

Comparative Analysis of the Performance for Cloud Computing Hypervisors with Encrypted Algorithms

تحليل مقارنة لأداء مراقب الأجهزة الافتراضية في بيئة الحوسبة
السحابية عند استخدام خوارزميات التشفير

By

Waleed Khaled Amin Abdulraheem

Supervisor

Dr. Ahmad Kayed

This thesis is submitted to the Department of Computer Information Systems, Faculty
of Information Technology, Middle East University in partial fulfillment of the
Requirements for Master Degree in Computer Information Systems.

Faculty of Information Technology

Middle East University

Amman, Jordan

(May, 2014)

AUTHORIZATION STATEMENT

I, Waleed Khaled Abdulrahim, authorize the Middle East University to provide hard copies or electronic copies of my thesis to libraries, institutions or individuals upon their request.

Name: Waleed Khaled

Date: June 4th, 2014

Signature: 

Name: Waleed Khaled

Date: June 4th, 2014

Signature:

إقرار تفويض

أنا وليد خالد عبدالرحيم، أفوض جامعة الشرق الأوسط للدراسات العليا بتزويد نسخ من رسالتي ورقياً أو إلكترونياً للمكتبات أو المنظمات أو الهيئات والمؤسسات المعنية بالأبحاث والدراسات العلمية عند طلبها.

الاسم: وليد خالد عبدالرحيم

التاريخ: ٢٠١٤/٦/٤

التوقيع: 

Examination Committee Decision

This is to certify that the thesis entitled “Resource Allocation Technique to Obtain Energy Efficient Cloud” was successfully defended and approved on May 10th, 2014

Examination Committee Members

Signature

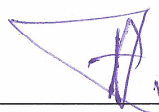
(Head of the Committee and Supervisor)

Dr. Ahmad Kayed

Associate Professor

Dean Faculty of IT

Middle East University



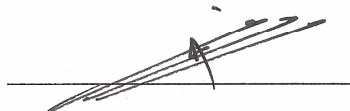
(Internal Committee Member)

Dr. Mamoun Khaled

Assistant Professor

Vice Dean Faculty of IT

Middle East University



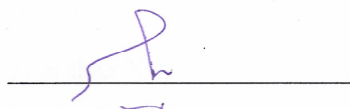
(External Committee Member)

Dr. Bayan Abushawar

Assistant Professor

Dean Faculty of IT

Arab Open University



ABSTRACT

Cloud Computing (CC) is Internet-based computing, where end users are provided with on demand shared resources, software and information. Security is being a major issue in the cloud, and it arise attention for Cloud Service Providers (CSP) and end users. Cloud security problem raises suspicions and makes many organizations refuse the idea of using the cloud in storing certain data within the cloud; especially data with high confidentiality. In addition, cloud users try to avoid being controlled by the CSPs. Encryption Algorithm (EA) is one of many techniques that is used to make data on the cloud secured. In this research, we applied EA on two different cloud hypervisors Xen and KVM. The possibility to measure and compare the performance of the two hypervisors is being explored. Performance in our research takes into consideration response time or duration at encrypt and decrypt and CPU utilization. We set the cloud environment and create instances. Eight encryption algorithms (EA) from different categories are deployed for each instance. These EAs are RSA, AES, DES, TripleDES, ARC4, CAST-128, Blowfish, and TwoFish. They are differ in properties such as Asymmetric, symmetric, block cipher, stream cipher, and key encryption algorithm. Many parameters are taken to compare such as key, number of cores, and data size. Results show that KVM is better than Xen at all CPU utilization results with 11% in average, while at time response KVM also is better at most results with 11.5% in average. TripleDES shows better time response at Xen at all results. RSA at big key and TwoFish at small data also show better response time at Xen.

Keywords: Cloud Computing, Xen, KVM, Encryption Algorithm, Performance.

الملخص

تعتبر الحوسبة السحابية مبنية على الوصول الى الشبكة العنكبوتية, حيث انها تشارك المصادر المتاحة والبرمجيات والمعرفة عند طلبها. لكنها من جهة اخرى تجعل مشكلة أمن المعلومات اكثر تعقيدا لكلا من مزودي خدمة الحوسبة السحابية وكذلك العملاء المستخدمين لهذه الخدمة. مشكلة أمن الحوسبة السحابية جعلت الكثير من المؤسسات تحجم عن استخدام هذه التكنولوجيا لأنهم لا يريدون ان يتحكم بمعلوماتهم مزودي الخدمة. تعتبر خوارزميات التشفير احدى الطرق التي تستخدم لجعل المعلومات في الحوسبة السحابية اكثر امنا. لقد طبقنا خوارزميات التشفير تلك على اثنين من اشهر مراقبي الأجهزة الافتراضية وهما Xen و KVM. هذه الرسالة تحاول ان تقارن وتناقش قياس أداء هاتان الأداتان. الأداء نقصد به وقت الاستجابة ونسبة استخدام وحدة المعالجة المركزية. لقد جهزنا بيئة الحوسبة السحابية وانشأنا نسختين افتراضيتين متطابقتين من أنظمة التشغيل. استخدمنا في كل نسخة منهن ثمانية خوارزميات تشفير وهم: RSA و AES و DES و TripleDES و ARC4 و CAST-128 و Blowfish وأخيرا TwoFish. هذه الخوارزميات مختلفة الخصائص, فبعضها متماثل وبعضها غير متماثل, بعضها يستخدم التشفير بالكتلة والبعض يستخدم التشفير بالتيار و فيما يستخدم البعض الآخر التشفير بواسطة المفتاح. لقد استخدمنا عدة معايير لمقارنة الأداء, كحجم المفتاح وعدد وحدات المعالجة المركزية وحجم البيانات المستخدمة. كذلك استخدمنا لتحليل البيانات عدة طرق مثل المتوسط الحسابي والانحراف المعياري ومعامل الارتباط. لقد أظهرت النتائج ان KVM هو افضل من Xen في معدل استهلاك وحدة المعالجة المركزية وبنسبة 11% كمتوسط, وكذلك افضل بالنسبة الى وقت الإستجابة وبنسبة تقدر ب 11.5%. خوارزمية التشفير TripleDES اظهرت أفضل أداء في وقت الإستجابة عند استخدام Xen. كذلك خوارزمية RSA في حالة المفاتيح كبيرة الحجم و خوارزمية TwoFish عند حجم البيانات الصغير أظهرت أفضل أداء في وقت الإستجابة عند استخدام Xen كذلك.

كلمات البحث: الحوسبة السحابية, Xen, KVM, خوارزميات التشفير, الأداء

Dedication

To My Mother and Father...

My Beloved Parents,

To my sister...

"Huda..."

And all of my sisters

To my Brothers

To the light of my life ...

My Wife "Bushra" ,

My Son "Yousef"

And My Two Daughters "Nada, Hala" ..

Acknowledgement

I'm particularly grateful to Dr. Ahmed Kayed supervising, helping and encouraging my efforts during this research and to all of my teachers for their support, especially Dr. Heba Naseruddin, Dr.Ma'moun Khaled and Dr.Bayan Abushawar.

I owe more than thanks to Dr.Samir AbuTahour for setting the environment. He supports me technically, physically, and psychologically. To the polite man Ammar Zakarneh.

I would like to thank my friends Ahmad Abulhaijaa and Eng.Mohammad AlGzawi for their supporting and to my friend Taleen Akijian for English reviewing.

For the past two years, my son and two daughters kept on reminding me to go to study and write the thesis! I'm very grateful to them.

Finally, and most importantly, I would like to thank my wife. Her support, encouragement, quiet patience and unwavering love.

Thank You All..

Table of Contents

Authorization Statement	i
Examination Committee Decision	iii
Abstract	iv
Abstract in Arabic	v
Dedicate.....	vi
Acknowledgements	vii
List of Figures.....	x
List of Tables.....	xiii
List of Abbreviations.....	xiv
Chapter 1	1
Introduction.....	1
1.1. Overview.....	1
1.2. Cloud Computing.....	1
1.3. Cryptography	6
1.4. Problem Statement	8
1.5. Research Questions.....	8
1.6. Objectives.....	9
1.7. Methodology	9
Chapter 2	10
Background and Literature Review	10
2.1. Introduction.....	10
2.2. Background.....	10
2.2.1 RSA	10
2.2.2. AES	11
2.2.3 DES	12
2.2.4 TripleDES	13
2.2.5. ARC4	14
2.2.6. CAST-128	14
2.2.7. BlowFish	14
2.2.8. TwoFish	15
2.3. Literature Review	16

2.4. Related Work.....	18
Chapter 3.....	20
Experiment Design.....	20
3.1. Introduction.....	20
3.2. Setting the Environment	20
3.3. Extract Data	23
3.5. Peoposed Approach	24
3.6. Result Analysis	27
Chapter 4.....	27
Results Analysis	27
4.1. Overview.....	27
4.2. Encryption Algorithms Results and Analysis	27
4.2.1. RSA	27
4.2.2. AES	39
4.2.3. DES	46
4.2.4. TripleDES	52
4.2.5. ARC4	58
4.2.6. CAST-128	64
4.2.7. BlowFish	70
4.2.8. TwoFish	75
4.3. Summary Results	81
Chapter 5.....	86
Conclusion and Future Work.....	86
5.1. Conclusion	86
5.1.1 Response time performance conclusion	86
5.1.1 CPU utilization performance conclusion	87
5.2. Research Contribution.....	87
5.3. Future Works	89
References and Links	90
Appendix.....	93

LIST OF FIGURES

Figure 2.1: AES structure (Zhiyi et.al 2013).	12
Figure 3.1: The proposed Approach	25
Figure 4.1: Encryption time for Xen and KVM	30
Figure 4.2: Decryption time for Xen and KVM	31
Figure 4.3: Xen time for encryption and decryption	31
Figure 4.4: CPU utilization of two hypervisors while encryption	32
Figure 4.5: CPU utilization of two hypervisors while Decryption.	32
Figure 4.6: Effect of Thread on Response Time	35
Figure 4.7: Effect of Thread on CPU utilization	35
Figure 4.8: Encryption time for hypervisors when changing keys.	36
Figure 4.9: Decryption time for hypervisors when changing keys.	36
Figure 4.10: Encryption CPU for hypervisors when changing keys.	37
Figure 4.11: Decryption CPU for hypervisors when changing keys.	37
Figure 4.12: Enc\Dec for different data size over Xen and KVM	38
Figure 4.13: Represents encryption time for Xen and KVM.	39
Figure 4.14: Represents decryption time for Xen and KVM.	40
Figure 4.15: The CPU at Encryption over Xen and KVM	40
Figure 4.16: The CPU at Decryption over Xen and KVM	40
Figure 4.17: Effect of Changing Core Number on Response Time	42
Figure 4.18: Effect of Changing Core Number on CPU Utilization	43
Figure 4.19: Effect of Changing Key on Time	43
Figure 4.20: Effect of changing key on CPU	44
Figure 4.21: Effect of Changing Data Sizes on Response Time	44
Figure 4.22: Effect of Changing Data Sizes on CPU Utilization	45
Figure 4.23: Encryption Time for Xen and KVM.	46
Figure 4.24: Decryption Time for Xen and KVM.	47
Figure 4.25: The CPU at Encryption over Xen and KVM	47
Figure 4.26: The CPU at Decryption over Xen and KVM	47
Figure 4.27: Effect of changing core number on time response	49
Figure 4.28: Effect of Changing Core Number on CPU Utilization	49
Figure 4.29: Effect of Changing Data Size on Response Time	50

Figure 4.30: Effect of changing data size on CPU utilization	51
Figure 4.31: Represents encryption time for Xen and KVM.	52
Figure 4.32: Represents decryption time for Xen and KVM.	53
Figure 4.33: Represent the CPU at encryption over Xen and KVM	53
Figure 4.34: Represent the CPU at decryption over Xen and KVM	53
Figure 4.35: Effect of changing core number on response time	55
Figure 4.36: Effect of Changing Core Number on CPU Utilization	55
Figure 4.37: Effect of Changing Data Size on Time Response	56
Figure 4.38: Effect of Changing Data Size on CPU Utilization	57
Figure 4.39: Represents Encryption Time for Xen and KVM.	58
Figure 4.40: Represents Decryption Time for Xen and KVM.	59
Figure 4.41: Represent the CPU at Encryption over Xen and KVM	59
Figure 4.42: Represent the CPU at Decryption over Xen and KVM	59
Figure 4.43: Effect of Changing Core Number on Response Time	61
Figure 4.44: Effect of Changing Core Number on CPU Utilization	61
Figure 4.45: Effect of Changing Key on Time	62
Figure 4.46: Effect of Changing the Key on CPU	62
Figure 4.47: Effect of changing Data Size on Response Time	63
Figure 4.48: Effect of changing the Data Size on CPU Utilization	63
Figure 4.49: Encryption Time for Xen and KVM	65
Figure 4.50: Represents Decryption Time for Xen and KVM.	65
Figure 4.51: The CPU at Encryption over Xen and KVM	65
Figure 4.52: Represent the CPU at Decryption over Xen and KVM	66
Figure 4.53: Effect of Changing Core Number on Response Time	67
Figure 4.54: Effect of changing core number on CPU utilization	68
Figure 4.55: Effect of changing data size on time response	68
Figure 4.56: Effect of changing data size on CPU utilization	69
Figure 4.57: Represents Encryption Time for Xen and KVM	70
Figure 4.58: Represents Decryption Time for Xen and KVM	71
Figure 4.59: Represent the CPU at Encryption over Xen and KVM	71
Figure 4.60: Represent the CPU at decryption over Xen and KVM	71
Figure 4.61: Effect of Changing Core Number on Response Time	73
Figure 4.62: Effect of Changing Core Number on CPU Utilization	73
Figure 4.63: Effect of Changing Data Size on time Response	74

Figure 4.64: Effect of Changing Data Size on CPU Utilization	74
Figure 4.65: Represents encryption time for Xen and KVM.	76
Figure 4.66: Represents decryption time for Xen and KVM.	76
Figure 4.67: Represent the CPU at encryption over Xen and KVM	76
Figure 4.68: Represent the CPU at decryption over Xen and KVM	77
Figure 4.69: Effect of changing core number on response time	78
Figure 4.70: Effect of changing core number on CPU utilization	79
Figure 4.71: Effect of changing key on time	79
Figure 4.72: Effect of changing key on CPU	80
Figure 4.73: Average encryption response time for all EA	82
Figure 4.74: Average decryption response time for all EA	82
Figure 4.75: Average encryption CPU utilization for all EA	84
Figure 4.76: Average decryption CPU utilization for all EA	84

LIST OF TABLES

Table 4. 1: RSA experiments result	28
Table 4.2: Results of Time and CPU on Hypervisors	30
Table 4.3: Average Time and CPU for RSA	33
Table 4.4: Correlation between RSA Results	34
Table 4.5: Result of all AES experiments.	39
Table 4.6: Average Time and CPU over AES experiments	41
Table 4.7: Correlation for AES Results	42
Table 4.8: Result of all DES experiments	46
Table 4.9 Average Time and CPU for Xen and KVM	48
Table 4.10: Correlation between all DES columns	48
Table 4.11: Result of all DES experiments	52
Table 4.12: Average Time and CPU for Xen and KVM	54
Table 4.13: Correlation between all 3DES columns	54
Table 4.14: Result of all ARC4 Experiments	58
Table 4.15 Average Time and CPU for Xen and KVM	60
Table 4.16: Correlation between all ARC4 columns	60
Table 4.17: Result of all CAST-128 Experiments	64
Table 4.18 Average Time and CPU for Xen and KVM	66
Table 4.19: Correlation between all CAST-128 Columns	67
Table 4.20: Result of all BlowFish experiments	70
Table 4.21 Average Time and CPU for Xen and KVM	72
Table 4.22: Correlation between all BlowFish Results	72
Table 4.23: Result of all TwoFish experiments	75
Table 4.24 Average Time and CPU for Xen and KVM	77
Table 4.25: Correlation between all TwoFish columns	78
Table 4.26: Table of all EA response time results	81
Table 4.26: Table of all EA CPU utilization results	83
Table 4.27: Correlation of all EA components	85

List of Abbreviations

3DES: Triple Data Encryption Standard.

AES: Advanced Encryption Standard.

ARC4: Alleged Ron's Code 4.

BSD: Berkeley Software Distribution.

CAST-128: Carlisle Adams and Stafford Tavares.

CC: Cloud Computing.

CPU: Central Processing Unit.

CSP: Cloud Service Provider

Dec: Decryption.

DES: Data Encryption Standard.

Dom 0: Domain Zero.

EA: Encryption Algorithm.

Enc\Dec: Encryption and Decryption.

Exp No: Experiment Number.

FV: Full-virtualization.

HPC: High Performance Computing.

IaaS: Infrastructure as a Service.

KVM: Kernel-based Virtual Machine.

KVM: Kernel-based Virtual Machine

NIST: U.S. National Institute of Standards and Technology.

Nr: Number.

OS: Operating System.

PaaS: Platform as a Service.

PHT: Pseudo-Hadamard Transform.

PV: Para Virtualization.

QEMU: Quick Emulation.

QEMU: Quick Emulator.

RAM: Random Access Memory.

RSA: Ron Rivest, Adi Shamir and Leonard Adleman.

SaaS: Service as a Service.

STDEV: standard deviation.

TLS: Transport Layer Security.

Utili: Utilization.

VM: Virtual Machine.

CHAPTER ONE

Introduction:

1.1 Overview

In this chapter, brief background about the scope of the thesis is presented; Cloud Computing, Hypervisors in the Cloud, Encryption Algorithms. Then an idea about the research problem is given and how it has been addressed, thesis questions, and research objectives.

1.2 Cloud Computing

In recent years, the term cloud computing has been used to identify an evolution paradigm in the computer industry. That is because of set of advanced technologies that affect the focus of the organizations and businesses on the cost. The base of the cloud computing is the evolution of three properties, virtualization, grid computing and web services. The increasing of Internet connection, mobile accessibility and portable devices has encouraged the spread of applications created for this environment and the access to available resources exclusively through the internet (Ercolani, G. 2013).

There are many definitions mentioned in papers and books, the standard definition of (CC) from NIST (National Institute of Standards and Technology) defines the cloud computing:" Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned

and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models" (Mell, P. et.al 2011)

Essential characteristics are on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service. Three service models Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). Four deployment models are private cloud, community cloud, public cloud and hybrid cloud.

As mentioned before that the base of the cloud computing is the evolution of three properties; virtualization, grid computing, and web services. Virtualization is one of the most important element that make the cloud, it helps organizations enabling much greater consolidation within private data centers, and more recently as a driving technology behind cloud computing.

Virtualization enables new features such as performance management and reliability services to be applied without requiring modifications to applications or operating systems. The operating system (OS) on the virtual machine is called guest, in the management layer the virtual machine monitor (VMM) or called the hypervisor response to create and control all virtual machines in virtual environment (Hwang J. et al.2013).

A hypervisor is one of many virtualization techniques which allow multiple operating systems (guests) to run concurrently on a host computer. The hypervisor presents to

the guest operating systems a virtual operating platform and monitors the execution of the guest operating systems. Multiple instances of a variety of operating systems may share the virtualized hardware resources. Hypervisor is installed on server hardware whose only task is to run guest operating systems (Sabahi, F. 2012).

In virtualization architecture, there are two different approaches: Type one Para-virtualization (PV), and Type two Full-virtualization (FV). Para-virtualization requires modification to the guest OS, essentially teaching the OS how to make requests to the hypervisor when it needs access to restricted resources. Full-Virtualization is designed to provide a complete simulation of the underlying physical system and creates a complete virtual system in which the guest operating systems can execute. No modification is required in the guest OS or application. This approach can be preferred because it enables complete decoupling of the software from the hardware.

Many kind of virtual platforms differ from open-source as KVM and Xen (that we used in this research) to commercial platform like VMware vSphere and Microsoft Hyper-V, their goal is one to manage the guest OS, but they differ in underlying technologies (Hwang J. et al. 2013).

In this research we used the most two famous kinds of hypervisors, Xen and KVM. Xen¹ is a very famous Para-virtualization solution, originally developed at the University of Cambridge. It's the only open source with bare-metal solution, using in many cloud providers like Amazon EC2. It consists of several components that work

¹ <http://www.xenproject.org/>

together to deliver the virtualization environment including Xen Hypervisor, Domain 0 Guest (referred as Dom0) which represent the layer zero or hardware, and Domain U Guest (Fayyad-Kazan et al.2013).

Responsibilities of the hypervisor include memory management and CPU scheduling of all virtual machines ("domains"), and for launching the most privileged domain ("dom0") - the only virtual machine which by default has direct access to hardware. From the dom0 the hypervisor can be managed and unprivileged domains ("domU") can be launched.

The dom0 domain is typically a version of Linux, or BSD (Berkeley Software Distribution Unix). User domains may either be traditional operating systems, such as Microsoft Windows under which privileged instructions are provided by hardware virtualization instructions (if the host processor supports x86 virtualization, e.g., Intel VT-x and AMD-V), or Para-virtualized operating system whereby the operating system is aware that it is running inside a virtual machine, and so makes hyper calls directly, rather than issuing privileged instructions.

KVM² (Kernel-based Virtual Machine) is very famous Full-virtualization solution. Using KVM, one can run multiple virtual machines running unmodified Linux or Windows images. KVM runs as a kernel module like (kvm-intel.ko/ kvm-amd.ko) and (kvm.ko) that's provides the core virtualization infrastructure, which means it uses most of the features of the Linux kernel operating system itself. For example, rather than providing its own CPU scheduler for VMs, KVM treats each VM as a process

² http://www.linux-kvm.org/page/Main_Page

and uses the default Linux scheduler to allocate resources to them. QEMU (Quick Emulator) is added also to KVM for emulating input and output.

OpenStack³ is a global collaboration of developers and cloud computing technologists producing the ubiquitous open source cloud computing platform for public and private clouds. OpenStack project aims to deliver solutions for all types of clouds by being simple to implement, massively scalable, and feature rich. The technology consists of a series of interrelated projects delivering various components for a cloud infrastructure solution.

We use OpenStack because most people use it and it's well known. Many companies produce open source cloud computing software for creating, managing, and deploying infrastructure cloud services like CloudStack, but in OpenStack project more than 200 companies have joined the project, including Arista Networks, AT&T, AMD, Canonical, Cisco, Dell, EMC, Ericsson, Go Daddy, Hewlett-Packard, IBM, Intel, NEC, NetApp, Nexenta, Red Hat, SUSE Linux, Mellanox, VMware, Oracle and Yahoo.

Many programs and applications are used to set the cloud environment. Debian GNU/Linux⁴ is frequently used as operating system for cloud servers; Debian GNU/Linux is a particular distribution of the Linux operating system, and numerous packages that run on it. Debian is open source and free to use has many features make it one of the best server operating system.

³ <https://www.openstack.org/>

⁴ <https://www.debian.org/>

Proxmox VE⁵ is a complete open source virtualization management solution for servers. It is based on KVM virtualization and container-based virtualization and manages virtual machines, storage, virtualized networks, and high availability Clustering.

1.3 Cryptography:

Cryptography is the art of keeping messages secured. Nowadays the growth of the Internet and electronic commerce has brought to the forefront the issue of privacy in electronic communication. Large volumes of personal and sensitive information are electronically transmitted and stored every day. (Prasanthi O. et al 2012).

Here are some cryptographically terminals definitions, the original message before being transformed is called plaintext. After the message is transformed, it is called cipher text. An encryption algorithm transforms the plaintext into cipher text; a decryption algorithm transforms the cipher text back into plaintext. Asymmetric EA that use two different keys for encryption and decryption, while symmetric EA that use same key for both encryption and decryption.

Encryption algorithm performance is effect with the hardware specification such as number of core and number of thread. Core or processor core is an individual processor within a CPU. Many computers today have multi-core processors, meaning the CPU contains more than one core. While threads allow the computer program to execute sequential actions or many actions at once.

⁵ <https://www.proxmox.com/>

In cloud computing our own data is not stored in our private computer. So many techniques used to keep privacy in CC like isolated, authentication, etc. cryptography of data is one of those popular ways to secure data in the cloud.

At this research, many well-known encryption algorithms (EA) are used; they are RSA, AES, DES, 3DES, ARC4, CAST-128, BLOWFISH, and TWOFISH. Choosing these EA depend on their variety. EA kinds such as Asymmetric, Symmetric, Block cipher and Stream cipher (Prasanthi O. et al 2012).

RSA stands for Ron Rivest, Adi Shamir and Leonard Adleman. AES stands for Advanced Encryption Standard. DES stands for Data Encryption Standard. 3DES stands for Triple Data Encryption Standard. ARC4 stands for Alleged Ron's Code 4. CAST-128 stands for Carlisle Adams and Stafford Tavares.

1.4 Problem Statement:

It is known that security in the cloud computing is a major issue. Many organizations don't accept to put their data in the cloud and they don't want to be controlled by cloud service providers. Many techniques are used to secure data in the cloud such as username password, isolation, permissions, and encryption algorithm. Encrypted data is a challenge for cloud computing researchers. Several encrypted algorithms exist. This research discusses the performance of these algorithms on hypervisors. Plus investigating the performance of CPU utilization and response time in the cloud hypervisors for several encryption algorithms. Both, type one (Para-Virtualization) and type two (Full-Virtualization) of hypervisors were investigated in particular KVM and Xen.

Encryption algorithms differ in their parameters such as complexity, key size, mathematical model, etc. In this research, many algorithms with different parameters such as key, data size, and number of core were deployed to study their effect on hypervisors types with regards of their performance.

1.5 Research Questions

This research aims to answer the following questions:

1. How to measure the effect of the hypervisors types on the CPU utilization and response time in the case of using encrypted data?
2. How to select the hypervisor that suite certain encryption algorithm?
3. How to classify encryption algorithm on the hypervisor types with regards to performance (CPU and time)?

1.6 Objectives

In this research we built a cloud environment using KVM and Xen as open source cloud providers, and assigning leading hypervisors in the domain of cloud computing to the proposed cloud, in order to run an instance on it, then to deploy some encryption algorithms code. The main objective of this research is to evaluate performance of CPU utilization and response time when executing Enc\Dec data on different platforms.

Additionally, numbers of issues were addressed, such as:

- Implementation and evaluation the proposed Approach.
- Make an approach for the algorithms with hypervisors among to the performance.

1.7 Methodology

The main idea in this research is to compare the performance of the two hypervisors type Xen and KVM, where the performance takes into consideration both factors the CPU utilization measured by percentage (%) and response time measured by minutes.

We established an approach to measure the results based on two real cloud servers for both Xen and KVM. After creating two instances and set up the eight EA, we measured the performance for encryption and decryption data. Least response time and minimum CPU utilization is preferred. Different ways were used to compare the results, such as graphs, average, standard deviation, and correlation.

CHAPTER TWO

Background and Literature Review

2.1 Introduction:

This chapter is divided into four parts. Section 2.2 provides the background. Section 2.3 discusses the literature review. The most related studies in the field of comparison between Xen and KVM are discussed in section 2.4.

2.2 Background

As per the problem statement; encryption algorithms' (EA) performance measurement is required. As a result, EA have to be chosen accordingly in order to be a representative sample for most EA categories.

Selected algorithms are considered to be compatible, with different kinds of EA such as: symmetric (CAST128, DES, 3DES), Asymmetric (RSA), small key, big key, block cipher (DES, CAST, BLOWFISH), stream cipher (ARC4), and public key cryptography (RSA).

2.2.1 RSA

RSA is the most well-known EA that is described as a secure, high quality, and public key algorithm. It stands to Ron Rivest, Adi Shamir, and Leonard Adleman. RSA requires two different keys; one for encryption and the other for decryption, so RSA is an Asymmetric EA. RSA has three steps: key generation, encryption, and decryption.

RSA Key Generator (Prasanthi et.al 2012):

1. Choose two large primes p and q .
2. Compute $n = p * q$.
3. Calculate $\phi(n) = \phi(p)\phi(q) = (p - 1)(q - 1)$.
4. Choose an integer e such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$
5. Determine d as $d \equiv e^{-1} \pmod{\phi(n)}$ as private key.

Encryption\Decryption:

1. $C = M^e \pmod{n}$ for encryption.
2. $M = C^d \pmod{n}$ for decryption.

2.2.2 AES (Rijndael)

AES stands for the Advanced Encryption Standard and it is defined as the specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001. AES is a kind of symmetric EA, that its encryption key and decryption key are both the same. AES is based on Rijndael cipher. Rijndael is a family of ciphers with different key and block sizes. For AES, NIST selected three members of the Rijndael family, each with a block size of 128 bits, but three different key lengths: 128, 192 and 256 bits (Zhiyi et.al 2013).

AES provides plaintext length that should be 128bit; its key length has three optional values: the first value is 128bit, the second value is 192bit, and the third value is 256bit. According to the secret key length, AES algorithm completed N_r iterations. The relationship of number (N_r) times and key length is 10 N_r times for 128 bit key, 12 N_r times for 192 bit key, and 14 N_r times for 256 key. Figure 2.1 shows the structure of the AES (Zhiyi et.al 2013).

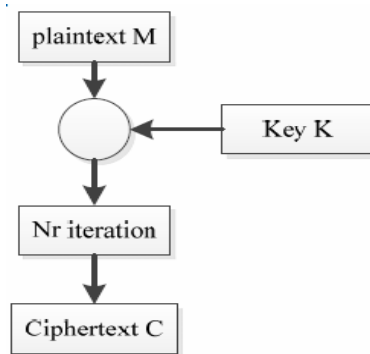


Figure 2.1: AES structure (Zhiyi et.al 2013).

3.2.3 DES

The Data Encryption Standard (DES) was developed in the early 1970s at IBM and based on an earlier design by Horst Feistel. DES is a block cipher. It encrypts data in blocks of size 64 bits each. That is 64-bits of plain text go as input to DES, which produces 64 bits of cipher text. The same algorithm and key are used for encryption and decryption, with minor differences. The key length of this algorithm is 56-bits; however a 64-bits key is actually input, DES is therefore a symmetric key algorithm.

Feistel functions that DES builds in consist of four stages (Rachna et.al 2013):

1. Expansion.
2. Key mixing.
3. Substitution
4. Permutation

3.2.4 Triple DES

Triple DES or TDES or 3DES or Triple Data Encryption Algorithm (TDEA or Triple DEA) is a symmetric-key block cipher, which applies the Data Encryption Standard (DES) cipher algorithm three times to each data block. A lot of cryptographers have argued that the security of DES would be endangered in present days due to its short key length. In order to overcome this problem, cryptographers use the DES algorithm three times so as to expand the key size.

Triple DES (3DES) has been adopted as a temporary standard and is incorporated in several international standards. Most 3DES implementations use two security keys. If the total length of the two keys has 112 bits, then cryptanalysis requires triple computational efforts compared to DES with 56-bit key length. The resultant 3DES cipher text is much harder to break (Jeon s. et.al 2013).

Triple DES uses comprises of three DES keys, K1, K2 and K3, each of 56 bits. (Gunasundari et.al 2014).

The encryption algorithm is:

$$\text{ciphertext} = \text{EK}_3(\text{DK}_2(\text{EK}_1(\text{plaintext}))).$$

Decryption is the reverse:

$$\text{plaintext} = \text{DK}_1(\text{EK}_2(\text{DK}_3(\text{ciphertext}))).$$

3.2.5 ARC4

ARCFOUR or ARC4 (meaning alleged RC4), RC4 was designed by Ron Rivest of RSA Security in 1987 as a stream cipher and a symmetric EA, while it is officially named "Rivest Cipher 4", RC4 was initially a trade secret, but in September 1994 a description of it was anonymously posted to the Cypherpunks mailing list. It was soon posted on the sci.crypt newsgroup, and from there to many sites on the Internet. RC4 has become part of some commonly used encryption protocols and standards, ARC4 is the most widely used software stream cipher and is used in popular protocols such as Transport Layer Security (TLS) to protect Internet traffic to secure wireless networks.

3.2.6 CAST-128

Designed by Carlisle Adams and Stafford Tavares, first published 1996, CAST-128 is a 12- or 16-round Feistel network with a 64-bit block size and a key size of between 40 to 128 bits (but only in 8-bit increments). The full 16 rounds are used when the key size is longer than 80 bits. Another member of the CAST family of ciphers, CAST-256 (a former AES candidate) was derived from CAST-128. CAST-128 is using as the default cipher in some versions of GPG and PGP. It has also been approved for Canadian government use by the Communications Security Establishment (Alam M. 2013).

3.2.7 BLOWFISH

Blowfish is a symmetric key cryptographic algorithm, designed in 1993 by Bruce Schneier and was included in a large number of cipher suites and encryption products. Blowfish encrypts 64-bit blocks with a variable length key of 128-448 bits.

Blowfish suits applications where the key remains constant for a long time (e.g. Communications link encryption), but not where the key changes frequently (e.g. Packet Switching).

According to Schneier, Blowfish was designed with the followings objectives:

- Fast- Blowfish rate on 32-bit microprocessors is 26 clock cycles per byte.
- Compact- Blowfish can execute in less than 5-kb memory. Simple-Blowfish uses only primitive operation –s, such as addition, XOR and table look up, making its design and implementation simple.
- Secure Blowfish has a variable key length up to maximum of 448-bit long, making it both secure and flexible (Arora et.al 2013).

3.2.8 TwoFish

TwoFish is a symmetric key and block cipher algorithm, two fish has been chosen from NIST as one of five algorithms standard with it, Twofish was designed by Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson. Two fish has a block size of 128-bits and key sizes up to 256-bits. Two Fish is an open source, it is one of ciphers that is included in Open PGP.

Two fish's distinctive features are the use of pre-computed key-dependent S-boxes, and a relatively complex key schedule. One half of an n-bit key is used as the actual encryption key and the other half of the n-bit key is used to modify the encryption algorithm (key-dependent S-boxes). Twofish borrows some elements from other designs; for example, the pseudo-Hadamard transform (PHT) from the SAFER family of ciphers (Schnier et.al 2000).

2.3 Literature Review

- Curran provided an overview of the key aspects of Cloud Computing which has five key attributes which grant it some advantages over similar technologies and these attributes include: 1- Multi-tenancy (shared resources): Unlike previous computing models, which assumed dedicated resources dedicated to a single user or owner, cloud computing is based on a business model in which resources are shared at the network, host and application level. 2- Massive scalability: Cloud computing provides the ability to scale to tens of thousands of systems, as well as the ability to massively scale bandwidth and storage space. 3- Elasticity: Users can rapidly increase and decrease their computing resources as needed, as well as release resources for other uses when they are no longer required. 4- Pay as you go: Users pay for only the resources they actually use and for only the time they require them. 5- Self-provisioning of resources: Users self-provision resources, such as additional systems (processing capability, software & storage) and network resources (**Curran K., 2011**).
- Prasanthi and Reddy presented the architecture of the RSA algorithm; they support multiple lengths like 128 bits, 256 bits, 512 bits of data. In this paper simple shift and add algorithm is used to implement the modular multiplication; Which makes the processing time faster and used comparatively smaller amount of space due to its reusability (**Prasanthi O. et al.2012**).
- Fang presented an introduction about AES algorithm, and then the importance of the security in cloud storage system, they proposed approach for security mechanism of users' files when uploading and downloading using AES algorithm (**Fang Z. et al.2013**).

- Arora think that encryption algorithms play an important role in data security on cloud, so they build a Java model consist of four EA. Which AES, DES, Blowfish and RSA algorithms to find the best one security algorithm, they compare it with different security parameters (**Arora R. et al.2013**).
- Jeon propose a novel optical implementation of a 3DES algorithm based on dual XOR logic operations for a cryptographic system. In the schematic architecture, the optical 3DES system consists of dual XOR logic operations, where XOR logic operation is implemented by using a free-space interconnected optical logic gate method (**Jeon S. et al.2013**).
- Gunasudari presented comparison between four symmetric encryption algorithms RC2, RC4, RC5 and RC6. This study concern with security comparison. They found that RC6 is less vulnerable to hacked (**Gunaseundari T. et al.2014**).
- Alam presented comparative study of different encryption algorithms kind; Asymmetric and symmetric, block cipher and stream cipher. These EA are AES, IDEA, DES, RC4 and RSA. At conclusion he found from the experimental results, that RSA has least performance efficiency as compared to DES, AES, IDEA and RC4 algorithm. Also conclude that performance of RC4 algorithm is best as compared to all other algorithms discussed in this paper (**Alam M. 2013**).

2.4 Related Works

- Chierici presented quantitative comparison between Xen and KVM, they measure the classic parameters of a machine (CPU, network and disk access), and totally KVM is better than Xen (**Chierici T. et.al 2010**).
- Younge presented analysis of virtualization technologies for high performance computing environments. They provides an in-depth analysis of some of commonly

accepted virtualization technologies from feature comparison to performance analysis, focusing on the applicability to High Performance Computing environments using Future Grid resources, the result was “KVM hyper visor is the optimal choice for supporting HPC applications within a Cloud infrastructure” (**Younge, A. et.al 2011**).

- Kazan made comparison between Full and Para Virtualization among the Xen Hypervisor, they proved a theoretical state “PV delivers higher performance than full virtualization because the operating system and hypervisor work together more efficiently, without the overhead imposed by the emulation of the system's resources” by using the experimental tests with result to achieve this state (**Kazan F. et al.2013**).
- Hwang made comparison between four popular virtualization platforms, Hyper-V, KVM, vSphere and Xen. They use many bench mark tools like Bytemark, Ramspeed, Bonnie++ & FileBench, Netperf, Application Workloads, and Multi-Tenant Interference, they found that "there is no perfect hypervisor that is always the best choice; different applications will benefit from different hypervisors depending on their performance needs and the precise features they require" (**Hwang J. et al.2013**).
- Xu tried to measure the performance of virtual machines running in the cloud from isolation and scalability point of view. They proposed test comparison using benchmark such as CPU, memory, and disk intensive. This study focus on comparing leading hypervisors such as KVM, Xen, and VMware (**Xu .X et al. 2008**).
- Kolhe tried to make a comparative analysis of KVM and Xen depending on various benchmarking tools. They studied concentrated on measuring CPU performance, network speed, and disk access using a secure shell connection (SSH), and applying benchmark tools for finding results (**Kolhe S. et al. 2012**).

- Schlosser proposed a novel study to find how isolation techniques have impacts on the performance of guest systems. They studied how hypervisors used in cloud computing such as KVM, Xen, and VirtualBox may affect network throughput. In more details, they worked on defining the size of packets in the network and measuring virtual machines CPU and memory utilizations, which will reflect the performance of virtual machines in the network (**Schlosser D. et al. 2011**).
- Yang proposed a way to build KVM environment in the cloud systems and operation. This study focus on building environment with respect to reduce the complexity of cloud resources access. They proposed an experiment to measure the performance of physical machine in order to calculate machine built time, start time, and computing performance. They used CPU utilization, disk usage, and memory utilization (**Yang C. et al. 2011**).

CHAPTER THREE

Experiment Design

3.1 Introduction:

Several researches have been conducted in security; where it is highly considered to be a major issue in cloud. This research takes into consideration encryption algorithms to ensure security and proposes approach that describes the methods used in extracting the results. It describes how the experiment has been implemented. The proposed approach describe the way how will we answer the questions as mentioned at chapter one.

3.2 Setting the Environment:

In order to run the experiment, we set our cloud environment which consisted of two servers; one that is used for KVM and the other for Xen. Debian GNU/Linux was installed for each as operating system; OpenStack Nova was installed inside the operating system to convert the virtual environment into a cloud virtual environment.

Servers' Description

Our cloud environment was built on two different servers; these servers have certain specifications that are listed below:

- Each server has Debian GNU/Linux as operating system.
- An OpenStack was used as a tool for building and managing cloud computing platforms in each server.

- The Proxmox was used as web Virtual Machine Manager (VMM) on KVM server.
- KVM with QEMU was used to prepare the KVM hypervisor, XenServer is based on the XenProject hypervisor.
- We connect the two servers with ProxMox VE so as to manage them together.

Xen boots from a bootloader of Debian GNU/Linux, and then usually loads a para-virtualized host operating system into the host domain (dom0) layer zero.

For each server, a single virtual machine (VM) was created.

Virtual Machine Specification

- The virtual machine has Windows7 as operating system.
- VM hardware specification has 4GB RAM.
- Four cores with ability to change into 2 cores, and 80GB HDD.

As a result, there are two typical windows7 virtual machines; one for Xen hypervisor and other for KVM hypervisor, each contains Visual Studio 10 to access the encryption algorithms then to start it.

The data that is chosen in this research to Encrypt\Decrypt varies in size and kind. Some EA encrypts big data (e.g. 5GB) within few minutes (e.g. ARC4), while other takes long time for small data (10-KB). At RSA, the data that is used from 1KB up to 660KB by changing the key size, rest of EA encrypt\decrypt data of size 800-MB up to 5-GB.

Data type at this thesis was not taken as parameter, although at Enc\Dec process we use different data type such as text, picture, compress, and PDF files.

Multi-Core in CPU is multi-processors or "execution cores" in the same integrated circuit. Each processor has its own cache and controller, which enables it to function as efficiently as a single processor. Thread in computer allows the program to execute sequential actions or many actions at once. Each thread in a program identifies a process that runs when the program asks it to.

As mentioned in the problem statement, this research compares Xen and KVM performance. The response time for encrypt\decrypt and the CPU utilization is the main parameter. Furthermore, least response time and CPU is preferred, while the core and RAM is changed from 2 cores into 4 cores to measure the EA response with time and CPU. RSA has ability to change the thread from 1 into 2 threads.

In RSA, the application we apply has ability to change the number of threads, while for other EA we can't so we change the number of core.

The data size as mentioned before, changes with different EA. So the Parameters for the research were:

- Hypervisor.
- Encryption Algorithm.
- EA key.
- Number of cores.
- Data size.

3.3 Extract Data

In this research, we focused on performance so we took into consideration response time and CPU utilization. Two ways were used to calculate the task duration (response time). RSA algorithm has an application within itself that calculates the duration time for the Encrypt\Decrypt process; while digital stop watch calculates the duration for other EA process.

The CPU utilization measured with the ProxMox CPU usage monitor; at ProxMox average CPU easily can be measured within certain time.

Time is measured by minutes (m) and seconds (s), while the CPU is measured by percentage (%).

We illustrate the extract data for each EA in a table, prepared to put CPU and time.

Table elements are:

- Hypervisor type, which it's either KVM or Xen.
- EA key.
- Number of cores (CPU).
- File Size.
- Duration of encryption task, measured by minutes and seconds.
- CPU utilization during the Enc\Dec task.
- Duration of decryption task, measured by minutes and seconds.
- In RSA data table, a number of Thread column was added.

3.4 Proposed Approach:

The proposed approach makes the user be able to control the two instances. User chooses the EA for both VMs, and then do encryption and decryption on a specific data. Performance results record for response time (Time Duration) and CPU utilization.

User changes many parameters for every Enc\Dec experiment, these parameters are changing of: Hypervisor, VM's core number, encryption algorithm, EA Key and data size, see Figure 3.1.

So, the proposed approach consists of five components:

- 1) Setting the cloud environment with Xen and KVM virtual machines.
- 2) Choosing EA.
- 3) Encrypt\Decrypt data.
- 4) Measure the performance.
- 5) Compare the performance of Xen and KVM.

For example, we choose Xen hypervisor, then using RSA with key size of 2048 bit and one thread, to encrypt data with 12.0 Kb size, we record the time duration in minutes and CPU utilization in percentage %.

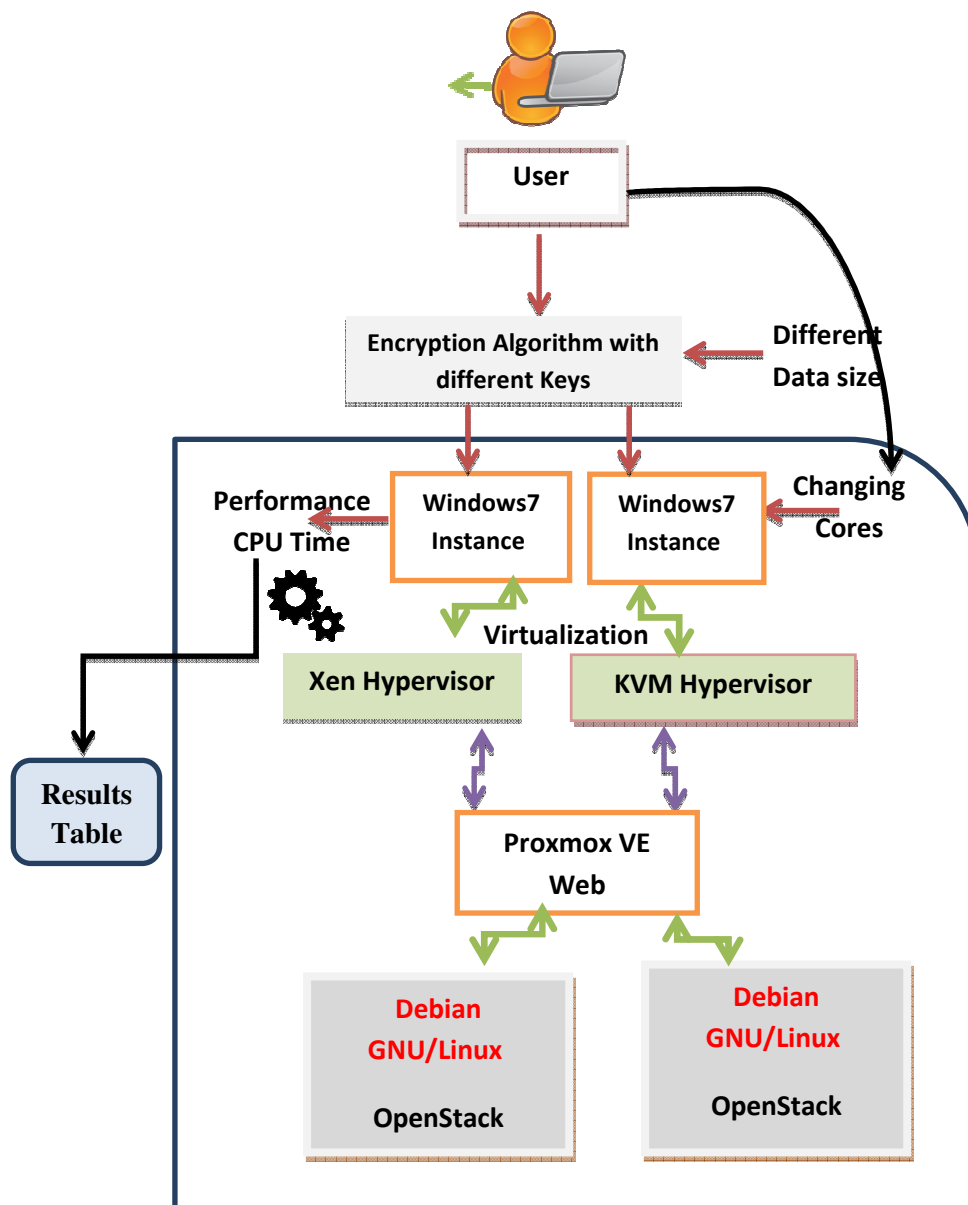


Figure 3.1: The proposed Approach

3.5 Results Analysis

To analyze extracting data and to find relationships between the different parameters like Encrypt\Decrypt, KVM and Xen, we use the standard deviation STDEV, average, and correlation for every EA in addition to tables and diagrams.

For every EA, these five questions were answered to evaluate the performance:

1. What is the **average** of response time in minute and seconds (m: s) and CPU utilization (percentage %) for every hypervisor?
2. What is the **correlation** for the EA?
3. What is the effect of number CPU **cores** on the performance?
4. What is the effect of the **key** on the performance?
5. What is the effect of the **Data size** on the performance?

CHAPTER FOUR

Results Analysis

4.1 Overview:

This chapter discusses and analyses the results related to each EA. This analysis takes into consideration the standard deviation, correlation and average in order to answer the five criteria's questions and compare the performance of Xen with KVM. We also presented and explained final tale results for all EA.

4.2 Encryption Algorithms Results and Analysis

In this section, Every EA results have been discussed to answer the five questions of the criteria and summarize the results and analysis for each EA.

4.2.1 RSA

This Algorithm as mentioned earlier is asymmetric EA, used for public key cryptography. Table 4.1 shows the RSA experiment results.

Table 4.1: RSA experiments result

CPU Dec %	Time Decr m.s	CPU Enc %	Time Encr m.s	File Type	File size	Threads	CPU Core No	Key Size	Hyp Type	Exp No
61	51.01	61	40.25	Pic	606k	1	2 core	64	Xen	1
60	29.22	59	26.23	Pic	606k	1	2 core	64	KVM	
54	23.10	56	14.45	Text	12.2k	1	2 core	1024	Xen	2
53	21.47	54	14.40	Text	12.2k	1	2 core	1024	KVM	
55	10.27	54	5.38	Text	1.0 k	1	2 core	2048	Xen	3
53	10.46	53	4.59	Text	1.0 k	1	2 core	2048	KVM	
89	26.37	100	15.48	Text	12.2k	2	2 core	1024	Xen	4
87	9.41	95	9.38	Text	12.2k	2	2 core	1024	KVM	
87	78.20	95	51.52	Pic	2.25M	2	2 core	1024	Xen	5
82	32.20	85	25.01	Pic	2.25M	2	2 core	1024	KVM	
96	7.53	96	4.09	Text	1.0 K	2	2 core	2048	Xen	6
94	7.00	94	3.52	Text	1.0 K	2	2 core	2048	KVM	
93	15.01	94	7.58	Text	2.0 K	2	2 core	2048	Xen	7
93	13.48	94	7.56	Text	2.0 K	2	2 core	2048	KVM	
64	3.13	64	2.32	Text	12.2k	1	2 core	256	Xen	8
55	2.07	61	2.05	Text	12.2k	1	2 core	256	KVM	
62	60.24	64	34.02	Text	12.2k	1	2 core	1536	Xen	9
52	60.02	56	35.48	Text	12.2k	1	2 core	1536	KVM	
56	106.45	52	67.15	Text	12.2k	1	2 core	2048	Xen	10
52	120.12	51	77.20	Text	12.2k	1	2 core	2048	KVM	

Where:

- Exp No: Experiment Number.
- Hyp type: hypervisor type, which it's either KVM or Xen.
- Key Size: EA key size in bit.
- CPU Core No: Number of cores.
- File Size: The size of the file.
- File Type: Kind of file.
- Time Encr m.s: Duration of encryption task, measured by minutes and seconds.
- CPU Utili %: CPU utilization during the Enc\Dec task.
- Time Decr m.s: Duration of decryption task, measured by minutes and seconds.

To have understand for the previous table let's take Exp No 1 as an example; This process when we use Xen hypervisor at RSA with Key of 64bit, 2 cores, and one thread, we encrypt picture of 606 K size, the response time at encryption were 40.25 minutes, and CPU utilization were 61%; during at decryption process, response time were 51.01 minutes and CPU utilization were 61. Same way for KVM.

Next EA experiments results table will be at appendix.

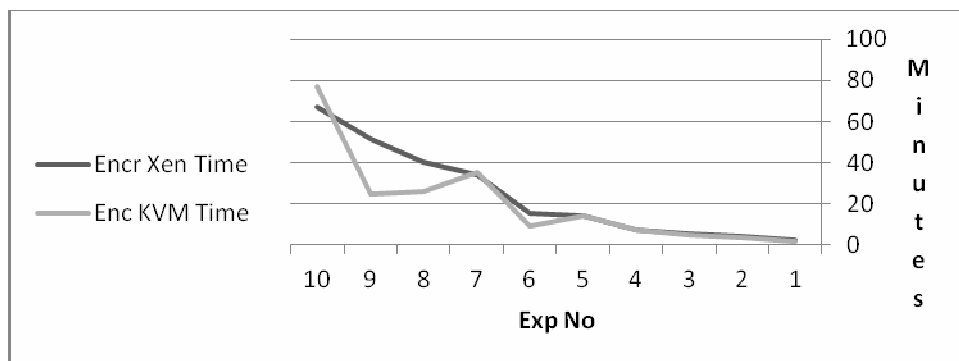
To make these results easy to understnd, for RSA and for all EA we summarise the results of Table 4.1 in a Table 4.2. This table shows the value of results without details of key, core, and data size and kind.

Table 4. 2 summarises the extracted results; where it shows the results of time and CPU on hypervisors with encryption and decryption process with two hypervisor.

Table 4.2: Results of Time and CPU on Hypervisors

RSA	Encr	Enc	Decr	Dec	Encry	Encry	Decr	Decr
Exp No	Xen Time	KVM Time	Xen Time	KVM Time	Xen CPU	KVM CPU	Xen CPU	KVM CPU
1	2.32	2.05	3.13	2.07	64	61	64	55
2	4.09	3.52	7.53	7	96	94	96	94
3	5.38	4.59	10.27	10.46	54	53	55	53
4	7.58	7.56	15.01	13.48	94	94	93	93
5	14.45	14.4	23.1	21.47	56	54	54	53
6	15.48	9.38	26.37	9.41	100	95	89	87
7	34.02	35.48	60.24	60.02	64	56	62	52
8	40.25	26.23	51.01	29.22	61	59	61	60
9	51.52	25.01	78.2	32.2	95	85	87	82
10	67.15	77.2	106.45	120.12	52	51	56	52

Exp No means the experiment number. To have a better understanding for the results, we presented the below diagrams to clarify the results. Figure 4.1 shows the encryption time for both the Xen and the KVM while figure 4.2 shows the decryption time for both the Xen and KVM.

**Figure 4.1: Encryption response time for Xen and KVM**

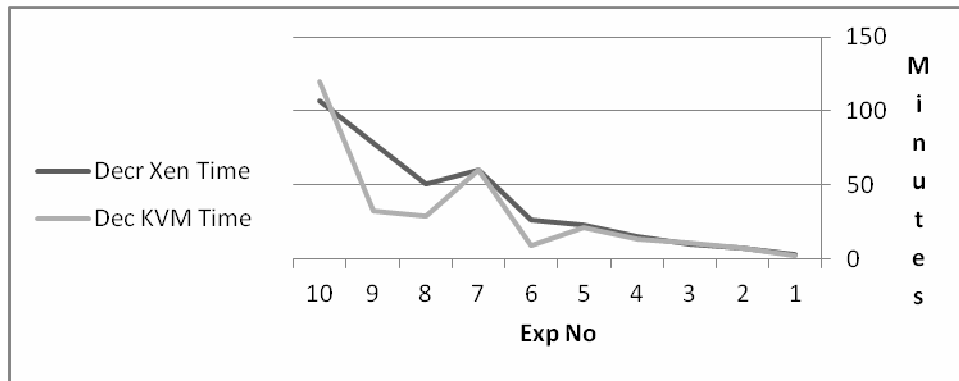


Figure 4.2: Decryption response time for Xen and KVM

Some Enc/Dec results with Xen were better than the results for the KVM but most of the results were better in KVM. At first five results, we found the two hypervisors are very close, but at Exp No 8 and 9 KVM is better than Xen because the data size is very big; while at Exp No 10 we found Xen is better because we use big key and 2 cores and relatively big data.

Figure 4.3 shows the Xen time for encryption and decryption where it is very clear that the time of decryption is bigger than the time for encryption especially for process of long response time.

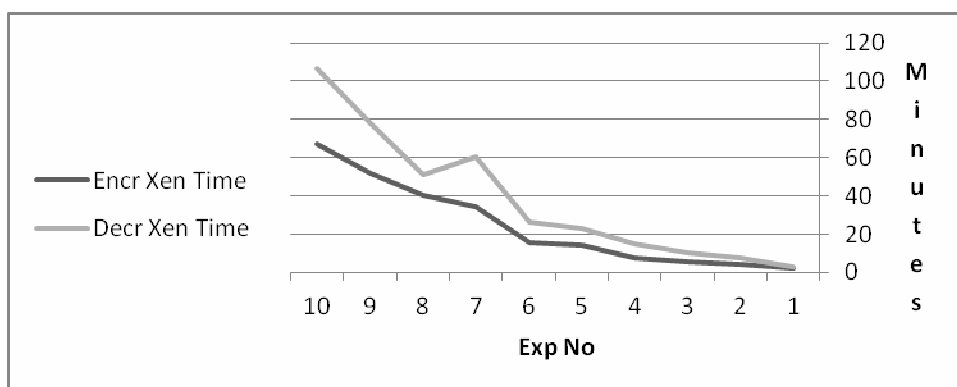


Figure 4.3: Xen response time for encryption and decryption

Figure 4.4 shows the CPU utilization of two hypervisors during encryption and figure 4.5 shows the CPU utilization of two hypervisors during decryption depending on the results from Figure 4.3.

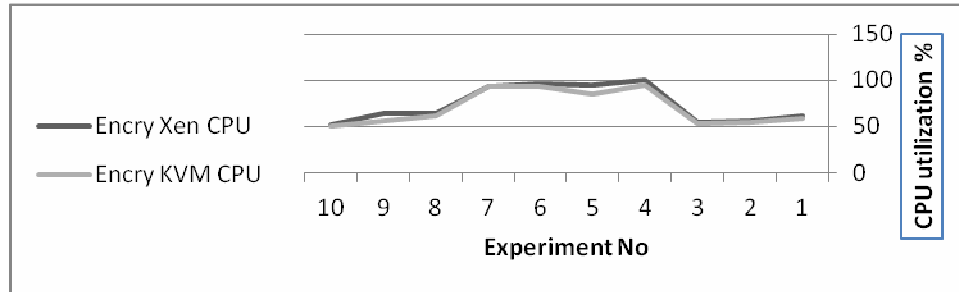


Figure 4.4: CPU utilization of two hypervisors during encryption

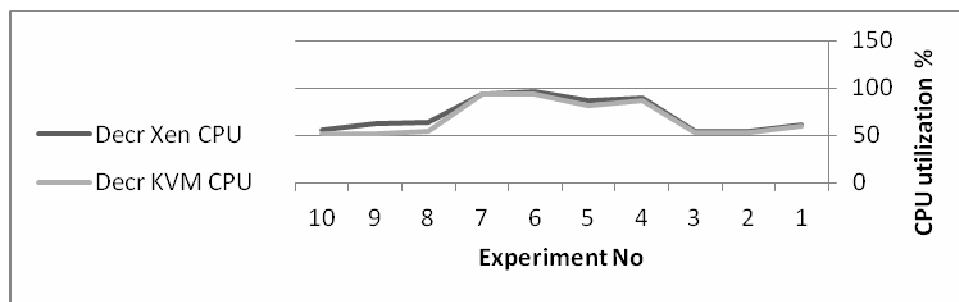


Figure 4.5: CPU utilization of two hypervisors during Decryption.

The last two diagrams for CPU utilization during encryption (Figure 4.4) and decryption (Figure 4.5) clearly show that CPU utilization in KVM is better than Xen at all levels. The high CPU utilization results were 2 threads.

After discussing the results, the below questions have been answered.

1. What is the **average** of time (m:s) and CPU utilization (percentage %) for every hypervisor?

Table 4.3: Average performance for RSA

Decr CPU	Encr CPU	Decr Time	Encr Time	RSA
71.7	73.6	38.13	24.22	Xen
68.1	70.2	30.54	20.54	KVM
3.6	3.4	7.19	3.28	Difference

Encr\Decr Time means the time of encryption\decryption for all experiments, Encr\Decr CPU represents the CPU utilization at encryption\decryption for all experiments; these results were extracted from Table 4.2. For example, Enc Time for Xen is 24.22 means the average time of all encryption response time in minutes at Xen hypervisor. Difference means the difference in time between Xen and KVM; positive value means KVM value less than Xen.

Table 4.3 shows the average time and CPU for RSA and as a result we can conclude that at RSA, KVM is better than Xen at encryption\decryption response time and encryption\decryption CPU utilization, so KVM is preferred.

2. What is the **correlation** for the EA?

Correlation shows the strength of relationship between two set of values. We use correlation at this research to show the relationship strength of the (Table 4. 2) results. We show the relationship strength between encryption with Xen and KVM, then encryption and decryption for Xen alone. In same way we found for decryption and KVM respectively. We found correlations for time and CPU.

Table 4.4: Correlation between RSA Results

Enc Dec KVM	Enc Dec Xen	Dec Xen KVM	Enc Xen KVM	correlation
0.99	0.99	0.89	0.90	Time
0.99	0.98	0.98	0.99	CPU

Table 4.4 shows the correlation between the RSA results; where at KVM, encryption and decryption has a strong relationship and this is also shown with Xen, but the relationship at encryption process with KVM and Xen is weak relatively, that's because of the big difference between the results. Some results show differences that reached up to 110%. *Refer to (Table 4.1).*

3. What is the effect of the CPU **cores** number on the performance?

This parameter shows the impact of changing the number of threads from one thread into two on performance. The effect on response time and CPU utilization is needed as well.

A sample was taken to see the changing in the thread from 1 into 2 threads; at this sample we fixed the data size and the key. Figure 4.6 shows, when increasing threads surely response time will decrease at all time, but as for KVM, it shows better impact when decreasing time with both encrypt and decrypt when changing to 2 threads. Figure 4.7 shows the effect of Thread on CPU utilization and clearly shows a big increase when changing the threads from one thread to two threads (e.g. from 53% into 94%).

Figure 4.6 shows the effect of threads on response time. Figure 4.7 shows the effect of threads on CPU utilization.

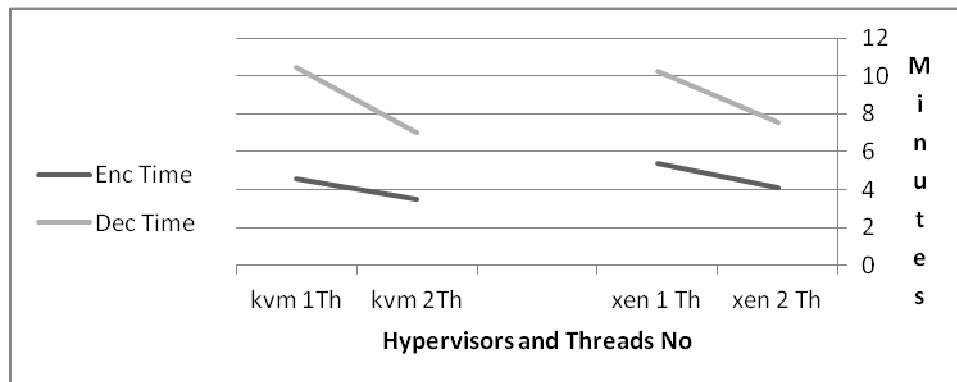


Figure 4.6: Effect of Thread on Response Time

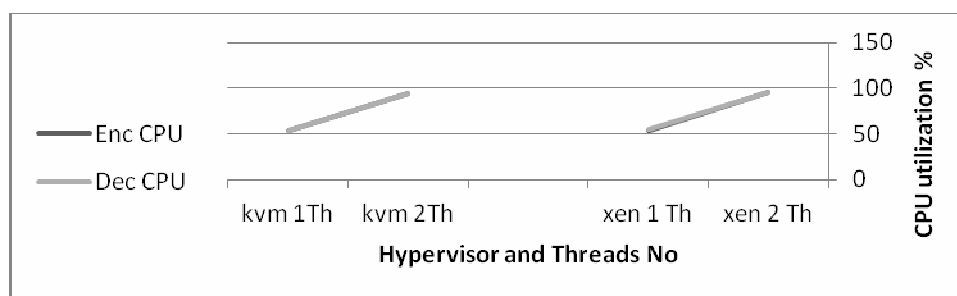


Figure 4.7: Effect of Thread on CPU utilization

4. What is the effect of the **key** on the performance?

This parameter shows the effect of changing the key on the performance (Time and CPU).

At this sample, thread and data size are fixed while the key differs. Results at response time show clearly that KVM are better than Xen at small key, but at big key

Xen is better than KVM. By the induction at big keys KVM and Xen are closer than small keys, but at small keys KVM is better than Xen with big difference.

It's clear that at small key encryption and decryption response time is less than big key (Figure 4.8 and Figure 4.9).

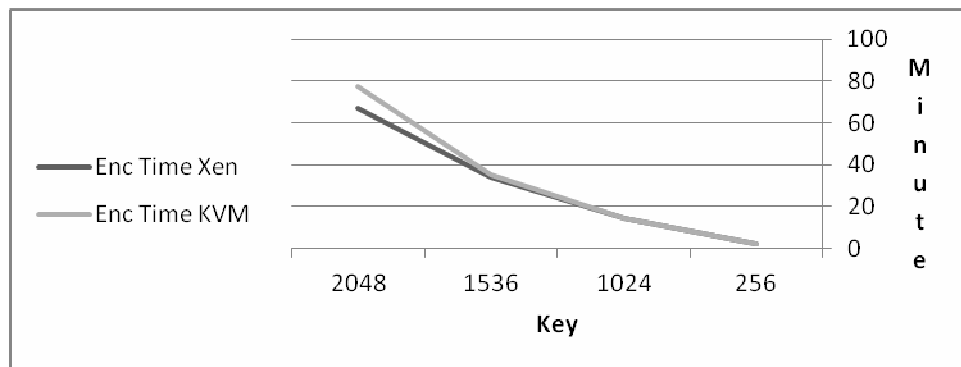


Figure 4.8: Encryption response time for hypervisors when changing keys.

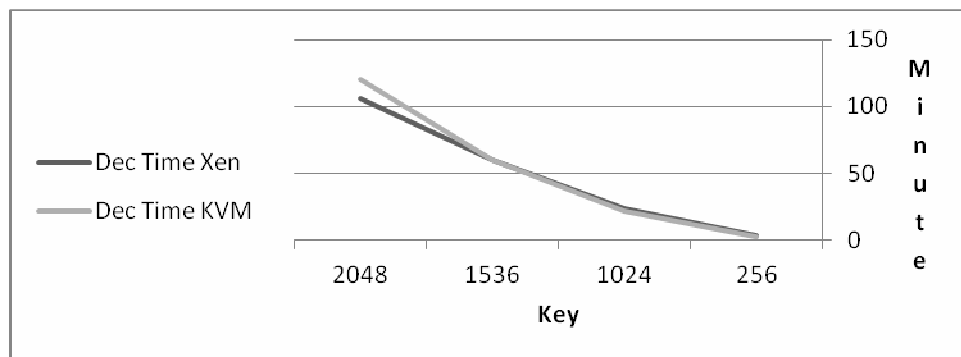


Figure 4.9: Decryption response time for hypervisors when changing keys.

CPU utilization shows that at big keys CPU utilization relatively is better than small key, and KVM is more stable than Xen (Figure 4.10 and 4.11).

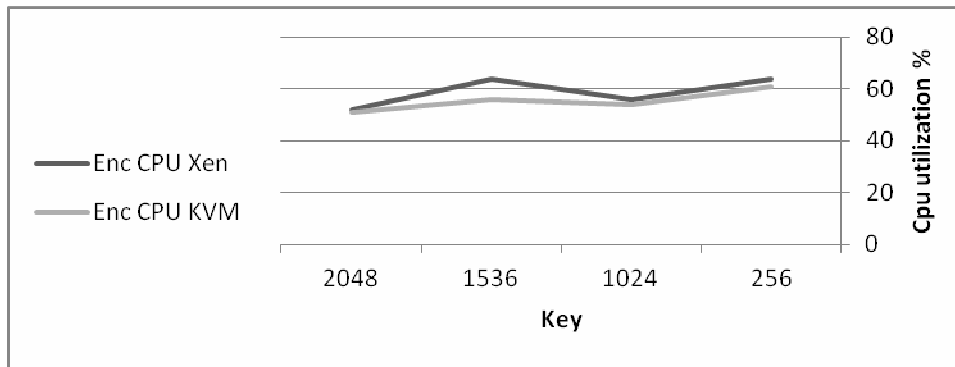


Figure 4.10: Encryption CPU for hypervisors when changing keys.

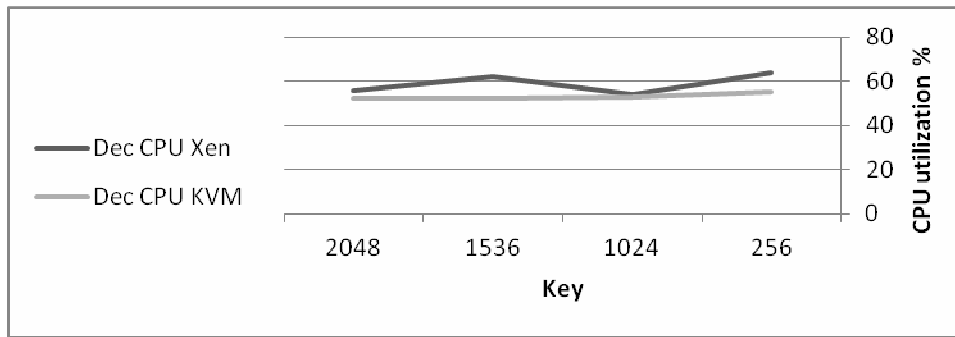


Figure 4.11: Decryption CPU for hypervisors when changing keys.

5. What is the effect of **Data size** on the performance?

This question aimed to study the effect of changing the data size on the response time and CPU utilization.

At this sample, both the thread and the key were fixed but the data size was differing. Results show that if we multiply data size with r times, response time will increase also but with less than r times, refer to figure 4.12. Results show that KVM is more efficient than Xen when increasing data size especially with decryption. CPU utilization shows no regular effect when changing the data size; the most concern was with time.

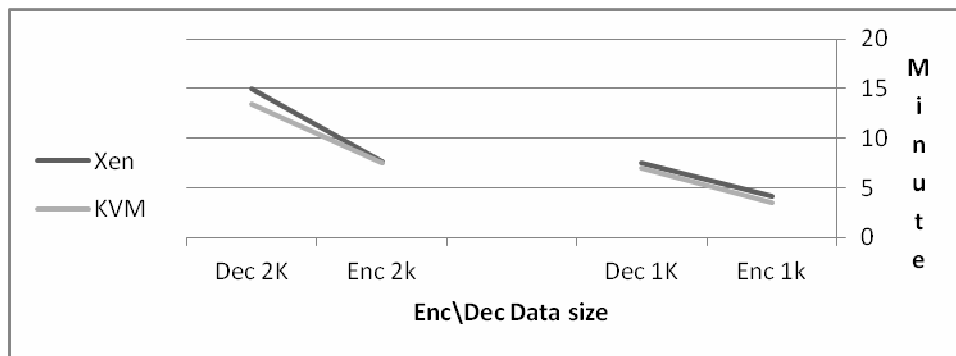


Figure 4.12: Enc\Dec for different data size over Xen and KVM

Finally, RSA performance summary among KVM and Xen shows:

- CPU Utilization in KVM is less than Xen in all results, so KVM is preferred.
- KVM time average is less than Xen.
- In thread increments, KVM is more efficient especially with decrypt.
- With small key, KVM is more efficient with big difference, in big key KVM and Xen are closer.
- KVM by increasing data size is more efficient than Xen especially with decryption.

4.2.2 AES (Rijndael)

This Algorithm as mentioned at background in chapter 3 is a symmetric EA using block cipher, to see the whole experiment table refer to appendix 2.

The results of the experiments are shown in table 4.5 for AES, where the time is in minutes and CPU in percentage % as:

Table 4.5: Result of all AES experiments

AES	Encr	Encr	Decr	Decr	Encr	Encr	Decr	Decr
Exp No	Xen Time	KVM Time	Xen Time	KVM Time	Xen CPU	KVM CPU	Xen CPU	KVM CPU
1	1.06	1.05	1.1	1.02	34	32	32	32
2	2.42	2.35	2.42	2.42	67	60	69	55
3	2.53	2.5	3.17	2.41	37	33	37	34
4	2.57	2.48	2.57	2.56	67	62	71	56
5	3.02	2.57	3.27	2.55	37	34	38	34
6	3.05	2.59	2.5	2.25	34	30	40	34
7	3.25	2.52	3.25	2.51	63	61	67	61
8	3.41	3.4	3.45	3.24	35	30	36	32
9	4.01	3.44	3.41	3.4	64	56	70	59
10	4.22	3.49	4	3.15	70	63	69	64
11	7.2	6.31	7.09	6.13	35	30	38	33

We need to understand the results, diagrams were used to represent them; for encryption response time (Figure 4.13) and decryption response time (Figure 4.14).

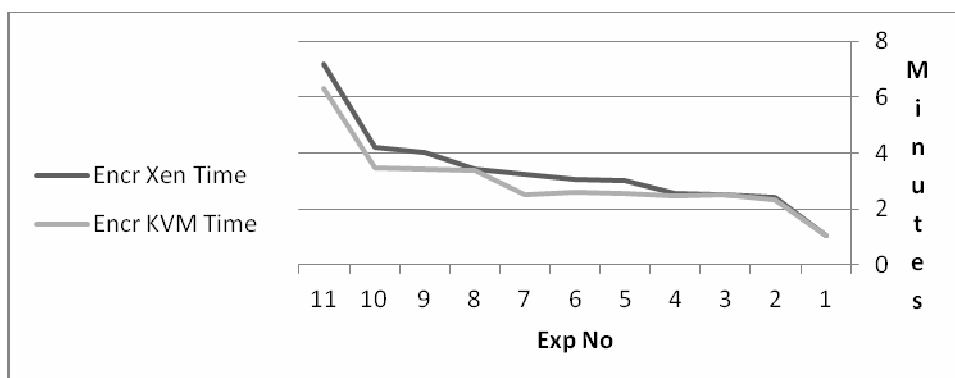


Figure 4.13: Represents encryption response time for Xen and KVM.

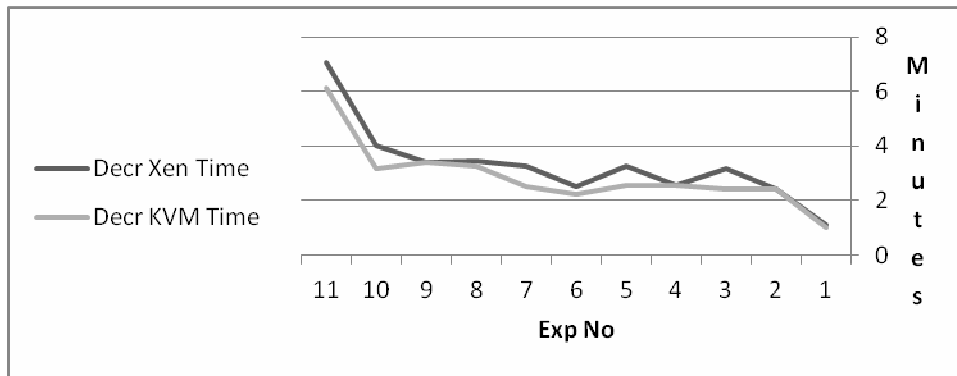


Figure 4.14: Represents decryption response time for Xen and KVM.

It's clear that response time with KVM is less than the response time with Xen at all results; here the CPU utilization results percentage % for the AES. Also the big difference at key 128 bit.

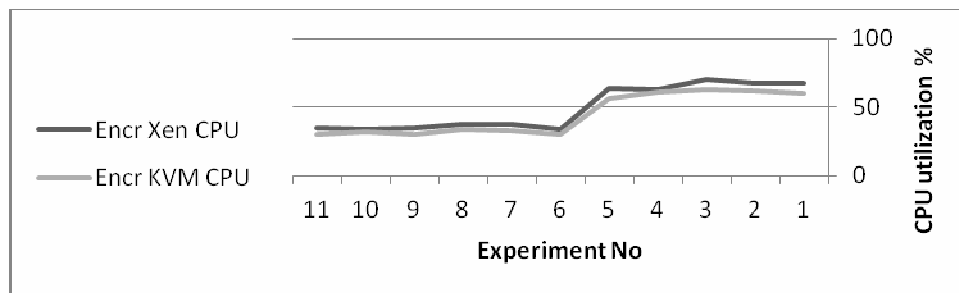


Figure 4.15: The CPU utilization at Encryption over Xen and KVM

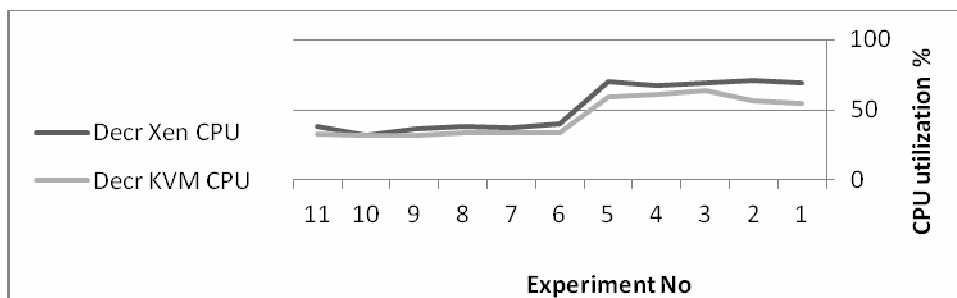


Figure 4.16: The CPU utilization at Decryption over Xen and KVM

Figure 4.15 and figure 4.16 show that CPU utilization in KVM is better than Xen in all results.

The five questions were answered according to the results above.

1. What is the **average** of time (m:s) and CPU utilization (percentage %) for every hypervisor?

Table 4.6: Average performance over AES experiments

Decr CPU	Encr CPU	Decr Time	Enc Time	AES
47.44	45.44	3.47	3.53	Xen
42.56	41.00	2.96	3.10	KVM
4.88	4.44	0.51	0.43	Difference

Table 4.6 shows the average time and CPU over AES experiments, where we can conclude that KVM is better than Xen at Enc\Dec with response time and CPU utilization.

2. What is the **correlation** for the EA?

We measure the relationship strength for the different results value column in (Table 4.5).

Table 4.7: Correlation for AES Results

Enc Dec KVM	Enc Dec Xen	Dec Xen KVM	Enc Xen KVM	correlation
0.99	0.97	0.97	0.99	Time
0.98	0.99	0.98	0.99	CPU

Table 4.2.3 shows strong relationship between all AES components, so AES is being stable EA at Enc\Dec, especially at KVM response time.

3. What is the effect of the CPU **cores** number on the performance?

A sample has been taken that has the same data and key, but different core numbers. Figure 4.17 shows the impact of changing the number of cores on response time and as a result we can say that when increasing the number of cores, the time decreases for both Xen and KVM.

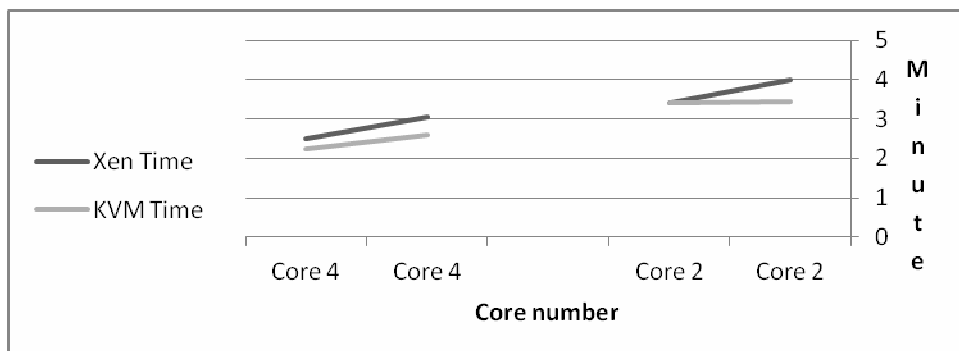
**Figure 4.17: Effect of Changing Core Number on Response Time**

Figure 4.18 shows the effect of changing the number of cores from 2 cores in to 4 cores on the CPU utilization. It shows that the CPU utilization decreases about to half. Also at core 2, KVM CPU utilization is better than Xen to core 4.

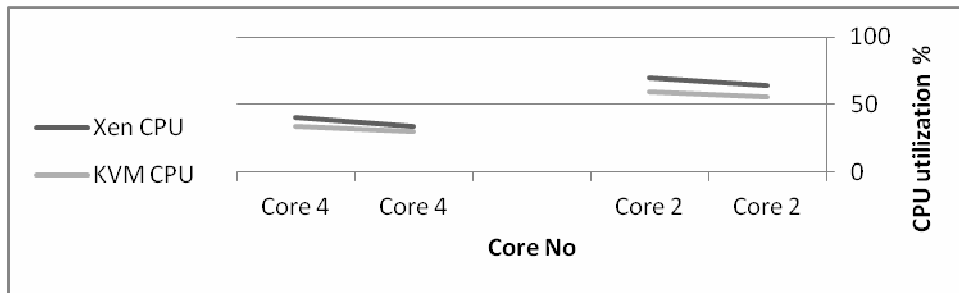


Figure 4.18: Effect of Changing Core Number on CPU Utilization

4. What is the effect of the **key** on the performance?

Using a sample of three different keys 128, 192 and 256 with the same data size and core numbers, the results for response time are illustrated in (Figure 4.19), results for the CPU utilization are illustrated in (Figure 4.20).

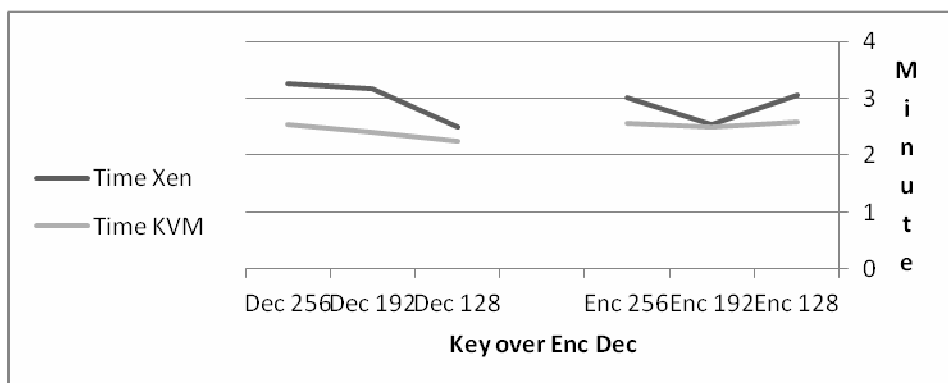


Figure 4.19: Effect of Changing Key on Time

Figure 4.19 shows the effect of changing the key on response time. For encryption, KVM has stable time while Xen is not stable. While at decryption, increasing the key will increase the response time.

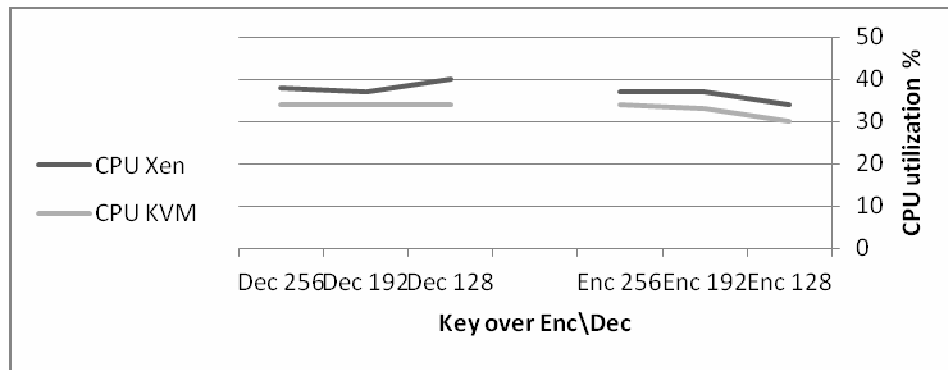


Figure 4.20: Effect of changing key on CPU

Figure 4.20 shows the effect of changing the key on CPU. Within encryption, increasing the key increases the CPU utilization for both KVM and Xen; while at decryption, KVM is stable, and Xen is not stable.

5. What is the effect of the **Data size** on the performance?

Using a sample of three different data sizes 2, 3 and 5 GB with fixed key and core, the results for response time are demonstrated in (Figure 4.21), while the results for CPU utilization are shown in (Figure

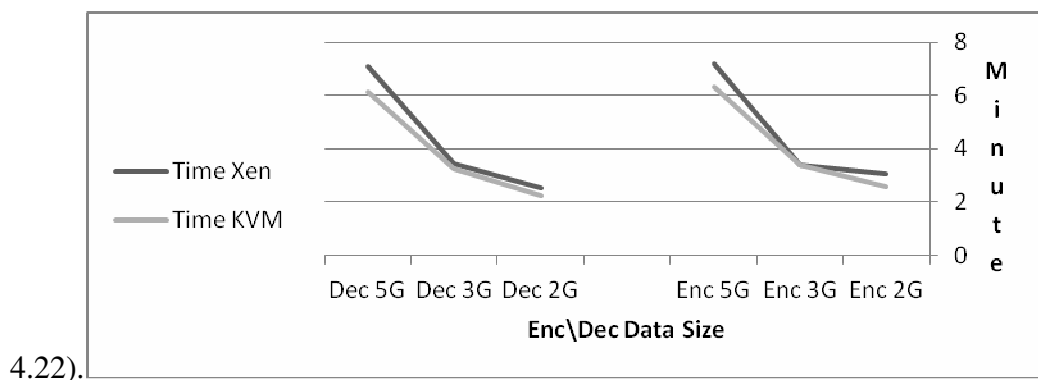


Figure 4.21: Effect of Changing Data Sizes on Response Time

Figure 4.21 shows that when increasing the data size, time will increase in both cases for either encryption or decryption. For the big data, KVM shows better results than Xen.

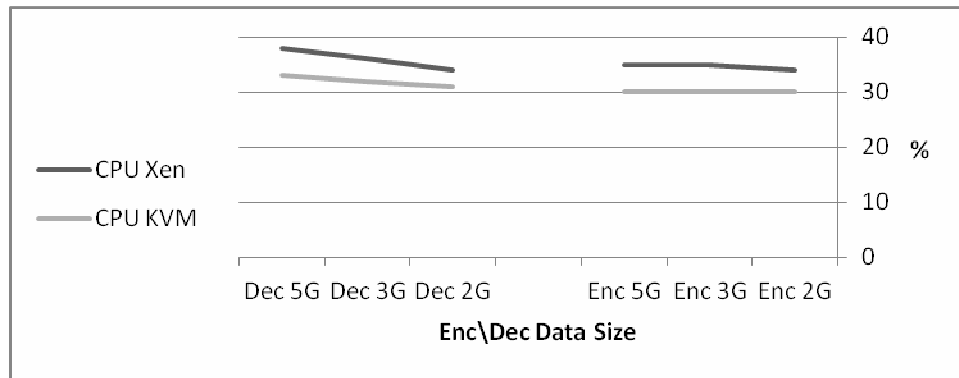


Figure 4.22: Effect of Changing Data Sizes on CPU Utilization

Figure 4.22 shows the effect of changing data sizes on CPU utilization. This figure shows that when increasing the data size, the CPU utilization relatively increases especially for decryption.

Finally, AES performance summary among KVM and Xen shows that:

- It is clear that KVM is better than Xen in all of results with time and CPU utilization.
- KVM is better in case of changing core, data size, and key.
- All results give benefit to KVM in case of average, standard deviation, and correlation.

4.2.3 DES

This Algorithm as mentioned in the background in chapter 3 is a symmetric EA using block cipher. Refer to appendix 3 to see the whole experiment table. The results of the experiments are shown in (Table 4.8). The results that are shown in the table set the time in minutes and the CPU in percentage % as:

Table 4.8: Result of all DES experiments

DES	Encr	Encr	Decr	Decr	Encr	Encr	Decr	Decr
Exp No	Xen Time	KVM Time	Xen Time	KVM Time	Xen CPU	KVM CPU	Xen CPU	KVM CPU
1	2.2	2.15	2.14	2.12	37	32	35	32
2	2.22	2.17	2.2	2.2	64	59	63	61
3	3.55	3.54	3.55	3.53	64	61	63	61
4	3.57	3.57	3.55	3.47	37	31	37	32
5	5.54	5.49	5.51	5.49	37	33	37	33
6	6.16	6.16	6.05	3.55	35	32	37	33
7	6.23	6.2	6.21	6.15	64	61	65	57
8	8.05	7.45	8.11	7.5	68	55	69	55
9	15.24	14.2	14.18	14.14	36	32	35	32
10	16.04	14.14	15.38	13.52	65	62	66	64

Diagrams are used to represent the results and for better understanding. Figure 4.23 represents encryption time for Xen and KVM and figure 4.24 represents decryption response time.

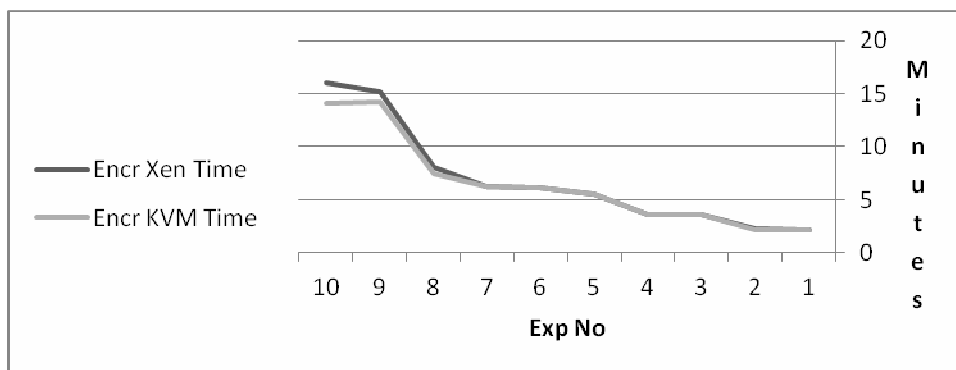


Figure 4.23: Encryption response Time for Xen and KVM.

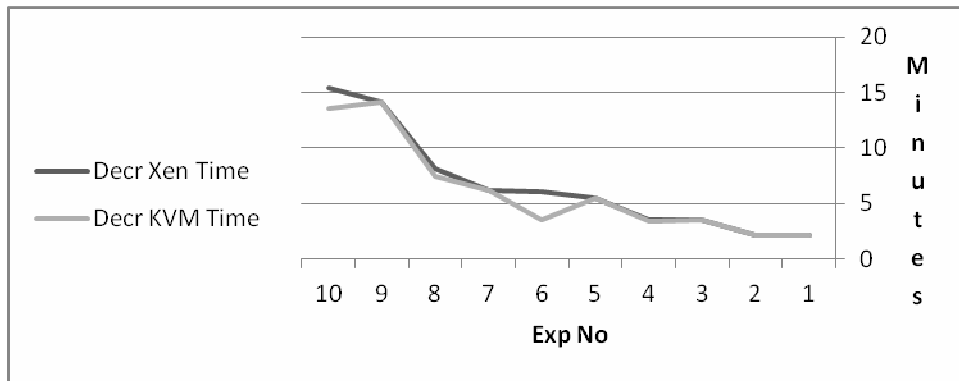


Figure 4.24: Decryption Time for Xen and KVM.

It is clearly shown that response time with KVM is less than Xen at all levels. But at long response time experiment, KVM show better performance than Xen.

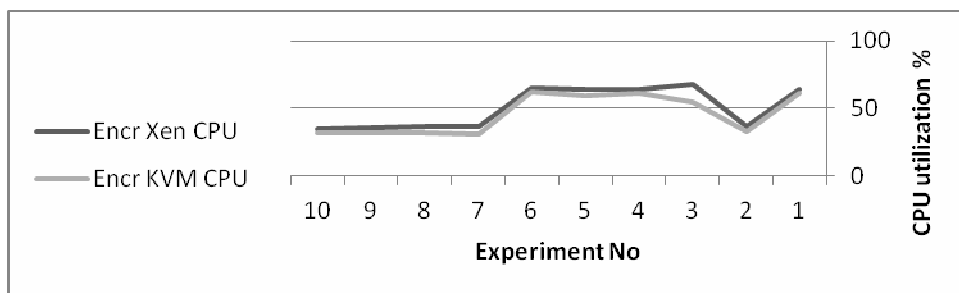


Figure 4.25: The CPU at Encryption over Xen and KVM

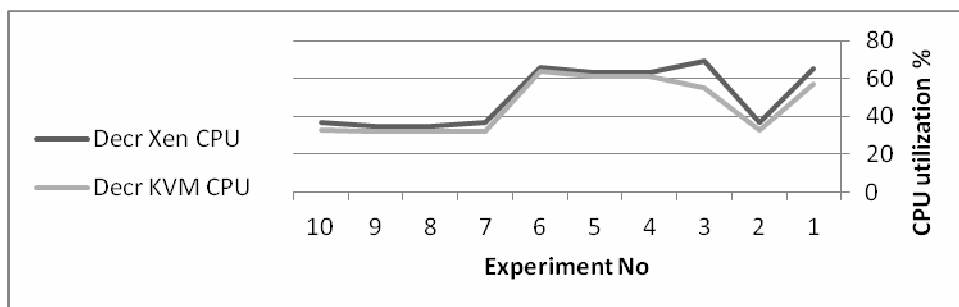


Figure 4.26: The CPU at Decryption over Xen and KVM

Two diagrams (Figure 4.25 and Figure 4.26) clearly show that the CPU utilization in KVM is better than Xen in all the results. At core 4 the CPU utilization about 60%.

Now let us to answer the five questions.

1. What is the **average** of time (m:s) and CPU utilization (percentage %) for every hypervisor?

Table 4.9 Average performance for Xen and KVM

Decr CPU	Encr CPU	Decr Time	Enc Time	DES
49.11	49.22	6.74	6.95	Xen
44.78	44.11	6.17	6.54	KVM
4.33	5.11	0.64	0.41	Difference

Table 4.9 shows that average time at KVM is better than Xen at Enc\Dec with response time and CPU utilization.

2. What is the **correlation** for the EA?

Table 4.10: Correlation between all DES

correlation	Enc Xen KVM	Dec Xen KVM	Enc Dec Xen	Enc Dec KVM
Time	1.00	0.98	1.00	0.98
CPU	0.98	0.97	1.00	0.99

Table 4.10 shows strong relationship between all AES components, so AES is like to be stable EA at Enc\Dec, especially with Xen.

3. What is the effect of the CPU core numbers on the performance?

Sample taken has same data and key, but different at core numbers. Response time (Figure 4.27) shows that at increasing cores, time is decreasing for both Xen and KVM.

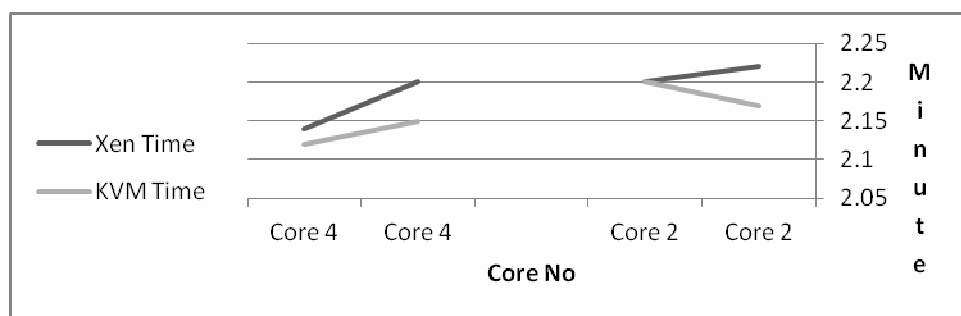


Figure 4.27: Effect of changing core number on time response

Figure 4.28 shows the CPU utilization results after changing the number of cores from 2 cores in to 4 cores. It shows that the CPU utilization decreases about to half, as well as Xen, where it shows that it is more stable than KVM especially at core 2.

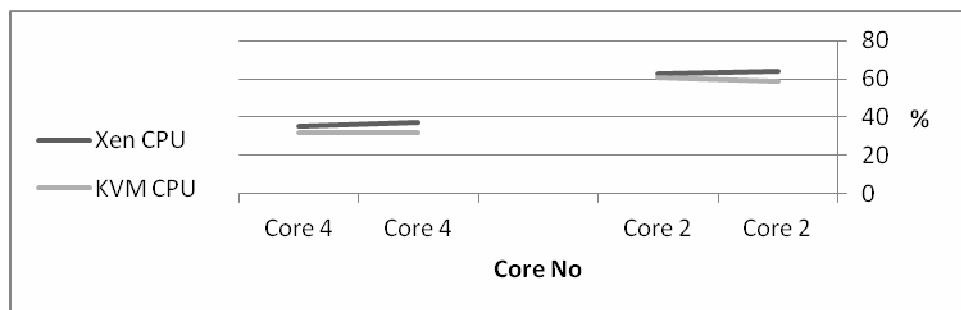


Figure 4.28: Effect of Changing Core Number on CPU Utilization

4. What is the effect of the **key** on the performance?

At DES EA, 64-bit key was only used.

5. What is the effect of the **Data size** on the performance?

Using a sample of three different data sizes 1, 2 and 5 GB with fixed key and core, the results for response time are illustrated in Figure 4.29, where the results for the CPU utilization are presented in figure 4.3.8.

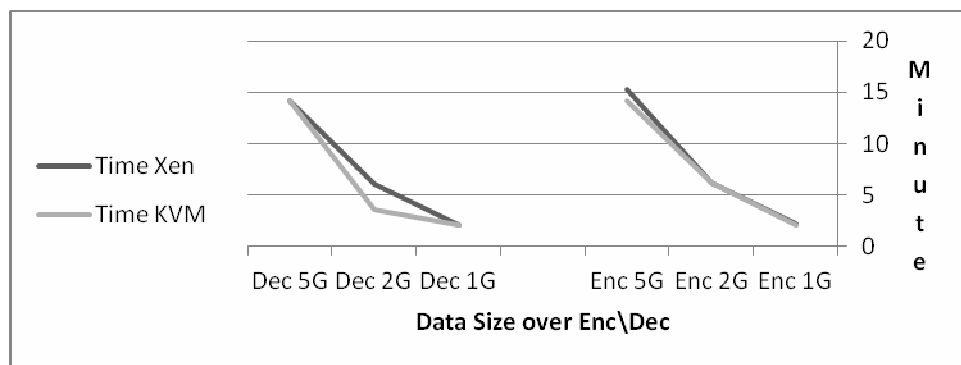


Figure 4.29: Effect of Changing Data Size on Response Time

This figure shows that when increasing the data size; time will increase for both encryption and decryption.

But by looking at figure 4.23 and figure 4.24, we can notice that increasing data size for KVM is better than Xen; while at small data, Xen and KVM are very close.

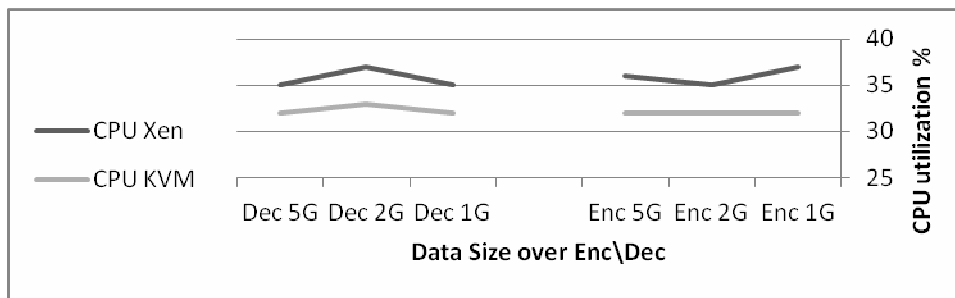


Figure 4.30: Effect of changing data size on CPU utilization

This figure shows that with different data sizes there is no stable behavior, but KVM shows more stability than Xen.

Finally, we can summarize the DES performance on KVM and Xen, as:

- KVM is better than Xen in the average, correlation at response time and CPU utilization.
- KVM is better when changing cores, and data size.
- At small data size, Xen and KVM are very close; while at big data KVM is better than Xen with big difference.

4.2.4 Triple DES

This Algorithm as mentioned in the background in chapter 3; is a symmetric EA using block cipher. Refer to appendix 4 to see the whole experiment table. The results of the experiments are shown in (Table 4.11).

Table 4.11: Result of all DES experiments

3DES	Encr	Encr	Decr	Decr	Encr	Encr	Decr	Decr
Exp No	Xen Time	KVM Time	Xen Time	KVM Time	Xen CPU	KVM CPU	Xen CPU	KVM CPU
1	5.02	5.19	5	5.08	35	30	35	31
2	5.15	5.23	5.11	5.23	62	60	61	60
3	8.35	8.45	8.3	8.45	35	31	35	32
4	8.48	9.08	8.42	9	61	55	62	60
5	13.33	13.57	13.18	13.57	34	32	35	31
6	14.03	14.35	13.47	14.25	62	61	62	61
7	32.09	33	32	33.11	65	61	64	60

Figure 4.31 represents encryption response time and figure 4.32 represents decryption response time.

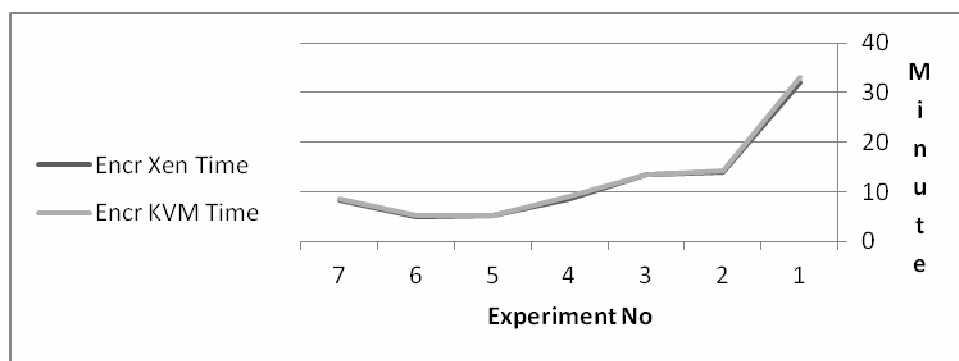


Figure 4.31: Encryption response time for Xen and KVM.

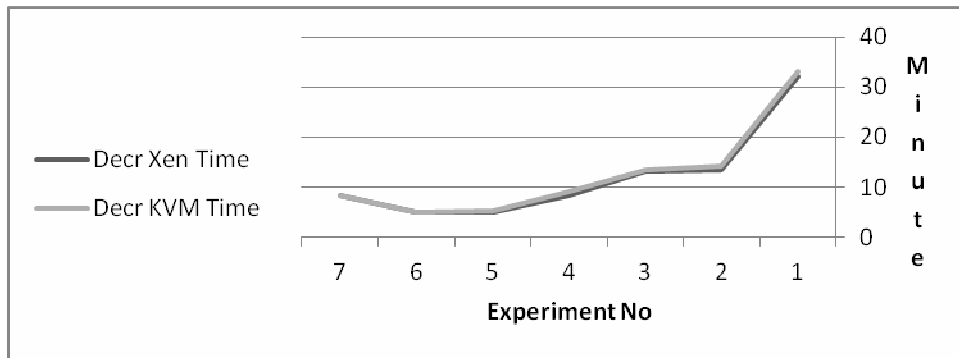


Figure 4.32: Decryption response time for Xen and KVM.

It is clearly shown that the response time with Xen is less than KVM at all results.

KVM is close to Xen at all results.

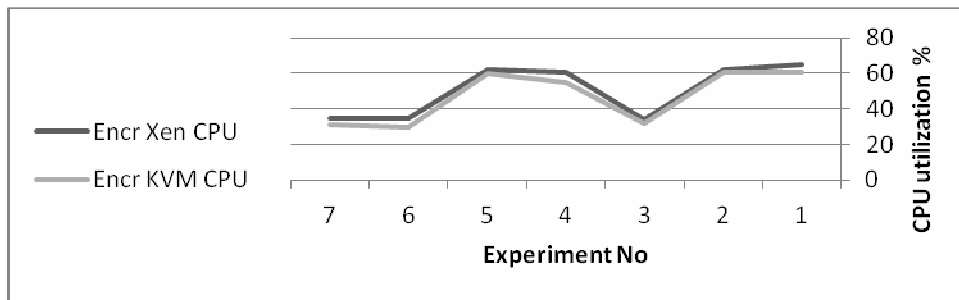


Figure 4.33: Represent the CPU at encryption over Xen and KVM

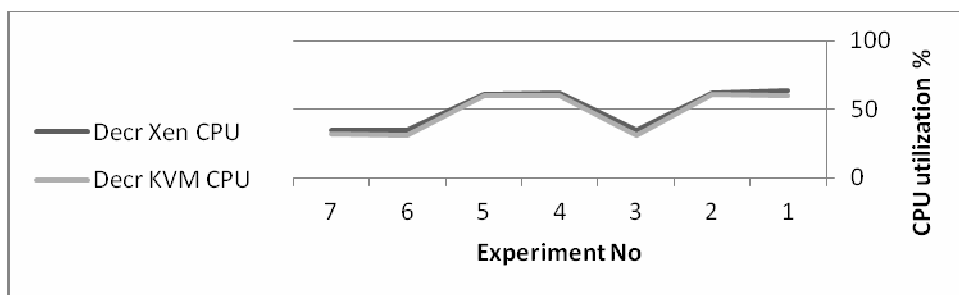


Figure 4.34: Represent the CPU at decryption over Xen and KVM

Figure 4.33 and figure 4.34 show that the CPU utilization for KVM is better than Xen in all the results.

After discussing the results, clear answers will be given to the five main questions.

1. What is the **average** of time (m:s) and CPU utilization (percentage %) for every hypervisor?

Table 4.12 Average performance for Xen and KVM

Decr CPU	Encr CPU	Decr Time	Enc Time	3DES
50.57	50.57	12.21	12.35	Xen
47.86	47.14	12.67	12.70	KVM
2.71	3.43	-0.46	-0.25	Difference

The average time table shows that Xen is better than KVM at Enc\Dec with response time not for the CPU utilization.

Although Triple DES is derived from DES by applying encryption three times, they differ in response time performance; at DES we saw that KVM is better than Xen, but at 3DES; Xen is better in time.

2. What is the **correlation** for the EA?

Table 4.13: Correlation between all 3DES columns

correlation	Enc Xen KVM	Dec Xen KVM	Enc Dec Xen	Enc Dec KVM
Time	1.00	1.00	1.00	1.00
CPU	0.99	1.00	1.00	0.99

It is clear that Xen and KVM have strong relationship at all components, even at encryption or decryption.

3. What is the effect of the CPU core numbers on the performance?

A sample was taken, this sample has the same data and key, but different in core numbers. Figure 4.35 shows the effect on response time when increasing the cores; time decreases for both Xen and KVM.

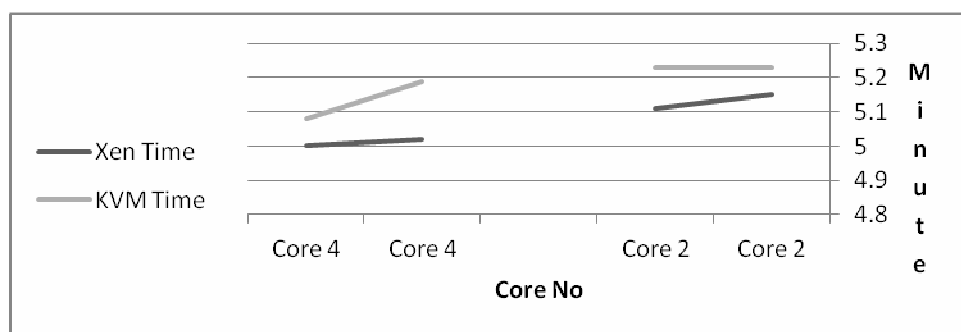


Figure 4.35: Effect of changing core number on response time

Figure 4.36 shows the CPU utilization when changing cores from 2 cores into 4 cores. It shows that the CPU utilization decreases less than half. As well as KVM shows better performance at core 4 than Xen.

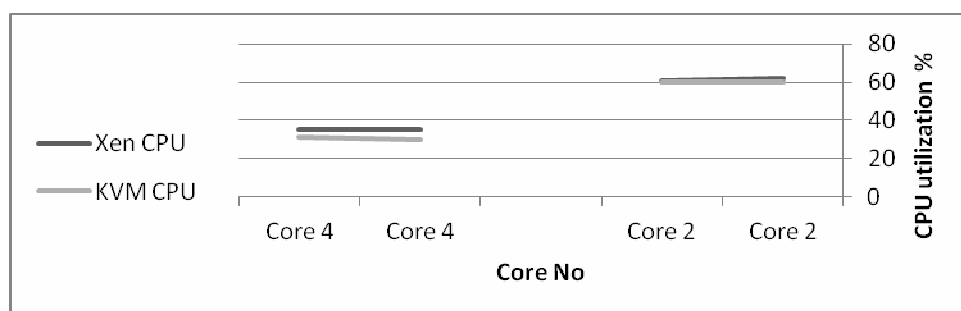


Figure 4.36: Effect of Changing Core Number on CPU Utilization

4. What is the effect of changing the **key** among the performance?

At 3DES EA, 192-bit key was only used.

5. What is the effect of changing the **Data size** among the performance?

Using a sample of three different data sizes 1, 2 and 5 GB with fixed key and core, results for response time are illustrated in figure 4.37, while the results for the CPU utilization were presented in figure 4.38.

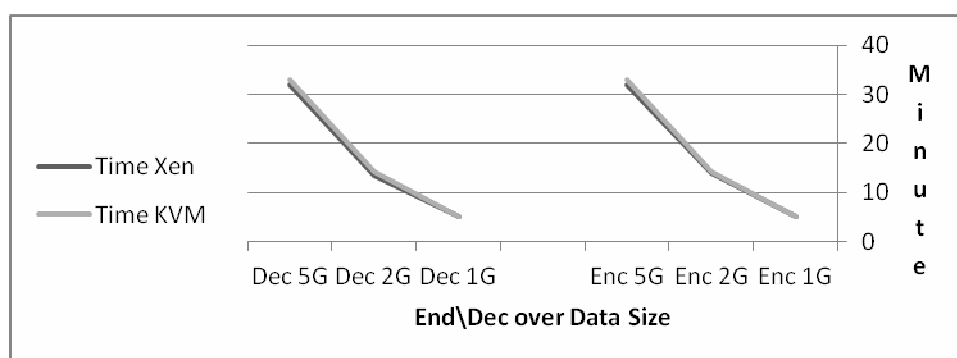


Figure 4.37: Effect of Changing Data Size on Time Response

This figure shows that when increasing the data size, time increases for both encryption and decryption.

By looking at figure 4.31 and figure 4.32, we can see that increasing the data size; makes KVM better than Xen relatively; while at small data, Xen and KVM are very close.

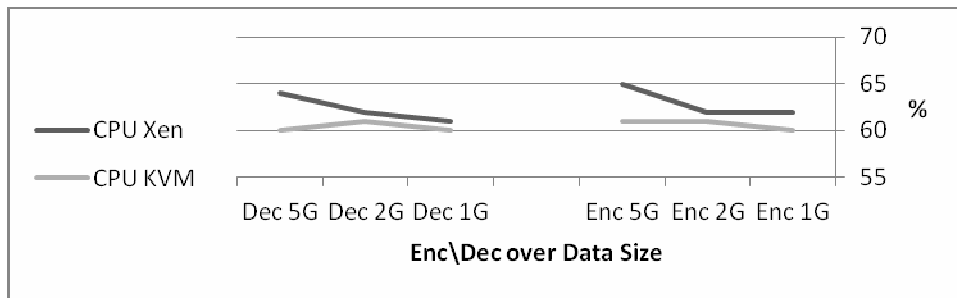


Figure 4.38: Effect of Changing Data Size on CPU Utilization

This figure shows that KVM has better CPU performance at big data.

Finally, 3DES performance summary among KVM and Xen shows:

- KVM is better than Xen in the case of CPU utilization for all results.
- Xen is better than KVM in case of Time.
- At small data Xen and KVM are closer than at big data.
- At core 4, KVM shows better performance than Xen.

4.2.5 ARC4

This algorithm as mentioned in the background in chapter 3 is a symmetric EA using stream cipher. Refer to appendix 5 to see the whole experiment table. The results of the experiments are shown in table 4.14.

Table 4.14: Result of all ARC4 Experiments

ARC4	Encr	Encr	Decr	Decr	Encr	Encr	Decr	Decr
Exp No	Xen Time	KVM Time	Xen Time	KVM Time	Xen CPU	KVM CPU	Xen CPU	KVM CPU
1	0.36	0.35	0.25	0.23	45	40	45	38
2	0.38	0.37	0.26	0.22	26	24	26	24
3	1.21	1.03	1.21	1.21	44	39	45	40
4	1.42	1.26	1.44	1.44	77	63	78	74
5	2.09	2.09	1.36	1.23	35	32	45	38
6	2.11	2.01	2.15	2.1	60	50	59	50
7	2.15	2.05	2.15	2.06	67	50	62	52
8	4.01	3.42	4.04	3.47	62	51	61	52
9	4.08	3.54	4	3.41	60	50	63	51
10	4.13	3.5	4.05	3.47	64	52	65	50
11	4.18	3.56	4.12	3.42	32	28	34	28
12	4.19	3.56	4.22	3.56	35	28	35	28
13	4.22	3.49	4.15	4.01	34	28	34	28

Figure 4.39 represents the encryption response time and figure 4.40 represents decryption response.

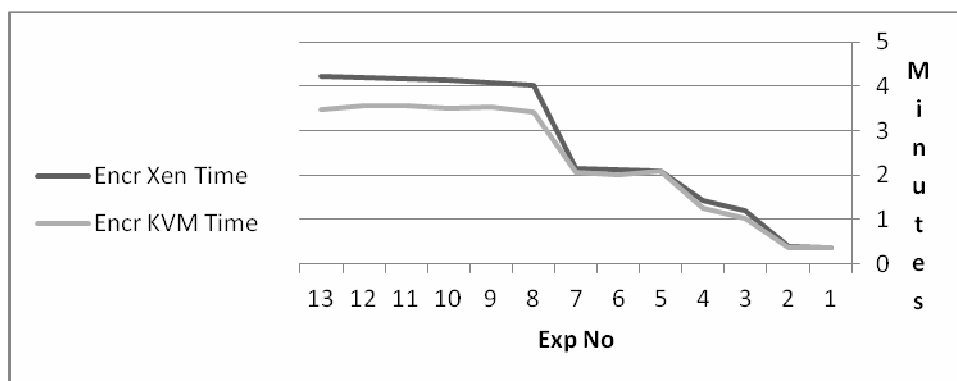


Figure 4.39: Represents Encryption Time for Xen and KVM.

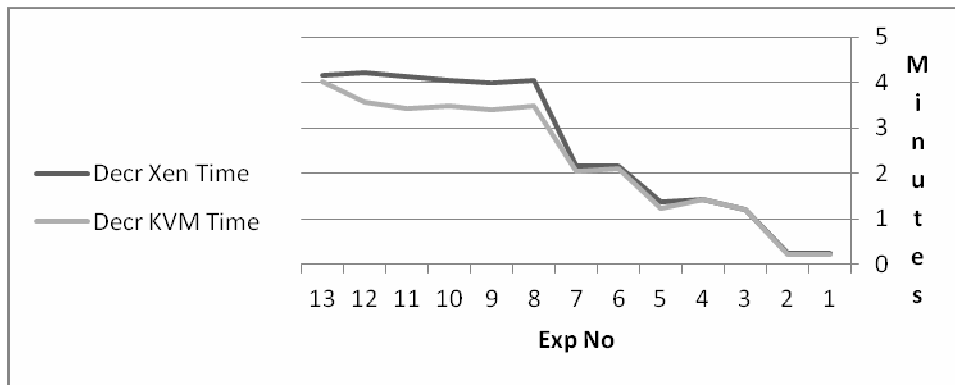


Figure 4.40: Represents Decryption Time for Xen and KVM.

It's clear that response time with KVM is less than Xen at all results. Also KVM is better at long response time than Xen. Here the CPU utilization results percentage % for the AES.

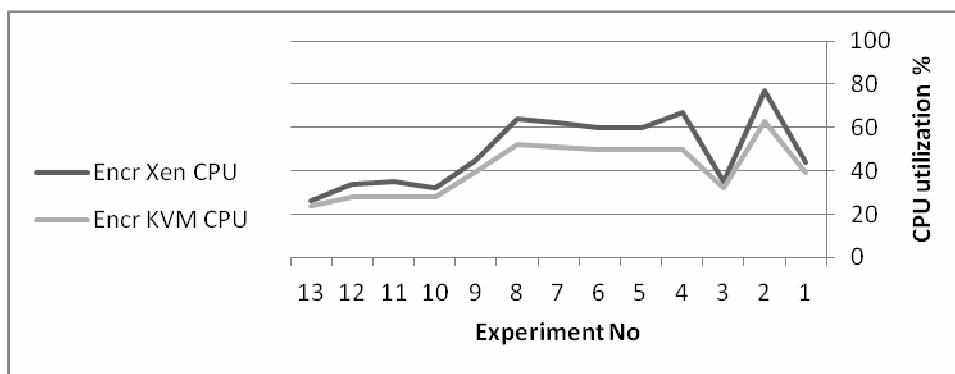


Figure 4.41: Represent the CPU at Encryption over Xen and KVM

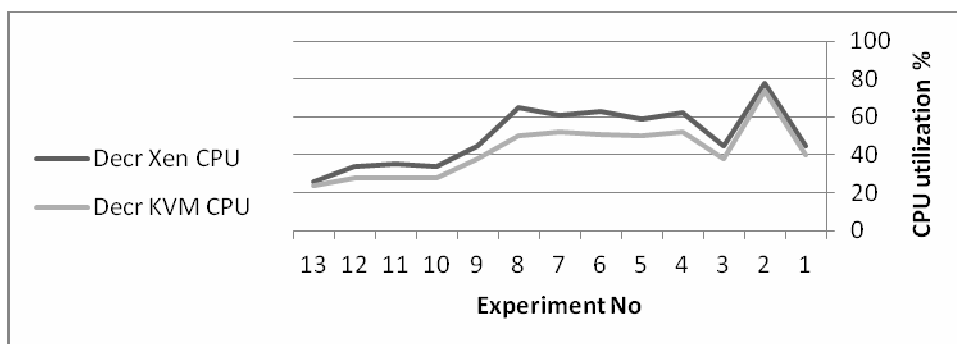


Figure 4.42: Represent the CPU at Decryption over Xen and KVM

Both diagrams (Figure 4.41 and Figure 4.42) clearly show that the CPU utilization in KVM is better than Xen in all results.

We have a clear vision in order to answer the five questions.

1. What is the **average** of time (m:s) and CPU utilization (percentage %) for every hypervisor?

Table 4.15 Average Time and CPU for Xen and KVM

Decr CPU	Encr CPU	Decr Time	Enc Time	ARC4
48.40	48.50	2.94	2.98	Xen
40.10	40.10	2.60	2.59	KVM
8.30	8.4	0.34	0.39	Difference

The average time table shows that the KVM is better than Xen at Enc\Dec with response time and CPU utilization.

2. What is the **correlation** for the EA?

Table 4.16: Correlation between all ARC4 columns

correlation	Enc Xen KVM	Dec Xen KVM	Enc Dec Xen	Enc Dec KVM
Time	1.00	0.99	0.99	0.97
CPU	0.99	0.98	0.98	0.97

Table 4.16 shows that there's a strong relationship between all of the ARC4 components especially at Xen and encryption.

3. What is the effect of changing the CPU core among the performance?

A sample was taken and this sample has the same data and key, but different in core numbers. Figure 4.43 shows that when increasing cores; response time is close for both Xen and KVM.

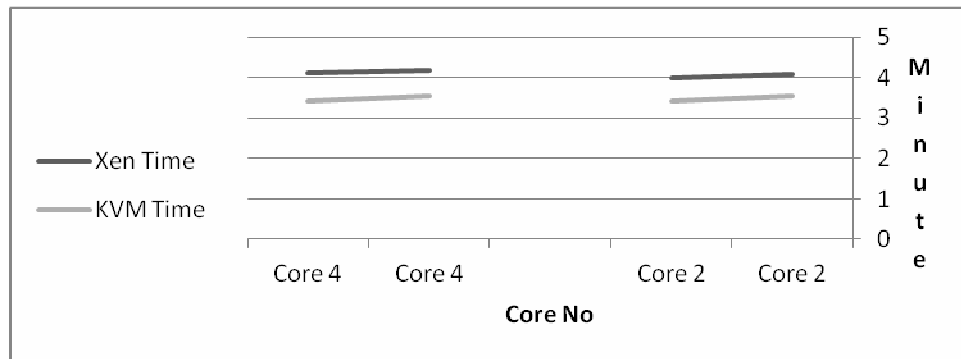


Figure 4.43: Effect of Changing Core Number on Response Time

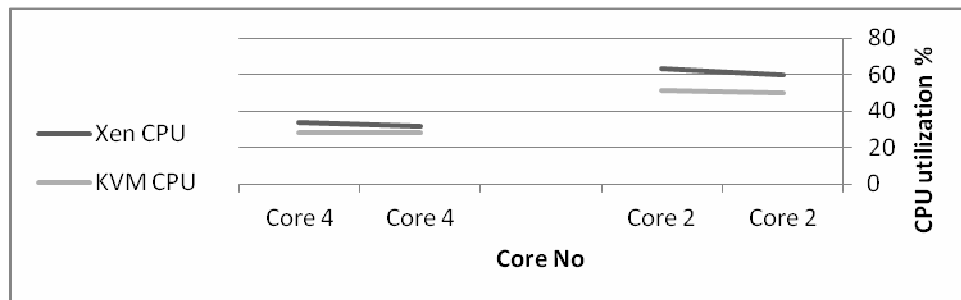


Figure 4.44: Effect of Changing Core Number on CPU Utilization

Figure 4.44 shows the CPU utilization results when changing cores from 2 cores into 4 cores. The CPU utilization decreases about half, also the KVM shows better performance at core 2 than Xen.

4. What is the effect of changing the **key** among the performance?

Using a sample of three different keys 24, 32 and 48 with the same data size and core numbers. Figure 4.45 shows response time results, while figure 4.46 shows the CPU utilization results.

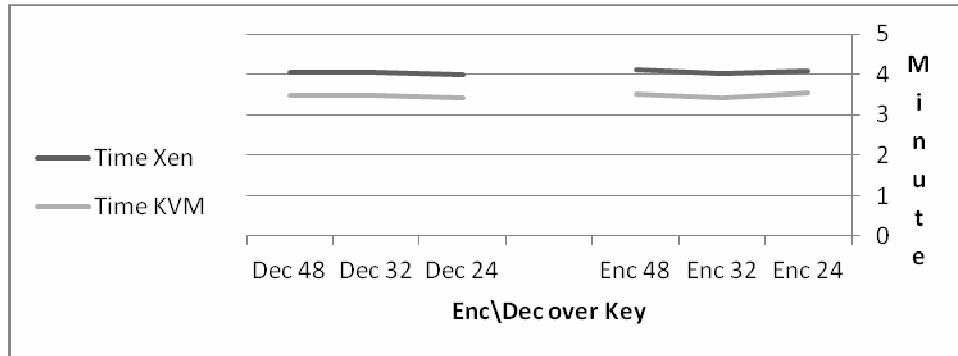


Figure 4.45: Effect of Changing Key on Time

This figure shows that when changing the key, there is no stable effect for increasing or decreasing the response time, especially for encryption process.

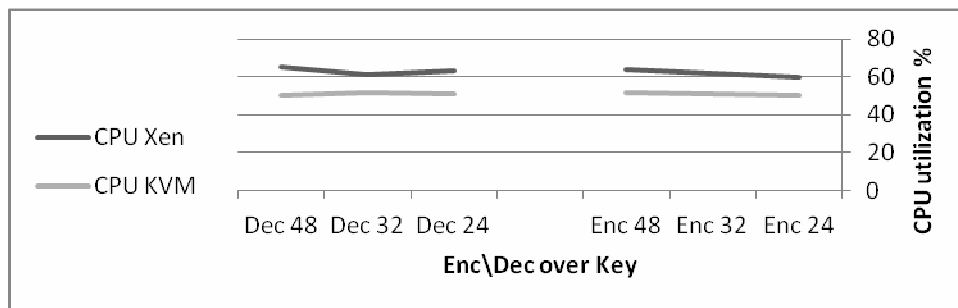


Figure 4.46: Effect of Changing the Key on CPU

This figure shows that when changing key, there is no stable effect for increasing or decreasing CPU utilization especially for decryption process.

5. What is the effect of changing the **Data size** among the performance?

Using a sample of three different data sizes 1, 2 and 5 GB with fixed key and core, results for response time are presented in figure 4.47, while the results for the CPU utilization are displayed in figure 4.48.

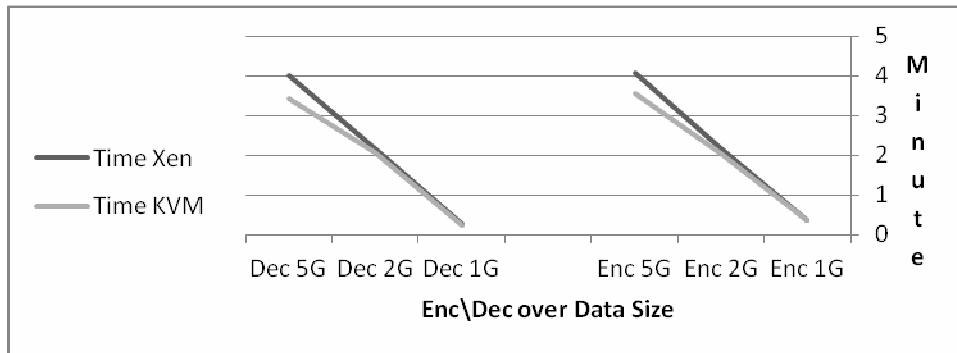


Figure 4.47: Effect of changing Data Size on Response Time

This figure shows that when increasing data size, time increases for both encryption and decryption.

By looking at figure 4.39 and figure 4.40, we can see that when increasing the data size; KVM is better than Xen relatively; while at small data, Xen and KVM are very close.

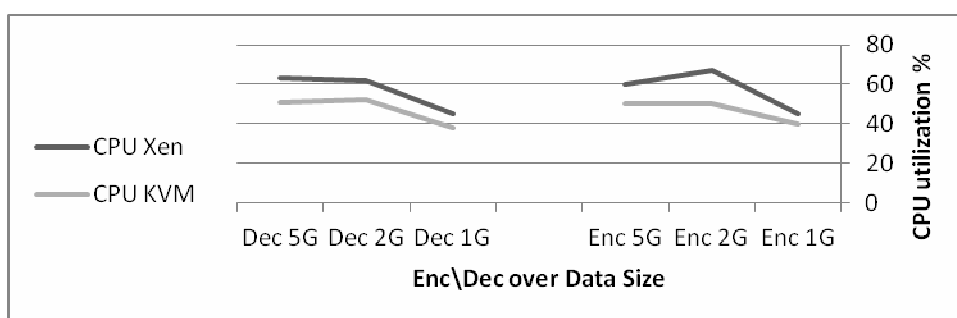


Figure 4.48: Effect of changing the Data Size on CPU Utilization

This figure shows that KVM has better CPU performance at big data especially at decryption.

Finally, ARC4 performance summary among KVM and Xen shows:

- KVM is better than Xen at average time and all CPU utilization.
- At Core 4 and small data, Xen and KVM are very close at performance; else KVM performance is better than Xen.
- Key has no significant effect on performance.

4.2.6 CAST-128

This Algorithm as mentioned in the background in chapter 3 is a symmetric EA using block cipher, refer to appendix 6 to see the whole experiment table. The results of the experiments are shown in (Table 4.17).

Table 4.17: Result of all CAST-128 Experiments

CAST-128	Encr	Encr	Decr	Decr	Encr	Encr	Decr	Decr
Exp No	Xen Time	KVM Time	Xen Time	KVM Time	Xen CPU	KVM CPU	Xen CPU	KVM CPU
1	0.37	0.32	0.35	0.3	50	45	50	39
2	0.55	0.46	0.41	0.4	34	32	32	31
3	1.27	1.2	1.12	1.01	54	51	65	60
4	1.35	0.58	1.3	0.55	34	31	32	28
5	1.52	1.38	1.5	1.43	39	30	40	37
6	2.33	2.02	2.24	2.23	39	37	37	35
7	2.4	2.29	2.29	2.27	65	55	66	54
8	4.28	4.24	3.32	3.3	35	27	36	28
9	5	4.36	5	4.56	67	55	66	57

Figure 4.6.1 shows encryption response time while figure 4.49 shows decryption response time (Figure 4.50).

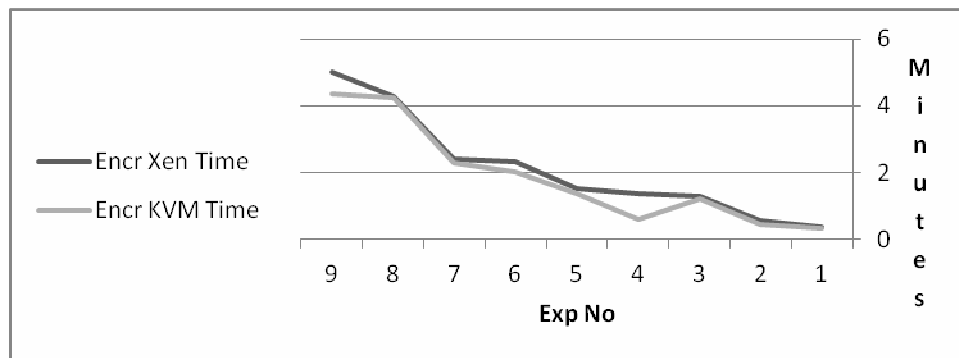


Figure 4.49: Encryption response Time for Xen and KVM

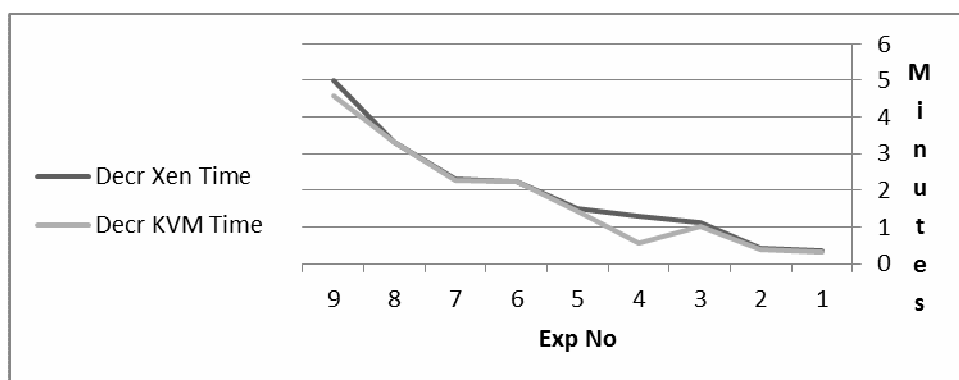


Figure 4.50: Decryption response Time for Xen and KVM.

It is clear that response time with KVM is less than Xen at all results. KVM are close to Xen since response time for CAST-128 is relatively short. Here the CPU utilization results percentage % for the CAST-128.

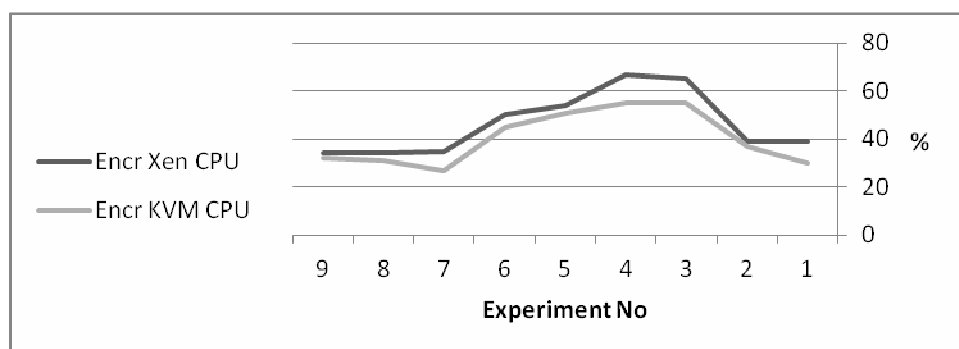


Figure 4.51: The CPU at Encryption over Xen and KVM

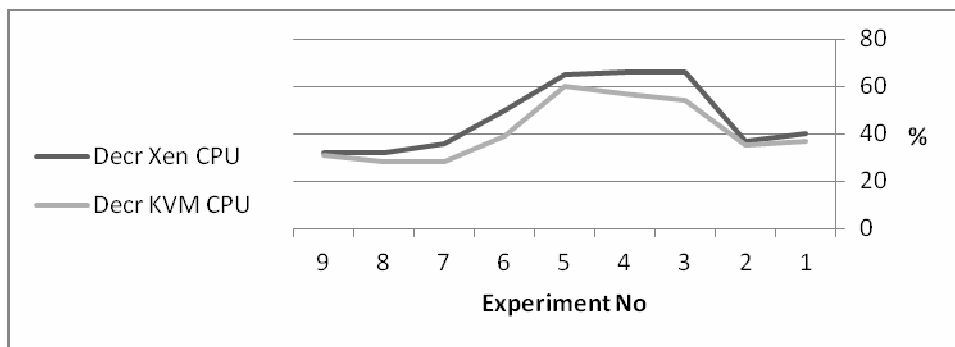


Figure 4.52: Represent the CPU at Decryption over Xen and KVM

(Figure 4.51 and Figure 4.52) clearly show that CPU utilization in KVM is better than Xen in all results.

After discussing the results, we stated the answers for the five questions below:

1. What is the **average** of time (m:s) and CPU utilization (percentage %) for every hypervisor?

Table 4.18 Average performance for Xen and KVM

Decr CPU	Encr CPU	Decr Time	Enc Time	CAST-128
48.33	46.33	1.95	2.11	Xen
41.00	40.33	1.90	1.87	KVM
7.33	6.0	0.05	0.34	Difference

The results in (Table 4.18) show that KVM is better than Xen in all CPU utilization and response time results also with average in encryption and decryption.

2. What is the **correlation** for the EA?

Table 4.19: Correlation between all CAST-128

correlation	Enc Xen KVM	Dec Xen KVM	Enc Dec Xen	Enc Dec KVM
Time	0.99	0.98	0.98	0.98
CPU	0.96	0.97	0.97	0.93

Correlation table (Table 4.19) shows that there is a strong relationship between all of CAST-128 components especially at Xen.

3. What is the effect of changing the CPU core among the performance?

A sample was taken and has the same data and key, but different at core numbers. Response time (Figure 4.53) shows that when increasing cores, response time decreases for both Xen and KVM, while at core 2, KVM has better response time performance.

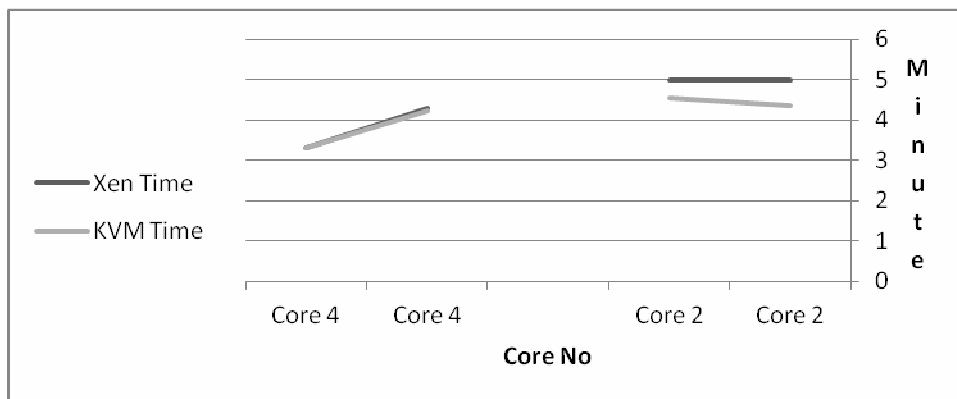


Figure 4.53: Effect of Changing Core Number on Response Time

CPU utilization results (Figure 4.54) for changing cores from 2 cores into 4 cores; show that CPU utilization will decrease about half, also KVM show more performance at core 2 than Xen.

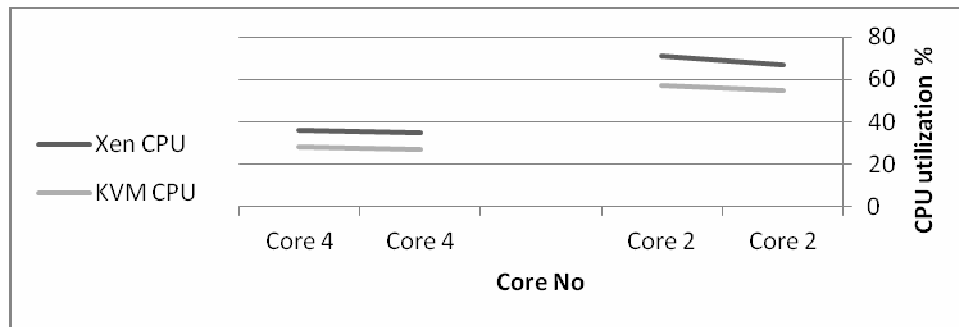


Figure 4.54: Effect of changing core number on CPU utilization

4. What is the effect of changing the **key** among the performance?

This is CAST-128.

5. What is the effect of changing the **Data size** among the performance?

Using a sample of three different data sizes 1,2 and 5 GB with fixed key and core, results for response time in (Figure 4.55), while results for CPU utilization in (Figure 4.56).

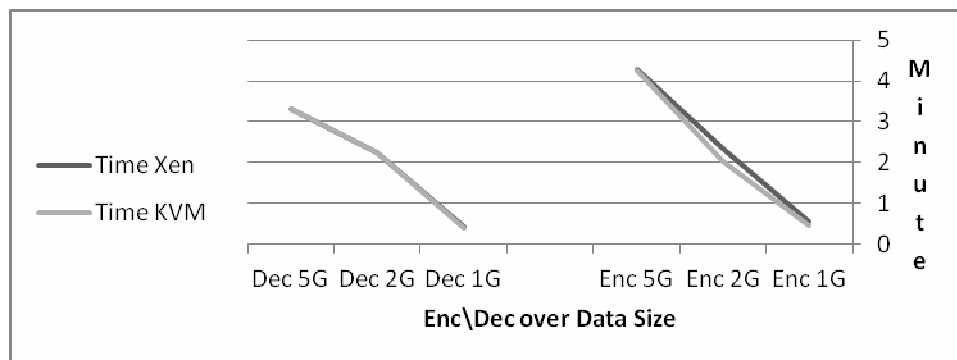


Figure 4.55: Effect of changing data size on time response

This figure show that when increasing data size, time increases for both encryption and decryption.

But it is clear that at decryption, Xen and KVM are very close than encryption.

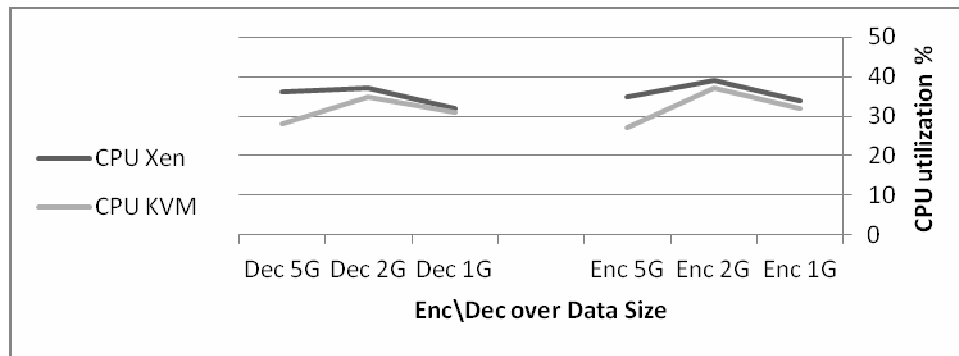


Figure 4.56: Effect of changing data size on CPU utilization

This figure shows that KVM has better CPU performance at big data than Xen.

Finally, CAST-128 performance summary on KVM and Xen shows:

- KVM is better than Xen at all response time and all CPU utilization results.
- Results show that at core 2 and small data there is more consistency between two hypervisors.

4.2.7 BLOWFISH

This Algorithm as mentioned in the background in chapter 3 is a symmetric EA using block cipher. Refer to appendix 7 to see the whole experiment table. The results of the experiments are shown in (Table 4.20).

Table 4.20: Result of all BlowFish experiments

BlowFish	Encr	Encr	Decr	Decr	Encr	Encr	Decr	Decr
Exp No	Xen Time	KVM Time	Xen Time	KVM Time	Xen CPU	KVM CPU	Xen CPU	KVM CPU
1	2.21	2.11	2.15	2.05	66	60	67	60
2	2.24	2.01	2.26	2	35	31	40	32
3	4.07	3.37	3.59	3.21	36	33	36	32
4	5.17	4.22	3.4	3.3	66	59	65	61
5	6	5.51	6	5.24	35	31	34	34
6	6.15	5.51	5.53	5.37	64	59	67	61
7	7.36	7.35	7.38	7.38	66	60	67	61
8	7.5	7.15	8.15	6.35	36	32	36	34
9	14.24	13.27	14.01	13.03	67	60	67	61
10	14.59	12.45	14.56	12.4	34	32	36	33

Figure 4.57 shows encryption response time while figure 4.58 shows decryption response time.

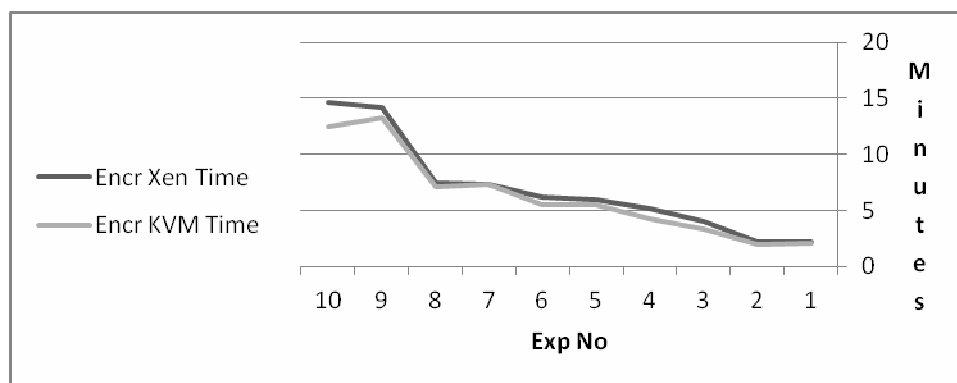


Figure 4.57: Represents Encryption Time for Xen and KVM

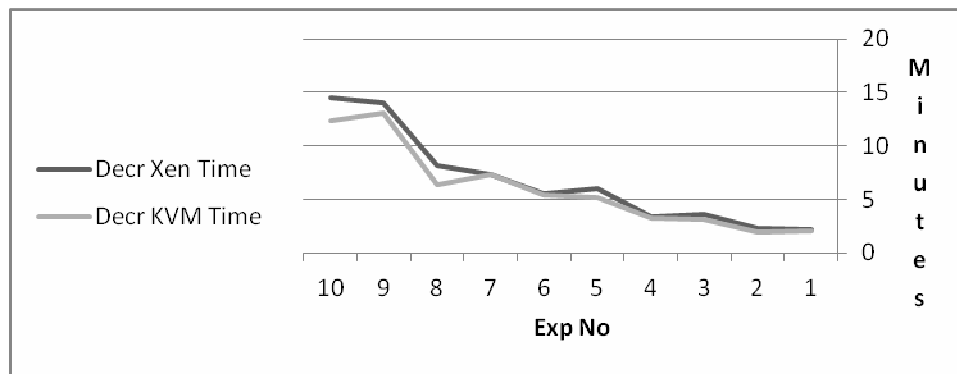


Figure 4.58: Represents Decryption Time for Xen and KVM

It is clear that response time for KVM is less than Xen at all results either for encryption or decryption. KVM shows better performance at long response time process. Here the CPU utilization results percentage % for the BlowFish.

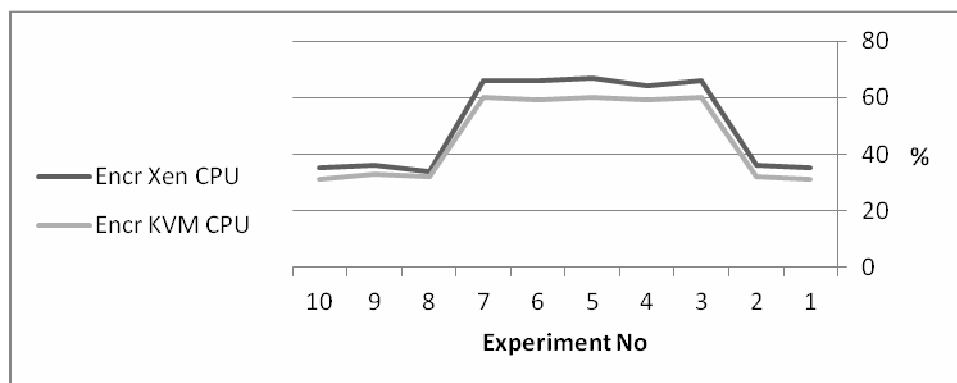


Figure 4.59: Represent the CPU at Encryption over Xen and KVM

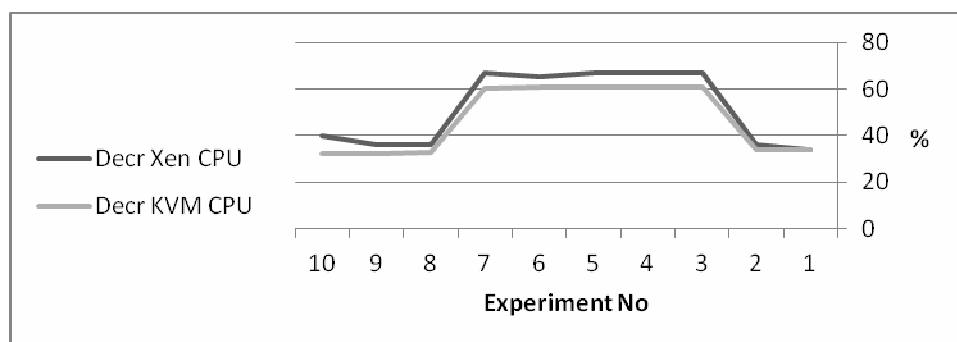


Figure 4.60: Represent the CPU at decryption over Xen and KVM

Both diagrams (Figure 4.59 and Figure 4.60) clearly show that the CPU utilization in KVM is better than Xen in all results. Big results are for core 4.

Below, we stated the answers for the five questions.

1. What is the **average** of time (m:s) and CPU utilization (percentage %) for every hypervisor?

Table 4.21 Average performance for Xen and KVM

Decr CPU	Encr CPU	Decr Time	Enc Time	BLOWFISH
51.50	50.50	7.10	7.35	Xen
46.90	45.70	6.03	6.30	KVM
5.6	4.8	1.07	1.05	Difference

Results at (Table 4.21) show that KVM is better than Xen in all CPU utilization and response time results also with average.

2. What is the **correlation** for the EA?

Table 4.22: Correlation between all BlowFish

correlation	Enc Xen KVM	Dec Xen KVM	Enc Dec Xen	Enc Dec KVM
Time	0.99	0.99	0.99	1.00
CPU	1.00	0.99	0.99	1.00

Results

Results of Table 4.22 show that there is a strong relationship between Xen and KVM at all times especially with KVM.

3. What is the effect of the CPU **core number** on the performance?

A sample was taken and has the same data and key, but different at core numbers. Response time in (Figure 4.61) shows that at increasing cores, response time is decreasing for both Xen and KVM, while at core 2 KVM has better response time performance.

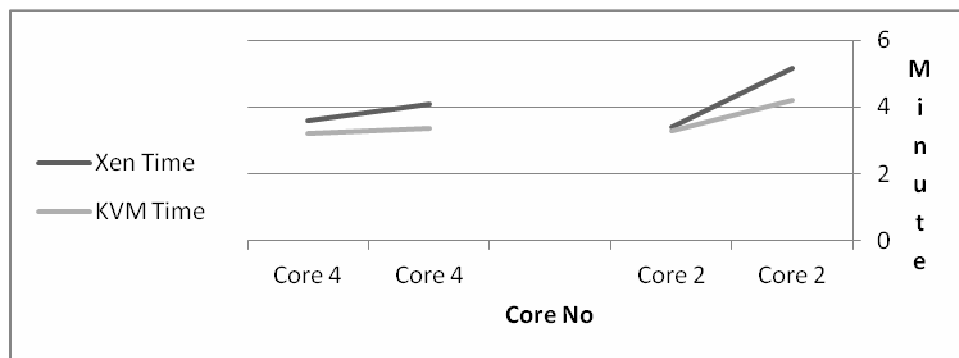


Figure 4.61: Effect of Changing Core Number on Response Time

CPU utilization results in (Figure 4.62) for changing cores from 2 cores in to 4 cores; show that CPU utilization decreases about half, also KVM shows better performance at core 2 than Xen.

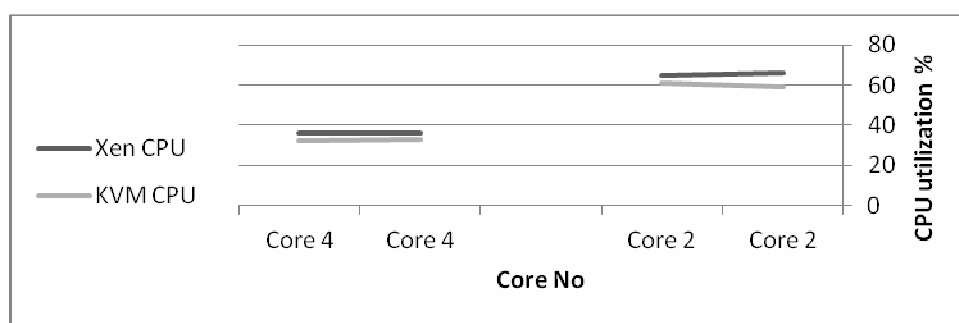


Figure 4.62: Effect of Changing Core Number on CPU Utilization

4. What is the effect of the **key** on the performance?

At BLOWFISH, 64-bit key is used only.

5. What is the effect of the **Data size** on the performance?

Using a sample of three different data sizes 1, 2 and 5 GB with fixed key and core, results for response time at (Figure 4.63), while results for CPU utilization at (Figure 4.64).

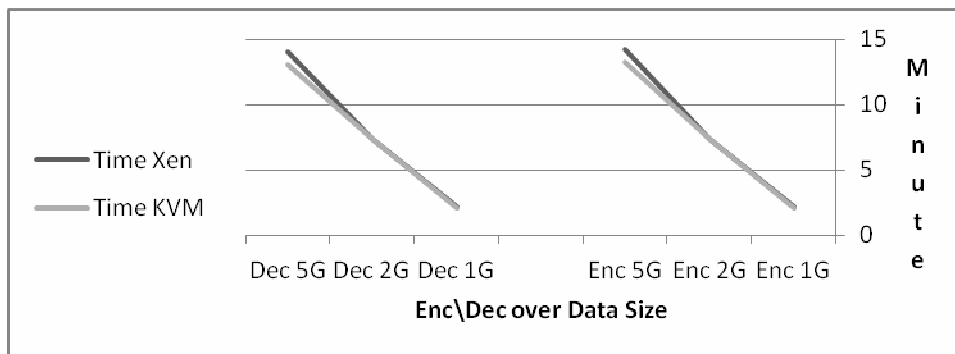


Figure 4.63: Effect of Changing Data Size on time Response

This figure shows that when increasing data size, time increases for both encryption and decryption.

But it is clear that big data, KVM shows better response time than Xen.

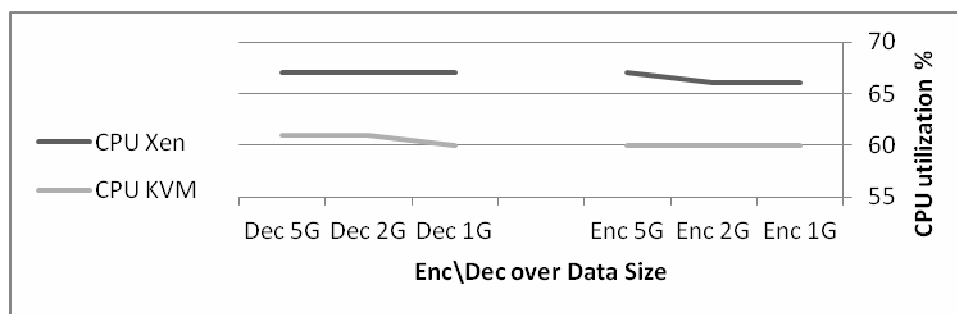


Figure 4.64: Effect of Changing Data Size on CPU Utilization

This figure shows that KVM and Xen at encryption have better performance at CPU utilization than decryption.

Finally, BLOWFISH performance summary among KVM and Xen shows:

- KVM is better than Xen at all CPU utilization, Time, averages and standard deviation with strong correlation at all Blowfish results.
- At core 2 and small data, KVM and Xen show more consistency, otherwise KVM is better than Xen.

4.2.8 TwoFish

This Algorithm as mentioned in the background in chapter 3 is a symmetric EA using block cipher, refer to appendix 8 to see the whole experiment table.

Table 4.23: Result of all TwoFish experiments

TwoFish	Encr	Encr	Decr	Decr	Encr	Encr	Decr	Decr
Exp No	Xen Time	KVM Time	Xen Time	KVM Time	Xen CPU	KVM CPU	Xen CPU	KVM CPU
1	3.04	3.29	3.08	3.15	65	61	66	64
2	3.15	3.16	3.12	3.13	36	32	35	32
3	3.19	3.23	3.21	3.22	64	61	65	64
4	3.27	3.29	3.19	3.22	35	32	38	36
5	3.29	3.31	3.29	3.32	65	62	64	64
6	3.35	3.38	3.33	3.33	35	32	35	32
7	9.21	9.01	9.15	8.52	34	32	34	32
8	9.48	9.36	9.59	9.24	36	32	36	32
9	11.21	11.21	11.07	11.06	35	32	36	33
10	12.43	12.13	12.3	12.13	63	59	64	61
11	12.5	11.5	12.54	11.39	65	61	64	61

Encryption response time is represented in (Figure 4.65) and decryption response time is represented in (Figure 4.66).

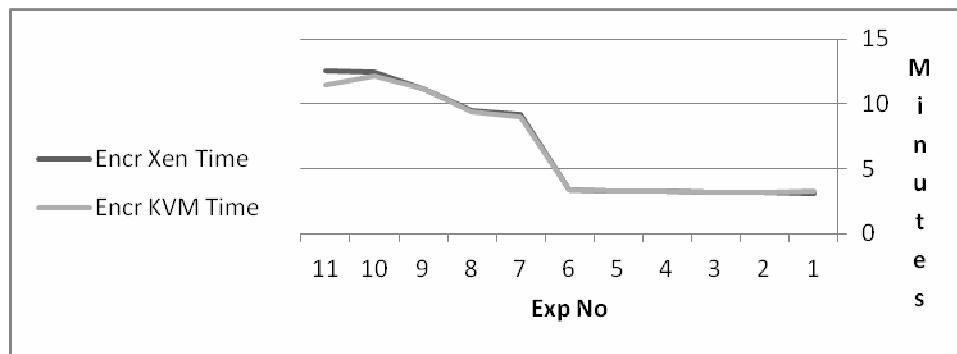


Figure 4.65: Represents encryption time for Xen and KVM.

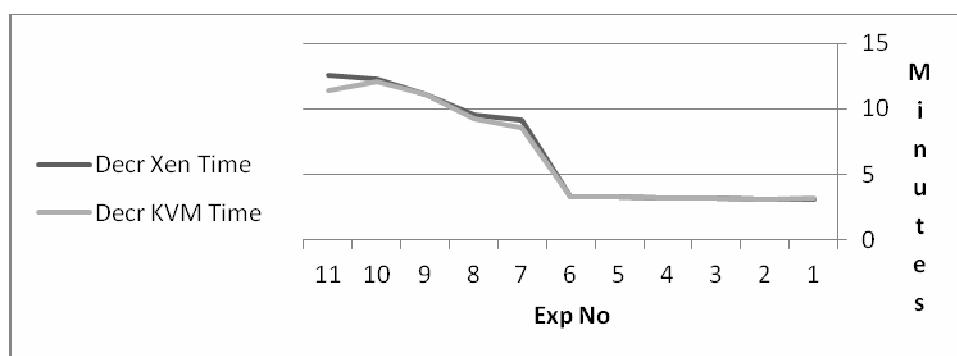


Figure 4.66: Represents decryption time for Xen and KVM.

It is clear that response time with KVM is less than Xen at all big data size results either for encryption or decryption; but at small data size Xen is better than KVM, while at long response time experiment KVM is better than Xen; here the CPU utilization results percentage % for the TwoFish.

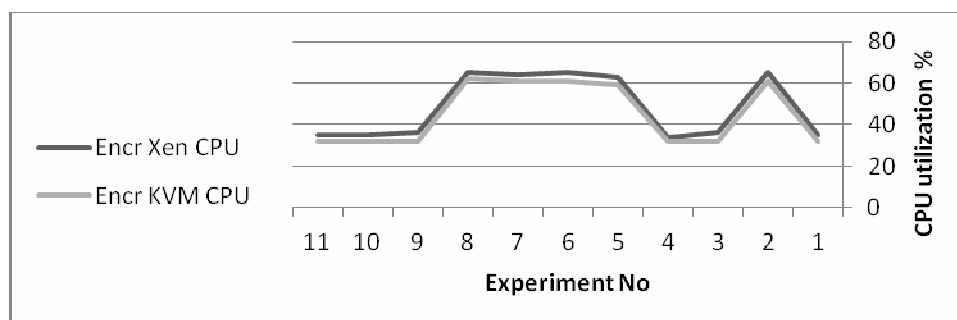


Figure 4.67: Represent the CPU at encryption over Xen and KVM

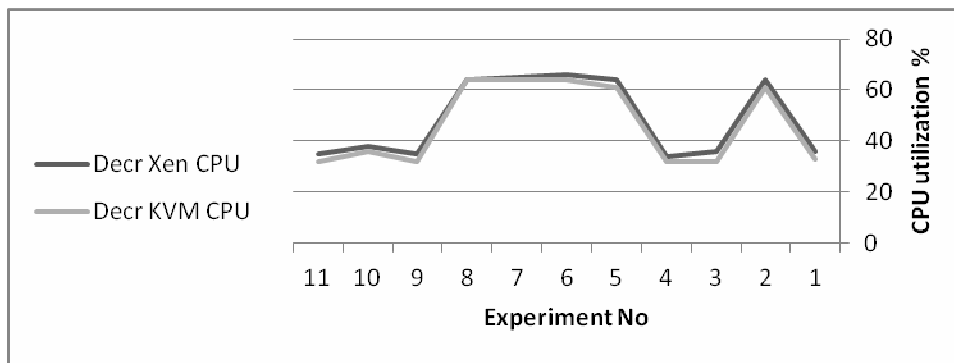


Figure 4.68: Represent the CPU at decryption over Xen and KVM

Both diagrams (Figure 4.67 and Figure 4.68) clearly show that CPU utilization in KVM is better than Xen in all results.

The answers for the five questions are stated below.

1. What is the **average** of time (m:s) and CPU utilization (percentage %) for every hypervisor?

Table 4.24 Average performance for Xen and KVM

Decr CPU	Encr CPU	Decr Time	Enc Time	TWOFISH
48.56	48.11	5.58	5.60	Xen
46.33	44.78	5.47	5.57	KVM
2.23	3.34	0.11	0.03	Difference

Using (Table 4.24), average response time and average CPU utilization show that KVM is better than Xen. At response time results, KVM is better than Xen at most results. KVM is better than Xen at all CPU utilization results.

2. What is the **correlation** for the EA?

Table 4.25: Correlation between all TwoFish

correlation	Enc Xen KVM	Dec Xen KVM	Enc Dec Xen	Enc Dec KVM
Time	1.00	1.00	1.00	1.00
CPU	1.00	1.00	1.00	1.00

Results of (Table 4.25) show that there is a strong relationship between Xen and KVM at all times; TwoFish is the best EA at all component correlation.

3. What is the effect of the CPU **core** number on the performance?

A sample was taken and has the same data and key, but different at core numbers. Response time in (Figure 4.69) shows that at increasing cores, response time is decreasing for both Xen and KVM, while at core 2 KVM has better response time performance.

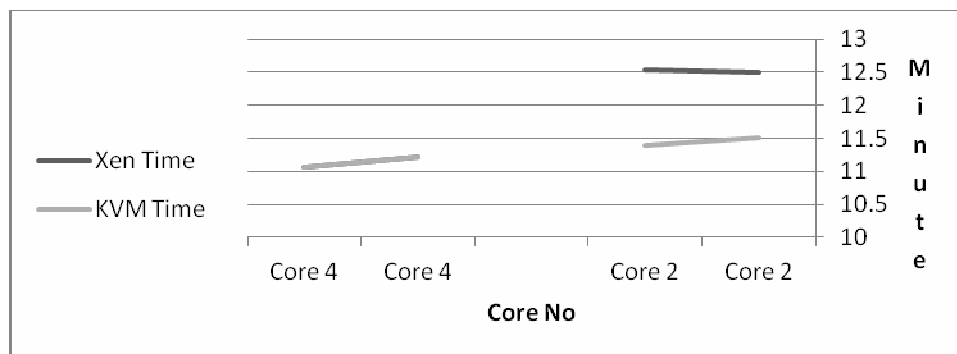


Figure 4.69: Effect of changing core number on response time

CPU utilization results in (Figure 4.70) for changing cores from 2 cores in to 4 cores; show that CPU utilization decreases less than a half, also KVM shows better performance at core 2 than Xen.

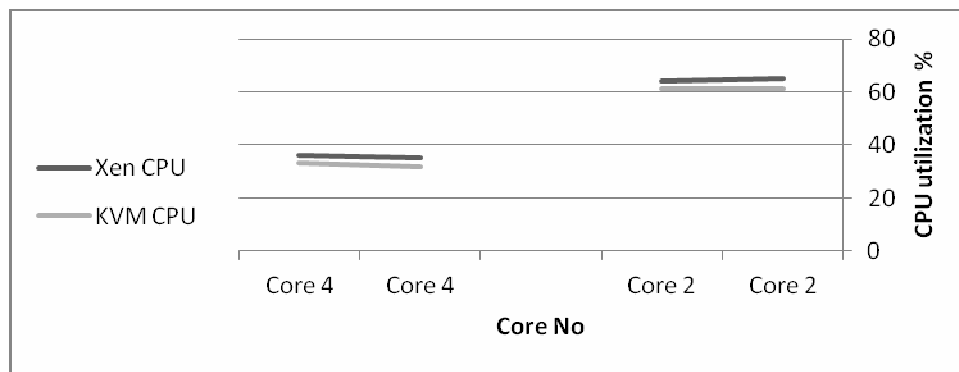


Figure 4.70: Effect of changing core number on CPU utilization

4. What is the effect of the **key** on the performance?

Using a sample of three different keys 128, 192 and 256 with same data size and core numbers; results for response time at (Figure 4.71), while for the CPU utilization in (Figure 4.72).

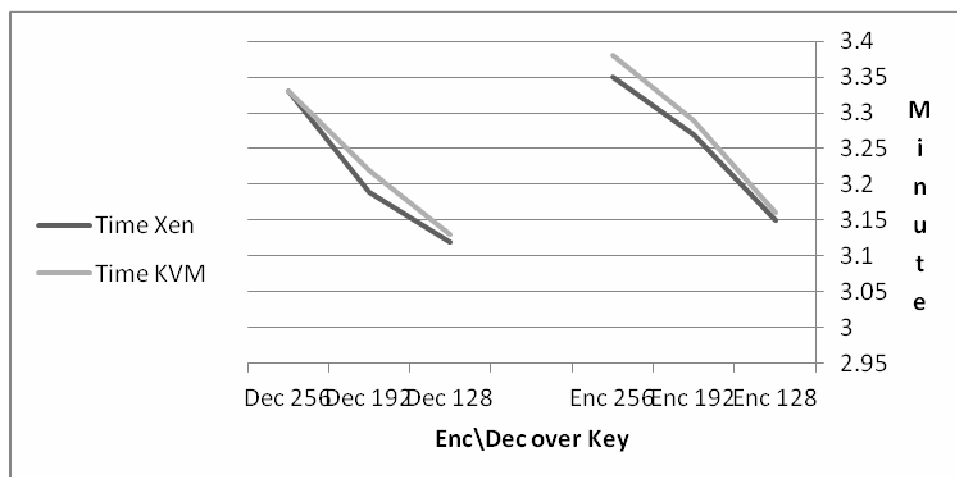


Figure 4.71: Effect of changing key on time

This figure shows that when increasing key value, there is an increase in response time especially for encryption process.

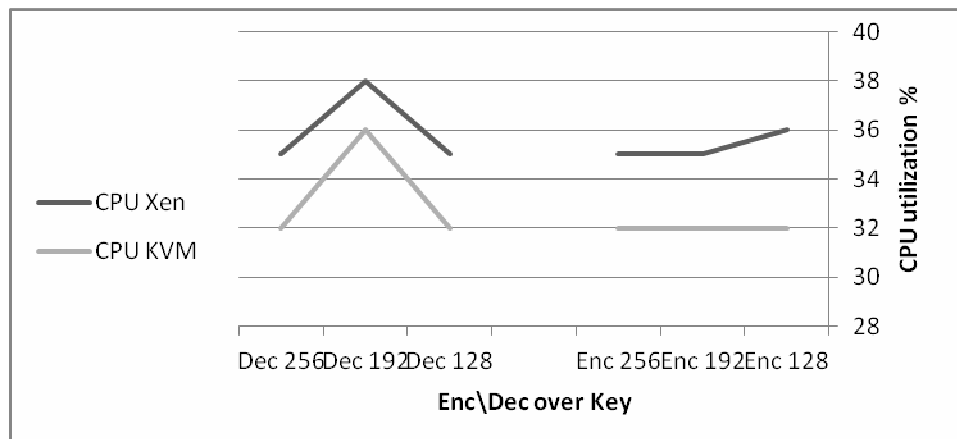


Figure 4.72: Effect of changing key on CPU utilization

This figure shows that when changing key, there is no stable effect also for increasing or decreasing CPU utilization especially for decryption process.

5. What is the effect of changing the **Data size** among the performance?

From index table results (see index 8) and (Table 4.8.1 and Table 4.8.2) it is clear that at small data size Xen is better than KVM at response time, even if we change Key or core; else KVM is better in all performance results.

Finally, Twofish performance summary among KVM and Xen show:

- This EA show that core 4 is better than 2 in general.
- Time is increasing by increasing key value.
- At small data, Xen is better than KVM in time response, else KVM is better.
- KVM is better than Xen at all CPU utilization results.

4.3 Summary Results

After all EA finished, we saw a comparative for all performance results of Enc\Dec over the two hypervisor; for every results set (columns), we show the Average and STDEV. Correlation calculates also between columns later.

Table 4.26: Table of all EA response time

All EA	En Time X	En Time K	De Time X	De Time K
RSA	2.12	1.87	1.95	1.78
AES	2.98	2.59	2.94	2.60
DES	3.53	3.10	3.47	2.96
3DES	5.60	5.57	5.58	5.47
ARC4	6.95	6.30	6.70	6.03
CAST128	6.95	6.54	6.74	6.17
BlowFish	12.35	12.70	12.21	12.67
TwoFish	24.22	20.54	38.13	30.55
Average	8.09	7.40	9.72	8.53
STDEV	6.80	5.90	11.15	8.91

Table 4.26 shows the eight encryption algorithms results for response time; the average Enc\Dec response time for the Xen and KVM hypervisors, then the results of total average and standard deviation.

To calculate the average between Xen and KVM we use formulas as follow:

$$\text{So, Enc Ratio} = (8.09 - 7.4) / 7.4 = 9.3\%$$

$$\text{Dec Ratio} = (9.72 - 8.53) / 8.53 = 14 \%$$

$$\text{So the average for Average time} = (9.3 + 14) / 2 = 11.65 \%$$

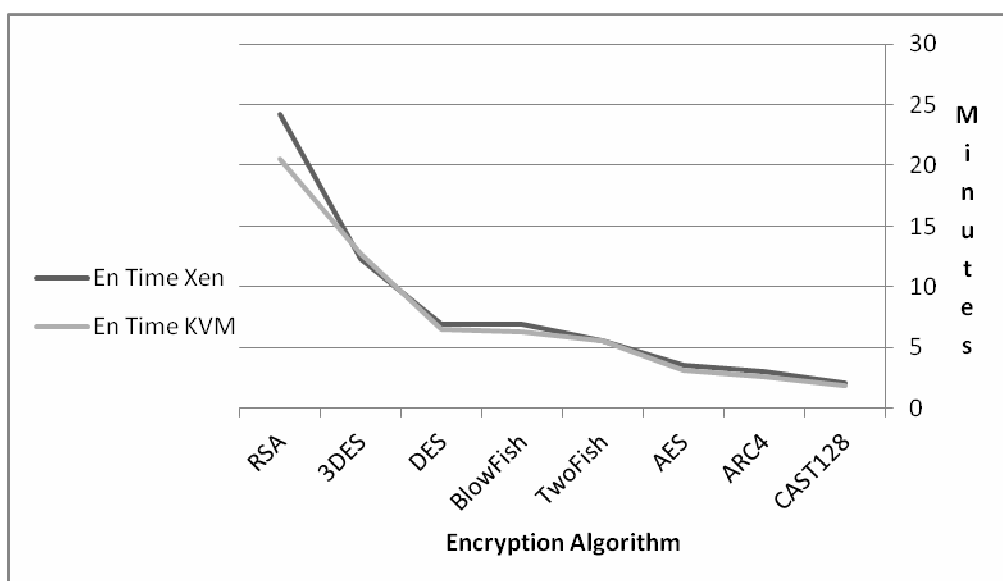


Figure 4.73: Average encryption response time for all EA

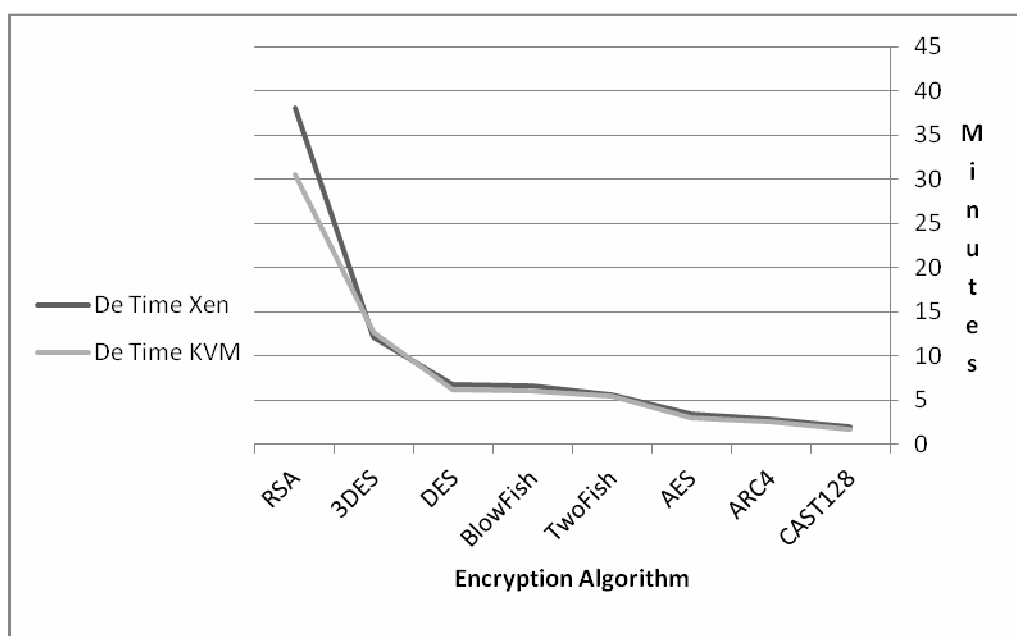


Figure 4.74: Average decryption response time for all EA

These two diagrams (Figure 4.73 and 4.74) show that at average Enc\Dec response time KVM is better than Xen except TripleDES.

We checked the average CPU utilization performance for the eight EA, with average and STDEV at (Table 4.27).

Table 4.27: Table of all EA CPU utilization results

All EA	Enc CPU %	Enc CPUK	Dec CPU %	Dec CPUK
CAST128	46.33	40.33	47.11	41.00
ARC4	48.50	40.10	48.40	40.10
AES	45.44	41.00	47.44	42.56
TwoFish	48.11	44.78	48.56	46.33
BlowFish	50.50	45.70	51.50	46.90
DES	49.22	44.11	49.11	44.78
3DES	50.57	47.14	50.57	47.86
RSA	73.60	70.20	71.70	68.10
Average	51.54	46.67	51.80	47.20
STDEV	8.51	9.22	7.65	8.32

Table 4.27 shows the average Enc\Dec CPU utilization for the Xen and KVM hypervisors, then the results of total average and standard deviation.

To calculate the average between Xen and KVM we use formulas as follow:

Enc Average between Xen and

KVM:

$$\text{So, Enc Ratio} = (51.72 - 46.67) / 46.67 = 10.8\%$$

$$\text{Dec Ratio} = (52.48 - 47.20) / 47.2 = 11.2\%$$

$$\text{So the average for Average time} = (10.2 + 11.2) / 2 = 11\%$$

By the same way, STDEV can be measured.

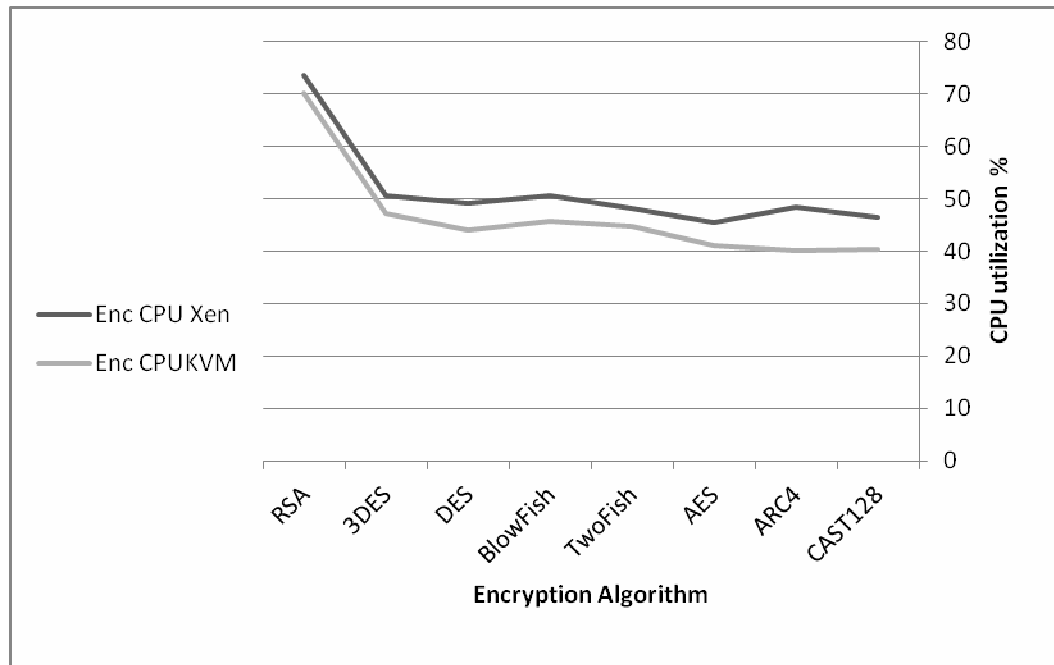


Figure 4.75: Average encryption CPU utilization for all EA

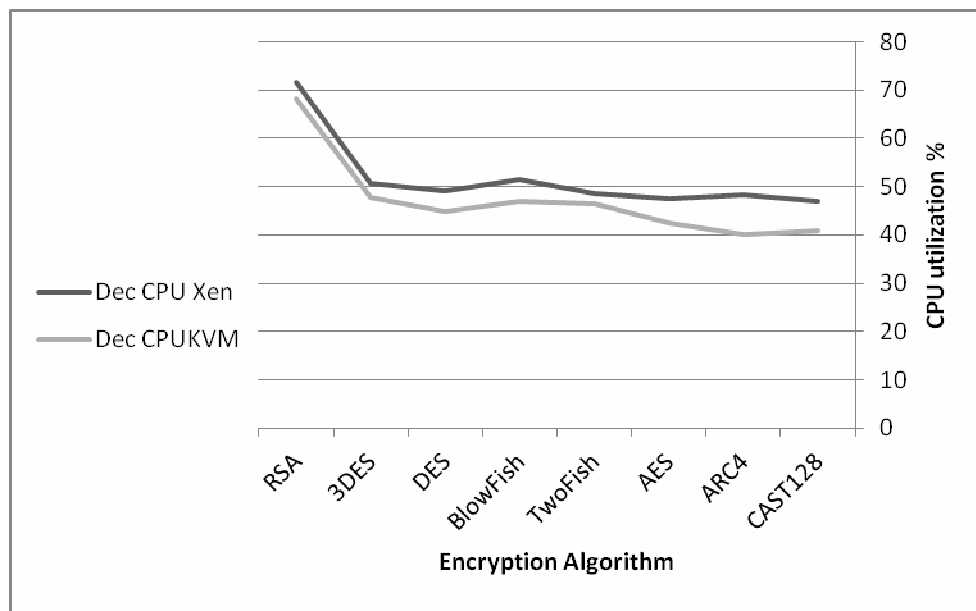


Figure 4.76: Average decryption CPU utilization for all EA

These two diagrams (Figure 4.75 and 4.76) show that at average Enc\Dec at CPU utilization, KVM is better than Xen. RSA have high performance since we use in some experiment two threads, some results when using it show 100% at CPU utilization

Now, the final correlations table to the eight encryption algorithms.

**Table 4.28: Correlation of all EA
components**

correlation	Enc Xen KVM	Dec Xen KVM	Enc Dec Xen	Enc Dec KVM
Time	0.99	1.00	0.98	0.98
CPU	0.99	0.98	1.00	1.00

Table 4.28 shows that correlation between the averages of all encryption algorithms components has strong relation in the performance results.

CHAPTER FIVE

Conclusion and Future Work

5.1 Conclusion

This chapter concludes all about the performance for the Xen and KVM hypervisors, the performance in this research takes into consideration response time and CPU utilization.

5.1.1 Response time performance conclusion:

- KVM is better than Xen at ratio Encrypt time with 9.3%.
- KVM is better than Xen at ratio Decrypt time with 14 %.
- Standard Deviation shows that KVM is better than Xen at Encrypt Time with 15.3%.
- Standard Deviation shows that KVM is better than Xen at Decrypt Time with 25.4%.
- Results show that KVM is better than Xen at **all** processes results in case of: AES, DES, ARC4, CAST-128, and BLOWFISH, with average of 62.5%
- Results show that KVM is better than Xen at **MOST** processes results in case of: RSA, and TWOFISH, with average of 25%.
- Results show that KVM is better than Xen at **all** processes results in case of: TripleDES, with average of 12.5%.

- RSA with big key and multithread, Xen and KVM are very close.
- In the case of RSA with small key, KVM is preferred with big difference at response time up to (200%).
- In the case of TWOFISH with small data, Xen is better than KVM even when changing core or key.

5.1.2 CPU utilization performance conclusion:

- KVM is better than Xen at ratio Encrypt CPU with 10.8%.
- KVM is better than Xen at ratio Decrypt CPU with 11.2%.
- Standard Deviation shows that Xen is better than KVM at Encrypt CPU with 8.9%.
- Standard Deviation shows that Xen is better than KVM at Decrypt CPU with 11.8%.
- KVM is better than Xen in all the results with all EA key, core, and data size.
- Most results make KVM and Xen consistent at core 4 than 2.

5.2 Research Contribution

- KVM shows best at CPU utilization at all results, this is very important for servers, providers, customers, electric consumer, and Green Cloud interested people.
- Using KVM is better at most time than Xen.
- If Xen was chosen to be used, then 3DES is preferred to be used among to time performance.
- Regardless with EA strength, CAST-128 and ARC4 are preferred with time response, while RSA and 3DES is the highest average response time respectively.
- Regardless with EA strength, ARC4 is preferred to use at KVM, while CAST-128 is preferred to use at both KVM and Xen, while RSA and 3DES are the highest CPU utilization at all.
- As a result, KVM is the optimal choice for cloud infrastructure, which it's agreed to the literature review.

5.3 Future Works

Through conducting this research, many ideas and issues were unfolded but not accomplished yet because of time, resources, and other constraints. We would like to suggest a few ideas for future study:

- 1- Interpret some response time results for RSA, TwoFish, and TripleDES.
- 2- Finding the RAM as performance parameter adding to CPU utilization and time response.
- 3- Study the performance (CPU & Time) when Enc\Dec concurrencies to other application such as ORACLE.
- 4- Study other Hypervisors performance like VMWARE and Hyper-V.
- 5- Study other encryption algorithms like CAST-256, ARC6, etc.
- 6- Study the performance of same key on different EA, for example key of 128 at CAST-128, TwoFish and AES for the same data size.
- 7- Using benchmark data to comparative the performance with our results.
- 8- Study the effect of data type on the performance.

References:

- Alam, M. I. (2013). A Comparative Analysis of Different Encryption Techniques of Cryptography. *International Journal of Advanced and Innovative Research* (2278-7844), 160.
- Arora, R., Parashar, A., & Transforming, C. C. I. (2013). Secure User Data in Cloud Computing Using Encryption Algorithms. *International Journal of Engineering Research and Applications (IJERA)*, 3(4), 1922-1926.
- Chierici, A., & Veraldi, R. (2010, April). A quantitative comparison between xen and kvm. In *Journal of Physics: Conference Series* (Vol. 219, No. 4, p. 042005). IOP Publishing.
- Curran, K., Carlin, S., & Adams, M. (2011). Security issues in cloud computing. *Elixir*, 38, 4069-72.
- Ercolani, G. (2013). Cloud Computing Services Potential Analysis. An integrated model for evaluating Software as a Service. *Cloud Computing*, 77-80.
- Fang, Z., Sun, Y., Sun, Y., & Yang, J. (2013, July). The Research of AES algorithm and application in cloud storage system. In *2nd International Conference on Science and Social Research (ICSSR 2013)*. Atlantis Press.
- Fayyad-Kazan, H., Perneel, L., & Timmerman, M. (2013). EVALUATING THE PERFORMANCE AND BEHAVIOUR OF RT-XEN. *International Journal of Embedded Systems & Applications*, 3(3).
- Gunasundari, T., & Elangovan, K. (2014). A Comparative Survey on Symmetric Key Encryption Algorithms.
- Hwang, J., Zeng, S., & Wood, T. (2013, May). A component-based performance comparison of four hypervisors. In *Integrated Network*

- Management (IM 2013), 2013 IFIP/IEEE International Symposium on* (pp. 269-276). IEEE.
- Jeon, S. H., & Gil, S. K. (2013). Optical implementation of triple DES algorithm based on dual XOR logic operations. *Journal of the Optical Society of Korea*, 17(5), 362-370.
 - Kolhe, S., & Dhage, S. (2012, October). Comparative study on Virtual Machine Monitors for cloud. In *Information and Communication Technologies (WICT), 2012 World Congress on* (pp. 425-430). IEEE.
 - Mell, P., & Grance, T. (2011). The NIST definition of cloud computing (draft). *NIST special publication*, 800(145), 7.
 - Prasanthi, O., & Reddy, M. S. (2012). RSA Algorithm Modular Multiplication. *International Journal of Computer Applications in Engineering Sciences*, 2(2).
 - Sabahi, F. (2012). Secure Virtualization for Cloud Environment Using Hypervisor-based Technology. *Int. Journal of Machine Learning and Computing*, 2(1).
 - Schlosser, D., Duelli, M., & Goll, S. (2011). Performance comparison of hardware virtualization platforms. In *NETWORKING 2011* (pp. 393-405). Springer Berlin Heidelberg.
 - Xu, X., Zhou, F., Wan, J., & Jiang, Y. (2008, December). Quantifying performance properties of virtual machine. In *Information Science and Engineering, 2008. ISISE'08. International Symposium on* (Vol. 1, pp. 24-28). IEEE.
 - Yang, C. T., Wang, S. F., Huang, K. L., & Liu, J. C. (2012). On construction

of cloud iaas for VM live migration using KVM and opennebula. In *Algorithms and Architectures for Parallel Processing* (pp. 225-234). Springer Berlin Heidelberg.

- Younge, A. J., Henschel, R., Brown, J. T., von Laszewski, G., Qiu, J., & Fox, G. C. (2011, July). Analysis of virtualization technologies for high performance computing environments. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on* (pp. 9-16). IEEE.

Links

- http://www.linux-kvm.org/page/Main_Page
- <http://www.techterms.com/definition/>
- <http://www.ubuntu.com/>
- <http://www.xenproject.org/>
- <https://www.debian.org/>
- <https://www.proxmox.com/>
- <http://www.advanceresearchlibrary.com/>

APPENDIX

1-RSA

CPU Dec %	Time Decr m.s	CPU Enc %	Time Encr m.s	File Kind	File size	Threads No	CPU Core No	Key Size	Hyp Type	Exp No
61	51.01	61	40.25	Pic	606k	1	2	64	Xen	1
60	29.22	59	26.23	Pic	606k	1	2	64	KVM	
54	23.10	56	14.45	Text	12.2k	1	2	1024	Xen	2
53	21.47	54	14.40	Text	12.2k	1	2	1024	KVM	
55	10.27	54	5.38	Text	1.0 k	1	2	2048	Xen	3
53	10.46	53	4.59	Text	1.0 k	1	2	2048	KVM	
89	26.37	100	15.48	Text	12.2k	2	2	1024	Xen	4
87	9.41	95	9.38	Text	12.2k	2	2	1024	KVM	
87	78.20	95	51.52	Pic	2.25M	2	2	1024	Xen	5
82	32.20	85	25.01	Pic	2.25M	2	2	1024	KVM	
96	7.53	96	4.09	Text	1.0 K	2	2	2048	Xen	6
94	7.00	94	3.52	Text	1.0 K	2	2	2048	KVM	
93	15.01	94	7.58	Text	2.0 K	2	2	2048	Xen	7
93	13.48	94	7.56	Text	2.0 K	2	2	2048	KVM	
64	3.13	64	2.32	Text	12.2k	1	2	256	Xen	8
55	2.07	61	2.05	Text	12.2k	1	2	256	KVM	
62	60.24	64	34.02	Text	12.2k	1	2	1536	Xen	9
52	60.02	56	35.48	Text	12.2k	1	2	1536	KVM	
56	106.45	52	67.15	Text	12.2k	1	2	2048	Xen	10
52	120.12	51	77.20	Text	12.2k	1	2	2048	KVM	

2-AES (Rijndael)

CPU utili %	Time Decr m:s	CPU utili %	Time Encr m:s	Data Type	File size	CPU Core No	Key Size	Hyp Type	Exp No
69	2.42	67	2.42	Windows7	2.09 G	2	128	Xen	1
55	2.42	60	2.35	Windows7	2.09 G	2	128	KVM	
71	2.57	67	2.57	Windows7	2.09 G	2	192	Xen	2
56	2.56	62	2.48	Windows7	2.09 G	2	192	KVM	
69	4.00	70	4.22	Compress pdf	2.68 G	2	256	Xen	3
64	3.15	63	3.49	Compress pdf	2.68 G	2	256	KVM	
67	3.25	63	3.25	Windows7	2.09 G	2	256	Xen	4
61	2.51	61	2.52	Windows7	2.09 G	2	256	KVM	
70	3.41	64	4.01	Compress pdf	2.68 G	2	128	Xen	5
59	3.40	56	3.44	Compress pdf	2.68 G	2	128	KVM	
40	2.50	34	3.05	Windows7	2.09 G	4	128	Xen	6
34	2.25	30	2.59	Windows7	2.09 G	4	128	KVM	
37	3.17	37	2.53	Windows7	2.09 G	4	192	Xen	7
34	2.41	33	2.50	Windows7	2.09 G	4	192	KVM	
38	3.27	37	3.02	Windows7	2.09 G	4	256	Xen	8
34	2.55	34	2.57	Windows7	2.09 G	4	256	KVM	
36	3.45	35	3.41	Compress pdf	2.68 G	4	128	Xen	9
32	3.24	30	3.40	Compress pdf	2.68 G	4	128	KVM	
32	1.10	34	1.06	Compress Win	800 M	4	256	Xen	10
32	1.02	32	1.05	Compress Win	800 M	4	256	KVM	
38	7.09	35	7.20	Compress fold	4.77 G	4	128	XEN	11
33	6.13	30	6.31	Compress fold	4.77 G	4	128	KVM	

3-DES

CPU utili %	Time Decr m:s	CPU utili %	Time Encr m:s	Data Type	File size	CPU Core No	Key Size	Hyp Type	Exp No
65	6.21	64	6.23	Windows7	2.09 G	2	64	XEN	1
57	6.15	61	6.20	Windows7	2.09 G	2	64	KVM	
37	5.51	37	5.54	Windows7	2.09 G	4	64	XEN	2
33	5.49	33	5.49	Windows7	2.09 G	4	64	KVM	
69	8.11	68	8.05	Compress pdf	2.68 G	2	64	XEN	3
55	7.50	55	7.45	Compress pdf	2.68 G	2	64	KVM	
63	3.55	64	3.55	Compress win	1.31 G	2	64	XEN	4
61	3.53	61	3.54	Compress win	1.31 G	2	64	KVM	
63	2.20	64	2.22	Compress win	800M	2	64	XEN	5
61	2.20	59	2.17	Compress win	800M	2	64	KVM	
66	15.38	65	16.04	Compress fold	4.77 G	2	64	XEN	6
64	13.52	62	14.14	Compress fold	4.77 G	2	64	KVM	
37	3.55	37	3.57	Compress win	1.31 G	4	64	XEN	7
32	3.47	31	3.57	Compress win	1.31 G	4	64	KVM	
35	2.14	37	2.20	Compress win	800 M	4	64	XEN	8
32	2.12	32	2.15	Compress win	800 M	4	64	KVM	
35	14.18	36	15.24	Compress fold	4.77 G	4	64	XEN	9
32	14.14	32	14.20	Compress fold	4.77 G	4	64	KVM	
37	6.05	35	6.16	Windows7	2.09 G	4	64	XEN	10
33	3.55	32	6.16	Windows7	2.09 G	4	64	KVM	

4-TripleDES

CPU utili %	Time Decr m:s	CPU utili %	Time Encr m:s	Data Type	File size	CPU Core No	Key Size	Hyp Type	Exp No
64	32.00	65	32.09	Compress fold	4.77 G	2	192	XEN	1
60	33.11	61	33.00	Compress fold	4.77 G	2	192	KVM	
62	13.47	62	14.03	Windows7	2.09 G	2	192	XEN	2
61	14.25	61	14.35	Windows7	2.09 G	2	192	KVM	
35	13.18	34	13.33	Windows7	2.09 G	4	192	XEN	3
31	13.57	32	13.57	Windows7	2.09 G	4	192	KVM	
62	8.42	61	8.48	Compress win	1.31 G	2	192	XEN	4
60	9.00	55	9.08	Compress win	1.31 G	2	192	KVM	
61	5.11	62	5.15	Compress win	800 M	2	192	XEN	5
60	5.23	60	5.23	Compress win	800 M	2	192	KVM	
35	5.00	35	5.02	Compress win	800 M	4	192	XEN	6
31	5.08	30	5.19	Compress win	800 M	4	192	KVM	
35	8.30	35	8.35	Compress win	1.31 G	4	192	XEN	7
32	8.45	31	8.45	Compress win	1.31 G	4	192	KVM	

5-ARC4

CPU utili %	Time Decr m:s	CPU utili %	Time Encr m:s	Data Type	File size	CPU Core No	Key Size	Hyp Type	Exp No
45	1.21	44	1.21	Windows 7	2.09 G	4	32	XEN	1
40	1.21	39	1.03	Windows 7	2.09 G	4	32	KVM	
78	1.44	77	1.42	Compress pdf	2.68 G	2	48	XEN	2
74	1.44	63	1.26	Compress pdf	2.68 G	2	48	KVM	
45	1.36	35	2.09	Compress pdf	2.68 G	4	48	XEN	3
38	1.23	32	2.09	Compress pdf	2.68 G	4	48	KVM	
62	2.15	67	2.15	Compress pdf	2.68 G	2	24	XEN	4
52	2.06	50	2.05	Compress pdf	2.68 G	2	24	KVM	
59	2.15	60	2.11	Compress pdf	2.68 G	2	32	XEN	5
50	2.10	50	2.01	Compress pdf	2.68 G	2	32	KVM	
63	4.00	60	4.08	Compress fold	4.77 G	2	24	XEN	6
51	3.41	50	3.54	Compress fold	4.77 G	2	24	KVM	
61	4.04	62	4.01	Compress fold	4.77 G	2	32	XEN	7
52	3.47	51	3.42	Compress fold	4.77 G	2	32	KVM	
65	4.05	64	4.13	Compress fold	4.77 G	2	48	XEN	8
50	3.47	52	3.50	Compress fold	4.77 G	2	48	KVM	
45	0.25	45	0.36	Compress win	800 M	2	24	XEN	9
38	0.23	40	0.35	Compress win	800 M	2	24	KVM	
34	4.12	32	4.18	Compress fold	4.77 G	4	24	XEN	10
28	3.42	28	3.56	Compress fold	4.77 G	4	24	KVM	
35	4.22	35	4.19	Compress fold	4.77 G	4	32	XEN	11
28	3.56	28	3.56	Compress fold	4.77 G	4	32	KVM	
34	4.15	34	4.22	Compress fold	4.77 G	4	48	XEN	12
28	4.01	28	3.49	Compress fold	4.77 G	4	48	KVM	
26	0.26	26	0.38	Compress win	800 M	4	24	XEN	13
24	0.22	24	0.37	Compress win	800 M	4	24	KVM	

6-CAST-128

CPU utili %	Time Decr m:s	CPU utili %	Time Encr m:s	Data Type	File size	CPU Core No	Key Size	Hyp Type	Exp No
40	1.50	39	1.52	Windows 7	2.09 G	4	128	XEN	1
37	1.43	30	1.38	windows 7	2.09 G	4	128	KVM	
37	2.24	39	2.33	Compress pdf	2.68 G	4	128	XEN	2
35	2.23	37	2.02	Compress pdf	2.68 G	4	128	KVM	
66	2.29	65	2.40	Compress pdf	2.68 G	2	128	XEN	3
54	2.27	55	2.29	Compress pdf	2.68 G	2	128	KVM	
71	5.00	67	5.00	Compress fold	4.77 G	2	128	XEN	4
57	4.56	55	4.36	Compress fold	4.77 G	2	128	KVM	
65	1.12	54	1.27	Compress win	1.31 G	2	128	XEN	5
60	1.01	51	1.20	Compress win	1.31 G	2	128	KVM	
50	0.35	50	0.37	Compress win	800 M	2	128	XEN	6
39	0.30	45	0.32	Compress win	800 M	2	128	KVM	
36	3.32	35	4.28	Compress fold	4.77 G	4	128	XEN	7
28	3.30	27	4.24	Compress fold	4.77 G	4	128	KVM	
32	1.30	34	1.35	Compress win	1.31 G	4	128	XEN	8
28	0.55	31	0.58	Compress win	1.31 G	4	128	KVM	
32	0.41	34	0.55	Compress win	800 M	4	128	XEN	9
31	0.40	32	0.46	Compress win	800 M	4	128	KVM	

7-BlowFish

CPU utili %	Time Decr m:s	CPU utili %	Time Encr m:s	Data Type	File size	CPU Core No	Key Size	Hyp Type	Exp No
34	6.00	35	6.00	Windows 7	2.09 G	4	64	XEN	1
34	5.24	31	5.51	Windows 7	2.09 G	4	64	KVM	
36	8.15	36	7.50	Compress pdf	2.68 G	4	64	XEN	2
34	6.35	32	7.15	Compress pdf	2.68 G	4	64	KVM	
67	7.38	66	7.36	Compress pdf	2.68 G	2	64	XEN	3
61	7.38	60	7.35	Compress pdf	2.68 G	2	64	KVM	
67	5.53	64	6.15	Windows 7	2.09 G	2	64	XEN	4
61	5.37	59	5.51	Windows 7	2.09 G	2	64	KVM	
67	14.01	67	14.24	Compress fold	4.77 G	2	64	XEN	5
61	13.03	60	13.27	Compress fold	4.77 G	2	64	KVM	
65	3.40	66	5.17	Compress win	1.31 G	2	64	XEN	6
61	3.30	59	4.22	Compress win	1.31 G	2	64	KVM	
67	2.15	66	2.21	Compress win	800 M	2	64	XEN	7
60	2.05	60	2.11	Compress win	800 M	2	64	KVM	
36	14.56	34	14.59	Compress fold	4.77 G	4	64	XEN	8
33	12.40	32	12.45	Compress fold	4.77 G	4	64	KVM	
36	3.59	36	4.07	Compress win	1.31 G	4	64	XEN	9
32	3.21	33	3.37	Compress win	1.31 G	4	64	KVM	
40	2.26	35	2.24	Compress win	800 M	4	64	XEN	10
32	2.00	31	2.01	Compress win	800 M	4	64	KVM	

8-TwoFish

CPU utili %	Time Decr m:s	CPU utili %	Time Encr m:s	Data Type	File size	CPU Core No	Key Size	Hyp Type	Exp No
36	11.07	35	11.21	Compress pdf	2.68 G	4	128	XEN	1
33	11.06	32	11.21	Compress pdf	2.68 G	4	128	KVM	
64	12.54	65	12.50	Compress pdf	2.68 G	2	128	XEN	2
61	11.39	61	11.50	Compress pdf	2.68 G	2	128	KVM	
36	9.59	36	9.48	Windows7	2.09	4	256	XEN	3
32	9.24	32	9.36	Windows7	2.09G	4	256	KVM	
34	9.15	34	9.21	Windows7	2.09G	4	192	XEN	4
32	8.52	32	9.01	Windows7	2.09G	4	192	KVM	
64	12.30	63	12.43	Compress pdf	2.68 G	2	192	XEN	5
61	12.13	59	12.13	Compress pdf	2.68 G	2	192	KVM	
66	3.08	65	3.04	Compress win	800 M	2	128	XEN	6
64	3.15	61	3.29	Compress win	800 M	2	128	KVM	
65	3.21	64	3.19	Compress win	800 M	2	192	XEN	7
64	3.22	61	3.23	Compress win	800 M	2	192	KVM	
64	3.29	65	3.29	Compress win	800 M	2	256	XEN	8
64	3.32	62	3.31	Compress win	800 M	2	256	KVM	
35	3.12	36	3.15	Compress win	800 M	4	128	XEN	9
32	3.13	32	3.16	Compress win	800 M	4	128	KVM	
38	3.19	35	3.27	Compress win	800 M	4	192	XEN	10
36	3.22	32	3.29	Compress win	800 M	4	192	KVM	
35	3.33	35	3.35	Compress win	800 M	4	256	XEN	11
32	3.33	32	3.38	Compress win	800 M	4	256	KVM	