including the read and write operations in 3:1 ratio and are the average of each mixed operation sub step in the ten iterations. The average read operations in a second is measured and displayed by the Cassandra-stress tool. In the physical machine arrangement, the average throughput for read operations is 144622 operations and for the virtual machine arrangement it is 48906 operations per second. Throughput in case of write operations performed using the stress tool in the physical machine arrangement is 133201 operations per second and 43314 operations per second in the virtual machine arrangement.

5.2.1.2 66% Workload

The latency values which is the response time in milliseconds is shown in the Figure 11 below.

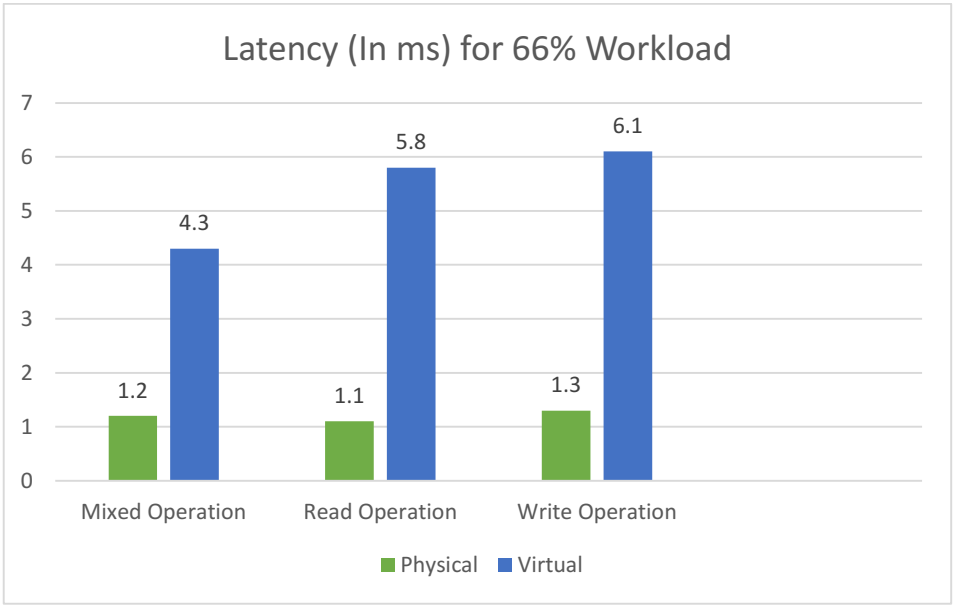

Figure 11 - 66% workload latency

The latency values distribution with 66% workload and the three operations, i.e., mixed, read and write operations is shown in the form of a boxplot for physical machines in the Figure 12 and for the virtual machines in the Figure 13.

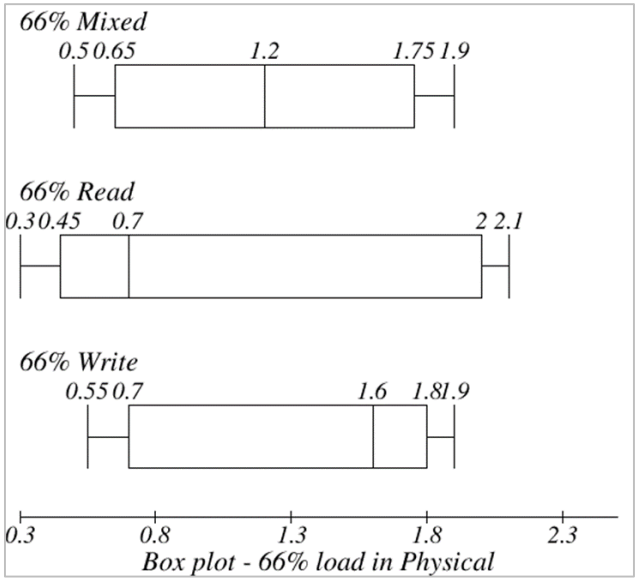

Figure 12 - Box plot for Physical machines with 66% workload

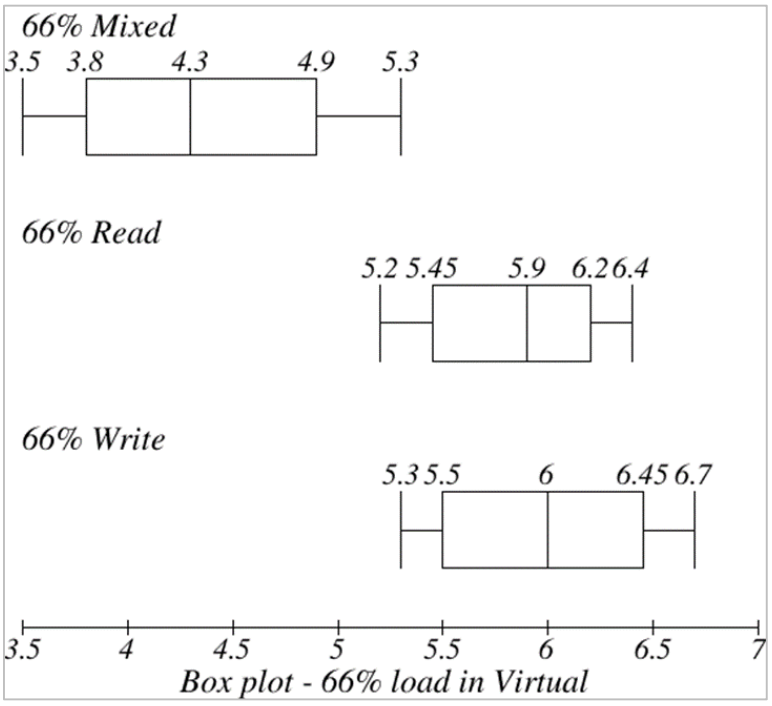

Figure 13 - Box plot for Virtual machines with 66% workload

The throughput values for mixed read and write operations with 66% workload which is set by fixing the thread count to 150 are tabulated below in the Table 4.

Table 4 - 66% workload throughput

Operation	Physical	Virtual
Mixed	110444	34993
Read	110769	24455
Write	86224	111314

5.2.2 Disk Utilization

Dstat tool is used to measure the disk utilization in terms of write operations made (in kB). The disk utilization values for physical and virtual machine arrangements are provided in the subsections below

5.2.2.1 100% Workload

The mean disk utilization for the mixed operation executed using the stress tool for physical machine arrangement is 5068.542 and 9629.292 for virtual machine arrangements. The disk utilization values for both the cases are compared in the Figure 14 below.

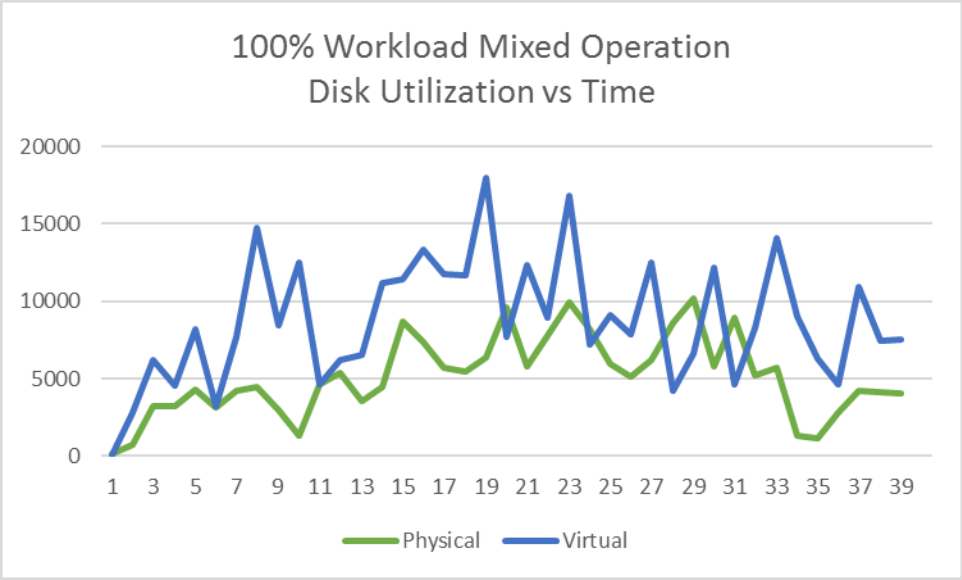

Figure 14 - 100% workload disk utilization mixed operation (Disk vs Time (30sec interval))

For the physical machine arrangement, the mean disk utilization during the write operations is 11883.76 and for the virtual machine arrangement it is 31960.74 and the average values for both the cases are shown in the Figure 15.

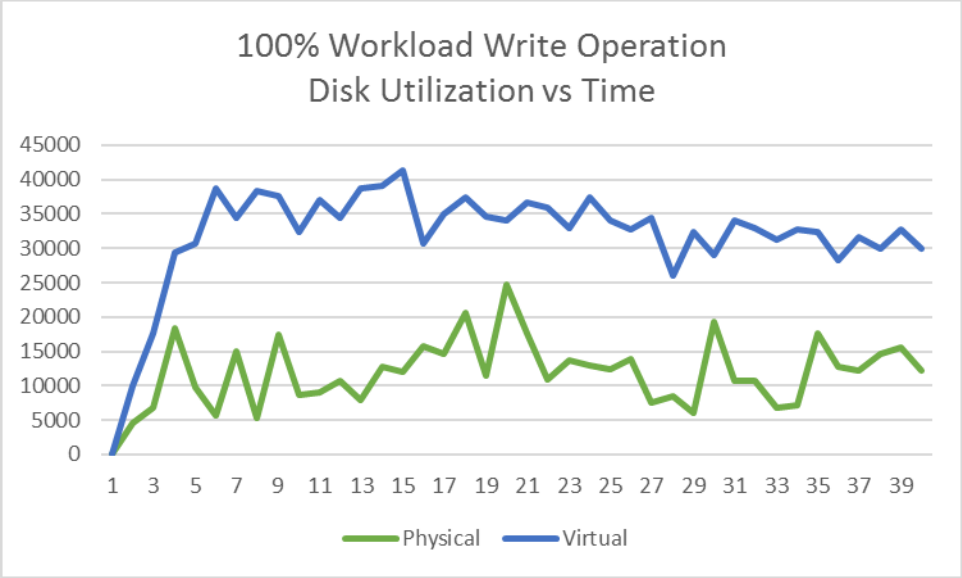

Figure 15 - 100% workload disk utilization write operation (Disk vs Time (30sec interval))

5.2.2.2 66% Workload

The average disk utilization value for physical arrangement during the mixed operation is 3517 whereas for virtual machine arrangement it is 7696. The mean values in both the cases are shown in the Figure 16.

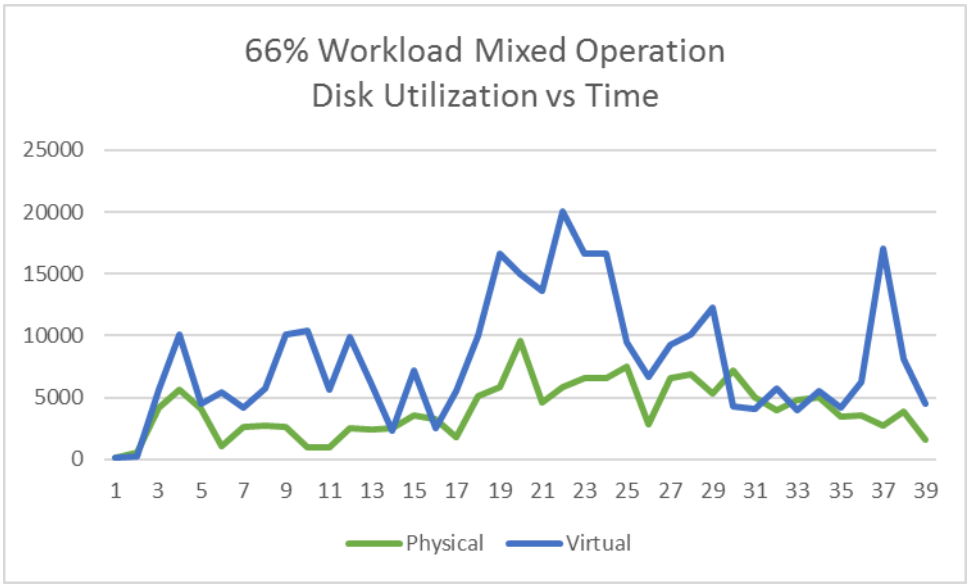

Figure 16 - 66% workload disk utilization mixed operation (Disk vs Time (30sec interval))

The same measurements during the write operation with 66% workload are 6923 for physical machine arrangement and 28169 for virtual machine arrangement. The following Figure 17 shows the average disk utilization in these cases.

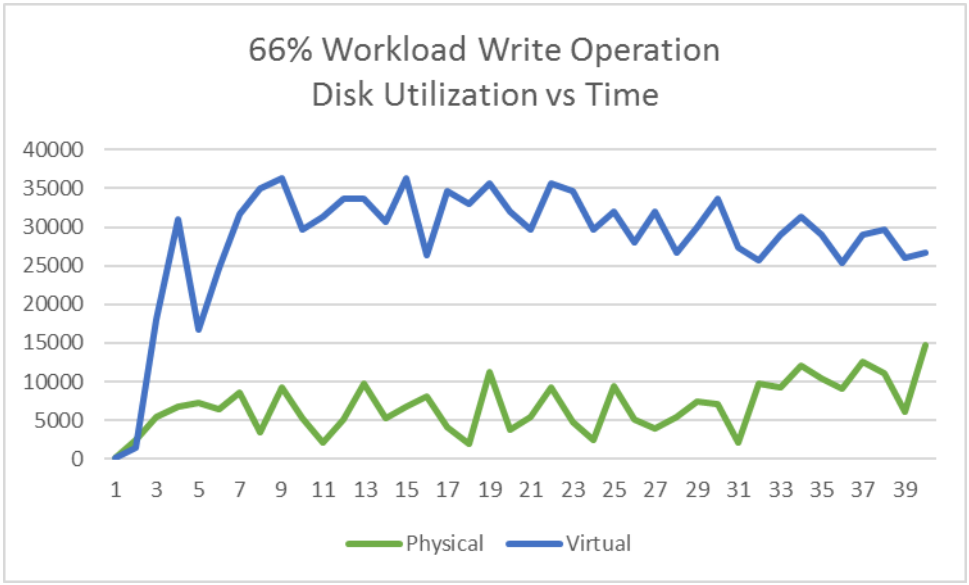

Figure 17 - 66% workload disk utilization mixed operation (Disk vs Time (30sec interval))

5.3 Analysis

5.3.1 CPU Utilization

It has been observed that the CPU utilization has increased in all the cases consisting of mixed, read and write operations with 100% or 66% workload. This increase can be attributed to the fact that the hypervisor or virtual machine monitor running, in this case, KVM uses up some of the CPU processing time for running the virtual machines and operating them.

The average increase in the utilization of CPU for the 100% workload and different operations is tabulated below in Table 5.

Table 5 - Change in CPU utilization for 100% workload

Operation	Increase of Utilization in case of Virtual Compared to Physical
Mixed	6.02%
Read	14.84%
Write	9.79%

The increase in CPU utilization for the three operations with 66% workload is mentioned in Table 6.

Table 6 - Change in CPU utilization for 66% workload

Operation	Increase of Utilization in case of Virtual Compared to Physical
Mixed	5.07%
Read	25.1%
Write	16.9%

The change in CPU utilization was significant in all the cases as it is evident from the Cohen's d effect size, which is used to indicate if the difference between means was small, medium or large. The effect size is calculated using the mean CPU utilization values of each operation in both physical and virtual machines. Their corresponding standard deviation values are used to calculate the pooled standard deviation value which is the square root of average of the standard deviation values in both the cases. Effect size is the value obtained by dividing the mean difference with the pooled standard deviation.

Cohen's d incase of 100% mixed workload is 1.58. This signifies that there was a large difference between the two mean values of CPU utilization in physical machines and virtual machines. The effect size value in case of read operations and100% workload was 3.05 and the same value for write operation was 2.95.

With the 66% workload, the Cohen's d value was 1.78 for mixed operations. For the read and write operations, the values were 8.66 and 7.53 respectively indicating that the overlap between the two mean values of utilization is less than 1%.

It can be safely concluded from the Cohens d values that the standardized difference in the CPU utilization values of physical machines and virtual machines is large in all the cases and hence there is a significant increase in utilization.

5.3.2 Throughput, Latency and Disk Utilization

Throughput is the operations performed in a second. This value can be affected by latency which is the response time or the time taken to perform a database transaction. Latency value can depend on the availability of disk access. When the disk utilization is high, or the disk is being used by other processes, the memory access may take a longer time than expected.

In the experiment, the virtual machine arrangement has utilized more disk compared to the physical machine arrangement as the hypervisor KVM accesses disk more frequently to run the virtual machines and this increase in utilization also leads to the increase in latency and finally leading to a fall in the throughput. Based on the comparison of throughput values

of Cassandra, it can be seen that its performance has decreased by approximately 55% – 75%.

6 DISCUSSION

In this chapter, we discuss the results with respect to the research questions and also the internal and external threats that validate our results.

6.1 Associating the experimental results with the Research Questions

6.1.1 Research Question 1

What is the overhead caused by virtualization in terms of CPU utilization?

The results obtained from the experiment prove that there is overhead caused by the hypervisor KVM. The overhead was measured in terms of increase in the utilization of CPU. In virtual machine arrangement the CPU utilization increased for the:

- 100% workload and mixed operation by 6.02%
- 100% workload and read operation by 14.84%
- 100% workload and write operation by 9.79%
- 66% workload and mixed operation by 8.07%
- 66% workload and read operation by 22.49%
- 66% workload and write operation by 15.9%

In case of mixed operation, it is noticed that the increase in the utilization is alike for both the 100% and 66% workload. It can be observed that read operation usually has a higher utilization of the CPU because Cassandra is faster at write operations compared to read operations as it simply writes to blocks in write operations whereas read involves finding the necessary block.

6.1.2 Research Question 2

How does the virtualization affect the performance of Cassandra in terms of disk utilization, throughput and latency?

Virtualization has resulted in the increase in the disk operations, which led to the increase of latency of Cassandra operations. This increase in latency has affected the throughput or the number of operations per second and resulted in a lower throughput compared to the physical machine arrangement. Hence, virtualization has led to the decrease in the throughput of Cassandra in all the cases.

6.2 Threats to Validity

6.2.1 Internal threats

According to [28], internal threats are usually occurring casually without the knowledge of a researcher. These threats may arise due to the problem in either selection of the components of an experiment or the instrumentation required for the experiment. It is suggested to make sure that the mistakes or errors occurred in the previous research are not repeated again. The internal threat of error commitment is reduced by repeating the experiment at every step for ten iterations and the average of the results obtained is measured.

6.2.2 External threats

In [28], whether the results of an experiment can be generalized to other scenarios or environments is referred as an external threat. Though the exact results could not be generalized, it can be agreed upon that the results would be valid for a few other abstract applications or environments rather than specialized ones.

7 CONCLUSION AND FUTURE WORK

7.1 Conclusion

The main aim of this research was to investigate the effects of virtualization of Cassandra by evaluating its performance in physical machine arrangement and then comparing it with virtual machine arrangement.

The experiment begins with finding the maximum performance by identifying the thread count at which highest operations per second is achieved using the write operation of the Cassandra stress tool. Based on the values obtained, the experiment is performed by specifying the thread count where the load was 100% and 66%.

The experiment was divided into sub steps where initially the database was populated with fifty million writes by writing data into the tables and keyspaces using the write command of Cassandra-stress tool. Once the data was written into the keyspaces, by specifying the thread count of 450 where maximum performance was noticed, initially a mixed operation which consisted of read and write operations in the ratio of 3:1 was performed using the stress tool for a duration of twenty minutes. After the mixed operation, a read only operation using the stress tool was performed for twenty minutes, followed by the write only operation for twenty minutes. These sub steps were performed in the same order for ten iterations to obtain consistent results and the same procedure was performed with a 66% load by specifying the thread count to be 150. The exact same procedure was repeated on the virtual machine arrangement as well.

The results obtained significantly indicated an increased utilization of the processor in case of virtual machines. This could be attributed to the overhead caused by the virtualization software, hypervisor, which is KVM in our case. An increase in the disk utilization has been observed when Cassandra was run on virtual machine arrangement and this affected the latency of stress tool operations. The increase in latency which is the average response time leads to the decrease in the number of operations made in a second, i.e., throughput.

7.2 Future Work

In this research, the primary focus was on the overhead caused by virtualization in terms of CPU utilization and its effects on the performance of Cassandra. Based on this, future research can be extended as mentioned below.

- For our research, KVM has been chosen as the hypervisor. Comparing the performance of Cassandra on different hypervisors can be explored to pick the best hypervisor overall or for individual operations like mixed or read only or write only.
- Database initially was populated with fifty million writes and operations were performed for a duration of twenty minutes on this database at 100% and 66% workloads. Various other workloads and operations on different database sizes can be performed to probe the effects on different workloads.
- Experiment has been primarily conducted with telecommunication systems applications in focus, performance of various other applications can be evaluated in future research.
- Fine tuning the parameters of Cassandra can lead to improvement in the performance. Factors like compaction strategies can be further studied from the perspective of Cassandra running in virtualized environments.
- The hardware used was provided by the university. Using a different processor or other hardware may lead to different results which can be studied in future.

REFERENCES

[1] A. Lakshman, "Cassandra - A Decentralized Structured Storage System."ACM SIGOPS Operating Systems Review, vol. 44, no. 2, pp. 35–40, 2010.

[2] S. D. Kuznetsov and A. V Poskonin, "NoSQL Data Management Systems," vol. 40, no. 6, pp. 323–332, 2014.

[3] M. Çal, "Benefits of the Virtualization Technologies with Intrusion Detection and Prevention Systems."

[4] S. Shirinbab, L. Lundberg, and D. Ilie, "Performance Comparison of KVM , VMware and XenServer using a Large Telecommunication Application."

[5] S. Fukuda, R. Kawashima, S. Saito, and H. Matsuo, "Improving Response Time for Cassandra with Query Scheduling," 2013.

[6] C. Feng, Y. Zou, and Z. Xu, "CCIndex for Cassandra : A Novel Scheme for Multi-dimensional Range Queries in Cassandra," no. 3, 2011.

[7] E. Dede, B. Sendir, P. Kuzlu, J. Hartog, and M. Govindaraju, "An Evaluation of Cassandra for Hadoop," pp. 494–501, 2013.

[8] P. Bagade, A. Chandra, and A. B. Dhende, "Designing Performance Monitoring Tool for NoSQLCassandra Distributed Database," 2012.

[9] V. Abramova, J. Bernardino, and P. Furtado, "Evaluating Cassandra Scalability with YCSB," pp. 199–207, 2014.

[10] P. Mell, T. Grance, and T. Grance, "The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology."

[11] M. Armbrust, A. D. Joseph, R. H. Katz, and D. A. Patterson, "Above the Clouds : A Berkeley View of Cloud Computing," 2009.

[12] U. Gurav, "Virtualization – A key feature of cloud computing," no. Icwet, pp. 227–229, 2010.

[13] R. Uhlig, L. Amy, V. Andrew, M. Steven, A. Kägi, H. Felix, and L. Smith, "Intel Virtualization Technology," 2005.

[14] R. Morabito, R. Morabito, J. Kjällman, and M. Komu, "Hypervisors vs . Lightweight Virtualization : A Performance Comparison Hypervisors vs . Lightweight Virtualization : a Performance Comparison," no. March, 2015.

[15] S. E. Kassahun, "A PMIPv6 Approach to Maintain Network Connectivity during VM Live Migration over the Internet," 2013.

[16] S. P. Ahuja and D. Ph, "Full and Para Virtualization," 2010.

[17] J. Che, Q. He, Q. Gao, and D. Huang, "Performance Measuring and Comparing of Virtual Machine Monitors ∗ ," pp. 381–386, 2008.

[18] A. Kivity, U. Lublin, and A. Liguori, "kvm : the Linux Virtual Machine Monitor," pp. 225–230.

[19] E. Hewitt, *Cassandra - The Definitive Guide*. 2010.

[20] "Documentation - Apache Cassandra ™," 2016.

[21] G. Martins, P. Bezerra, R. Gomes, and A. Costa, "Evaluating Performance Degradation in No S QL Databases Generated by Virtualization."

[22] Y. Abubakar and I. G. Auta, "Performance Evaluation of NoSQL Systems Using YCSB in a resource Austere Environment," vol. 7, no. 8, pp. 23–27, 2014.

[23] A. Elsayed, "Performance Evaluation and Comparison of The Top Market Virtualization Hypervisors," pp. 45–50, 2013.

[24] L. Cherkasova and R. Gardner, "Measuring CPU Overhead for I / O Processing in the Xen Virtual Machine Monitor," pp. 387–390, 2005.

[25] M. Berndtsson, J. Hansson, B. Olsson, and B. Lundell, *Thesis Projects A Guide for Students in Computer Science and Information Systems*. 2008.

[26] Committee on Academic Careers for Experimental Computer and Scientists, *Academic Careers for Experimental Computer Scientists and Engineers*. 1994.

[27] "Dstat Documentation." [Online]. Available: https://github.com/dagwieers/dstat/blob/master/docs/dstat.1.adoc.

[28] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén,

Experimentation in software engineering. Springer, 2012.