Performance Evaluation of Virtualization with Cloud Computing

Christophe PELLETINGEAS

07013760

Submitted in partial fulfilment of
the requirements of Edinburgh Napier University
for the Degree of
MSc Advanced Networking

School of Computing

December 2010

Authorship Declaration

I, Christophe PELLETINGEAS, confirm that this dissertation and the work presented in it are my own achievement.

Where I have consulted the published work of others this is always clearly attributed;

Where I have quoted from the work of others the source is always given. With the exception of such quotations this dissertation is entirely my own work;

I have acknowledged all main sources of help;

If my research follows on from previous work or is part of a larger collaborative research project I have made clear exactly what was done by others and what I have contributed myself;

I have read and understand the penalties associated with Academic Misconduct.

I also confirm that I have obtained **informed consent** from all people I have involved in the work in this dissertation following the School's ethical guidelines

Signed:

Date:

Matriculation no: 07013760

Data Protection Declaration

Under the 1998 Data Protection Act, The University cannot disclose your grade to an unauthorised person. However, other students benefit from studying dissertations that have their grades attached.

Please sign your name below *one* of the options below to state your preference.

The University may make this dissertation, with indicative grade, available to others.

The University may make this dissertation available to others, but the grade may not be disclosed.

The University may not make this dissertation available to others.

Abstract

Cloud computing has been the subject of many researches. Researches shows that cloud computing permit to reduce hardware cost, reduce the energy consumption and allow a more efficient use of servers. Nowadays lot of servers are used inefficiently because they are underutilized. The uses of cloud computing associate to virtualization have been a solution to the underutilisation of those servers. However the virtualization performances with cloud computing cannot offers performances equal to the native performances.

The aim of this project was to study the performances of the virtualization with cloud computing. To be able to meet this aim it has been review at first the previous researches on this area. It has been outline the different types of cloud toolkit as well as the different ways available to virtualize machines. In addition to that it has been examined open source solutions available to implement a private cloud. The findings of the literature review have been used to realize the design of the different experiments and also in the choice the tools used to implement a private cloud. In the design and the implementation it has been setup experiment to evaluate the performances of public and private cloud.

The results obtains through those experiments have outline the performances of public cloud and shows that the virtualization of Linux gives better performances than the virtualization of Windows. This is explained by the fact that Linux is using paravitualization while Windows is using HVM. The evaluation of performances on the private cloud has permitted the comparison of native performance with paravirtualization and HVM. It has been seen that paravirtualization has performances really close to the native performances contrary to HVM. Finally it has been presented the cost of the different solutions and their advantages.

Contents

List of Figures

Acknowledgements

I wanted to thanks the Professor Bill Buchanan for his support, his supervision and his time. He manages to give me some credit on the Amazon cloud which has permitted me to realize my tests on it. Also I wanted to thanks Sofyane Khedim for providing me the hardware to be able to implement a private cloud and his help and support on my project. I would also like to thank Italo Madalozo for his help in the implementation of the private cloud. Additionally I want to thanks Jim Jackson for being part of the marking process.

1 Introduction

1.1 Context

Within Information Technologies (IT) everything is moving very fast and technologies progress in an exponential manner according to Moore's law: “The complexity for minimum component costs has increased at a rate of roughly a factor of two per year... Certainly over the short term this rate can be expected to continue, if not to increase” (Moore, 1965). It has been reported that a large majority of server are underutilised. Virtualization and also cloud computing has become the solution to increase the efficiency of servers utilisation. Dell has conducted some researches and has found that three quarters of the servers that they were managing globally were not exceeded 20 percent of processor utilisation. The processor utilisation of those servers low because only one or two applications were running on those servers (Brooks, 2008). To overcome the problem of under utilisation of server’s solutions such as virtualization and cloud computing have been deployed.

1.2 Background

Cloud computing regroups the use of different technologies such as virtualization, clustering, and so on. This technology begins to become really popular because it offers to the customers the choice to pay only for what they use. If a company needs an important computing power for few days they can buy it for few days on the cloud rather than to by machine especially for that purpose. Cloud computing has permitted to optimize server utilisation due to the use of virtualization. And with the apparition of processor with multiple cores and virtualization support few years ago virtualization has gain popularity.

Cloud computing is an on demand service able to share resources easily over the internet. It exists different models of cloud that can be used for different usages (Amrhein et al, 2009):

- Public cloud – Manages by a third party service provider to offer services accessible by clients through the internet.
- Private cloud – It is basically the same as public cloud but inside the company network. It loses some benefits of public cloud but it has the advantages to be more secure.
- Hybrid cloud – It is a combination of public and private cloud.
- Community cloud – It is a cloud shared by multiple organisations.

All of those models can implement virtualization. For implementing those different models multiple solutions exists. Eucalyptus and OpenNebula are open source solutions available for implementing cloud computing. The different solutions implementing cloud computing don't have the same structures (Peng, 2009). It can be though that the management of the virtualization is different between the different solutions.

Cloud computing doesn't necessarily use virtualization. Different virtualization solution can be implemented within a cloud. The most popular solutions are Xen, Vmware and KVM. Xen and KVM are two open source solutions for virtualization contrary to Vmware which is a proprietary solution. However the utilization of virtualization within cloud computing provide the virtualization and sharing of physical resources within the cloud allow higher utilisation rates and reduce in the same time the hardware investment, save space and reduce power consumption (Kamoun, 2009). The usage of virtualization introduces a degradation of performances because it introduces additional overhead. Virtualization affects CPU usage, network, memory and storage performances as well as applications performances (Armstrong, 2007). Within virtualization great performances depend essentially of the tasks scheduling and the workload on the system. Kim et al (2009) experiment how to improve I/O performances by the scheduling of virtual machine tasks. It can be found at the fifth place “performance Unpredictability”. It is explained in the article that shared CPU and memory are pretty well managed by virtual machines rather than network and disk I/O sharing which are not. Virtualization in the cloud can meet some I/O interference between virtual machines. The scheduling tasks is an important part of the management of virtualization in cloud computing.

1.3 Aim and objectives

The aim of this project is to evaluate the performance of virtualization in cloud computing environment in terms of CPU, memory and I/O devices performances. This evaluation will outline the performance that a customer can expect for public and private cloud. To be able to meet this it has been define four objectives:

- Critically review the different solutions available for cloud computing. Then it will be study the different virtualisation solutions both Open-Source and commercial that can be used with cloud computing. Finally it will be review the study about the performances of the different hypervisor.
- Design scenarios to be able to measure the performances of both public and private cloud. Also to be able to test the performance of a private cloud it will be required to design the implementation of a private cloud.
- Implement the different scenario to be able to evaluate the performances of public and private cloud as well as the implementation of the private cloud.
- Evaluate the performance of public and private cloud to be able to give an idea of the performance that can expect a customer of each service.

1.4 Thesis structure

Chapter 1 – Introduction: This chapter provides an overview of the project presented the aims and objective to meet.

Chapter 2 – Technology review: This chapter has for aim to review the different technology present in virtualization and cloud computing. It presents the theoretical aspect of virtualization and cloud computing

Chapter 3 – Literature review: This chapter provide a critical analysis on the different virtualization technology available. It also does a critical analysis of the different cloud toolkit available for the cloud. It criticizes commercial solution as well as open source solutions.

Chapter 4 – Design: This chapter describe the design of the tests performed to measure virtualization performance. It also describes the design of the implementation of the private cloud.

Chapter 5 – Implementation: The chapter gives the details of the implementation of the tests. In addition to that it presents the implementation of the private cloud.

Chapter 6 – Evaluation: This chapter evaluate the results obtain during the implementation of the different experiments.

Chapter 7 – Conclusion: The chapter summarise the entire project. Also it gives a critical point of view and some recommendations for future researches.

2 Technologies review

2.1 Introduction

In this part it will be outline the main concept associate to cloud computing. For being able to reduce hardware cost, cloud computing uses virtualization. Virtualization technology has evolved really quickly during those past few years. Also it is particularly due to hardware progresses made by AMD and Intel. Both processors constructor have permit to improve considerably virtualisation of the Virtual Machine Monitor of type I by bringing modification to their processors. Virtualization is generally associated to cloud computing. In the following it will be presented the main evolution of processor which have permit to virtualization to be considerably improved. Then it will be outline the different types of virtualization. Finally it will be presented the general architecture of the cloud as well as the different types of deployment available.

2.2 Architecture of x86 processors supporting Virtualization

The x86 processor family has been design with 4 levels of privileges. Those levels are called rings. Ring 0 represent the most privileged and the ring 3 represent the least privileged (Intel Corporation, 1986). Ring 0 contains the most critical software which is generally the kernel of the operating system. The rest of the rings are less privileged rings. The utilisation of all the levels is not a mandatory. Also most of the time only ring 0 and ring 3 are used (Figure 1).

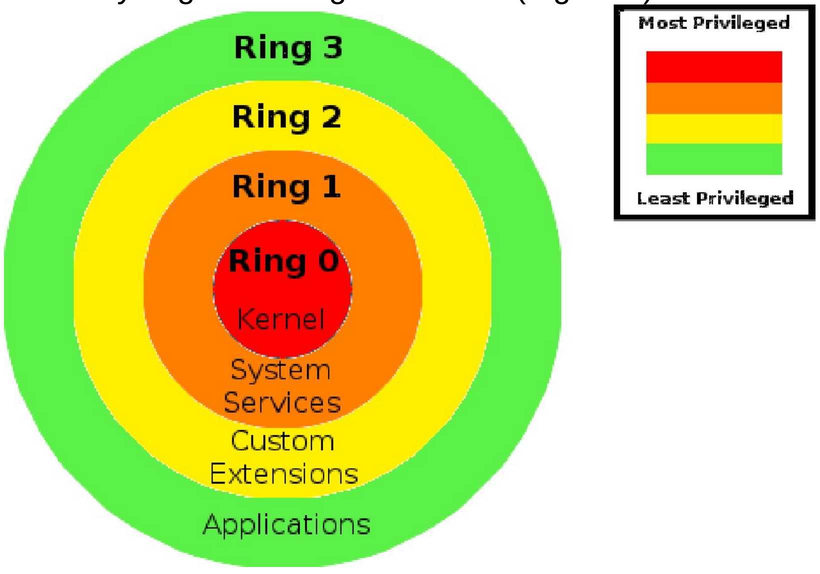


Figure 1 - Architecture of x86 processor

x86 processors has been built without support for visualization. In 2005 Intel and AMD have begun to modify the architecture of their x86 processors with new features to have a real support for virtualization technologies. With virtualization the hypervisor is installed on ring 0 and has the most privileged level. Also the OS will run at ring 1 because ring 0 is occupied by the hypervisor. To have the OS running (without having to modify the OS) on ring 1 INTEL and AMD had to modify x86 processors. Intel-VT and AMD-V processors have been build to support virtualization due to this operating system can run at ring 1 without any modification. When it comes to virtualization with VMM type II no problems about CPU architecture occurs because everything is emulated.

2.3 Different types of virtualization

It is IBM which has invented the concept of virtual machine (Creasy, 1981; Goldberg, 1974). IBM has first defined virtual machine as an isolated and fully protected copy of the physical hardware of machine (Sugerman et al, 2001). Thus virtualization refers to the virtual emulation of computing elements such as hardware, memory, storage, software, network and so on. Virtualization brings security by virtualizing software such as operating system because it permits to run the Operating System (OS) into a sandbox. Everything that happens in the sandbox has no consequences on the environment that host the virtual OS.
It improves the management of the system with no extra cost (Sugerman et al, 2001). For example the hard drive can be separate in different partition there is no need to buy multiple hard drive to achieve the same goal. The principal aims of virtualization are to increase the hardware utilisation to a maximum, decrease hardware costs by regrouping multiple machines virtualized into one physical machine, reduce energy consumption and simplify security and system management. To virtualize the hardware of the machine the use of a Virtual Machine Monitor (VMM) is required. There are three main techniques that can be used to virtualize a guest OS:

- Full virtualization
- Hardware-assisted virtualization
- Paravirtualization

2.3.1 Full virtualization

The challenge with full virtualization was to virtualize guest Operating System (OS) without any modification. Guest OS in fully virtualized environment are provided with all the services provided by physical systems which include a virtual BIOS, virtual devices as well as virtualized memory (VMware, 2007). Within this kind of virtualization the OS is not aware of being virtualized that is the reason why it does not require modification. Because the VMM runs on ring 0 (sometime the VMM can be shared with an OS in the case of VMM type II) the guest OS will run on ring 1. To allow the guest OS to run without any modification a technique called binary translation of OS requests is used (Figure 2).

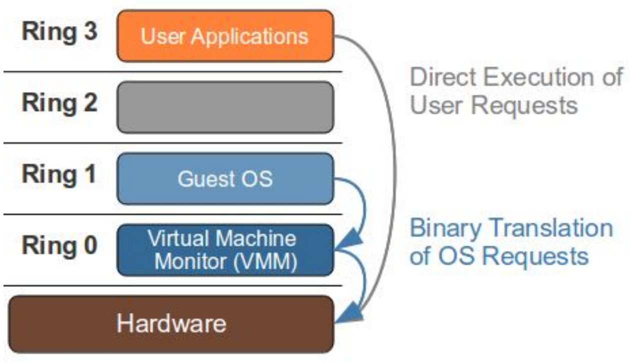

Ring 3
User Applications
Ring 2
Ring 1
Guest OS
Ring 0
Virtual Machine Monitor (VMM)
Hardware
Direct Execution of User Requests
Binary Translation of OS Requests

Figure 2 - Operation performed to implement full virtualization at the kernel level

2.3.2 Paravirtualization

Paravirtualization is a technique used to virtualize a guest OS to allow better performance than full virtualization or hardware-assisted virtualization. With paravirtualization the guest OS is able to communicate with the hypervisor. To allow the guest OS to communicate with the hypervisor modifications of the guest OS are required to translate non-virtualizable instruction with hypercalls (Figure 3). Hypercalls are instructions that are able to communicate directly with the virtualization layer. Modification of the guest OS involves low compatibility and portability problems (VMware, 2007).

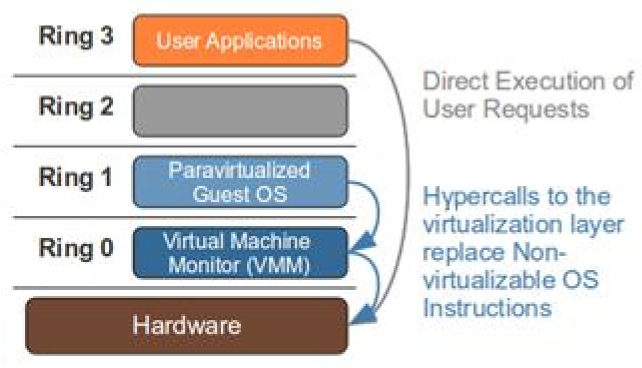

Ring 3
User Applications
Ring 2
Ring 1
Paravirtualized Guest OS
Ring 0
Virtual Machine Monitor (VMM)
Hardware
Direct Execution of User Requests
Hypercalls to the virtualization layer replace Non-virtualizable OS Instructions

Figure 3 - Operation performed to implement para-virtualization virtualization at the kernel level

2.3.3 Hardware-assisted virtualization

Hardware-assisted virtualization has been developed to overcome the drawback of paravirtualization due to hardware modification that allow the guest OS to communicate with the VMM without modifications. Hardware types supporting this kind of virtualization are Intel Virtualization technology (Intel VT) and AMD virtualization (AMD-V). Those processors have features that trap OS requests that come from ring 1 (Figure 4). Due to that ring 0 is transformed as a virtual ring -1 and ring 1 is a virtual ring 0 where the guest OS can operate without any modifications (VMware, 2007).

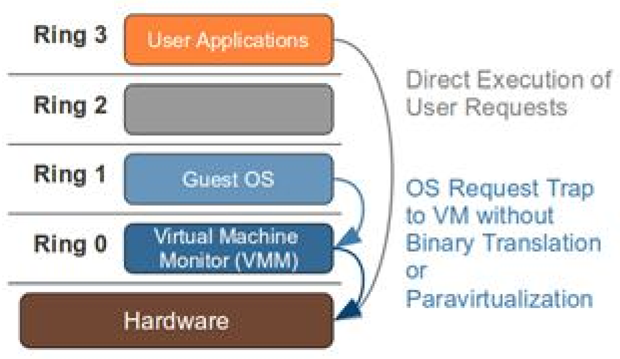

Ring 3
User Applications
Ring 2
Ring 1
Guest OS
Ring 0
Virtual Machine Monitor (VMM)
Hardware
Direct Execution of User Requests
OS Request Trap to VM without Binary Translation or Paravirtualization

Figure 4 - Operation performed to implement hardware-assisted virtualization at the kernel level

2.4 Architecture of Cloud Computing

Cloud computing provide three main service models which meet different consumer requirement. The three services can be seen as a pyramidal model (Figure 5) with at the bottom the Infrastructure as a Service which provide the all infrastructure that will be manage by the consumer. In the middle the Platform as a Service which provide the infrastructure as well as the platform to the consumer. So the consumers don't have to deal with the infrastructure and can manage the platform. Finally on the top it can be found the Software as a Service which provide to the consumer a software where the consumer as no need to worry about the infrastructure or the platform which are managed by the provider.

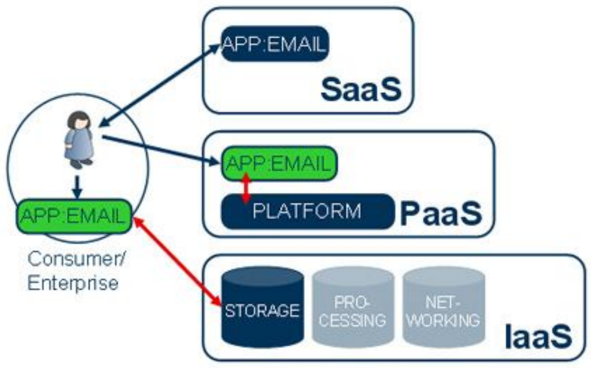


Figure 5 - Architecture of the cloud computing with and Email application example. (Gabrielsson et al, 2009)

The Infrastructure as a Service (IaaS) allows consumers to provision resources of the cloud such as storage, network, processing, memory and other fundamental computing resources such as operating system and applications. The consumer has no control over the underlying cloud infrastructure however he is able to control operating system, storage, memory, processing, as well as application deployment and also a selected number network component such as host firewall.

Platform as a Service (PaaS) provides to the consumer the ability to deploy applications onto the cloud infrastructure. It gives the ability to provide a platform such as development platform or any other platform. In that configuration consumer has control over the deployed applications and sometimes application hosting environment configurations. However consumer has no control over the infrastructure on which rely the applications.

Software as a Service (SaaS) provides to consumers applications which run on the cloud. Due to that consumers of the hosted applications of the cloud have no need to worry about the infrastructure or the platform. Applications provided as SaaS can be accessed through many devices by a web browser. An example of Software as a service can be a web mail client. (Figure 7)

2.5 Deployment of Cloud Computing

Different deployment of the cloud can be performed depending of the services that want to be provided. It can be differentiate four types of deployment depending of the requirements:

- Public cloud: This cloud infrastructure is made for the public. The resources can be access by companies as well as private customers. It makes the resources of the cloud available for anybody which wants to consume resources from it. To access resources provided by a public cloud the customers need to register with a cloud provider such as Amazon Web Services (AWS).

- Private cloud: Private clouds are different from public cloud because it is deployed behind the firewall of the company also the infrastructure of the cloud is not managed by a third party. This solution is adopted by companies which want to manage their cloud and keep the control over their data. It is usually more secure than public cloud because it is hosted by the company and thus it is located behind the company firewalls.
- Community cloud: With this configuration the cloud is shared between different organisations. It is used to support communities which share the same concerns. The infrastructure of community cloud can be managed by the organisations or a third party.
- Hybrid cloud: The infrastructure of hybrid cloud is composed of two or more clouds (public cloud, private cloud or community cloud). Each cloud remains unique entities which are linked all together. This solution allows the portability of data and applications.

2.6 Conclusion

The development of virtualization have been an important factor in the development of cloud computing also this have been permitted by the development of new x86 processors which come with support for virtualization. Para-virtualization is able to give the best performances because it is the type of virtualization with the least overhead. However with this type of virtualization it is required to modify the guest OS system to be able perform virtualization. Also the use of para-virtualization is often completed with hardware assisted virtualization to virtualize guest Operating System (OS) such as Windows. Cloud computing can be provided a different levels such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS). The services provided can be deployed on different type of cloud such as public cloud, private cloud, community cloud and hybrid cloud.

3 Literature review

3.1 Introduction

Cloud computing is a recent technology and a lot of research are made in that domain to improve it. Also due to the relation between cloud and virtualization there are as well many researches on virtualization to enhance virtualization performances. Cloud computing is more and more popular and most of the enterprise begin to adopt it. However there are still some obstacles which can restrained the adoption of cloud services by enterprise such as the lack of standardisation, reliability associate to the cloud, the security and so on. The reason of the adoption of cloud computing by enterprise is principally for economical reasons because cloud computing allow customers to reduce their hardware cost as well as energy consumption and so on. Also there is no waste because customers only pay for what they are using. Because of the popularity of cloud computing a large number of cloud toolkits begin to appear such as OpenNebula, Eucalyptus and so on. Also those cloud toolkits are able to associate with different hypervisors such as Xen, KVM and VMware ESX.

As seen previously there are many different type of virtualization. To be able to provide the best performances cloud computing is using para-virtualization as well as hardware-assisted virtualization. Full virtualization is not used in cloud computing due to poor performances cause by its considerable overhead. Virtualization technology is not a new technology however it has regain popularity in 2005 with the apparition of AMD and Intel processors which had support for virtualization. Virtualization brings many advantages such as the improvement of security, the enhancement of the efficiency of server utilisation and so on. Also during the past few years due to the popularity of virtualization and its utilisation in the cloud computing many researches have been made. From those research lot of improvement have been made to try to obtain performances near to native performances.

In the following it will be outline in a first part the research that have been made on cloud computing. It will be explained the problems due to the lack of standardisation in the cloud as well as the economic impact that can have the usage of the cloud and the majors obstacles that can be encounter to the growth of cloud computing. Finally it will be compare different solutions of cloud toolkits. In a second part it will be compare the two different type of Virtual Machine Monitor (VMM). Then it will also be compare the native performances and virtual performances. It will end with a comparison of the different hypervisor for both commercial and open source product.

3.2 Cloud computing

3.2.1 Standardisation of cloud computing

Cloud computing is a new technology and evolve rapidly also it is difficult to match a good definition of cloud computing. Because cloud computing is an evolving

technology the definition is changes over the time. The U.S. Government's National Institute of Standards and Technology (NIST) tries to give an up to date definition of the cloud computing. The actual version of their definition is the version 15 in date of 10 July 2009 (Mell et al, 2009).
According to the NIST cloud computing is on demand service which shares a pool of computer resources over a network. Cloud computing matches five essential characteristics which define the main functionalities provided by the cloud, three service models which give the level of service provided and four deployment models which indicate where the cloud is deployed and who can access to it.

The main characteristics of the clouds are the following (Mell et al, 2009):

- On demand self-service: Users of the cloud can manage the resources in on demand basis and they only paid for what they consume.
- Broad network access: The resources provided by the cloud can be access by as any normal services through thin or thick clients such as laptop, PDA, mobile phones and so on.
- Resource pooling: The cloud provider serves pool of resources over multiple customers according to the demand. Client which access the service have no knowledge of the exact location of the cloud but may be able to provide a location at higher abstraction level such as country, state, datacenter and so on.
- Rapid elasticity: The resources provided by the cloud are highly scalable. Customer can rapidly scale up the resources that they need and then scale them down if there is no need to use it anymore. The scalability of the cloud gives a real modularity to the cloud. Also resources appear as infinite and customers have no need to make plan for provisioning (Armbrust et al, 2010).
- Measured service: The resources provided by the cloud are controlled and optimized according to the resources capabilities. Also resources usage can be monitored control and reported to be able to provide transparency for both provider and consumer of the resources.

Because of the diversity of the cloud toolkits available and also the diversity of hypervisor there is an important problem of interoperability. The open cloud manifesto tries to establish a set of core principles for cloud providers (Open Cloud Manifesto, 2009). This manifesto is supported by over 300 companies including Novell, VMware, AMD, IBM, etc. Also the number of supporters is still growing. There are six principles that are defined by the cloud manifesto which are:

- Cloud providers must work together
- Cloud providers must use and adopt the existing standards
- New standards need to promote innovation
- Community effort should be driven by customer needs
- Cloud computing standard organisations should work together
- Cloud provider must not use their popularity to lock the customer to their platform

Another group called Cloud Computing Use Case group also work to brought together cloud providers and cloud consumer. The aim of this group is to define common use case scenario for cloud computing (Ahronovitz et Al, 2010). Those groups which tries to uniform cloud computing are not adopted by everybody and company such as Amazon does not want to adopt them yet because Amazon locks users to their services considering the fact that migration over another cloud provider is not easy (Buyya et al, 2009).

3.2.2 Economical impact of cloud computing

From an economical point of view cloud computing is really useful because it is an on demand service. Also it permits to improve significantly the efficiency of server usage due to the utilisation of virtualisation. A great benefit of using virtualization is to reduce hardware cost because multiple machines are running on the same hardware. Virtualization can be used to reduce cost of datacenter, testing platform and so on. Weltzin et al show for example how the utilisation of the virtualization can help to reduce the cost of the test (Weltzin, 2009). According to Weltzin et al in test application virtualization can be useful for:

- Produce real-time test in parallel within a user interface
- Providing database logging capability while executing real-time tests
- Running multiple instances of the same OS allows the isolation of applications
- Software that have been written for different OSs can work on the same computer

Also virtualization helps to improve the efficiency by optimizing the utilisation of servers. Thus for reduce cost and avoid the creation of new datacenter company such as Dell have tried to find solution for having more efficiency (Brooks, 2008). Dell has conducted some researches and has found that three quarters of the servers that they were managing globally were not exceeded 20 percent of processor utilisation. The processor utilisation of those servers low because only one or two applications were running on those servers (Brooks, 2008). Due to this evaluation it has been concluded that the resources could be used more efficiently and intelligently. In this case virtualize servers that were not exceeded 20 percent of processor utilisation on one server would permit to increase the efficiency of processors utilisation. Rather to have 5 servers running at a maximum of 20 percent of processing we could have one server running five virtual servers. By this virtualization allows efficiency and cost reduction by reducing the number of physical servers. Also it involves a reduction of cost because it requires less space to store the servers and also a reduction of power consumption. For Dell virtualization has been really helpful because it has permitted to reduce the deployment of applications by 90 percent passing of a deployment of 45 days to only 4 days (Brooks, 2008).

The virtualization and the improvement of the efficiency of server utilisation is not the only advantages associate to cloud computing. Also because cloud computing is an on demand service it can be use for example for a datacenter which encounter peak load few days per month. In that case the datacenter can choose to buy resources to a cloud provider for provisioning peak load. This solution can be much cheaper than buying new hardware which will be underutilized most of the time (Armbrust et al, 2010). Another example can be an unknown demand which can be present with for example a website. If a website become rapidly popular it can occurs a high demand but this demand can sometimes be temporary and see a reduction of the demand if user turn away. So it can be useful to use to resources of the cloud in that case because the resource is available directly and there is no installation and a minimum configuration required (Armbrust et al, 2010). Finally if a company needs a important computation performance for a short period of time it is sometime better to buy this resource on the cloud rather than buying a server farm to be able to perform the computation task in a short period of time (Armbrust et al, 2010). Cloud computing allows customers to reduce the cost of the hardware by allowing resources on demand. Also customers of the service need to have guaranty of the good functioning of the service provided by the cloud. The Service Level Agreement brokered between the providers of cloud and the customers is the guarantees from the provider that the service will be delivered properly (Buyyaa et al, 2009).

3.2.3 Obstacles and opportunities associate to cloud computing

M. Armbrust et al. (April 2010) have made a list of the top 10 obstacles to and opportunities for growth of cloud computing. To each obstacle is associate an opportunity that overcome the obstacle. Obstacles are divided in three parts:

- Obstacles affecting the adoption of the cloud
- Obstacles affecting the growth of the cloud
- Business and policy obstacles

When it comes to the adoption of new services such as cloud computing enterprise are firstly worried about the business continuity as well as the availability of the service. Enterprise needs reliable service with high availability. However the problem with services having high reliability even small interruption of the service can become major news (Helft, 2009). The reliability and availability of cloud provider is regrouped in the Service Level Agreement (SLA). The SLA is a service contract negotiated between the customer and the service provider. For example Amazon assures an Annual Uptime Percentage of 99.95% for the service year. If the Annual Uptime Percentage drops below 99.95% Amazon engages itself to compensate customers (Amazon, 2010). Because the availability of a service can never be guaranty at 100% it is better to do not rely on only one service to avoid a single point of failure. The solution to obtain very high availability is to have multiple cloud computing providers in different geographic regions. The interoperability between cloud provider can be a major drawback to do not adopt cloud computing. Lock-in customer can be attractive for cloud provider but the fact that data or application will not be compatible from one cloud to another can be a major problem for companies. The solution to avoid data lock-in would be to standardize the cloud. The cloud manifesto tries to make to cloud uniform it by establishing a set of core principles for cloud providers (Open Cloud Manifesto, 2009). Finally concerning the obstacle affecting the adoption of the cloud there is the concern about of the security of the data. Security in the cloud needs to asset many areas such as privacy, data integrity, recovery and evaluation of legal issues (Brodkin, 2008).

Technical issues can be an obstacle to the growth of cloud computing. Armbrust et al. (April 2010) list five technical obstacles to the growth of cloud computing:

- Data transfer bottlenecks
- Performance Unpredictability
- Scalable Storage
- Scaling Quickly
- Bugs in Large-Scale Distributed System

Data transfer requires a large bandwidth also according to A. Fox it would take 45 days at 20 megabytes/sec to transfer 10 terabytes from UC Berkeley to Amazon in Seattle, Washington (Fox, 2010). In addition to that comes the expensive price which is of 100$ to 150$ per TB transferred over the cloud. Also it has been found that it was faster and cheaper to FexEx disks rather than using the network to transfer large quantity of information. Even if the performances concerning CPU and memory are almost equal to native performances cloud computing encounter more problems with I/O performances. To overcome this problems hardware improvement needs to be made. Another problem is the scalability of the storage. The storage need to be able to scale up and down (Greenberg et al, 2009). Also the scaling needs to be fast to respond to customers need. Because of the large scale application of the cloud when a bug occur it is really difficult to remove the errors.

The final obstacles to cloud computing are applied to policy and business. The bad usage that can make some people on the cloud can affect other users of the service. For example EC2 IP addresses can be used to make spam which can result a blacklist of those IP addresses (Krebs, 2008). It comes then the question of the legal issue of what people can do and cannot do on the cloud. Finally problems about software licensing come up. On the cloud users have to pay for the software as well as an annual maintenance fees (Armbrust et al, 2010). To overcome the licensing problem lot of open source solution are used.

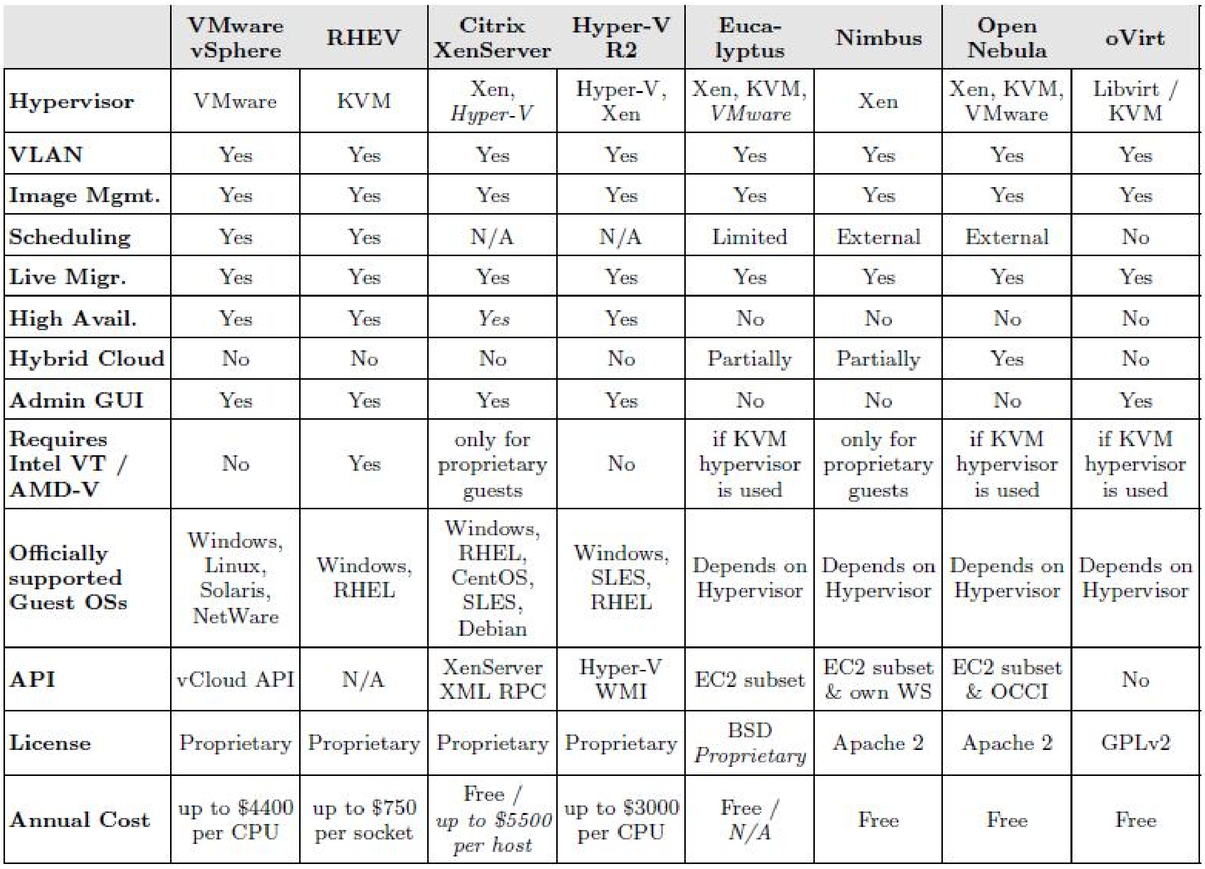
3.2.4 Comparison of cloud toolkits

3.2.4.1 Presentation of the cloud toolkits

There are several toolkits that can be used to implement a private cloud. It exists commercial solutions as well as open source solutions. Among the open-source solutions there are OpenNebula, Eucalyptus, Nimbus, and so on. The commercial solutions are Hyper-V, VMware ESX, etc. The table below (Figure 6) compares the different popular toolkit that can be used to implement a cloud. Commercial as well as open-source solutions are compared. It can be observed that the open-source solution such as Eucalyptus, Nimbus, OpenNebula and oVirt seem to have more flexibility than the commercial ones. However open-source solutions suffer of a lack of documentation and are more difficult to implement. Commercial solutions offer services which are better than open-source solution because there are beneficiating of a good support. However open-source solutions are developing very quickly and tend to offers services almost as good as the commercial one. It occurs that the main advantage of commercial solutions over open-source solution is the support.

3.2.4.2 Open-source solution hypervisor

A comparison of the different characteristics and the main architectures proposed by open source solutions is made in the article “A Survey on Open-Source Cloud Computing Solution” (Endo, 2010). The Xen Cloud Platform (XCP) is a solution for infrastructure virtualization that use Xen has the hypervisor. The XCP is different from the other platforms because according to the article it does not provide the entire architecture for cloud services. However it provides a tool to deal with automatic configuration and also maintenance of the cloud platform. The architecture of the XCP (Figure 7) is composed of a master XCP host, one or more XCP host and a shared storage. The Master XCP host manages the XCP host and offers an administration interface. XCP host as their name indicate host the virtual machine which are organised in XCP resource pool and used shared storage.



	VMware vSphere	RHEV	Citrix XenServer	Hyper-V R2	Euca-lyptus	Nimbus	Open Nebula	oVirt
Hypervisor	VMware	KVM	Xen, *Hyper-V*	Hyper-V, Xen	Xen, KVM, *VMware*	Xen	Xen, KVM, VMware	Libvirt / KVM
VLAN	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Image Mgmt.	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Scheduling	Yes	Yes	N/A	N/A	Limited	External	External	No
Live Migr.	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
High Avail.	Yes	Yes	*Yes*	Yes	No	No	No	No
Hybrid Cloud	No	No	No	No	Partially	Partially	Yes	No
Admin GUI	Yes	Yes	Yes	Yes	No	No	No	Yes
Requires Intel VT / AMD-V	No	Yes	only for proprietary guests	No	if KVM hypervisor is used	only for proprietary guests	if KVM hypervisor is used	if KVM hypervisor is used
Officially supported Guest OSs	Windows, Linux, Solaris, NetWare	Windows, RHEL	Windows, RHEL, CentOS, SLES, Debian	Windows, SLES, RHEL	Depends on Hypervisor	Depends on Hypervisor	Depends on Hypervisor	Depends on Hypervisor
API	vCloud API	N/A	XenServer XML RPC	Hyper-V WMI	EC2 subset	EC2 subset & own WS	EC2 subset & OCCI	No
License	Proprietary	Proprietary	Proprietary	Proprietary	BSD *Proprietary*	Apache 2	Apache 2	GPLv2
Annual Cost	up to $4400 per CPU	up to $750 per socket	Free / *up to $5500 per host*	up to $3000 per CPU	Free / *N/A*	Free	Free	Free

Figure 6 - Table comparing different cloud toolkit solution (Schader, 2010)

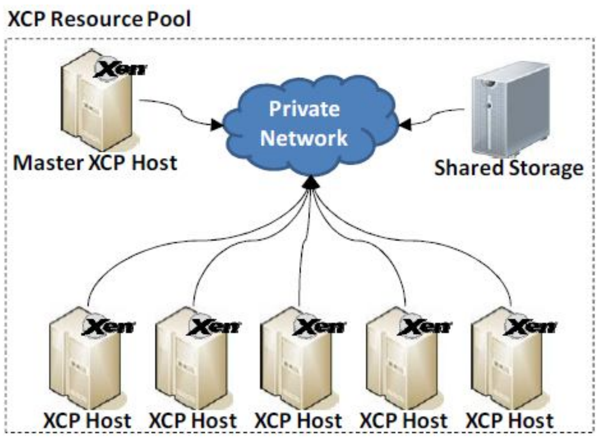


Figure 7 - Architecture of the Xen Cloud Platform (XCP) ***(Endo, 2010)***

OpenNebula is another cloud toolkit which can be used to implement private, hybrid or public cloud. The architecture of OpenNebula is very flexible which give a great modularity of the toolkit (Figure 8). The modularity of OpenNebula is due to its drivers based architecture. It comprises different sets of drivers for the different utilities required by the cloud such as virtualization, network, storage and external cloud (Sotomayor et al, 2009). The OpenNebula Core is responsible of managing three different areas which are the choice of the hypervisor that will be used to virtualized hosts, the choice of the network (fixe IP or DHCP) that will be used by the virtual hosts and finally the type of storage that will be used. Also the OpenNebula core will be able to deal with external cloud in the case of hybrid cloud.

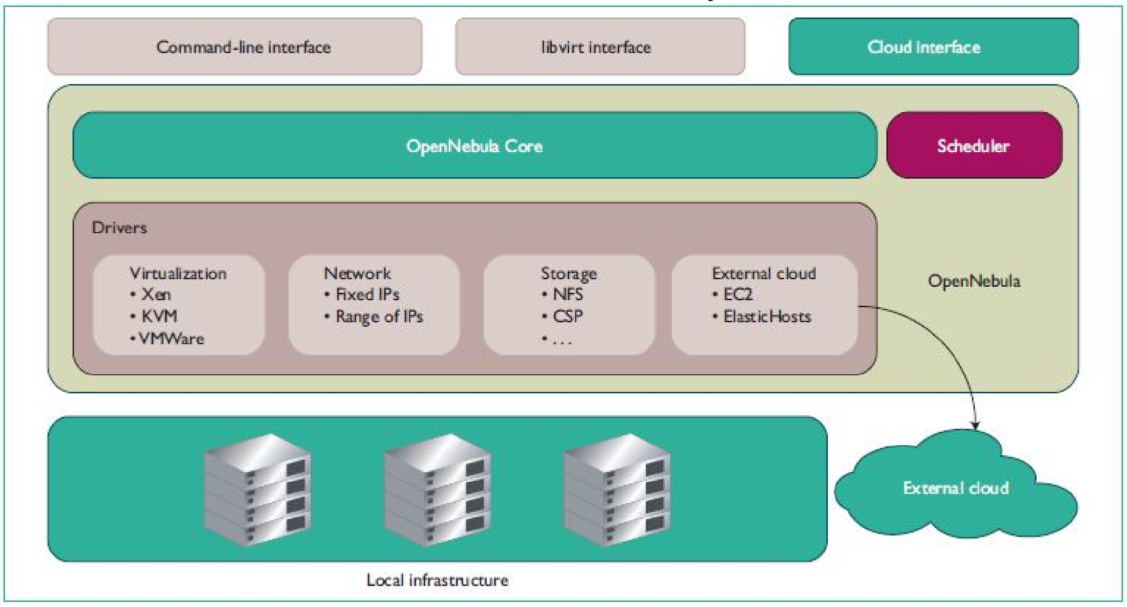

Command-line interface
libvirt interface
Cloud interface
OpenNebula Core
Scheduler
Drivers
Virtualization • Xen • KVM • VMWare
Network • Fixed IPs • Range of IPs
Storage • NFS • CSP • . . .
External cloud • EC2 • ElasticHosts
OpenNebula
External cloud
Local infrastructure

Figure 8 – Architecture of OpenNebula (Sotomayor et al, 2009)

The architecture of Eucalyptus is also different from OpenNebula and XCP. The architecture of Eucalyptus is hierarchical (Figure 9). It is composed of five components:

- **Cloud controller (clc):** is the interface that is used to interact with the cloud such as the command to create or terminate virtual entities through the API interface.
- **The node controller (nc):** is used to compute the elements of the cloud such as the creation of a virtual machine.
- **Cluster controller (cc):** is used for manages the different nodes connected to the cloud. Also it provides traffic isolation for the cloud.
- **Walrus (the S3-like storage service):** is a persistent storage for eucalyptus. It uses buckets and objects to organise the data.
- **Storage controller (sc):** is used to manage dynamics block devices that can be used by the virtual machine for persistent storage. (It is the equivalent of EBS for the Amazon cloud)

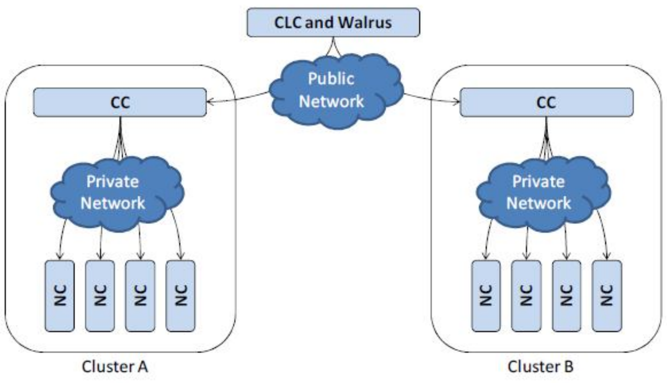


Figure 9 – Architecture of Eucalyptus (Endo, 2010)

Every cloud adopts different architectures to provide the same services. Depending of their architecture cloud toolkit offers more or less flexibility to users. Thus it can be seen that OpenNebula have more flexibility than others cloud toolkit because it disposed many configurable drivers. However OpenNebula seems to be more difficult to implement than Xen Cloud Platform (XCP) or Eucalyptus because XCP dispose of a tool to automate the maintenance of the cloud also Eucalyptus beneficiate of an excellent hierarchical structure (Enzo, 2010).

3.3 Virtualization

3.3.1 Differences between Virtual Machine Monitor (VMM)

Virtual Machine Monitor (VMM) is a thin software layer that is used to map virtual resources to physical resources. The resources provided by the hardware can be partitioned, shared with time sharing or emulated in software. The layer provided by the Virtual Machine Monitor has always more privileges than the guest Virtual Machine (VM) because the executions of instructions which are passed by the guest VM are managed by the VMM.
VMM can be categorized in two types (Rosenblum et al, 2005):

- Type I (called Hypervisor): VMM of type I is located just above the hardware and virtualize the entire hardware. Examples of VMM of type I are VMware ESX server, Hyper-V and Xen. (Figure 10)
- Type II (called Hosted system): VMM of type II runs within an Operating System (OS). Examples of VMM type II are WMware Workstation, QEMU, VirtualBox. (Figure 11)

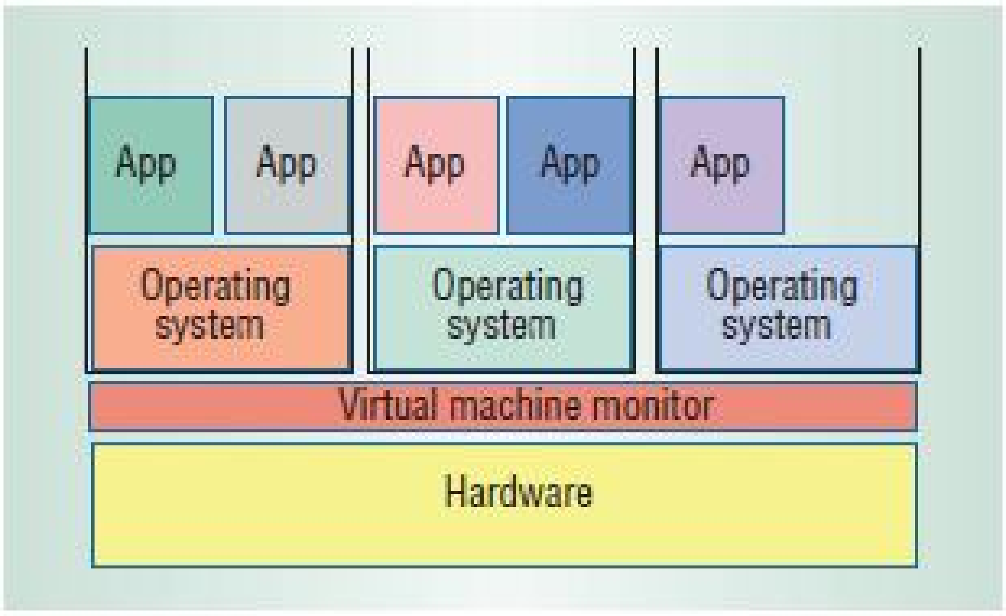


Figure 10 - VMM of type I (Rosenblum et al, 2005)

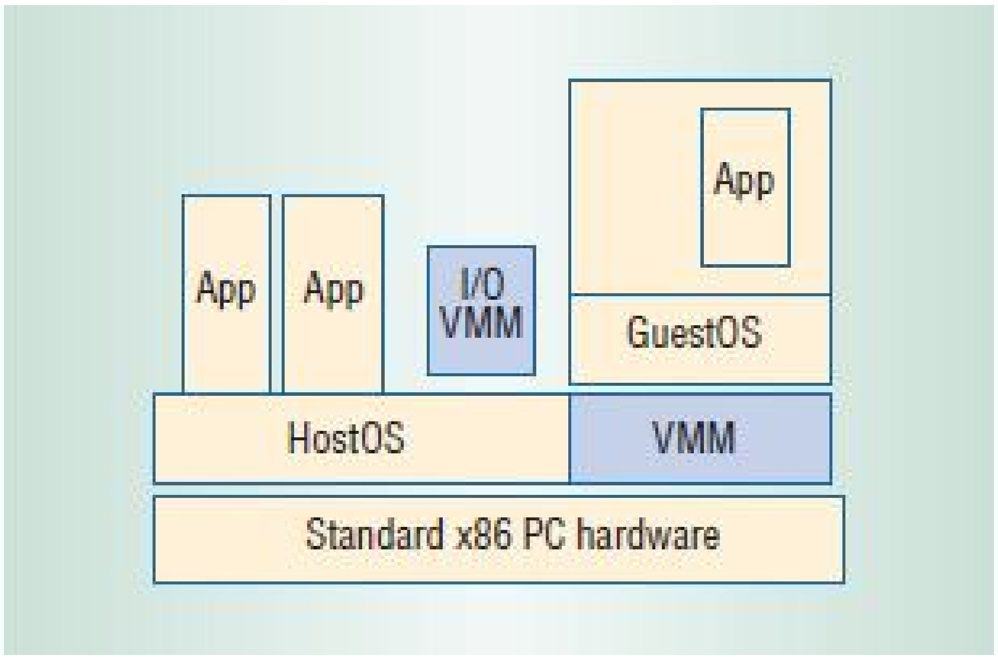


Figure 11 - VMM of type II (Rosenblum et al, 2005)

VMM of type I increase security and isolation because each virtual machine has its own execution environment. With VMM of type I the hardware of the virtual machine can be either access directly through mapping or virtualized. The architecture of VMM of type II is different from the type I in which the VMM runs within an OS. By running within the OS it makes it easier to implement however it reduces the isolation (Intel Corporation, 2008).

3.3.2 Comparison between physical and virtualized machines

In the technical review it has been presented the different virtualization techniques used to virtualized machine. Full virtualization is the type of virtualization which possesses the most overhead because it requires the use of a full trap-and-emulate scheme to be able to virtualize a machine (Kloster et al, 2007). Full virtualization has an important overhead because it requires to provide an entire vitualize architecture. Paravitualization impose minimal performance overhead contrary to full virtualization (Youseff et al, 2008). With paravirtualization there is no need to make a full trap of the instructions passed form the guest OS to the hardware. However paravirtualization requires modification of the guest OS for be able to be virtualized it. The performance of paravirtualization manages to be nearly equal to native performance. To be able obtain performances close to native the OS kernel is modify to replace nonvirtualizable instruction with hypercall (VMware, 2007). Hypercall will be able to communicate directly with the hypervisor. The major problem with paravirtualization is the need to modify the guest OS to be able to perform the virtualization. Hardware assisted virtualization (also called Hardware Virtual Machine or HVM) overcome the problem of having to modify the guest OS by making the trap of instruction at the hardware level. Hardware assisted virtualization is between full virtualization and paravirtualization. The hardware assisted virtualization is become possible in 2005 with the help AMD and Intel (AMD, 2005; Uhlig et al, 2005).

A study has been made to compare HVM with native performance (Kloster et al, 2007). In this study it has been performed on both Linux (Debian 4.0) and Windows

(Windows XP). The HVM performance has been tested on various conditions. To realize the test many tools such as OSDB, SPECweb99, Scimark, Bashmark, Nbench, NetIO, Aim9, Kernel compile and MemoryMotion have been used. It is observed than CPU performances are near to native performance. After having tested CPU the I/O performance are tested. Also it has been decided in their experiment to simulate connection to a web server at three different rate low (16 connections), medium (50 connections) and high (84 connections). On windows with HVM it has been observe an overhead of 6.33% at low rate however at medium and high rate the throughput drop drastically. While Windows and Linux when using native machine can handle medium rate connection Windows virtualized with HVM stop having conforming connection. The performances of Debian with HVM are a way better but still far away from native performances at high rate. Disk I/O performances seems to be better than network I/O performances with an overhead of 20% to 30% on average. Finally memory performances are tested and it is outline an overhead of 75% for Debian and 85% for windows. This high overhead is due to the use of shadow page table used to virtualize the MMU. It results of this study that the performances of HVM are satisfying in term of CPU power. However the management of I/O is need to be improved. Finally the constructor AMD and Intel need to do some efforts to improve the support for MMU virtualization (Kloster et al, 2007).

A previous study about HVM had been made by VMware however this study was not comparing HVM and native performances but HVM with the VMM provided by VMware. Of this study it was already found that performance limitation on HVM was coming from the lack of MMU virtualization (Adams et al, 2006). Even if the performances in that domains of virtualization is evolving really quickly it can be observe that in one year time from 2006 to 2007 no progress at the hardware level have been made to overcome the lack of MMU virtualization. When there is not special hardware support memory and MMU virtualization are perform due to the use of Shadow Page Tables. It is only recently in 2009 that Intel and AMD have provide new processors for trying to overcome the lack of MMU virtualization (Agesen, 2009). Because of the concurrence between AMD and Intel they come with two different solutions:

- The support for VMM virtualization introduced by AMD is called RVI (Rapid virtualization indexing) and the first processor to beneficiate of this technology is the quad-core Opteron ("Barcelona") CPU.
- On the other side the support for VMM virtualization of Intel is called EPT (for Extended Page Table)

Both solutions provide an overall gain of performances. However RVI and EPT suffer frequent Translation Lookaside Buffer (TLB) misses.
It has been measured performance gains of up to 42% for MMU-intensive benchmarks and up to 500% for MMU-intensive microbenchmarks for RVI. For Intel the performance measure gains of up to 48% for MMU-intensive benchmarks and up to 600% for MMU-intensive microbenchmarks (Bhatia, 2009). According to the result the performance of AMD processors are a little behind Intel performances.

3.3.3 Performances of Open Source products

The virtualization overhead involves performances depreciation rather to native performances. Research have been made to measure the overhead of the virtualization for different hypervisor such as XEN, KVM and VMware ESX (Apparao et Al, 2006; Menon et Al, 2005; Jianhua et Al, 2008; Shan et Al, 2009; VMware 2007). For their researches Menon et Al have used a toolkit called Xenoprof which is a system wide statistical tool implemented specially for Xen (Menon et Al, 2005). Due to this toolkit they have managed to analyse the performances of the overhead of network I/O devices. Their study has been performed within uniprocessor as well as multiprocessor. A part of their research has been dedicated to performance debugging of Xen using Xenoprof. Those researches have permitted to correct bugs and improve by that the network performances significantly.

After the debugging part it has been focused on the network performances. It has been observed that the performance seems to be almost the same between Xen Domain0 and native performances. However if the number of interfaces increase, the receive throughput of the domain0 is significantly smaller than the native performances. This degradation of network performances is cause by an increasing CPU utilisation. Because of the overhead caused by the virtualization there are more instructions that need to be managed by the CPU. This involves more information to treat and bufferization by the CPU which cause a degradation of receive throughput compared to native performances.

After having studied the performances of the domain0 they have observe that in a guest OS the receive throughput is slower than the receive throughput of the domain0 even with only one NIC. Once again this is because of the number of instructions which is more important than the number of instructions of the domain0.
Thanks to their study Manon et Al have managed to measure the overhead of the network for Xen version 2.0.3 (which is today out of date because the development of Xen is very fast and the last version stable version of Xen is the version 4.0.1). Due to their study it came out solutions to improve network performances of Xen virtualization. Manon et Al outline those performances degradation is generally caused by a CPU utilisation which is more important within virtualization because virtualization increase the number of instructions that needs to be managed by the CPU.

More recent studies compares try to compare the differences between hypervisors and especially the performances of each one according to their overhead (Jianhua et Al, 2008; VMware, 2007). Jianhua et Al compare the performances of KVM and Xen which are the two most popular Open Source hypervisor. The tools used by Jianhua et Al are different from the tools that were used by Menon et al. They are using three different benchmark tools to measure the performances: LINPACK, LMbench and Iozone. Their experiment is divided in three parts according to the specific utilisation of each tool. With LINPACK Jianhua et al have tested the processing efficiency on floating point. Different pick value has been observed over the different systems tested which are native performance, Xen and KVM. The result of this show that the processing efficiency of Xen on floating point is better than KVM because Fedora 8 virtualized with Xen have performances which represent 97.28% of the native rather than Fedora 8 virtualized with KVM represent only 83.46% of the native performances. The virtualization of Windows XP comes up with better performances than with the virtualization of fedora 8 on Xen. This is explained by the authors by the fact that Xen own fewer enhancement packages for windows XP than for fedora 8 because of that the performances of virtualized windows XP are slightly better than virtualized fedora 8.

After having testing the processing efficiency with LINPACK Jianhua et al have analysed memory virtualization of Xen and KVM compared to native memory performances with LMbench. It has been observed that the memory bandwidth in reading and writing of Xen are really close to native performances. However the performances of KVM are slightly slower for reading but significantly slower concerning the writing performances.

The last tool used by Jianhua et al is IOzone which is used to perform filesystem benchmark. Once again the native performances are compared to the virtualization performances of Xen and KVM. Without Intel-VT processor the performances of either Xen or KVM are around 6 or 7 times slower than the native performances. However within the Intel-VT processor the performances of Xen increase significantly because the performances are even better than native performances. However KVM does not exploit the functionalities of the Intel-VT processors and because of that does not improve his performances.

3.3.4 Performances of commercial Products

The competition between the different hypervisors have involves many studies to compare the performances of the different product to try to prove the efficiency of the product. Also XenSource have published an article comparing XenEnterprise and VMware ESX. To compare both hypervisor it has been performed test workloads (XenSource, 2007). Many tools have been used to benchmark both hypervisor. For evaluate intensive workload on the CPU SPECcpu2000 benchmark suite have been used. Passmark has been used to benchmark typical system workload. Netperf has been used to measure network performance. Also SPECjbb2005 have been used to represent performance of application server and it workload. The result obtained with SPECcpu2000 show that the results between ESX 3.0.1 and XenEnterprise 3.2 are nearly identical.
For the test perform with Passmark, Netperf and SPECjbb2005 the results are relatively equivalent again. Nothing significant can be observed between the performances of both hypervisor. This study concludes that two commercial solutions are almost at the same level of performances. Also because of the rapid development of cloud computing commercial many commercials solutions have appeared. Commercial solution of cloud computing include V-Sphere, VMware ESX, Windows Azure and many others. There are a growing business associate to cloud computing besides most of open-source solution available for cloud computing begin to develop commercial solution as well.

3.4 Conclusion

Because of the relatively recent development a lot of research are been made on cloud computing. Also each provider wants to impose its own technology which creates a lack of standardisation. To overcome this problem solutions such as the cloud manifesto try to be found. Progress have been made in the standardisation of the cloud however there are still some progress to make because providers such as Amazon still use proprietary standard which make Amazon images incompatible with others cloud providers. Even if there are still some fears about this technology the use of cloud computing is growing in IaaS, PaaS as well as SaaS.

The use of the virtualization has rapidly increased those past few years because of the benefits associate to it. Virtualization allows more security, permit to reduce the hardware cost with performances which are improving continually. Many researches are made in this domain to try to improve the performances. Also by implementing new types of processors AMD and Intel have manage to boost the utilization of virtualization. Also AMD and Intel both continue to try to make progress to improve virtualization in hardware. Progress are not only made in hardware there are also progress in software such as with Xen, KVM, VMware and so on which always try to improve the performance of their hypervisor.

4 Design and methodology

4.1 Introduction

Cloud computing can provide services at different level such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) (Gabrielsson et al, 2009). The infrastructure is the most fundamental part of the cloud without it the others layer of the cloud cannot exist. The performances of all services depend of the performances of the infrastructure provided by the cloud. Cloud computing performances depend of different parameters such as the CPU speed, the amount of memory, network and hard drive speed. In virtual environment the hardware is shared between virtual machines (Karger et al, 2008). To be able to have good performances machines hosting the hypervisors need to be really powerful to be able to provide the services requires for create and manage virtual machines. In this chapter it will be outline the design of a public and private cloud and the scenario that will be decided to test the performances of both public and private cloud. In a first place it will be outline the design of the performances tests of the Amazon public cloud in terms of CPU consumption, network speed and storage management. Secondly the conception of a private cloud only based on open source solutions will be designed and tested to identify the level of performances of a private cloud compare to a public one. Finally the tools used for the tests will be presented and it will be explained how they can be used to evaluate the performances of the clouds.

4.2 Platform Amazon (public cloud)

4.2.1 Description

Public cloud is a service which is provided by a third party. To define their commitment to the customers the Amazon Web Services (AWS) is using a Service Level Agreement (SLA). The SLA is an agreement between the AWS and the AWS customers where the AWS provides commitments to their customers. If the commitments are not honoured the customers can ask for compensation. Amazon with Amazon EC2 promises an Annual Uptime Percentage of at least 99,95% during the service year. If Amazon does not manage to meet this commitment customers will be eligible to receive a Service Credit equal to 10% of their bill. Indemnification of the client does not concern any performances problems (Amazon, 2010). Also Amazon keeps the right to modify the performances of the instances for the different instance types which involve a lack of flexibility for the customers.

The Amazon cloud provides an Infrastructure as a Service (IaaS). This service is an on demand service access through a web interface. Within the Amazon cloud customers are able to instantiate virtual machines corresponding to their need. Amazon provides 10 different types of instances (Appendix 1) which goes from a price of $0.029 per hour for the cheapest to $2.76 per hour for the most expensive (Amazon, 2010). The service is on demand customers only pay for what they are using. The Amazon cloud is using an architecture which use Xen as hypervisor to virtualize the virtual machines (Wang et al, 2010). Also Amazon has created their own standard of images for their virtual machines called AMI. The virtualization provided by Xen can be either paravirtualization or hardware-assisted depending of the Operating System virtualize (Figure 13). In fact Linux and Unix based machine can be paravirtualized within Xen however Microsoft Windows instances cannot.

Thus to overcome this drawback Xen uses hardware-assisted virtualization to virtualize Windows machines. In the Xen system the hardware-assisted virtualization is called Hardware Virtual Machine (HVM). The virtualization using HVM has lower performances than paravirtualization. It will be exposed the degradation of performances that can occurs with hardware-assisted compare to the paravirtualization. The performances' tests will permit to outline the performances that can be expected when using the Amazon cloud because differences of performances occur between paravirtulization and HVM.

The Amazon cloud is composed of different of five principle components (Figure 12):

- Amazon SQS which is used to manage durably buffering request.
- A controller used to link user information to instances.
- Amazon SimpleDB stores the information relatives to customers such as status, logs and so on
- Amazon EC2 is used for running virtual instances
- Amazon S3 acts as a bucket to store or retrieve information on the cloud.

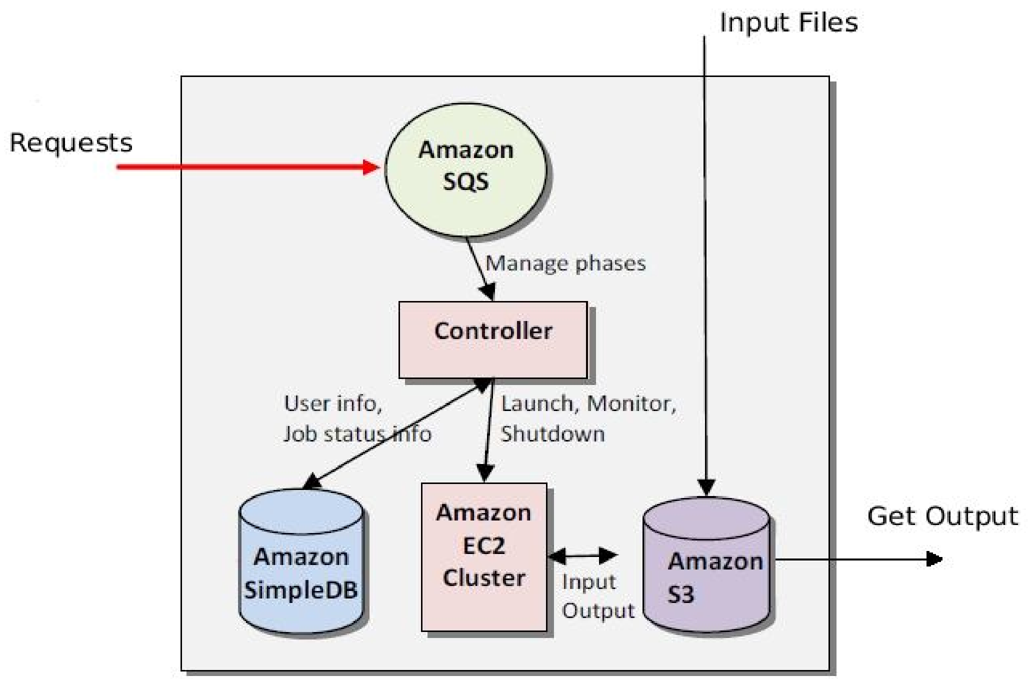


Figure 12 - Architecture of the Amazon cloud (Varia, 2008)

4.2.2 Design of the experiment

4.2.2.1 Presentation of the system studied

As describe previously the Amazon cloud can provide different instances types such as standard, micro, high-memory, high-CPU and cluster compute instances. Each of the machines has specific characteristics which are available on the Amazon website (Amazon, 2010) or in Appendix 1. It has been decided to use only 32 bits machines for the test because those machines are the smallest ones and have the cheapest cost. For the 32 bits machines the Amazon cloud provide only three types of instances which are:

1. The standard instance– Small:
   - 1.7 GB memory
   - 1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit)
   - 160 GB instance storage
   - 32-bit platform
   - I/O Performance: Moderate
   - API name: m1.small

2. The micro instance:
   - 613 MB memory
   - Up to 2 EC2 Compute Units (for short periodic bursts)
   - EBS storage only
   - 32-bit or 64-bit platform
   - I/O Performance: Low
   - API name: t1.micro

3. The high-CPU instance – Medium:
   - 1.7 GB of memory
   - 5 EC2 Compute Units (2 virtual cores with 2.5 EC2 Compute Units each)
   - 350 GB of instance storage
   - 32-bit platform
   - I/O Performance: Moderate
   - API name: c1.medium

All instances of the cloud are virtualized with Xen. Figure 12 present a simplify design of the Amazon EC2 architecture.

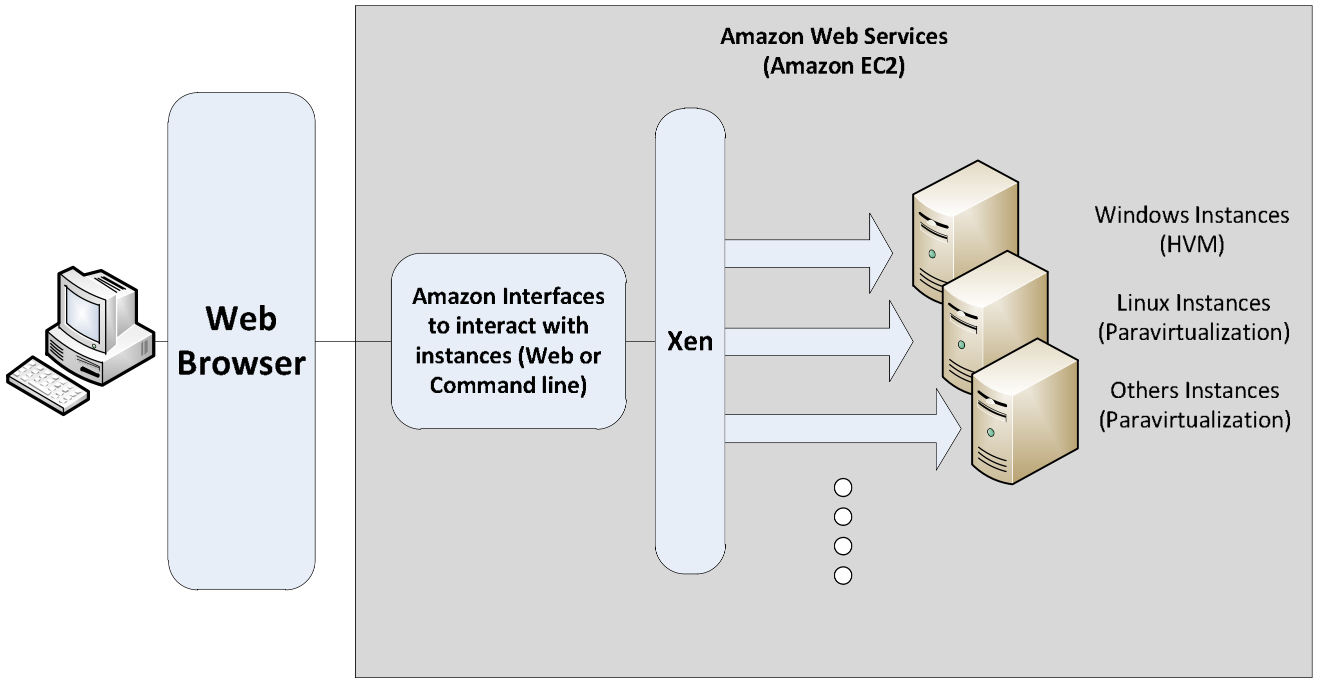

Amazon Web Services
(Amazon EC2)
Web Browser
Amazon Interfaces to interact with instances (Web or Command line)
Xen
Windows Instances (HVM)
Linux Instances (Paravirtualization)
Others Instances (Paravirtualization)

Figure 13 - Design simplify of the Amazon EC2 implemented with Xen has hypervisor

4.2.2.2 Testing the CPU and memory

Amazon qualifies the CPU power of the machines with the EC2 Compute Units (Amazon 2010). These Units represent the amount of CPU that is allocated to a particular instance. According to Amazon: “One EC2 Compute Unit provides the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor”. The results of the tests provided by Amazon to test their machine give brief idea of the power of the different instances. It will be performed tests that will gives a better idea of the power that a customer can expect for the instances previously describes. Also the results will give comparison with recent machines to give a better idea of the performances.

Because Amazon is using Xen it provides paravirtualization for Linux and Unix instances and HVM virtualization with Windows instances. It will be compared the differences of performance between those type of virtualization for the different type of instances described previously. Few researches have already been made to compare HVM and paravirtualization (Zhao et al, 2009). Also because windows machines are virtualized using HVM it can be expected lower performances than with Linux (or Unix) which are paravirtualized. To evaluate the performances of the different instance types it will be used a benchmarking tool named Geekbench because it is multi-platform and available for free for 32 bits machines. Geekbench will be installed on the different instance types define previously. The results provided with Geekbench will be retrieved in a text file.

4.2.2.3 Testing the network

The network of the cloud is shared between the different instances running on the cloud. The performances will be reduced if there are a large number of virtual machines running on the same node. If too many machines are running and are using at the same time the network will have to dealing with a large quantity of information. Thus fast equipments and fast connections are required. Also the network card of the node virtualizing all those machines can become a bottleneck (Isogai et al, 2009). Within the Amazon cloud there is no way to know how many machines are running on the same nodes. However in the documentation provided by Amazon it is explained that if different machines try to use as much resource as possible the resource will be equally shared between the different virtual entities (Amazon, 2010). Also when a resource is underutilized most of the time the customer will be able to use a higher share of that resource while it is available. For each instance performance are presented with the indicator low, moderate or high. It will be once again tried to give an idea of the performances that we can have expect more precisely. For the networking test it will be used Iperf which is a tool used to measure network performances (He et al, 2010). Iperf works as a client/server application. The tests will be divided in three scenarios (Figure 14):

- It will be measured the performances of the localhost (The client and the server will be install on the virtual machine.
- It will be measure the performances with the machines that are inside the cloud (The client will be install on one virtual machine and the server will be on another machine of the cloud)
- It will be measured the performances between a machine which is on the cloud and a machine which is outside the cloud.

All three scenarios will be performed on the different types of instances describe earlier.

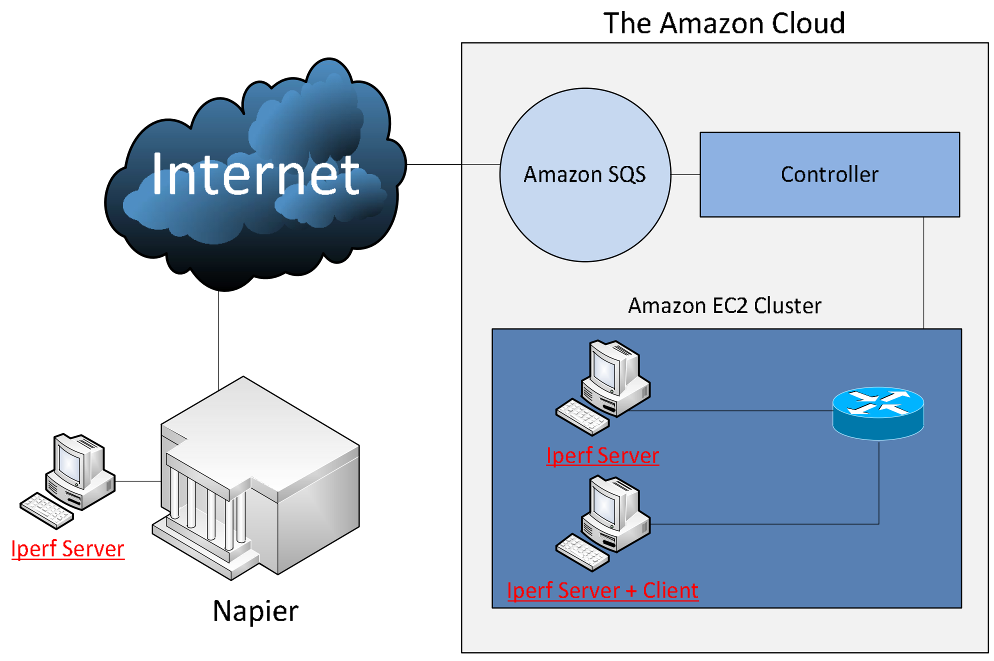


Figure 14 – Design of the network tests

4.2.2.4 Testing the Hard Drive performances

Such as the network, the access to the hard drive is shared between the virtual entities. Because both disk access and network usage are shared resources they are consider in the same way in the Amazon cloud. No differences are made and that means that the disk performance and the network performances are related according to the Amazon documentation (Amazon, 2010). The performances are indicated by low, moderate or high such as for the network. Performances of the different instance types will be compared. However the performance between Windows (HVM) and Linux (paravirtualization) won't be accurate because it is not the same software that will be used to measure the performances on the different systems. For testing Linux performances it will be used hdparm which is a Linux tools used to measure the access to the disk. For Windows it will be used PassMark Performance Test 7.0. This software can be used to benchmark different component such as CPU, memory, disk and so on. For the test it will be only used the disk performances.

4.3 OpenNebula platform (private cloud)

The differences of private and public cloud are where the cloud is deployed. For public cloud it is usually deployed over the internet rather than for private cloud it is deployed behind the firewall of the company (King, 2008). To perform test on a private cloud it has been decided to implement a cloud by using only open source solution. The choice of the hypervisor has been Xen and the cloud toolkit chosen was OpenNebula. Xen is one of the most famous Open Source hypervisor available and OpenNebula has been chosen for it flexibility because it can for example interact with different hypervisor such as Xen, KVM and VMware ESX (Sotomayor, 2007). For implement a private cloud it will be required 2 machines:

- A frontend: It requires a machine with good performances and lot of storage.
- A node: It requires a machine with high CPU performance capable of support virtualization. As seen previously only AMD-V and Intel-VT have support for virtualization.

The minimum requirement for the implementation of a cloud is usually two machines. One machine being called frontend on which is implemented the cloud toolkit. The second being the node implementing the hypervisor used to virtualize machines.
The test platform will be composed of the implementation of OpenNebula on the frontend and Xen on the node (Figure 15). For the OS it has been decided to use OpenSuse because Xen have a good support on this distribution. Step by step: it will be first implemented the OpenNebula toolkit, then it will be implemented the hypervisor and finally it will be made the connection between the cloud toolkit with the hypervisor.

After the platform being in place it will be created images of Windows and Linux to perform tests. One Linux image and one Windows server 2008 image will be created to run on the cloud. The images of both Windows Server 2008 and Linux will be created by using Xen and then it will be used the OpenNebula toolkit to manage and deploy those images for users. Tests such as the test perform for measuring the performances of the Amazon cloud will be perform. By performing the same test than the tests perform on the Amazon cloud for the CPU and memory. Also it will be possible to compare the performances of the Amazon cloud with the performances of a private cloud. It will be possible to compare as well the average price of a machine running on the Amazon cloud with the price of a private cloud. Finally it will be performed few tests to observe how react the CPU and network performances in function of the number of machines running on the node.

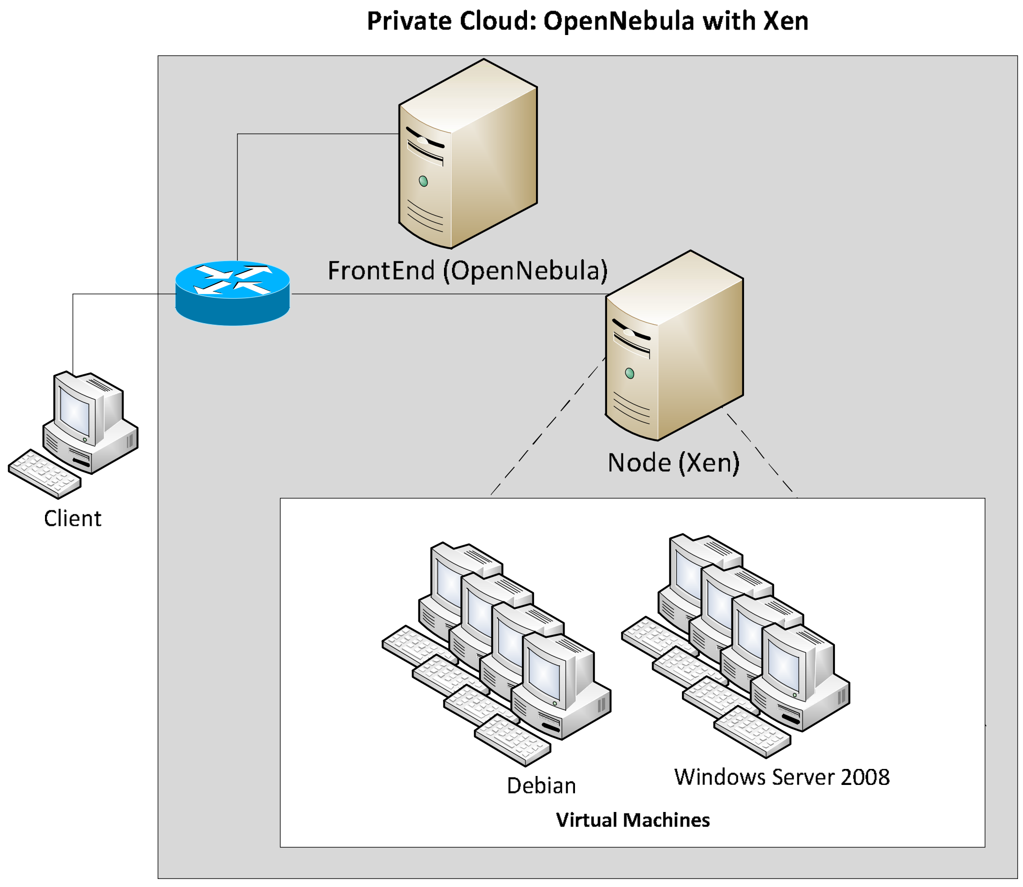


Figure 15 - Test platform implementing OpenNebula with Xen

4.3.1 Measuring CPU performances in function of the number of virtual machine running on the node

To be able to measure the CPU performances the CPU utilisation will be increase with CPUkiller3 on Windows server. The number of machine will be progressively increase and the CPU of each machine will be increased as well to compared the performances of one machine running next to other machine running with 100% of CPU utilisation. The figure 16 explains how the experiment will be performed. The tests will be only made within HVM it is easier to implement this experiment. Also the degradation of performance should be relatively equivalent within the paravirtualization.

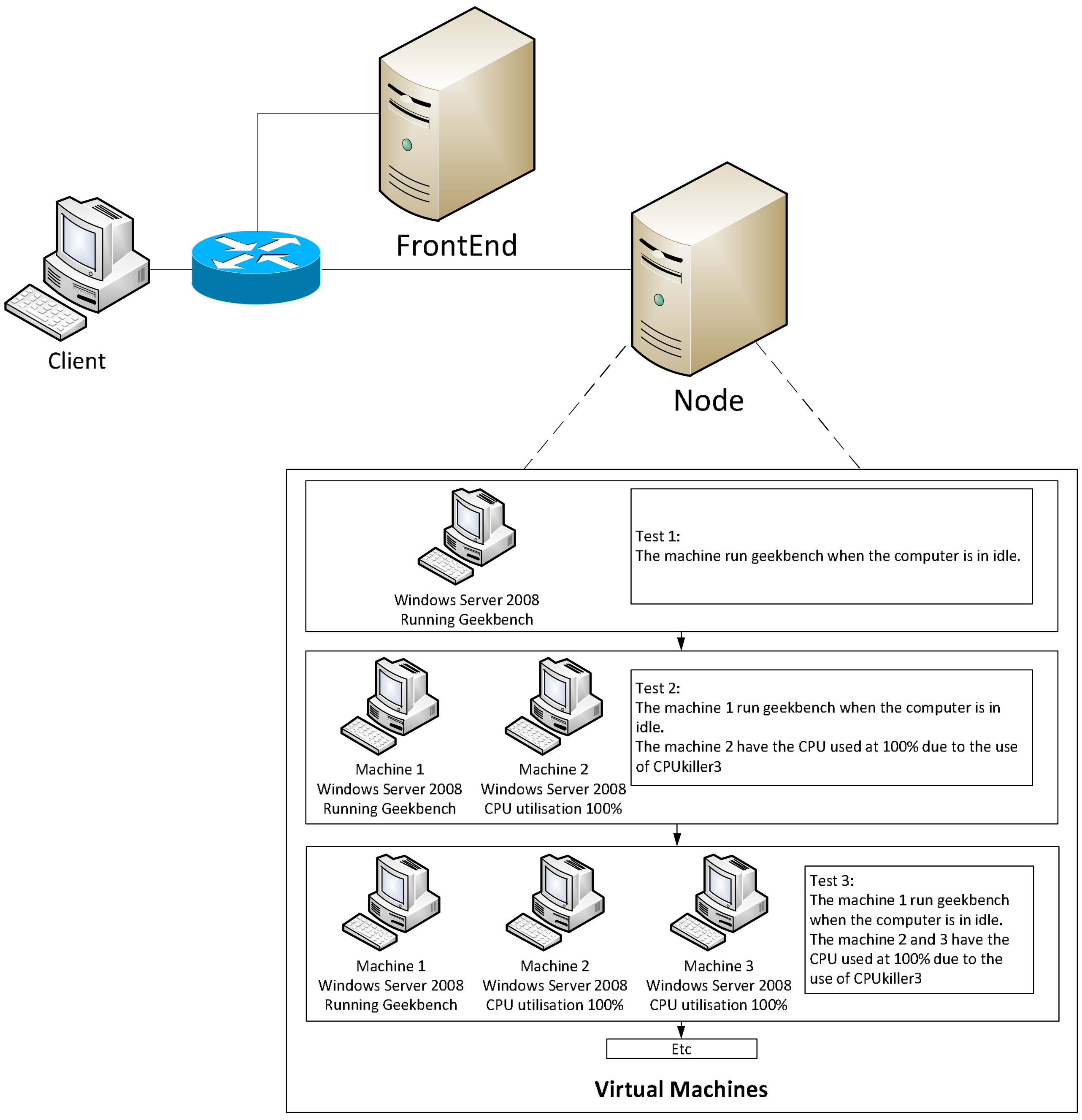


Figure 16 - CPU performances in function of the number of virtual machine running on the node

4.3.2 Measuring network performances in function of the number of virtual machine running on the node

For measuring network performances it will be used a tool to flood the network. This will permitted to overload the network. The number of machines will be progressively increased to outline the performances of the node when multiple machines are using intensively the network card (Figure 17). To flood the network it will be used hping3 on Linux. Also to measure the network performances it will be used netperf and iperf. The following diagram will explain the design that it will be used for performing those tests.

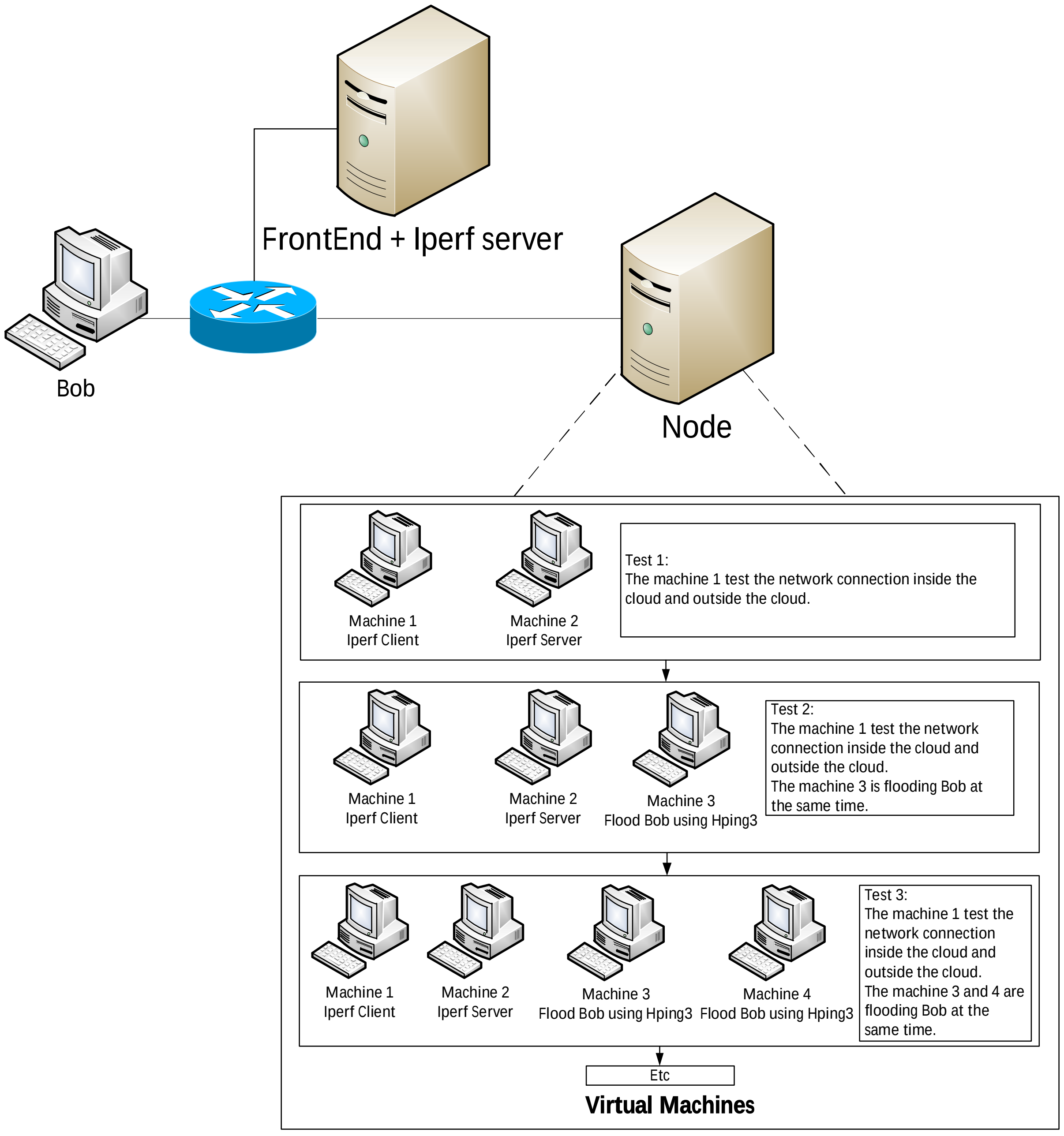

FrontEnd + Iperf server
Bob
Node
Machine 1
Iperf Client
Machine 2
Iperf Server
Test 1:
The machine 1 test the network connection inside the cloud and outside the cloud.
Machine 1
Iperf Client
Machine 2
Iperf Server
Machine 3
Flood Bob using Hping3
Test 2:
The machine 1 test the network connection inside the cloud and outside the cloud.
The machine 3 is flooding Bob at the same time.
Machine 1
Iperf Client
Machine 2
Iperf Server
Machine 3
Flood Bob using Hping3
Machine 4
Flood Bob using Hping3
Test 3:
The machine 1 test the network connection inside the cloud and outside the cloud.
The machine 3 and 4 are flooding Bob at the same time.
Etc
Virtual Machines

Figure 17 - Network performances in function of the number of virtual machine running on the node

4.4 Benchmarking tools

GeekBench is a benchmarking tool that can be used to benchmark CPU and memory. It is multiplatform and can be used on Windows, Linux, Mac OS X, Solaris and Haiku. It is also possible to install it on an iPhone and iPad. GeekBench possesses a large database of reference machine which has been divided in two categories: PC and Mac. It is precised that of the tests made for obtain reference result have been performed on computers which had at least 512 MB of RAM memory and the processors weren't overclocked. There are 2 versions of the software one is for the 32 bits machines and can be used for free the second version is for the 64 bits machine and needs to be purchase to be able to use it. The results of GeekBench can be saved in a text file or it can also be sent to the GeekBench website and accessed through a web page. Iperf is a networking tool that will be used to measure network performances. It is multiplatform thus it can be used on Linux and Windows. Also it is a tool that can be used to measure the maximum TCP and UDP performances. It is rely flexible because multiple parameters can be tuned. Different parameters such as bandwidth, delay jitter, datagram loss can be reported. Hdparm is a Linux tool used to measure disk performances. It can give information for perform device read timings and perform cache read timings. PassMark Performance Test 7.0 will be used to measure disk performances on Windows machines.

4.5 Conclusion

The test has been divided in two parts. At first it is tested the performances of the Amazon cloud (Public cloud). The utilisation of those kinds of cloud is generally used for deal with data that are not sensible. More sensible data are generally treated within a private cloud because those one are located behind the firewall of the company. In the second part a private cloud will be implemented to be able to perform some tests on it. The design has for aim to outline the performances of the cloud through many different experiments. It will be tested different parameters such as the CPU, the memory, the network and the hard drive. The experiment will require the utilisation of different tools such as networking tools to measure the network speed, CPU benchmarking tools and so on. In the next part it will be implemented the different experiment presented previously.

5 Implementation

5.1 Introduction

In this part the design previously presented is going to be implemented. The implementation will be divided in two major parts the first part represent the implementation of the test perform on the Amazon cloud. In this part it will be outiline the way of how Amazon is working. Also different type of instances will be launched and tested. In a second part it will be implemented a private cloud and test will be made on it. For this part the implementation of the private cloud was requiring a long part of configuration to be able to instantiate machines on the private cloud.

5.2 The Amazon cloud

5.2.1 Connection to the Amazon cloud

The Amazon cloud is a public cloud thus it requires the creation of an account to be able to use it. During the creation of the new account it is required to give a name, an email, an address, a phone number and your credit card number. To be able to create the account the customer need to accept the AWS customer agreement (Amazon, 2010). The machine of the cloud can be managed by the web interface or within EC2 command line tools. It will be essentially used the web interface to manage the instances of the Amazon cloud during the tests because it can be access from any computer and there is no need to install the credential on all the machines that will be used. To be able to use EC2 command line it is requested to install the credentials to access the service.
Windows instances are accessed through the remote desktop tool available with all Windows. To connect to the remote machine it is required to specify the public IP address of the instance given by Amazon. The username for the session for default machines available through the Amazon cloud is Administrator. The password is encrypted and can only be decrypted on Amazon by giving the private key on the website. Windows machine beneficiate by default of a graphical interface. For Linux instances there is no graphical interface available by default. The connection to those machine is using SSH. Linux instances can be accessed through any client SSH. To connect to the remote instance it is required to specify an IP address, the private key specified during the creation and the username of the account (such as root, ubuntu, and so on). It will be deployed machines of each following type for windows and Linux:

- Micro Instance
- Small Instance
- Medium Instance

5.2.2 CPU and memory test

To perform the CPU test of the Amazon cloud it has been install Geekbench on each machine. Because Geekbench is multiplatform it can be used with both Windows and

Linux. To perform the tests Geekbench only require the launch of the executable. Afterward the result of the test can be copied in a text file or can be sent to the Geekbench website. It will be chosen to copy the results in a text file. After getting all the results from micro, small and medium instances in a text file for both Windows and Linux the results will be regrouped in a table to be able to analyse them.

5.2.3 Networking and hard disk tests

As explained earlier the tool used to measure network performance work as a client server tool. It has been decided to install server at different places to evaluate the performances of the network inside the Amazon cloud as well as between the Amazon cloud and Edinburgh Napier University. Also it will be tested the performances of the localhost. To be able to perform the test an Iperf server has been deployed on a machine at Napier. Also on the Amazon cloud it has been deployed a machine that will act as an Iperf server as well. Once the platform has been setup it is only required to launch the client for testing the performances of the different scenario proposed.
To be able to measure hard disk drive performances it has been used two different tools on Windows and Linux. On Windows it has been used PassMark Performance Test 7.0 and for Linux it has been used hdparm. The results for Windows and Linux will be different because two different tools will be used. It will be try to correlate the different results obtained within the different tools.
To simplify the test it has been implemented a script for Linux to automated and simplified the test on each instance (Figure 18). For windows no script has been implemented because it is easier to use the graphical interface rather than to use a script.

```
#!bin/bash

if [ "$#" -eq 0 ]; then
echo "Need to specify an argument"
exit 1
fi

amazon=[ip_address_of_the_amazon_iperf_server]
napier=[ip_address_of_the_napier_iperf_server]

cd
cp sources.list /etc/apt/sources.list
sudo apt-get update

# Benchmark CPU + memory
## Geekbench
wget http://www.primatelabs.ca/download/Geekbench21-Linux.tar.gz
mkdir geekbench
cd geekbench
tar xvzf ../Geekbench21-Linux.tar.gz
./dist/Geekbench21-Linux/geekbench_x86_32
cd
mkdir $1
cp geekbench/geekbench_$1 $1/
```

```
# Benchmark Network
## iperf
sudo apt-get install iperf
iperf -s -p 5001 -f m &
iperf -c $amazon -i 1 -p 5001 -f m -t 10 > $1/iperf_amazon_$1
iperf -c $napier -i 1 -p 5001 -f m -t 10 > $1/iperf_napier_$1
iperf -c 127.0.0.1 -i 1 -p 5001 -f m -t 10 > $1/iperf_lo_$1

# hdparm -> HDD performances
sudo apt-get install hdparm
hdparm -tT /dev/sda1 > $1/hdparm_xvda1_$1
```

Figure 18 - Linux Script

5.3 Implementation of a cloud using OpenSuse, OpenNebula and Xen

It has been decided to implement a private cloud to be able to perform tests and compare private and public cloud. For the implementation it was required to choose a platform such as Linux, Unix, Windows or Mac associate to the use of a toolkit such as OpenNebula, Hyper-V, Eucalyptus and so on. Finally it was required to choose a hypervisor such as Xen, KVM or VMware to provide virtualization of the machines. It has been decided to create a cloud by using Open Source solutions for economic reasons. Thus it has been decided to use Xen has a hypervisor. After that is has been decided to use OpenSuse because Xen is well supported by this distribution of Linux. OpenNebula is the toolkit solution that will be used because it provides a great modularity.
There are requirements that need to be achieved to successfully install the cloud. A minimum of two machines are required for the creation of a cloud. One machine will be used as a front-end and the other one will be used for the node. The front-end is the part of the cloud that manages instances of the cloud. It is through the front-end that users manage their instances. In the basic architecture of OpenNebula the front-end contain also the storage of the virtual machines. The node is used for the virtualization of the instances of the cloud. The front-end commands to the node and manages the allocation of the resources between different node if different node are available. The resources of the node are used only for virtualization purposes. To achieve good performances the node need to be powerful and the processor of the node needs to support virtualization capabilities.

5.3.1 Implementation of the frontend

For the installation of the frontend it has been decided to install the last available version of OpenSuse which is OpenSuse 11.3 64 bit version. On this version of OpenSuse it will be installed OpenNebula 1.4.
A package is available for OpenNebula 1.4 at the OpenNebula website is compiled in OpenSuse 32 bits version. We will use this package for facilitate the installation.
According to the website the specifications of the package are (OpenNebula, 2010):

- It comes statically linked to xmlrpc library so it easier to install. The rest of the dependencies from the platform notes should be met.
- The software is installed in /srv/one/cloud.
- oneadmin user is created. oneadmin user in execution hosts should be created with the same id as in the master.

Before to install OpenNebula some dependencies need to be install on the host OS.

The dependencies requires by OpenNebula for OpenSuse are:

- It is required to install at first the building tools:

```
zypper install gcc gcc-c++ make patch
```

- Then it is required to install the libraries:

```
zypper install libopenssl-devel libcurl-devel scons pkg-config sqlite3-devel
libxslt-devel libxmlrpc_server_abyss++3 libxmlrpc_client++3 libexpat-devel
libxmlrpc_server++3 libopenssl0_9_8 sqlite3
```

- Ruby needs to be installed:

```
zypper install ruby ruby-doc-ri ruby-doc-html ruby-devel rubygems
```

- xmlrpc-c:

```
yast --install subversion
cd /opt
svn co http://xmlrpc-c.svn.sourceforge.net/svnroot/xmlrpc-c/super_stable xmlrpc-c
cd xmlrpc-c
./configure
make
make install
```

Once all the dependencies have been installed the OpenNebula package can be installed on the front-end.

```
#Rpm -Uvh one-1.4.0-1.i586.rpm
```

After having installed OpenNebula some configuration are required such as the creation of a password for the user oneadmin which is the main user for OpenNebula, the creation of a directory to share the images of the instances and so on.

To create a password for oneadmin:

```
passwd oneadmin
```

Oneadmin is the user used to manage the cloud. From this user it can be managed the network, the nodes, the users and the deployment of instances. To connect with the user oneadmin the following command was used:

```
su – oneadmin
```

It has been required to create a folder that will be share by NFS on the frontend. The folder have been created at /srv/cloud/images:

```
mkdir /srv/cloud/images
```

OpenNebula needs the one_auth file to work this file contain username and password of users. Also by convention the first user need to be oneadmin. As it is a self contain installation the home folder for oneadmin is /srv/cloud/one.
Because it is not created during the installation it is required to create it manually:

```
mkdir /srv/cloud/one/.one
```

```
touch /srv/cloud/one/.one/one_auth
echo “oneadmin:oneadmin” >> /srv/cloud/one/.one/one_auth
```

OpenNebula can be launch now with the command:

```
one start
```

It is required to specify the Environment variables associate to OpenNebula in the file /etc/bash.bashrc.
This is done by adding to /etc/bash.bashrc the following lines:

```
ONE_LOCATION="/srv/cloud/one"
export ONE_LOCATION
PATH=$PATH:$ONE_LOCATION/bin
export PATH
```

Finally to allow OpenNebula to work with Xen the file oned.conf located at $ONE_LOCATION/etc/oned.conf needs to be modified and some options need to be uncommented (See file used in appendix 2). After having accomplished all those step the front-end is ready to work and required the association with one or more nodes.

5.3.2 Implementation of the node

For the node it has been decided to use OpenSuse 11.2 rather than 11.3 because the 11.2 was still supporting the version 3.4 of Xen. For OpenSuse 11.3 the support for Xen was for Xen 4 which is new and less stable than the 3.4 version. Thus with OpenSuse 11.2 it has been install Xen 3.4 which is provided with this version of the OS. To install Xen it has been used the install Hypervisor tools which automatically install Xen with all the dependencies. After the installation of Xen the node needed to be restarted to boot on the Xen kernel. To be able to associate the node and the front-end together it is required to exchange the RSA keys. Once the keys have been exchanged between the front-end and the node OpenNebula can associate the node with the use of the following command:

```
onehost create [ip_address_of_the_node] im_xen vmm_xen tm_nfs
```

The association of the front-end with the node can be verify due to the following command:

```
onehost list
```

The output of this command looks like the following example when the node is associated:

```
ID NAME                        RVM  TCPU  FCPU  ACPU  TMEM     FMEM     STAT
0 [ip_address_of_the_node]    3    400   395   395   1257267  2302976  on
```

To finalize the creation of the cloud it is needed to create a network. To create the network of the machines it will be used the following command:

```
onevnet create network.txt
```

The network file contains the information of the network:

```
NAME = "Network"
TYPE = RANGED
# Network binds to ''br0'' for Internet Access
BRIDGE = br0
NETWORK_SIZE = C
```

```
NETWORK_ADDRESS = 192.168.0.0
```

The status of the network can be verify with the command:

```
onevnet list
```

If the network has been correctly created it should output:

```
  ID USER      NAME      TYPE      BRIDGE    #LEASES
   0 oneadmin  Network   Ranged    br0       0
```

After having setting up the cloud it was required to create machines to associate to the cloud. It has been chosen to implement one Debian machine and one Windows Server 2008. After created the images of the two different machines it was required to copy the images on the front-end in the folder “/srv/cloud/images” created to share the images. Also NFS needs to be configured to be able to share the folder “/srv/cloud/images” over the network. Templates that have been used to launch the Debian instance as well as the Windows instance are presented in appendix 3 and 4.

5.3.3 Differences of CPU performances between native, para-virtualisation and HVM

Native performances have been measured at first. To be able to measure the native performances it has been used the same hardware that is used on the node within Ubuntu has an Operating System. On this machine Geekbench has been run to measure the native performances. After having performed the test to measure the native performances it has been launched on the cloud a Debian instance which was using the entire resources available on the cloud. Once the machine was running it has been run Geekbench. Finally the Debian machine has been destroyed and the same test has been performed on a Windows Server 2008 which was using all the resources available of the cloud such as the Debian machine.

To observe the performances of the CPU in function of the number of instances running on the cloud it has been launched one instance of Windows used to perform the performance tests with geekbench. After each test one more machine Windows machine will be launch on the cloud. The new instances that will be launched will be running at 100% of CPU usage. To be able to run the machines at 100% of CPU usage it has been used CPUkiller3. Also each new machine have the same configuration than the one used to measure the performances. The tests have been performed 3 times with three different configurations. At first all the instances were using 1 CPU. Then all the instances were using 2 CPU. Finally all the instances were using 4 CPU.

For measure network performances of the private cloud Iperf have been used. Such as for the tests on the Amazon cloud Iperf servers have been implemented at different parts of the network. It has been dedicated one Debian machine on the cloud to act like an Iperf server. Also another machine external to the cloud have been implemented to act like an Iperf server. Another machine on the cloud has been used to perform the measure of the network. After each measure one new machine has been added to flood the network and see how evolve the performances. However after the use of two machines flooding the network the router that was used for the test was not able to manage all the traffic transiting through it. Thus the test has only been performed with 2 machines flooding the network.

5.4 Conclusion

The implementation was requiring competences in many different domains. It has been learn in this part how to use a public cloud such as Amazon. Also the utilization of the different tools used for benchmarking the performances of the different instances. The implementation of the private cloud has required knowledge in Linux system. The entire configuration made to implement the OpenNebula cloud with Xen has permit to realize some tests on a private cloud. The implementation of OpenNebula was difficult because the documentation was not easy to find and problems have been encounter. However all the problems encounter have been solved and at the end the cloud was running. The implementation of the private cloud was one of the most difficult part of the implementation and have required lot of researches. In the next part it will be evaluate the results found for the public cloud as well as for the private cloud.

6 Evaluation

6.1 Introduction

In this part it will be evaluate the performances for public and private cloud. It will be first analyse the performances of the Amazon cloud for three different types of instances. After that it will be evaluate the performances of the private cloud and compared the virtualization performances of paravirtualization and HVM to the native performances. Finally it will by outline the price of the two different solutions used to implement a cloud and compared to see if one solution is better than the other.

6.2 The Amazon cloud

The performances of the Amazon cloud are described in the Amazon documentation in terms of EC2 Compute Unit. It is explained that “one EC2 Compute Unit is the equivalent of the CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor” (Amazon, 2010). To give a better idea of the performances that can offers Amazon machines compared to reel machines. The tests have been performed with Geekbench on small, micro and medium instances. Also the tests have been made on Ubuntu which is para-virtualized and Windows which use HVM virtualization. According to the documentation the performances of:

- Micro instances are variable and can vary from one EC2 Compute Unit to two EC2 Compute Unit
- Small instances are one EC2 Compute Unit
- Medium instances are five EC2 Compute Unit

The figures below (Figure 19 and 20) show the CPU performances of both AMD and Intel processors compare to three types of instances that can be used in the Amazon cloud. It is observed that the performances of the different instances is a little bit different from the performances described in the Amazon documentation because it can be observed difference of performances between the virtualization of Ubuntu and Windows for the same type of instance used. This difference of performances between Windows and Ubuntu can be explained by the fact that Ubuntu is using para-virtualization while Windows is using HVM. Also it has been explained before that para-virtualization offers better performances than HVM virtualization. There are indeed differences between each type of instances in function of the virtualization used. Thus if a customer want the best performance possible he should use instances using para-virtualization rather than HVM. The performance of micro instances are variable also it is shown on Figure 19 and 20 that the micro instance with Ubuntu have better performance than the medium instance with Windows.
In the following it will be compare Amazon instances for the processor integer performance, the processor floating point performance, the memory performance and the memory bandwidth performance. Also it will be study the in more detail the differences of performance between para-virtualization and HVM within the Amazon cloud.

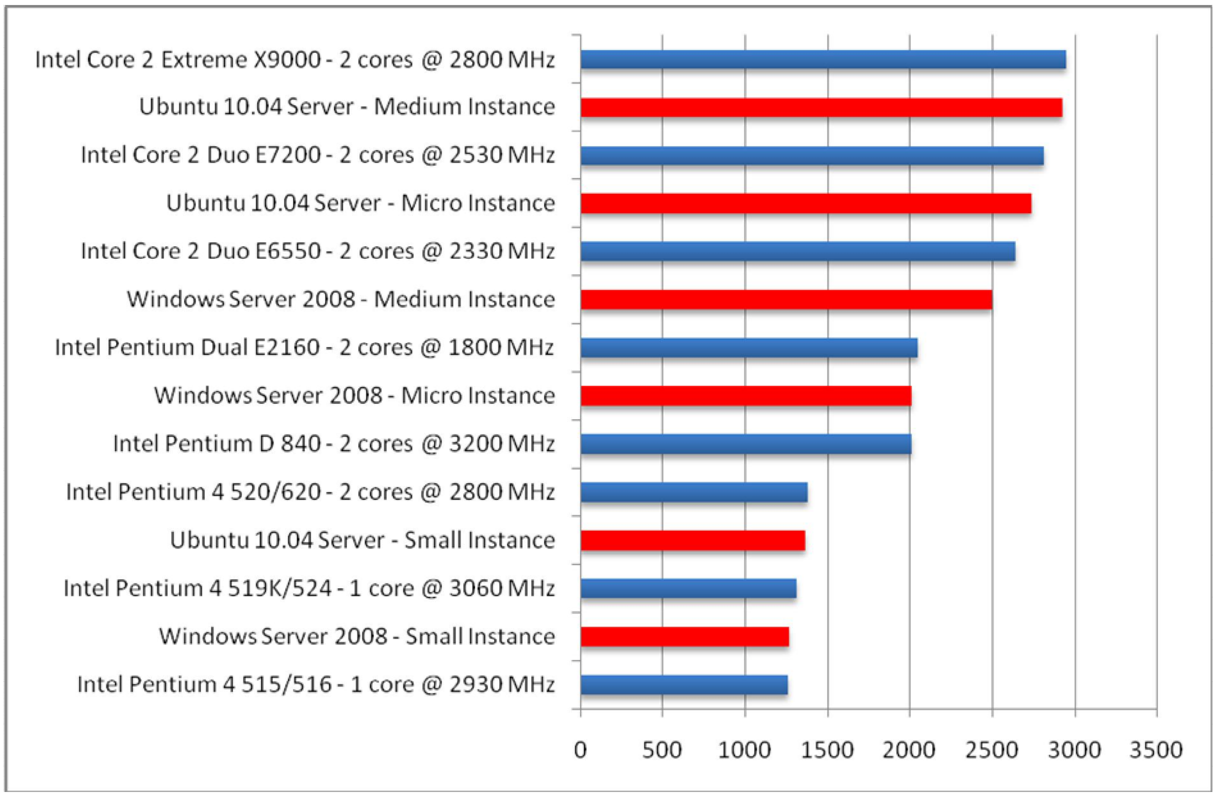


Figure 19 – CPU performances of Amazon instances compared to reel computer possessing Intel processors

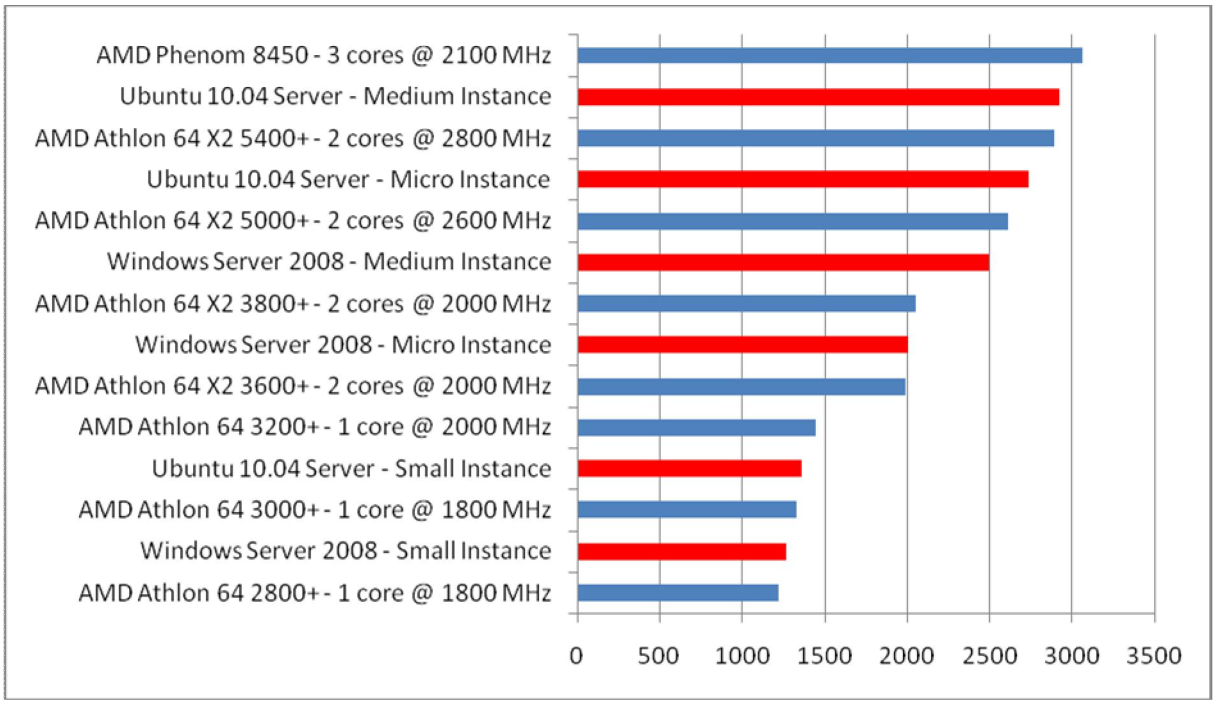


Figure 20 - CPU performances of Amazon instances compared to reel computer possessing AMD processors

6.3 Comparison of para-virtualization and HVM apply to the Amazon cloud.

6.3.1 Processor and memory

In this part it will be evaluated the process performance of integer and floating point unit as well as memory and memory stream. Process integer data consists of process whole numbers, text and the like. However when it comes to the process of Floating Point Unit (FPU) operations are more complicated than with integer. Examples of applications making a heavy use of FPU are spreadsheets, games, graphical applications and so on. Thus to be able to have good performance processors need to perform the process of integer and FPU as fast as possible.
The following results show the performances for process integer and FPU on the different instances of the Amazon cloud (Micro, Small and Medium). It was expecting to have lower performances for all the tests within HVM against paravirtualization however it occurs that HVM manage to perform integer processing a little bit faster than with the paravirtualization. Those differences observed are not really significant and the overall results for those tests remain relatively equal. For the test about the Floating Point Unit (FPU) where more complex calculations need to be performed it occurs that the HVM performances are significantly under the performances of para-virtualization. The degradation seems to increase with the power of the machine virtualized. It could be deduced that more powerful is the machine more the degradation between HVM and paravirtualization performance is important. However there is not enough information to confirm this theory. More tests will be required to be sure that the differences continue to grow. In term of processing it can be observe that the performances of HVM are slightly better than para-virtualization for integer process. However HVM have much lower performances than paravirtualization for processing FPU (Figure 21).
Many researches have been conducted on memory performances during the last few years (Adams et al, 2006; Agesen, 2009; Bhatia, 2009). Those researches had for aim to improve the performance of memory virtualization. According to the results obtained the memory performance of HVM are still significantly under paravirtualization performance. However it can be observe than for the small instance of Windows the performance are almost similar to the performance of Ubuntu. It has been noticed that the small instance of windows were not using the same type of processor than the other instances because all are running on Intel processor whereas the Windows small instance is running on an AMD processor (See appendix 5). Once again there is not enough information to conclude that the memory performances on AMD are better than on Intel processor. Also the previous researches tend to show the opposite (Bhatia, 2009). The memory bandwidth performance (also called stream by Geekbench) is almost the same on the three instance type studied. Also once again paravirtualization has slightly better than HVM.

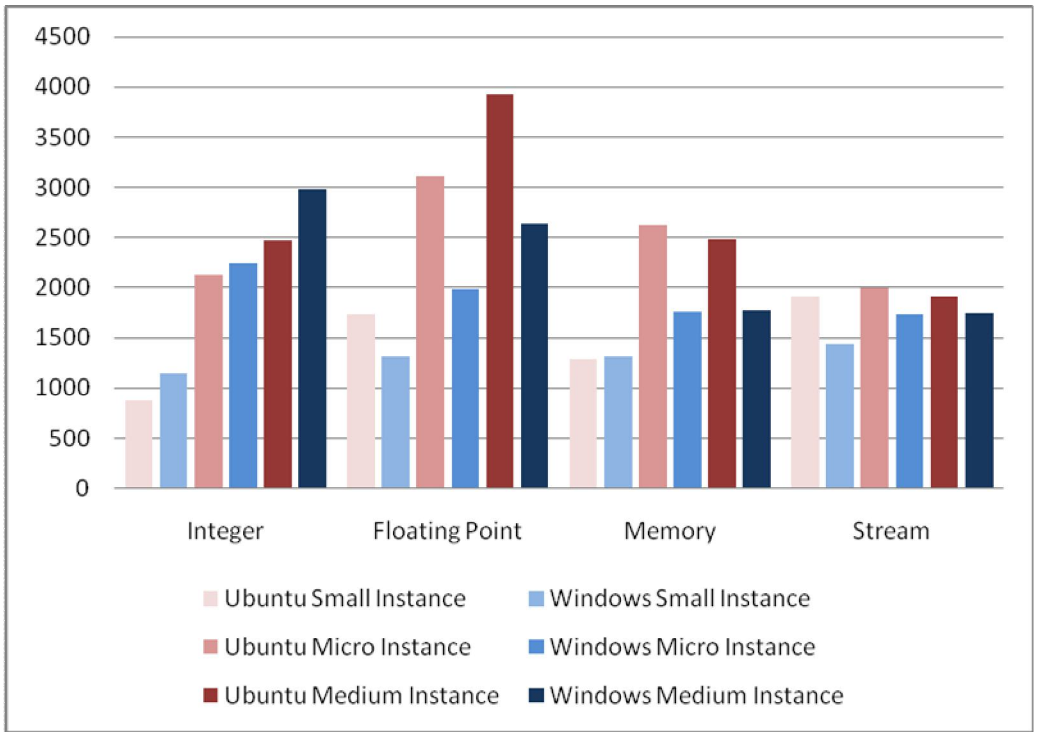
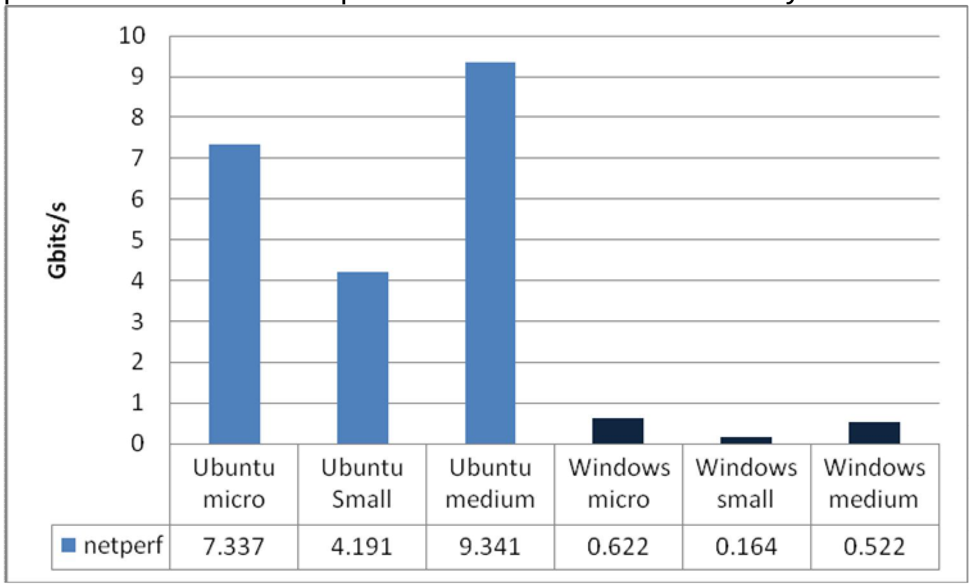


Figure 21 - Comparison of the performance for Integer, Floating Point, Memory and Memory Stream

6.3.2 Network

In the Amazon cloud the network and disk performances are indicated with low, moderate, high and very high (for cluster only). Because virtual machine can be used to realize test it has been made performance test on the loopback interface. This interface is a virtual network interface and the performance between Windows and Linux would be expected to be relatively similar. However the test shows that the performances of the loopback interface are considerably better in Linux (Figure 22).

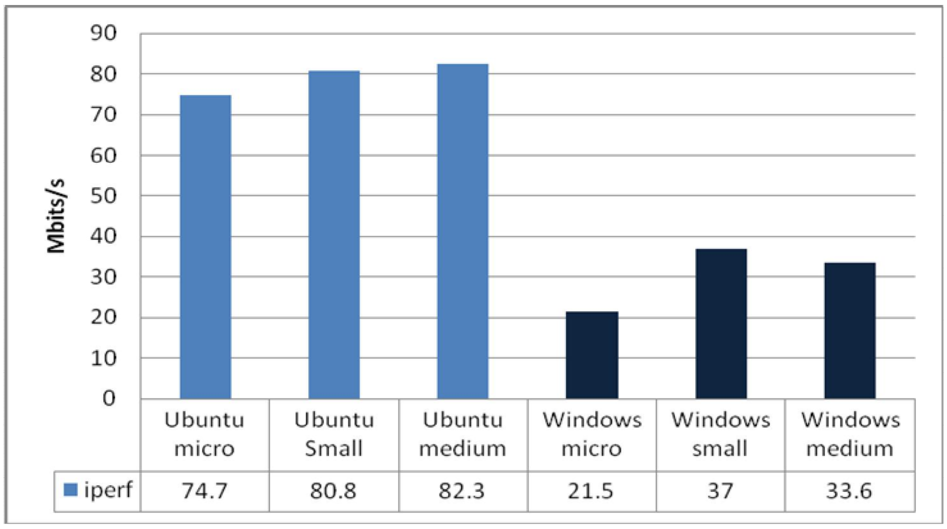


	Ubuntu micro	Ubuntu Small	Ubuntu medium	Windows micro	Windows small	Windows medium
■ netperf	7.337	4.191	9.341	0.622	0.164	0.522

Figure 22 - Performances of the loopback interface for Windows and Ubuntu into the Amazon cloud

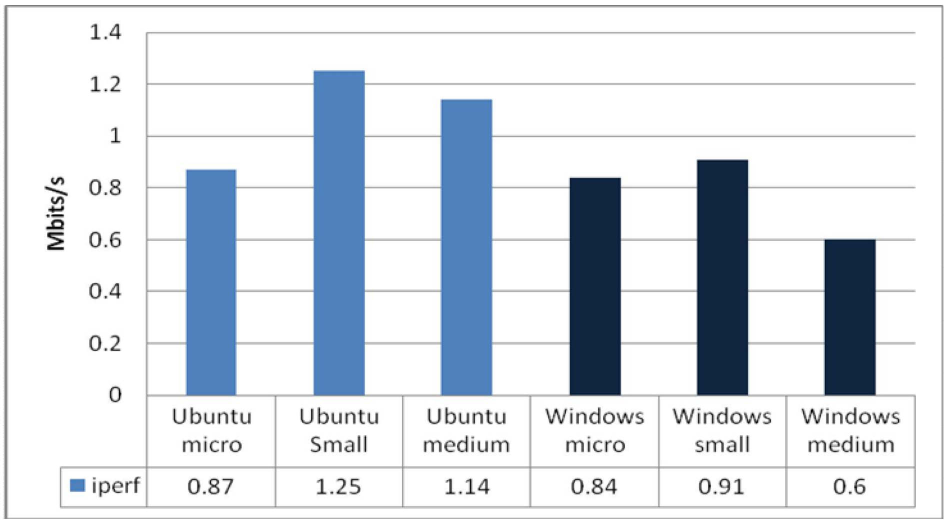
A hypothesis to explain those differences is to look at the implementation of the loopback for Linux and Windows. Because Windows and Linux implement differently the loopback the performances can differ. Also the virtualization performance of the paravirtualization and HVM should certainly have an impact on this difference of performance but probably not to involve such differences.

To continue within the network performance test it has been evaluated the speed of the network inside the Amazon cloud as well as the performances between the Amazon cloud and Napier (Figure 23 and 24). It can be observed that the performances inside the Amazon cloud are pretty good because customers can expect a bandwidth of about 80 Mbits/sec (10 MBytes/sec) for Linux and about 30 Mbits/sec (3.75MBytes/sec). The performance to reach another network decrease considerably because the bandwidth between Amazon and Napier is only of about 1 Mbits/sec (0.125Mbytes/sec) for Linux and about 0.80 Mbits/sec for Windows (0.1 Mbytes/sec). Those low performances can be explained by the fact that the machines used are located in East America.



	Ubuntu micro	Ubuntu Small	Ubuntu medium	Windows micro	Windows small	Windows medium
🟦 iperf	74.7	80.8	82.3	21.5	37	33.6

Figure 23 - Network performance inside the Amazon cloud



	Ubuntu micro	Ubuntu Small	Ubuntu medium	Windows micro	Windows small	Windows medium
🟦 iperf	0.87	1.25	1.14	0.84	0.91	0.6

Figure 24 - Network performance between Amazon and Napier

The disk performances in reading and writing represent an important part of the performances of the server. The timing buffered disk reads is the speed of the direct lecture on the disk. On Ubuntu the speed to read on the disk seems to be of about 60

Mbytes/sec. On windows the speed is better and reaches about 96.4 Mbytes/sec. The disk speed on Windows machine seems to be better than with Ubuntu.

	Ubuntu - Micro Instance	Ubuntu - Small Instance	Ubuntu - Medium Instance
Timing buffered disk reads (Mbytes/sec)	13.85	59.46	63.43

Figure 25 - Hard Disk speed on Linux

	Windows – Micro Instance	Windows – Small Instance	Windows – Medium Instance
Sequential Read (in Mbytes/sec)	20.3	94.6	96.4
Sequential Write (in Mbytes/sec)	27.3	54.7	55.4
Random Seek + Rewrite (in Mbytes/sec)	22.6	39.5	45.7

Figure 26 - Hard Disk speed on Windows

6.3.3 The OpenNebula Cloud

To measure the overhead of the CPU and the overhead of the memory many parameter will be tested such as the Integer processing, the FPU processing, the memory and memory stream. It has been build a private cloud running OpenNebula on the Front-End and Xen 3.4 to be able to perform tests. It has been compared the performances of reel machines with the performances of virtualization such as paravirtualization and HVM (Figure 27). As expected the general performances of the paravirtualization are better than HVM. The results shown that paravirtualization manages to perform really good virtualization of CPU and memory. HVM gives good results however the results are significantly under the performance of paravirtualization.

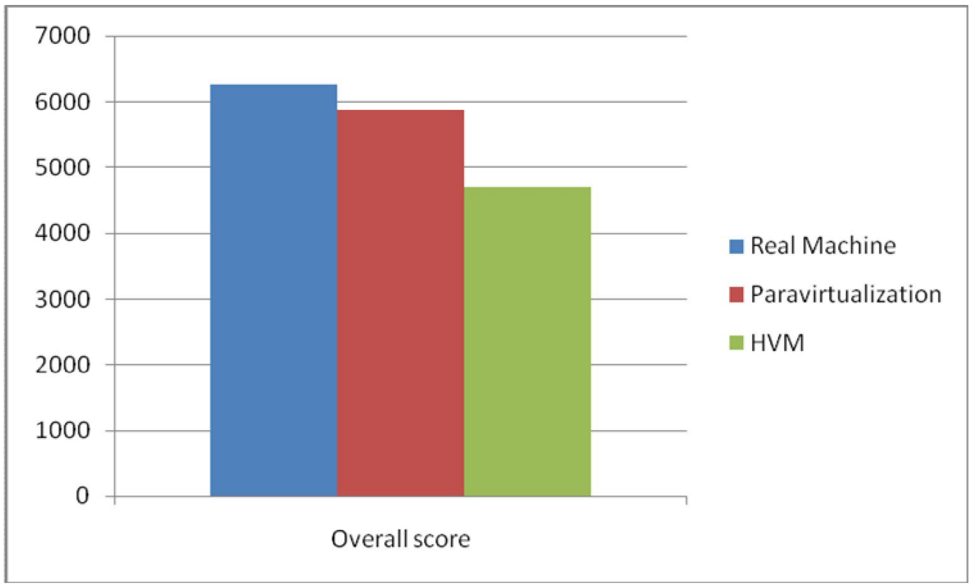


Figure 27 - Overall Geekbench score

In more detail it is observed that the performances for realized integer operations are almost equal between native performance and paravirtualization. However HVM performance occurs to be better than the native performances.
The results obtains prove that HVM manages to realize arithmetic without overhead and even better than the native performance. This improvement should come from the implementation of HVM to perform simple calculation. However this implementation of HVM in the way it improves performance with simple calculation have probably an impact on the on the performances of complex calculation which are considerably decreased in comparison to native performance. Also according to the Geekbench results there is a degradation of 36.5%. For the memory the overhead of paravirtualization is not very important. The overhead of the HVM for memory even if it has been improved during the past few years is still significantly under the performances of paravirtualization as well as native performances.

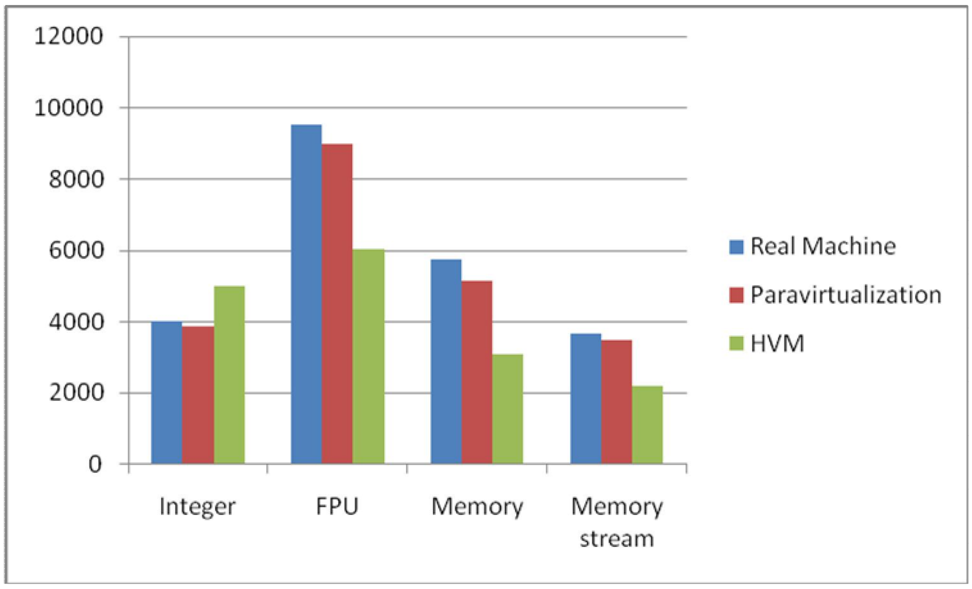


Figure 28 - Geekbench results by categories

Private cloud can demand to run a large quantity of machines on it. Also it has been determine the performance that can be expected when the number of machine increase. It has been run tests to see how evolve the CPU performance when the machine is in the node with multiple machine running at 100% of CPU utilisation. This test has been divided in three parts. In the first part the test will be made with one CPU allocated per machine then two CPU and finally four CPU allocated per machine. It can be observed that from 1 to 3 machines running on the cloud the performances are decreasing. After that it can be observed that the performances grow significantly (Figure 29). The CPU as well as the memory gain performances when the 4th machine is launch (Figure 29). The launch of the 4th machine force the use the all the core of the CPU. Also it can be though that some features of the CPU are unlock when all the core are used such as Intel® Turbo Boost Technology, Enhanced Intel SpeedStep® Technology and so on. According to Intel, The Enhanced Intel SpeedStep® Technology increase of performance because it manage voltage and frequency between high and low levels in function of the processor load. Also it can be seen that during the second test when there is only one machine with two CPU the performances are lower than when two machines are launched (Figure 30). During the first test when only one CPU per machine is used it can be observed that after the launch of eight machines the same performances than there were only one machine running can be expected. Even when there are six virtual machines running with each using the four cores of the processor the performances are almost similar to the performances that were found when there were six machines running with one core used per machine (Figure 31). From that it can be conclude that when the number of CPU available is reached Xen is still able to allocate CPU. However if the CPU is used the resources are equally shared between the machines.

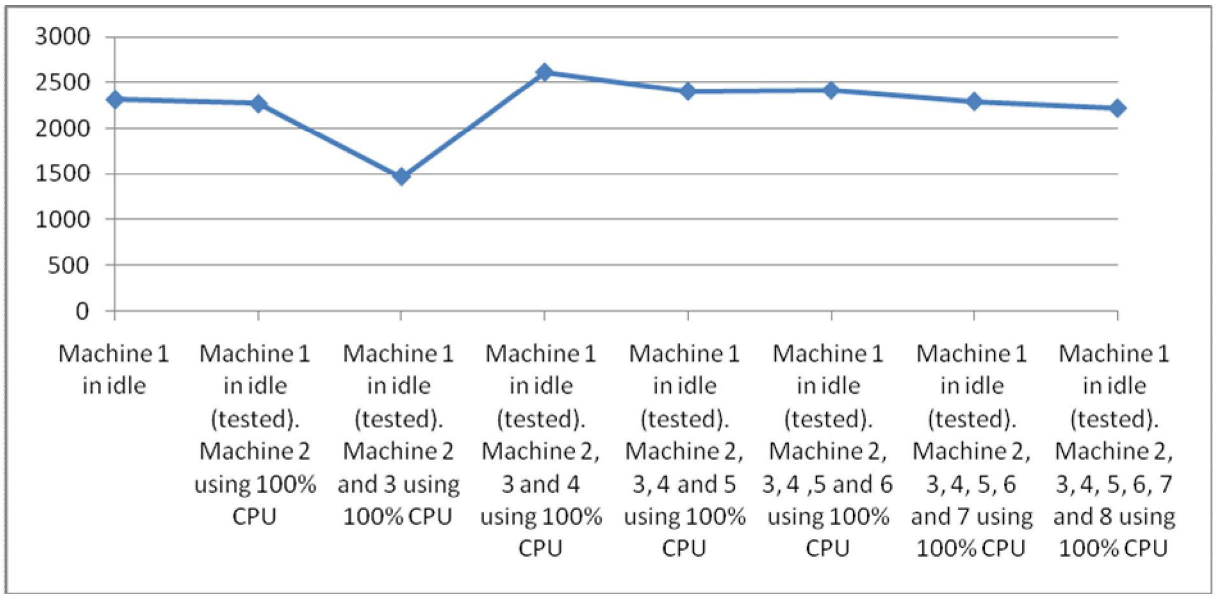


Figure 29 - Geekbench results with 1 core per machine

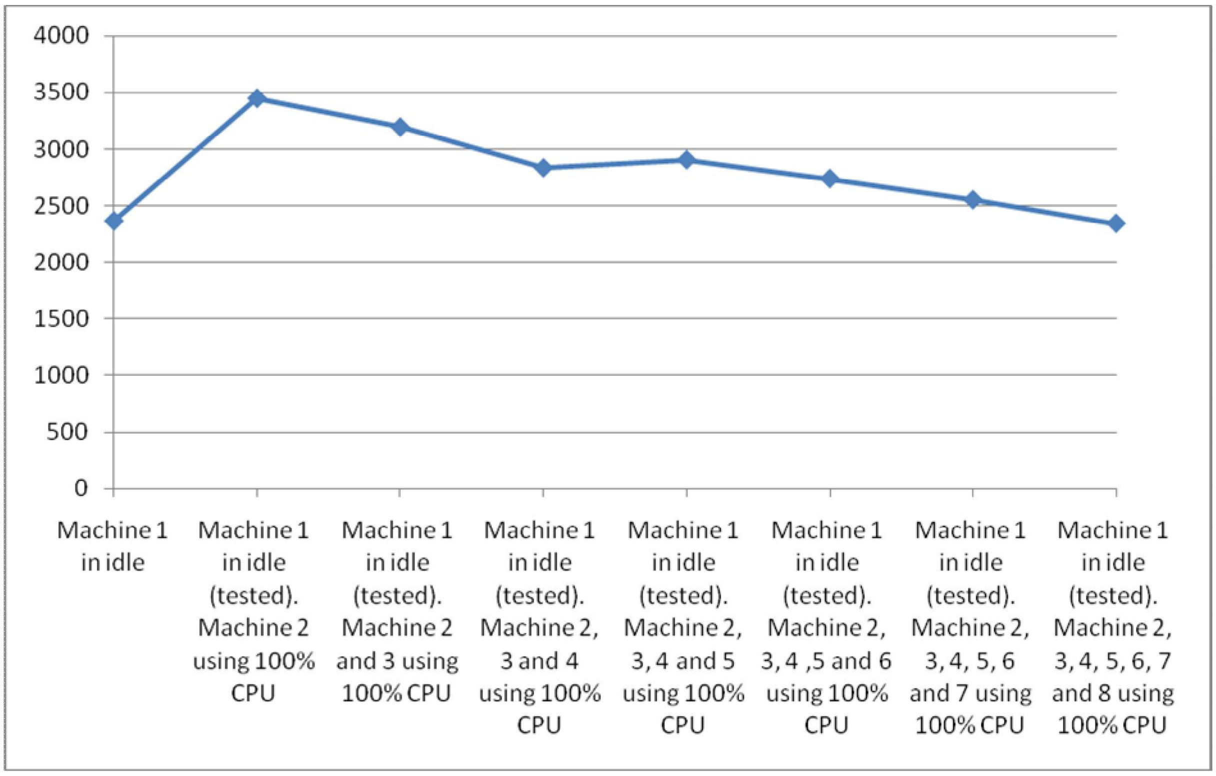


Figure 30 - Geekbench results with 2 cores per machine

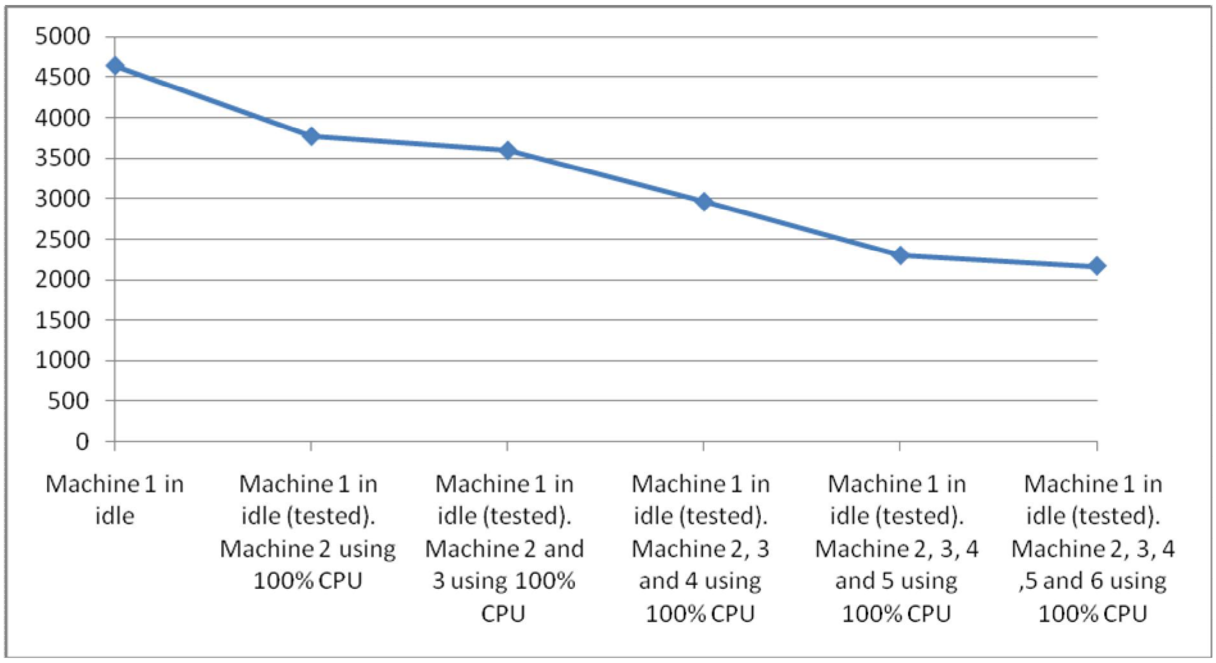


Figure 31 - Geekbench results with 4 cores per machine

Because of the charge of the network the router used to perform the test wasn't able to support the flood of the network. Also it has been unable to perform the test with more than two clients flooding the network. The use of the bandwidth should be theoretically be divided by the number of machines running on the cloud. The results observed (Figure 32) seems to correlate with the reality because when there is only one machine the bandwidth is of about 100 Mbits/sec which is normal because the switch used to connect the equipment together have maximum performances of 100Mbits/sec. When one machines is flooding the network the performances of the first machine decrease of almost 50% and when a second machine is flooding the network performances decreased again of about 50% (Figure 33). Inside the node the performances was much bigger than the performances found for reach a machine outside of the cloud. Also the performances are not decreasing in the same way. Network performances inside the node decrease of the 95% after the first flood. Inside the node the network between the machines is assured by a bridge connecting all the machines together.

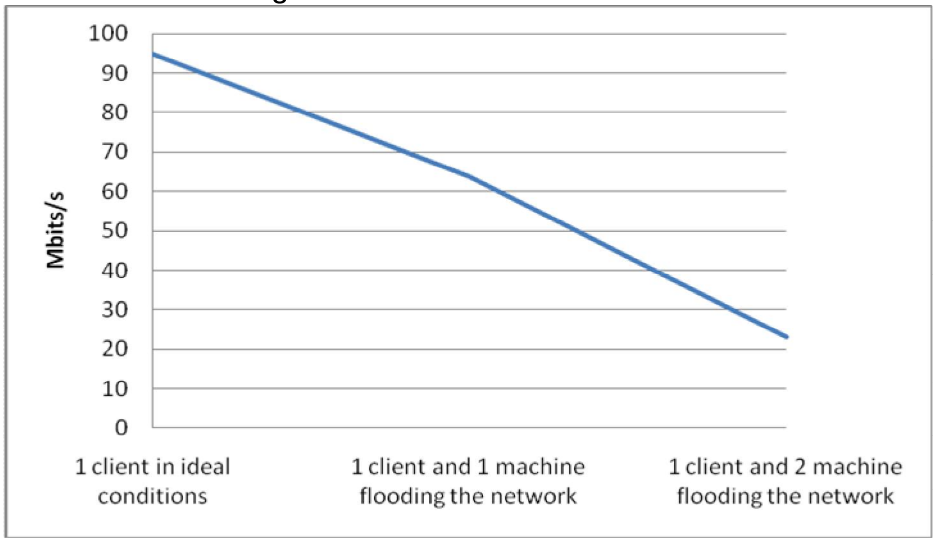


Figure 32 - Network performance inside the private cloud

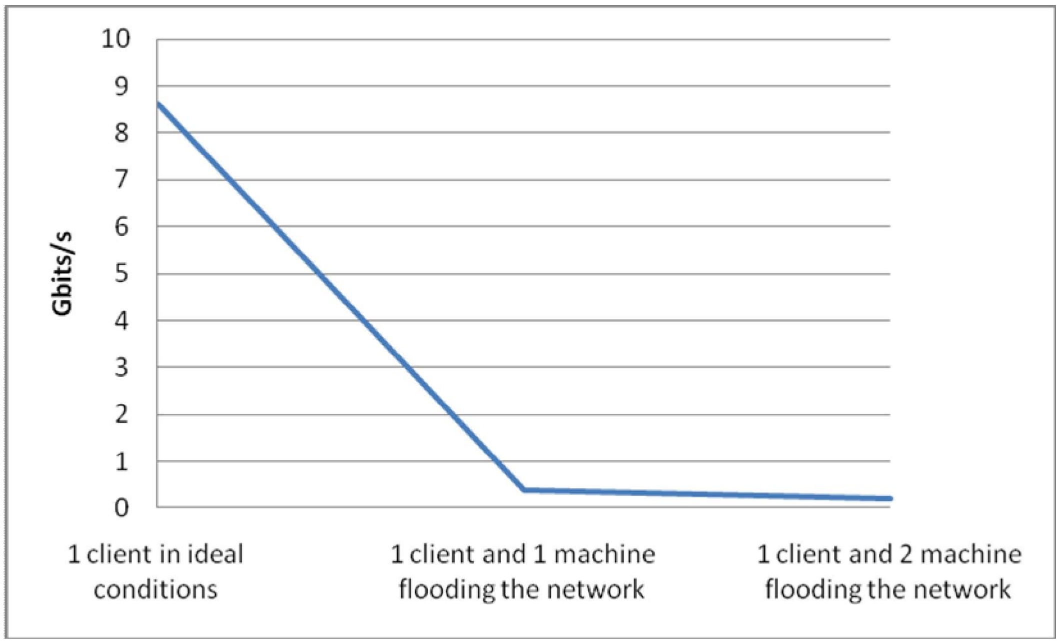


Figure 33 - Network performances from inside to outside

6.4 Comparison of public and private cloud

The performance of public and private cloud can similar however with public cloud there is less flexibility but more resources. Also a major difference between public cloud and private cloud is the price that they cost. For the public cloud users only pay for what they use rather than with private cloud because companies need to invest on the infrastructure to build the cloud and maintain the cloud. The advantage for company to have a private cloud is at the security level. Private clouds are deployed behind the company firewall thus the company have full control of the data transiting on the cloud.

The Price for a small instance on the Amazon cloud for one year will be $746 if the instance is used on an on-demand basis. However if this instance is reserved for 1 year it only cost $227.5. Thus if a company know that they will need to use a machine on a public cloud on a regular basis it is cheaper to reserve the instance for 1 year rather than to use it on-demand (Figure 34).

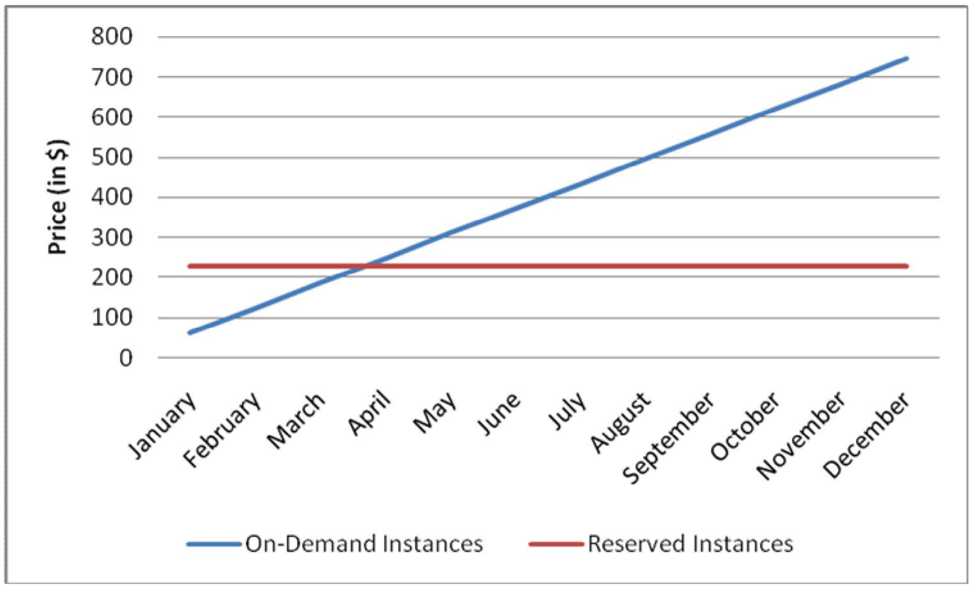


Figure 34 - Price for a small instance on the Amazon cloud

For the private cloud the price is more expensive because it is required to buy the hardware. For the OpenNebula Cloud it has cost approximately £1600 to buy the node and the frontend. To that it is required to add the price of at least one technician to build maintain the cloud. For the implementation of the private cloud made previously only open source solution have been used however if a commercial solution is use it is required to add that price as well. By comparison it can be reserve ten small instances during one year on the Amazon cloud only with the price of the hardware of the private cloud. Private clouds more expensive than public clouds however they offer a better security to the data. To be able to reduce the cost at a maximum the best is to implement a private cloud to manage sensible data and use a public cloud for non-sensible data.

6.5 Conclusion

Lots of results have been produced due to the evaluation of the public and private cloud. First with the Amazon cloud it has been outline the performances of the virtual machine hosted on Amazon compared to reel machine to show the performances which can be expected for a virtual machine. Also it has been outline that the performances for Windows and Linux were not the same because Linux was using paravirtualization while Windows was using HVM. With the private cloud it has been compared the native performances with paravirtualization and HVM. From that it has been seen that the overhead of the paravirtualization was really small. Contrary to that the overhead of HVM is more significant. The performance of the private cloud have shows that for the processor used on the node it was requiring to be fully utilized to be able to release full performances. Finally it has been conclude on the fact that public cloud are relatively cheaper than private cloud however private cloud assure more security over the data. The best solution to cloud computing would be the utilisation of hybrid cloud which are able to protect sensible data on a private cloud and use the public cloud to the use of non sensible data.

7 Conclusion

7.1 Introduction

The aim of this project was to evaluate the performances of virtualization with cloud computing. The performances have been evaluated for both private and public cloud. Also in the previous parts it has been designed and implemented experiment to be able to evaluate the performances of the virtualization within those two different types of cloud. In this chapter it will analysed how the objective have been accomplished and the difficulty encounter to be able to realize those objectives. Also a critical analysis of the entire project will be performed to outline the positives as well as the negative aspect of this project. Finally it will be proposed future work that could be done to continue the research in cloud computing and virtualization to be able to improve the capabilities of those technologies.

7.2 Meeting the objectives

It has been define at the beginning four objectives to the project:

- Critically review the different solutions available for cloud computing. Then it will be study the different virtualisation solutions both Open-Source and commercial that can be used with cloud computing. Finally it will be review the study about the performances of the different hypervisor.
- Design scenarios to be able to measure the performances of both public and private cloud. Also to be able to test the performance of a private cloud it will be required to design the implementation of a private cloud.
- Implement the different scenario to be able to evaluate the performances of public and private cloud as well as the implementation of the private cloud.
- Evaluate the performance of public and private cloud to be able to give an idea of the performance that can expect a customer of each service.

The first objective has been to review the previous work which has been done on that subject. To be able to review on the previous work it has been outline at first the main concept associate to virtualization and cloud computing in the technical review. After the technical review the literature review has permitted to critic the previous work on virtualization technology and cloud computing. The different articles review was showing that a lot of progress have been made to try to uniform cloud computing and improve virtualization performances. Both Intel and AMD try to bring hardware solution to improve hardware performances and Xen, VMware, KVM and so on try constantly to improve the performance of their hypervisor by reducing the overhead.

The second objective was to design scenario to be able to test the performances of the Amazon cloud as well as design a private cloud and the tests to measure its performances. To be able to analyse the performances of the Amazon cloud different experiment have been design to evaluate CPU, network and hard drive. Also it has been design a private cloud based on OpenNebula and Xen. For the design of the private cloud it has been use open source solution to be able to implement the cheapest possible solution for the cloud. For testing the performances of the private cloud experiments have been setup to be able to evaluate its performances.

The third objective was the implementation of the different experiments and also the implementation of the private cloud. The implementation required to learn how was working the Amazon cloud and how to implement the different tools to be able to perform the test on each instances. The choice and the implementation of the different tools was an issue because tools needed to be compatible with both Windows and Linux to be able to compare the results obtains with accuracy. Then the implementation of the private cloud has been an important work because it was required to implement a complete cloud solution. Also because the implementation was based on Open Source solution it was sometime difficult to find documentation on the different problems that have been met during the implementation. However it has been manage to solve all the problems encounter during the implementation. Finally the implementation of the tests on the private cloud has been made to evaluate the performance of the private cloud.

The last objective was to evaluate the performances of both Amazon and the private cloud. It has been evaluated at first the performances of the Amazon cloud for different types of instances. The different instances have been tested for CPU performances, network performances as well as hard drive performances. Also it has been evaluate the difference of performances between paravirtualization and HVM. After having evaluated the performances of the Amazon cloud it has been perform an evaluation of the private cloud. For the private cloud it has been compare native performance to paravitualization and HVM. Also some tests have been performed to evaluate the limits of the private cloud. Finally it has been discussed of the cost of public and private cloud.

7.3 Conclusion

The evaluation of the virtualization with cloud computing has showed that virtualization performances are improving and tend to be as good as the native performances. Also the researches permit to outline the strengths and weaknesses of the different technologies used in cloud computing. Due to that solutions are found to overcome the weaknesses. Also cloud computing and virtualization became more and more popular.

The main reasons to explain the popularity of cloud computing is that it offers the possibility to reduce cost. Also it offers relatively good performance and due to the researches performances continues to improve. In addition to that the effort to uniform cloud computing make it easier to use and tend to make cloud compatible between them to facilitate the deployment of hybrid cloud. Hybrid cloud seems to be the best alternative to cloud computing to be able to secure sensible data on the private cloud and allow non-sensible data to use public cloud.

7.4 Critical analysis

It has been managed to achieve the aims and objectives of this project. It has been represented the general performances of three basic instances of the Amazon cloud. The performances tests on the Amazon cloud have been perform on both Linux and Windows and have showed that Linux performances was better because Linux was using paravirtualization while Windows was using HVM. The Amazon cloud wasn't using the same hardware with all the machines tested. More experiment may have permit to see if the performances was changing depending on the hardware hosting the virtual instance. The Amazon cloud has been tested for CPU performances, Network performances and Hard Drive performances. The results obtains for the Hard Drive performances to compared Windows and Linux was not really accurate because two different software have been require to test Windows and Linux performances. Also it could be good to implement a software to be able to test hard drive performances on both Windows and Linux.

The implementation of the private cloud has been complicated because of the lack of documentation on the Open Source tools used to implement it. However due to the implementation of the private cloud many test have been performed to outline the performances. It has been compared the performance between native, paravirtualization and HVM. From this test it came out that the paravirtualization tend to have performance almost as good as the native performances. Also HVM performances are still under the performance of paravirtualization but due to the constant progresses it manages to have good enough performances. The tests to evaluate the degradation of CPU performances have only been performed on Windows. This it will be interesting to see it the same degradation is observed with the utilisation of the paravirtualization.

7.5 Future work

After all the tests that have been performed on cloud computing it will be interesting to evaluate in more detail the cost of a private cloud for a company or a university such as Napier and evaluate the economy that the usage of cloud computing could bring. It would require to evaluate the materiel needed to be able to virtualize a large number of machines and the price that it would cost. Also it would required to study the cost of solution that would be implemented such as the comparison of Open Source and commercial solution. Study the cost of the implementation in terms of employees and price of the maintenance of the service. Finally it would be necessary to evaluate the economy in terms of energy because the usage of cloud computing permit through the use of virtualization to reduce the energy consumption.

References

Adams, K. & Agesen, O. (2006) A Comparison of Software and Hardware Techniques for x86 Virtualization.

Agesen, O. (2009) Software and Hardware Techniques for x86 Virtualization.

Ahronovitz, M. et Al. (2010) Cloud Computing Use Cases. A white paper produced by the Cloud Computing Use Case Discussion Group Version 4.0 (2 July 2010). Creative Commons Attribution-Share Alike 3.0 Unported License.

Amazon (2010) Amazon EC2 Service Level Agreement. Effective Date: October 23, 2008. http://aws.amazon.com/ec2-sla/ retrieved the 03/11/2010.

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

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

Armbrust, M. & Fox, A. & Griffith, R.& Joseph, A. D. & Katz, R. & Konwinski, A. & Lee, G. & Patterson, D. & Rabkin, A. & Stoica, I.& Zaharia, M. (2010) A View of Cloud Computing. The Digital Library is published by the Association for Computing Machinery.

Bhatia, N (2009) Performance Evaluation of AMD RVI Hardware Assist. VMware.

Bhatia, N (2009) Performance Evaluation of Intel EPT Hardware Assist. VMware.

Brodkin, J (2008) Gartner: Seven cloud-computing security risks. Network World.

Brooks, M. (2008) A Model of Virtualization. October 2008. 2008 Dell, Inc.

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

Che, J.& He, Q. & Gao, Q. & Huang, D. (2009) Performance Measuring and Comparing of Virtual Machine Monitors.

Cisco Cloud Computing. (2009). Data Center Strategy, Architecture, and Solutions. Point of View White Paper for U.S. Public Sector. 1st Edition.

Creasy, R. J. (1981) The Origin of the VM/370 Time-Sharing System. IBM *Journal of Research and Development*, 25(5):483–490, September 1981

Endo, P. T. & Gonçalves, G. E. & Kelner, J. & Sadok, D. (2010) A Survey on Open-source Cloud Computing Solutions. Universidade Federal de Pernambuco – UFPE.

Fox, A. (2010) The Potential of Cloud Computing: Opportunities and Challenges.
Gabrielsson, J. & Karlsson, P. & Skog, R. (2009) The cloud opportunity. Sony Ericson.

Goldberg, R. P. (1974) Survey of virtual machine research. *IEEE Computer Magazine*, 7(6):34–45, 1974

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

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

Helft, M. (2009) Google confirms problems with reaching its services.

Intel Corporation. (1986) Intel 80386 programmer's reference manual. Intel Corporation 1987 CG-5/26/87.

Intel Corporation. (2008) Enabling Dynamic Virtual Client Computing with Intel vPro Technology. Intel Technology Journal, Volume 12 Issue 04 published December 23, 2008

Isogai, M. & Funabiki, N. & Nakanishi, T. & Isshiki, Y. (2009) A node configuration algorithm with bandwidth bottleneck resolution for WDM ring networks.

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

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

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

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

Krebs, B. (2008) Amazon: Hey spammers, Get off my cloud! Washington Post.

Lombardi, F. & Di Pietro, R. (2010). Transparent security for cloud. In SAC '10: Proceedings of the 2010 ACM symposium on Applied Computing, New York, NY, USA, 2010. ACM.

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

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

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

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

Moore, G. E. (1965) Cramming more components onto integrated circuits, Electronics, Volume 38, Number 8, April 19, 1965.

Open cloud manifesto (2009) Open Cloud Manifesto - Dedicated to the belief that the cloud should be open. Creative Commons Attribution-Share Alike 3.0 Unported License.

OpenNebula (2010) http://www.opennebula.org/ information retrieved in august 2010.

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

Rosenblum, M. & Garfinkel, T. (2005) Virtual machine monitors: current technology and future trends. Computer, 16 May 2005.

Schader, M. (2010) Hybrid cloud: Comparing toolkit. Chair in Information Systems. April 2010.

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

Sotomayor, B. & Montero, R. S. & Llorente, I. M. & Foster, I. (2007) Capacity Leasing in Cloud Systems using the OpenNebula Engine.

Sotomayor, B. & Montero, R.S. & Llorente, I.M. & Foster, I. (2009) Resource Leasing and the Art of Suspending Virtual Machines.

Sugerman, J. & Venkitachalam, G. & Lim, B. H. (2001) Virtualizing I/O Devices on VMware Workstation’s Hosted Virtual Machine Monitor. VMware, Inc.

Uhlig, R. & Neiger, G.& Rodgers, D. & Santoni, A.L. & Martins, F. C. M. & Anderson, A. V. & Bennett, S. M. & Kagi, A. & Leung, F. H. & Smith, L. (2005) Intel virtualization technology. Computer, 38(5):48–56, 2005.

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

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

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

Wang, G. & Ng, T. S. E. (2010) The Impact of Virtualization on Network Performance of Amazon EC2 Data Center. INFOCOM, 2010 Proceedings IEEE.

Weiping, L. (2009). An analysis of new features for workflow system in the SaaS software. ICIS 2009, November 24-26, 2009 Seoul, Korea.

Weltzin, C. & Delgado, S. (2009) Using virtualization to reduce the cost of test. AUTOTESTCON, 2009 IEEE.

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

Youseff, L. & Seymour, K. & You, H. & Dongarra, J. & Wolski, R. (2008) The Impact of Paravirtualized Memory Hierarchy on Linear Algebra Computational Kernels and Software.

Youseff, L., & Butrico, M., & Da Silva, D. (2008). Toward a Unified Ontology of Cloud Computing. In Grid Computing Environments Workshop, 2008. GCE \'08 (Nov 2008), pp. 1-10.

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

Appendix 1 – Different instance type available within the Amazon cloud

Standard Instances

Instances of this family are well suited for most applications.

Small Instance – default*

1.7 GB memory
1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit)
160 GB instance storage
32-bit platform
I/O Performance: Moderate
API name: m1.small

Large Instance

7.5 GB memory
4 EC2 Compute Units (2 virtual cores with 2 EC2 Compute Units each)
850 GB instance storage
64-bit platform
I/O Performance: High
API name: m1.large

Extra Large Instance

15 GB memory
8 EC2 Compute Units (4 virtual cores with 2 EC2 Compute Units each)
1,690 GB instance storage
64-bit platform
I/O Performance: High
API name: m1.xlarge

Micro Instances

Instances of this family provide a small amount of consistent CPU resources and allow you to burst CPU capacity when additional cycles are available. They are well suited for lower throughput applications and web sites that consume significant compute cycles periodically.

Micro Instance

613 MB memory
Up to 2 EC2 Compute Units (for short periodic bursts)
EBS storage only
32-bit or 64-bit platform
I/O Performance: Low
API name: t1.micro

High-Memory Instances

Instances of this family offer large memory sizes for high throughput applications, including database and memory caching applications.

High-Memory Extra Large Instance

17.1 GB of memory
6.5 EC2 Compute Units (2 virtual cores with 3.25 EC2 Compute Units each)
420 GB of instance storage
64-bit platform
I/O Performance: Moderate
API name: m2.xlarge

High-Memory Double Extra Large Instance

34.2 GB of memory
13 EC2 Compute Units (4 virtual cores with 3.25 EC2 Compute Units each)
850 GB of instance storage
64-bit platform
I/O Performance: High
API name: m2.2xlarge

High-Memory Quadruple Extra Large Instance

68.4 GB of memory
26 EC2 Compute Units (8 virtual cores with 3.25 EC2 Compute Units each)
1690 GB of instance storage
64-bit platform
I/O Performance: High
API name: m2.4xlarge

High-CPU Instances

Instances of this family have proportionally more CPU resources than memory (RAM) and are well suited for compute-intensive applications.

High-CPU Medium Instance

1.7 GB of memory
5 EC2 Compute Units (2 virtual cores with 2.5 EC2 Compute Units each)
350 GB of instance storage
32-bit platform
I/O Performance: Moderate
API name: c1.medium

High-CPU Extra Large Instance

7 GB of memory
20 EC2 Compute Units (8 virtual cores with 2.5 EC2 Compute Units each)
1690 GB of instance storage
64-bit platform
I/O Performance: High
API name: c1.xlarge

Cluster Compute Instances

Instances of this family provide proportionally high CPU resources with increased network performance and are well suited for High Performance Compute (HPC) applications and other demanding network-bound applications. Learn more about use of this instance type for HPC applications.

Cluster Compute Quadruple Extra Large Instance

23 GB of memory
33.5 EC2 Compute Units (2 x Intel Xeon X5570, quad-core “Nehalem” architecture)
1690 GB of instance storage
64-bit platform
I/O Performance: Very High (10 Gigabit Ethernet)
API name: cc1.4xlarge

Appendix 2 – OpenNebula Configuration file oned.conf

```
#*******************************************************************************
#                       OpenNebula Configuration file
#*******************************************************************************

#*******************************************************************************
# Daemon configuration attributes
#-------------------------------------------------------------------------------
#  HOST_MONITORING_INTERVAL: Time in seconds between host monitorization
#
#  VM_POLLING_INTERVAL: Time in seconds between virtual machine monitorization
#
#  VM_DIR: Remote path to store the VM images, it should be shared between all
#  the cluster nodes to perform live migrations. This variable is the default
#  for all the hosts in the cluster.
#
#  PORT: Port where oned will listen for xmlrpc calls.
#
#  DEBUG_LEVEL: 0 = ERROR, 1 = WARNING, 2 = INFO, 3 = DEBUG
#*******************************************************************************

HOST_MONITORING_INTERVAL = 60

VM_POLLING_INTERVAL      = 60

#VM_DIR=/srv/cloud/one/var

PORT=2633

DEBUG_LEVEL=3

#*******************************************************************************
# Physical Networks configuration
#*******************************************************************************
#  NETWORK_SIZE: Here you can define the default size for the virtual networks
#
#  MAC_PREFIX: Default MAC prefix to be used to create the auto-generated MAC
#  addresses is defined here (this can be overrided by the Virtual Network
#  template)
#*******************************************************************************

NETWORK_SIZE = 254

MAC_PREFIX   = "00:03"

#*******************************************************************************
# Information Driver Configuration
#*******************************************************************************
# You can add more information managers with different configurations but make
# sure it has different names.
#
#   name      : name for this information manager
#
#   executable: path of the information driver executable, can be an
#               absolute path or relative to $ONE_LOCATION/lib/mads (or
#               /usr/lib/one/mads/ if OpenNebula was installed in /)
#
#   arguments : for the driver executable, usually a probe configuration file,
#               can be an absolute path or relative to $ONE_LOCATION/etc (or
#               /etc/one/ if OpenNebula was installed in /)
#*******************************************************************************

IM_MAD = [
    name       = "im_xen",
    executable = "one_im_ssh",
    arguments  = "im_xen/im_xen.conf" ]

#-------------------------------------------------------------------------------
#  KVM Information Driver Manager sample configuration
#-------------------------------------------------------------------------------
IM_MAD = [
      name       = "im_kvm",
      executable = "one_im_ssh",
      arguments  = "im_kvm/im_kvm.conf" ]
#-------------------------------------------------------------------------------

#-------------------------------------------------------------------------------
#  EC2 Information Driver Manager sample configuration
#-------------------------------------------------------------------------------
#IM_MAD = [
#      name       = "im_ec2",
#      executable = "one_im_ec2",
#      arguments  = "im_ec2/im_ec2.conf" ]
#-------------------------------------------------------------------------------

#*******************************************************************************
# Virtualization Driver Configuration
#*******************************************************************************
# You can add more virtualization managers with different configurations but
# make sure it has different names.
#
#   name      : name of the virtual machine manager driver
#
#   executable: path of the virtualization driver executable, can be an
#               absolute path or relative to $ONE_LOCATION/lib/mads (or
#               /usr/lib/one/mads/ if OpenNebula was installed in /)
#
#   arguments : for the driver executable
#
#   default   : default values and configuration parameters for the driver, can
#               be an absolute path or relative to $ONE_LOCATION/etc (or
#               /etc/one/ if OpenNebula was installed in /)
#
#   type      : driver type, supported drivers: xen, kvm, xml
#*******************************************************************************

VM_MAD = [
    name       = "vmm_xen",
    executable = "one_vmm_xen",
    default    = "vmm_xen/vmm_xen.conf",
    type       = "xen" ]

#-------------------------------------------------------------------------------
#  KVM Virtualization Driver Manager sample configuration
#-------------------------------------------------------------------------------
VM_MAD = [
    name       = "vmm_kvm",
    executable = "one_vmm_kvm",
    default    = "vmm_kvm/vmm_kvm.conf",
    type       = "kvm" ]
#-------------------------------------------------------------------------------

#-------------------------------------------------------------------------------
#  EC2 Virtualization Driver Manager sample configuration
#    arguments: default values for the EC2 driver, can be an absolute path or
#               relative to $ONE_LOCATION/etc (or /etc/one/ if OpenNebula was
#               installed in /).
#-------------------------------------------------------------------------------
#VM_MAD = [
#    name       = "vmm_ec2",
#    executable = "one_vmm_ec2",
#    arguments  = "vmm_ec2/vmm_ec2.conf",
#    type       = "xml" ]
#-------------------------------------------------------------------------------

#*******************************************************************************
# Transfer Manager Driver Configuration
#*******************************************************************************
# You can add more transfer managers with different configurations but make
# sure it has different names.
#   name      : name for this transfer driver
#
#   executable: path of the transfer driver executable, can be an
#               absolute path or relative to $ONE_LOCATION/lib/mads (or
#               /usr/lib/one/mads/ if OpenNebula was installed in /)
#
#   arguments : for the driver executable, usually a commands configuration file
#               , can be an absolute path or relative to $ONE_LOCATION/etc (or
#               /etc/one/ if OpenNebula was installed in /)
#*******************************************************************************

TM_MAD = [
    name       = "tm_ssh",
    executable = "one_tm",
    arguments  = "tm_ssh/tm_ssh.conf" ]

#-------------------------------------------------------------------------------
# NFS Transfer Manager Driver sample configuration
#-------------------------------------------------------------------------------
TM_MAD = [
    name       = "tm_nfs",
    executable = "one_tm",
    arguments  = "tm_nfs/tm_nfs.conf" ]
#-------------------------------------------------------------------------------

#-------------------------------------------------------------------------------
# Dummy Transfer Manager Driver sample configuration
#-------------------------------------------------------------------------------
#TM_MAD = [
#    name       = "tm_dummy",
#    executable = "one_tm",
#    arguments  = "tm_dummy/tm_dummy.conf" ]
#-------------------------------------------------------------------------------

#-------------------------------------------------------------------------------
# LVM Transfer Manager Driver sample configuration
#-------------------------------------------------------------------------------
#TM_MAD = [
#    name       = "tm_lvm",
#    executable = "one_tm",
#    arguments  = "tm_lvm/tm_lvm.conf" ]
#-------------------------------------------------------------------------------

#*******************************************************************************
# Hook Manager Configuration
#*******************************************************************************
# The Driver (HM_MAD), used to execute the Hooks
#   executable: path of the hook driver executable, can be an
#               absolute path or relative to $ONE_LOCATION/lib/mads (or
#               /usr/lib/one/mads/ if OpenNebula was installed in /)
#
#   arguments : for the driver executable, can be an absolute path or relative
#               to $ONE_LOCATION/etc (or /etc/one/ if OpenNebula was installed
#               in /)
#
# Virtual Machine Hooks (VM_HOOK) defined by:
#   name      : for the hook, useful to track the hook (OPTIONAL)
#   on        : when the hook should be executed,
#               - CREATE, when the VM is created (onevm create)
#               - RUNNING, after the VM is successfully booted
#               - SHUTDOWN, after the VM is shutdown
#               - STOP, after the VM is stopped (including VM image transfers)
#               - DONE, after the VM is deleted or shutdown
#   command   : use absolute path here
#   arguments : for the hook. You can access to VM template variables with $
#               - $ATTR, the value of an attribute e.g. $NAME or $VMID
#               - $ATTR[VAR], the value of a vector e.g. $NIC[MAC]
#               - $ATTR[VAR, COND], same of previous but COND select between
#                 multiple ATTRs e.g. $NIC[MAC, NETWORK="Public"]
#   remote    : values,
#               - YES, The hook is executed in the host where the VM was
#                 allocated
#               - NO, The hook is executed in the OpenNebula server (default)
#-------------------------------------------------------------------------------
HM_MAD = [
    executable = "one_hm" ]

#-------------------------------- Hook Examples --------------------------------
#VM_HOOK = [
#    name      = "dhcp",
#    on        = "create",
#    command   = "/bin/echo",
#    arguments = "$NAME > /tmp/test.$VMID" ]
#-------------------------------------------------------------------------------
#VM_HOOK = [
#    name      = "ebtables",
#    on        = "running",
#    command   = "/usr/local/one/bin/set_net",
#    arguments = '$NIC[MAC, Network = "Private"]',
#    remote    = "yes" ]
#-------------------------------------------------------------------------------
#VM_HOOK = [
#    name      = "mail",
#    on        = "running",
#    command   = "/usr/local/one/bin/send_mail",
#    arguments = "$VMID $NAME",
#    remote    = "no" ]
#------------------------------------------------------------------------------
```

Appendix 3 – Debian Template

```
NAME = "Debian"

#The number of CPU use by the virtual instance can be specified here
CPU    = 1

#The number of memory use by the virtual instance can be specified here
MEMORY = 1024

# --- kernel & boot device ---
OS = [
  kernel   = "/srv/cloud/images/debian/vmlinuz-2.6.32-3-686-bigmem",
  initrd   = "/srv/cloud/images/debian/initrd.img-2.6.32-3-686-bigmem",
  root     = "xvda1"
]

# --- 1 disks ---
DISK = [
  source     = /srv/cloud/images/debian/debian.img",
  target     = "xvda1",
  readonly   = "no",
  save = "yes"
]

# --- 1 NIC ---
NIC = [ NETWORK="Network" ]

# --- VNC server ---
GRAPHICS = [
  type    = "vnc",
  vnclisten="192.168.1.4",
  port    = "1"
]
```

Appendix 4 - Windows Template

```
NAME = "Windows Server 2008"

#The number of CPU use by the virtual instance can be specified here
CPU    = 1

#The number of memory use by the virtual instance can be specified here
MEMORY = 1024

# --- kernel & boot device ---
OS = [ kernel = "/usr/lib/xen/boot/hvmloader" ]

# --- 1 disks ---
DISK = [
  source     = /srv/cloud/images/Windows_Server/Windows_Server.img",
  target     = "sda",
  readonly   = "no",
  save = "yes"
]

# --- 1 NIC ---
NIC = [ NETWORK="Network" ]

# --- VNC server ---
GRAPHICS = [
  type    = "vnc",
  vnclisten="192.168.1.4",
  port    = "2"
]

RAW = [type ="xen",data="builder='hvm'"]
```

Appendix 5 – Hardware of the machine tested on the Amazon cloud

The hardware of the machines has been outline with Geekbench.

Ubuntu Micro instance:

```
System Information
  Platform:             Linux x86 (32-bit)
  Compiler:             GCC 4.1.2 20070925 (Red Hat 4.1.2-33)
  Operating System:     Ubuntu 10.04.1 LTS 2.6.32-308-ec2 i686
  Model:                Linux PC (Intel Xeon E5430)
  Motherboard:          Unknown Motherboard
  Processor:            Intel Xeon E5430
  Processor ID:         GenuineIntel Family 6 Model 23 Stepping 10
  Logical Processors:   1
  Physical Processors:  1
  Processor Frequency:  2.66 GHz
  L1 Instruction Cache:  0.00 B
  L1 Data Cache:        0.00 B
  L2 Cache:             6.00 MB
  L3 Cache:             0.00 B
  Bus Frequency:        0.00 Hz
  Memory:               615 MB
  Memory Type:          N/A
  SIMD:                 1
  BIOS:                 N/A
  Processor Model:      Intel Xeon E5430
  Processor Cores:      1
```

Ubuntu Small instance:

```
System Information
  Platform:             Linux x86 (32-bit)
  Compiler:             GCC 4.1.2 20070925 (Red Hat 4.1.2-33)
  Operating System:     Ubuntu 10.04.1 LTS 2.6.32-308-ec2 i686
  Model:                Linux PC (Intel Xeon E5430)
  Motherboard:          Unknown Motherboard
  Processor:            Intel Xeon E5430
  Processor ID:         GenuineIntel Family 6 Model 23 Stepping 10
  Logical Processors:   1
  Physical Processors:  1
  Processor Frequency:  2.66 GHz
  L1 Instruction Cache: 0.00 B
  L1 Data Cache:        0.00 B
  L2 Cache:             6.00 MB
  L3 Cache:             0.00 B
  Bus Frequency:        0.00 Hz
  Memory:               1.66 GB
  Memory Type:          N/A
  SIMD:                 1
  BIOS:                 N/A
  Processor Model:      Intel Xeon E5430
  Processor Cores:      1
```

Ubuntu Medium instance:

```
System Information
  Platform:             Linux x86 (32-bit)
  Compiler:             GCC 4.1.2 20070925 (Red Hat 4.1.2-33)
  Operating System:     Ubuntu 10.04.1 LTS 2.6.32-308-ec2 i686
  Model:                Linux PC (Intel Xeon E5410)
  Motherboard:          Unknown Motherboard
  Processor:            Intel Xeon E5410
  Processor ID:         GenuineIntel Family 6 Model 23 Stepping 10
  Logical Processors:   2
  Physical Processors:  2
  Processor Frequency:  2.33 GHz
  L1 Instruction Cache: 0.00 B
  L1 Data Cache:        0.00 B
  L2 Cache:             6.00 MB
  L3 Cache:             0.00 B
  Bus Frequency:        0.00 Hz
  Memory:               1.70 GB
  Memory Type:          N/A
  SIMD:                 1
  BIOS:                 N/A
  Processor Model:      Intel Xeon E5410
  Processor Cores:      2
```

Windows Micro instance:

```
Platform:               Windows x86 (32-bit)
Compiler:               Visual C++ 2008
Operating System:       Microsoft Windows Server 2008 Datacenter
Model:                  Xen HVM domU
Motherboard:            Unknown Motherboard
Processor:              Intel(R) Xeon(R) CPU         E5430  @ 2.66GHz
Processor ID:           GenuineIntel Family 6 Model 23 Stepping 10
Logical Processors:     1
Physical Processors:    1
Processor Frequency:    2.97 GHz
L1 Instruction Cache:   0.00 B
L1 Data Cache:          0.00 B
L2 Cache:               6.00 MB
L3 Cache:               0.00 B
Bus Frequency:          0.00 Hz
Memory:                 614 MB
Memory Type:            0 MHz
SIMD:                   1
BIOS:                   Xen 3.1.2-128.1.10.el5
Processor Model:        Intel Xeon E5430
Processor Cores:        1
```

Windows Small instance:

Platform:	Windows x86 (32-bit)
Compiler:	Visual C++ 2008
Operating System:	Microsoft Windows Server 2008 Datacenter
Model:	Xen HVM domU
Motherboard:	Unknown Motherboard
Processor:	Dual-Core AMD Opteron(tm) Processor 2218 HE
Processor ID:	AuthenticAMD Family 15 Model 65 Stepping 3
Logical Processors:	1
Physical Processors:	1
Processor Frequency:	2.89 GHz
L1 Instruction Cache:	64.0 KB
L1 Data Cache:	64.0 KB
L2 Cache:	1.00 MB
L3 Cache:	0.00 B
Bus Frequency:	0.00 Hz
Memory:	1.66 GB
Memory Type:	0 MHz
SIMD:	1
BIOS:	Xen 3.1.2-92.1.13.el5.
Processor Model:	AMD Opteron 2218 HE
Processor Cores:	1

Windows Medium instance:

Platform:	Windows x86 (32-bit)
Compiler:	Visual C++ 2008
Operating System:	Microsoft Windows Server 2008 Datacenter
Model:	Xen HVM domU
Motherboard:	Unknown Motherboard
Processor:	Intel(R) Xeon(R) CPU E5410 @ 2.33GHz
Processor ID:	GenuineIntel Family 6 Model 23 Stepping 10
Logical Processors:	2
Physical Processors:	2
Processor Frequency:	2.32 GHz
L1 Instruction Cache:	0.00 B
L1 Data Cache:	0.00 B
L2 Cache:	6.00 MB
L3 Cache:	0.00 B
Bus Frequency:	0.00 Hz
Memory:	1.70 GB
Memory Type:	0 MHz
SIMD:	1
BIOS:	Xen 3.1.2-92.1.13.el5.
Processor Model:	Intel Xeon E5410
Processor Cores:	2

Appendix 6 – Full geekbench results for the Amazon cloud

	Ubuntu - Micro Instance	Ubuntu - Small Instance	Ubuntu - Medium Instance
Integer Performances			
Integer			
Blowfish (in MB/sec)			
single-threaded scalar	83.4	34.80	73.5
multi-threaded scalar	83.3	34.10	134
Text Compress (in MB/sec)			
single-threaded scalar	6.64	2.63	5.91
multi-threaded scalar	6.46	2.71	11
Text Decompress (in MB/sec)			
single-threaded scalar	7.84	3.15	6.98
multi-threaded scalar	7.27	3.06	12.5
Image Compress (in Mpixels/sec)			
single-threaded scalar	15.1	5.96	13.3
multi-threaded scalar	15.1	6.66	20.4
Image Decompress (in Mpixels/sec)			
single-threaded scalar	26.5	13.00	23
multi-threaded scalar	26.7	13.10	38.4
Lua (in Mnodes/sec)			
single-threaded scalar	1.3	0.53	1.14
multi-threaded scalar	1.32	0.50	1.49
Floating Point Performance			
Floating Point			
Mandelbrot (in Gflops)			
single-threaded scalar	1.32	0.57	1.16
multi-threaded scalar	1.32	0.55	2.07
Dot Product (in Gflops)			
single-threaded scalar	1.76	0.73	1.54
multi-threaded scalar	1.76	0.70	2.73
single-threaded vector	3.23	1.87	2.84
multi-threaded vector	3.21	1.35	3.68
LU Decomposition (in Gflops)			
single-threaded scalar	2.05	2.07	1.8
multi-threaded scalar	2.08	2.08	1.82
Primality Test (in Mflops)			
single-threaded scalar	463.2	178.10	411.4
multi-threaded scalar	462.1	178.50	751
Sharpen Image (in Mpixels/sec)			

single-threaded scalar	15	6.81	13.1
multi-threaded scalar	14.9	7.26	16.4
Blur Image (in Mpixels/sec)			
single-threaded scalar	3.71	1.47	3.26
multi-threaded scalar	3.7	1.49	5.73
Memory Performance			
Memory			
Read Sequential (in GB/sec)			
single-threaded scalar	3.11	1.53	2.92
Write Sequential (in GB/sec)			
single-threaded scalar	2.42	1.01	2.31
Stdlib Allocate (in Mallocs/sec)			
single-threaded scalar	8.59	3.64	7.4
Stdlib Write (in GB/sec)			
single-threaded scalar	3.75	2.32	3.29
Stdlib Copy (in GB/sec)			
single-threaded scalar	3.76	1.66	3.18
Stream Performance			
Stream			
Stream Copy (in GB/sec)			
single-threaded scalar	2.7	2.60	2.44
single-threaded vector	2.85	2.74	2.66
Stream Scale (in GB/sec)			
single-threaded scalar	2.7	2.55	2.64
single-threaded vector	2.84	2.70	2.78
Stream Add (in GB/sec)			
single-threaded scalar	2.8	2.68	2.71
single-threaded vector	2.97	2.87	2.85
Stream Triad (in GB/sec)			
single-threaded scalar	2.76	2.68	2.7
single-threaded vector	2.99	2.85	2.92
Overall Geekbench Score	2737	1361	2926
Integer	2132	880	2467
Floating Point	3110	1733	3930
Memory	2622	1280	2481
Stream	2000	1909	1910

	Windows - Micro Instance	Windows - Small Instance	Windows - Medium Instance
Integer Performances			
Integer			
Blowfish (in MB/sec)			
single-threaded scalar	66.7	62.9	61.1
multi-threaded scalar	66.3	38.7	121.4
Text Compress (in MB/sec)			
single-threaded scalar	7.15	3.02	6.15
multi-threaded scalar	7.18	2.69	12.5
Text Decompress (in MB/sec)			
single-threaded scalar	9.69	4.36	8.78
multi-threaded scalar	9.66	3.88	17.4
Image Compress (in Mpixels/sec)			
single-threaded scalar	17.6	14.9	16.4
multi-threaded scalar	18	14.2	32.5
Image Decompress (in Mpixels/sec)			
single-threaded scalar	26.1	18	22.9
multi-threaded scalar	25.7	15.5	38.8
Lua (in Mnodes/sec)			
single-threaded scalar	1.39	0.4	1.23
multi-threaded scalar	1.39	0.4	2.45
Floating Point Performance			
Floating Point			
Mandelbrot (in Gflops)			
single-threaded scalar	1.17	0.65	1.06
multi-threaded scalar	1.17	0.5	2.13
Dot Product (in Gflops)			
single-threaded scalar	0.56	0.18	0.51
multi-threaded scalar	0.56	0.18	1.02
single-threaded vector	3.35	2.81	3.03
multi-threaded vector	3.34	1.53	6
LU Decomposition (in Gflops)			
single-threaded scalar	1.81	1.39	1.5
multi-threaded scalar	1.84	1.39	2.7
Primality Test (in Mflops)			
single-threaded scalar	507.8	159.5	463.8
multi-threaded scalar	508	146.4	911.8
Sharpen Image (in Mpixels/sec)			
single-threaded scalar	1.31	2.51	1.19
multi-threaded scalar	1.3	2.19	2.34

Blur Image (in Mpixels/sec)			
single-threaded scalar	1.78	1.74	1.64
multi-threaded scalar	1.79	2.22	3.24
Memory Performance			
<u>Memory</u>			
Read Sequential (in GB/sec)			
single-threaded scalar	2.7	2.26	2.81
Write Sequential (in GB/sec)			
single-threaded scalar	2.17	1.7	2.15
Stdlib Allocate (in Mallocs/sec)			
single-threaded scalar	3.38	1.27	3.04
Stdlib Write (in GB/sec)			
single-threaded scalar	2.23	1.78	2.32
Stdlib Copy (in GB/sec)			
single-threaded scalar	1.45	1.07	1.5
Stream Performance			
<u>Stream</u>			
Stream Copy (in GB/sec)			
single-threaded scalar	2.09	2.03	2.49
single-threaded vector	2.58	2.28	2.67
Stream Scale (in GB/sec)			
single-threaded scalar	2.16	1.88	2.53
single-threaded vector	2.34	2.28	2.41
Stream Add (in GB/sec)			
single-threaded scalar	2.49	1.51	2.43
single-threaded vector	2.7	2.36	2.47
Stream Triad (in GB/sec)			
single-threaded scalar	2.69	1.46	2.39
single-threaded vector	2.63	2.5	2.34
Overall Geekbench Score	2006	1266	2493
Integer	2246	1147	2979
Floating Point	1991	1311	2638
Memory	1752	1315	1763
Stream	1732	1431	1748

Appendix 7 – Networking results of the private cloud

Node	Network performance (in Gbits/s)
1 client in ideal conditions	8.63
1 client and 1 machine flooding the network	0.378
1 client and 2 machine flooding the network	0.211

FrontEnd	Network performance (in Mbits/s)
1 client in ideal conditions	95
1 client and 1 machine flooding the network	63.9
1 client and 2 machine flooding the network	22.9

Appendix 8 – Characteristics of the Frontend and the Node used in the private cloud

For the FrontEnd it has been used:

- A server Dell Poweredge T110
- A processor Intel Pentium Dual-Core E6500 2.93GHz 2MB L2 Cache LGA 775 65W
- 2048Mo of RAM memory

For the Node it has been used:

- A server Dell poweredge T310
- A processor Intel Xeon X3430 Lynnfield 2.4GHz 8MB L3 Cache LGA 1156 95W
- 12Go of RAM memory

Appendix 9 – Gant chart

	June	July	August	September	October	November	December
Research proposal	▨						
Literature research		▨					
Literature review		▨	▨				
Design			▨	▨			
Implementation			▨	▨	▨		
Evaluation						▨	
Report update and completion				▨	▨	▨	▨