POLITECNICO DI TORINO

Department of Electronics and Telecommunications

Master of Science in Telecommunications Engineering

Master Thesis

SURA PRIVATE CLOUD: IMPLEMENTATION OF A PRIVATE INFRASTRUCTURE AS A SERVICE CLOUD.

Supervisor:

Roberto Garello

Mauricio Palacio

Candidate:

Andrea Trujillo López

Academic Year 2018/2019

Dedicate to my family,
to my Friends,
to my teachers.

Acknowledgements

First of all, I would like to thank my thesis advisor, Professor Roberto Garello, for his guidance and support throughout its development. Also to the teacher Andrea Carena for the support provided and for all the administrative management he did to finish my career. To both Thanks for the extensive knowledge you shared during the courses I had the opportunity to attend.

I also thank my advisor in Suramericana, Mauricio Palacio, for giving me the opportunity to participate in this great project, for everything taught and for his interest in my professional development.

I also want to express my gratitude to all the friends who gave me their support during all these years, thanks to them I managed to live this experience and not give up in difficult times.

Finally, I must express my deep gratitude to my family for the collaboration, unconditional support and for their encouragement words throughout my years of study.

This achievement would not have been possible without all these people, thank you very much to each of them.

Table of Contents

Acknowledgements

List of Acronyms 1

List of Figures 3

List of Tables 4

Abstract 5

1 Introduction **6**

1.1 Thesis outline 7

2 Objectives **8**

2.1 General Objective 8

2.2 Specific Objectives 8

3 Theorical Framework **9**

3.1 IT Service 9

3.2 IT Infrastructure Service 9

3.3 Virtualization 9

3.4 Cloud Computing 10

3.4.1 Key Features of Cloud Computing 10

3.4.1.1 Broad Network Access 10

3.4.1.2 On Demand Self-Service 10

3.4.1.3 Rapid Elasticity 10

3.4.1.4 Multitenant and resource Pooling 11

3.4.1.5 Measured service 11

3.4.2 Service Models 11

3.4.2.1 Infrastructure as a Service (IaaS) 11

3.4.2.2 Platform as a Service (PaaS) 12

3.4.2.3 Software as a Service (SaaS) 12

3.4.3 Deployment models ……..12

3.4.3.1 Private Cloud ……..12

3.4.3.2 Public Cloud ……13

3.4.3.3 Hybrid Cloud ……13

4 Methodology 14

5 Results and Analysis 16

5.1 Understanding the organization's technological ecosystem …… 16

5.1.1 Perspective of technological infrastructure ……...17

5.1.2 Application Perspective ……19

5.1.3 Perspective of the business model ……19

5.2 Why does the Organization require a Private IaaS Cloud? ……..20

5.3 Tool analysis of Private Cloud type Iaas ……..21

5.3.1 OpenStack …… 21

5.3.2 CloudStack …… 21

5.3.3 Eucalyptus ……21

5.3.4 Open Nebula ……21

5.4 Evaluation matrix ……22

5.4.1 General Comparison ……22

5.4.2 Functional comparison ……..24

6 Brief description of OpenStack 27

6.1 General Architecture ……...31

6.1.1 Description of the OpenStack Networking Architecture (Neutron) three nodes ……... 31

6.1.2 Description of the architecture Openstack Networking (Legacy Networking) two nodes ……..33

6.2 Considerations about examples of architectures ……34

6.3 Cloud scalability ……34

7 Prototype SURACLOUD 36

7.1 Operation platform SURACLOUD ……36

7.2 Case study: delivery of an infrastructure service ……42

7.3 Proof of Concept ……49

7.3.1 Proof ……49

7.3.2 Prototype ……50

7.3.2.1 Hardware Specifications ……50

7.3.2.2 Topology ……52

7.3.3 OpenStack installations ……52

7.3.3.1 Configuration of the identity service (Keystone) in the controller node 56

7.3.3.2 Configuration of the image service (Glance) in the controller node … 57

7.3.3.3 Configuration of the compute service on the controller node ……58

7.3.3.4 Configuration of the compute service on the compute node …… 58

7.3.3.5 Configuration of the network service (legacy networking) on the controller node ……59

7.3.3.6 Configuration of the network service (legacy networking) on the compute nodes ……59

7.3.3.7 Configuration of the Horizon Service on the controller node ……60

7.4 Findings and recommendations ……60

7.4.1 Design of Computer Resources (CPU and Memory) ……60

7.4.2 Network topology design ……61

7.4.3 Design of High Availability Schemes (HA) ……62

7.4.4 General ……63

7.4.5 Installation ……64

8 Conclusions 66

10 Bibliography 68

Glosary 71

List of Acronyms

AIX	Advanced Interactive eXecutive
AMD	Advanced Micro Devices
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
AWS	Amazon Web Services
BIOS	Basic Input/Output System
CISC	Complex Instruction Set Computer
CPU	Central Processing Unit
DHCP	Discover Host Configuration Protocol
DSA	Distributed Systems Architecture Research Group
ERP	Enterprise Resourse Planning
FTP	File Transfer Protocol
HP	Hewlett-Packard
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
IBM	International Business Machines
IP	Internet Protocol
IT	Information Technology
KVM	Kernel-based Virtual Machine
LDAP	Lightweight Directory Access Protocol
LPAR	Logical Partition
NAT	Network Address Translation
NIST	National lnstitute of Standards and Technology
NTP	Network Time Protocol
OVS	Open VSwitch
PaaS	Plataform as a Service
QoS	Quality of Service
RAM	Random Access Memory
RISC	Reduces Instruction Ser Computer
SaaS	Software as a Service
SAN	Storage Area Network
SLA	Service Level Agreement
SOA	Service Oriented Architecture
SSH	Secure SHell
TB	Terabyte
VCPU	Virtual CPU
VLAN	Virtual Local Area Network

VNC	Virtual Network Computing
VXLAN	Virtual eXtensible LAN

List of Figures

6.1 OpenStack description 27
6.2 OpenStack Neutron 32
6.3 OpenStack Legacy Networking 33
7.1 Platform Access SURACLOUD 36
7.2 APIs exposed 37
7.3 Overview 37
7.4 Instances 38
7.5 Images 39
7.6 Key pairs 39
7.7 Network topologies 40
7.8 Security Groups 40
7.9 Projects 41
7.10 Users 41
7.11 Delivery time of a current service 42
7.12 Instance menu 43
7.13 Launch instance menu 44
7.14 Boot Source 45
7.15 Available Images 46
7.16 Available Flavors 47
7.17 Available Networks 48
7.18 Security Groups 49
7.19 Key Pair 50
7.20 Instance Creation 51
7.21 Creation time of the instance 51
7.22 Network Topology 55
7.23 Installation architecture Prototype SURACLOUD 56
7.24 Network configuration on the controller node and compute nodes 57

List of Tables

4.1 Specific objectives and methodological 15
5.1 General Comparison 22
5.2 Functional Comparison 24
6.1 Descripción of main services of OpenStack 28
6.2 OpenStack Versions 29

Abstract

Cloud Computing represents a service technology model of the Information (IT) that has revolutionized the way they have been delivered to users, which has generated great changes in the technological scene and therefore also in the different sectors where it is used.

The final report shows the reasons and the details of the implementation of a prototype of a Private Cloud in the Infrastructure as a Service (IaaS) model in an organization.

To do this, the different Open Source alternatives were explored in the concerning of the available tools for the deployment; Subsequently, an analysis was carried out that led to the selection of the most appropriate tool to be used in the prototype, based on general selection criteria, but also to be coherent with respect to the needs of the organization and its technological ecosystem.

Finally, the execution of the prototype was carried out, supported by the selected tool in a very enriching exercise that made it possible to demonstrate first hand the details behind the implementation of a Private Cloud and the benefits that this technology can offer in favor of the organization. Also, this exercise allowed to exhibit the most relevant aspects to design and dimension the efforts that require a possible formulation of a project of adoption of a Private Cloud in the productive environment. So, in this writing are represented the details about the prototype, the matrix of selection of the most appropriate tool and the description of it. And shows the results found in a practical case of delivery of infrastructure service, and the design recommendations and adoption references that emerged through the experimentation.

Keywords: Cloud Computing, Private Cloud, Infrastructure as a Service, IaaS, Virtualization, Open Source, Platforms, Software, Technologies, Technology management.

Chapter 1

Introduction

The biggest problem facing IT units in many enterprises and organizations is that the IT infrastructure services they offer are not being competitive with the supply of Public Clouds (AWS, Windows Azure, Rackspace, SoftLayer , among others) which are why they are becoming expendable for the business areas that are increasingly contracting these services of Public Clouds to support the applications and processes of your business, instead of using the services that your Technology units can provide them. From the above, it is evident that the management of IT operations in Suramericana S.A is no stranger to this circumstance and hopes to be able to respond to the challenge that this demands.

This problem stems from the fact that the delivery of internal IT infrastructure services from companies is too slow and costly compared to Public Clouds, and even so, when internal capabilities are finally delivered, such services often lack the required levels of availability. It has been identified that the internal infrastructure of the IT units of these companies is too rigid and fragile, as it turns out that the fluctuation in the behavior of business transactions biunivvally leads to two situations: the first one when you have infrastructure capacities very adjusted to the demand, in which case it happens that investors in business transactions overflow Capacities of IT infrastructure, affecting the availability of business services and just when it is most need it; and the second situation, where infrastructure capabilities are too large to endure such peaks in transactional business demand, but then such IT capabilities are substantially wasted for the vast majority of the time, resulting in large overcharges.

It has been observed that Cloud Computing represents an integral solution for the removal of such impediments and is shown as a very important tool to improve the delivery of IT infrastructure services. It is considered that if it is used properly and if it comes to play in favor, Cloud Computing, and especially a Private Cloud initiative, constitutes an opportunity to make the internal departments of the technology areas competitive in a way that The business prefers to consume internal IT infrastructure services rather than the services offered by Public Clouds.

In the present work pretends to support the adoption of a Private Cloud for Suramericana S.A with a model IaaS type of deployment, producing a prototype that allows direct interaction with an implementation of this type and observe all the benefits that would be achieved. Likewise, it is expected to participate actively in the technical support required for the formulation and possible implementation of the project that will implement the Private Cloud.

1.1 Outline

The outline of this thesis is as follow:

In chapter 2 only the different objectives that are given with the proposed project to the company are mentioned, in chapter 3 the theoretical framework for a better understanding of the topic to be worked is explained, it is explained what Cloud Computing is, its relevant characteristics and its service models (IaaS, PaaS, SaaS).

In chapter 4 explains the methodologies used to achieve each of the objectives set forth in chapter 2.

Then, in chapter 5, which is the results and analysis, explains the need presented by the company from the business side and from the technology side, it is understood the technological environment of the same in order to start and have a good development of the Project, it is understood that the need arises and why the need to have a private cloud type IAAS. In the same chapter you can see the research carried out on the cloud computing tools with the greatest acceptance worldwide, making a general and functional comparison.

In chapter 6 talks about what OpenStack is, its features, components and versions, as well as the most common architectures and calculations that must be taken into account to have an idea of the system's scalability.

In chapter 7 shows the assembled prototype and explains the options of the platform in detail, a proof of concept is also carried out and there is evidence of the advantages that this project brings. The first scenario, which served as a test, was developed on a laptop by installing virtual machines in the same, and then a prototype was created using some servers that were available. The scope of the project, the proposal and the approval of the company are also disclosed.

In the later chapters, the conclusions that this project gave us, the future works to be carried out, the consulted bibliographical sources and a glossary that is of great help to understand the work done.

Chapter 2

Objectives

2.1 General Objective

Implementing a prototype Infrastructure as a Service (IaaS) in Suramericana S.A for the automation of the provisioning of some IT infrastructure services.

2.2 Specific Objectives:

- Understand the technological ecosystem of Suramericana S.A, from the perspectives of its infrastructure, its applications and the business model, understanding its current condition and the challenges they face.

- Explore approved solutions available for infrastructure service provisioning models that meet the desired characteristics.

- Analyze a IaaS Private Cloud tool appropriate for use in the prototype and according to the technological ecosystem of Suramericana S.A.

- Implement a proof of concept (prototype) with a practical case of an infrastructure service delivery process, which allows practical experimentation to the capabilities that a Private Cloud can deliver, especially pretends a reduction of the provisioning time of a Virtual Machine with installed operating system and with the respective configuration of connectivity to the network.

- Provide technical recommendations for conception of the IT infrastructure services model with the application of Private Cloud technologies for productive use.

Chapter 3

Theoretical Framework

As a first aspect, it is important to properly understand wich is the product that will be delivered, in this case it is IT infrastructure services, but the appropriation of it arises from a broader concept that is IT service; the definitions of both are presented below.

3.1 IT service

"Refers to the application of business and technical expertise to enable organizations in the creation, management and optimization of or access to information and business processes." [11]

3.2 IT Infrastructure Service

"The IT Services market can be classified according to the skills used to deliver the service (Design, Construction, Execution). There are also different service categories: Business Process Services, Application Services and Infrastructure Services" [11]. For this purpose, it is focused on the latters, that is, to the infrastructure services that correspond to the delivery of computational resources on the technological platform, so that the applications can be supported and transitively the business processes.

3.3 Virtualization

By virtue of the good understanding of how cloud computing is structured, it is convenient to introduce the term "virtualization"

NIST defines "Virtualization is the simulation of the software and / or hardware on which other software runs." [22]. Another very successful definition is that given by Red Hat "Virtualization allows multiple operating systems to run simultaneously on a single computer, breaking the relationship between the hardware and a single operating system."[26]

While virtualization is not a recent invention, it is the very essence of cloud computing, which is used to separate a single physical machine into multiple virtual machines in an efficient manner. Virtualization is cost effective, allowing the execution of virtual machines on the same physical equipment, being able to rent part of the resources of physical servers and other interested parties.

Among the most typical virtualizations are: storage virtualization, server virtualization, operating system virtualization and computer application virtualization.

3.4 Cloud Computing

It is also important to present the concept of Cloud, its essential characteristics, the service models and the deployment models that conform it.

Gartner defines it as "massively scalable IT capabilities that are delivered 'as a service' to users using internet technology." [19]

However, one of the most widely accepted definitions and one of the first to deliver a global understanding of the concept of Cloud are those delivered by NIST who define Cloud Computing as: "A technology model that enables ubiquitous, convenient and on-demand access via the Internet to a shared set of configurable computing resources (such as networks, servers, storage equipment, applications and services), which can be quickly Provisioned and delivered with minimal management effort and minimal interaction with the service provider. " [13]

3.4.1 Key Features of Cloud Computing

3.4.1.1 Broad Network Access

Cloud services architectures are designed in such a way that processing and data are held on servers that are accessed via a network, so there are not components in the users' terminals. It promotes the use of heterogeneous technologies of light clients (mobile phones, tablets, laptops), this feature is closely related to the mobility and ubiquity that have enabled the emergence of a movement known as the internet of things.

3.4.1.2 On Demand Self-Service

Is related to the fact that services are offered in an ephemeral way, that is, users can use Cloud services automatically and in record time when they need them through a portal or an application, abstracting the underlying technical complexity and without Need to communicate with the service provider.

3.4.1.3 Rapid Elasticity

Services can scale rapidly on demand through the addition or elimination of computational resources. That is to say that the capacities can increase or diminish quickly, in some cases automatically, as the demand in the transactions of the users behaves. Therefore, the

capabilities will give the feeling of being unlimited, although beyond a certain level can have a cost, and can be purchased in any quantity or time.

3.4.1.4 Multitenant and resource Pooling

Resources are grouped into pools in the multi-owner model, that is to say, computing capacities are shared by all owners (different companies or different business areas within a large company) This characteristic is what makes it possible to take full advantage of these capacities and therefore be cost efficient, well the owners that is not using its resources, another owner who requires them can make use of them. This means that each owner can dispose of the resources share by demand but privacy and security must be guaranteed.

3.4.1.5 Measured Service

The cloud infrastructure systems automatically control and optimize the resources that compose it, enhancing the measurement capacity to an abstraction level appropriate to the type of service. The use of resources can be monitored through metrics that allow different payment models. In this way the end user pays strictly for what they consume.

3.4.2 Service Models

The levels of service delivered to the end user include the following models offered through the cloud:

3.4.2.1 Infrastructure as a Service (IaaS)

A service model in which the client is offered computing capacity (processing, storage, networks and other fundamental computing resources) or a pool of resources where it is able to deploy and execute the software in an ephemeral way, that is to say can be create, delete or modify it at any time and it will be available through App. Gartner defines IaaS as: "is a standardized, highly automated offering, where compute resources, complemented by storage and networking capabilities are owned and hosted by a service provider and offered to customers on-demand." [10].

Examples of IaaS are Amazon Web services, OpenStack, Windows Azure.

3.4.2.2 Platform as a Service (PaaS)

Platform as a service: Service model in which the client is offered everything necessary for the deployment of applications (software). The easiest way to understand it is to associate the middleware and runtime with web servers, databases, application servers, java virtual machines, containerization of applications, FTPs, user directories (LDAP) and integration platform among others with a service focus that enables quickly create Solutions for different users in a variety of execution environments. The NIST defines PaaS "The capability offered to the user to deploy applications in cloud infrastructure using programming languages, libraries, services and tools supported by the provider." [13]. Some examples of PaaS are: Cloud Foundry, Openshift, Heroku, Amazon EC2, App Engine, Rackspace Cloud Servers, BlueMix

3.4.2.3 Software as a Service (SaaS)

They are application functionalities offered to end users who use the services running from a cloud infrastructure. Applications are accessed from different type's devices from the client-side such as light clients, in a web browser or a program interface (eliminates the need to install or run applications on individual computers). Gartner, defines SaaS as: "software that is owned, delivered, and managed by a provider." [25].

When the software is contracted as a service, there is no control or Responsibilities on the infrastructure or underlying platform that supports such a system.
Some examples of SaaS are: SalesForce, Office 365 and google services - Gmail, Drive, YouTube.

3.4.3 Deployment models

3.4.3.1 Private Cloud

It is essentially the extension of a company's traditional datacenter that is optimized to provide IT resources (applications, computing, storage and networks). The Private Cloud is defined as "The cloud infrastructure is prepared for the exclusive use of a single organization that comprising multiple consumers (for example, business units). It may be owned, managed and operated by the organization, a third part, or any combination of them, and may exist on or off the installations. " [13]. It is possible to have a Private Cloud

in a datacenter of a third party provided that the services are only and exclusively used by users belonging to the organization.

3.4.3.2 Public Cloud

It is a set of shared resources that serves many organizations and is provided as a service through the internet. The Public Cloud is defined as "Cloud infrastructure is delivered for open use by the general public. It could possibly be managed, operated, and owned by a business, academic institution, governmental organization, or some combination of these. The infrastructure of this type of Cloud is always located in the facilities of the one who provides the cloud services." [13]

3.4.3.3 Hybrid Cloud

It is a combination of public and private cloud, it is the perfect solution if, for example, you want to expand the IT infrastructure to include services and applications where you can get the most out of the cloud. The hybrid cloud is defined as "a composition of two or more different cloud infrastructures (private or public) that belong to unique entities, but are united by standardized or proprietary technology that allows data portability and application." [13]. For example, Cloud Bursting to balance the load between clouds.

Chapter 4

Methodology

Specific objectives	Methodological
Understand the technological ecosystem of Suramericana S.A, from the perspectives of its infrastructure, its applications and the business model, understanding which is its condition and the challenges they face.	The corresponding collection of information was carried out on the different technologies that handle infrastructure and its applications. The respective meetings were held with the technology specialist staff of the company to make a classification of technologies according to the particularities of the company.
	We explored the business model of Suramericana S.A, the company's overall strategy and the IT strategy that supports it. Meetings were held with heads of IT planning and government departments.
Explore proven solutions available for infrastructure service provisioning models that meet the desired characteristics.	The evaluation criteria were defined to support the exploration of the available tools, being these a filter that made it possible to focus efforts by limiting the evaluated solutions.
	We searched for bibliographic sources and advice from manufacturers and service providers of IT infrastructure to search for the tools available in the market that meet the needs of the infrastructure services model of Suramericana S.A.

Analyze a IaaS Private Cloud tool appropriate to be used in the prototype and according to the technological ecosystem of Suramericana S.A.	An evaluation matrix was realized, crossing the defined criteria with the different tools proposed, making an evaluation of each tool for each of the criteria.
	The evaluation matrix was analyzed, where was weighted the qualification that allowed the selection of the best tool.
Implement a proof of concept (prototype) with a practical case of an infrastructure service delivery process, which allows submit the practical experimentation of the capabilities that a Private Cloud can deliver, in particular it aims to a reduction of the provisioning time of a Virtual machine with installed operating system and with the respective network connectivity configuration.	Use was made of the different knowledge acquired in the career and installation manuals for the implementation of cloud operating system.
	The applicability of the tool was verified with a practical case of a process of providing infrastructure services, in a way that showed a more efficient management of delivery times. The results obtained were disclosed to the different stakeholders in South American S.A.
Provide technical recommendations for the conception of the IT infrastructure services model with the application of Private Cloud technologies for productive use.	With the knowledge, experience gained and lessons learned in the implementation of the prototype, it was actively participated from the technical perspective in the conception of the solution.
	Technical recommendations were generated for the design of the Private Cloud solution for the productive environment .

Table 4.1. Specific and methodological objectives

Chapter 5

Results and Analysis

Cloud computing burst onto the global technological scene in a disruptive way, representing a major shift in the delivery of technological capabilities in a much more efficient and agile way as a service. This has allowed organizations to reduce IT operational costs, at the same time as they have been able to cut considerably their cycles of delivery of value to the business to respond to the growth of the markets and encouraging the advance in the innovation of the services that are offered, which has resulted a more competitive environment and greater dynamics.

Such dynamism represents opportunities for the business when it is innovative in the services and it achieves to reduce the time to market to get them on time to the market obtaining competitive advantages. But at the same time, such dynamics represent a great requirement for IT units, since they must in turn provide the business with much more agile, cost-efficient and quality IT services. Because in the digital age business services are generally supported by IT services.

Transitively, these IT services need to be implanted on an infrastructure appropriate to their needs, consuming IT infrastructure services that must also be provided in a timely manner to comply the opportunity requirement of business services.

Showing up next is the proposal developed for the adoption of a IaaS Private Cloud in an organization for IT infrastructure services, understanding how is its technological ecosystem, exploring the most appropriate Private Cloud tools options and implementing in a prototype of Private cloud that allows the organization to demonstrate firsthand how this technology would accelerate the delivery of IT infrastructure services, to leverage agility in the delivery of IT services and the consequent reduction of time to market for business services.

5.1 Understanding the organization's technological ecosystem

Suramericana S.A is a holding Company made up of several companies that mainly work in the insurance and social security sector in South America. At the level of corporate organization, it is important to point out that due to the needs derived from its international expansion, at the moment the organizational structure is subject to changes. However, it can be identified that the need is maintained to centralize certain types of operations that produce synergies and efficiencies transversally to the companies that compose the group, among this type of operations the largest are those of IT, punctually those of IT infrastructure and IT governance. In virtue of providing a

clearer picture of the organization's technological ecosystem, it is important to approach the description from three perspectives: the technological infrastructure perpective, the application and the business model, which reflect an integral vision since different points of view.

5.1.1 Perspective of technological infrastructure

Traditionally the technological infrastructure has had in a large percentage installed in own datacenter within the physical facilities of the company. However, because efforts to consolidate servers and redistribute racks to optimize the use of physical space have not been enough to respond to the need for infrastructure growth to support the demand for IT services, it was decided to hire the Datacenter service externally.

In terms of servers, there is a composition that contains RISC processing architectures with IBM Power series machines with AIX operating system and also part with CISC processing architectures with predominance of Red Hat Enterprise Linux and Windows Server operating systems.

The infrastructure is composed of several platforms that have databases, application servers, user directory, mail servers, FTP servers, file servers, print servers, firewalls, monitoring systems, networks, service bus, telephony, management Documentary, among others.

The P-series infrastructure with RISC architecture is intended for database platform and back office operations (the ERP) because it is considered that the workload on these platforms are processing intensive and therefore have a better Performance and retain a better cost benefit implemented on RISC-type processing architectures.

On this platform, virtualisation is physical based on the native system of IBM's Power technology, likewise it obtains the shared use of computational resources that are dynamically assigned to LPARs that representing virtual machine instances.

The remaining platforms are installed on less costly and lighter x86 servers with CISC processor architectures, which It matches with the workload patterns of these platforms, which is less intensive in heavy sequential processing In lightweight concurrent operations, a part of this infrastructure is virtualized over a VMware-based system; However, approximately 50% x86 servers of CISC technology are physical (not virtualized), so there is still an opportunity for optimizing resource use through of virtualization technologies for infrastructure consolidation.

In the observation the organization's physical infrastructure environment that was performed, it was generally found that the greatest waste of resources is focused on the CPU on physical machines x-86 (CISC), where it was found that the vast majority use

Only the 20% of the total CPU capacity allocated to the machine, while in RAM the consumptions generally range between 60% and 85%.

In the storage, the use of SANs predominates, and it has different disk technologies in a multi-layer environment that allows storage stratification. In terms of network virtualization has VLANs that segment by environments, by platform and by business domain the network.

At the level of IT infrastructure service delivery processes, there is a good level of maturity in several of the processes that make up the IT management cycle, they count with controls, and there is high standardization in the most important processes.

However, it is observed that the delivery times are perceived as extensive for the needs of the interested parties, it is noticed that there is a lot of dependence on manual steps that depend on several internal and third parties (the supplier that administers the platform), which Makes it somewhat complicated for users to process and receive the infrastructure services they require. In the same way, the costs of the solutions offered are perceived as high when compared to the alternatives of Public Clouds, this occurs because generally the whole year is charged in advance to the project by a projection of the estimated use of the resources of Infrastructure that will have the solution, and often these projections end up being well above the actual utilization. Likewise, the budgets for the projects of new solutions are fixed based on drivers who consider the costs of the infrastructure including the payment of the resources that are assigned but that are not really being consumed, therefore the projects assume the inefficiencies Derived from underutilization of resources.

Finally, it was possible to show that the infrastructure is somewhat rigid, in the sense that it is not possible to adapt to the demand, and it is understood that the workloads oscillate to an intermediate level and are enclosed to the physical domains. It happens that monthly closures produce processing spikes that lead to high consumptions, which exceed the available capacities of the infrastructure of several critical systems, which is why it is common that the process dilate, leading some delays in delivery Of reports that must be presented for closing, and slowdowns in the transactional systems for the end users by the competence of computational resources, at that moment there are capacities available in other, less critical systems that are located in a different and dedicated infrastructure domain and, therefore, those capacities can not be shifted to cover the need for critical system peaks.

For this reason, it is thought that providing the infrastructure with the characteristics of Cloud, such as rapid self-provisioning on demand, elasticity, payment for use, multitenant, (shared use of Computational resources) can help to solve several of the Aspects that inhibit the improvement of IT infrastructure services.

5.1.2 Application Perspective

Currently the application architectures that are supported are: client server with front-end of the client side and back-end in the database, also have web architectures with front-end and business logic in application server and the persistence in Databases and architecture of self-contained applications based on micro-services where business-by-protocol REST services are exposed from middleware servers to the front-end layer that is developed according to the needs of mobility and user experience.

Within the application reference architecture, there are some non cross-functional logistics services, such as those that handle authentication and authorization, a message exchange system, or the log and event logging system. Also, the integration between applications is done through a service bus, supporting a service-oriented architecture (SOA).

From this application perspective, it is important to note that achievements have been made regarding the adoption of agile methodologies (Specifically SCRUM) And practices of deployment, integration and continuous delivery, which has been achieved significantly reduce the development cycles of solutions to users, However, when it has needs infrastructure provisioning, it has waiting times that globally affect the deployment cycle and delivery time of the solutions.

5.1.3 Perspective of the business model

At a high level, it is possible to say that a model is maintained between coordinated and diversified, even though there are some strategic guidelines and some global services for the whole organization, because they are companies of different nature regarding the services that are offered, each of them has operational independence, as well as some strategic autonomy that allows each company to develop without dependencies of the other companies / areas that make up the group.

The business services are articulated through processes that correspond to the specific needs of each company, but also, there are support processes that are transversal to the companies that allow to generate synergies and operational efficiencies, optimizing the use of some Resources, services and capabilities.

5.2 Why does the Organization require a Private IaaS Cloud?

Initially it will be argued why a Private Cloud is required and later it will be argued why it should be of IaaS type.

It must be a Private Cloud because of the long-term costs to the organization's workload pattern, the confidentiality of the information, and the design for the integration system. In the technological environment the demand fluctuation is intermediate, and the solutions that are implemented last several years in the market. In formal studies realized as Cloudonomics [26]. It is demonstrated that this type of workloads in the long term are much costlier if they are carried to Public Clouds, therefore they recommend implementing them in Private Clouds to achieve an adequate balance of cost-efficiency.

Likewise, the organization is obliged by the regulations of the monitoring and control entities, such as the financial superintendence, to guarantee the confidentiality of the information and to avoid leakage of data or inappropriate alterations of the same within the business processes.

Therefore, the organization prefers to have such information and its applications under its control, and it is not considered appropriate to lose control of them deploying them in Public Clouds that in many cases do not guarantee the security and confidentiality of the Information to the level that is required.

On the other hand, the global architecture of the environment is highly integrated (SOA), which is why regularly the new solutions must consume and offer services, interacting with systems that are implanted locally in the organization's datacenter. In tests performed, carrying some components to public clouds and generating simulated concurrency, it was possible to determine that the latency in the communication induces representative degradation in the QoS (Quality of Service) of the solutions.

However, the Cloud should be type IaaS because analysts and the capabilities of development teams (as well as those of the business), many times are waiting to be provided with infrastructure services, given that they achieve to dispatch their processes with high efficiency thanks to the practices they have adopted (SCRUM, continuous delivery) Therefore the bottleneck of the entire delivery chain is located in infrastructure services, so an improvement in this point would directly lead to a global optimization of the solutions service and the speed of delivery of services of business.

5.3 Tool analysis of Private Cloud type Iaas

Nowadays there are many solutions in the market for the construction of private clouds. In this context, many Open Source platforms in the cloud were born in response to the need for infrastructure as a service (IaaS) to provide solutions in privacy, optimization and control over virtualized environments. These open source cloud platforms were initially used to build private clouds. Over time, these Open Source solutions evolved to be used in the configuration of public, private or hybrid clouds. To choose among the different open source solutions, it is necessary the analysis to select the most appropriate tool according to the objectives and requirements of the company. Following is presented a comparison of the four most popular open source tools: OpenStack, CloudStack, Eucalyptus and OpenNebula.

5.3.1 OpenStack

Nowadays is the most important Open Source cloud project worldwide, and is used and promoted by leading technology companies. It is a platform type IaaS with a Apache 2.0 license founded by Rackspace and NASA.

5.3.2 CloudStack

"It's the open source IaaS platform from Citrix, it has an ASF 2.0d license developed under the incubator of the Apache Software Foundation's project. As main features include support for hypervisors (XEN, KVM, etc.) and Amazon Web Services, among others". [12].

5.3.3 Eucalyptus

Open source platform type hybrid IaaS belonging to Canonical, has "license GPLv3 and BSD, still has a proprietary part. It is mainly used for the creation of clouds, hybrids or private. " [12].

5.3.4 Open Nebula

"Cloud platform with ASF 2.0 license of Spanish origin, which has evolved towards the formation of a company called C12GLabs. Scalable platform whose objective is the management of data virtualization and the creation of IaaS structures " [12]. Was born in the DSA (Distributed Systems Architecture Research Group), A research group with

headquarters at the Universidad Complutense de Madrid, focused on distributed computing, visualization and IaaS platforms.

5.4 Evaluation matrix

This section provides a comparison of the cloud solutions analyzed, with the objective in two different levels: (i) the general comparison aimed at providing a high level analysis in terms of model, the policy and architecture. And (ii) the functional comparison, whose objective is to compare the supported functionalities.

5.4.1 General Comparison

The overall comparison is provided in Table 5.1, taking into account general aspects: the licensing concession, business model, compatibility of cloud models, ease of installation, architecture, sponsoring companies and frequency of release of new versions.

Properties	OpenStack	CloudStack	Eucalyptus	OpenNebula
Discharge	Apache 2.0 ***	Apache 2.0 ***	Licencia BSD **	Apache 2.0 ***
Commercial model	Free ***	Free ***	Free ***	Free ***
Compatibility Cloud models	**Private, public and hybrid clouds** ***	**Private, public and hybrid clouds** ***	**Private and hybrid clouds** **	**Private, public and hybrid clouds** ***
Installation	**Difficult (Many Options,**	**Medium (few Parts to install)** **	**Difficult (different configuration**	**Easy (no root account required)** ***

	Insufficient automation) *		**possibilities)** *	
Architecture	**Fragmented in many pieces** ***	**Monolithic** *	**Fragmented in many pieces** ***	**Modular (Three components)** **
Businesses that support	AT&T, IBM, Canonical, HP Enterprise ,Rackspace, Redhat, Suse, Intel, Yahoo, Cisco, Dell, VMWare, Huawei, Oracle, Sap, Alcatel, etc. ***	Nokia, Orange, Manzana, Citrix, Huawei, TomTom, Tata, etc. **	UEC, la NASA, Sony, HP, Cloudera, Puma, USDA, FDA **	CERN, CloudWeavers, IBM, Hexafrid *
Average Launch Frequency	<4 months ***	<4 months **	<4 months **	<4 months *
TOTAL	19	16	15	16

Table 5.1. General Comparison.

*Note: * Positive evaluation indicators. The sources consulted for the realization of the table were:* (Serrano, Gallardo, & Hernantes, 2015) [23], (BARKAT, DINIZ DOS SANTOS, & IKKEN, 2015) [3], (Apache CloudStack Features, s.f.) [1], (ReleaseNotes/Liberty, s.f.) [20], (OPENNEBULA KEY FEATURES, s.f.) [14] *y* (Mahmood, 2014) [11].

5.4.2 Functional comparison

The functional comparison looks at the functionalities offered or technical aspects of the four solutions presented, as described in the Table 5.2. The most popular Hypervisors like: Xen, KVM and Vmware are compatible with all platforms, but for example HyperV is not

available in Eucalyptus and OpenNebula. It is important to mention that OpenStack and CloudStack have the highest number of compatible hypervisors, even though there are ways to interact with the unadmitted. Apart from this we compared aspects such as Networks, Interface, Administration, DevOps, Authentication, Live Migration, Load Balancer, Fault Tolerance, High Availability and Security.

Functionalities	OpenStack	CloudStack	Eucalyptus	OpenNebula
Hypervisors that supports	KVM, QEMU, XEN, Vmware Vcenter, LXC, UML, BareMetal y HyperV. ***	KVM, XEN, Vmware VSphere, HyperV y LXC. **	KVM, XEN y Vmware. **	KVM, XEN y Vmware Vcenter. **
Networks	**-Two modes** 1. Legacy: Flat Networking, VLan Networking y VMWare 2. Neutron: Firewall, DNS, servidor DHCP, servicios Layer 2 and 3 y VMWare ***	VLan Networking, VMWare, Firewall, Reserved System IP Addresses **	**IP Elastic, safety groups, DHCP Server, Layer 2 Services, Isolated VMs in Four Modes:** 1. Managed, 2. ManagednovLAN, 3. Static, en (1) y (2) Automatic creation of bridges. **	**Services layer** 2, Network - dummy, 802.1Q, IPtables, ovswitch y VMWare **

Interface	**Web interface easy to use, API compatible with EC2, S3, REST interface. Work at OCCI** ***	**Web interface easy to use, Command-line tools, REST interface. API compatible with AWS EC2 and S3** ***	Work at EC2 and S3, REST Interface. **	Work at OCCI, EC2 , EBS, Ruby, Java and API XML-RPC **
Administration	Web UI and, CLI ***	Web UI and, CLI ***	Web UI and, CLI ***	Web UI and, CLI ***
User administration	**Yes** ***	**Yes** ***	**Yes** ***	**Yes** ***
DevOps	Chef, puppet, and Ansible. ***	Chef and puppet. **	Puppet *	Chef and puppet. **
Authentication	**X509 and LDAP Credentials** ***	LDAP and SSH ***	**Credentials X509, CHAP** ***	**Credentials X509, LDAP** ***
Live migration	**Yes** ***	**Yes** ***	**no**	**Yes** ***
Load Balancer	**Yes** ***	**Yes** ***	**Yes** ***	**Yes** ***

Fault tolerance	**VM Programming, replication, and AWS load balancer. *** **	**VM Programming and Replication ** **	**Through AWS load balancer * **	**VM Programming and Replication ** **
High availability	**Yes *** **	**Yes *** **	**Yes *** **	**Yes *** **
TOTAL	**36**	**31**	**24**	**29**

Table 5.2. Functional comparison.

*Note: * Positive evaluation indicators. The sources consulted for the realization of the table were:* (State of the Art in Virtualization Solutions / Cloud Management Systems, Openstack, 2013) (Serrano, Gallardo, & Hernantes, 2015) [23], (BARKAT, DINIZ DOS SANTOS, & IKKEN, 2015) [3], (Apache CloudStack Features, s.f.) [1], (ReleaseNotes/Liberty, s.f.) [20], (OPENNEBULA KEY FEATURES, s.f.) [14] *y* (Mahmood, 2014) [11].

As detailed in the results obtained, OpenStack has a higher score in the parameters evaluated in the matrices with respect to the other platforms evaluated ($16 + 39 = 55\ points$), Overcoming CloudStack ($16 + 31 = 47\ points$), Openebula ($16 + 29 = 45\ points$) y Eucalyptus ($15 + 24 = 39\ points$), by means of which it has been determined that the platform to implement is: OpenStack.

Chapter 6

Brief description of OpenStack

"OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter, all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface.". [24]. As illustrated in figure 6.1.

Trying to achieve a simple definition but at the same time demonstrate all the strengths of OpenStack, it is possible to point out that: OpenStack provides an intermediate layer between hardware and virtualization, where it allows to manage computing resources in an optimized way whenever it was conceived So that the infrastructure is delivered as a service that fulfills the Cloud characteristics such as self-provisioning, Pay per use, network access, elasticity and multitenant.

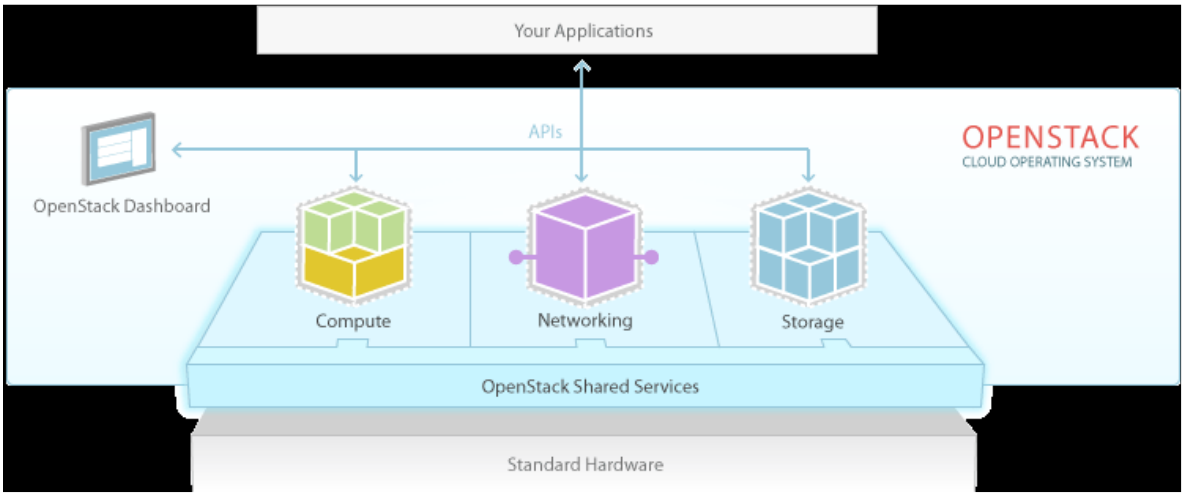

Figure 6.1. OpenStack description. Taken from [24]

This solution has emerged from a series of interrelated projects, each of which produces components that fulfill specific functions that contribute to the integrated global construction, that are completely articulated through a general project that integrates all the components.

The result is that OpenStack allows the implementation of IaaS through multiple services that, in a systematized way, fulfill different purposes to achieve the correct operation of said infrastructure. The main services, the name of the project and its functionality in OpenStack are described in Table 6.1 follow.

SERVICE	PROJECT	DESCRIPTION
Compute	(Nova)	It manages the life cycles of the instances in the OpenStack environment.
Networking	(Neutrón)	It is a system for the networks management (layer 2 and 3 service, DNS, Firewall, Load Balancing as Service (LBaaS), Intrusion Detection System (IDS), ..) and IP addressing.
Dashboard	(Horizon)	Provide self-service web interface to interact with OpenStack services.
Identity Service	(Keystone)	Provides an authentication and authorization service for OpenStack services.
Image Service	(Glance)	Provides an image storage service (template) of OpenStack.
Block Storage	(Cinder)	Provides a storage volume service for Compute.

Table 6.1. Descripción of main services of OpenStack .Taken from [2]

OpenStack has gone through different versions until it reaches to the last one valid at the time of writing this report. Currently the most recent version is called **Pike.**

The project began in September 2016 for infrastructure initiatives, and it was decided to work with the Kilo version since it was the most stable at the time.

Then with the improvements that were being presented in the later versions began to study a little of them to perform tests and if it justified the change of version.

The table 6.2 shows the different versions that have released of OpenStack and shows the incorporations and changes that have had in the services by each version:

NAME VERSION	CREATION DATE	COMPONENTS INCLUDED
Austin	**October 21, 2010**	Nova, Swift
Bexar	**February 3, 2011**	Nova, Glance, Swift
Cactus	**April 15, 2011**	Nova, Glance, Swift
Diablo	**April 5, 2012**	Nova, Glance, Swift, Horizon, Keystone
Folsom	**September 27, 2012**	Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder
Grizzly	**April 4, 2013**	Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder
Havana	**October 17, 2013**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder
IceHouse	**April 2014**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder
Juno	**October 11, 2014**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Ceilometer, Heat, Trove, Sahara.
Kilo	**April 30, 2015**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Ceilometer, Heat, Trove, Sahara, Ironic.

Liberty	**November 20, 2015**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Ceilometer, Heat, Trove, Sahara, Ironic, Zaqar, Magnum, Manila, Barbican, Congress.
Mitaka	**7 April 2016**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Heat, Ceilometer, Trove, Sahara, Ironic, Zaqar, Manila, Designate, Barbican, Searchlight, Magnum
Newton	**6 October 2016**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Heat, Ceilometer, Trove, Sahara, Ironic, Zaqar, Manila, Designate, Barbican, Searchlight, Magnum, aodh, cloudkitty, congress, freezer, mistral, monasca-api, monasca-log-api, murano, panko, senlin, solum, tacker, vitrage, Watcher
Ocata	**22 February 2017**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Heat, Ceilometer, Trove, Sahara, Ironic, Zaqar, Manila, Designate, Barbican, Searchlight, Magnum, aodh, cloudkitty, congress, freezer, mistral, monasca-api, monasca-log-api, murano, panko, senlin, solum, tacker, vitrage, Watcher
Pike	**30 August 2017**	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, Heat, Ceilometer, Trove,

		Sahara, Ironic, Zaqar, Manila, Designate, Barbican, Searchlight, Magnum, aodh, cloudkitty, congress, freezer, mistral, monasca-api, monasca-log-api, murano, panko, senlin, solum, tacker, vitrage, Watcher

Table 6.2. OpenStack Versions. Taken from [21]

6.1 General Architecture

"OpenStack is highly configurable to satisfy different needs based on computing, networking and storage requirements". [2]

The services seen in Table 3 must be available in the system. There are different ways to group these services to give consistency to the OpenStack infrastructure. These services are installed in the nodes (virtual or real machines) where the services will be executed.

There are several ways to group these services to give more scalability and distribution to the infrastructure. To explain some of the clusters there have been references in the installation guide of "OpenStack Kilo for CentOS 7". [15]

Some of its architectures will be explained below.

6.1.1 Description of the OpenStack Networking Architecture (Neutron) three nodes

The nodes are the physical machines where the services of OpenStack are executed, specifically with this architecture of 3 nodes as shown in figure 6.2, the services are distributed as follows:

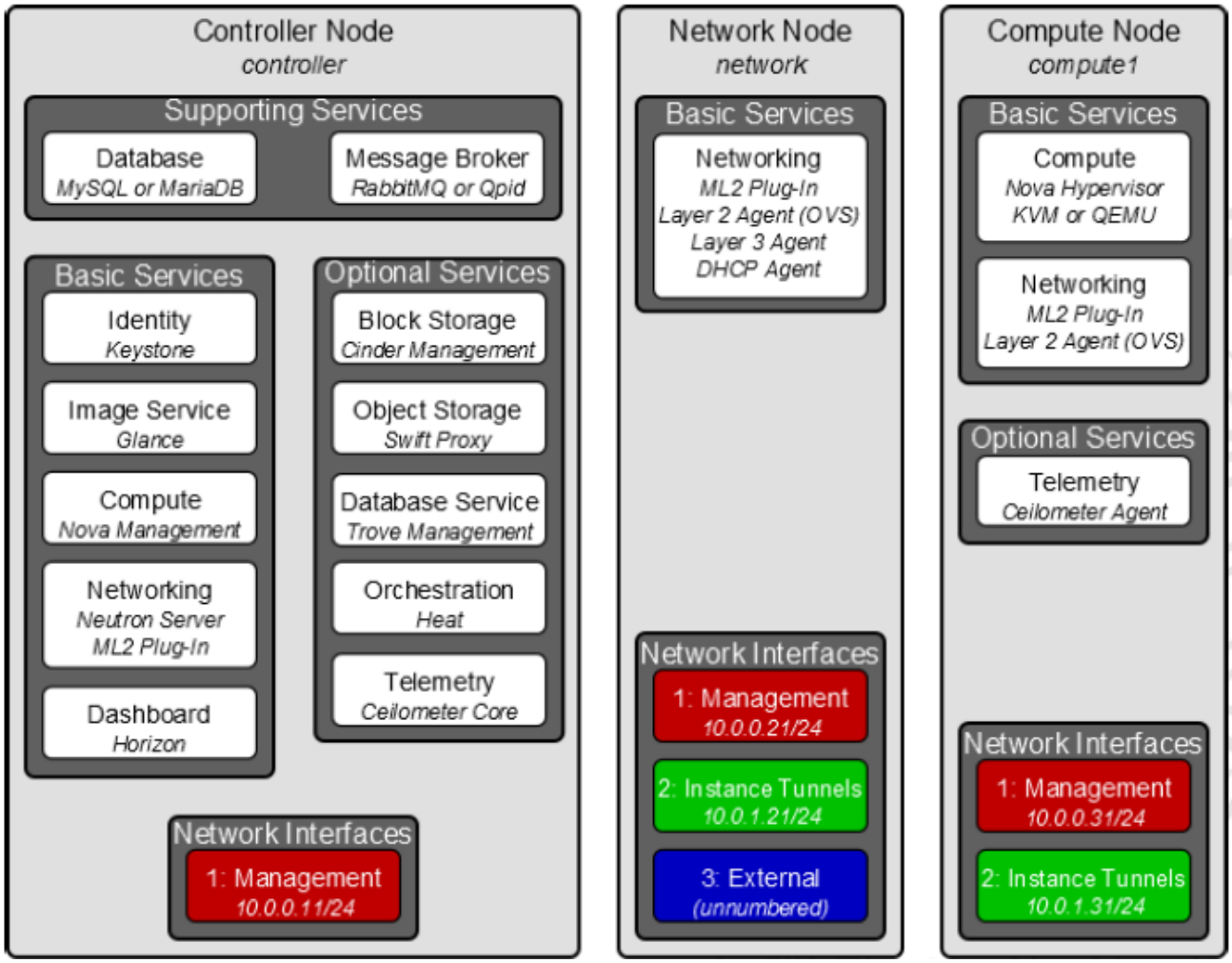

Figure 6.2. OpenStack Neutron. Taken from [17]

The controller node: executes the services of authentication, imaging, Compute management, Neutron networks and the control panel service. It also includes support services such as database, tail messaging and NTP services.

The network node: executes network plug-in, Ml2, Layer 2 agent (OVS), Layer 3 agent and DHCP agent. It also includes Routing, NAT and DHCP services. It also provides internet connectivity to instances in OpenStack.

The compute node: Excecute the hypervisor that operates the instances or virtual machines. By default, KVM as the hypervisor. It also includes the plug-in and a layer 2 agent that connects the tenant networks of OpenStack.

6.1.2 Description of the architecture Openstack Networking (Legacy Networking) two nodes

In the architecture of 2 nodes, as shown in figure 6.3, the services are distributed as follows:

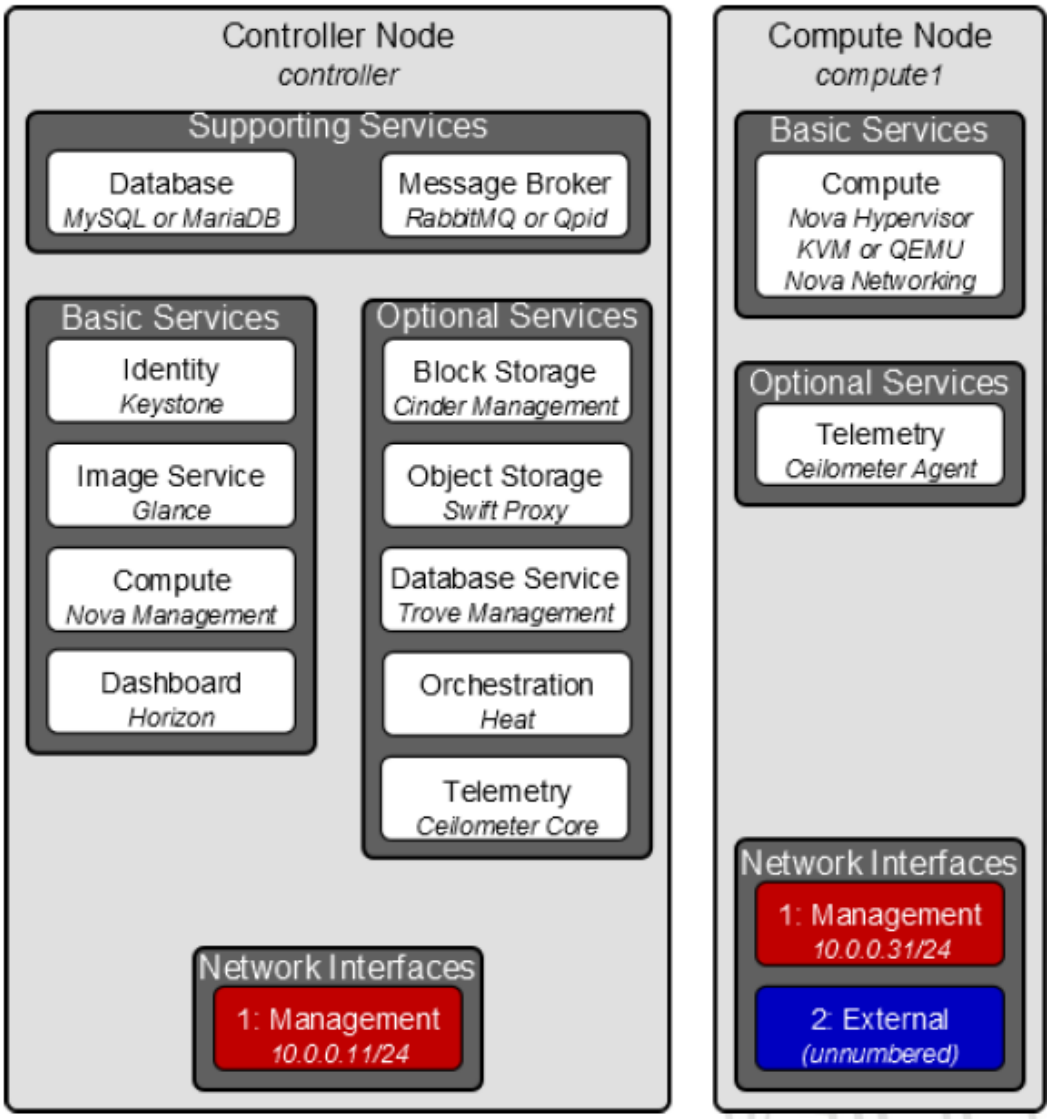

Figure 6.3. OpenStack Legacy Networking. Taken from [17]

The controller node: executes the services of authentication, of images, management of the Compute, Legacy networks and the service of control board. It also supports database services, queue messaging and NTP services.

The compute node: runs the hypervisor that operates the instances or virtual machines. By default, KVM is the hypervisor. It also includes a network module and provides a firewall.

6.2 Considerations about examples of architectures

The previous section shows two examples of architectures that were extracted from the official installation guide of OpenStack and are shown in this guide for the basic learning of services.

OpenStack is a very broad infrastructure, and these two are the main architectures that show in the guide as a learning process, but in a more pleasant and simple way you can learn the operation, installation and how each does it works of the services of OpenStack.

In "section 6.6.2.2 Installations of OpenStack.", Use of the architecture "Legacy Networking" for the prototype.

6.3 Cloud scalability

In the Chapter 5 of the OpenStack Operations Guide [16] treat the scalability of the system.

"The determination of the scalability of the cloud and how to improve it is an exercise with many variables to balance. No solution meets the scalability goals of all systems. However, it is useful for tracking metrics." [4]

The starting point for most systems is the number of cores in the cloud. Through the monitoring application that integrates OpenStack, it can gather information about the number of virtual machines that can queue up in the system and the amount of storage that is required.

For example, they have physical nodes with 10 physical cores, 48 GB of RAM and 1 TB of storage; If the instances are executed with the average flavor, that is to say with 2 virtual cores, 4 RAM and 40GB of storage. It can have 18 virtual machines in Execution.

The calculations for finding the number of instances or virtual machines are done as follows (the values defined for the ratio in RAM and cores are defined by OpenStack in the chapter 5. Scaling, s.f.):

As the ratio of RAM is 1.5, the calculations are as follows:

$$Virtual\ Memorial\ Calculation: 48 * 1.5 = 72GB$$

$$Instances\ Calculation: \frac{72GB}{4GB} = 18$$

The storage size calculation is the multiplication of the number of virtual machines by size. In the example

$$Storage\ Calculation: 18 * 40GB = 720GB$$

In addition, as the ratio of cores is 16: 1, that is to say a physical core are equivalent to 16 virtual, it could have instances up to 8 cores, as doing the calculation.

$$Cores\ calculation: \frac{10\ cores*16}{18} = 8\ possible\ colors\ per\ instance$$

Chapter 7

Prototype SURACLOUD

Firstly, the result obtained is shown, illustrating how the Private Cloud implemented in the prototype manages to deliver infrastructure services in an easy and intuitive way and the use case that shows its impact on the reduction in the delivery time of the same ones.

7.1 Operation platform SURACLOUD

The first step is to access the portal by the following link: http://10.203.92.45

Figure 7.1 shows the login screen of the platform of SURACLOUD, it is observed that it is a friendly environment with the user. Enter credentials to start the world SURACLOUD.

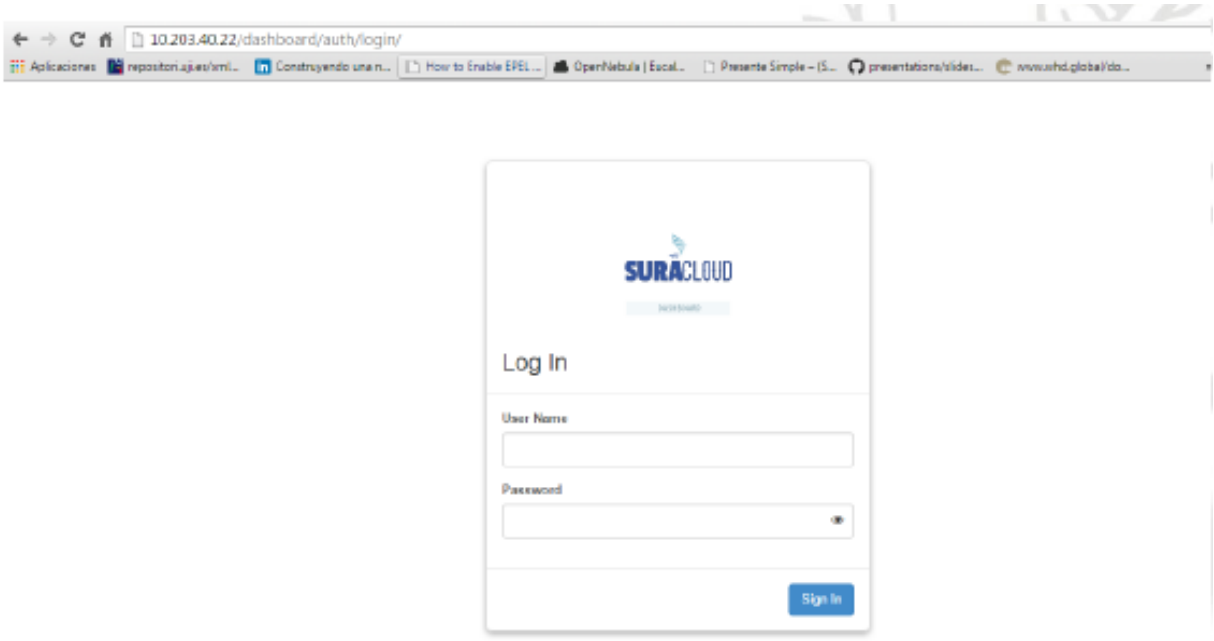

Figure 7.1. Platform Access SURACLOUD

Figure 7.2 shows when entering the "project" button we can access "API Access" where we can see the APIs that are exposed.

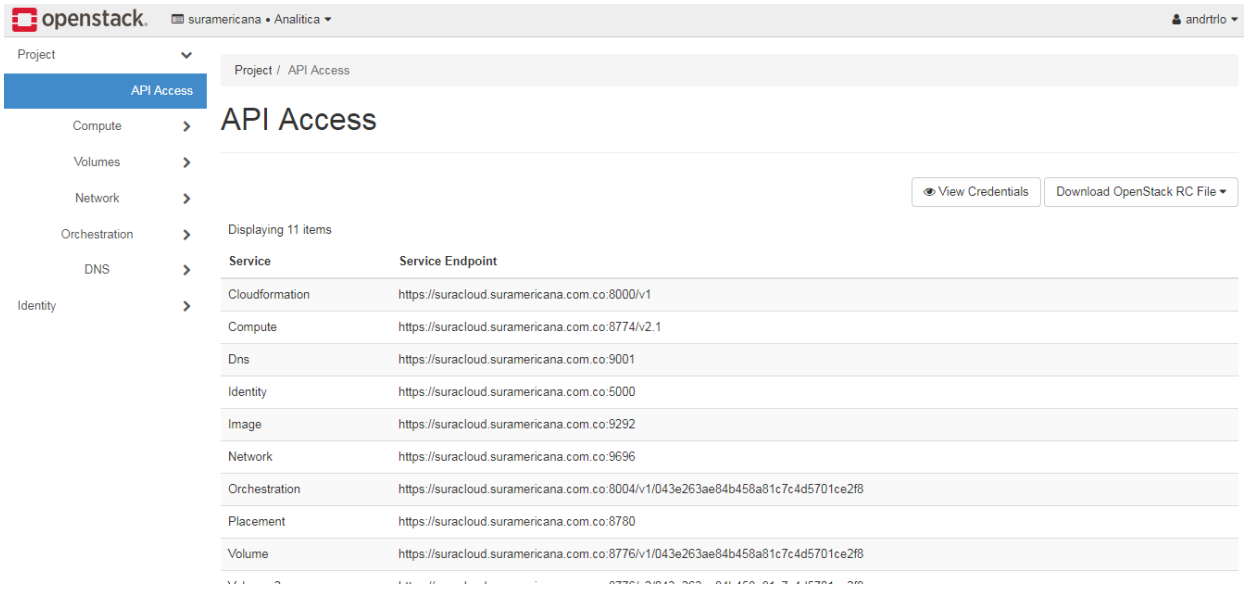

Figure 7.2. APIs exposed

In addition, we can see in the top left the Openstack services, if we click on "compute" and then in "Overview" the options of figure 7.3 are shown, which are the resources that the administrator gives the user, in the image you can see that it can create up to 10 instances, with a capacity of 20 VCPUs, 50 GB of RAM and other available features.

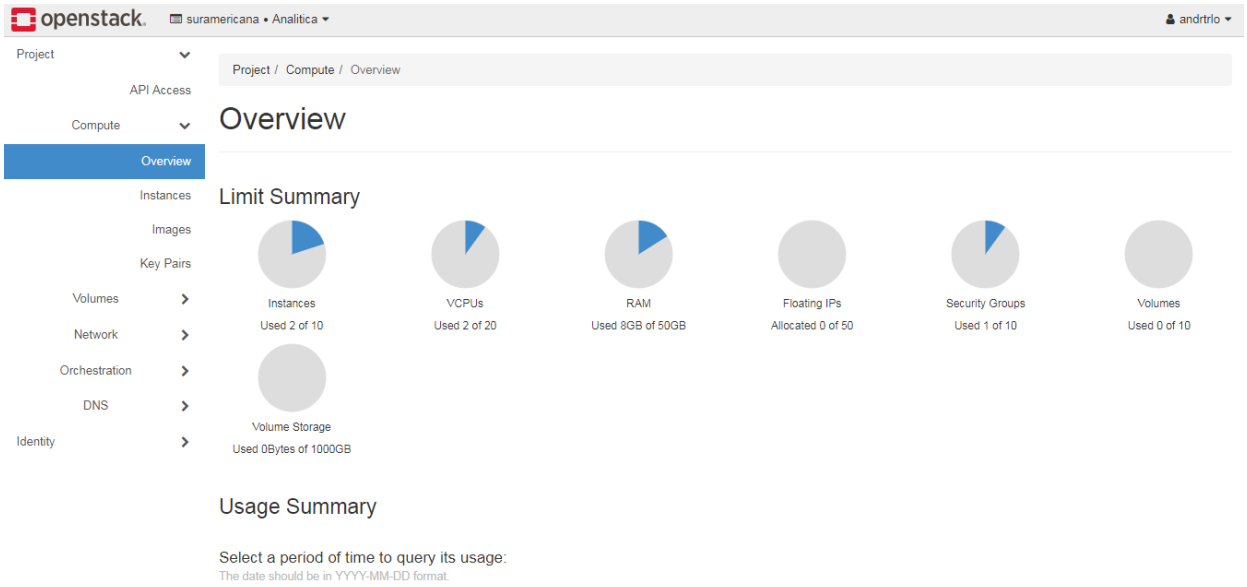

Figure 7.3. Overview

In "instances" we can see the instances that have been created and those that are being created at the moment, also in the upper right we can launch one of them by clicking on the "launch instance" button as shown in figure number 7.4.

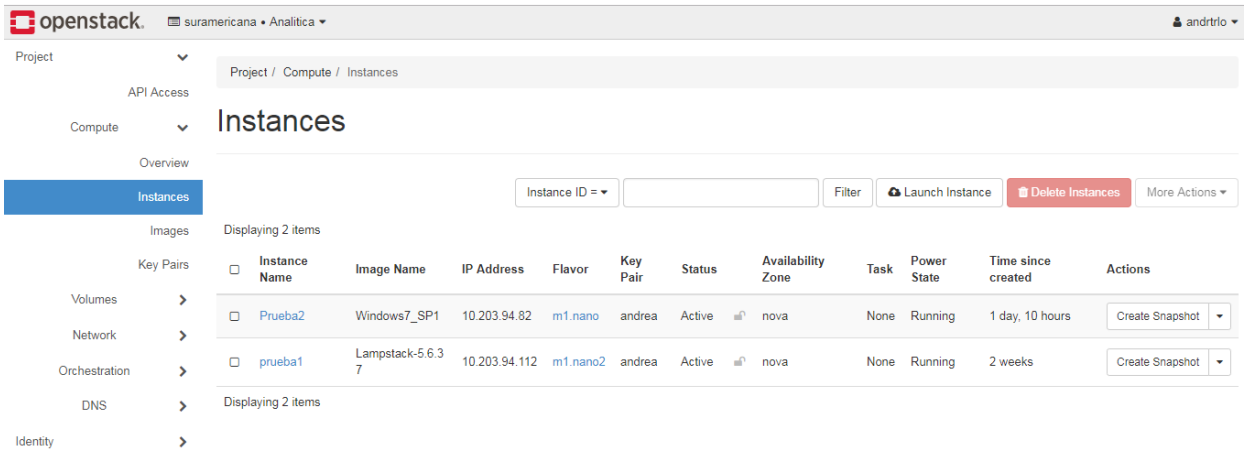

Figure 7.4. Instances

In figure 7.5 we can see the available images, which have been previously loaded by different users. Each user can upload their own images, which must have a .qcow2 format; if you have an ISO you must first perform a conversion, which is somewhat complex.

This is another benefit that the platform offers us, the ability to display an image in seconds, compared to a traditional hypervisor like VMware.

Among the available images are: CentOS 7, mongodb, OpenSuse, Ubuntu 14.04, Windows 7, etc.

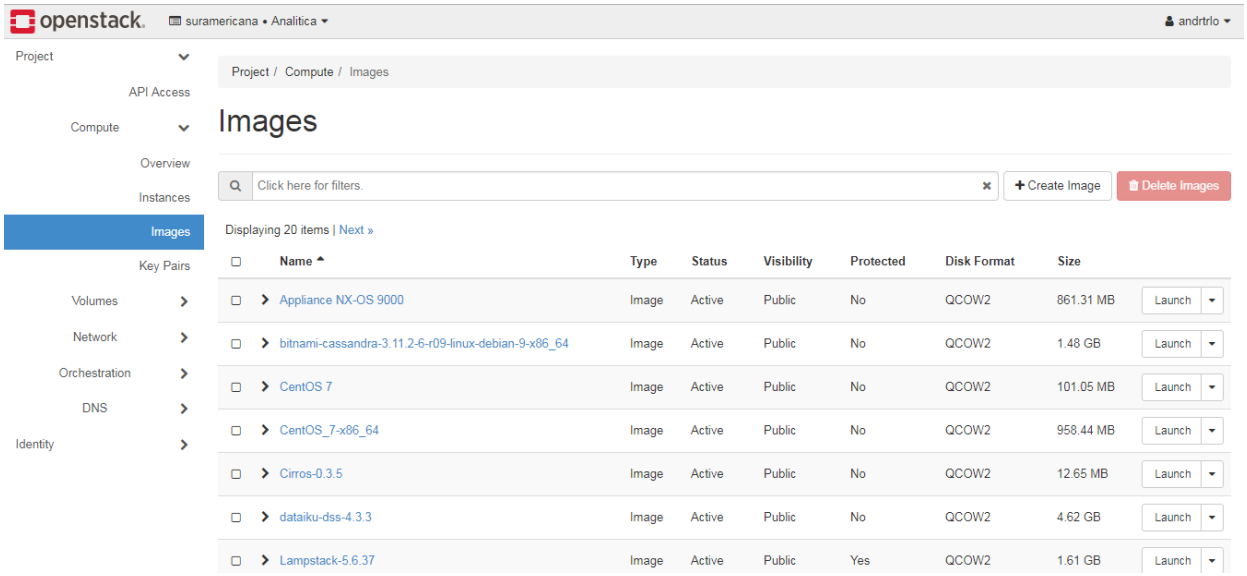

Figure 7.5. Images

In the option "key pairs" is the public key that is generated together with the administrator the first time a user is going to enter the platform, in figure 7.6 you can see the data of the user "Andrea".

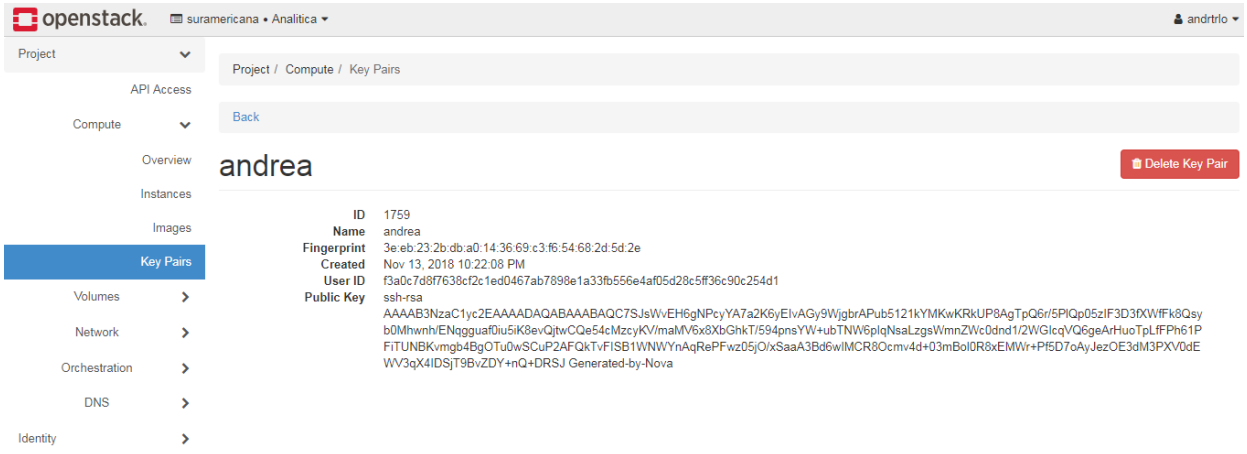

Figure 7.6. Key pairs

If we access the options that "network" gives us, we can see the network topology that the project manages, in figure 7.7 we can see the different networks, the address range Ip and the instances that are connected to them. You can also create routers and create the routing that is required.

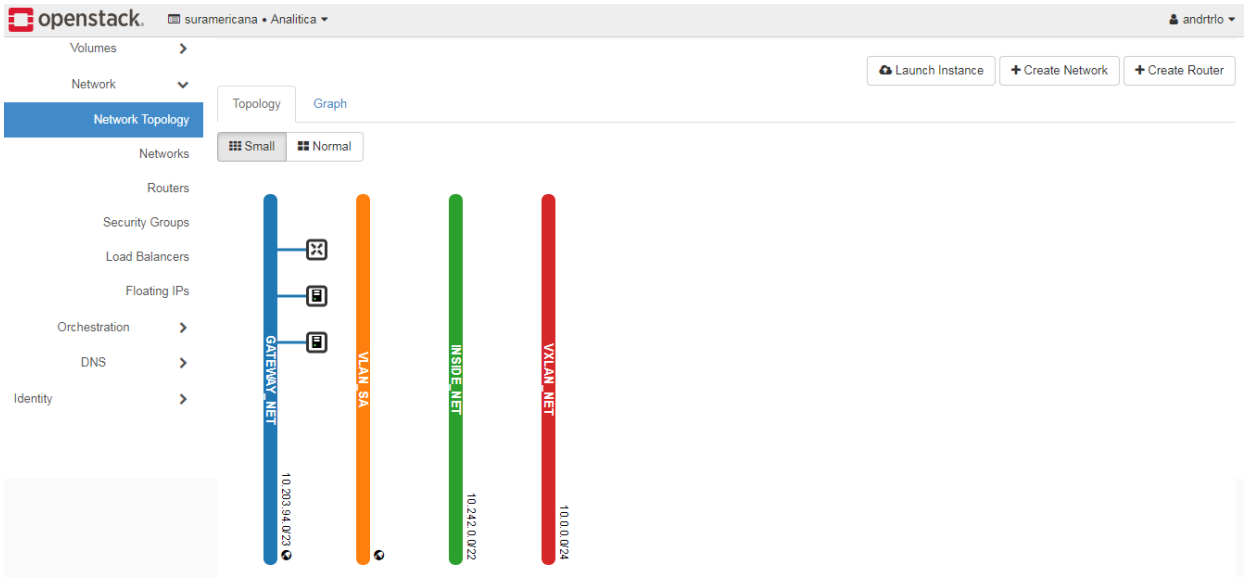

Figure 7.7. Network topologies

In figure 7.8 we see that we enter the option "security groups". There we can create different groups, and when entering each of them manage the rules that are desired, such as through which ports or what machines can access, incoming and outgoing traffic, etc.

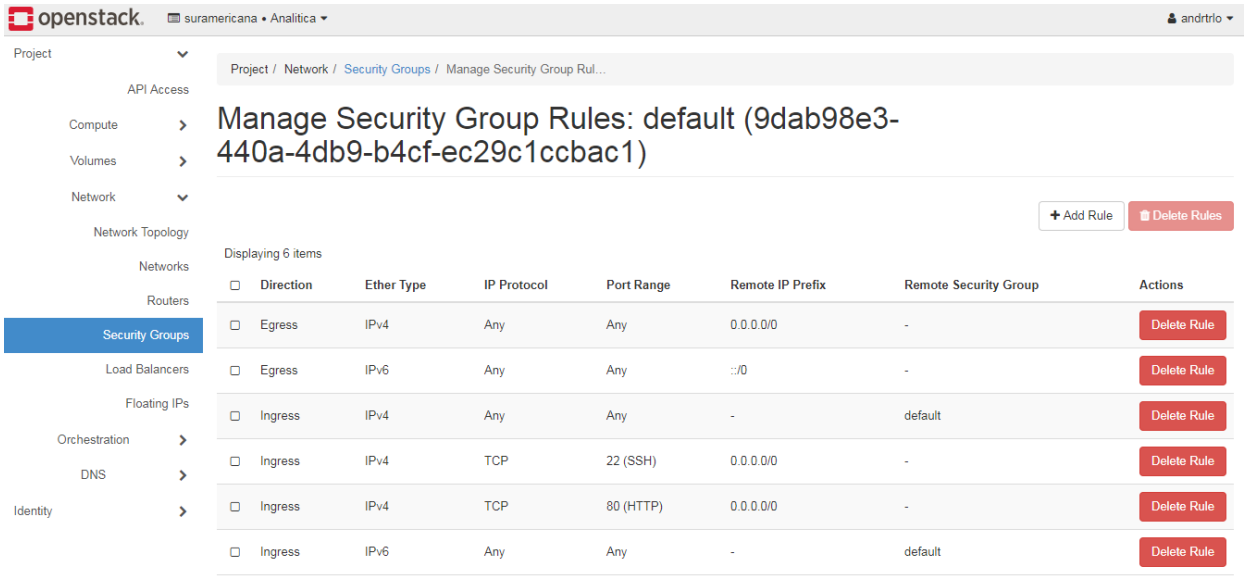

Figure 7.8. Security Groups

If we enter the option "identity" as shown in figure 7.9 and 7.10 we can see the different projects installed, and the different users, being a common user I can only see my own information, but if we enter with the administrator user we can see all the projects and all the users that can access them.

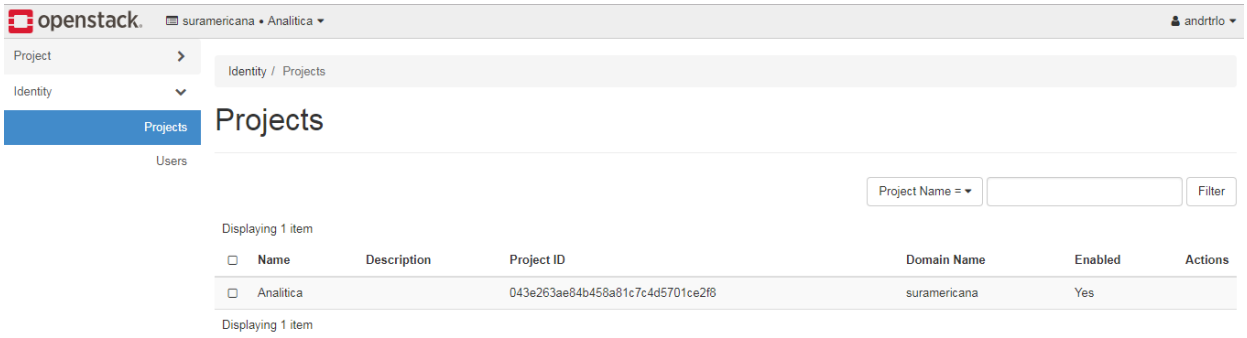

Figure 7.9. Projects

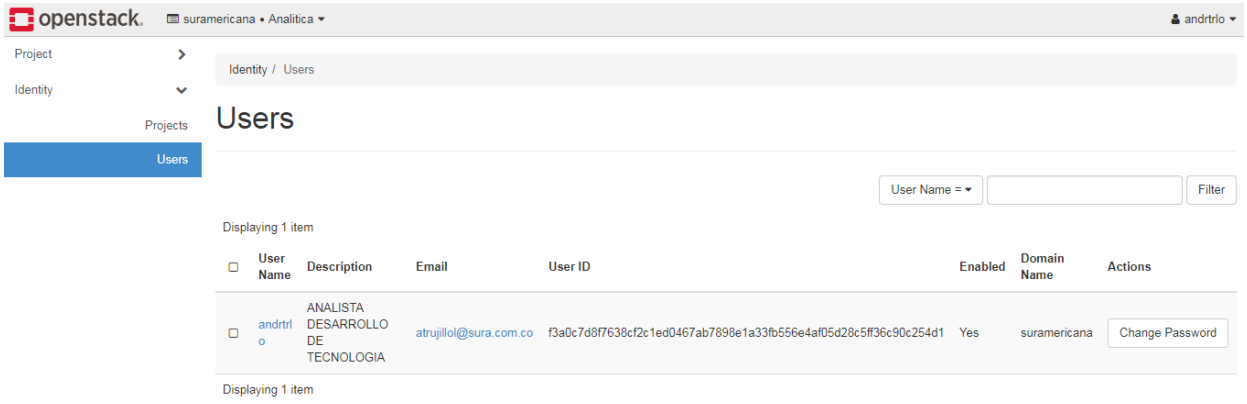

Figure 7.10. Users

7.2 Case study: delivery of an infrastructure service

A proof of concept was performed with a practical case of delivery of an infrastructure service, which required the provisioning of a virtual machine, with an operating system installed and its network configuration with an IP of the VLAN of the environment that would allow it Communication with external components.

Figure 7.11 shows a comparison of the agreed time in the SLA contracted in the current scheme and real time of execution time under the conditions in which the indicated infrastructure service is currently provided. As shown in the figure, the interested in receiving the service are required to interact with many actors (Infrastructure analyst, telecommunications analyst, virtualization provider, telecommunications provider, and operating system administrator) That participate in the process, and also the output of each step of the process that serves as an input to the next step, is done manually by intermediation of the interested, this induces dead times that are sometimes very long and are not calculated in illustration. So the reality is that the service delivery time is several days.

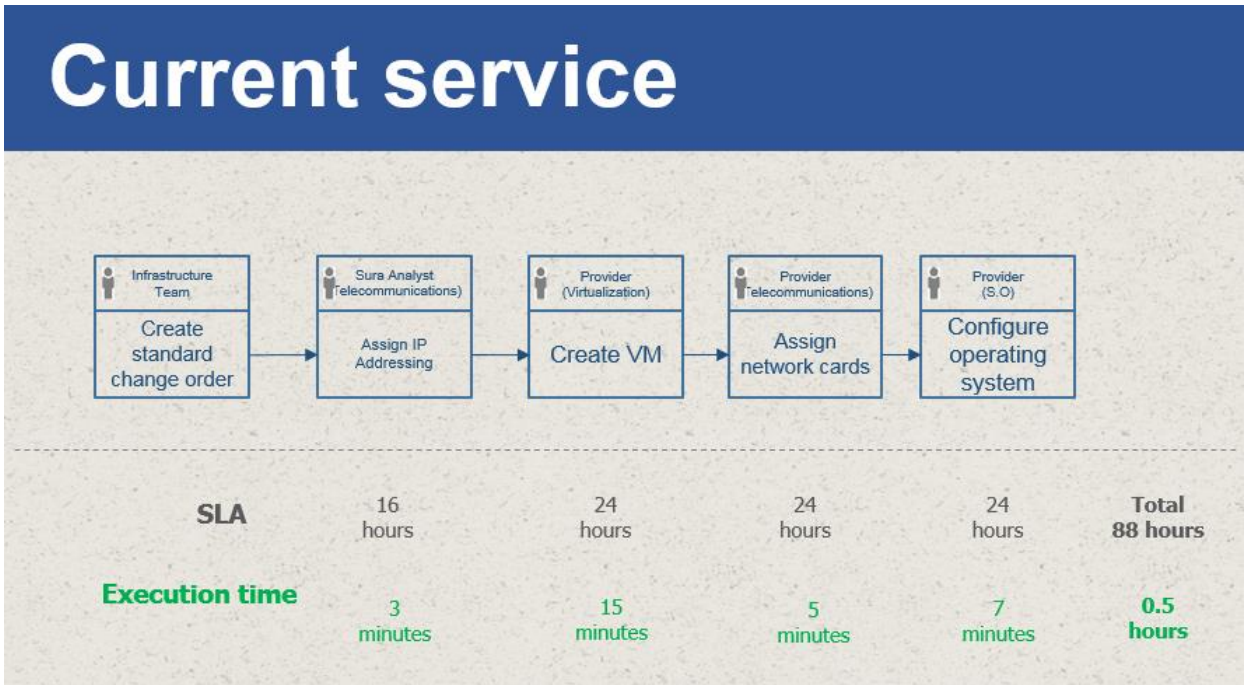

Figure 7.11. Delivery time of a current service.

Now proceed to demonstrate the steps and time for the creation of an instance in SURACLOUD.

For the creation of an instance in SURACLOUD it is enough with some simple steps:

a) In Figure 7.12, in the left menu select Project, then Compute and then click on Instances and then on the "Launch Instance" button in the top right (yellow).

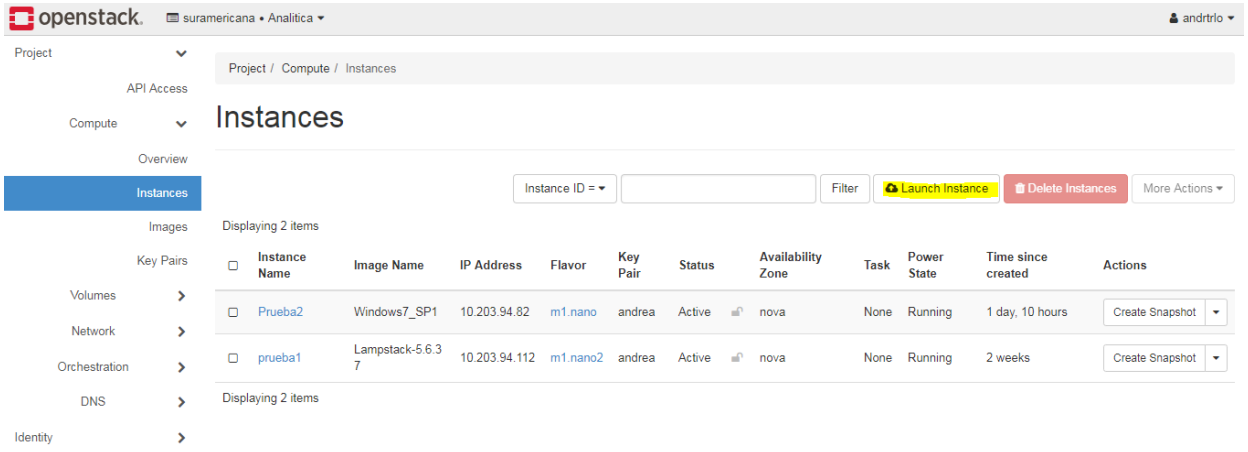

Figure 7.12. Instance menu.

b) In figure 7.13, the menu to launch instance is shown, in this menu the different parameters are entered, those with asterisk are obligatory.

First of all, we give a name to our new instance, then a small description if desired, then we choose an availability zone, these areas are like datacenters, in which I can divide the environments, be it development, be it laboratory or production, or also different regions, such as Torino, Milano, Rome, etc.

Coun is an indicator that shows the number of machines that will be deployed with the same characteristics, in terms of time if 3, 4 or 5 instances are created, this will take the same time if for example I only create one.

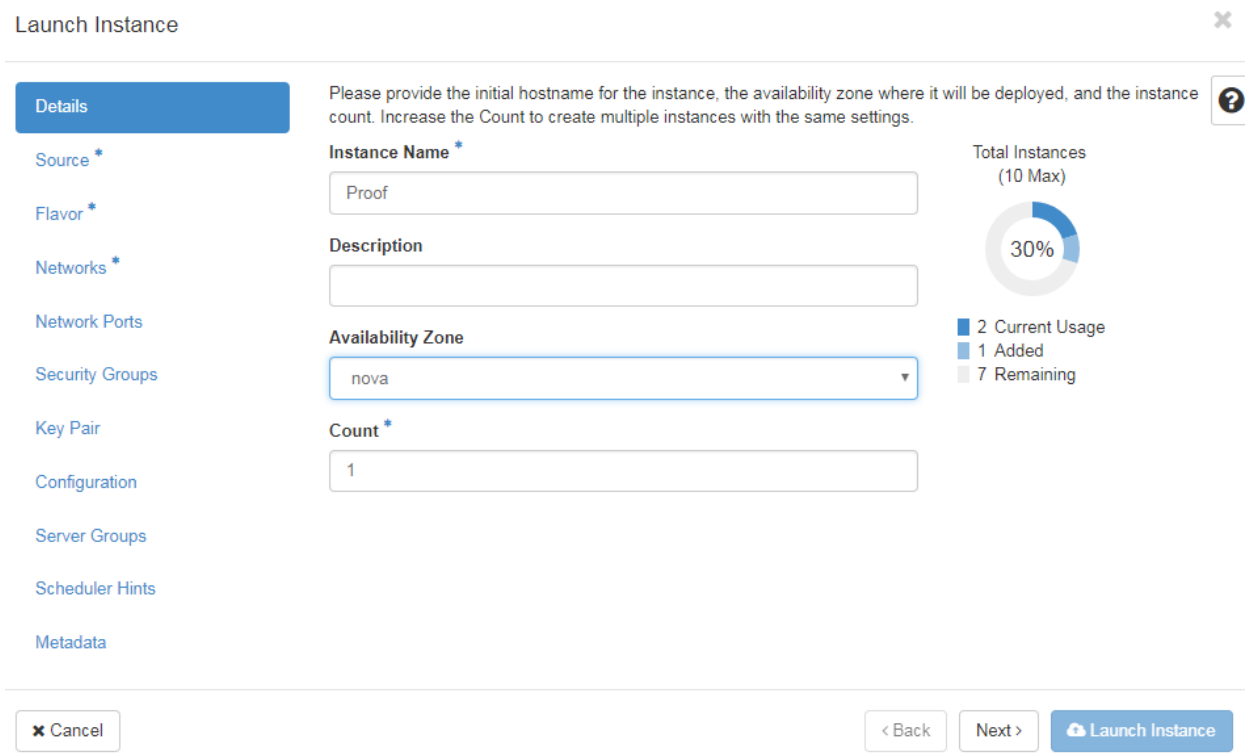

Figure 7.13. Launch instance menu

c) We click on next and we see the options of figure 7.14. Here we must choose the origin source of our instance, we choose "image".

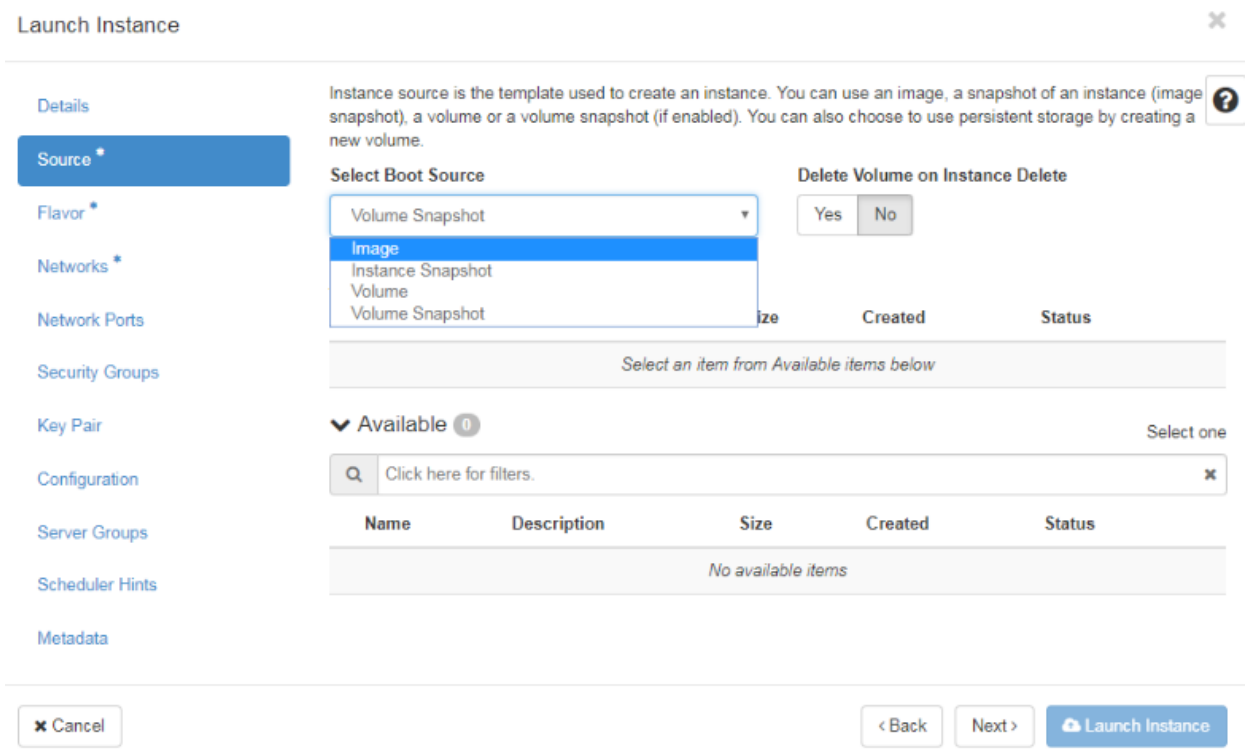

Figure 7.14. Boot Source

d) by clicking on the image a series of images that are already available are displayed and we choose one of them, as shown in figure 7.15.

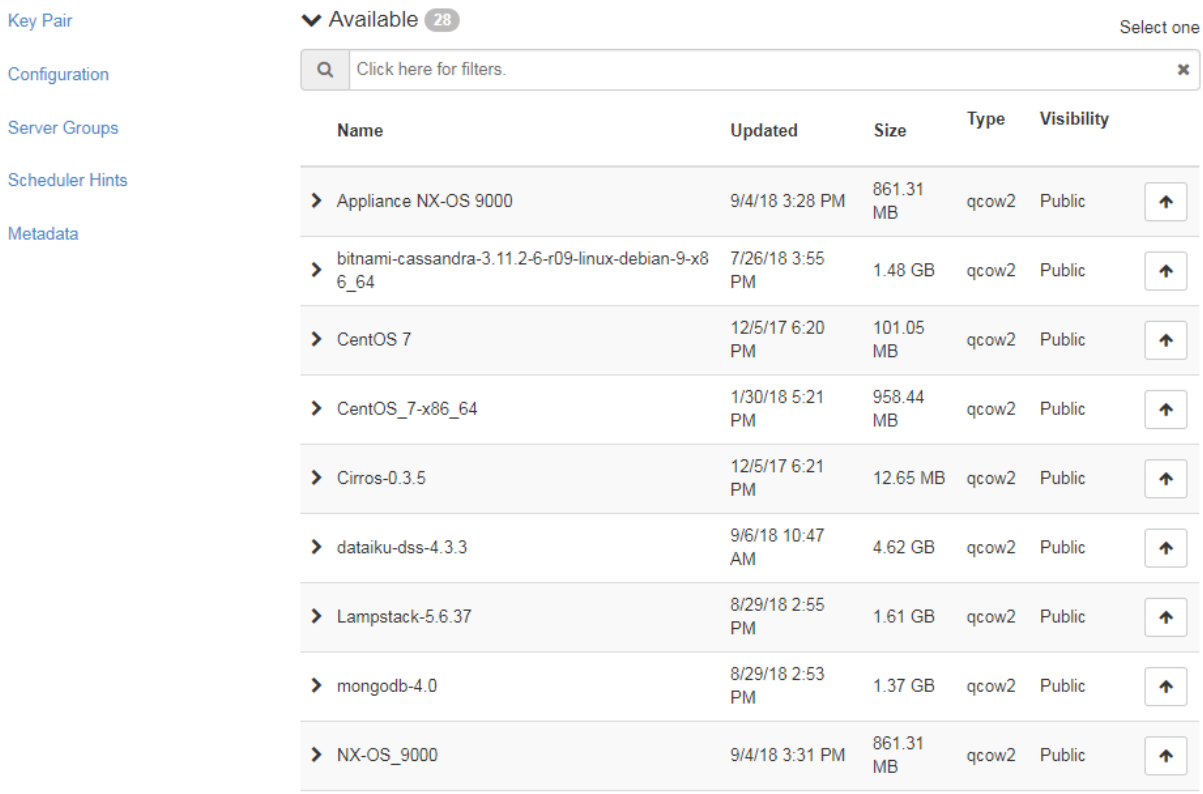

Figure 7.15. Available Images

e) then we must choose the characteristics that the instance will have (Flavors). Figure 7.16 shows predefined characteristics, where the VCPU, the RAM memory, the total disk are indicated, dividing into Root Disk and Ephemeral Disk.

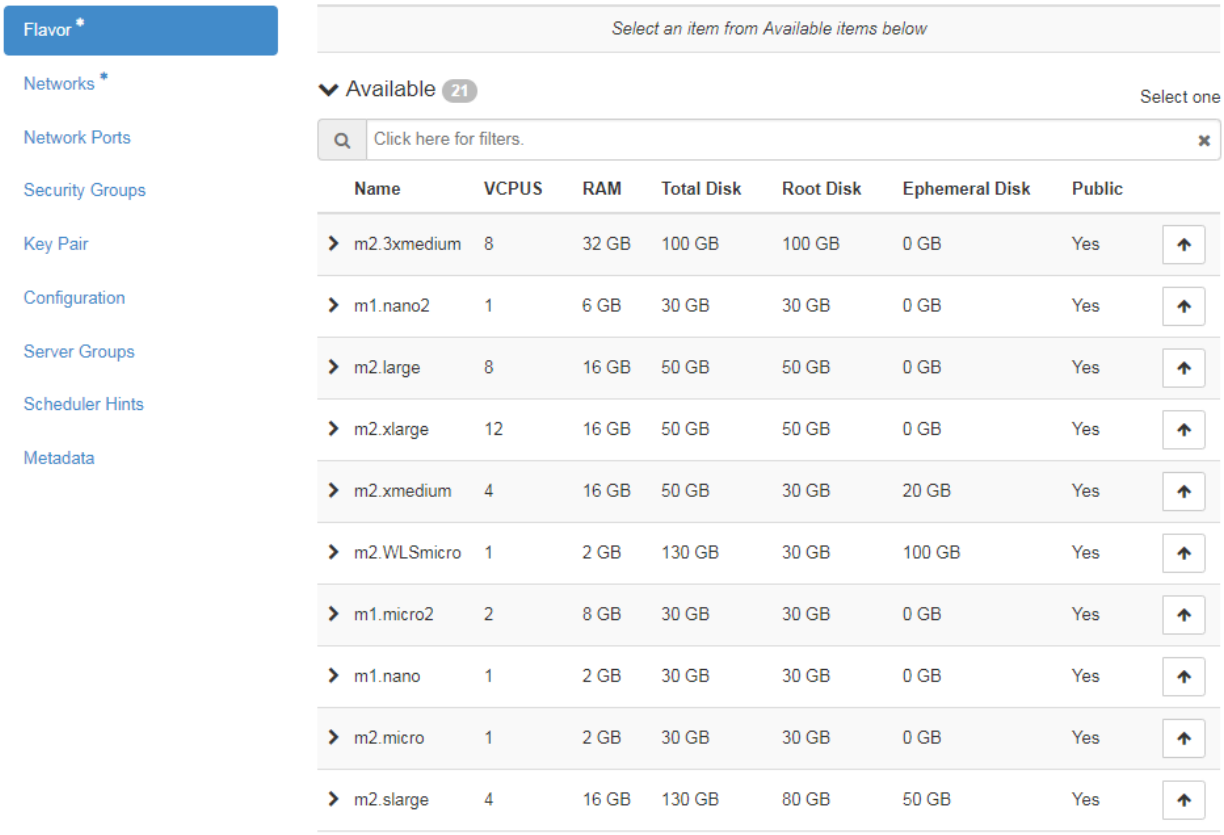

Figure 7.16. Available Flavors

f) in networks, we select a network, in this case the first option that appears in figure 7.17, GATEWAY_NET.

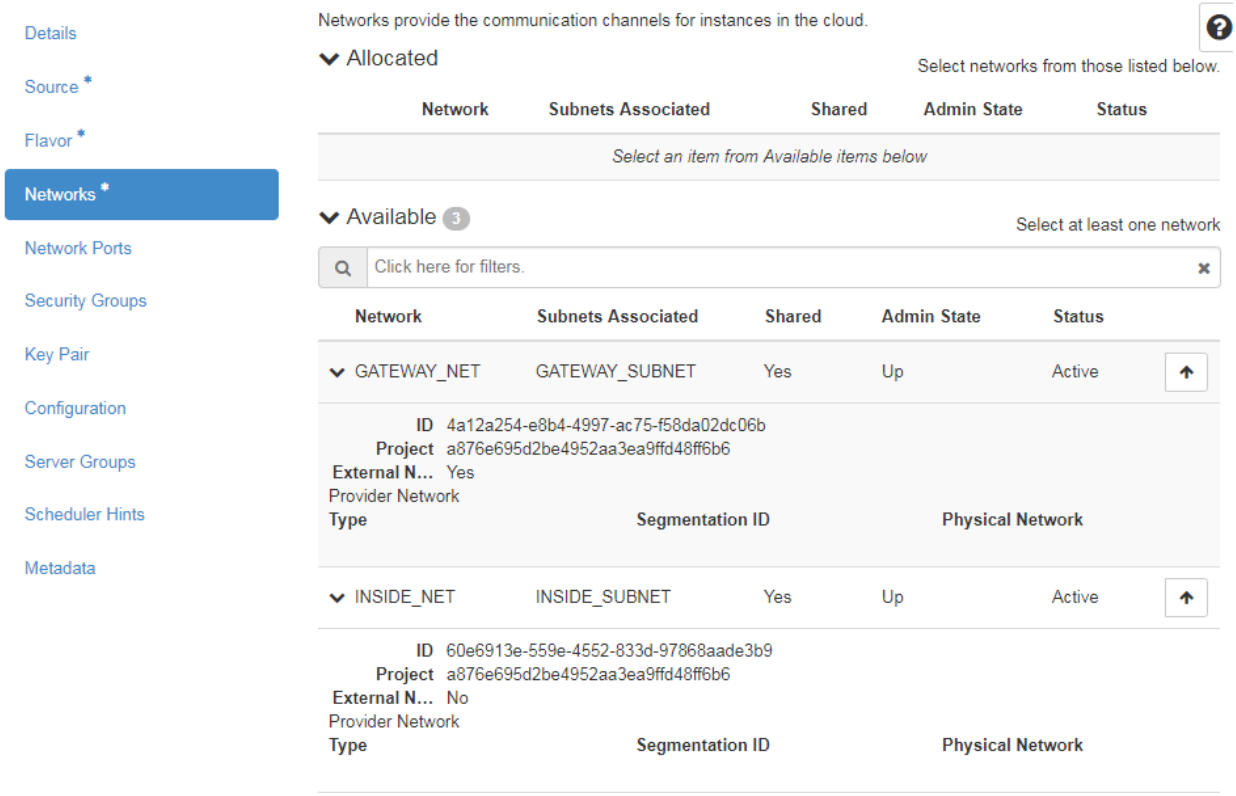

Figure 7.17. Available Networks

g) Since the other options are not obligatory, they set default values, for example in the case of security groups, as shown in Figure 7.18 is "default" with its security rules, just like that can create many more groups

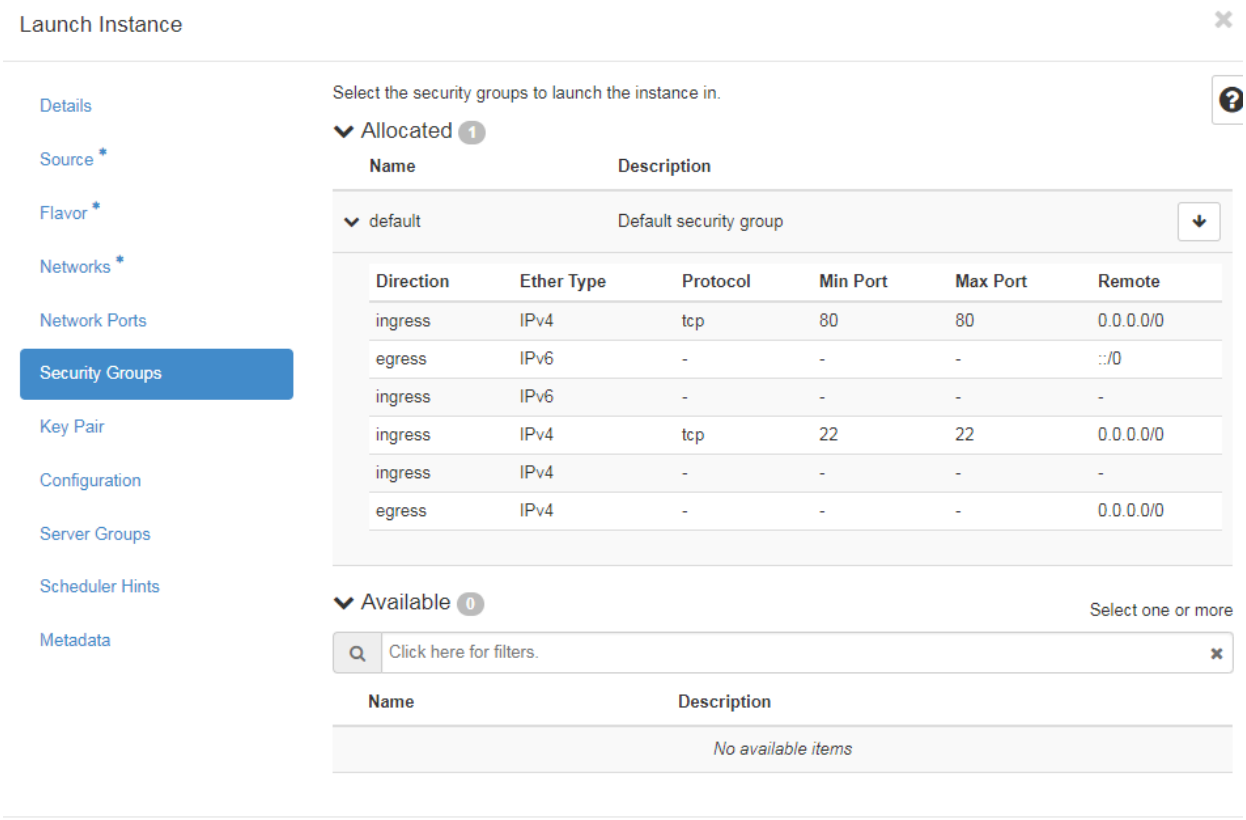

Figure 7.18. Security Groups

h) in the option "key pair", it is by default the name "Andrea" with the public key that is detailed in figure 7.19.

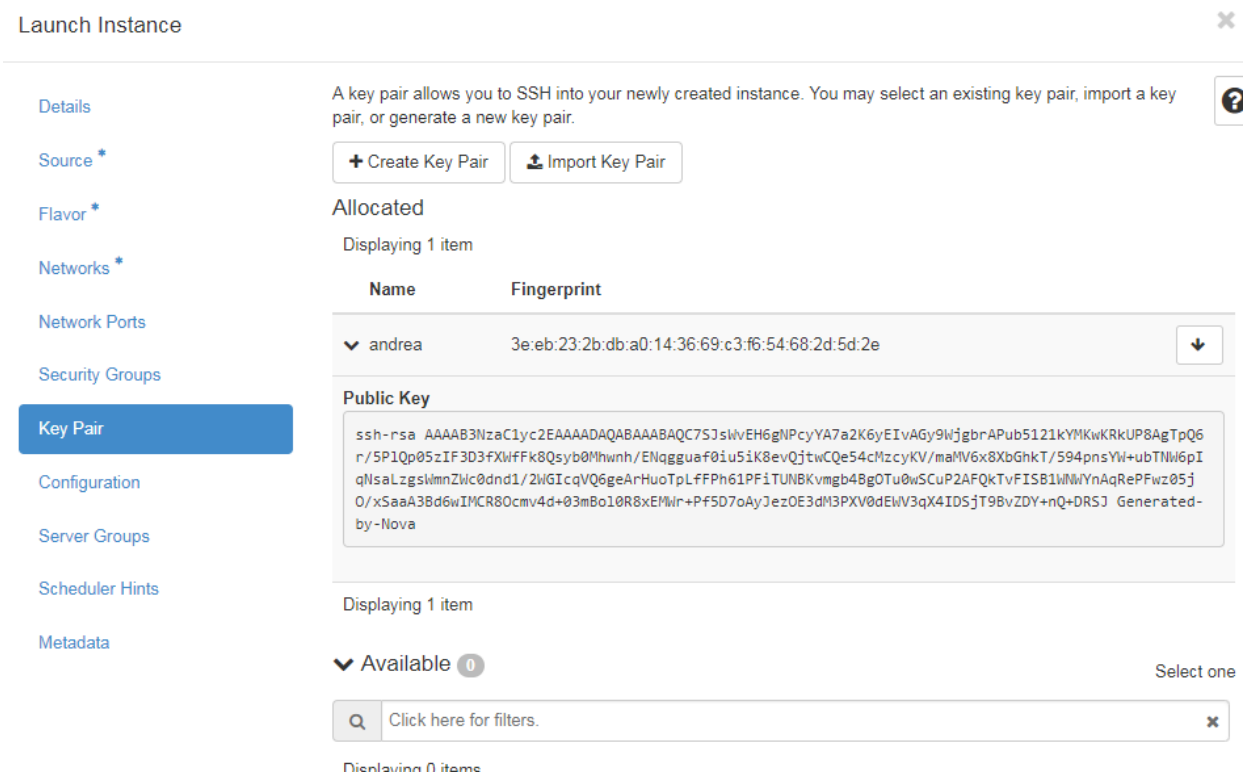

Figure 7.19. Key Pair

i) finally we click on the "Launch instance" button. In figure 7.20, it is observed how the instance of the new server is being generated.

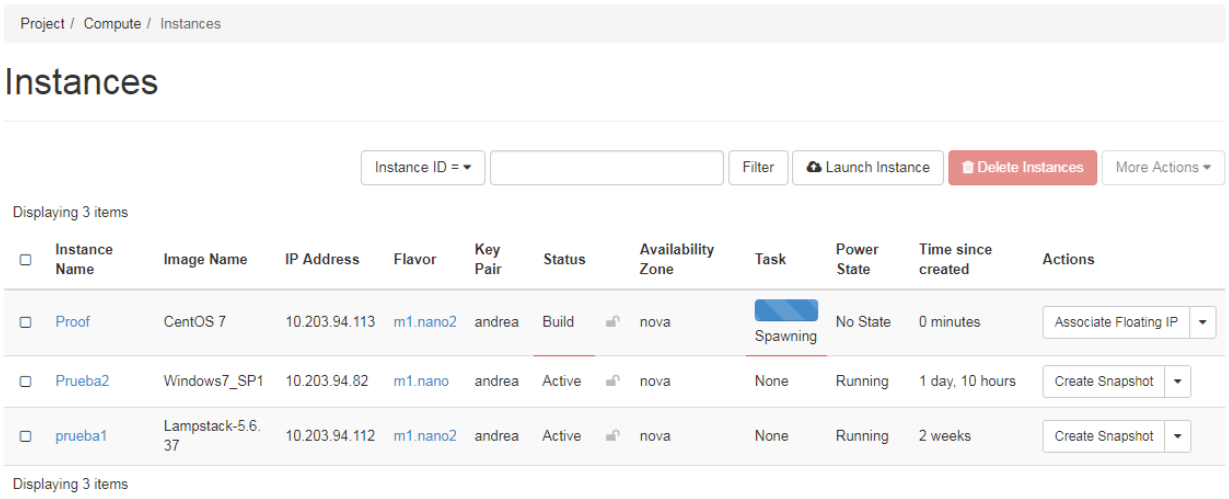

Figure 7.20. Instance Creation

j) In Figure 7.21, it is observed that the process of generating the new server with the provisioning of a virtual machine, with the operating system installed and its network configuration with an IP, has been completed.

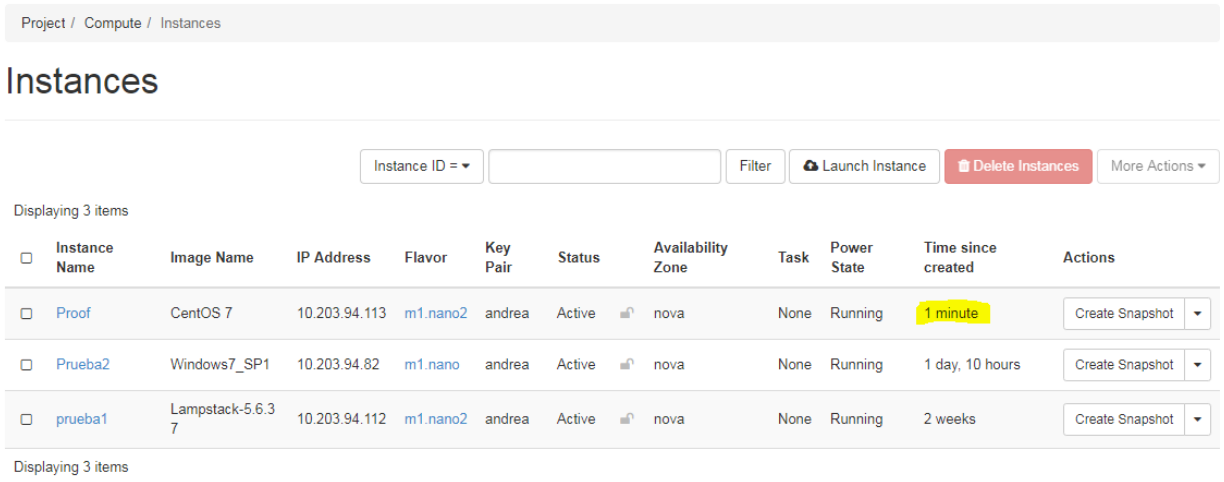

Figure 7.21. Creation time of the instance

The delivery process in the illustrated infrastructure service is done in one minute and the user self-provision without interacting with third parties, which simplifies the process completely, reducing it to a single step the delivery of the provisioning service.

The improvement in delivery time is enormous, and given that currently the infrastructure service has the main bottleneck located, it is not only an impact on the level of this, but this efficiency transitively propagates towards an improvement in delivery of the application service, which in turn induces the effect on the improvement in the delivery time of the business service.

Below is illustrated all the installation procedure that was followed for the implementation:

7.3 Proof of Concept

With the aim of reducing the risks of installation errors and to dissipate the complexity of the same in the final environment of the prototype, it was decided to perform a previous installation on a test environment in a laptop that allowed to test the installation and configuration procedures and were documented Difficulties and findings.

7.3.1 Proof

For the first scenario, an OpenStack test environment was proposed in a DELL laptop with latitude E7450 with the following specifications:

Processor: Core i7 2.6 Ghz fourth generation.
RAM Memory: 16 GB DDR3 a 1600Mhz
Storage: Solid state hybrid hard disk 500 Gb

The Ubuntu 15.04 and VirtualBox operating system were installed on the laptop, and two virtual machines were created with the minimum requirements recommended by the OpenStack Installation [15] and each instance was installed with a operative system Ubuntu server 14.04.

The beggining of the OpenStack kilo installation with Legacy architecture, the step-by-step guide was followed, installing the different repositories on both nodes (Controller and Compute) and services such as: (database, queue messaging and NTP server) In the Controller, the different Openstack APIs (Keystone, Glance, Nova and Horizon) were also installed in the controller and in the compute APIs of Nova and Nova network.

At the end of the installation, the findings were documented and the different tests were performed, such as creation, programming and removal of instances, and everything works without any novelty.

The advances were presented in the architecture design table, primary groups of Infrastructure platform and primary group of operations management.

7.3.2 Prototype

After the presentation in the different technology environments in South America, we proceeded to start the implementation of the prototype, for it, was realized meetings were held with the different areas in charge of Infrastructure, networks and security. For the prototype HP blade were assigned 4 servers, then the physical characteristics of the infrastructure will be shown.

7.3.2.1 Hardware Specifications

- **Controller Node**

This node was deployed on an HP ProLiant BL465c G1 blade server located in the company datacenter with CentOS 7 x86_64 as base operating system and accounts with the following features:

- Processor: Dual Core AMD Opteron, 2800 MHz of 2 processors with 2 cores each.
- Network interfaces: 2
- RAM memory: 16 GB
- Storage: 250 GB

- **Compute node**

This node was deployed on an HP ProLiant BL460c G7 blade server located in the company datacenter with CentOS 7 x86_64 as base operating system and accounts with the following features:

- Processor: Intel (R) Xeon (R) CPU X5650 @ 2.67 GHz of 1 processor with 12 cores.
- Network interfaces: 2
- RAM memory: 16 GB
- Storage: 250 GB
- Intel VT-x virtualization support

- **Compute Node 1**

This node was deployed on an HP ProLiant BL460c G7 blade server in the company datacenter with CentOS 7 x86_64 as base operating system and accounts with the following features:

- Processor: Intel (R) Xeon (R) CPU X5650 @ 2.67 GHz of 1 processor with 12 cores.
- Network interfaces: 2
- RAM memory: 16 GB
- Storage: 250 GB
- Intel VT-x virtualization support

- **Compute Node 2**

This node was deployed on an HP ProLiant BL460c G8 blade server located in the enterprise datacenter with CentOS 7 x86_64 as base operating system and accounts with the following features:

- Processor: Intel (R) Xeon (R) CPU E5-2609 V2 @ 2.67 GHz of 1 processor with 8 cores.
- Network interfaces: 2
- RAM memory: 28 GB
- Storage: 250 GB
- Intel VT-x virtualization support

7.3.2.2 Topology

In figure 7.22 shows the network topology and VLAN configuration for the prototype.

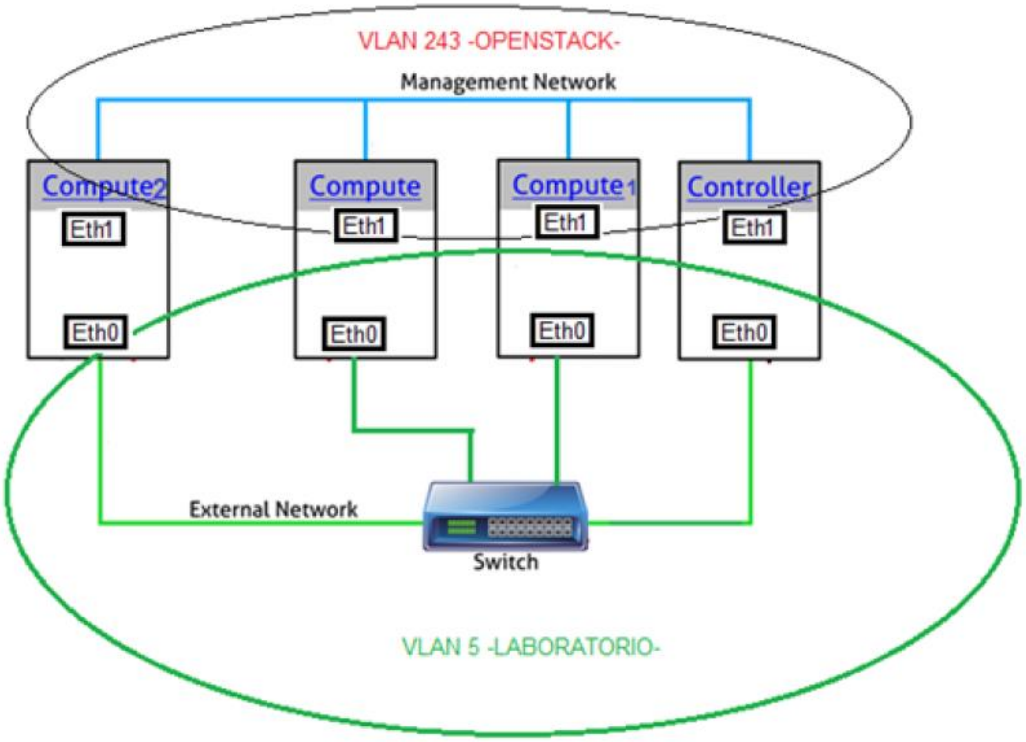

Figure 7.22. Network Topology

7.3.3 OpenStack installations.

For the installation of openstack kilo is used the "official guide for the version for CenOS 7" [15], Are verified the services that are installed in each node and the structure that is used to make these services functional. In particular, will be installed the architecture with Legacy networking (nova-network), reflected in the figure 7.23.

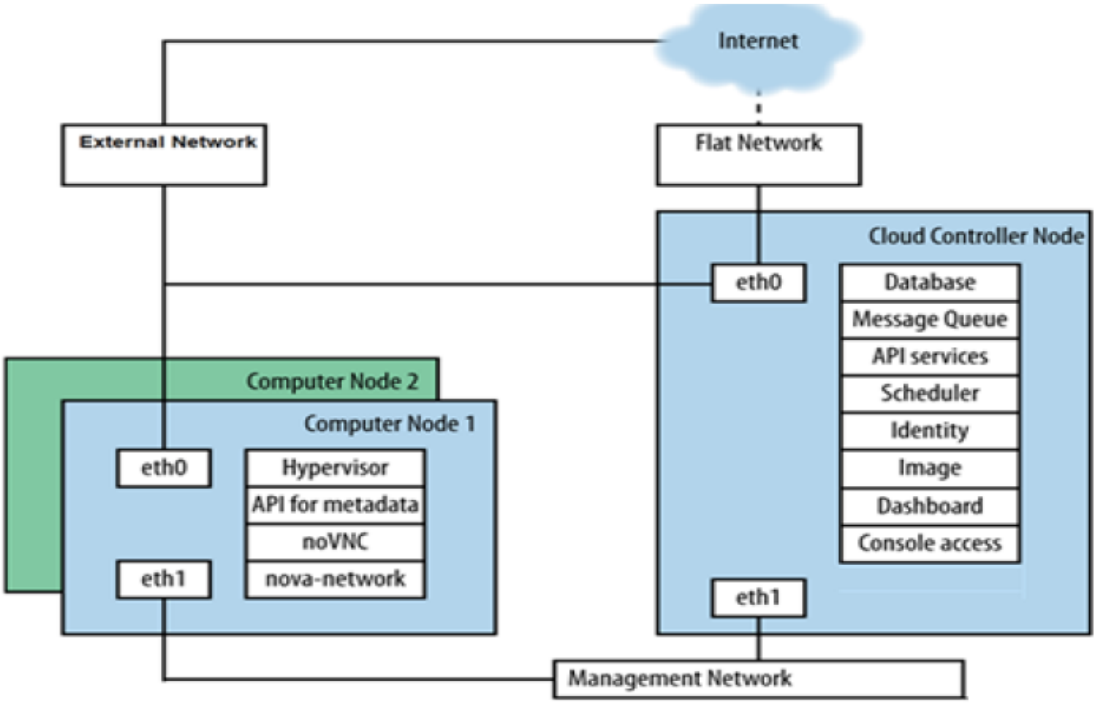

Figure 7.23. Installation architecture Prototype SURACLOUD

The architecture manages at least two nodes, the controller that implements the basic services of Keystone, Glance, Nova, Horizan, MariadB and RabbitMQ. The compute node implements Nova hypervisor (KVM) and Nova networking.

If this document is to be used as an installation manual, it is advisable to have the official guide at hand, because only important parameters and findings will be appended during the installation.

- **Basic configuration**

As a first step, of the machines, it is necessary to make some previous steps where they stop and deactivate processes like the network administrator and firewall.

To stop and disable the network administrator, the following commands are executed "*systemctl stop NetworkManager.service*" y "*systemctl disable NetworkManager.service*"

To stop and disable the firewall, the following commands are executed "*systemctl stop firewalld.service*" y "*systemctl disable firewalld.service*"

Notes:

- *Make sure the network interface be activated on promiscuous mode.*

- *The distribution of CentOS 7 allows by default restrictive policies of firewall. During the installation process, certain steps will fail unless modify or disable the firewall.*

- *Reconfiguring Network Interfaces will interrupt network connectivity. It is advisable to use a local terminal session or Hilo administration for these procedures.*

- *Some distributions add an entry in the / etc / hosts archive that resolves the actual hostname to another loopback IP address as 127.0.1.1. You must comment or delete this entry to avoid name resolution problems.*
-

As a second step, the network cards of each node must be configured. All nodes must implement the NTP protocol. Figure 7.24 shows the network configuration.

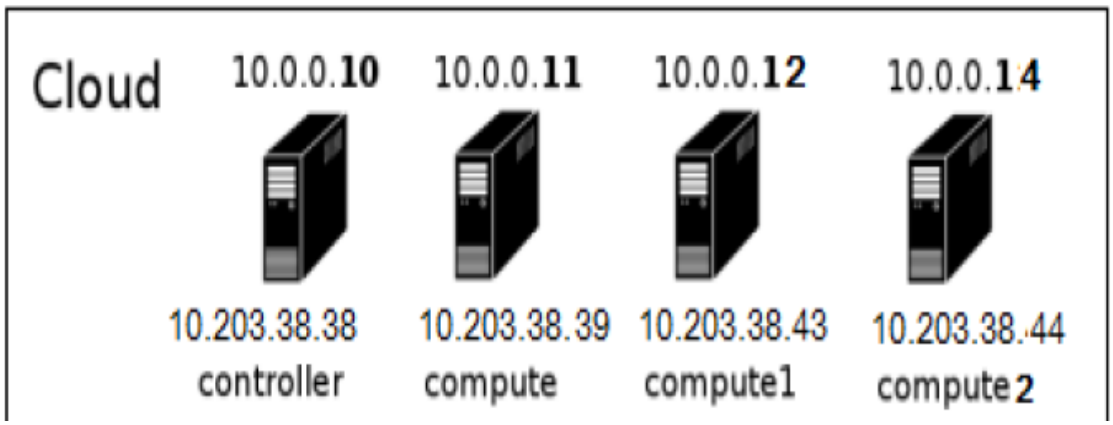

Figure 7.24 Network configuration on the controller node and compute nodes.

Notes:

- It is important to realize ping against the same machines, the other nodes and the internet (controller, compute, compute1, compute2 and openstack.org).

- In case the internet has disabled the ping to public domains, the wget command was used.

- For the synchronization process, the controller queries the local NTP server of the company and the compute nodes consult the controller. This configuration was done because: First, the nearest NTP server is the local one, and second, because the controller is not always going to have an internet connection.

- When the NTP service is restarted, the synchronization can take up to 3 minutes, to perform manual synchronization of the servers or desktop computers you can run the ntpdate command with the IP of the NTP server or the host name that can be successfully resolved.

As a third step, the installation of OpenStack packages was performed on all nodes (controller, ncompute)

The installation of the repositories was done with the following commands:

```
# yum install http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-5.noarch.rpm

# yum install http://rdo.fedorapeople.org/openstack-kilo/rdo-release-kilo.rpm

# yum upgrade
```

Automatic updates were disabled with the following command:

```
# ntsysv
```

Once inside the interface, the services are disabled.

- **Note:** By recommendation automatic centos 7 updates are disabled to prevent damage to the openstack kilo environment.

Many of the openstack services require a database to store information. In this installation the database that has been used MariaDB. And the installation was done in controller node.

The database was installed with the command:

```
# yum install mariadb mariadb-server MySQL-python
```

The installation guide explains how the file is modified "*/etc/my.cnf.d/mariadb_openstack.cnf*" To give access to the compute nodes and to modify the specific characteristics of the database like InnoDB, UTF-8, etc.

In the controller node, the queue messaging (Rabbitmq) must be installed, with the following command: "yum install rabbitmq-server". From now on, can already be install the OpenStack services.

- **Note:** *It is important to the connection between the nodes do not fail, to be written in the configuration file of each service, the assigned password of Rabbitmq, otherwise the system will not communicate correctly.*

7.3.3.1 Configuration of the identity service (Keystone) in the controller node

The service has the function of management, tracking and permissions for users, and offers the catalog of services. In Figure 7.25, it shows how the Keystone Identity Manager works.

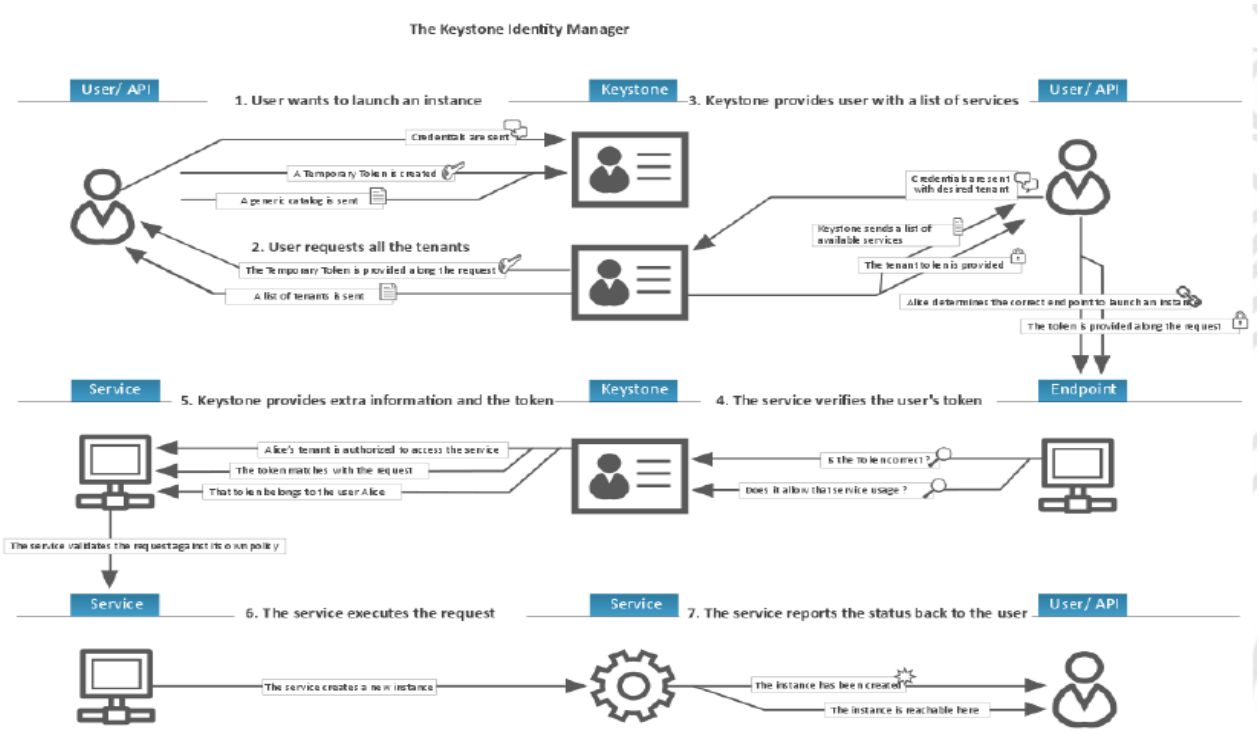

Figure 7.25. Keystone Identity Manager Operation. Taken from [9]

The installation of Keystone was done with the following command "yum install openstack-keystone httpd mod_wsgi python-openstackclient memcached python-memcached" And uses the database to store information. The file is modified "/etc/keystone/keystone.conf" To enter the configuration parameters and enter the passwords.

In addition, the Keystone database must be created with the following command:

```
# mysql -u root -p mysql> CREATE DATABASE keystone;
mysql> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' \
IDENTIFIED BY 'KEYSTONE_DBPASS';
mysql> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' \ IDENTIFIED BY
'KEYSTONE_DBPASS';
```

To use the different services, we must have the authorization, they must create a script with the services exported, for example, save them in a file named adminrc.sh and then run with the command source adminrc.sh. The content of adminrc.sh may be similar to the following:

```
export OS_PROJECT_DOMAIN_ID=default
export OS_USER_DOMAIN_ID=default
export OS_PROJECT_NAME=admin
export OS_TENANT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=ADMIN_PASS
export OS_AUTH_URL=http://controller:35357/v3
```

Note:

- To have better authentication performance, the Keystone services are deactivated and deployed on an Apache HTTP service to handle requests to Keystone on ports 5000 and 35357.

- If during the installation the following error occurs: -Openstack an unexpected error prevented the server from fulfilling your request. (HTTP 500) - check the database.

7.3.3.2 Configuration of the image service (Glance) in the controller node.

Image Service allows users to discover, register and retrieve images from virtual machines using the glance-api component. And with the glance-registry component, it stores, records and retrieves metadata information about the images.

Glance is called the project and uses a REST API to perform queries on the metadata of the images.

The installation of Glance was done with the following command *"yum install openstack-glance python-glance python-glanceclient"*. In addition, the files must be edited *"/etc/glance/glanceapi.conf"* and *"/etc/glance/glance-registry.conf"* To enter the different configurations and modify the password.

You must create the database for the glance service, the table, the user, and realize the authentication process against Keystone. All of these steps are fully explained in the OpenStack Installation Guide on CentOS 7 (OpenStack Installation Guide, s.f.). The images support different formats like qcow2, raw, vdi and vmdk, for more information consult [5].

7.3.3.3 Configuration of the compute service on the controller node.

The Compute service is the main part of the IaaS type deployment model. Interacts with Identity Service for authentication, Image Service for images, and Dashboard for infrastructure management.

Definitely, Compute is a collection of services that enables the launch of instances. To perform the installation, the following command is executed:

```
# yum install openstack-nova-api openstack-nova-cert openstack-nova-conductor \
  openstack-nova-console openstack-nova-novncproxy openstack-nova-scheduler \
  python-novaclient
```

The "/etc/nova/nova.conf" file must be edited to enter the different configurations and modify the service password. In addition, a database must be created for the service. To see all the steps of the installation, consult the reference [15].

7.3.3.4 Configuration of the compute service on the compute node.

Configuring the compute service on the compute nodes. It is convenient to install other nodes as Compute node. In this way it can be scaled horizontally in a simple way. The installation of the Compute service was performed with the following command "yum install openstack-new-compute sysfsutils". In addition, the file "/etc/nova/nova.conf" must be edited to enter the different configurations and modify the password.

Note:

- *Replace MANAGEMENT_INTERFACE_IP_ADDRESS with the IP address of the network management interface of the computation node example 10.0.0.12 or 10.0.0.13 or 10.0.0.14.*

- *If the web browser to access the remote consoles resides in a host that can not resolve the controller by the DNS, it must replace in the controller with the IP address of the management interface of the controller node in novncproxy_base_url. Example: novncproxy_base_url = http://10.0.0.11:6080/vnc_auto.html.*

It is convenient to determine if the processors of the compute nodes are compatible with the acceleration of the virtual machines, for this the following command is executed: *"egrep -c '(vmx|svm)' /proc/cpuinfo"*. This way if the command returns one or more means that it is compatible with hardware acceleration. If on the other hand it returns a zero, restart the server and enter the server BIOS and activate the VIRTUALIZE INTEL VT-x / EPT parameter, save the changes and enter the command again to obtain the expected results.

Note:

- *If the compute node does not compatible with hardware acceleration, it should be configured in the QEMU library instead of KVM.*

- *For the installation of the different hypervisors you can consult the "reliable openstack" [8].*

7.3.3.5 Configuration of the network service (legacy networking) on the controller node.

The installation of Legacy networking mainly involves compute nodes. However, the file "/etc/keystone/keystone.conf" is modified to enter the configuration parameters for network management.

7.3.3.6 Configuration of the network service (legacy networking) on the compute nodes.

The installation of the legacy network service was done with the following command "yum install openstack-nova-network openstack-nova-api". In addition, the /etc/nova/nova.conf file must be edited to enter the differents legacy networking configurations.

Note:

- *For proper operation of the network parameters replace INTERFACE_NAME with the actual interface name or Br100.*

For the creation of networks, it is necessary to enter the controller as administrator and then add the credential with the following command "source admin-openrc.sh". The following steps are performed:

To create the private network, the following command is executed *"nova network-create demo-net --bridge br100 --multi-host T \ --fixed-range-v4 10.0.2.0/24"*

To create the floating network, the following command is executed *"nova-manage floating create --pool=nova --ip_range=10.203.38.224/28"*

To verify the creation of the private and floating network, the following commands are executed "*nova net-list*" y "*nova-manage floating list*" en el nodo controller.

7.3.3.7 Configuration of the Horizon Service on the controller node

Horizon is the web interface that allows to manage the resources and services of OpenStack graphically. Interact with OpenStack Compute through the OpenStack API.

To realize the installation of the horizon service the following command is executed:

```
# yum install openstack-dashboard httpd mod_wsgi memcached python-memcached
```

The following configuration steps can be found in the reference [15].

7.4 Findings and recommendations

7.4.1 Design of Computer Resources (CPU and Memory)

With regard to the design of computational resources, the architecture used in the prototype is a single pool type, which means that the physical compute node that compose the pool may have different hardward characteristics between them, but are grouped into a single pool On which OpenStack distributes virtual machine instances indistinctly.

However, for production installation a multiple pool type design is recommended, which means that the CPU and memory resources of the compute node are grouped into several pools according to the hardware physical characteristics of the servers that conform it.

This design would achieve greater efficiencies in the use of physical resources, since a pool of thin machines of low cost and moderate physical characteristics could be destined to associate to flavors of small virtual machines (like "x-small"), which They would be intended for light workloads (such as cloud desktops); While it could have another pool with a group of physical machines with high-capacity characteristics that are associated with flavors of large machines (such as "x-large") intended for virtual machines for processing-intensive workloads (Such as databases with high batch processing).

This design is also known as "bin packing design", and in addition to improving the efficiency in the distribution of workloads, also ensures separation and isolation between them, allowing to perform specific optimizations on the configuration of each pool (Physical server group) without affecting the others.

On the other hand, in relation to sizing (sizing of hardware needs) it is recommended to use an indicator known as overcommit ratio, it corresponds to the existing reason between the virtual resources that are obtained on the physical resources that are allocated.

$$Overcommit\ Ratio = available\ virtual\ resource / available\ phisical\ resources$$

$$Default\ CPU\ Overcommit\ Ratio = 16:1$$

$$Default\ Memory\ Overcommit\ Ratio = 1.5:1$$

In the observation of the organization's physical infrastructure environment that was performed, it was generally found that the greatest waste of resources is focused on the CPU on physical machines, where it was found that the vast majority use on average only 20% of the Total CPU capacity assigned to the machine, while in memory RAM the consumptions generally range between 60% and 85%.

This means that if you have a total installed physical capacity of 10 CPUs on all the physical machines that fulfill the role of compute node, you can get to take advantage of (and assign in the top of the tenant) a total of $10 * 16 = 1600\ virtual\ CPUs\ (vCPUS)$.

7.4.2 Network topology design

To decide the most appropriate network module between the Nova Legacy and Neutron, the main criterion should be whether it truly require the advanced capabilities that the Neutron module offers that are not available in the Nova Legacy. Well Neutron is a network module too robust, it provides very advanced capabilities that should be chosen if they are going to take advantage of, among these capabilities it stands out that allows for advanced routing and custom creation of subnets and Network segments on demand and automatically, As well as high layer services such as the dynamic load balancer as a service.

In the case of deciding for an installation with the Neutron network module, an additional physical node is required in the deployment architecture, and this node can become a bottleneck in its configuration since, by default, all service traffic of the virtual machine instances of the system is routed through that node to default gateway mode.

In the prototype, the Nova Legacy module was used, and it adequately supplied all the network needs of the test cases, with the same were able to asign ability to assign floating IPs on demand very easily. It is thought that this module is sufficient to cover the great majority of the cases of use of the clouds that have a general purpose; likewise, it was found that it is much easier to configure and manage.

In any case, it is important to take into account that for the intercommunication of the controller node with the compute node it is necessary to have a control network (management), which OpenStack recommends, be a totally independent network, what is suggested is to conform to this Purpose a subnet dedicated so that it can limit the traffic to that domain, because in this network of control is recurrent messages of type multicast between the members of the group of compute nodes and the controller. Therefore, it is suggested that such a network be configured as an independent "back" network with unique physical network interfaces, thereby ensuring that the cloud can scale by adding compute nodes without being limited by collisions and rude from traffic the control network.

Also the external network, which attends traffic between virtual private cloud instances with components external to the same, it is advisable to have it in a subnet or a separate segment. In the case of the prototype, an IP range belonging to the VLAN of the laboratory was used, and the floating IP service offered by OpenStack was configured where it assigned and released IPs of that range on demand, as the Cloud users decided it after they created and removed instances of virtual machines.

Having this network configuration in such instances of virtual machines, bottlenecks in the Nova Legacy network central module (physically located on the compute node) were avoided, since the traffic is directed directly to the default gateway of the laboratory network Without going through the controller's new legacy module.

Finally, OpenStack requires an internal network that it uses to deliver the initial network configuration to the virtual machine instances that are delivered and through which the virtual machines can communicate within the Private Cloud. Subsequently, to said virtual machine the user can assign an IP of the external network (the laboratory in case of the prototype) so that it has communication with the external components to the Cloud.

7.4.3 Design of High Availability Schemes (HA)

For the possible implementation of Private cloud with OpenStack in productive environment, where the concept of High Availability or High Availability (HA) is a fundamental requirement. Openstack can be deployed with at least two or three Controller Nodes in a way that reduces possible single points of failure that affect cloud availability. This kind of implementations are possible due to the Openstack services are communicated through RESTful APIs through HTTP protocol and in AMQP-based messages using the RPC protocol. For this, it is possible to make a configuration with any tool that allows configuration of clusters. For example, the OpenStack

architecture guide refers to the Pacemaker solution [18], which is a scalable and clustered resource manager for high availability, but can actually be implemented with any product that offers the same purpose.

It should be noted that in the case of a fall of the controller node the instances of virtual machines that are already operating from the compute node are not affected in their availability for service to end users, since they run autonomously from the compute nodes, but it will not be able to manage or control them from the OpenStack control layer to perform administrative and control tasks.

In order to provide high-availability schemes at the level of the virtual machine instances that are running on the compute nodes, there are different techniques, of course, the easiest and the most recommended practice is to use the same high availability practices that are used to create a cluster that on the Traditional HA installations on physical machines, since, in the end, such virtual machines can form clusters in the logical layer, in exactly the same way as the physical machines do to the lower layer.However, it is important to ensure that at the physical layer the compute node where each virtual machine is located has sufficient physical resources to support the automatic movement of virtual machines that OpenStack does When a physical compute node suffers a fall or retreat.

Other techniques are to use scripts that invoke the OpenStack API to induce elastic behavior so that when a machine drops from the cluster, a new replacement virtual machine is automatically created from a template with the image from which it was dropped. In this way, the new instance can receive the workload of the virtual machine that underwent the fall, to achieve this automatically one must think of a load balancer that may have the ability to bind and withdraw members to the cluster dynamically in response to Events of the monitoring system that detects the fall.

7.4.4 General

Regarding the choice of hardware for each component that conforms the architecture is suggested at a general level the following configuration:

- **Controller node:** A general-purpose machine with intermediate resources.

- **Networking node:** if it is decided to install the neutron component it is recommended that the machine destined to fulfill this role has 3 different physical network interfaces, so that the service network traffic, system control traffic and outgoing traffic to the internet of the Cloud is performed by independent interfaces.

- **Compute nodes:** For general-purpose clouds is recommend low cost commodity servers that favor horizontal scaling, as previously noted, for productive environments it is highly

recommended a multiple pool configuration having heterogeneity in hardware with coherent correspondences associated with workloads.

7.4.5 Installation

Some findings that were found that are not properly documented in the installation guide or in the official OpenStack Installation Guide (s.f.) [15] are as follows:

- For the installation of the NTP in the controller node it is necessary to verify that the internal firewall of the operating system is not active, that many times by default is enabled, with this is guaranteed that the compute node can communicate with the controller for synchronization and functions of NTP by UDP port 123.
- In the installation of the identity and keystone service, you may receive the error message "Openstack an unexpected error prevented the server from fulfilling your request. (HTTP 500), "this usually occurs because the MySQL database service on the controller may be down or may require some revision.
- To allow access to the instances locally from the compute nodes it is necessary to enter on the file "/etc/sysctl.conf" and enter this parameter "net.ipv4.ip_forward = 1". To implement the changes run the following command "sysctl -p"
- Error in dashboard (horizon) you may get the error message "Something went wrong! An unexpected error has occurred. Try refreshing the page. If this does not fix it, contact your local administrator. "This occurs because there is a conflict of language configuration in the message translation of the dashboard. To fix it, in the browser chrome or firefox, change the language or restore it to the default values.If the problem continues, edit the following file: / etc / openstack-dashboard / local_settings, entering the following lines:
- SESSION_ENGINE = 'django.contrib.sessions.backends.cache'
  AUTH_USER_MODEL = 'openstack_auth.User'

  Afterwards it is necessary to restart the apache service with the following command "systemctl restart httpd.service".
- If the following error appears in the VNC: "Failed to connect to server (code: 1006)" when running the console of the instance, verify that the internal firewall that the Linux distribution brings is disabled.
- Failed to launch instance in Legacy architecture: "Code 500-No valid host was found. There are not enough hosts available", Check the "virsh nwfilter-list" command to check the available filters. In that list you should be able to see "no-mac-spoofing" filter, if it is not on the list, you have permission problems. To resolve this issue, you will need to ensure that the following packages are installed on the system:

- libvirt-daemon-config-nwfilter
- libvirt-daemon-driver-nwfilter

To do this, execute the following command “yum install libvirt-daemon-config-nwfilter libvirt-daemon-driver-nwfilter”, it is restart the services with the following command “systemctl restart libvirtd.service openstack-nova-compute.service” And it is run again the option to launch an instance where it will succeed.

- How to resize resources of a hot instance, for this it is necessary to enter in the compute and controller nodes and edit the next line in the file “/etc/nova/nova.conf” And enable the following property "allow_resize_to_same_host = True".
  Then restart the following services:
  In the ecompute nodes “systemctl restart openstack-new-compute”
  In the controller node "systemctl restart openstack-new-api"
- To solve the problem to get out to the external network or internet from an OpenStack instance with Legacy architecture, the following rule is executed in the firewall of the compute node “iptables -t nat -I POSTROUTING -o br100 -j MASQUERADE” This enables the output to external networks or the internet.

Chapter 8

Conclusions

- A Private Cloud initiative under the IaaS model facilitates the rapid delivery of IT infrastructure services, which can induce agility in the delivery cycles of IT solutions offered to the organization, which in turn transitively can induce a reduction in the time of arrival of business services to the market.

- There are several Open Source tools with which it is possible to realize a Private Cloud implementation, among them are: OpenStack, CloudStack, Eucalyptus and OpenNebula, these tools are used by cloud providers.

- The Open Source Private Cloud tool with IaaS model that best fits for the implementation in the organization, is OpenStack, is the most mature tool with respect to other solutions, since with the results obtained from the implemented prototype, it was possible to test agility and speed for the provision of computing needs.

- The choice of the OpenStack deployment architecture requires an appropriate design at the level of computational resources, network topology and high availability schemes, aspects that must be observed in light of the needs that are required to be supplied.

- The OpenStack installation process has some level of complexity and requires attention to details that are not necessarily documented in the manufacturer's guides.

- The private cloud implemented in the prototype was able to fully comply with the goal of significantly reducing the provisioning time of virtual machine instances with the installed operating system and network configuration.

- The prototype allowed to perceive other benefits additional to those raised in the objectives that can be delivered by the Private Cloud, such as dynamic elasticity and scalability capabilities, self-service and easy access to IT infrastructure services, measured service and the multitenant scheme that enables the use of the infrastructure to be cost efficient in the services that are offered. These capabilities can be exploited to improve the delivery of IT infrastructure services to the organization.

- There were valuable findings that allowed for making design recommendations, which can be taken into account in the possible formulation of a private cloud implementation project in the organization, for a production environment.

Chapter 10

Bibliography

[1] Apache CloudStack Features. (s.f.). Obtenido de CloudStack: https://cloudstack.apache.org/features.html

[2] Architecture. (s.f.). Obtenido de OpenStack: http://docs.openstack.org/kilo/install-guide/install/yum/content/ch_overview.html#architecture_overview

[3] BARKAT, A., DINIZ DOS SANTOS, A., & IKKEN, S. (24 de junio de 2015). OPEN SOURCE SOLUTIONS FOR BUILDING IAAS CLOUDS. Obtenido de Scalable Computing: Practice and Experience: https://www.scpe.org/index.php/scpe/article/view/1089/436

[4] Chapter 5. Scaling. (s.f.). Obtenido de OpenStack: http://docs.openstack.org/openstack-ops/content/scaling.html

[5] Disk and container formats for images. (s.f.). Obtenido de OPenStack: http://docs.openstack.org/image-guide/image-formats.html

[6] Estado del Arte en Soluciones de Virtualización/Sistemas Gestores Cloud. Openstack. (31 de diciembre de 2013). Obtenido de http://openaccess.uoc.edu/webapps/o2/bitstream/10609/27101/14/jlpd1966TFM1213memoria.pdf

[7] Example architecture. (s.f.). Obtenido de OpenStack: http://docs.openstack.org/openstack-ops/content/example_architecture.html

[8] Hypervisors. (s.f.). Obtenido de OpenStack: http://docs.openstack.org/kilo/config-reference/content/section_compute-hypervisors.html

[9] Identity concepts. (s.f.). Obtenido de OpenStack: http://docs.openstack.org/kilo/install-guide/install/yum/content/keystone-concepts.html

[10] Infrastructure as a Service (IaaS). (s.f.). Obtenido de gartner: http://www.gartner.com/it-glossary/infrastructure-as-a-service-iaas

[11] IT Glossary. (s.f.). Obtenido de Gartner: http://www.gartner.com/it-glossary/it-services
Mahmood, Z. (2014). Continued Rise of the Cloud: Advances and Trends in Cloud Computing. London: Springer.

[12] Martín, J. (11 de septiembre de 2013). Cloud, Open Source y Startups. Obtenido de Cenatic: http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=818:cloud-open-source-y-startups&catid=108:blog-cenatic&Itemid=150

[13] Mell, P., & Grance, T. (septiembre de 2011). The NIST Definition of Cloud Computing. Obtenido de nist: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[14] OPENNEBULA KEY FEATURES. (s.f.). Obtenido de Opennebula: http://opennebula.org/about/key-features/

[15] OpenStack Installation Guide. (s.f.). Obtenido de OpenStack: http://docs.openstack.org/kilo/install-guide/install/yum/content/

[16] Operations Guide. (s.f.). Obtenido de OpenStack: http://docs.openstack.org/openstack-ops/content/

[17] Overview. (s.f.). Obtenido de OPenStack: http://docs.openstack.org/icehouse/install-guide/install/yum/content/ch_overview.html

[18] Pacemaker cluster stack. (s.f.). Obtenido de OpenStack: http://docs.openstack.org/ha-guide/controller-ha-pacemaker.html

[19] Plummer, D., Bittman, T., Austin, T., Cearley, D., & Smith , D. M. (17 de junio de 2008). Cloud Computing: Defining and Describing an Emerging. Obtenido de Gartner: http://cmapspublic3.ihmc.us/rid=1JZJDNJXW-174MCCX-5CJ/Gartner_cloud_computing_defining.pdf

[20] ReleaseNotes/Liberty. (s.f.). Obtenido de openstack: https://wiki.openstack.org/wiki/ReleaseNotes/Liberty

[21] Releases. (s.f.). Obtenido de openstack: https://wiki.openstack.org/wiki/Releases

[22] Scarfone, K., Souppaya, M., & Hoffman, P. (enero de 2011). Guide to Security for Full Virtualization Technologies. Obtenido de National Institute of Standards and Technology: http://csrc.nist.gov/publications/nistpubs/800-125/SP800-125-final.pdf

[23] Serrano, N., Gallardo, G., & Hernantes, J. (abril de 2015). Infrastructure as a Service and Cloud Technologies. Obtenido de computer: http://www.computer.org/csdl/mags/so/2015/02/mso2015020030.pdf

[24] Software. (s.f.). Obtenido de openstack: http://www.openstack.org/software/

[25] Software as a Service (SaaS). (s.f.). Obtenido de gartner: http://www.gartner.com/it-glossary/software-as-a-service-saas/

[26] The Business Value of Cloud Computing. (s.f.). Obtenido de Cloudonomics: http://www.cloudonomics.com/
What is Virtualization? (s.f.). Obtenido de redhat: http://www.redhat.com/f/pdf/virtualization/gunner_virtual_paper2.pdf

Glossary

Ansible: Is the easiest way to automate IT applications and infrastructure.

Apache 2.0: The Apache License is a free software license that allows the user to use the software for any purpose, distribute, modify, and distribute modified versions of that software.

Authentication: act of confirming the identification of a user.

API (Application Programming Interface): is a set of functions and procedures that fulfill one or many functions in order to be used by other software.

AIX: is a UNIX® operating system based on open standards which let's run the applications that want and on the hardware that choose, servers based on the IBM UNIX operating system.

AWS: (Amazon Web Services) is a collection of cloud computing services (also called web services) that together form a cloud computing platform offered through the Internet

Back-end: it's the back that somehow is not visible to the user since it's not about design, or graphics elements, it's about programming the functions that a site will have.

BareMetal: refers to the underlying physical architecture of a computer. Running an operating system in a bare-metal is another way of referring to running an unmodified version of an operating system on physical hardware.

Blade: Server is an architecture that has managed to integrate in a card all the typical elements of a server. These cards (blades) are inserted into the backplane inside a chassis which in turn integrates and allows sharing common elements such as ventilation, network switches, power, etc.

Bridge: is the action taken by the team to create a global network of two or more communication networks, or two or more network segments.

Chef: is a DevOps tool that provides a framework for Automate and manage the infrastructure.

CLI (Command Line Interface): A utility that provides an alternate method for running ESM commands in UNIX and Windows server environments.

Cloud Bursting: is related to the event of when a cloud surpasses the demand peaks of resources in a private cloud, exceeding in this way the available capacity of the cloud. When this is done the same can make use of external resources of any other cloud: public or private. Once the demand has been satisfied the cloud will return to its original typology as a private cloud.

Credentials: data known only for the user who proves that it is him.

CISC (Reduced Instruction Set Computer): A type of computer architectures that promotes small and simple sets of instructions that may take little time to execute.

CPU (Central Processing Unit): is the part of a computer in which are the elements that serve to process data.

Daemons: special type of non-interactive computer process, that is, it runs in the background instead of being directly controlled by the user. This type of programs continues in the system, that is, it can be executed persistently or reset if try to kill the process depending on the configuration of the daemon and system policies. The word daemon comes from the acronym D.A.E.MON (Disk And Execution Monitor).

DNS (Domain Name System): is a system to assign names to computers and network services that is organized in a hierarchy of domains.

Datacenter: a data center is a data processing center, an installation used to host an information system of associated components such as telecommunications and storage systems.

DHCP (Discover Host Configuration Protocol): is a client-server protocol that automatically provides an Internet Protocol (IP) host with its IP address and other configuration information, related such as the default gateway and subnet mask.

EC2 (Elastic Cloud Computing): is a web service which provides computing capacity in the cloud. It is designed to ease to the developers scalable computing resources based in the web.

ERP (Enterprice resourse Planning): Corresponds to an integral business management system that is designed to model and automate most processes in the company (accounting area, finance, commercial, logistics, production, etc.).

Endpoint: accessible network address, usually described by a URL, where the service can be accessed.

Erlang: A concurrent programming language (or concurrency oriented) and an execution system that includes a virtual machine (BEAM) and libraries (OTP).

Framewotk: defined conceptual and technological structure of support, usually with specific software artifacts or modules that can serve as a base for the organization and software development. Typically, it can include support for programs, libraries, and interpreted language, among other tools, to help develop and unit the different components of a project.

File servers: is a type of server that stores and distributes different types of computer files among the clients of a computer network.

Firewall: is a device that works as a firewall between networks, allowing or denying the transmissions from one network to another.

Flat network: is one in which all instances reside on the same network (that can also be shared by hosts).

Flavors: recipe that is defined to the different options of virtual machines available in Openstack

Front-end: part of the software that interacts with the user (s).

FTP: It is a file transfer protocol.

GARNER: is a US company that performing research and analysis for the related computer hardware, software, communications and information technology (IT) industries.

Hipervisor: It is a platform that allows to apply diverse techniques of control of virtualization to use, at the same time, different operating systems (unmodified or modified in the case of paravirtualization) in a same computer. It is an extension of an earlier term, "supervisor," which applied to the kernels of operating systems.

Instance: Clone of an image that creates the demand of the user in one of the nodes of the cloud.

Floating IPs: Public IP address that can be associated with different instances in order to access them from outside.

JSON (JavaScript Object Notation): is a light data exchange format. Reading and writing it is simple for humans, while for machines it is simple to interpret and generate it.

KVM: Virtual machine that takes advantage of the virtualization in the nucleus. It is a solution to implement complete virtualization with Linux.

LDAP (Lightweight Directory Access Protocol): is a standard protocol that allows managing directories, that is, access to information bases of users of a network using TCP / IP protocols.

BSD License: The Berkeley Software Distribution License allows programmers to use, modify and distribute the source code and binary code of the original software program with or without modifications to third parties under open source or commercial licenses.

LOG: is an official register of events during a particular time range.

LXC (Linux Containers): is a virtualization technology at the operating system (OS) level for Linux. LXC allows a physical server to run multiple instances of isolated operating systems known as Virtual Private Servers.

Metadata: The concept of metadata is analogous to the use of indexes to locate objects instead of data. For example, a library uses tabs that specify authors, titles, publishers, and places to search for books. Thus, metadata helps to locate data.

ML2 (Modular Layer 2): is a framework that allows OpenStack networks to simultaneously use the variety of Layer 2 network technologies that are found in real-world data centers.

NAS: is a storage server that can easily connect to a company's network to assist the file server and provide fault-tolerant storage space.

Node: Point of intersection, connection or union of several elements that converge in the same place.

NIST: is the US National Standards Institute and its mission is to develop and promote measurements, standards and technology to increase productivity, facilitate trade and improve quality of life.

Middleware: defined as a software layer that is located or placed between the operating system and the applications of the system

Open Source: is a term that applies to Software distributed under a license that allows to the user access to the source code of the Software, and also allows to study and modify it freely, without restrictions on the use of it.

OVS (Open VSwitch): is open source software, designed to be used as a virtual switch in virtualized server environments.

P-series: IBM server family.

Puppet: is a good tool that can be used to implement new servers, virtual machines or services.

QEMU: processor emulator based on the dynamic translation of binaries (Conversion of the binary code of the source architecture into code understandable by the host architecture). QEMU also has virtualization capabilities within an operating system, whether GNU / Linux, Windows, or any of the Admitted.

RAM (Random Access Memory): Main memory of the computer, where programs and data reside, on which you can run read and write operations.

REST: (Representational State Transfer), is a type of web development architecture that relies entirely on the HTTP standard. REST allows us to create services and applications that can be used by any device or client that understands HTTP, so it is incredibly simpler and more conventional than other alternatives that have been used in the last ten years as SOAP and XML-RPC.

Role: Personality that a user assumes and that this one knows to what specific thing can accede with that roll.

RPC - Remote Procedure Call: Is a protocol that allows a computer program to execute code on another remote machine without having to worry about the communications between them.

Ruby: Interpreted, reflective and object-oriented programming language.

Runtime: execution times.

SAN: is a system of disks that connects to the servers by networks of very high speed (usually fiber channel).

SCRUM: is a process of Agile Methodology that is used to minimize risks during the execution of a project, but in a collaborative way.

Service: Provides one or more endpoints through which users can access to performance operations and resource accesses.

S.O: Operating System