



Escola Politècnica Superior
de Castelldefels

UNIVERSITAT POLITÈCNICA DE CATALUNYA

MASTER THESIS

TITLE: Infrastructure as a Service (IaaS): Application case for TrustedX

MASTER DEGREE: Master in Science in Telecommunication Engineering & Management

AUTHOR: Alberto Guirao Villalonga

DIRECTOR: Francisco Jordán Fernández

DATE: January, 25th 2011

Título: Infrastructure as a Service (IaaS): Caso de aplicación para TrustedX

Autor: Alberto Guirao Villalonga

Director: Francisco Jordán Fernández

Fecha: 25 de Enero de 2011

Resumen

Cada año el precio del hardware disminuye haciendo posible a las empresas comprar servidores cada vez más potentes. Sin embargo, los gastos operacionales como el mantenimiento de estos servidores crece cada año, por lo que se hace necesario administrar mejor nuestros recursos hardware o incluso se puede pensar en externalizar este servicio.

Las redes de Infraestructura como un Servicio (IaaS) nos ofrecen la posibilidad de administrar de una manera más óptima los recursos hardware. Gracias a la virtualización de los recursos hardware, estas redes ofrecen únicamente el hardware que el usuario necesita, creando y destruyendo máquinas virtuales en tiempo real para adaptarse a la potencia que el usuario necesita en cada momento. De esta forma, una red IaaS necesitará menos recursos hardware que una red de servidores convencionales para realizar la misma carga de trabajo.

La reducción en el número de servidores aporta una reducción de costes para las empresas, tanto a nivel de adquisición de nuevos servidores, como el mantenimiento de los mismos al reducirse las tareas de administración. Además la reducción del número de servidores nos aporta una disminución del consumo eléctrico (tanto del consumo de los servidores como del consumo en refrigeración). Por tanto, las soluciones de IaaS ofrecen a las empresas la posibilidad de un crecimiento más sostenible, tanto económicamente como desde una vista medioambiental.

En este proyecto se describirán las características de una red de IaaS y se ofrecerá un caso práctico de uso. En la solución que ofreceremos, usaremos Eucalyptus como software de IaaS y crearemos un portal web para administrar los recursos de la red. Los recursos software ofrecidos por nuestra red se basarán en servidores de TrustedX. TrustedX es una plataforma de servicios web que aporta mecanismos de seguridad y confianza en Arquitecturas Orientadas a Servicios (SOA). TrustedX ha sido desarrollado por Safelayer Secure Communications.

Title: Infrastructure as a Service (IaaS): Application case for TrustedX

Author: Alberto Guirao Villalonga

Director: Francisco Jordán Fernández

Date: January, 25th 2011

Overview

The hardware price decreases every year, enabling the companies to buy servers that are becoming more powerful. However, operating expenses like server maintenance grows every year, so we need a new way to manage our hardware resources or even externalize this service.

Infrastructure as a Service (IaaS) networks offer the ability to manage our hardware resources in a more optimal way. Thanks to the virtualization of hardware resources, these networks offer only the hardware that the user really needs, creating and destroying virtual machines in real time to adapt to the user needs at any time. Thus, an IaaS network requires less hardware resources than a conventional server network to perform the same workload.

The reduction on the number of servers provides a cost reduction for businesses, in terms of less acquisition of new servers, and the maintenance of servers by reducing administrative tasks. Besides, reducing the number of servers implies a decrease in power consumption in two aspects: server consumption and cooling consumption. Therefore, an IaaS solution offers to companies the possibility of growing in a more sustainable way, both in economic and environmental terms.

This project describes the characteristics of an IaaS network and explains a practical use case. In this solution, we will use Eucalyptus as IaaS software and we will create a web portal to manage the network resources. The software resources offered by our network will be based on TrustedX servers. TrustedX is a web services platform that provides security and trust mechanisms in a service-oriented architecture (SOA). TrustedX has been developed by Safelayer Secure Communications.

Dedicatoria

En primer lugar querría dedicar este trabajo a mis padres, Amparo y José, y a mi hermana Marta, por todo el apoyo durante estos años ya que sin ellos sería imposible que hoy me encontrara aquí.

A Francisco Jordán y Helena Pujol por asesorarme y guiarme durante la realización de este proyecto y por darme la posibilidad de poder realizarlo en *Safelayer*.

A todos los compañeros de *Safelayer*, en especial a Ana y Jorge y a la gente de *TrustedX*, por resolverme las dudas que me han surgido realizando este proyecto.

A mis compañeros de trabajo y becarios, Isaac, Jonatan, Irene, Lucas, Alex y Sergio por hacer más amenos los días de trabajo.

A todos vosotros, ¡muchas gracias!

INDEX

| | |
|--|-----------|
| CHAPTER 1. INTRODUCTION | 1 |
| CHAPTER 2. TECHNOLOGIES | 6 |
| 2.1. Virtualization | 6 |
| 2.1.1. VMWare..... | 7 |
| 2.1.2. Xen..... | 8 |
| 2.1.3. KVM..... | 9 |
| 2.2. Network monitoring | 9 |
| 2.2.1. Nagios..... | 10 |
| 2.2.2. Ganglia..... | 10 |
| 2.3. Web services | 10 |
| 2.3.1. Tomcat..... | 11 |
| 2.3.2. Spring MVC..... | 11 |
| 2.4. TrustedX: security service platform | 12 |
| CHAPTER 3. SOFTWARE AND PROVIDERS OF IAAS | 13 |
| 3.1. Public networks | 13 |
| 3.1.1. Amazon..... | 13 |
| 3.2. Private networks | 14 |
| 3.2.1. Eucalyptus..... | 15 |
| 3.2.2. OpenNebula..... | 20 |
| 3.3. Comparison between IaaS solutions | 21 |
| CHAPTER 4. IAAS SERVICE APPROACH APPLIED TO SECURITY SERVICES | 23 |
| 4.1. Service description: use case | 23 |
| 4.2. Eucalyptus network | 25 |
| 4.2.1. Nodes distribution..... | 25 |
| 4.2.2. Instances communication..... | 27 |
| 4.2.3. Network configuration..... | 28 |
| 4.3. TrustedX over Eucalyptus | 30 |
| 4.3.1. TrustedX over Xen..... | 30 |
| 4.3.2. Uploading an image on Eucalyptus..... | 32 |
| 4.4. Eucalyptus monitoring with Nagios | 33 |
| 4.5. Web portal | 35 |
| 4.5.1. User configuration..... | 35 |
| 4.5.2. Instance management..... | 37 |
| 4.5.3. TrustedX with high availability..... | 40 |

| | |
|--|-----------|
| CHAPTER 5. ENVIRONMENTAL STUDY | 42 |
| 5.1. Worldwide energy consumption | 42 |
| 5.2. Green IT | 43 |
| 5.2.1. Virtualization as a green IT technology | 43 |
| 5.3. Environmental conclusions for an IaaS network | 44 |
| CHAPTER 6. PLANNING OF THIS PROJECT | 45 |
| 6.1. Project scheduler..... | 45 |
| 6.2. Economic analysis | 47 |
| CHAPTER 7. CONCLUSIONS AND FUTURE LINES..... | 49 |
| 7.1. Conclusions | 49 |
| 7.2. Future lines | 50 |
| BIBLIOGRAPHY | 51 |
| ANNEXES | 54 |
| A. Acronyms | 54 |
| B. Hybridfox | 56 |
| C. CloudBerry S3 Explorer | 58 |

INDEX OF FIGURES

| | |
|--|----|
| Fig. 1.1 Cloud computing evolution | 1 |
| Fig. 1.2 Gartner's 2010 Hype Cycle | 3 |
| Fig. 2.1 Virtualization scheme..... | 7 |
| Fig. 2.2 VMWare virtualization..... | 7 |
| Fig. 2.3 Xen virtualization..... | 8 |
| Fig. 2.4 KVM virtualization | 9 |
| Fig. 2.5 MVC pattern | 11 |
| Fig. 2.6 TrustedX Platform | 12 |
| Fig. 3.1 Amazon EC2 flexibility | 14 |
| Fig. 3.2 Eucalyptus cloud..... | 15 |
| Fig. 3.3 Credentials administration in Eucalyptus | 19 |
| Fig. 3.4 OpenNebula cloud | 20 |
| Fig. 4.1 Interaction with the hybrid cloud..... | 24 |
| Fig. 4.2 Eucalyptus network configuration..... | 25 |
| Fig. 4.3 Eucalyptus physical network implementation | 26 |
| Fig. 4.4 NAT example with a virtual machine on Eucalyptus managed mode.. | 28 |
| Fig. 4.5 Virtual machines per cluster | 29 |
| Fig. 4.6 Example of parted command | 31 |
| Fig. 4.7 Example of uploading a kernel | 32 |
| Fig. 4.8 Example of uploading a image | 33 |
| Fig. 4.9 Nagios network implementation | 33 |
| Fig. 4.10 NRPE communication..... | 34 |
| Fig. 4.11 Network monitoring log..... | 35 |
| Fig. 4.12 Web user configuration..... | 36 |
| Fig. 4.13 Information about running instances..... | 37 |
| Fig. 4.14 Configuration file selection for a new instance | 38 |
| Fig. 4.15 Front-end flowchart..... | 39 |
| Fig. 4.16 High availability | 40 |
| Fig. 5.1 Worldwide spending in servers, in \$billion | 42 |
| Fig. 5.2 Green IT techniques | 44 |
| Fig. 6.1 Gantt chart of the project | 46 |
| Fig. B.1 Hybridfox Configuration..... | 56 |
| Fig. B.2 Image list with Hybridfox | 56 |
| Fig. B.3 Instance list with Hybridfox..... | 57 |
| Fig. B.4 Public IP list with Hybridfox | 57 |
| Fig. B.5 Public IP change with Hybridfox | 58 |
| Fig. B.6 Security groups with Hybridfox | 58 |
| Fig. C.1 CloudBerry configuration | 59 |
| Fig. C.2 Bucket list with CloudBerry | 59 |
| Fig. C.3 Bucket management with CloudBerry | 60 |
| Fig. C.4 File with public permissions | 60 |

CHAPTER 1. INTRODUCTION

The definition of "Cloud Computing" does not refer to a new technology. Instead, it refers to a union of several well-known technologies that provide a global service and scalable depending on the user demand.

Figure 1.1 shows different technologies that have allowed the concept of cloud computing.

In the earlier days, two different technologies allowed to solve the problem of high computer resources demand: cluster computing and super computing. The first solution involved a computer network with different servers, and each user was redirected to one server. By contrast, the second solution referred to one powerful computer with specialized resources that was able to work for all users.

The term Grid Computing was used to describe a heterogeneous computer network where all devices were working together for a specific result. An example of this type of network is the project SETI@home [1], a collaborative software that uses wasted CPU cycles from personal computers, to analyze radio transmissions received from the space in search of intelligent life.



Fig. 1.1 Cloud computing evolution

Although it is difficult to come up with a precise and comprehensive definition of Cloud Computing, at the heart of it is the idea that applications run somewhere on the "cloud". We do not know or care where exactly the application is.

This is not a big issue in a not critical scenario. As end users, we use web applications daily, without any concern about where the application actually runs. But on an enterprise scenario, where the data is confidential and must be carefully protected, this issue is critical.

The US NIST (National Institute of Standards and Technology) defines cloud computing as follows: *"Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics (On-demand self-service, broad network access, resource pooling, rapid elasticity and measured service), three service models (SaaS, PaaS and IaaS), and four deployment models (Private cloud, community cloud, public cloud and hybrid cloud)."* [2]

In Cloud Computing we can find mainly three different service models: Software as a Service (SaaS), Platform as a Service (PaaS) or Infrastructure as a Service (IaaS).

- **SaaS:** Delivery of an application as a service over the Internet, eliminating the need to install and run the application on the customer computer and simplifying the maintenance and support. An example of SaaS is Gmail [3], an e-mail application based on web browser developed by Google.
- **PaaS:** Delivery of a platform for deployment of applications without the cost and complexity of buying and managing the underlying hardware and software layers. An example of PaaS is Google App Engine [4], a platform for developing and hosting web applications in Google-managed data centers.
- **IaaS:** Delivery of a computer infrastructure, typically a platform virtualization environment as a service. Rather than purchasing servers, software, data center space or network equipment, clients rent those resources as a fully outsourced service. An example of IaaS is Amazon Elastic Compute Cloud (Amazon EC2) [5] that allows users to rent virtual computers on which they can run their own computer applications.

Also, Cloud Computing can be split into four different models depending on the deployment model of network:

- **Private cloud:** The cloud infrastructure is operated solely for an organization. It may be managed by the organization or by a trusted partner.
In this type of clouds we cannot have infinite resources from a user point of view. To achieve this objective, we should have large free hardware resources that allowed absorbing work peak loads. On this hypothetical scenario we would have a high number of servers with a low utilization so we would not have any benefits in contrast to a scenario without cloud computing.
- **Community cloud:** The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance

considerations). It may be managed by the organizations or by a trusted partner.

- **Public cloud:** The cloud is managed by a third-party provider, whereby resources are dynamically provisioned on a fine-grained, self-service basis over the Internet, via web applications/web services. On this scenario we have to trust on this third-party provider that will manage our data to be safe and secure.

Moreover, a public cloud have elasticity. This means, that this type of network have infinite resources from a user point of view.

- **Hybrid cloud:** A hybrid cloud is a combination of two or more cloud models (private, community or public). On this scenario we can split risks, we can put the critical data on a private or community cloud and the not critical data on a public cloud. And we can have elasticity, if the private cloud is out of resources we can use the public cloud.

Although we have cloud computing commercial solutions available at this moment, this technology is still under development. Gartner, a company specialized in information technology research and advisory, makes every year a graphic (**Fig. 1.2**) with the top new technologies of the moment. This graphic is called “hype cycle” and represents the over-enthusiasm or “hype” and subsequent disappointment that typically happens with the introduction of new technologies [6].

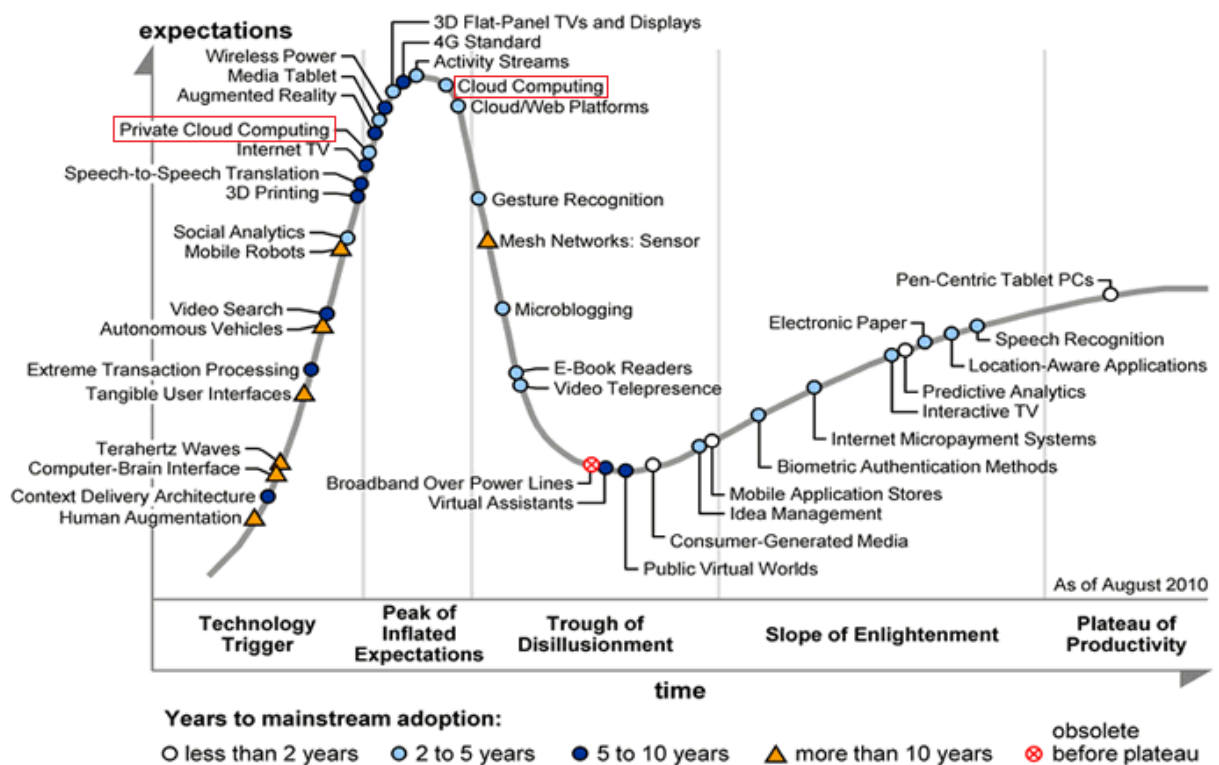


Fig. 1.2 Gartner's 2010 Hype Cycle

A hype cycle in Gartner's interpretation comprises five phases:

- **Technology Trigger:** Product launch or other event that generates significant press and interest.
- **Peak of Inflated Expectations:** In this phase, a frenzy of publicity typically generates over-enthusiasm and unrealistic expectations. There may be some successful applications of a technology, but there are typically more failures.
- **Trough of Disillusionment:** Technologies enter the "trough of disillusionment" because they fail to meet expectations and quickly become unfashionable. Consequently, the press usually abandons the topic and the technology.
- **Slope of Enlightenment:** Although the press may have stopped covering the technology, some businesses continue through the "slope of enlightenment" and experiment to understand the benefits and practical application of the technology.
- **Plateau of Productivity:** A technology reaches the "plateau of productivity" as the benefits of it become widely demonstrated and accepted. The technology becomes increasingly stable and evolves in second and third generations. The final height of the plateau varies according to whether the technology is broadly applicable or benefits only a niche market.

So according to Gartner's point of view, cloud computing is in the middle of the "peak of inflated expectations" phase and the technology needs from 2 to 5 years to be considered a mature technology. Also we can see that private cloud computing solutions are a step behind the rest of cloud computing solutions.

This actual state of the technology has two main problems:

- The actual market solutions may not be the final solutions because they are evolving quickly.
- Most companies cannot trust on cloud computing due to security issues.

For these reasons it is important to do research projects about cloud computing, to evaluate and understand the current state of the technology and the actual benefits that we can have if we use these solutions.

In this project we meet the expectation of a private company (Safelayer Secure Communications [7]) to offer their products on a SaaS model. To reach this objective we choose the TrustedX [8] platform, one of the Safelayer products.

TrustedX is a web services platform that provides security and trust mechanisms (authentication, authorization, electronic signatures and data protection) in Service-Oriented Architectures (SOA).

Nowadays, TrustedX is distributed with a physical or virtual appliance, on both solutions the final user has to manage the server. With the new characteristics of SaaS solutions we want to offer TrustedX as a service, without the need of manage the server or configure the service.

In this project we can see a particular cloud computing environment based on a hybrid cloud. With a hybrid cloud we can avoid most of the security issues, because we can choose the private cloud for critical services and a public cloud for not critical services. Moreover we avoid the elasticity problems of a private cloud, because if this network is out of resources, we can use the resources of the public cloud.

As a cloud computing network we choose an Infrastructure as a Service (IaaS) solution. This is because the other two solutions (SaaS and PaaS) are closed and packaged solutions. So, to offer TrustedX on a SaaS model, we will have an IaaS network, an on top of this network, we will have the SaaS solution.

On this solution an administrator user can manage the hybrid cloud through a web portal. This user can create, configure and destroy virtual machines in real time to match the work load that is needed in each moment. These virtual machines will run the services that will use the final users.

As a basic service, an administrator can start, stop and configure different TrustedX, and as an advanced service, the administrator can start a TrustedX with high availability, that action will start four different virtual machines to run the service (a load balancer, a database and two TrustedX). All these administrative tools will be offered through a web portal.

This document is divided into different chapters. In chapter two, we describe different technologies involved in an Infrastructure as a Service implementation. In chapter three, we list different market solutions in an Infrastructure as a Service implementation in the two variants: public and private networks. In chapter four, we show the particularities about our use case and we describe our solution. In chapter five, we explain the environmental benefits of a cloud computing solution. Finally in chapter six we explain the planning of this project, and in chapter seven, we explain our final conclusions about this project.

CHAPTER 2. TECHNOLOGIES

2.1. Virtualization

Virtualization is the main technology in an Infrastructure as a Service solution. In all these solutions, we always end up having a virtual machine.

Virtual machines, from a conceptual point of view, are not different from a physical computer. So inside a virtual machine we can deploy any type of operating system, and within an operating system, we can deploy any type of service.

The main difference between a virtual machine and a physical computer is that a physical computer can host a certain number of virtual machines. This number depends on the resources of the physical computer, and the resources that we want to assign to each virtual machine, but it is estimated that the current average ratio are in 6 virtual machines per physical computer [9].

The key benefits of virtualization are:

- **Cost reduction:** Nowadays, the hardware price is low. For this reason the price of huge servers is relative low, but without virtualization, the effective utilization is also very low. Through virtualization we can split these huge servers into small servers dedicated to one application each one. The cost reduction in relation to a scenario with many servers involve less hardware expenses, less energy consumption (operating and cooling consumption), less space in data centers and less time expense in servers maintenance.
- **Risk reduction:** In a scenario with a huge server that runs a lot of applications and without virtualization, each application can have a failure risk and can affect to the whole server. With virtualization, each application can run in a dedicated virtual server, and for this reason, is isolated from other virtual machines. Also, virtual machines can be useful in any disaster recovery plan, because we can have virtual machines with the key applications for our company ready to start in case of a failure in our main services.
- **Flexibility:** Virtualization removes the dependency of the operating system on a particular piece of hardware. It allows to us to grow, shrink or move the virtual machines without having to modify the hardware.

But also, virtualization has some disadvantages:

- **Server failure:** A general failure in one server means that all the virtual machines that are hosted in that server will fail too.

- **Degraded performance:** Virtualization requires extra hardware resources in terms of overhead. The problem is that it is almost impossible to estimate in advance how many extra resources will be needed.

As we can see in the next figure (**Fig. 2.1**), the general scheme of a virtualization is to add an extra layer between the hardware and the Operating System. This layer is called hypervisor, because it will manage the hardware resource between the host (the physical computer) and the different guests (the virtual machines).

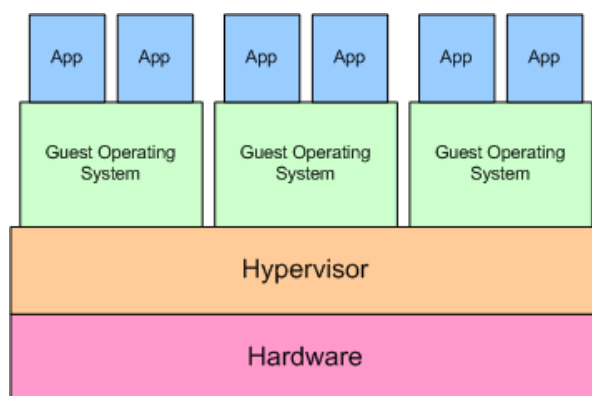


Fig. 2.1 Virtualization scheme

2.1.1. VMWare

VMWare [10] is an enterprise that offers solutions to do full virtualizations based on software. In this type of virtualization, a sufficient quantity of hardware is simulated so guest operating system can run without modifications (**Fig. 2.2**). With this technique, the guest operating system does not know that is running inside a virtual machine.

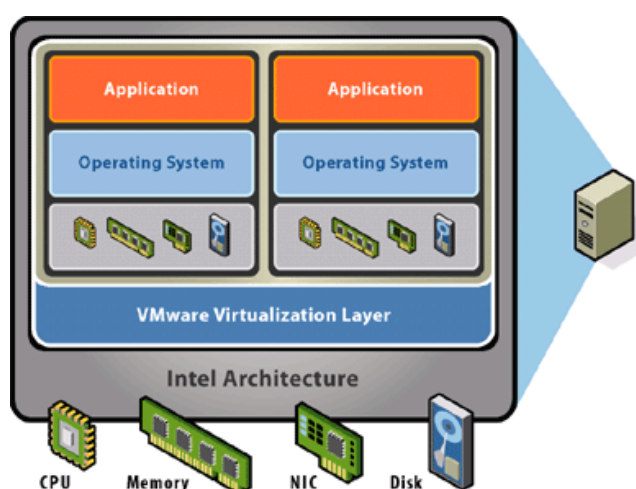


Fig. 2.2 VMWare virtualization

As the guest operating systems do not know that they are running in a virtual machine, two different operating systems may want to access the physical resources at the same time. The hypervisor controls the access to the physical resources between these competing operating system instances.

2.1.2. Xen

Xen [11] systems have a structure with the Xen hypervisor as the lowest and most privileged layer (**Fig. 2.3**). Above this layer we can execute one or more guest operating systems, which the hypervisor schedules across the physical CPUs. This type of virtualization is called paravirtualization. The first guest operating system, called in Xen terminology "Domain 0" (Dom0), boots automatically when the hypervisor boots and receives special management privileges and direct access to all physical hardware by default. The system administrator can log into Dom0 in order to manage any further guest operating systems, called "Domain U" (DomU) in Xen terminology.

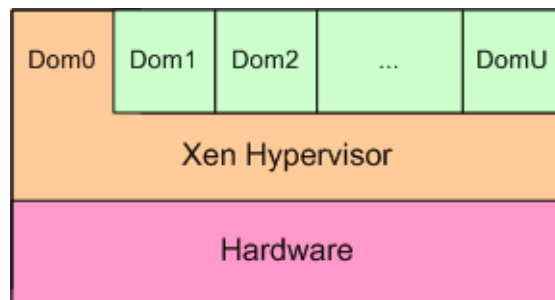


Fig. 2.3 Xen virtualization

Paravirtualization is different in that the hypervisor operates in a more cooperative way, because each guest Operating System (OS) is aware that it is running in a virtualized environment, so each one cooperates with the hypervisor to share the underlying hardware. To get that each OS cooperates, it is necessary to modify each OS kernel to add special instructions.

The main advantage of paravirtualization approach is that it allows the fastest possible software-based virtualization, at the cost of not supporting proprietary operating systems, because it is not possible to modify a closed OS software.

2.1.3. KVM

Kernel-based Virtual Machine (KVM) [12] is a Linux kernel virtualization infrastructure. KVM replaces the hypervisor layer and turns the Linux kernel itself to work as a hypervisor (**Fig. 2.4**). This type of virtualization is not based on software virtualization like Xen or VMware, it is based on hardware virtualization.

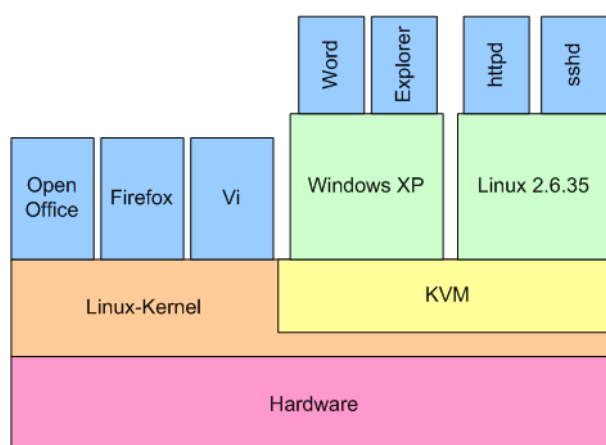


Fig. 2.4 KVM virtualization

Under this model, every guest Operating System is a regular Linux process, scheduled by the standard Linux scheduler. Traditionally, a normal Linux process has two execution modes: kernel and user. The user mode is the default mode for applications, and an application goes into kernel mode when it requires some service from the kernel, such as writing to the hard disk. KVM adds a third mode, the guest mode. Guest mode processes run inside of the virtual machine. The guest mode, just like the normal mode (non-virtualized instance), has its own kernel and user-space variations.

The main problem with virtualizations based on hardware, is that it needs a specific hardware support in the CPU to allow this type of instructions. Only modern CPUs (since 2006) can have this type of support.

2.2. Network monitoring

In an IaaS scenario (a network with multiple servers) it's essential to make a central monitoring to know the state of the network. If we do not do this motorization, it can be hard to know if any server is not working if we do not stay in this machine.

If our solution of IaaS does not include a network monitoring state, can be a good idea to use one of the commercial solutions for network monitoring.

2.2.1. Nagios

Nagios [13] is a network monitoring system, which monitors the equipment (hardware) and services (software) that we are interested in monitoring, alerting us when their behavior is not desired. Among its main features include monitoring of service networks (like SMTP, HTTP, FTP...) and the system hardware resources (CPU load, disk usage, RAM usage...).

Nagios is independent of the Operating System that it is monitoring and does the possibility of remote monitoring using encrypted SSL or SSH tunnels, and the ability to schedule specific plugins for new applications.

2.2.2. Ganglia

Ganglia [14] is a scalable distributed system monitor tool for high-performance computing systems such as clusters and grids. It is based on a hierarchical design targeted at federations of clusters. It relies on a multicast-based listen/announce protocol to monitor state within clusters and uses a tree of point-to-point connections amongst representative cluster nodes to federate clusters and aggregate their state.

2.3. Web services

One of the main objectives of a cloud computing solution is that a user can use the cloud without the installation of any type of software. In this scenario we have to choose to interact with the user through a web portal, because we may expect that the final user have at least a web explorer in their operating system.

For this purpose, a cloud computing environment has to implement a SOA (Service-Oriented Architecture) interface to communicate between the application web and the cloud. This interface makes a standard communication, on each petition is replied by the cloud asynchronously.

XML is commonly used for interfacing with SOA services, though this is not required. Although the ideal communication between the end user and the cloud is through an application web, the abstraction provided by the SOA/XML interface makes possible to implement an external program with interact to the cloud through the SOA interface.

2.3.1. Tomcat

Apache Tomcat [15] (or Jakarta Tomcat or simply Tomcat) is an open source servlet container developed by the Apache Software Foundation (ASF). Tomcat implements the Java Servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, and provides a "pure Java" HTTP web server environment for Java code to run.

2.3.2. Spring MVC

The Spring Framework [16] is an open source application framework for the Java platform. The Spring Framework comprises several modules that provide a range of services. One of these modules is the MVC module.

Model–View–Controller (MVC) is a software architecture, currently considered an architectural pattern, used in software engineering (**Fig. 2.5**). The pattern isolates "domain logic" (the application logic for the user) from input and presentation (the user interface), permitting independent development, testing and maintenance of each one.

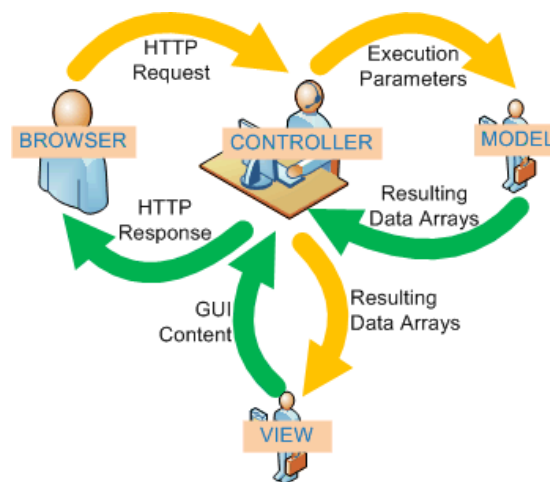


Fig. 2.5 MVC pattern

The main advantage of this pattern is that a graphical designer can work with the view part, and a software engineer can work with the model and controller part without any collision.

The graphical designer will not see any line of code that he does not understand, and the software engineer will be able to change all the logic of the application without destroying the view. This solution is useful also to implement different views to work with the same code, for example, to implement a web with different languages.

2.4. TrustedX: security service platform

TrustedX is a product developed by Safelayer Secure Communications. TrustedX is a Web services platform which provides mechanisms for handling authentication, electronic signature and data encryption. TrustedX is based entirely on policies, and once configured, it manages corporate trust automatically (**Fig. 2.6**).

TrustedX is used by corporate applications to provide security and optimize business process, automating transactions by means of electronic mechanisms and without the need for documents in paper form. Examples of applications are e-Procurement, e-Contracting, e-Invoicing, e-Banking, e-Government, e-Learning, etc.

The platform resolves the complexity of endowing applications with security mechanisms using classic integration tools. The platform's Services-Oriented Architecture (SOA) and complete information management system simplify the integration of trusted mechanisms in business processes, making them independent of one another and offering the capacity of centrally managing security policies and auditing.

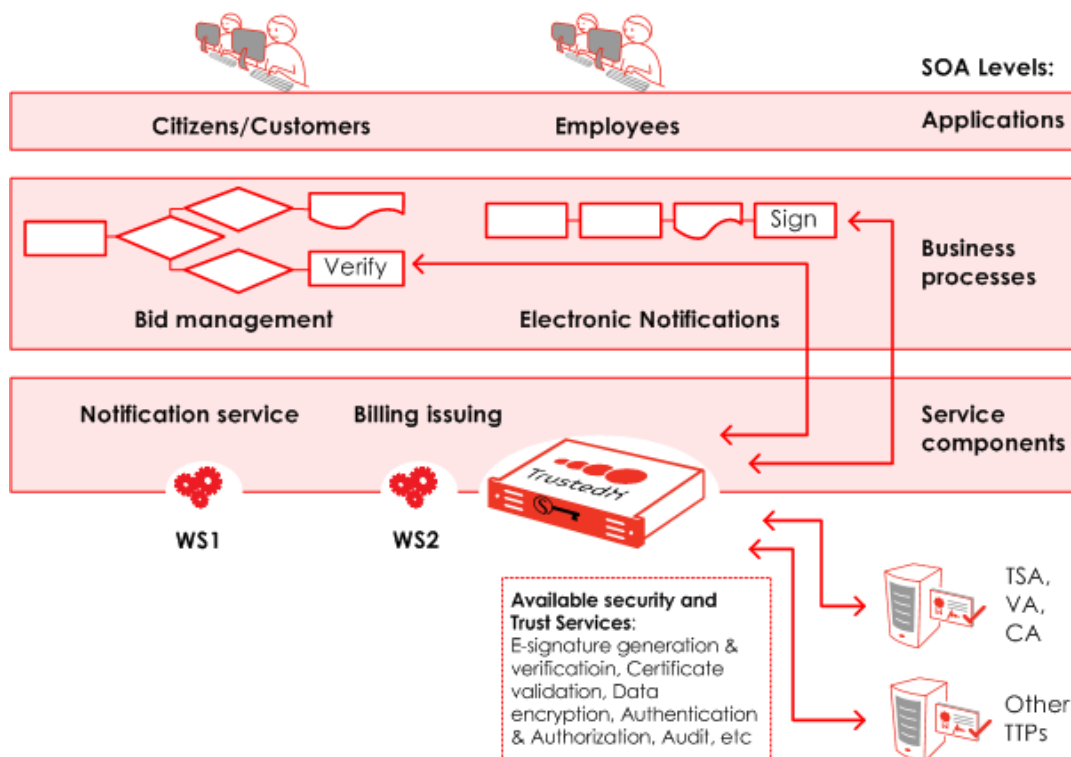


Fig. 2.6 TrustedX Platform

CHAPTER 3. SOFTWARE AND PROVIDERS OF IaaS

3.1. Public networks

In a public IaaS network, a provider offers its infrastructure network to other users. Depending on the business model of the provider, this access can be with a payment subscription or can be free.

A large number of companies offer their own solutions of Infrastructure as a Service, but nowadays, the company that has more successfully in this market is Amazon.

Other companies that also offer their own solutions of IaaS networks are for example: GoGrid, AT&T Synaptic Compute as a Service, Rackspace Cloud, Savvis or Terremark.

3.1.1. Amazon

Amazon.com [17] is an American-based multinational electronic commerce company. The company began as an online bookstore, but nowadays, the company has a broad product diversification. In 2006, Amazon introduced Amazon Elastic Compute Cloud (Amazon EC2).

EC2 allows users to rent virtual computers to run their own computer applications. EC2 offers different configuration types (number of CPUs, Hard Disk capacity, RAM capacity) of virtual machines to match the different needs of a user.

These virtual machines are rented by hour usage, and can be created or deleted in real time by the user, giving to the user the flexibility to pay only for resources that he really needs at any moment, and also, the flexibility to absorb load peaks (**Fig. 3.1**).

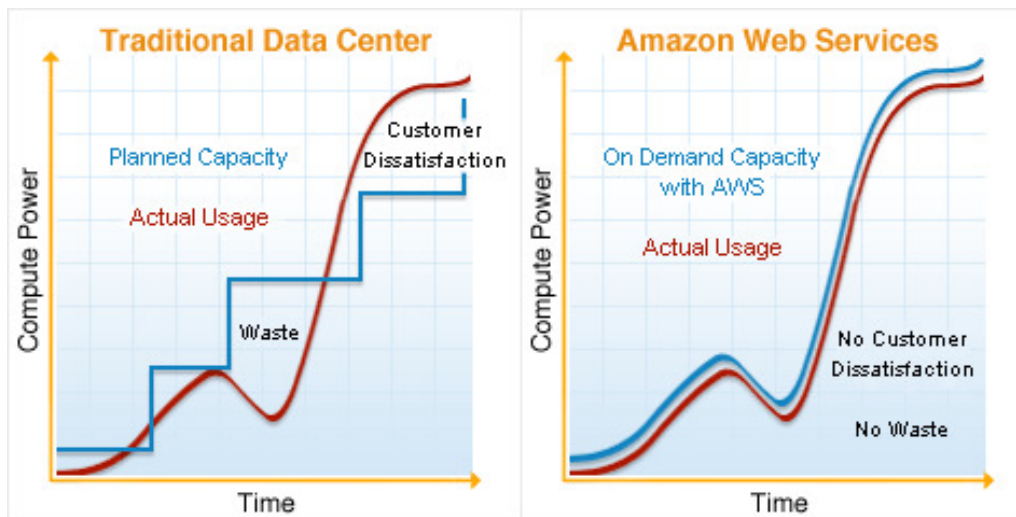


Fig. 3.1 Amazon EC2 flexibility

In addition to EC2, Amazon also offers the Amazon Simple Storage Service [18] (Amazon S3). S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. This service is similar to a FTP Server, in which each user has a number of buckets (similar to a folder in an OS) to store and manage his data.

Amazon also implements a SOA/XML and REST interface to interact with their network (EC2 and S3) [19]. Also, Amazon offers implementations of this interface for popular programming languages allowing to the programmer to not work with XML code, because these libraries translates the programming code calls into XML code.

These libraries developed by Amazon are for these programming languages:

- Java
- Windows and .Net
- PHP
- Ruby

3.2. Private networks

Some companies also offer software pieces that allow users to deploy private cloud computing networks. These networks normally are Infrastructure as a Service, because the other two solutions are more specific.

As public networks, this type of software can be free or licensed by a subscription fee.

3.2.1. Eucalyptus

Eucalyptus [20] is an open source solution for IaaS network that originates from a research project at University of California in the Fall of 2007. Eucalyptus provides an EC2 compatible cloud computing platform and S3 compatible cloud storage platform. Since Eucalyptus makes its services available through EC2/S3 compatible APIs, the client tools written for Amazon can be used with Eucalyptus as well.

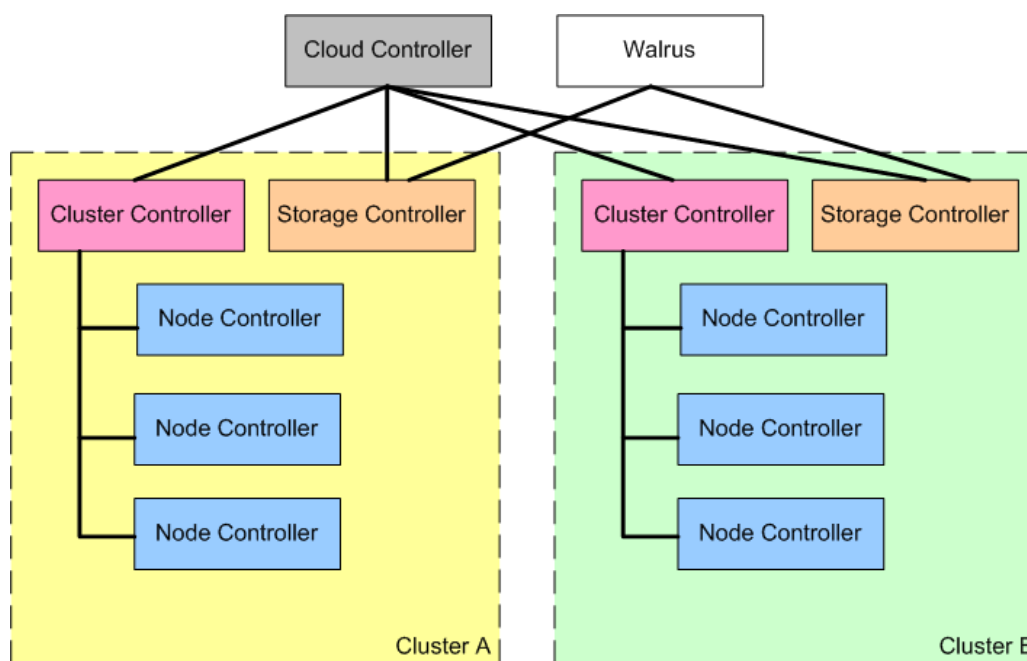


Fig. 3.2 Eucalyptus cloud

3.2.1.1. Network nodes

In an Eucalyptus cloud we have different logical nodes (**Fig. 3.2**):

Cloud controller: The Cloud Controller (CLC) is the front end to the entire cloud infrastructure. CLC provides an EC2/S3 compliant web services interface to the client tools on one side and interacts with the rest of components of the Eucalyptus infrastructure on the other side. CLC also provides a web interface to the users for managing their credentials in the network.

The main functions of CLC are:

- Monitor the availability of resources on various components of the cloud infrastructure, including hypervisor nodes that are used to actually provision the instances and the cluster controllers that manage the hypervisor nodes.

- Resource arbitration: Deciding which clusters will be used for provisioning the instances.
- Monitoring the running instances.

In short, CLC has a comprehensive knowledge of the availability and usage of resources in the cloud and the state of the cloud.

Walrus Storage Controller: The Walrus Storage Controller (WS3) provides a persistent simple storage service using REST and SOAP APIs compatible with S3 APIs provided from Amazon. WS3 should be considered as a simple file storage system, similar to a FTP server.

The main functions of Walrus are:

- Storing the machine images for run instances into Eucalyptus.
- Storing snapshots.
- Storing and serving files using the S3 API.

Cluster Controller: A Cluster Controller (CC) manages one or more Node Controllers and deploys/manages instances on them. CC also manages the networking for the instances running on the nodes under certain types of networking modes of Eucalyptus (**Table 3.1**). CC communicates with CLC on one side and NCs on the other side.

The main functions of Cluster Controller are:

- To receive requests from CLC to deploy instances.
- To decide which NCs to use for deploying the instances on.
- To control the virtual network available to the instances.
- To collect information about the NCs registered with it and report it to the CLC.

Storage Controller: Storage Controller (SC) provides persistent block storage for use by the instances. This storage is like a hard disk storage that can be accessible by one or more instances.

The advantage of this type of storage is that is persistent even though the instance associated to this volume is deleted. This service is similar to the Elastic Block Storage (EBS) service provides from Amazon.

The main functions of Storage Controller are:

- Creation of persistent EBS (Elastic Block Storage) devices.
- Providing the block storage over the network to the instances.
- allowing creation of snapshots of volumes.

Node Controller: In each Node Controller (NC) runs a hypervisor (Xen or KVM), and controls the life cycle of instances running on the node. The NC interacts with the OS and the hypervisor running on the node on one side and the CC on the other side. NC queries the Operating System running on the node to discover the node's physical resources (the number of cores, the size of memory, the available disk space) and also to learn about the state of VM instances running on the node and propagates this data up to the CC.

The main functions of a Node Controller are:

- Collection of data related to the resource availability and utilization on the Node and reporting the data to CC.
- Instance life cycle management.

3.2.1.2. *Network configuration*

Depending on the user needs, an Eucalyptus cloud can be configured in four different scenarios:

System: In System mode, CC generates and assigns a random MAC address (Media Access Control address) to the virtual machine instance while requesting NC to bring up the instance. NC attaches the virtual machine instance's virtual NIC (Network Interface Card) to the physical NIC on the node through a bridge. This mode requires that the Nodes are connected to the same network that the final user is connected. Instances obtain an IP address using a DHCP (Dynamic Host Configuration Protocol) server.

Static: Static mode offers the Eucalyptus administrator more control over the instance IP address assignment than System mode does. In this mode, the administrator configures Eucalyptus with a "map" of MAC address/IP Address pairs on CC.

Before requesting NC to raise an instance, CC sets up a static entry within an Eucalyptus controlled DHCP server, takes the next free MAC/IP pair, and passes on to NC, which attaches the virtual NIC of the instance to the physical NIC of the Node through a bridge similar to how it is handled in 'System' mode.

Managed: Managed mode is the most feature rich mode offered by Eucalyptus. In this mode, the Eucalyptus administrator defines a large network (usually

private and unroutable) from which virtual machine instances will draw their IP addresses. As with Static mode, CC will maintain a DHCP server with static mappings for each instance that is raised and allocate the right IPs at the time of requesting an NC to raise the instance.

Managed mode implements "security groups" for ingress filtering and isolation of instances. The user specifies a security group to which the new instance should be associated with, at the time of requesting a new instance. CC allocates a subset of the entire range of IPs to each security group in such a way that all the instances raised to be a part of the same security group use IPs from the same subset. The isolation between security groups is possible because each security group is in a different VLAN (Virtual Local Area Network).

The user can define ingress filtering rules at the security group level. This ingress filtering rules is like a Firewall that will be the same for the whole security group. In addition, the administrator can specify a pool of public IP addresses that users may allocate, either while raising the instances or later at run-time. This functionality is similar to "elastic IPs" of Amazon.

Managed-NOVLAN: This mode is identical to managed mode in terms of features (dynamic IPs and security groups), but does not provide virtual machine network isolation. This mode is useful if we want all functions, but we are not able to run a VLAN environment in our network.

Table 3.1. Eucalyptus Network Type Comparison

| Networking type | DHCP Server running on the network | CC runs its own DHCP server | Instance Isolation | Private IPs | Ingress Filtering |
|-----------------------|------------------------------------|-----------------------------|--------------------|-------------|-------------------|
| System | Required | No | No | No | No |
| Static | No | Yes | No | No | No |
| Managed | No | Yes | Yes | Yes | Yes |
| Managed-NOVLAN | No | Yes | No | Yes | Yes |

3.2.1.3. User administration

For administrative tasks inside an Eucalyptus network, it includes a web portal. Through this portal, it is possible to register, modify or delete eucalyptus accounts. Only a user with an eucalyptus account can use the cloud computing services of the network.

We can have two different types of eucalyptus accounts:

- **Administrator user:** This type of user can manage his account and the rest of accounts. Moreover, when he asks for the status of the network, he can see all the running instances on the network.
- **User without privileges:** This type of user only can manage his account. Moreover, when he asks for the status of the network, he only can see his own running instances.

In both cases, through this portal, users can access to their credentials to access inside the cloud. These credentials consist in two strings used for applications that require access through the EC2 API, and X.509 certificates for command line tools for Linux from the euca2ools [21] package (**Fig. 3.3**).

Credentials Images

User account Information

Login: **test**
 Name: **tester**
 Email: **test@t.er**

Feel free to change the account information (except the login) and the password whenever you want. The cryptographic credentials for the Web services associated with this account, shown below, will not be affected by these changes.

Edit Account Information

Change Password

Credentials ZIP-file

Click the button to download a ZIP file with your Eucalyptus credentials. Use the public/private key pair included therein with tools that require X.509 certificates, such as Amazon's EC2 command-line tools.

Download Credentials

Query interface credentials

Use this pair of strings with tools - such as [euca2ools](#) - that utilize the "query interface" in which requests and parameters are encoded in the URL.

Query ID: kKPtnjKyqvTGHEEOuSVCYRnhqdxT1Chq3pmoCQ
Secret Key: cl7jSJgmUdw7jjxYwnLEJVxG9j3c5erHtlkAw

Hide keys

Fig. 3.3 Credentials administration in Eucalyptus

Euca2ools are administrative command-line tools for interacting with Web services that are compatible with Amazon EC2 and S3 services. With euca2ools it is possible, for example, to manage the status of the network, manage the images of the cloud, or create new instances. These tools were inspired by command-line tools distributed by Amazon.

3.2.2. OpenNebula

OpenNebula [22] is also an open source solution for Infrastructure as a Service network, started in 2005 as a research project at Universidad Complutense of Madrid.

To interact with the cloud, OpenNebula implements a new protocol called OCCI [23] (Open Cloud Computing Interface). This standard is an attempt by the industry to offer an open source protocol to interact with any type of cloud, but for now, only OpenNebula and INFN (*Istituto Nazionale di Fisica Nucleare*, Italy), implement this standard. Also, OpenNebula implements a subset of the main functions of the EC2 API developed by Amazon.

3.2.2.1. Network nodes

The basic components of an OpenNebula cloud are two, distributed in a classical cluster configuration (**Fig. 3.4**):

Front-end: Is the central node of the cloud. Implements the OCCI interface and the restricted EC2 interface on one side and interacts with the rest of nodes on the other side. Executes the OpenNebula and cluster services.

Cluster Nodes: In each node runs a hypervisor (Xen, KVM, VMware or VirtualBox) that provides the resources needed by the Virtual Machines. The image system communication between the front-end and the clusters is done by a shared file system between the nodes (NFS protocol).

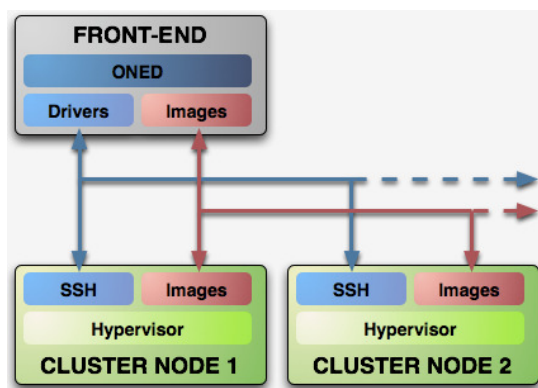


Fig. 3.4 OpenNebula cloud

3.3. Comparison between IaaS solutions

In this section we will compare the two main solutions for private networks based on open source: OpenNebula and Eucalyptus. We have to note that these IaaS solutions are evolving quickly.

The first thing to evaluate is the difficulty for install the IaaS solution. Although Eucalyptus has more number of logical nodes than OpenNebula, Eucalyptus is easier to install. The installation of Eucalyptus is always the same while OpenNebula may require extra steps in the installation to use all its functions. Also, the install documentation is clearer in Eucalyptus than in OpenNebula.

In the hypervisor view, OpenNebula wins. Eucalyptus only offers KVM or Xen solutions as a hypervisor, while OpenNebula offers KVM, Xen and VMWare solutions. Eucalyptus also offers the VMWare solution, but this feature is only available on the enterprise edition, which is not open source.

The strongest point of Eucalyptus is that it mimics the EC2 network of Amazon, making possible that software written to interact with Amazon can be able to interact with Eucalyptus without changes. But this point is also the weakness of Eucalyptus, because to make possible this compatibility, Eucalyptus does not offer any special functionality that is not supported by Amazon.

In contrast, OpenNebula offers more added options than Eucalyptus, and can work with the Amazon EC2 network like a special node inside the OpenNebula network. But these added options don't work with the Amazon network, and the API compatibility with the Amazon network is limited, so the software written to interact with this type of network must be specific for OpenNebula.

Another important point is the support of the community for each solution. This point is difficult to evaluate in an unbiased manner, so to do this, we will based on the discussion forums for each one.

OpenNebula offer a discussion mailing list for the community. The amount of mails per month is around 200 messages (6 or 7 mails per day). Instead, Eucalyptus offers to the community different forums for discussion. The amount of posts in these forums is higher than the mailing list from OpenNebula, and also, a forum solution is more intuitive and attractive for an user than a mailing list solution.

Finally, we need to have the clear idea that an IaaS technology is only an instrument to create new services, not a final solution for a client, these words from David Chernicoff tell us this concept:

- "The vast majority of people following the progress of cloud technologies are going to be consumers of cloud services, not creators. To them, it is the service that will matter, not the underlying technologies. As technologists, we have an annoying tendency to get swept up with our own personal beliefs and a fascination with what's cool at the moment,

especially those of us who evaluate technology and write about it on a regular basis.

But this is an issue that needs to be looked at from the business perspective and not the IT view. It just doesn't matter what the back end is as long as it delivers the services that our users need. Unless we are hosting and creating our own cloud services, the technology that drives the cloud is far less important than the business value of the delivered services." [24]

CHAPTER 4. IaaS SERVICE APPROACH APPLIED TO SECURITY SERVICES

4.1. Service description: use case

Nowadays, Safelayer Secure Communications distributes TrustedX in two different ways:

- **Hardware appliance:** A physical secure device that contains the TrustedX service.
- **Virtual appliance:** A virtual image of TrustedX service that is run on a virtual machine.

In both solutions, the final user has to manage their own servers to deploy the software. With this project we propose to offer a new way of distribution of the TrustedX using the new characteristics of the cloud computing.

For our service we use a hybrid cloud, that means that the service administrator can choose between a public cloud and our private cloud.

As a public cloud we choose the Amazon EC2 cloud. This choice is due to the flexibility that offers Amazon as a provider of IaaS. Amazon EC2 offers a business model in which a company can put an image on the Amazon EC2 with execution taxes over the price of a normal instance.

In this business model, Amazon gets the normal price of execution for an instance plus the 3% extra price associated to the image, and the image owner gets the 97% extra price associated to the image. This extra price is assigned only by the image owner and can be, for example, a monthly fee or a usage-based fee. This service is called Amazon DevPay [25].

As a private cloud we choose the Eucalyptus software. This choice is due to Eucalyptus was made to mimic the Amazon EC2 network, so, we can have a hybrid cloud that will work and respond on the same way. So we can interact with both networks through the same connector.

Both networks, public and private networks, will be managed by the same web application (**Fig. 4.1**). So, an administrator can interact with both networks if he has the right permissions. To interact with the Amazon EC2 network, an administrator must have an account on Amazon, and to interact with Eucalyptus an administrator must have an account on our private Eucalyptus network.

But in both cases, the web application will work on the same way as a front-end. The only difference in the application is the destination network to work with it.

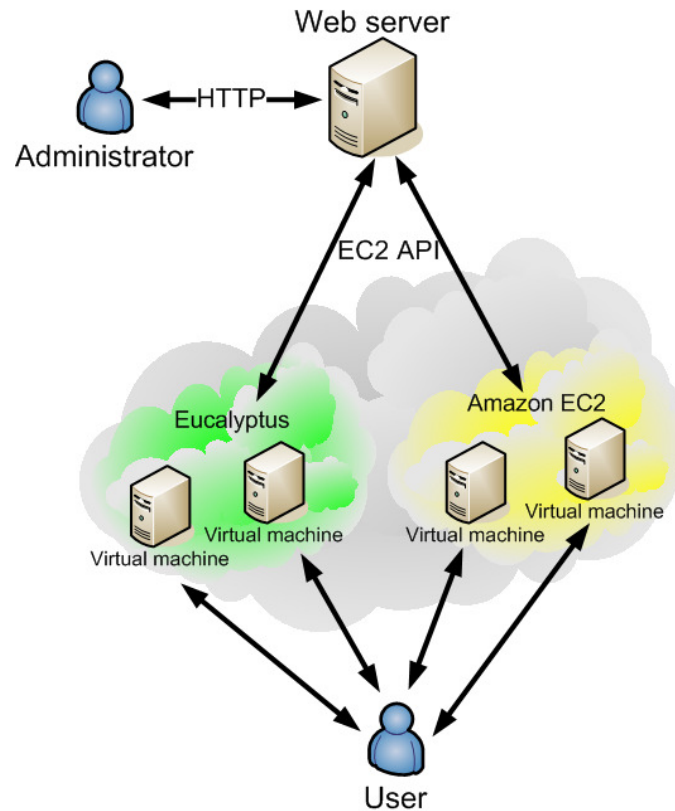


Fig. 4.1 Interaction with the hybrid cloud

The second purpose of the web server will be to do the initial configuration of the TrustedX platform automatically. Nowadays, this initial configuration is done by the administrator through an SSH connection with a Linux console.

With the new proposed configuration, administrators only have to specify the requested configuration file that they want to load into the TrustedX, and the web server will configure the TrustedX automatically, through an SSH API.

Also, as an advanced service, we propose a specific configuration with a cluster of virtual machines that will run TrustedX as a high availability service.

In short, with this new service, an administrator can deploy a TrustedX platform without any knowledge of hardware or software configuration, and with a short time response.

4.2. Eucalyptus network

In this section we can see our particular eucalyptus network configuration, that will manage the private network inside the hybrid cloud. The Amazon configuration is not mentioned because it is a public network, so that we cannot choose the network configuration.

4.2.1. Nodes distribution

To explore all features of an Eucalyptus network, we decide to use a multicluster configuration. But we decide to use only 2 different clusters, this decision is due to use the minimum number of nodes in a test scenario environment.

So we have two different clusters, each cluster will be deployed over a different LAN (Local Area Network) that will be unroutable by the final users, so these LANs will be private networks. Over this LANs we will have the node controllers that will run the virtual machines.

To connect these LANs to the external world, we must have an extra LAN that will be visible by the final users, this is because in this LAN we will have the cloud controller and this node will be the access point of the network (**Fig. 4.2**).

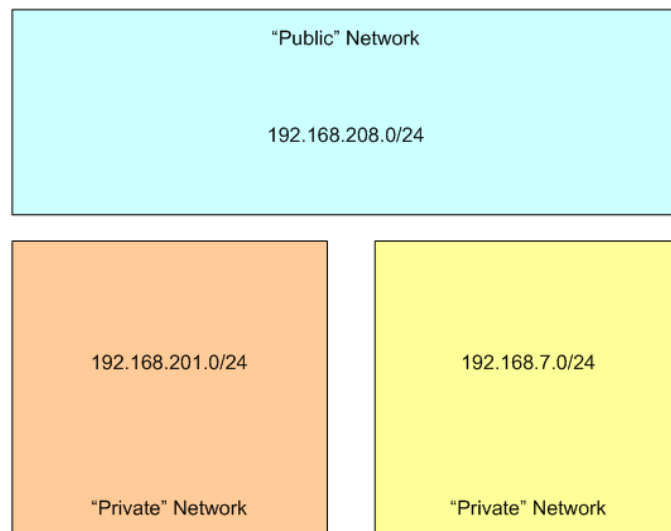


Fig. 4.2 Eucalyptus network configuration

Over these LANs, as shown in the chart below (**Fig. 4.3**), we have only 4 nodes:

- **Node 192.168.208.166 / 192.168.201.2:** This node will be connected to two different LANs (192.168.208.0/24 and 192.168.201.0/24), so it will

have two different interface networks. Moreover, this node will run the next services: Cloud Controller, Cluster Controller and the Walrus. So this node will be the access point to the network. Although each service can be deployed over a different physical machine, we decide to deploy these three services over a single machine to reduce the number of computers on the scenario.

- **Node 192.168.201.3:** This node will run the Node Controller service, and its Cluster Controller is the 192.168.208.166 node.
- **Node 192.168.208.168 / 192.168.7.236:** This node will be connected to two different LANs (192.168.208.0/24 and 192.168.7.0/24), so it will have two different interface networks. Moreover, this node will run the Cluster Controller. Each Cluster Controller has to be on the same network that the Cloud Controller, so this node will have 2 interfaces: one connected to the Cloud Controller, and the other one, connected to its Node Controllers.
- **Node 192.168.7.231:** This node will run the Node Controller service, and its Cluster Controller is the 192.168.208.168 node.

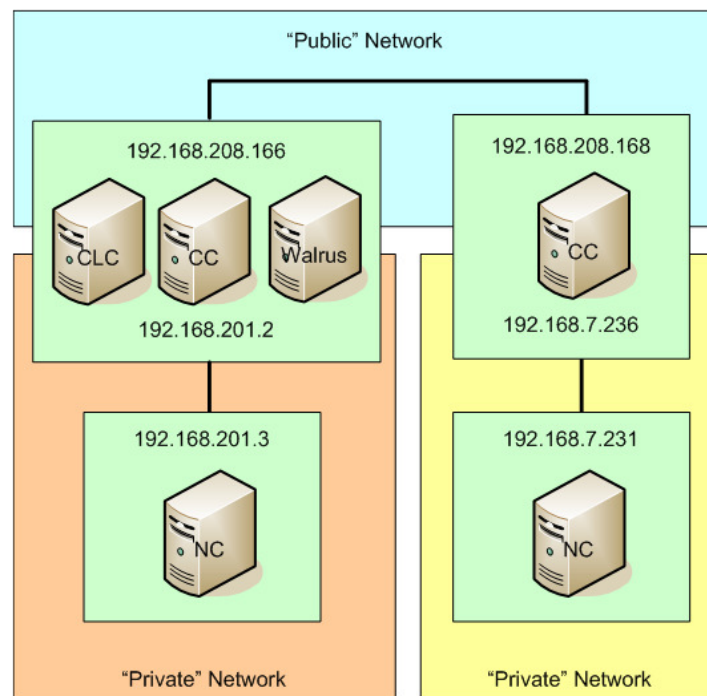


Fig. 4.3 Eucalyptus physical network implementation

We use the managed no-VLAN configuration that offers Eucalyptus. In this scenario we must have a public network that can be visible by all the users. In this public network we will have the Cloud Controller node and all our Clusters Controllers. To configure this scenario we have to modify the configuration file located in `"/etc/eucalyptus/eucalyptus.conf"` on each node.

4.2.2. Instances communication

All the virtual machines deployed over the Node Controllers will have two different IPs, a public IP and a private IP:

- **Public IP:** An IP of the same range network as the Cluster Controller. This is the IP that users will use to connect to the virtual machines, so this is the main reason that this network must be accessible by the users, if not; no users can reach their virtual machines.
- **Private IP:** An IP on the range of private IPs that must be different from the IP range of the network that the Node Controllers belong to (For example, the private IP range for the virtual machines deployed on the network 192.168.7.0/24 is the 10.35.24.0/24). This IP is the real IP that the instance will have and must be used to communicate between instances located on the same cluster.

With this configuration, a virtual machine only can see their private IP. The association between a public IP and a private IP is made in their Cluster Controller. This association is like a NAT (Network Address Translation), so each package received to the public IP of a virtual machine is responsible of the cluster controller associated to route to their private IP for reach the final destination.

The difference between a NAT and this service is that each virtual machine has a different and unique public IP that is not shared by other virtual machine. The main advantage of this technique, is that is possible to reserve a specific public IP to a service, making it independent of the real virtual machine that is doing the service.

For example, if we see the next figure (**Fig. 4.4**), we can see a ping petition to a virtual machine. From a user's point of view, the virtual machine has an IP 192.168.208.3. This virtual machine only noticed that its IP is 10.35.24.1, but the Cluster Controller has assigned to this virtual machine a public IP 192.168.208.3. When the request reaches the network of the cluster controller, this node acts as a NAT and redirects the package to the right destination.

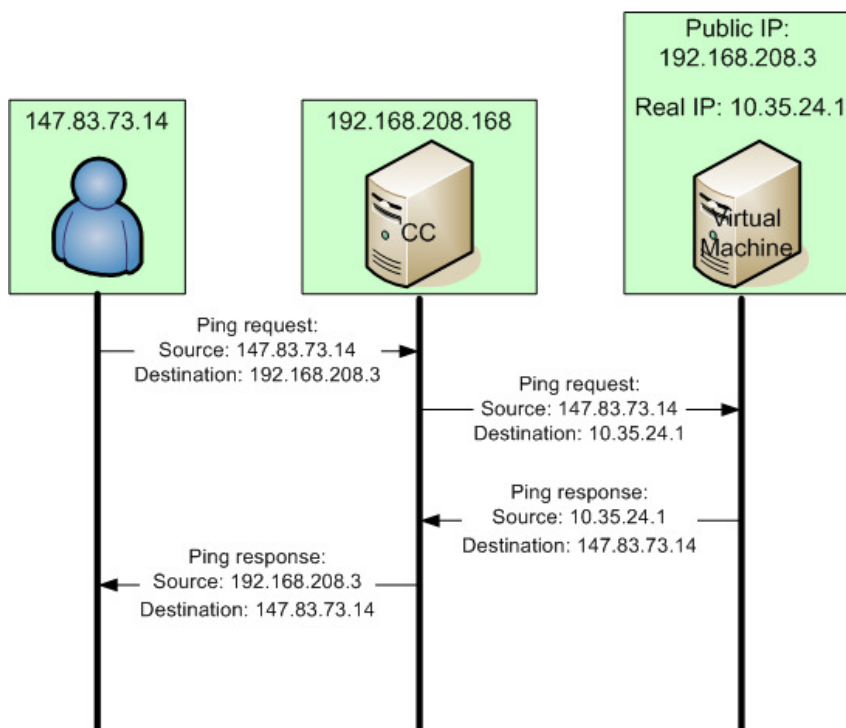


Fig. 4.4 NAT example with a virtual machine on Eucalyptus managed mode

This configuration also allows the possibility of making firewall rules. This is named security-groups on Eucalyptus (or Amazon) terminology, and allows that each virtual machine in the same security group will have the same firewall rules. If a destination port of a package is to a forbidden port according to its security-group, the package is discarded on the Cluster Controller.

4.2.3. Network configuration

The Walrus node must have visibility to all node controllers of the network. This is because the Walrus node contains the whole images of the system and a node controller can request to the Walrus to send a specific image file.

For this purpose a cluster controller that is connected to two networks must act as a router between both networks. So we have to configure the iptables to allow the traffic through the cluster controller:

- `iptables --append FORWARD --in-interface eth1 -j ACCEPT`
- `iptables --append FORWARD --in-interface eth0 -j ACCEPT`
- `iptables --table nat --append POSTROUTING --out-interface ppp0 -j MASQUERADE`
- `echo 1 > /proc/sys/net/ipv4/ip_forward`

With these rules, the cluster controller will act as a router, but we have to configure on the Walrus and on the node controllers a new route to reach the other network across the cluster controller:

- On the Walrus, we have to add a destination route for the 192.168.7.0 network: `route add -net 192.168.7.0/24 gw 192.168.208.168`
- On the 192.168.7.231 node, we have to add a destination route for the 192.168.208.0 network: `route add -net 192.168.208.0/24 gw 192.168.7.236`

Now each node controller can reach the Walrus node and the Eucalyptus network is correctly configured to work in a managed no-VLAN mode. The final step is to register each service into the network with the help of the `euca2ools` commands:

- On the 192.168.208.166 node: This node is the cloud controller, so we have to register the main services: the walrus service, the Cluster Controller and Storage Controller services:
 - o `euca_conf --register-walrus 192.168.208.166`
 - o `euca_conf --register-cluster Cluster_A 192.168.208.166`
 - o `euca_conf --register-sc Cluster_A 192.168.208.166`
 - o `euca_conf --register-cluster Cluster_B 192.168.208.168`
 - o `euca_conf --register-sc Cluster_B 192.168.208.168`
- On each Cluster Controller we have to register its Node Controllers, so also on the 192.168.208.166 node we have to register its Node Controller:
 - o `euca_conf --register-nodes "192.168.201.3"`
- And on 192.168.208.168 node we have to register its Node Controller:
 - o `euca_conf --register-nodes "192.168.7.231"`

After the registration of each node, we can use the command "`euca-describe-availability-zones verbose`" to know the number of virtual machines than each cluster can run (**Fig. 4.5**).

```
[root@localhost /]# euca-describe-availability-zones verbose
AVAILABILITYZONE      Cluster_A      192.168.208.166
AVAILABILITYZONE      |- vm types    free / max    cpu    ram    disk
AVAILABILITYZONE      |- m1.small    0005 / 0005  1     512   5
AVAILABILITYZONE      |- m1.large    0005 / 0005  1     512   7
AVAILABILITYZONE      |- c1.medium   0002 / 0002  1     1024  7
AVAILABILITYZONE      |- m1.xlarge   0002 / 0002  2     1024  10
AVAILABILITYZONE      |- c1.xlarge   0001 / 0001  4     2048  10
AVAILABILITYZONE      Cluster_B      192.168.208.168
AVAILABILITYZONE      |- vm types    free / max    cpu    ram    disk
AVAILABILITYZONE      |- m1.small    0002 / 0002  1     512   5
AVAILABILITYZONE      |- m1.large    0002 / 0002  1     512   7
AVAILABILITYZONE      |- c1.medium   0002 / 0002  1     1024  7
AVAILABILITYZONE      |- m1.xlarge   0001 / 0001  2     1024  10
AVAILABILITYZONE      |- c1.xlarge   0000 / 0000  4     2048  10
```

Fig. 4.5 Virtual machines per cluster

4.3. TrustedX over Eucalyptus

With Eucalyptus we can only have a KVM or a Xen as a hypervisor in our Node Controllers. We choose Xen as hypervisor because we do not have computers with CPU support for virtualized instructions, that is a requirement to run the KVM hypervisor.

As Linux distribution we have chosen CentOS [26]. This choice is due to officially Xen support in CentOS. Moreover, CentOS is a very stable distribution for servers because it is derived from source code released by Red Hat Inc [27]. Also, Eucalyptus offers official support for CentOS distribution.

4.3.1 TrustedX over Xen

To make possible that TrustedX will run with Eucalyptus is mandatory that the image runs over Xen.

TrustedX is distributed only for run over a physical server or in VMWare format to deploy over a virtual machine, but is not distributed on Xen. For this reason we have to adapt a VMWare image to a Xen image.

The first step is to reduce the maximum hard disk capacity. In VMWare, the files of an image can grow or decrease up to the maximum hard disk capacity, but in Xen, the size of an image is the maximum hard disk capacity, and any type of compression is not allowed.

TrustedX over VMWare is configured to have 20GB of hard disk capacity, so decrease the maximum capacity is mandatory to have manageable images. For this purpose we use GParted liveCD [28]. GParted is a small bootable GNU/Linux application that can manage the disk partition of the file system.

The second step is make a compatible Xen image. VMWare keeps the hard disk information in different files, but Xen only allow a unique file for the file system folders. So, the first operation is to put together all the VMWare files with a tool included with VMWare package:

```
- vmware-vdiskmanager -r vmware_image.vmdk -t 0  
temporary_image.vmdk
```

The second operation is make the conversion of the generated file to a Xen compatible image. For this conversion we use the `qemu`, a package for Linux:

```
- qemu-img convert -f vmdk temporary_image.vmdk -O raw  
xen_compatible.img
```

Now we have an image compatible with Xen format, but is still unbootable. A typical Linux installation can have more than one partition, but a valid Xen image only can have the root partition. So, we have to extract the root partition

from the global image. To do this, we use the command "parted", to know the extension of each partition (**Fig. 4.6**).

```
[root@localhost vmware]# parted xen_compatible.img
GNU Parted 1.8.1
Using /home/vmware/xen_compatible.img
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) U
Unit? [compact]? b
(parted) p

Model: (file)
Disk /home/vmware/xen_compatible.img: 21474836479B
Sector size (logical/physical): 512B/512B
Partition Table: msdos

Number  Start      End          Size         Type         File system  Flags
  1      32256B    106928639B  106896384B  primary     ext3         boot
  2      106928640B 4400524799B 4293596160B primary
```

Fig. 4.6 Example of parted command

In the example of this figure, we only have interested in the first partition, that it contains the root partition, so the important info is the start Byte (32256) and the size in Bytes of the partition (106896384).

Now, for extract this partition we use the Linux command "dd":

- `dd if=xen_compatible.img of=rootfs.img bs=512 skip=63 count=208782`

Note that we use a block size of 512 (bs=512), so the start Byte is 63 (32256/512=63) and the size of the partition is 208782 (106896384/512=208782). This change is because it is faster to work with blocks of 512 Bytes than work with blocks of 1 Byte.

Now we have a file with only the root partition, but this partition do not have the right kernel to work with Xen hypervisor. The right kernel depends on the Linux distribution than we are using in our image, in our case, TrustedX is deployed on a RedHat Linux distribution.

To match a correct pair of kernel and ramdisk that will run over Xen and Eucalyptus, we use the same kernels that Eucalyptus offers as a prepacked Images. Eucalyptus doesn't offer prepacked image for RedHat, but it does for CentOS. CentOS is based on the RedHat code, so these files are compatible.

Now, we have the ramdisk and kernel files to run with our root partition, but we must add the kernel modules inside the root partition. So, we have to mount the root file system to add these files that will make possible to run over Xen:

- `mount -o loop rootfs.img /mnt/destination_folder`

In `/mnt/destination_folder` now we have the whole file system of our image. Inside the folder `"lib/modules"` of the image we must add the module files that will need the kernel to make it bootable in a Xen environment.

Also to make possible that our image will be bootable over Eucalyptus we must be sure that the image will not have any information related to its MAC address, and we have to be sure that the network interface will use a DHCP server to get its IP address because this info will be provided by Eucalyptus, failure to do so may leave our instance without network.

4.3.2 Uploading an image on Eucalyptus

Once we have the right image, kernel and ramdisk created to work with Eucalyptus, we have to upload and register these elements on the Walrus directory.

For this procedure, we use the `euca2ools` commands. First we have to upload and register the kernel (**Fig. 4.7**) and ramdisk into the system:

```
[root@localhost image]# euca-bundle-image -i vmlinuz-2.6.24-19-xen --kernel true
Checking image
Tarring image
Encrypting image
Splitting image...
Part: vmlinuz-2.6.24-19-xen.part.0
Generating manifest /tmp/vmlinuz-2.6.24-19-xen.manifest.xml
[root@localhost image]# euca-upload-bundle -b kernel-bucket -m /tmp/vmlinuz-2.6.24-19-xen.manifest.xml
Checking bucket: kernel-bucket
Creating bucket: kernel-bucket
Uploading manifest file
Uploading part: vmlinuz-2.6.24-19-xen.part.0
Uploaded image as kernel-bucket/vmlinuz-2.6.24-19-xen.manifest.xml
[root@localhost image]# euca-register kernel-bucket/vmlinuz-2.6.24-19-xen.manifest.xml
IMAGE     eki-901C1374
```

Fig. 4.7 Example of uploading a kernel

We see that the operation of uploading a kernel returns an EKI (Eucalyptus Kernel Image). The same operation with a ramdisk returns an ERI (Eucalyptus Ramdisk Image). With this two ids, we can upload our image associating with its defaults kernel and ramdisk (**Fig. 4.8**):


```

[root@localhost image]# euca-bundle-image -i trustedx-32b.img --kernel eki-901C1374 --ramdisk eri-C3B31792
Checking image
Tarring image
Encrypting image
Splitting image...
Part: trustedx-32b.img.part.0
Part: trustedx-32b.img.part.1
Part: trustedx-32b.img.part.2
      :
      :
Part: trustedx-32b.img.part.21
Part: trustedx-32b.img.part.22
Generating manifest /tmp/trustedx-32b.img.manifest.xml
[root@localhost image]# euca-upload-bundle -b trustedx-bucket -m /tmp/trustedx-32b.img.manifest.xml
Checking bucket: trustedx-bucket
Creating bucket: trustedx-bucket
Uploading manifest file
Uploading part: trustedx-32b.img.part.0
Uploading part: trustedx-32b.img.part.1
Uploading part: trustedx-32b.img.part.2
      :
      :
Uploading part: trustedx-32b.img.part.21
Uploading part: trustedx-32b.img.part.22
Uploaded image as trustedx-bucket/trustedx-32b.img.manifest.xml
[root@localhost image]# euca-register trustedx-bucket/trustedx-32b.img.manifest.xml
IMAGE   emi-87D113E0

```

Fig. 4.8 Example of uploading a image

Now, we have an EMI (Eucalyptus Machine Image) that contains all the information to run our image.

4.4. Eucalyptus monitoring with Nagios

Nagios is focused on monitoring service/host availability. For this purpose, Nagios runs scripts over the monitored host. In a network environment, the communication between nodes is based on a client-server structure.

In the Nagios implementation on our project (**Fig. 4.9**), we choose as a server the 192.168.208.168 node. This decision is based on the fact that is the less used node (only runs the CC service), since NCs has to deploy all the virtual machines and the front-end node has also CLC, CC and Walrus services.

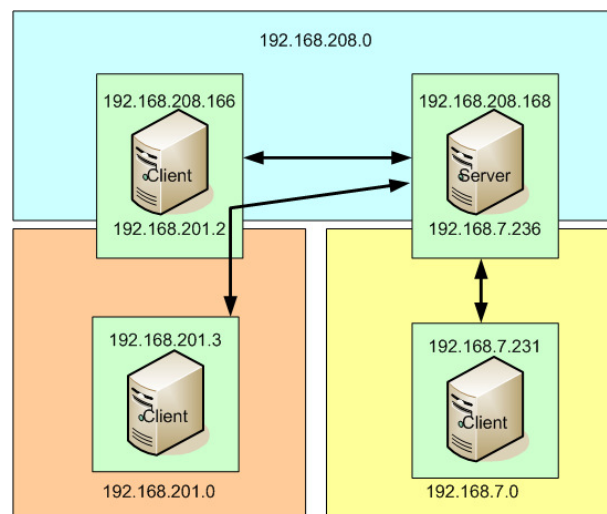


Fig. 4.9 Nagios network implementation

Communications between the server and clients are encrypted by a SSL connection to ensure the authenticity of the server (**Fig. 4.10**). Once the server is authenticated, it can execute commands like a localhost scenario. For this purpose, each client has to be running the NRPE daemon with the info of the Nagios server.

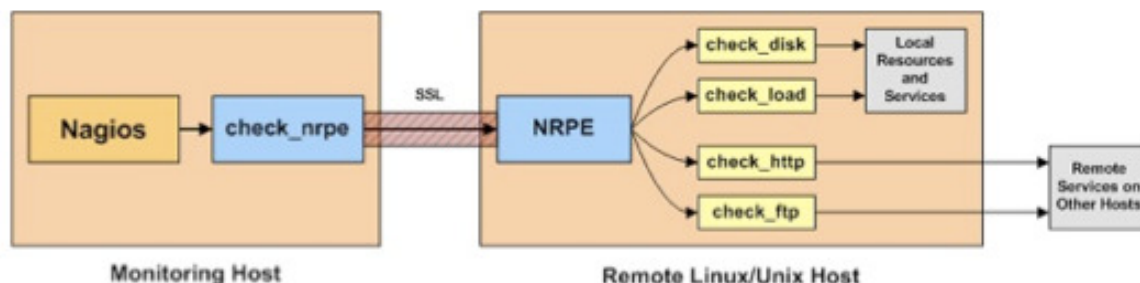


Fig. 4.10 NRPE communication

The server has also to run the NRPE daemon because we have to monitoring their services too. Also, the Nagios server has to run the Nagios daemon to control the whole network and a http server to display the collected network information.

In the next figure (**Fig. 4.11**), we can see all the services monitored by Nagios in our network. For each node we are monitoring standard services to check the proper state of the node:

- **Ping service:** Monitoring this service we know if the node is alive.
- **Free Space:** Monitoring the disk free space we can prevent future failures. Short space on the hard disk can cause the node to work slower than normal and may even cause more serious errors if the node runs out of space.
- **Total CPU process:** The more processes running in a node, the more resources will be needed in terms of CPU and RAM.

Also, for each node we are monitoring the status of specific eucalyptus services that they are running.

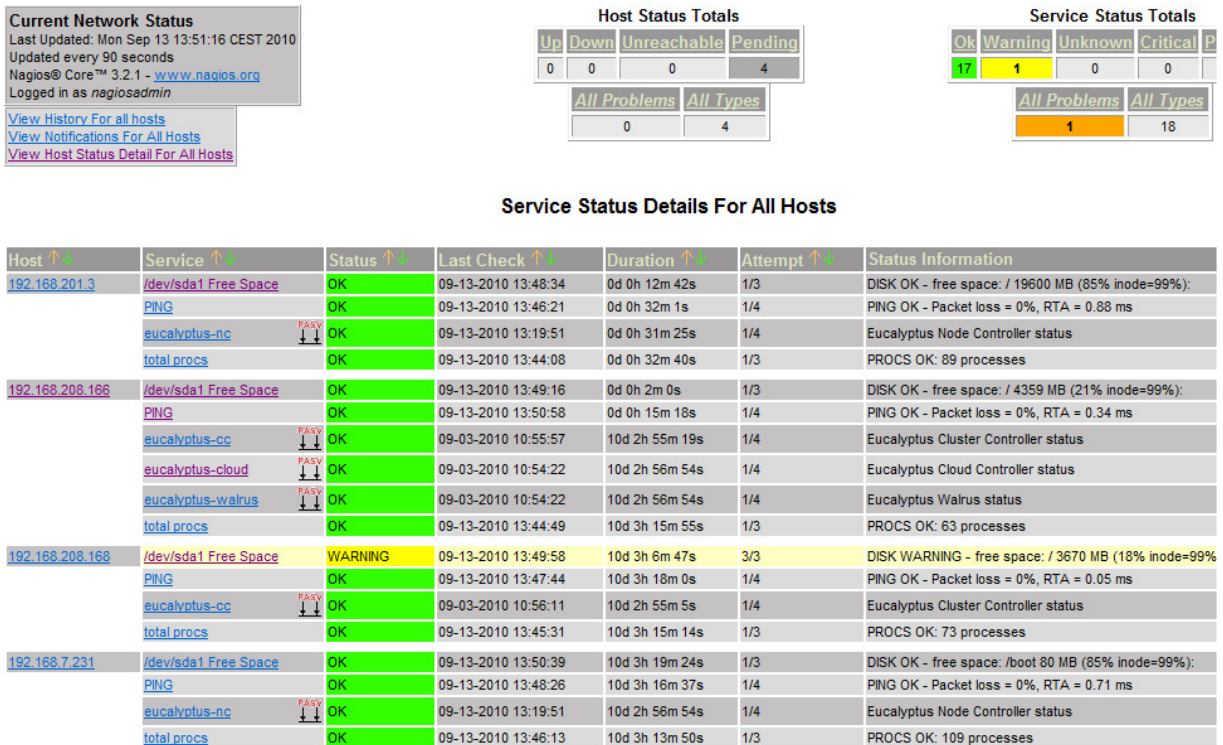


Fig. 4.11 Network monitoring log

4.5. Web portal

An administrator can interact with our hybrid cloud with a web portal. This web has been deployed over a Tomcat server with the help of the MVC Spring Framework, so all the view part of the web is independence of the logic part.

Also, this web has an internal database to save the configuration profiles of administrators that have access to the web. As a database, we use a MySQL server, and the communication between the database and the web portal is done through Hibernate [29].

Hibernate is an object-relational mapping (ORM) library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions.

4.5.1. User configuration

As we can see on the next figure (Fig. 4.12), an authenticated administrator can change their password and their credentials on the private network (Eucalyptus network) and on the public network (Amazon network). The only parameter that

is not allowed to modify is the user name, this is for ensure that two different users will not have the same user name.

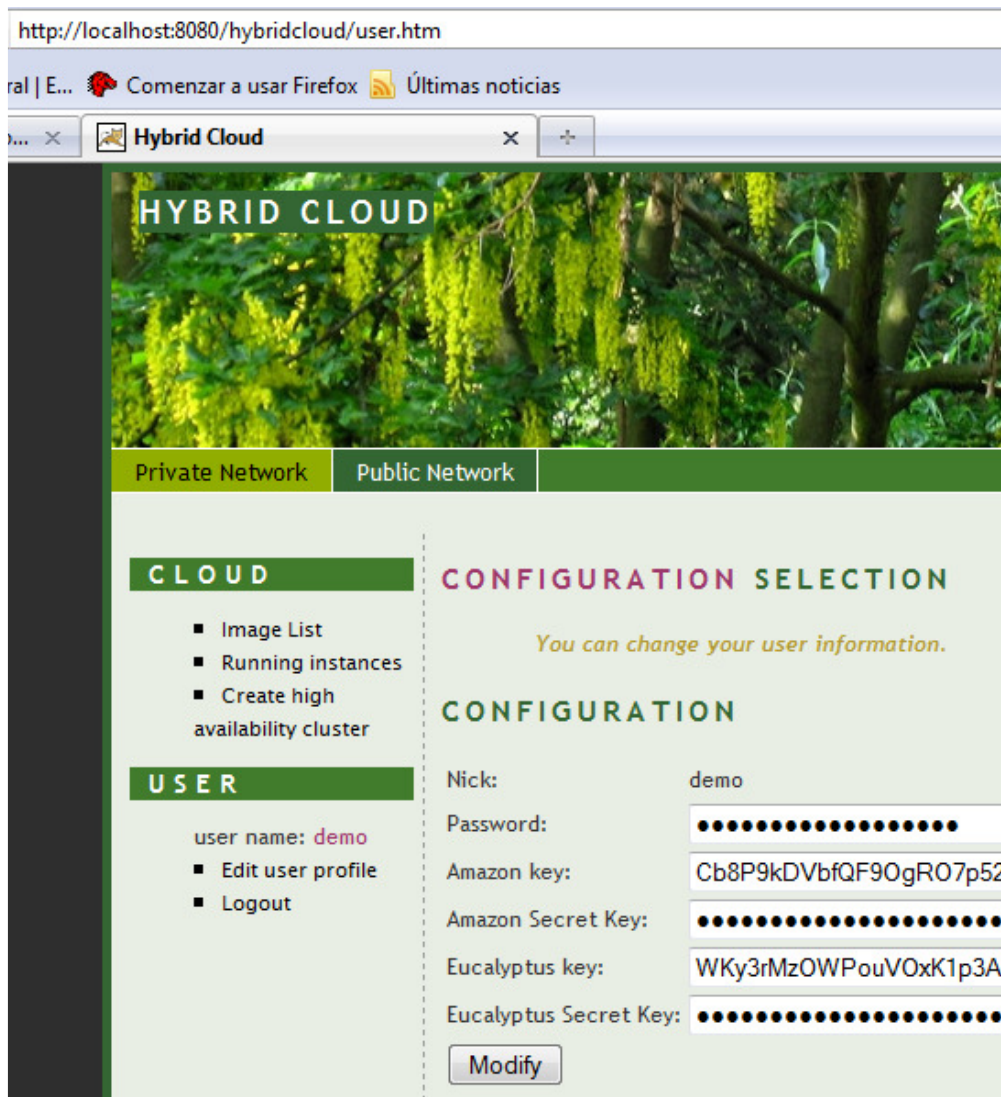


Fig. 4.12 Web user configuration

For use each network, an administrator have to put their appropriate credentials, this credentials are obtained on the registration process on each network. If an administrator only has the credentials of one network, he only can use this network.

Each administrator must have different credentials because each administrator only must administrate their own instances, so he must have a unique user on each network.

4.5.2. Instance management

The main purposes of the web are two:

- Know the status of the network: Know how many virtual machines are running and the IPs of these virtual machines (**Fig. 4.13**).

The screenshot shows a web browser window with the URL `http://localhost:8080/hybridcloud/showinstances.htm`. The page title is "Hybrid Cloud". The interface is divided into several sections:

- HYBRID CLOUD**: A header section with a background image of green foliage.
- Private Network** and **Public Network**: Two tabs for navigation.
- CLOUD**: A sidebar menu with options: Image List, Running instances, and Create high availability cluster.
- USER**: A sidebar menu with options: user name: demo, Edit user profile, and Logout.
- RUNNING INSTANCES INFO**: A main section with a heading and a paragraph: "In this page you can see the number of running instances in the cloud, and the information about each running instance". Below this, it states "Number of running instances: 2".
- INSTANCE LIST**: A table listing two running instances with their details and control buttons.




| INSTANCE LIST | |
|----------------------------|---|
| Instance Info: | |
| Instance Id: | i-49ED0830 |
| Image Id: | emi-3E6816B3 |
| State: | {Code: 16, Name: running, } |
| Launch Time: | Thu Dec 09 12:09:19 CET 2010 |
| Private IP: | 10.35.1.4 |
| Public IP: | 192.168.208.6 |
| Cluster: | Cluster_B |
| Stop this instance: |  |
| Copy actual configuration: |  |
| Instance Info: | |
| Instance Id: | i-33510735 |
| Image Id: | emi-3E6816B3 |
| State: | {Code: 16, Name: running, } |
| Launch Time: | Thu Dec 09 11:16:55 CET 2010 |
| Private IP: | 10.35.1.3 |
| Public IP: | 192.168.208.2 |
| Cluster: | Cluster_A |
| Stop this instance: |  |

Fig. 4.13 Information about running instances

- Launch, configure and manage a TrustedX: Select to launch a TrustedX with a specific configuration (**Fig. 4.14**), stop a specific TrustedX or save a specific TrustedX configuration.

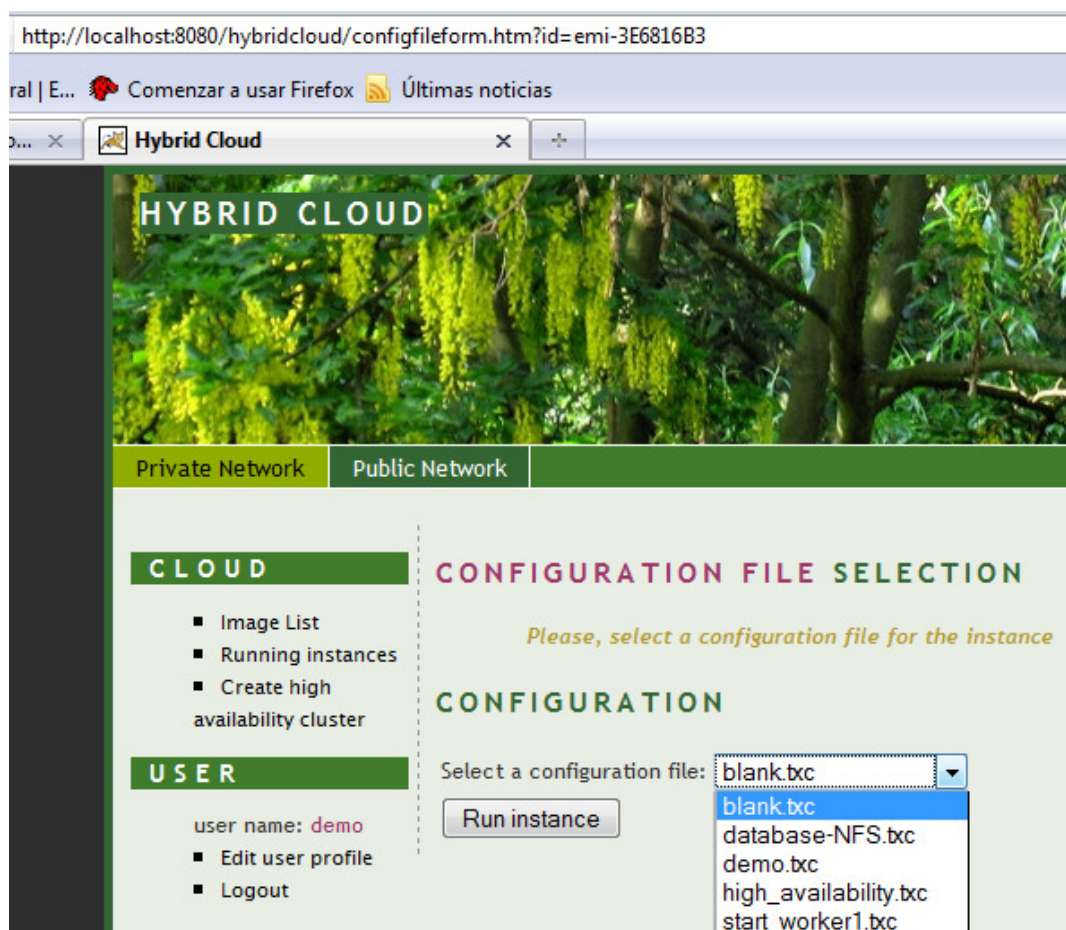


Fig. 4.14 Configuration file selection for a new instance

On the next figure (**Fig. 4.15**) we can see the flowchart of the web. The first petition of the web consist on the authentication of the administrator that wants access to the web.

After the authentication, the administrator has to specify a network, he can choose between the private network (Eucalyptus Network) or a public network (Amazon Network).

Once the administrator has selected the network, he can do different actions:

- **See all running instances:** The administrator can see the information of his own instances. This information includes the public IP and the state of the instance (pending, running, terminate...).

Also an administrator can decide to save the actual configuration of a TrustedX. With this option, a new configuration file will be created with the actual configuration of the TrustedX, this new configuration file, will be available in the web to deploy new TrustedX with this configuration.

On the other hand, an administrator can decide to stop a specific TrustedX. If a TrustedX is put to terminate, this instance of TrustedX will be destroyed.

- **Launch an instance:** The administrator can decide to run a new TrustedX. To do this action, first he has to select the TrustedX image (can be more than one TrustedX version), once he has selected a TrustedX, he has to select a configuration file between the stored configuration files.

Once he has selected the configuration file, the image will be launched, and it will appear into the running instances information.

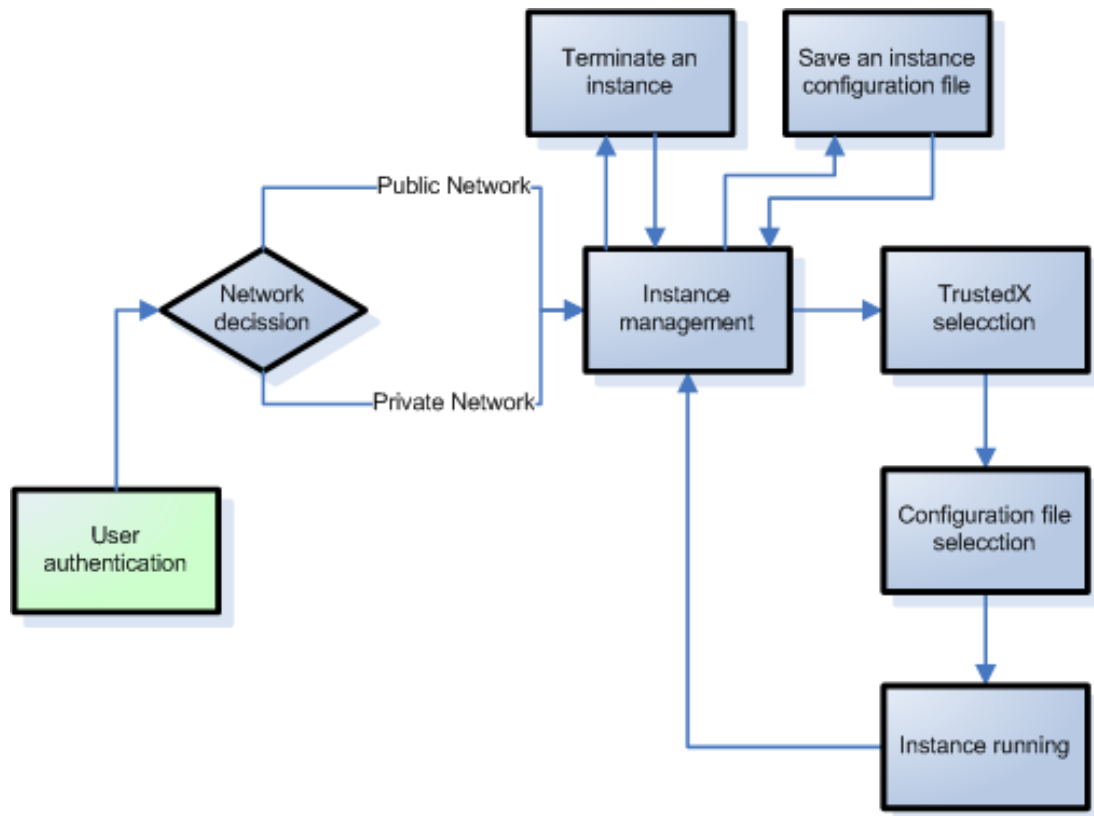


Fig. 4.15 Front-end flowchart

4.5.3. TrustedX with high availability

As an advanced service, we implemented an option on the web portal that an administrator can deploy a high availability cluster of TrustedX.

With this use case, we are consuming more resources on the cloud than with the case of a single TrustedX for doing the same service. The reason to do this use case is based on the capacity increase of petitions that a TrustedX can absorb with a high availability configuration, and the elasticity of a hybrid cloud. With this elasticity, we will be sure that this increase on the consumption of resources will not leave the network to run out of resources.

This cluster consists of four different machines (**Fig. 4.16**):

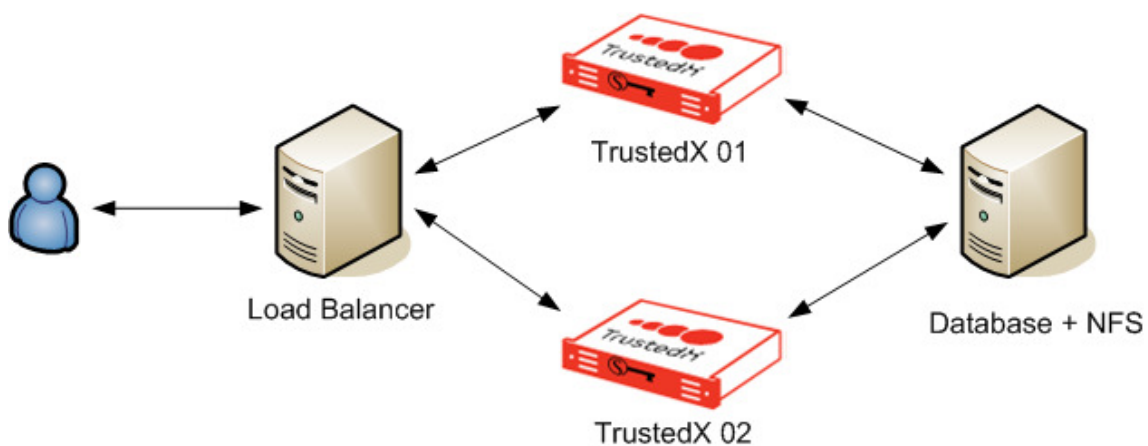


Fig. 4.16 High availability

Load balancer: A virtual machine with a CentOS Linux image that runs an Apache HTTP server [30] with the mod_jk module [31]. With this module, the Apache HTTP server acts as a load balancer.

The work load is distributed between worker nodes with a round robin distribution, that means that is distributed in an statistical way. In other words, a new request have the same chances to be attended by each worker node.

Moreover, the load balancer checks that the selected work node to attend the request is working, if not, this node is discarded and other node will be selected.

- **Database and NFS:** A virtual machine with a CentOS Linux image that runs a MySQL server [32] and a NFS server [33].

The MySQL server acts as a remote database for a TrustedX system. So with this configuration, all TrustedX will share the same database, and therefore, will share the same info.

The NFS server will have the configuration files of the TrustedX. So, each TrustedX will mount the remote directory of the NFS server to access to this configuration. Therefore, all the TrustedX will share the same configuration files, and a modification on the configuration file on a specific TrustedX will be propagated to the others TrustedX.

- **TrustedX01 and TrustedX02:** These TrustedX will be the worker nodes of this cluster and will attend the user requests. These TrustedX will be configured to listen to the 8009 port to attend the request from the load balancer. Also, these TrustedX will be configured to work with an external database and external configuration folder to share the same configuration.

Finally, an user only can see the load balancer as a provider of the service. The load balancer acts as a final point of the cluster and the user doesn't notice if his request is attended by the TrustedX01 or the TrustedX02.

CHAPTER 5. ENVIRONMENTAL STUDY

5.1. Worldwide energy consumption

Available figures (**Fig. 5.1**) indicate that while in 1996 the number of servers was less than 5 million, the projection for 2011 is right less than 40 million [34]. This brings us the problem of providing the energy required, not only to keep data centers working, but also to set up new installations.

Money spending for the servers installed in 1996 was less than 100 \$ billions while the projection for 2011 is about 250 \$ billions. Energy needs are growing dramatically and the implications of this include a number of issues, one of which is environmental.

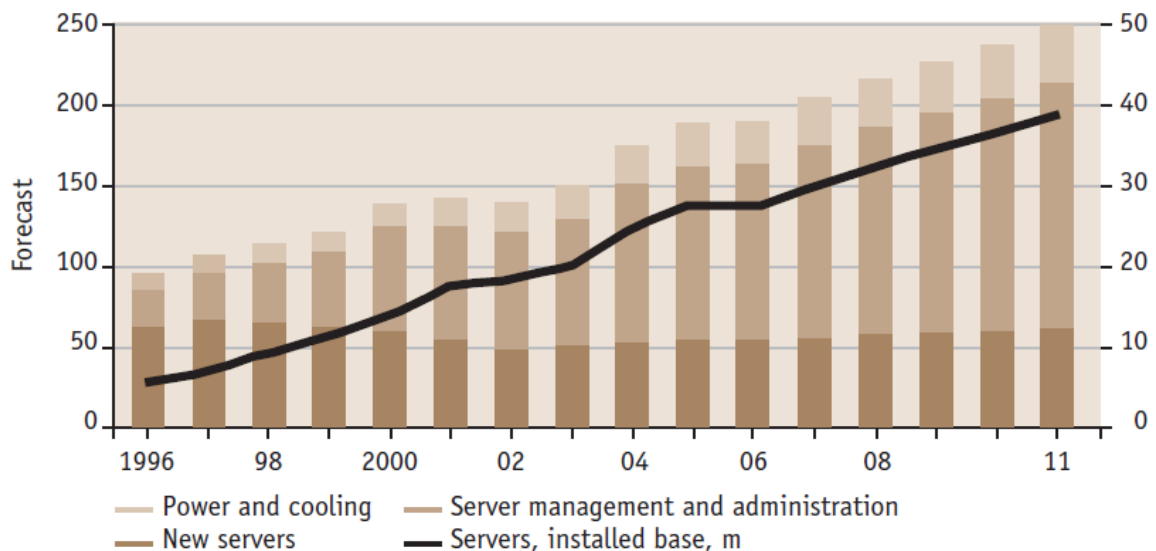


Fig. 5.1 Worldwide spending in servers, in \$billion

In addition, most servers and desktop computers are used only 8-15% of the time when they are running, although most x86 hardware consumes 60-90% of their maximum power when they are in idle state. The analyst firm IDC [35] says that this unused capacity of these servers is approximately equal to:

- 140.000 \$millions.
- 3 years of hardware supplies.
- More than 20 million of servers.

With 4 tons of carbon dioxide (CO₂) emitted annually per server, these unused servers capacity produce more than 80 million tons of CO₂ per year. This is more than the quantity emitted annually in Thailand and more than a half of the emissions in all South American countries.

5.2. Green IT

We live in an era which serious issues on shortage of energy and fossil combustibles, global warming and the greenhouse effect, that it have made the environment care to be a priority for governments and businesses, and the society as a whole.

Information Technologies (IT) cannot be an exception in these issues. Green computing (or green IT), refers to the environmentally sustainable of the IT. This term was born in 1992, in U.S. with the Energy Star [36]: a voluntary labeling program which was designed to promote and recognize energy-efficiency in monitors, climate control equipment, and other technologies.

Nowadays, green IT refers to any strategy for reduce the carbon footprint caused by the Information Technologies. The different strategies to do this objective can be, for example: an intelligent use of the energy, recycling old hardware or better strategies in hardware cooling.

5.2.1. Virtualization as a green IT technology

Virtualization, as shown in the graph (**Fig. 5.2**), is one of the main technologies for the green IT. This is because with virtualization, we can reduce the number of physical computers that we need to be running, and this reduction affects in two different factors:

- **Reduction in energy consumption:** We decrease the number of physical machines thanks to an increase of utilization of our servers. With the reduction on physical machines, we reduce the energy needed to run these machines, but also, the energy needed to cooling the data centers.
- **Reduction in wasted materials:** With a reduction in physical computers, we are saving to purchase more hardware, and for it, we are saving to consume all the materials and the energy needed in the construction and the recycling of this hardware.

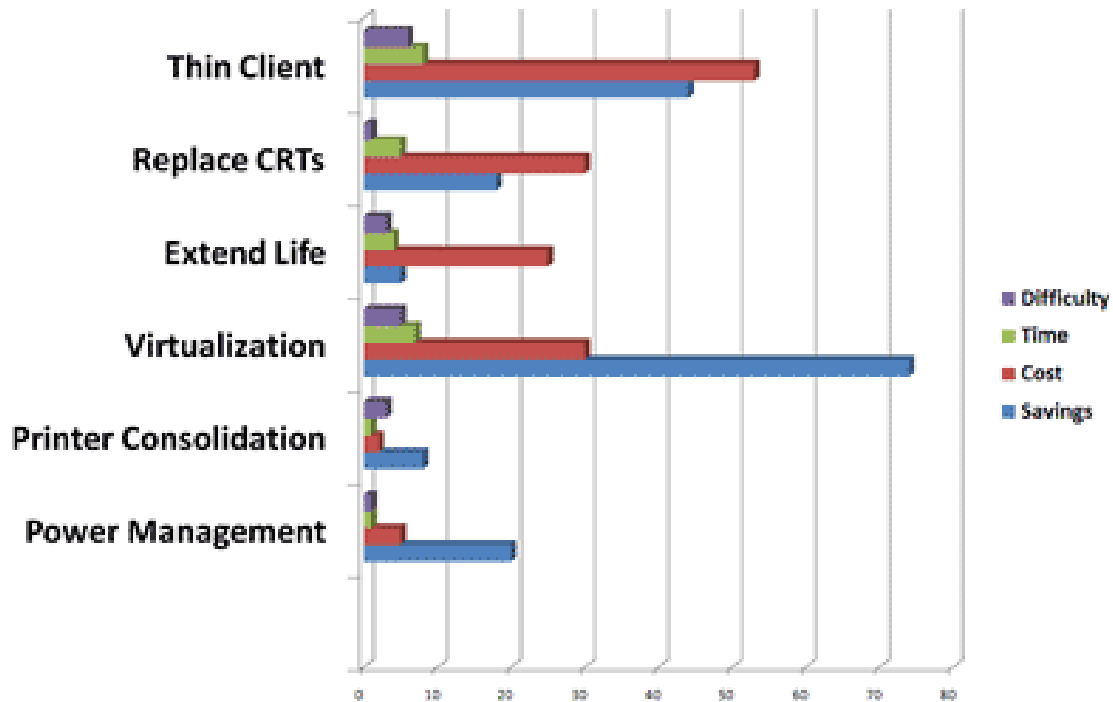


Fig. 5.2 Green IT techniques

5.3. Environmental conclusions for an IaaS network

With an Infrastructure as a Service solution, we are contributing in a green IT Project, reducing our carbon footprint. This is thanks to the use of virtual machines to provide all the services to the final users of our applications, reducing the number of servers needed to run all the services.

Also, with a virtual scenario, we are reducing the maintenance cost of our solution, doing possible to offer a solution that can grow with a more sustainable economy way.

As a practical example of the green IT conscience, Eucalyptus offers a special configuration to save more energy. On this type of configuration, node controllers that are not running any instance are put into sleep mode to save more energy. On this mode, Eucalyptus always choose to deploy a new virtual machine inside a node that is not sleeping. Only a sleeping node will be choose to wake up and deploy a new instance if all the nodes that are awake are running without resources.

CHAPTER 6. PLANNING OF THIS PROJECT

6.1. Project scheduler

The project has lasted 10 months and a half, since 15 of February to the end of December, with a personal dedication of 4 hours per working day. This amounts to 840 hours, exceeding the minimum of 675 hours marked in the project regulations.

In the next figure (**Fig. 6.1**) we can see the Gantt chart that illustrates the project scheduler.

The main threads of the project listed on the Gantt diagram are these:

- **Research:** Search about the different solutions of cloud computing focusing on Infrastructure as a Service solutions.
- **Testing:** Test of the two main IaaS solutions explained in this documentation, Eucalyptus and OpenNebula.
- **Xen:** Testing and configuration of the virtualization technology chosen in this project.
- **IaaS implementation and testing:** Implementation and testing of Eucalyptus with their full capacities (multicenter configuration, Eucalyptus managed mode, nagios monitorization).
- **TrustedX over Eucalyptus:** Adaptation and testing of TrustedX for run over a Xen hypervisor, and later, over Eucalyptus network.
- **Web portal:** Development of a web portal, using the MVC spring framework, for manage the Eucalyptus network through the Amazon API.
- **TrustedX with high availability:** Development and testing of auxiliary services (MySQL service, load balancer service, NFS service) and integration of this service to work with two TrustedX servers on a high availability scenario.
- **Documentation:** Writing and revision of this document and the preparation of the oral presentation.

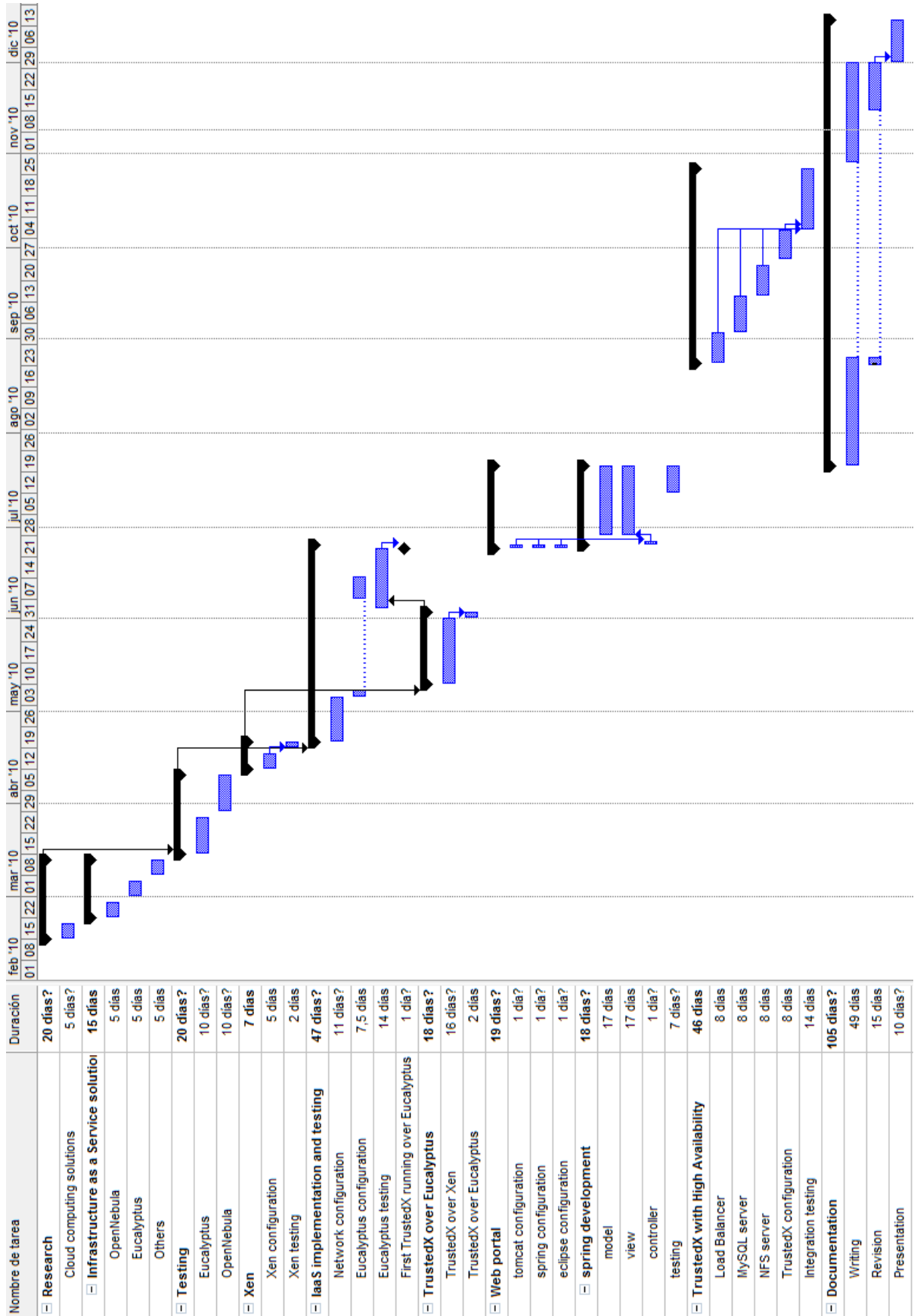


Fig. 6.1 Gantt chart of the project

6.2. Economic analysis

In this section we will see the economic costs of this project. The first cost is related to the human resources. On this project has participated an internship with an agreement scheme enterprise-university. So, to calculate the salary is necessary to multiply the number of hours spent by the base salary **(6.1)**.

$$840 \text{ hours} * 7 \text{ €/h} = 5880 \text{ €} \quad \text{(6.1)}$$

Moreover, it's necessary to add an additional cost in terms of employee maintenance (electricity, telephone costs, ...) that it is fixed on a 15% of the employee salary **(6.2)**.

$$5880 \text{ €} * 1.15 = 6762 \text{ €} \quad \text{(6.2)}$$

The second cost is related to the hardware resources used on the project. Although on the test scenario presented in this project we saw 4 different computers, the scenario has simulated actually with only 2 computers, and we use the VMWare software to create virtual machines to reach the 4 different computers.

So, as a hardware resources we use two computers: *Dell Precision 370* purchased on 2005 and a *Dell Precision T3500* purchased on 2010. We assume that the price for both computer are 1500 €, but we have to consider its age. Also in the acquisition price we assume license costs in terms of operating system and office software. We assume an annual amortization coefficient of 25% of the value of the computers.

Table 6.1. *Dell Precision 370* amortization

| Year | Amortization | Value |
|-----------|---|---|
| 2005-2006 | $0.25 * 1500 \text{ €} = 375 \text{ €}$ | $1500 \text{ €} - 375 \text{ €} = 1125 \text{ €}$ |
| 2006-2007 | $0.25 * 1500 \text{ €} = 375 \text{ €}$ | $1125 \text{ €} - 375 \text{ €} = 750 \text{ €}$ |
| 2007-2008 | $0.25 * 1500 \text{ €} = 375 \text{ €}$ | $750 \text{ €} - 375 \text{ €} = 375 \text{ €}$ |
| 2008-2009 | $0.25 * 1500 \text{ €} = 375 \text{ €}$ | $375 \text{ €} - 375 \text{ €} = 0 \text{ €}$ |

Table 6.2. *Dell Precision T3500* amortization

| Year | Amortization | Value |
|-----------|---|---|
| 2010-2011 | $0.25 * 1500 \text{ €} = 375 \text{ €}$ | $1500 \text{ €} - 325 \text{ €} = 1125 \text{ €}$ |

So, as we can see on the **table 6.1**, the computer *Dell Precision 370* is fully amortized in 2010 and the actual value of this computer is 0 €. Instead, as we can see on the **table 6.2**, the computer *Dell Precision T3500*, is in its first year of amortization, and its final value on this year is 1125 €, so, we have to count 375 € as a hardware costs.

As a software spends we don't have any associated cost because the whole software used on this project is opensource and its license is free.

Finally, as a furniture expenses, we assume a desk and a chair with a value of 300 €, and a amortization rate of 10% per year **(6.3)**.

$$300 \text{ €} * 0.1 = 30 \text{ €} \quad \textbf{(6.3)}$$

So, as we can see on the **Table 6.3**. the final costs of this project is 7167 €.

Table 6.3. Expenses

| Item | Cost |
|-----------------|---------------|
| Human resources | 6762 € |
| Hardware | 375 € |
| Software | 0 € |
| Furniture | 30 € |
| Total | 7167 € |

CHAPTER 7. CONCLUSIONS AND FUTURE LINES

7.1. Conclusions

Cloud computing is a topic that is growing faster than other technologies used to, so it can be possible to think that cloud computing will be broadly use in the near future.

In this project we meet the interest of a software development company (Safelayer Secure Communications) in adapting one of their products (TrustedX) to a Software as a Service model (TrustedX as a Service).

To reach this objective we deployed a hybrid cloud based on an Infrastructure as a Service network. As a private cloud, we choose Eucalyptus, and as a public cloud we choose Amazon EC2, because they work analogously. Moreover, we have implemented a web portal to administrate the whole cloud with a simple interface.

From the service point of view, on this project we offer a new way to offer the TrustedX service over a cloud computing environment. The key benefits with this solution are that an administrator does not have to manage a physical server to deploy the service and can access the service within a range of minutes. So the administrator experiences an improving of the system time response and he only uses the resources that he really needs at every moment.

But cloud computing still has some issues that must to be solved. The Cloud Security Alliance [37] points to these main threads: trust issues (lack of provider transparency, governance impacts, risk management, compliance), data privacy issues (leakage, loss or storage in unfriendly geography), insecure cloud software or malicious use of cloud services.

Hybrid cloud solutions solve some of these threats about cloud computing. These threats are related to the distrust on where a key service is running in a cloud computing environment. For companies that cannot trust a public cloud like Amazon EC2, we propose a private cloud solution.

As a personal conclusion, I think that the current state of the available solutions for Infrastructure as a Service are still immature, as pointed out by Gartner's hype cycle report for 2010. During the development of this project I have changed the Eucalyptus version due to a major upgrade of the Eucalyptus project (from version 1.6.2 to 2.0.0). Also during this period, OpenNebula announced an update that increased its compatibility with the Amazon EC2 interface. Even during this time, RackSpace and NASA have developed a new open source solution for IaaS called OpenStack [38], which may have a very good future. Also IaaS public networks evolve. For example, Amazon develops upgrades to its network every one or two months.

As a SOA interface for an IaaS network, I think that Open Cloud Computing Interface (OCCI) will not be generally adopted. The fact that only OpenNebula is betting on with this interface is already significant; even OpenStack has opted for Amazon's EC2 interface, which in fact can be considered as a de facto standard as SOA interface for IaaS. Moreover, nowadays we can easily find a wide range of software solutions that are ready to work with the EC2 interface.

From an economical point of view, an IaaS solution can help in cutting costs. This is due to the use of a virtualized environment that helps in the reduction of the number of physical servers. This reduction implies a reduction on the operating expense (OPEX), due to a reduction on energy consumption, and a reduction on the capital expenditures (CAPEX), due to a reduction on new hardware requirements.

Finally, from an environmental point of view, cloud computing solutions can help to meet future energy challenges and reduce CO2 emissions. This will help IT enterprises to grow in a more sustainable way.

7.2. Future lines

With this project we only have taken into account the technological feasibility, leaving the economic feasibility for future revisions.

Also, we have to keep improving the interaction on the web portal for administrator user and the response of the network if a new virtual machine fails on the deploy operation.

So far, it is expected that the initial cost of an IaaS solution can be higher than a traditional network. This is due to the need of time for the complexity on the configuration, but it is lowered by the reduction of the hardware requirements. Instead, it is clear that the operating expense should be lower than a traditional network, so this type of project will be profitable in a middle-long term.

The reduction on the hardware resources implies also a reduction of software licenses, so it also implies that these licenses will be easier to be amortized.

Moreover we also have to keep looking into the IaaS solutions to evaluate the future changes of their technological solutions in order to find the best choice for us.

BIBLIOGRAPHY

- [1] SETI@home. "Example of grid computing network", <http://setiathome.ssl.berkeley.edu/>
- [2] National Institute of Standards and Technology. "The NIST Definition of Cloud Computing", <http://www.nist.gov/itl/cloud/upload/cloud-def-v15.pdf>
- [3] Gmail. "Example of Software as a Service (SaaS)", <http://www.gmail.com>
- [4] Google App Engine. "Example of Platform as a Service (PaaS)", <http://code.google.com/intl/es-ES/appengine/>
- [5] Amazon Elastic Cloud Computing (Amazon EC2). "Example of Infrastructure as a service (IaaS)", <http://aws.amazon.com/ec2/>
- [6] Gartner Inc. "Gartner's 2010 Hype Cycle Special Report Evaluates Maturity of 8001,800 Technologies", [1447613http://www.gartner.com/it/page.jsp?id=1447613](http://www.gartner.com/it/page.jsp?id=1447613)
- [7] Safelayer Secure Communications. "Company focus on technology based in PKI", <http://www.safelayer.com/en/about-safelayer/the-company>
- [8] TrustedX. "Web services platform for security and trust mechanisms", <http://www.safelayer.com/en/solutions/products-and-services/trustedx>
- [9] David Herranz, Dell Computers. "La virtualización para las pymes", http://muycomputerpro.com/Expertos/La-virtualizacion-para-las-pymes/_wE9ERk2XxDafXMUKhIMERBa3ni-MO6KAhIYY9hn8cwbuCD4itWWp5z2BlmiC2hO8
- [10] VMWare. "Full virtualization systems", <http://www.vmware.com/es/>
- [11] Xen. "Para-virtualization systems", <http://www.xen.org/>
- [12] KVM. "Hardware assist virtualization systems", http://www.linux-kvm.org/page/Main_Page
- [13] Nagios. "Network monitoring management", <http://www.nagios.org/>
- [14] Ganglia. "Network monitoring management", <http://ganglia.sourceforge.net/>
- [15] Apache Tomcat. "Web server with implements Java Servlet and JavaServer Pages technologies", <http://tomcat.apache.org/>
- [16] Spring Framework. "Java Framework that implements the MVC architecture", <http://www.springsource.org/>
- [17] Amazon.com, Inc. "Amazon.com enterprise", <http://www.amazon.com/>

- [18] Amazon S3. "Amazon Simple Storage Service", <http://aws.amazon.com/s3/>
- [19] EC2 API. "API for interact with the EC2 cloud", <http://docs.amazonwebservices.com/AWSEC2/2009-04-04/DeveloperGuide/>
- [20] Eucalyptus. "Open Source software for Infrastructure as a Service network", <http://open.eucalyptus.com>
- [21] Euca2ools. "Command line tools for Linux for administrative purpose in Eucalyptus", <http://open.eucalyptus.com/wiki/Euca2oolsGuide>
- [22] OpenNebula. "Open Source software for Infrastructure as a Service network", <http://www.opennebula.org/>
- [23] OCCI. "Open Cloud Computing Interface, Open standard to interact with a cloud", <http://www.occ-wg.org/>
- [24] David Chernicoff. "Does Open Source Matter in the Cloud?", <http://www.zdnet.com/blog/datacenter/does-open-source-matter-in-the-cloud/372>
- [25] Amazon DevPay. "Business model for enterprises to work with Amazon EC2 and S3", <http://aws.amazon.com/devpay>
- [26] CentOS. "Linux distribution based on RHEL", <http://www.centos.org>
- [27] Red Hat Inc. "Linux distributor of Red Hat Enterprise Linux (RHEL)", <http://www.redhat.com/>
- [28] GParted liveCD. "Graphical disk partition editor", <http://gparted.sourceforge.net/livecd.php>
- [29] Hibernate. "Open source Java persistence framework project", <http://www.hibernate.org/>
- [30] Apache HTTP server. "Open-source HTTP server", <http://httpd.apache.org/>
- [31] Mod_jk module. "Connector between Apache HTTP server and Apache Tomcat", <http://tomcat.apache.org/connectors-doc/>
- [32] MySQL server. "Open source of a relational database management system", <http://www.mysql.com/>
- [33] NFS server. "Network file system protocol", <http://nfs.sourceforge.net/>

- [34] Madri+d. “green IT: tecnologías para la eficiencia energética en los sistemas_TI”, [http://www.madrimasd.org/informacionidi/biblioteca/publicacion/doc/VT/VT19_green IT tecnologias eficiencia energetica sistemas TI.pdf](http://www.madrimasd.org/informacionidi/biblioteca/publicacion/doc/VT/VT19_green_IT_tecnologias_eficiencia_energetica_sistemas_TI.pdf)
- [35] International Data Corporation (IDC). “Analyst firm”, <http://www.idc.com/>
- [36] Energy Star. “Program of the U.S. Environmental Protection Agency”, <http://www.energystar.gov>
- [37] CSA. “Cloud Security Alliance”, <http://www.cloudsecurityalliance.org/>
- [38] OpenStack. “Open Source software for Infrastructure as a Service network”, <http://openstack.org/>

ANNEXES

A. Acronyms

| | |
|-----------------|---------------------------------------|
| ASF | Apache Software Foundation |
| API | Application Programming Interface |
| CC | Cluster Controller |
| CD | Compact Disk |
| CLC | Cluster Controller |
| CO ₂ | Carbon Dioxide |
| CPU | Central Processing Unit |
| DHCP | Dynamic Host Configuration Protocol |
| EBS | Elastic Block Storage |
| EC ₂ | Elastic Compute Cloud |
| EKI | Eucalyptus Kernel Image |
| EMI | Eucalyptus Machine Image |
| ERI | Eucalyptus Ramdisk Image |
| FTP | File Transfer Protocol |
| HTTP | HyperText Transfer Protocol |
| IaaS | Infrastructure as a Service |
| IDC | International Data Corporation |
| INFN | Istituto Nazionale di Fisica Nucleare |
| IP | Internet Protocol |
| IT | Information Technology |
| JSP | JavaServer Pages |
| KVM | Kernel-based Virtual Machine |

| | |
|-------|--|
| MAC | Media Access Control |
| MVC | Model-View-Controller |
| MySQL | My Structured Query Language |
| NAT | Network Address Translation |
| NC | Node Controller |
| NFS | Network File System |
| NIC | Network Interface Controller |
| NIST | National Institute of Standards and Technology |
| NRPE | Nagios Remote Plugin Executor |
| OCCI | Open Cloud Computing Interface |
| OS | Operating System |
| PaaS | Platform as a Service |
| PHP | PHP Hypertext Preprocessor |
| RAM | Random Access Memory |
| REST | Representational State Transfer |
| S3 | Simple Storage Service |
| SaaS | Software as a Service |
| SC | Storage Controller |
| SMTP | Simple Mail Transfer Protocol |
| SOA | Service Oriented Architecture |
| SSH | Secure Shell |
| SSL | Secure Sockets Layer |
| VLAN | Virtual Local Area Network |
| WS3 | Walrus Simple Storage Service |
| XML | Extensible Markup Language |

B. Hybridfox

Hybridfox is based on Elasticfox. Elasticfox is an open source Firefox add-on developed by Amazon that is restricted only to the EC2 environment. Instead, Hybridfox is able to work with Amazon EC2 and Eucalyptus.

As show on the next figure (**Fig B.1**) the only two parameters to configure Hybridfox are the CLC endpoint and the user credentials.

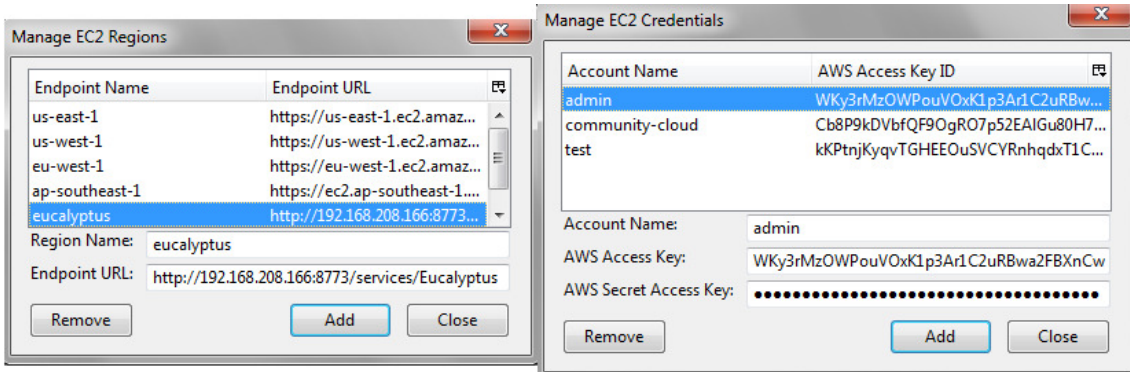


Fig. B.1 Hybridfox Configuration

On the next figure (**Fig. B.2**) we can see the images and the attributes of these images that are registered and ready to run on the Network.

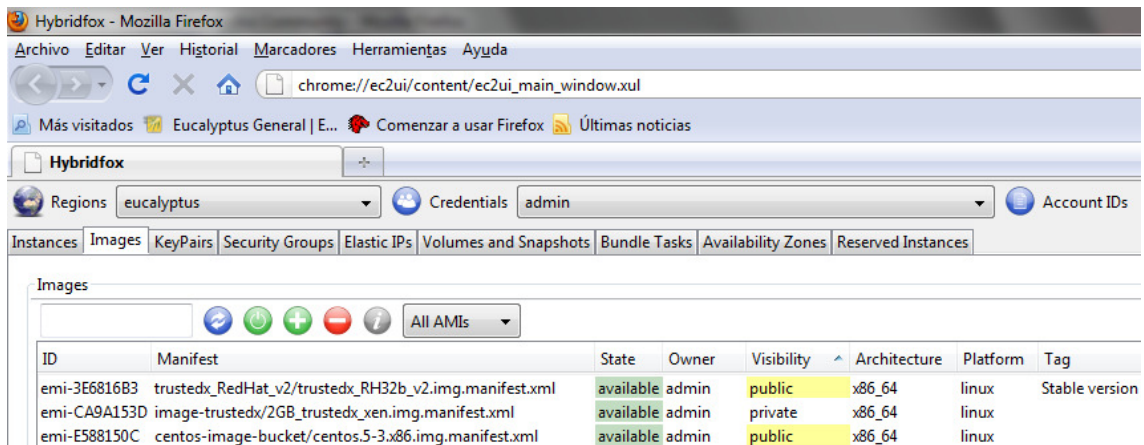


Fig. B.2 Image list with Hybridfox

On the instances tab (**Fig. B.3**) we can see the whole list of instances that are running on the network with its attributes. With an eucalyptus administrator user, we are be able to see the instances running of all users, while as a normal eucalyptus user, we only can see our own instances.

The screenshot shows the Hybridfox web interface in a Mozilla Firefox browser. The browser address bar shows the URL `chrome://ec2ui/content/ec2ui_main_window.xul`. The interface includes a navigation menu with tabs for 'Instances', 'Images', 'KeyPairs', 'Security Groups', 'Elastic IPs', 'Volumes and Snapshots', 'Bundle Tasks', 'Availability Zones', and 'Reserved Instances'. The 'Instances' tab is active, displaying a table of instances.

| Reservation ID | Owner | Instance ID | AMI | AKI | ARI | State | Public DNS | Private DNS | ... | Groups | Reason | Idx | Type | Local Launch Time | Availability Zone |
|----------------|-------|-------------|--------------|--------------|--------------|---------|---------------|--------------|-----|------------|-----------|-----|-----------|---------------------|-------------------|
| r-404E07CB | admin | i-486108BC | emi-3E6816B3 | eki-419B1630 | eri-A4871775 | running | 192.168.208.2 | 10.35.25.130 | ... | group-test | NORMAL... | 0 | c1.medium | 2010-09-16 11:25:29 | Cluster_A |
| r-4CF4092F | admin | i-31B9068C | emi-3E6816B3 | eki-419B1630 | eri-A4871775 | running | 192.168.208.6 | 10.35.24.132 | ... | group-test | NORMAL... | 0 | c1.medium | 2010-09-16 12:09:23 | Cluster_B |
| r-3EC608A3 | admin | i-2EE6057C | emi-3E6816B3 | eki-419B1630 | eri-A4871775 | running | 192.168.208.7 | 10.35.24.131 | ... | group-test | NORMAL... | 0 | c1.medium | 2010-09-16 12:20:23 | Cluster_B |

Fig. B.3 Instance list with Hybridfox

On the Elastic IPs tab we can see the complete list of public IPs that can be associated to an instance. Also we can see the association between these IPs and instances and who has done these associations (**Fig. B.4**).

Moreover, in this figure we can see that the IP 192.168.208.9 is not assigned to any instance, but it is reserved to the admin user, for this reason, Eucalyptus is not allowed to assign this IP to an instance, only the admin user can do it.

The screenshot shows the Hybridfox web interface with the 'Elastic IPs' tab selected. A dialog box titled 'Associate Address with Instance' is open, asking 'Which Instance would you like to associate 192.168.208.9 with?'. The dialog box lists three instance IDs: i-486108BC, i-31B9068C, and i-2EE6057C. The 'Acceptar' button is highlighted.

| Address | Instance ID |
|---------------|-------------------------|
| 192.168.208.2 | i-486108BC (eucalyptus) |
| 192.168.208.3 | nobody |
| 192.168.208.4 | nobody |
| 192.168.208.5 | nobody |
| 192.168.208.6 | i-31B9068C (eucalyptus) |
| 192.168.208.7 | i-2EE6057C (eucalyptus) |
| 192.168.208.8 | nobody |
| 192.168.208.9 | available (admin) |

Fig. B.4 Public IP list with Hybridfox

On the next figure (**Fig B.5**) we can see that we are assigned the IP 192.168.208.9 to the instance i-486108BC. With this change, the old IP 192.168.208.2 is released for a future use.

| Reservation ID | Owner | Instance ID | AMI | AKI | ARI | State | Public DNS | Private DNS | ... | Groups | Reason | Idx | Type | Local Launch Time | Availability Zone |
|----------------|-------|-------------|--------------|--------------|--------------|---------|---------------|--------------|-----|------------|-----------|-----|-----------|---------------------|-------------------|
| r-404E07CB | admin | i-486108BC | emi-3E6816B3 | eki-419B1630 | eri-A4871775 | running | 192.168.208.9 | 10.35.25.130 | ... | group-test | NORMAL... | 0 | c1.medium | 2010-09-16 11:25:29 | Cluster_A |
| r-4CF4092F | admin | i-3189068C | emi-3E6816B3 | eki-419B1630 | eri-A4871775 | running | 192.168.208.6 | 10.35.24.132 | ... | group-test | NORMAL... | 0 | c1.medium | 2010-09-16 12:09:23 | Cluster_B |
| r-3EC608A3 | admin | i-2EE6057C | emi-3E6816B3 | eki-419B1630 | eri-A4871775 | running | 192.168.208.7 | 10.35.24.131 | ... | group-test | NORMAL... | 0 | c1.medium | 2010-09-16 12:20:23 | Cluster_B |

Fig. B.5 Public IP change with Hybridfox

On this figure (Fig B.6) we see different security groups defined for each user. An user can have more than one security group, each group defines different firewall rules for each port. Different instances can belong to the same security group.

| Owner | Name | Description |
|-------|-----------------|-------------------|
| test | default | default group |
| admin | default | default group |
| admin | group-test | dummy-description |
| test | group-test_user | dummy description |

| Protocol | From Port/ICMP Type | To Port/ICMP Code | Source User:Group |
|----------|---------------------|-------------------|-------------------|
| icmp | -1 | -1 | admin:group-test |
| tcp | 0 | 65535 | admin:group-test |
| udp | 0 | 65535 | admin:group-test |

Fig. B.6 Security groups with Hybridfox

C. CloudBerry S3 Explorer

CloudBerry Explorer is a program compatible with the Amazon S3 interface, making easy to manage files in Walrus and Amazon S3 storage.

CloudBerry provide an user interface to Walrus and Amazon S3 accounts, files, and buckets, CloudBerry lets to manage the files on the cloud like a FTP program.

As show on the next figure (**Fig C.1**) the only two parameters for configure CloudBerry are the Walrus endpoint and the user credentials.

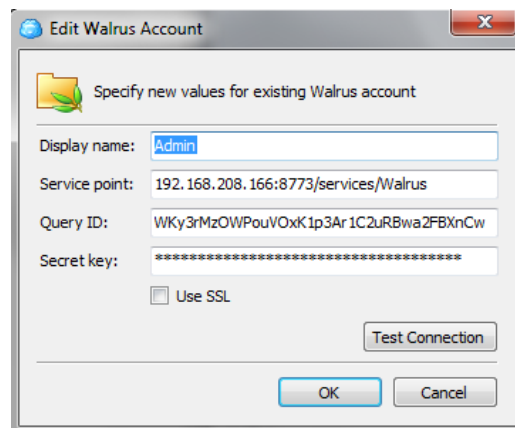


Fig. C.1 CloudBerry configuration

We can see on the next figure (**Fig. C.2**) all buckets created by the admin user. These buckets contain the images of the network.

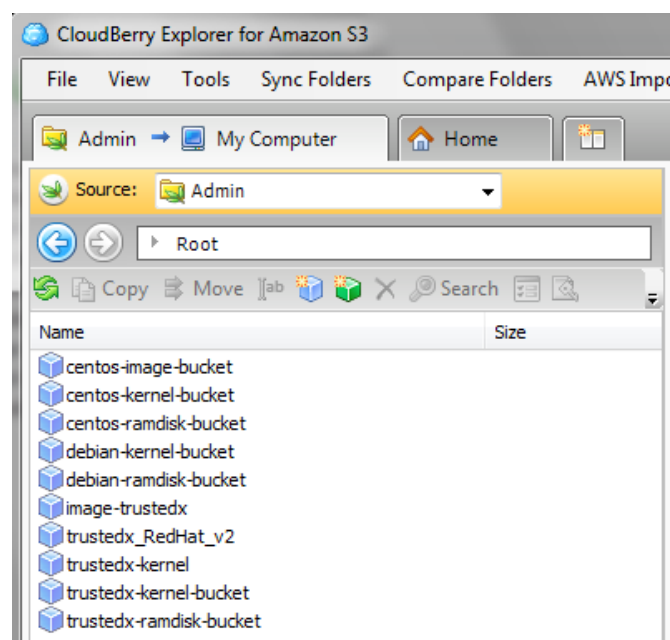


Fig. C.2 Bucket list with CloudBerry

But the S3 interface is not only use for uploading images, it is use also for upload files. In the next figure (**Fig. C.3**), we see a new bucket created by the admin user and called "test-bucket" that contains a PDF file. In this file we are changed the default permissions to make this file readable by any user.

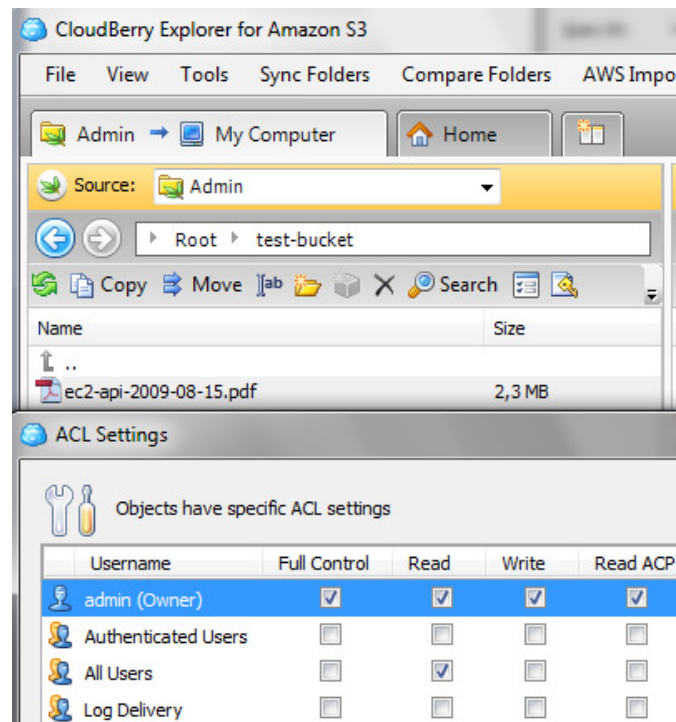


Fig. C.3 Bucket management with CloudBerry

Now with a browser, we are able to download this file, thanks to the public permissions (Fig. C.4).

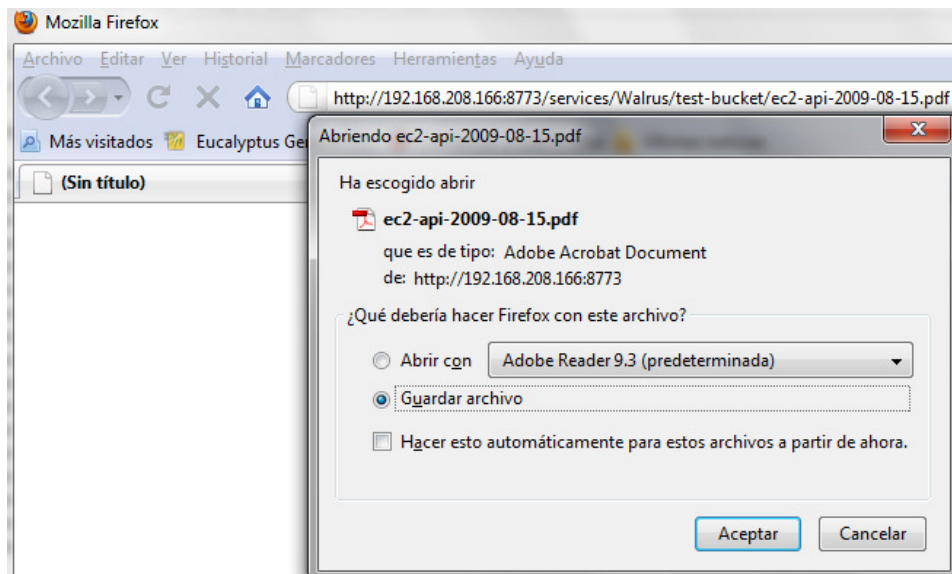


Fig. C.4 File with public permissions