# ADMINISTRACIÓN DE PROTOTIPO INFRAESTRUCTURA DE COMPUTACIÓN EN LA NUBE DEL GRUPO GID-CONUSS CON ÉNFASIS EN LA IMPLEMENTACIÓN DE NUEVOS MÓDULOS DE OPENSTACK

**JULIÁN ANDRÉS MARTÍNEZ RANGEL**
**CESAR DAVID VÁSQUEZ ROMERO**

**UNIVERSIDAD INDUSTRIAL DE SANTANDER**
**FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS**
**ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA**
**BUCARAMANGA**
**2017**

# ADMINISTRACIÓN DE PROTOTIPO INFRAESTRUCTURA DE COMPUTACIÓN EN LA NUBE DEL GRUPO GID-CONUSS CON ÉNFASIS EN LA IMPLEMENTACIÓN DE NUEVOS MÓDULOS DE OPENSTACK

**JULIÁN ANDRÉS MARTÍNEZ RANGEL**
**CESAR DAVID VÁSQUEZ ROMERO**

Trabajo de grado para optar al título de
**INGENIERO DE SISTEMAS**

Director
**Manuel Guillermo Flórez Becerra**
M.Sc. en Informática

**UNIVERSIDAD INDUSTRIAL DE SANTANDER**
**FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS**
**ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA**
**BUCARAMANGA**
**2017**

## DEDICATORIA

*Principalmente a Dios por darme la salud y la sabiduría para realizar este proyecto.*

*A mi madre Rosa Romero Pérez por todos los esfuerzos y sacrificios que hizo para que yo pudiera estudiar, por su ayuda y apoyo incondicional en todas mis decisiones.*

*A mis abuelos Abelardo Romero Gómez y Socorro Pérez Delgado por su apoyo incondicional.*

*A mi Padre Cesar Alfredo Vásquez Fuentes por brindarme apoyo y consejo*

*A mis tíos Luis, Amparo, Esperanza, Margarita, Graciela, Pedro, por apoyarme siempre.*

*A mi Gran amigo y compañero de proyecto, Julián Andrés Martínez Rangel, por Acompañarme en este proceso de aprendizaje y por brindarme su ayuda y su conocimiento para superar cada obstáculo y cumplir las metas propuestas.*

*A mi amigo Juan Fernando Jojoa Gómez, por su apoyo incondicional.*

*A Stefhany Rangel, por estar a mi lado y acompañarme en los momentos buenos y malos, por creer en mí y darme ánimos en los momentos que más necesitaba.*

*A mis amigos, Jerson Villamizar, Viviana Torres, Andrea Rueda, Daniel Andrade, Leidy Carrillo, Carlos Peñaloza, Fernando Torres, Rafael Sánchez, Jheferson Gálvez, que me acompañaron en esta etapa de aprendizaje y me brindaron su apoyo y amistad.*

*A mis compañeros de Mac Carnes, por aportar a mi formación personal y brindarme su apoyo.*

*A todos mis amigos, primos y familiares que me acompañaron en este proceso y me apoyaron para cumplir este sueño.*

*A todas las personas que de una u otra manera influyeron para que pueda ser la persona que soy hoy en día y pudiera alcanzar este logro.*

***Cesar David Vásquez Romero***

## DEDICATORIA

*A mis padres Elian y Cecilia, quienes siempre han sido mi ejemplo a seguir, apoyo incondicional y principales motores de mi vida.*

*A mi tía Lucila Rangel quien fue mi segunda madre y donde se encuentre sé que estará orgullosa por este logro alcanzado.*

*A mi compañero de proyecto y gran amigo César David Vásquez, que a través del tiempo se convirtió en mi familia. También por su apoyo durante el desarrollo de este proyecto y por ayudarme a mejorar personal y profesionalmente.*

*A Andrea Torres por ser mi gran apoyo, por estar a mi lado en los buenos y malos momentos de esta etapa de mi vida y por darme buenos consejos que han hecho de mí una mejor persona.*

*A mis familiares que siempre han estado pendientes de mí y han aportado su granito de arena durante el transcurso de mis estudios.*

*A mis amigos Jerson Villamizar, Fernando Torres, Jerson Flores, Rafael Sánchez, Daniel Andrade, Jheferson Galvez, Edward Rueda, Julian Rojas, Andrea Rueda y Leidy Carrillo que me han enseñado el gran valor de la amistad.*

*A mis compañeros de la selección de fútbol sala UIS, con quienes compartí momentos únicos y me enseñaron lo que significa trabajo en equipo.*

***Julián Andrés Martínez Rangel***

## AGRADECIMIENTOS

Al profesor Manuel Guillermo Flórez Becerra, por darnos la oportunidad de pertenecer al grupo de investigación GID-CONUSS y por guiar con gran sabiduría y conocimiento este proyecto

Al grupo GID-CONUSS por permitir adquirir grandes conocimientos de valor agregado y por proporcionar los recursos para la realización de este proyecto.

Al MSc. Emmanuell Díaz Carreño, por brindarnos su ayuda y conocimiento que fue vital para la realización de este proyecto.

A la comunidad de software libre por sus grandes aportes en distintos campos que permiten la puesta en marcha de cualquier proyecto de investigación.

A los docentes de la Escuela de Ingeniería de Sistemas por guiarnos y formarnos como profesionales.

A Jhon Moshé, por brindarnos su ayuda y conocimiento que fue vital para la realización de este proyecto.

A la Escuela de Ingeniería de Sistemas y a la Universidad Industrial de Santander por permitir espacios para la formación de personas con cualidades éticas y profesionales.

**TABLA DE CONTENIDO**

**Pág.**

## LISTA DE IMÁGENES

**Pág.**

## RESUMEN


**TÍTULO**: ADMINISTRACIÓN DE PROTOTIPO INFRAESTRUCTURA DE COMPUTACIÓN EN LA NUBE DEL GRUPO GID-CONUSS CON ÉNFASIS EN LA IMPLEMENTACIÓN DE NUEVOS MÓDULOS DE OPENSTACK*

**AUTORES**: JULIÁN ANDRÉS MARTÍNEZ RANGEL, CESAR DAVID VÁSQUEZ ROMERO**

**PALABRAS CLAVE**: Virtualización, Computación En La Nube, Firewall, Contenedores, Alta Disponibilidad, Infraestructura Como Servicio, Openstack.

**DESCRIPCIÓN**:

La aparición de nuevas tecnologías libres para la creación y administración de nube, tanto privada, pública como hibrida, ha generado un gran avance en la forma en que se puede ofrecer servicios computacionales en entornos de investigación y producción, educativos y empresariales.

Sabiendo esto es necesario estar a la vanguardia de dichas tecnologías para sacar el máximo provecho de los beneficios que estas nos ofrecen y poder integrarlas a la infraestructura actual garantizando continuidad y escalabilidad en la misma.

En este trabajo se presenta la actualización de una infraestructura computación en la nube y modelo de alta disponibilidad, junto con la investigación de nuevas tecnologías buscando una mejora continua en la calidad en el servicio haciendo un mejor uso de los recursos de hardware con los que cuenta el grupo GID-CONUSS.

Con la integración de herramientas como OpenStack que hacen uso del concepto de la virtualización es posible ofrecer un servicio de nube flexible acorde a las necesidades de los usuarios, buscando que cada vez la interacción entre administrador y usuario sea menor.

También se hace la implementación de herramientas que permiten brindar y mejorar la seguridad en la red de la infraestructura, generando confianza en los usuarios que utilizan estos servicios


---

* Trabajo de Grado en la Modalidad de Investigación

** Facultad de Ingenierías Físico Mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director M.Sc. Manuel Guillermo Flórez Becerra

## ABSTRACT

**TITLE**: ADMINISTRATION OF PROTOTYPE COMPUTER INFRASTRUCTURE IN THE CLOUD OF THE GID-CONUSS GROUP WITH EMPHASIS IN THE IMPLEMENTATION OF NEW OPENSTACK MODULES[*]

**AUTHORS**: JULIAN ANDRÉS MARTÍNEZ RANGEL, CESAR DAVID VASQUEZ ROMERO[**]

**KEYWORDS**: Virtualization, Cloud Computing, Firewall, Containers, High Availability, Infrastructure As A Service, Openstack.

**DESCRIPTION**:

The emergence of new free technologies for the creation and administration of cloud, both private, public and hybrid, has generated a great advance in the way in which computational services can be offered in research and production, educational and business environments.

Knowing this, it is necessary to be at the forefront of these technologies to take full advantage of the benefits they offer us and be able to integrate them into the current infrastructure, ensuring continuity and scalability.

This work presents the update of a cloud computing infrastructure and high availability model, together with the investigation of new technologies seeking a continuous improvement in the quality of the service making better use of the hardware resources with which it counts The GID-CONUSS group looking for that every time the interaction between administrator and user is much smaller.

With the integration of tools like OpenStack that make use of the concept of virtualization, it is possible to offer an optimal and flexible cloud service according to the needs of the users.

It also makes the implementation of tools that allow to provide and improve the security in the network of the infrastructure generating confidence in the users who use these services.


[*] Undergraduate final Project, research modality

[**] Physico-Mechanical Engineering Faculty. Systems Engineering Science School. Director M.Sc. Manuel Guillermo Flórez Becerra

# INTRODUCCIÓN

La computación en la nube proporciona tecnología de la información como un servicio a través de internet o una red configurada para este propósito y funciona según lo demande el usuario. Los servicios de cómputo que ofrece esta misma abarcan desde plataformas de desarrollo y aplicaciones completas, almacenamiento, equipos de escritorio virtuales hasta servidores.

Las organizaciones aprovechan esta infraestructura para desarrollar nubes híbridas y privadas que ofrezcan servicios de cómputo. Como parte de la modernización del centro de datos, diferentes entidades académicas y comerciales están implementando la infraestructura habilitada para la nube a fin de ofrecer los beneficios que esta provee de manera completa y efectiva.

De igual manera las aplicaciones de código abierto y software libre, ofrecen una gran oportunidad para que las organizaciones interesadas en la computación en la nube, puedan hacer uso de ella, evaluar el desempeño y facilitar los procesos a un costo muy bajo.

Este trabajo de investigación presentará la actualización de herramientas ya implementadas en proyectos anteriores como lo es Openstack. La cual permite administrar de manera eficiente la nube para proporcionar infraestructura como servicio. Se presentará una versión más reciente, a su vez que se implementarán varios de sus nuevos módulos. De igual manera se buscará mejorar la seguridad de los servicios virtuales y se actualizarán las herramientas usadas en el modelo de alta disponibilidad para las aplicaciones web usadas por la escuela de ingeniería de sistemas de la UIS; todo esto con el fin de ofrecer a los usuarios una mayor calidad en el servicio.

# 1. INFORMACIÓN DEL PROYECTO

## 1.1 DESCRIPCIÓN DEL PROBLEMA

El grupo GID-CONUSS ha venido desarrollando una infraestructura de computación en la nube y modelo de alta disponibilidad con el propósito de brindar servicios computacionales a los estudiantes y profesores de la escuela de ingeniería de sistemas e informática y comunidad universitaria en general. En este punto, teniendo en cuenta el crecimiento y la demanda de servicios computacionales es necesario continuar el estudio de un modelo basado en la tecnología de Openstack, así como la exploración e investigación de nuevas tecnologías de “Cloud Computing” y virtualización que nos permitan avanzar en el desarrollo de este paradigma.

Este proyecto se plantea con el fin de mantener y mejorar la infraestructura actual permitiendo una mayor escalabilidad y ofrecer un mejor servicio a la comunidad universitaria, implementando soluciones de seguridad buscando evitar posibles alteraciones externas a los proyectos y servicios computacionales prestados por el grupo GID-CONUSS

## 1.2 JUSTIFICACIÓN

Ante el crecimiento de la demanda de servicios computacionales, es necesario continuar con la implementación de tecnologías como Openstack y contenedores Linux, que permitan utilizar de manera óptima los recursos de los servidores físicos sin disminuir funcionalidades y rendimiento a los servicios virtualizados.

Teniendo una infraestructura que permite mayor escalabilidad y optimización de los servicios prestados por el grupo GID-CONUSS es necesario brindarle seguridad a esta misma evitando que agentes externos puedan alterar sin

autorización los diferentes proyectos mediante ataques de autenticación por fuerza bruta o a aprovechando vulnerabilidades en la configuración de la red.

Para esto se propone la implementación de un conjunto de herramientas libres que conformarán un firewall para contrarrestar estas situaciones y generar mayor confianza hacia los estudiantes y comunidad universitaria en general que busquen adquirir los servicios prestados por el grupo.

## 1.3 VIABILIDAD

Actualmente el grupo de investigación GID-CONUSS, cuenta con equipos de cómputo de altas prestaciones que nos permiten realizar pruebas e implementar diferentes herramientas generando un entorno de laboratorio para así tomar la decisión de llevarlo al sistema en producción.

El software libre está en constante crecimiento y se cuentan con herramientas de un alto grado de calidad que permiten la implementación de modelos e infraestructuras sólidas y seguras para un sistema de producción tal como con el que se cuenta en este momento a costos muy bajos

De igual forma los integrantes del grupo GID-CONUSS, desde su director hasta los estudiantes que realizan proyectos allí, tienen alto grado de conocimiento en la administración de servidores Linux y herramientas de software libre lo cual permite aportar al mejoramiento y desarrollo de esta infraestructura.

# 2. OBJETIVOS

## 2.1 OBJETIVO GENERAL

Administrar, monitorear y mantener el funcionamiento de la infraestructura de computación en la nube modelo de alta disponibilidad del grupo gid-conuss, con énfasis en la exploración de nuevas tecnologías y otros módulos de openstack como alternativa y/o complemento a la virtualización que permitan prestar un servicio de computación en la nube a los estudiantes y profesores de la escuela de ingeniería de sistemas e informática de la Universidad Industrial de Santander.

## 2.2 OBJETIVOS ESPECÍFICOS

- Monitorear, Administrar, y mantener la infraestructura actual de computación en la nube y modelos de alta disponibilidad del grupo GID-CONUSS.
- Realizar tareas de mantenimiento programadas y copias de seguridad, Mantenimiento y administración de las unidades de almacenamiento.
- Garantizar para los servicios virtualizados su funcionalidad, seguridad, alta disponibilidad, escalabilidad, eficiencia en la nueva infraestructura.
- Recuperación del sistema en caso de fallos.
- Realizar tareas de atención a los usuarios de los diferentes servicios.
  - Registro de las solicitudes de servicios usando formularios del grupo.
  - Comunicación y Asesoría a usuarios del CloudEisi.
  - Implementar servicios virtuales solicitados por la comunidad.
- Continuar la instalación y configuración de los módulos de OPENSTACK, mejorando la infraestructura
- Actualizar y/o mejorar los manuales de administración, automatización de tareas y las políticas de seguridad.

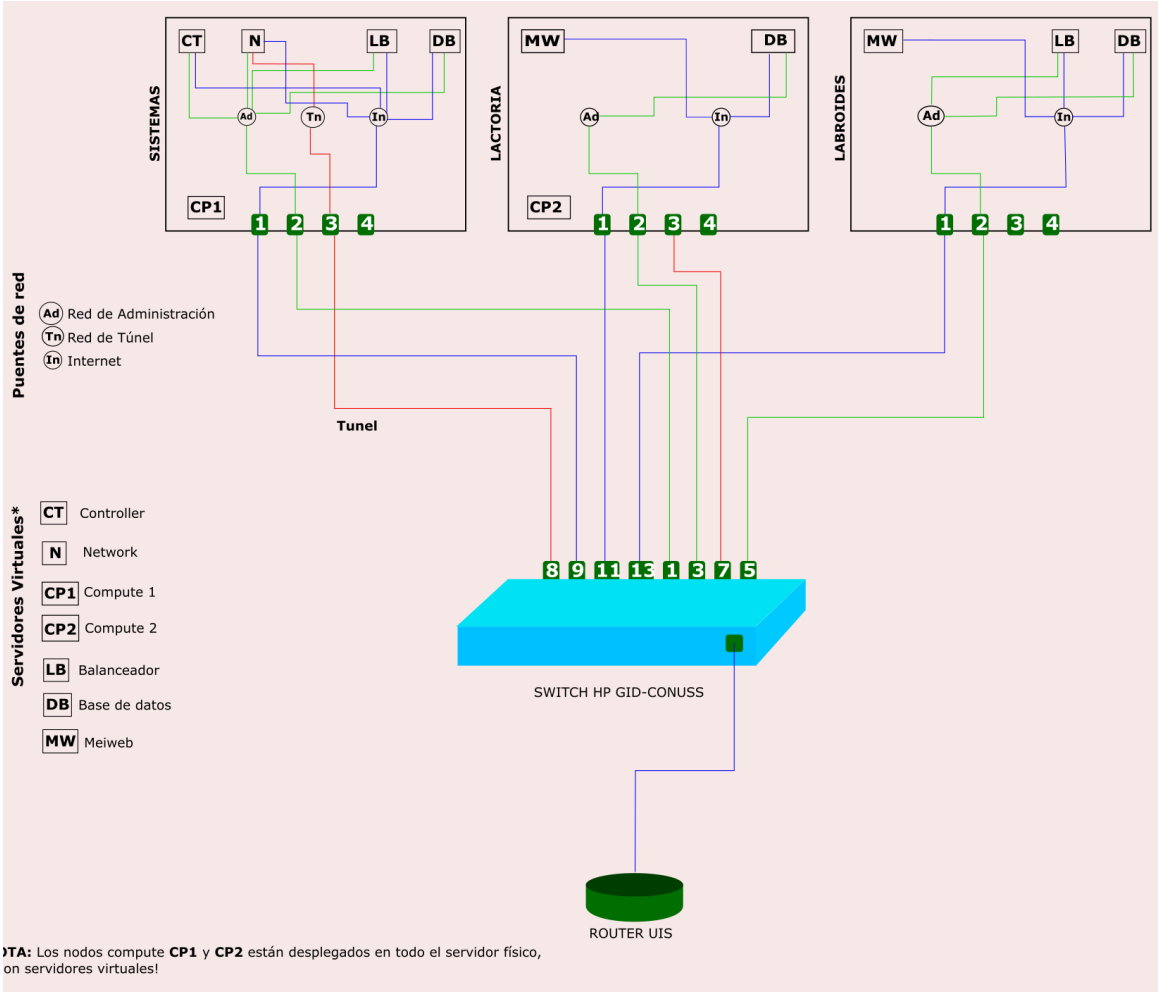
- Exploración de factibilidad para implementar la tecnología de contenedores Linux como alternativa y/o complemento a la virtualización.
- Alta Disponibilidad de datos: Realizar las pruebas del sistema de archivos GlusterFS, como una alternativa al sistema de replicación de datos actual, buscando mayor escalabilidad, para considerar o no la futura implementación en el sistema en producción.
- Seguridad:
  - Mejorar la seguridad de la infraestructura con herramientas dedicadas a este fin, acorde a las recomendaciones proyectos anteriores
  - Implementación de un FireWall UTM para la red de máquinas virtuales de la infraestructura.
- Entrenar y asesorar los relevos administrativos para garantizar la continuidad del proyecto, mediante un curso de mínimo 20 horas durante 5 días.
- Escribir un artículo con los resultados del proyecto y las recomendaciones para próximas investigaciones y desarrollos.

# 3. MARCO TEÓRICO

## 3.1 INFRAESTRUCTURA INICIAL

La infraestructura inicial con la cual se prestaban los servicios de computación en la nube, era la siguiente:

Imagen 1: “Infraestructura inicial”




En esta infraestructura se cuenta con un switch HP 1910, tres servidores físicos, de los cuales dos de ellos cuentan con 24 Gb de memoria RAM, 2 Tb de Disco

duro y un procesador Intel(R) Xeon(R) CPU 2.67GHz con 8 núcleos, el servidor restante posee 32 Gb de memoria RAM, 2 Tb de disco duro y un procesador Intel(R) Xeon(R) CPU 2.67GHz con 24 núcleos. Estos tres servidores tienen instalado el sistema operativo Ubuntu 14.04.

## 3.2 CONCEPTOS BÁSICOS

Para entender mejor la imagen 1 es necesario comprender los siguientes conceptos:

- **Máquina virtual**: Una máquina virtual es un software capaz de emular un ordenador físico.
- **Hipervisor**: monitor de máquinas virtuales, que permite la creación y administración de las mismas.
- **Alta disponibilidad**: se refiere a “la disponibilidad de los recursos en un sistema computacional en el evento de fallo de componentes del sistema. Esto se puede lograr de varias maneras, abarcando todo el espectro que va desde un extremo con soluciones que utilizan hardware personalizado y redundante para garantizar la disponibilidad, hasta otro extremo con herramientas que ofrecen las soluciones de software que utilizan componentes de hardware genéricos”.
- **Replicación de almacenamiento**: es un servicio el cual permite la duplicación de archivos en tiempo real a través de una red.
- **Infraestructura como servicio (IaaS)** [1], [20]: La capacidad de cómputo proporcionada al usuario es el aprovisionamiento de redes, procesamiento, almacenamiento y otros recursos computacionales fundamentales sobre los cuales el cliente puede desplegar y ejecutar aplicaciones según su criterio, lo cual incluye sistemas operativos y aplicaciones. El usuario no administra o controla la infraestructura subyacente, pero tiene control sobre los sistemas

operativos, almacenamiento y aplicaciones desplegadas; y posiblemente un control limitado sobre algunos componentes de red seleccionados (ej. firewall).

- **Openstack** [17]: es un proyecto de computación en la nube libre y de código abierto. Este sistema permite controlar grandes agrupaciones de recursos de computación, almacenamiento y redes a través de un centro de datos, todo administrado a través de un tablero de mandos que da control a los administradores mientras habilita a sus usuarios a proporcionar recursos a través de una interfaz web.

  Openstack cuenta con diversos proyectos que en conjunto permiten el desarrollo y administración de nubes públicas y privadas, estos proyectos tienen un desarrollo modular lo que se pueden ir implementando en el sistema en producción a medida que van saliendo sus actualizaciones estables.

## 3.3 MÓDULOS DE OPENSTACK EXISTENTES EN LA INFRAESTRUCTURA INICIAL

**3.3.1 Servicio de identidad**. El servicio identidad de Openstack proporciona un único punto de integración para gestionar los servicios de autenticación, autorización y servicio de catálogo. Otros servicios de Openstack utilizan el servicio identidad como una API común unificada. Además, los servicios que proporcionan información sobre los usuarios pero que no están incluidos en Openstack (como los servicios LDAP) pueden integrarse en una infraestructura preexistente. Para beneficiarse del servicio identidad, otros servicios Openstack deben colaborar con él. Cuando un servicio Openstack recibe una solicitud de un usuario, comprueba con el servicio identidad si el usuario está autorizado para realizar la solicitud.[1]

[1] Docs.openstack.org. (2016). *OpenStack Docs: Identity service overview*. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_identity.html [Accessed Nov. 2016].

**3.3.2 Servicio de imágenes (glance)**: El servicio de imágenes de Openstack es fundamental para la infraestructura como servicio (IaaS) como se muestra en la arquitectura conceptual. Acepta solicitudes de API para imágenes de disco o servidor y definiciones de metadatos de usuarios finales o componentes de Openstack. También admite el almacenamiento de imágenes de disco en varios tipos de repositorio, incluido Openstack Object Storage. Varios procesos periódicos se ejecutan en el servicio de imágenes de Openstack para admitir el almacenamiento en caché. Los servicios de replicación garantizan la coherencia y la disponibilidad a través del clúster.[2]

**3.3.3 Servicio de cómputo (nova)**: Utilice el servicio de cómputo Openstack para alojar y administrar sistemas de computación en la nube. El servicio de cómputo Openstack es una parte importante de un sistema de Infraestructura como servicio (IaaS). Los módulos principales se implementan en Python. El servicio de computo Openstack interactúa con el servicio identidad de Openstack para la autenticación; el servicio de imágenes Openstack para imágenes de disco; Y el dashboard de Openstack para el usuario y la interfaz administrativa. El acceso a la imagen está limitado por los proyectos y por los usuarios; Las cuotas son limitadas por proyecto (el número de casos, por ejemplo). El servicio de cómputo Openstack puede escalar horizontalmente en hardware estándar y descargar imágenes para iniciar instancias.[3]

**3.3.4 Servicio de red (neutrón)**: el servicio de red Openstack gestiona todas las redes para la infraestructura de red virtual (VNI) y los aspectos de la capa de acceso de la infraestructura de red física (PNI) en el entorno de Openstack. Permite a los inquilinos para crear topologías de red virtuales avanzados,

[2] Docs.openstack.org. (2016). *OpenStack Docs: Image service*. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/glance.html [Recuperado en: Nov. 2016].
[3] OpenStack Docs: Compute service overview", Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_compute.html. [Recuperado en: Nov. 2016].

incluyendo servicios tales como firewalls, equilibradores de carga y redes privadas virtuales (VPN).[4]

**3.3.5 Dashboard (horizon)**: Consiste en un panel de control gráfico, mediante el cual es posible la administración de la mayor parte de la infraestructura

## 3.4 DESCRIPCIÓN DE INFRAESTRUCTURA INICIAL

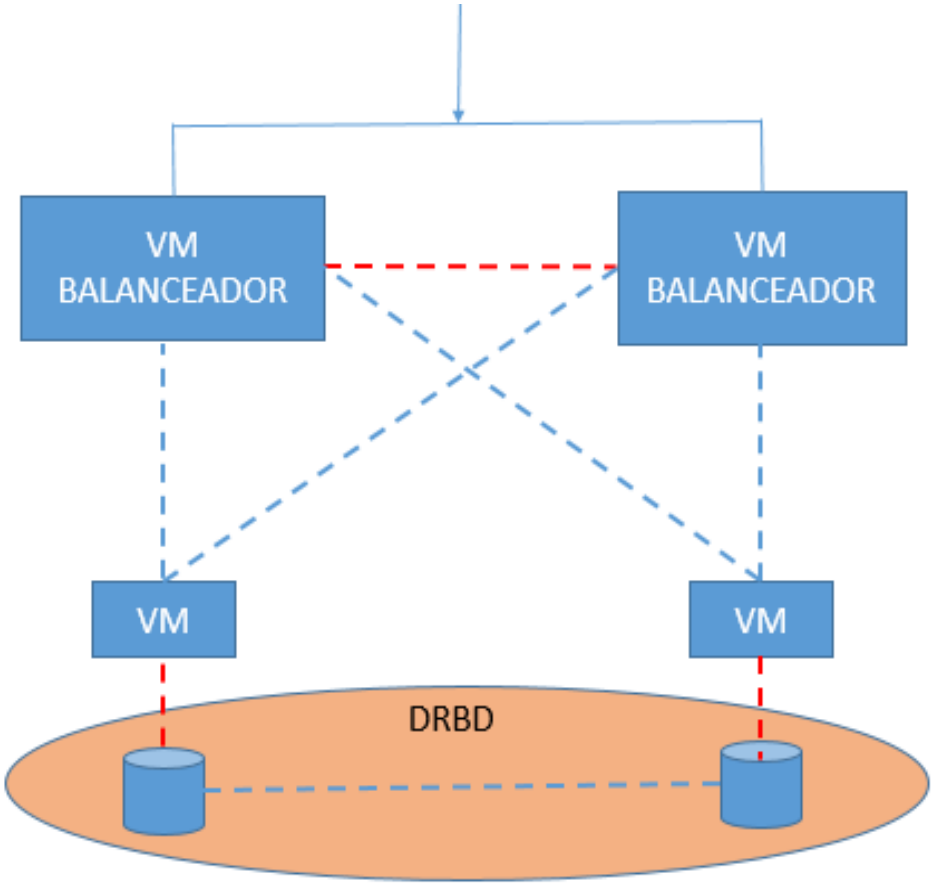
En la arquitectura mostrada en la figura uno se muestra el punto de partida de este proyecto, en esta existe un modelo de computación en la nube conocido con el nombre de infraestructura como servicio, el cual era prestado a través de la adjudicación de servidores o máquinas virtuales a los usuarios, dichas máquinas son creadas y administradas por un hipervisor llamado kvm*.

Esta infraestructura cuenta adicionalmente con un modelo de alta disponibilidad de un servidor web (meiweb) mostrado en la imagen 2

[4] OpenStack Docs: Networking (neutron) concepts", Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/neutron-concepts.html. [Recuperado en: Nov. 2016].

* Kernel Virtual Machine

Imagen 2: “Modelo de alta disponibilidad”



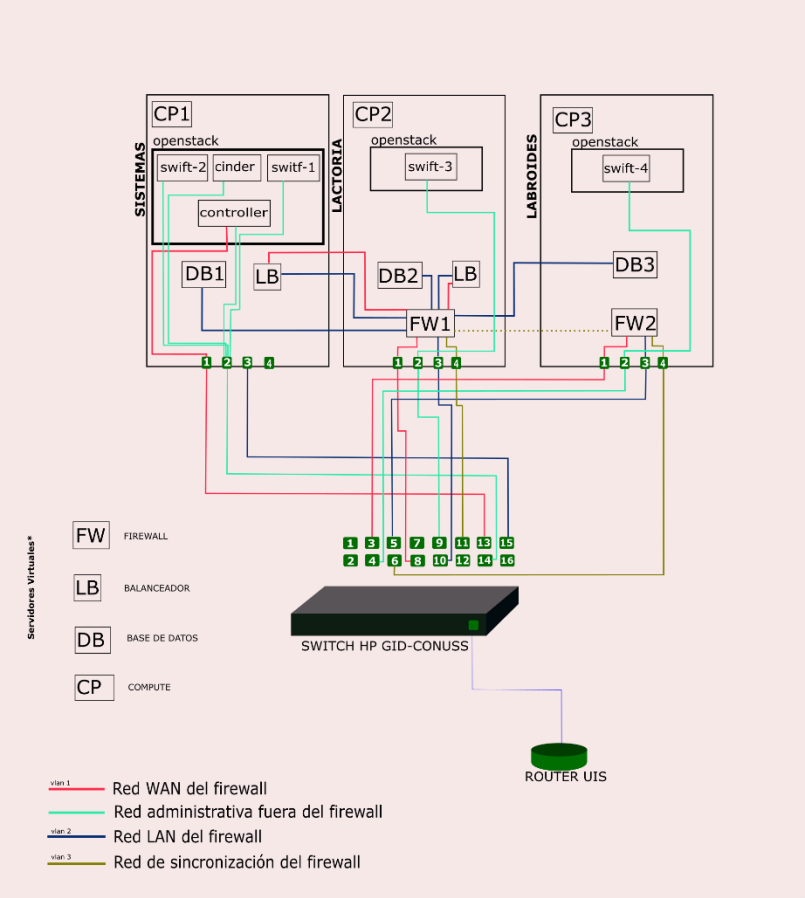

En la imagen podemos observar un modelo de alta disponibilidad, el cual consta de 4 máquinas virtuales dos de las cuales tendrán la función de balancear la carga entrante entre los servidores meiweb.

En conjunto con el servicio de computación en la nube utilizando kvm, se encuentra implementada la plataforma openstack, con sus módulos principales.
Existe una herramienta la cual permite la replicación de datos, la cual se encuentra implementada en la infraestructura.

## 3.5 INFRAESTRUCTURA FINAL

Una vez comprendida la infraestructura inicial, se mostrará el resultado final, con los nuevos componentes.

Imagen 3:" Infraestructura final"




Los componentes hardware con los que se cuenta en la infraestructura final, son los mismos que se utilizaron en la infraestructura inicial.

## 3.6 NUEVOS COMPONENTES DE LA INFRAESTRUCTURA FINAL

**3.6.1 Cortafuegos o Firewall**. Un cortafuegos o firewall es un sistema de defensa que se basa en el hecho de que todo tráfico entrante o saliente a la red, debe pasar de manera obligada por un sistema capaz de autorizar, denegar y tomar nota de todo lo que ocurre, de acuerdo con la política de control de acceso entre redes.

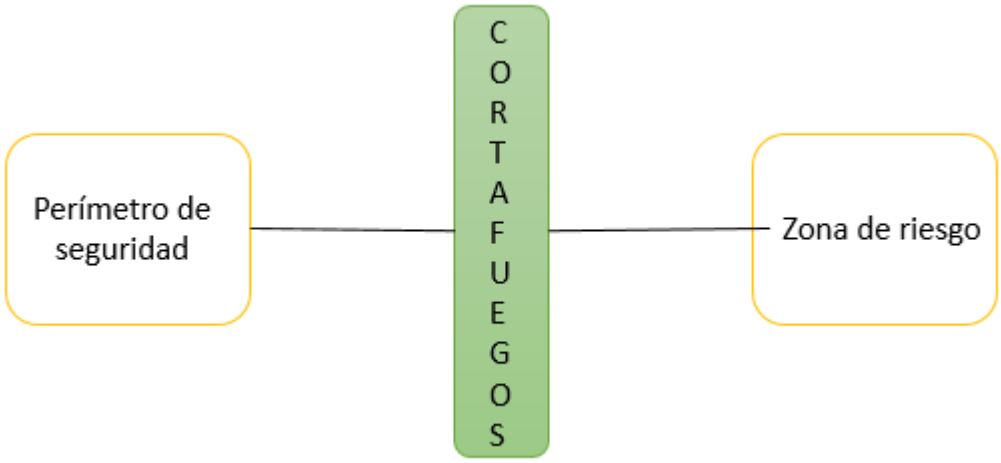
Controla tanto la comunicación desde el exterior como el tráfico generado desde la propia máquina o red interna. Actúa a base de normas que establece el administrador de seguridad o, en su defecto, el administrador de red o el usuario final. Dichas reglas definen las acciones correspondientes a llevar a cabo cuando se recibe un paquete que cumpla unas determinadas características.

A pesar de que hay programas que se venden bajo la denominación de firewall, un firewall NO es un programa, sino que consiste en un conjunto de medidas hardware y software destinadas a asegurar una instalación de red.

En definitiva, se trata de cualquier sistema empleado para "separar" una máquina o una subred del resto, protegiéndola de servicios y protocolos que puedan suponer una amenaza a la seguridad desde el exterior. El espacio protegido por un cortafuegos se denomina **perímetro de seguridad**, mientras que la red externa recibe el nombre de **zona de riesgo**.[5]

[5] Firewalls - definición", Spi1.nisu.org. [En línea]. Disponible en: http://spi1.nisu.org/recop/al01/rmoreno/definicion.html [Recuperado en: Dic. 2016].

Imagen 4: “Modelo básico de un firewall”




**3.6.1.1 PFSENSE**. El software pfSense es una distribución personalizada gratuita de FreeBSD diseñada específicamente para su uso como cortafuegos y enrutador que se gestiona completamente a través de la interfaz web.

Esta herramienta, incluye una larga lista de características relacionadas y un sistema de paquetes que permite una mayor capacidad de expansión.

Para poder la implementación del firewall PfSense es importante tener en cuenta los siguientes conceptos:

- **Nodo:** máquina o servidor virtual.
- **Pfsync** es un protocolo usado para sincronizar estados de cortafuegos entre máquinas que ejecutan el filtro de paquetes.[6]
- **Carp failover:** El protocolo de redundancia de direcciones comunes (CARP, Common Address Redundancy Protocol) por sus siglas en inglés es utilizado

[6] https://en.wikipedia.org/wiki/Pfsync

por múltiples nodos para "compartir" una dirección IP virtual de tal manera que, si el nodo principal o master falla, otro se hará cargo de forma transparente.[7]

- **Sincronización de estado:** La sincronización de estado es manejada por pfsync que transmite información de tabla de estado (contenidos, inserciones, eliminaciones, etc.) a otros nodos en una interfaz privada compartida para que los estados de conexión sean válidos en todos los nodos. De esta forma, si un nodo falla y otro se hace cargo, las conexiones de usuario continúan fluyendo y los clientes no necesitan volver a conectarse.
- **Sincronización de configuración:** Es manejada por XMLRPC Sync, que es un mecanismo de PfSense que permite transmitir determinada información de configuración y comandos entre nodos. Normalmente, el nodo primario sincronizará sus áreas de configuración con un nodo secundario.

**3.6.2 Firewalls UTM (Unified Threat Management)**. Los firewalls UTM por hardware o software conforman un grupo de herramientas o servicios que trabajan conjuntamente para brindar seguridad de frontera a una infraestructura de computación; entre los servicios más relevantes se encuentran los siguientes:

- Función de un firewall de inspección de paquetes
- Función de VPN
- DNS
- Monitoreo de red
- Filtrado de contenidos (para el bloqueo de sitios no permitidos mediante categorías)
- Antivirus de perímetro (evitar la infección de virus informáticos en computadoras clientes y servidores)
- Detección/Prevención de Intrusos (IDS/IPS)

[7] https://en.wikipedia.org/wiki/Common_Address_Redundancy_Protocol

Unas de las principales herramientas UTM de software libre son las siguientes:

### 3.6.3 Herramientas y Complementos del Firewall

**3.6.3.1 Ntop-ng**. Es una herramienta que permite monitorizar en tiempo real una red. Es útil para controlar los usuarios y aplicaciones que están consumiendo recursos de red en un instante concreto y para ayudarnos a detectar malas configuraciones de algún equipo, (facilitando la tarea ya que gráficamente nos muestra una advertencia del error que esté ocurriendo. Posee una interfaz web desde el que cualquier usuario con acceso puede ver las estadísticas del monitoreo.[8] Esta herramienta se integra con PfSense para complementar la seguridad de la red.

Imagen 5: “Ntop”


http://www.ntop.org/wp-content/uploads/2015/05/ntop.png)

**3.6.4 Cambios y Nuevas Implementaciones en Openstack**. Para este proyecto teniendo en cuenta el crecimiento de Openstack y la madurez que ha alcanzado permitiendo la implementación de nuevos módulos para ofrecer un mejor servicio y abarcar más necesidades de computación en la nube actualizamos Openstack a la versión MITAKA para ofrecer una mayor calidad, seguridad y confiabilidad en los servicios ofrecidos. A su vez también se busca corregir errores que se presentaban en las anteriores versiones.

[8] “Ntop", Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Ntop. [Recuperado en: Dic- 2016].

Cabe resaltar que al hacer la actualización de la versión de Openstack, se tuvo que implementar de cero todos los módulos que estaban en la versión anterior.

Una vez hecho esto se implementaron los nuevos módulos.

### 3.6.5 Características de la Nueva Versión

**3.6.5.1 Escalabilidad**. El motor de convergencia de Heat (servicio de orquestación), que apareció por primera vez en el lanzamiento de la versión Liberty, ahora puede manejar cargas más grandes y acciones más complejas para la escala horizontal y ofrecer un mejor rendimiento para el modo sin estado. Los tokens fernet* de Keystone ahora manejan más acciones y solicitudes de autenticación. Los desarrolladores también hicieron un progreso significativo en Cells v2**, una característica que se introdujo en la versión Liberty.

**3.6.5.2 Experiencia de Usuario**. La comunidad se dedica a mejorar la experiencia del usuario de la nube, no sólo el operador de la nube, sino también el usuario final de la nube, es decir, el desarrollador o el implementador de la aplicación. Openstack Client proporciona un conjunto coherente de llamadas para crear recursos para que los usuarios finales no tengan que aprender las complejidades de cada API de servicio. Mitaka también ofrece soporte mejorado para kits de desarrollo de software (SDK) en varios idiomas. Neutron introdujo "give me a network", que consolida el proceso de crear una red, adjuntar un servidor a ella, asignar un IP a ese servidor, y hacer que la red sea accesible, en una sola acción.

**3.6.5.3 Administración**. Numerosos avances se centran en mejorar la facilidad de uso cotidiana para los desplegadores y administradores de la nube. Una configuración simplificada para el servicio de computación de Nova introduce

* Formato de token implementado por Openstack, para realizar procesos de autenticación de una forma más eficiente

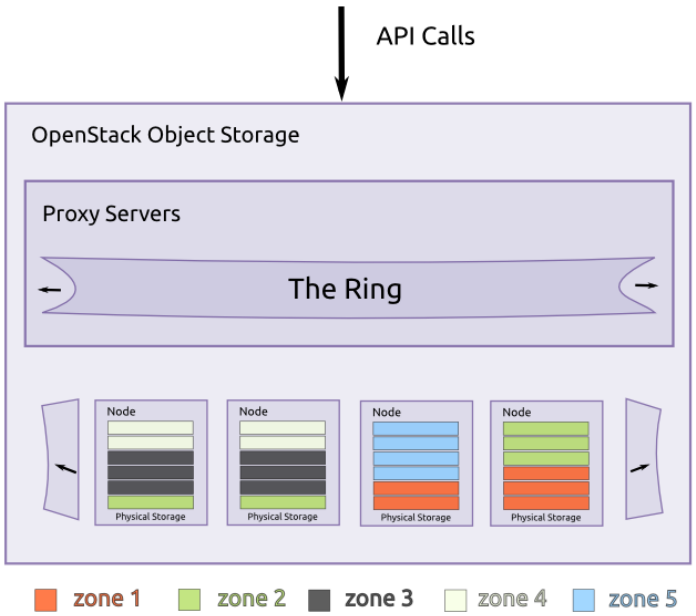
** Arquitectura que permite la escalabilidad de nodos compute para la ejecución del servicio nova

valores predeterminados estándar adicionales y reduce el número de opciones que deben seleccionarse manualmente. El servicio de identidad de Keystone se ha simplificado desde un proceso de varios pasos para configurar las funciones de administración de identidades de una red en la nube: instalar, ejecutar, autenticar, distribuir tokens, etc., a un proceso de un solo paso. Neutron mejora la capacidad de gestión a través de la mejora de la red de Capa 3 y el soporte de Virtual Router (DVR)

### 3.6.6 Nuevos Módulos de Openstack Implementados

**3.6.6.1 Servicio de Almacenamiento de Objetos (swift)**. [17] El servicio de almacenamiento de objetos Openstack es un sistema de almacenamiento redundante y escalable. Los objetos y los archivos se escriben en varias unidades de disco repartidos por los servidores del centro de datos, con el software Openstack responsable de asegurar la replicación y la integridad de los datos en el clúster. Agrupaciones de almacenamiento escalar horizontalmente simplemente añadiendo nuevos servidores.

Imagen 6: “descripción de swift”




https://docs.openstack.org/admin-guide/_images/objectstorage-ildingblocks.png

En la imagen anterior se ven los principales componentes que intervienen en el almacenamiento de objetos (swift).

El proxy server es la cara pública de Swift, es el encargado de recibir y responder las peticiones de la API, los anillos contienen la información del lugar donde se encuentra cada dato almacenado y las réplicas que existen de él, los nodos representan propiamente los servidores de almacenamiento.

Swift permite la utilización de zonas para garantizar la durabilidad de los datos, estas zonas pueden ser particiones, racks*, o lugares geográficos.

**3.6.6.2 Servicio de Orquestación (heat)**. El servicio de orquestación proporciona una orquestación basada en plantillas para describir una aplicación en la nube

* Es un soporte metálico destinado a alojar equipamiento electrónico, informático y de comunicaciones

ejecutando llamadas de la API de Openstack para generar aplicaciones en la nube en ejecución. El software integra otros componentes básicos de Openstack en un sistema de plantilla de un solo archivo. Las plantillas le permiten crear la mayoría de los tipos de recursos de Openstack, como instancias, IP flotantes, volúmenes, grupos de seguridad y usuarios. También proporciona funciones avanzadas como la alta disponibilidad de instancias, la escalabilidad automática de instancias y las pilas anidadas. Esto permite que los proyectos básicos de Openstack reciban una base de usuarios mayor.[9]

**3.6.6.3 Servicio de Telemetría (ceilometer)**. [17]. Los servicios de recopilación de datos de telemetría proporcionan las siguientes funciones:

- Medición de datos relacionados con los servicios de Openstack
- Recoge datos de eventos y mediciones mediante el monitoreo de las notificaciones enviadas desde los servicios.
- Publica los datos recopilados a varios objetivos, incluidos los almacenes de datos y las colas de mensajes.

**3.6.6.4 Servicio de almacenamiento en bloque (cinder)**. El servicio de almacenamiento en bloque de Openstack (cinder), agrega almacenamiento persistente a una máquina virtual. El almacenamiento en bloque proporciona una infraestructura para gestionar volúmenes, e interactúa con el servicio de cómputo Openstack para proporcionar volúmenes de instancias. El servicio también permite la gestión de instantáneas (snapshots) de volumen y tipos de volumen.10

[9] OpenStack Docs: Orchestration service overview, Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_orchestration.html. [Recuperado en: Nov. 2016].
[10] OpenStack Docs: Block Storage service overview, Docs.openstack.org, 2016. [En línea]. Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_block_storage.html. [Recuperado en: Nov- 2016].

# 4. ANÁLISIS Y DISEÑO

## 4.1 ALTA DISPONIBILIDAD DE DATOS

Actualmente se cuenta con DRBD, un sistema distribuido de almacenamiento replicado para plataformas Linux que permite complementar el modelo de alta disponibilidad.

Este modelo en la versión estable más actual, permite la replicación en dos nodos primarios.

**4.1.1 Cambios y Nuevas Implementaciones**. En vista de que el modelo en general presenta un alto grado de madurez, es importante seguir investigando y buscando que la infraestructura tenga mayor escalabilidad, por esto, en este proyecto se plantea la implementación de un sistema de archivos de almacenamiento en red de escalabilidad horizontal, que puede presentar una mejor adaptación al crecimiento de la misma.

## 4.2 SEGURIDAD

La infraestructura cuenta con políticas de seguridad para controlar el acceso a los servidores tanto físicos como virtuales además de una herramienta para la detección de ataques por fuerza bruta

**4.2.1 Cambios y Nuevas Implementaciones de Seguridad**. En una infraestructura de computación en la nube y servicios virtualizados, la seguridad juega un papel muy importante.

Al utilizar implementar estos servicios en una red considerablemente grande como es el caso en la universidad, se ve la necesidad de poder brindarle protección a

las máquinas virtuales; teniendo en cuenta que el riesgo aumenta al haber diferentes usuarios con y sin conocimiento sobre la administración de redes y servidores.

Sabiendo esto, se plantea la implementación de un firewall utm para la red de estas mismas, haciendo uso de software libre y herramientas que permiten cumplir este objetivo.

## 4.3 PROPUESTAS Y USO DE NUEVAS TECNOLOGÍAS

**4.3.1 Contenedores Linux**. Lxc o contenedores Linux es una tecnología de virtualización en el nivel de sistema operativo (SO) para Linux. LXC permite que un servidor físico ejecute múltiples instancias de sistemas operativos aislados, conocidos como Servidores Privados Virtuales (SPV o VPS en inglés) o Entornos Virtuales (EV). LXC no provee de una máquina virtual, más bien provee un entorno virtual que tiene su propio espacio de procesos y redes.[11]

### 4.3.2 Docker

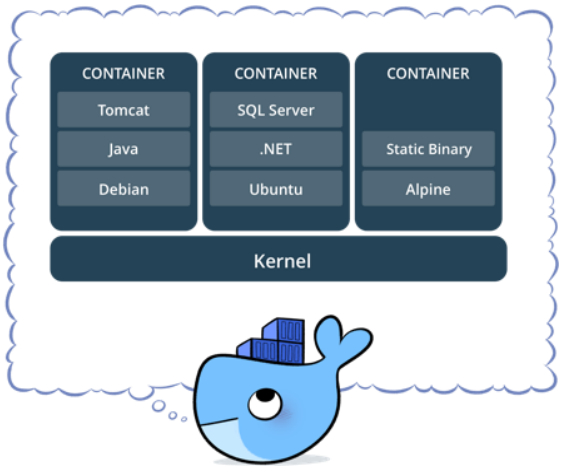
Imagen 7: “Logo Docker”


http://facilcloud.com/noticias/wp-content/uploads/2015/07/docker-logo.png)

[11] LXC, Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/LXC. [Recuperado en Feb- 2017].

Docker [22] es un proyecto de código abierto para crear contenedores que sean ligeros y portables y así lograr que las aplicaciones software se puedan ejecutar desde cualquier máquina virtual, sin importar el sistema operativo de la misma, siendo así muy fácil el despliegue de dichas aplicaciones.

Imagen 8: “Despliegue Docker”




https://www.docker.com/sites/default/files/what_is_a_container.png)

**4.3.3 ¿Qué es un Contenedor Docker?** Un contenedor docker se puede definir como una máquina virtual ligera menos exigente, donde dentro del mismo se encuentra todo lo necesario para ser ejecutado.

Estos contenedores no tienen un sistema operativo completo, sino únicamente librerías y archivos necesarios para su propio despliegue. Así se puede llevar un contenedor a cualquier máquina o host anfitrión que contenga docker instalado y ejecutar la aplicación, sin preocuparse por si dicho host o máquina poseen los elementos necesarios para ejecutar la aplicación.

El software en un contenedor siempre funcionará igual, independientemente del entorno

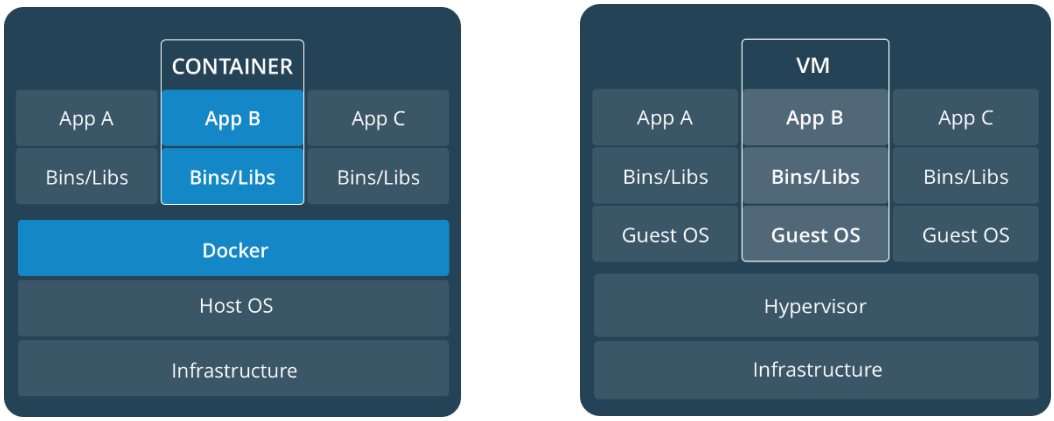
**4.3.4 Los contenedores y las Máquinas Virtuales**. Las máquinas virtuales y los contendores tienen beneficios similares en cuanto a aislamiento y asignación de recursos, pero funcionan de manera muy diferente puesto que los contenedores virtualizan el sistema operativo, más no el hardware. Es por esto que los contenedores son más portátiles y eficientes.[12]

**4.3.4.1 Contenedores**. [22] Múltiples contenedores pueden ejecutarse en la misma máquina y comparten el núcleo del sistema operativo con otros contenedores, cada uno ejecutándose como un proceso aislado. Los contenedores ocupan menos espacio que las máquinas virtuales (las imágenes de contenedores suelen tener decenas de MB de tamaño) y comienzan casi al instante.

**4.3.4.2 Máquinas Virtuales**. [22] Las máquinas virtuales (VM) son una abstracción del hardware físico que convierten a un servidor en muchos servidores[13]. El hypervisor permite que múltiples máquinas virtuales se ejecuten en una sola máquina. Cada VM incluye una copia completa de un sistema operativo, una o más aplicaciones, binarios y bibliotecas necesarias, ocupando decenas de GB. Las VM también pueden ser lentas para arrancar.

[12] What is Docker, Docker. [En línea]. Disponible en: https://www.docker.com/what-docker. [Recuperado en: Feb- 2017].
[13] "Máquina virtual", Docker.com, 2016. [En línea]. Disponible en: https://www.docker.com/what-container [Recuperado en: Dic- 2016].

Imagen 9: “Arquitectura contenedor, arquitectura máquina virtual”




https://www.docker.com/sites/default/files/Container%402x.png, https://www.docker.com/sites/default/files/VM%402x.png)

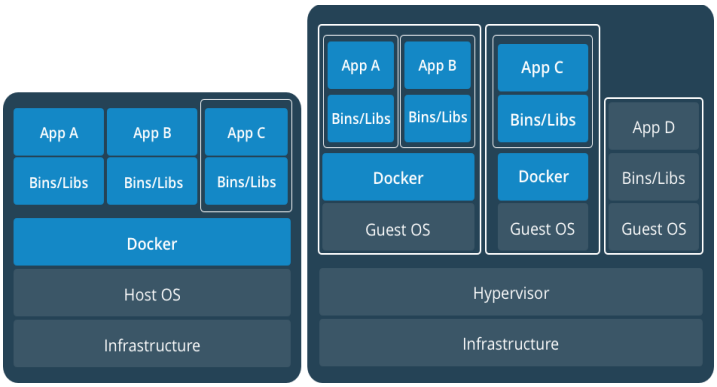
**4.3.5 Contenedores y Máquinas Virtuales Juntos**. Los contenedores y las máquinas virtuales utilizados conjuntamente proporcionan una gran flexibilidad en la implementación y administración de aplicaciones. Una pregunta frecuente es si los servicios Docker pueden interactuar con los de las máquinas virtuales. La respuesta es sí. La ejecución de un contenedor Docker no impide que sus servicios hablen con los de la máquina virtual que lo aloja. Por ejemplo, es posible que la aplicación en dicho contenedor interactúe con una base de datos que reside en la máquina virtual, si la red está correctamente configurada, no habrá problema en dicha interacción.

Docker aprovecha la ejecución de hosts Docker en una amplia variedad de plataformas de virtualización. Docker Cloud y Docker Datacenter pueden administrar fácilmente los hosts Docker sin importar donde se ejecutan. Y con Docker Machine puede proporcionar nuevos hosts Docker a una amplia variedad de plataformas, incluyendo VMware, vSphere, Microsoft Hyper-V, Azure y AWS.

Una de las cosas más poderosas de Docker es la flexibilidad que ofrece a las organizaciones. La decisión de dónde ejecutar sus copias puede basarse 100% en

lo que es correcto para su negocio. No se está encerrado en ninguna infraestructura, se puede escoger y combinar de cualquier manera que tenga sentido para la organización[14].

Imagen 10: “Despliegue Docker”




https://www.docker.com/sites/default/files/containers-vms-together.png

### 4.3.6 Ventajas y Desventajas de las Máquinas Virtuales

#### 4.3.6.1 Ventajas

- **Aprovechamiento de los recursos**: [10] en un servidor físico se pueden tener varias máquinas virtuales. El hypervisor puede compartir recursos basados en la prioridad, y las máquinas virtuales pueden hacer uso de ciclos no utilizados que otras máquinas virtuales no están usando. Cada máquina virtual utiliza su propia CPU, RAM, almacenamiento y asignación de red. Cada máquina virtual también ejecuta su propio sistema operativo.

[14] What is Docker, Docker. [En línea]. Disponible en: https://www.docker.com/what-docker. [Recuperado en: Feb- 2017].

- **Fácil escalabilidad**: [10] cuando se necesiten ejecutar más aplicaciones, se puede hacer uso de más máquinas virtuales, siempre y cuando el servidor huésped cuente con los recursos necesarios.

- **Copias de seguridad y migración fáciles**: [10] es muy fácil migrar una configuración de la máquina virtual a otro hardware. La mayoría de los proveedores proporcionan mecanismos sencillos para realizar copias de seguridad, restaurar y migrar las máquinas virtuales detenidas o en ejecución.

- **Flexibilidad**: [10] máquinas virtuales con diferentes arquitecturas pueden funcionar juntas, por ejemplo, máquinas Linux y máquinas Windows.

#### 4.3.6.2 Desventajas

- **Problemas a la hora de asignar recursos**: [10] en ocasiones es difícil asignar recursos a una máquina virtual específica, ya que todos los recursos se comparten.

- **Dependencia de un proveedor**: [10] el uso de una solución de virtualización dificulta el paso a otro. Existen algunas herramientas de migración, pero cada proveedor tiene un conjunto diferente de formatos.

- **Configuración compleja**: [10] ejecutar una aplicación dentro de una máquina virtual puede ser problemático. Si bien el sistema operativo puede ser más fácil de administrar, prepararlo para una aplicación puede seguir siendo un proceso engorroso. Diferentes instancias de la misma imagen pueden diferenciarse en términos de configuración y dependencias.

### 4.3.7 Ventajas y Desventajas de los Contenedores

#### 4.3.7.1 Ventajas

- **Aislamiento**: [10] cada aplicación se ejecuta en su propia instancia de contenedor. Debido a que su código está en un contenedor, debe ejecutarse de forma coherente entre instancias. Las preocupaciones de configuración pueden reducirse al mínimo utilizando contenedores. No hay necesidad de preocuparse por el sistema operativo del usuario final si envía el sistema operativo con el software.

- **Ligero**: [10] haciendo uso del núcleo del sistema operativo y no dependiendo de un hypervisor, la pila de contenedores necesita mucho menos recursos que cualquier otra instalación. Esto permite más contenedores en una máquina de lo que sería posible con máquinas virtuales.

- **Fácil de migrar**: [10] debido a que cada contenedor es una instancia aislada que no tiene un sistema operativo invitado, es muy fácil migrar de una implementación a otra. La pila de contenedores permite una gran portabilidad.

- **Seguridad**: [10] tener cada aplicación ejecutada en una instancia aislada también permite una gran seguridad. Sin embargo, se debe tener en cuenta que, dependiendo de la tecnología del contenedor, puede haber algunos agujeros en entornos de contenedores con múltiples tenientes.

- **Gastos indirectos bajos**: [10] debido comparte el núcleo con el sistema operativo anfitrión, es muy barato iniciar y detener contenedores.

- **Portables**: [10] Cada aplicación se ejecuta en un contenedor aislado. Estos contenedores aislados pueden ser enviados fácilmente desde una máquina a otra, sin presentar problemas.

- **Comunidad**: [10] En los últimos tiempos, la virtualización de contenedores ha evolucionado a pasos agigantados debido a la ayuda de muchos colaboradores de todo el mundo. Docker ha ayudado especialmente con esto, normalizó el flujo de trabajo de contenedores con herramientas de código abierto y ha creado una comunidad vibrante. Si se tiene un problema, es muy probable que alguien más ya haya encontrado el problema y pueda proporcionar ayuda o que su problema está siendo trabajado activamente.

#### 4.3.7.2 Desventajas

- Arquitectura: [10] los contenedores comparten el kernel del sistema operativo host, lo que significa que están limitados a la misma arquitectura. Las imágenes de contenedor se pueden construir para diferentes arquitecturas, pero no son portátiles a través de diferentes arquitecturas.

- Cargar aplicaciones pesadas: [10] Las aplicaciones grandes y con muchos recursos pueden estar mejor adaptadas a una configuración de servidor tradicional en la que tienen acceso más directo a los recursos proporcionados por el servidor.

La diferencia clave entre la configuración del contenedor y las máquinas virtuales es que la virtualización basada en contenedores utiliza el núcleo del sistema operativo host para ejecutar varias instancias invitadas aisladas. Estas instancias invitadas se llaman contenedores. El host puede ser un servidor físico o una máquina virtual.

Lo que hace que esta plataforma sea tan grande es que con Docker, se obtiene más que acceso a un conjunto de herramientas de núcleo de alta calidad. También se tiene acceso a una comunidad y todo un ecosistema de productos y servicios de terceros que pueden ayudar desde los primeros pasos a desarrollar una aplicación a través de cada implementación incremental de la misma.

La plataforma principal Docker consiste en una variedad de productos y servicios:

- Docker Engine (Docker Daemon)
- Docker Hub (Registro)
- Docker Machine (Provisión de host)
- Enjambre Docker (agrupación de host)
- Docker Compose (Container Orchestration)
- Docker Orca (Gestión de aplicaciones)
- Kitematic (GUI)

**4.3.8 Uso de Contenedores Docker**. La optimización en la utilización de recursos computacionales en una infraestructura que va en crecimiento es uno de los principales objetivos; por esto es necesario buscar alternativas y/o complementos a la virtualización que permitan hacer mejor uso de ellos

Es así que se plantea una propuesta para empezar a experimentar con contenedores Linux más específicamente con Docker y ver qué ventajas traerán para en un futuro implementarlos en proyectos en los cuales se le pueda sacar mayor provecho a esta tecnología.

Para el caso en particular de la infraestructura del grupo GID-CONUSS, la cual está implementada en máquinas virtuales, se realizaron unas pruebas en las cuales se quiso emular parte de la infraestructura y observar las diferencias más significativas.

**4.3.9 GLUSTERFS**. GlusterFS es un sistema de archivos de red escalable. Mediante el uso de hardware común, puede crear soluciones de almacenamiento distribuidas grandes para la transmisión de medios, el análisis de datos y otras tareas intensivas de datos y ancho de banda. GlusterFS es un software libre y de código abierto.[15]

El GlusterFS se basa en la interacción de componentes cliente y servidor. Los servidores normalmente se implementan como almacenamiento en bloques, en cada servidor el proceso daemon glusterfsd exporta un sistema de archivos local como un volumen. El proceso cliente glusterfs, se conecta a los servidores a través de algún protocolo TCP/IP, InfiniBand o SDP, compone volúmenes compuestos virtuales a partir de los múltiples servidores remotos, mediante el uso de traductores. Por defecto, los archivos son almacenados enteros, pero también puede configurarse que se fragmente en múltiples porciones en cada servidor. Los volúmenes pueden ser montados en los equipos cliente mediante el módulo FUSE o acceder a través de la librería cliente libglusterfs sin incurrir en problemas con el sistema de archivos FUSE.[16]

- **Funcionalidades**:

- Espejo y la replicación de archivos.
- Fragmentación de los archivos o Data striping.
- Balanceo de carga para la lectura y escritura de archivos.
- Volúmenes con tolerancia a fallos.
- Planificación de E/S y almacenamiento en caché de disco.

[15] Gluster Docs, Gluster.readthedocs.io, 2017. [En linea]. Disponible en: https://gluster.readthedocs.io/en/latest/. [Recuperado en: Ene- 2017].
[16] Gluster File System, Es.wikipedia.org, 2016. [En linea]. Disponible en: https://es.wikipedia.org/wiki/Gluster_File_System. [Recuperado en: Ene- 2017].

El servidor GlusterFS se mantiene mínimamente simple: exporta un sistema de archivos existente como está, dejando en manos de los traductores del lado del cliente a la estructura del almacenamiento. Los propios clientes se manejan independientemente, no se comunican directamente entre sí, y los traductores administran la consistencia de los datos entre ellos. El GlusterFS se basa en un algoritmo de hash* elástico en vez de utilizar un modelo de metadatos centralizados o distribuidos. Desde la versión 3.1, los volúmenes pueden ser agregados, eliminados o migrados en forma dinámica, esto ayuda a prever problemas de consistencia, y permite que el GlusterFS pueda ser escalado a varios petabytes sobre hardware de bajo coste, evitando así los cuellos de botella que normalmente afectan a muchos sistemas de archivos distribuidos con múltiple concurrencia.[17]

Para la selección de este sistema de replicación se tuvieron en cuenta recomendaciones de proyectos anteriores, así como también una comparación entre con sistemas de replicación como ceph y drbd.

De los cuales se descartó drbd[18] ya que es el sistema de replicación actual y el cual se quería mejorar, quedado ceph[19] y glusterfs, de los cuales el primero es muy común en implementaciones de openstack y manejo de grandes volúmenes de datos, y está mucho más encaminado al almacenamiento de objetos similar a object storage, por lo tanto se quería un sistema fácil de escalar, seguro, rápido y que tuviera bastante documentación, más acorde a las necesidades de la infraestructura y replicación en aplicaciones web

* Algoritmo matemático que transforma cualquier bloque arbitrario de datos en una nueva serie de caracteres con una longitud fija

[17] Gluster File System, Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Gluster_File_System. [Recuperado en: Ene- 2017].

[18] https://docs.linbit.com/

[19] http://ceph.com/

# 5. DESARROLLO Y PUESTA EN MARCHA

## 5.1 SELECCIÓN DEL FIREWALL

Para este proyecto se tuvieron en cuenta principalmente dos soluciones firewall utm, las cuales fueron:

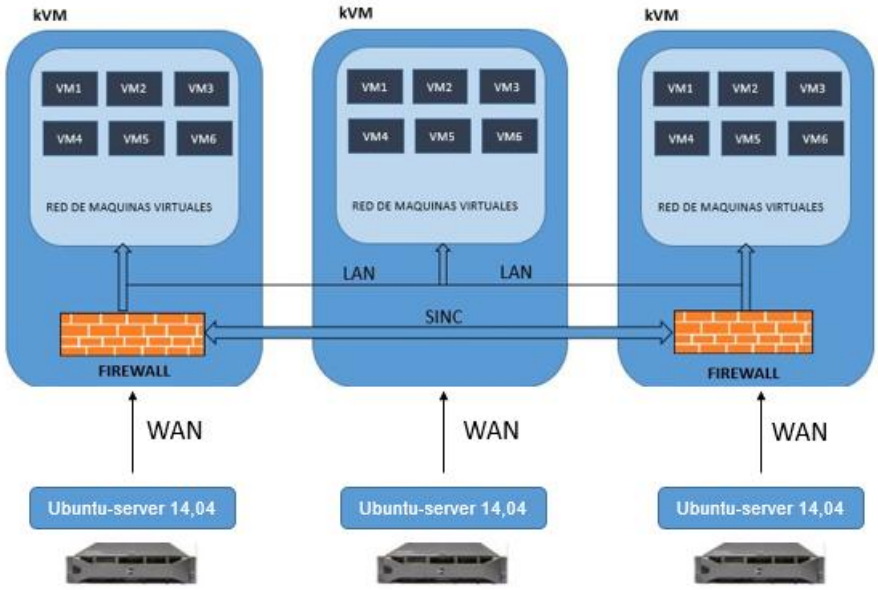
- **Endian community**
- **PfSense**

Estas dos herramientas prestan servicios similares y para el propósito que se le daría, las dos permitían cumplir con el objetivo; aunque la herramienta PfSense presenta en un principio un grado de complejidad mayor al momento de su implementación con respecto al endian community, nos permite en su versión libre trabajar el carp failover o alta disponibilidad del mismo mientras que en el endian solo es posible realizar este proceso en su versión paga.

El funcionamiento de esta característica que era indispensable ya que todo el tráfico de la red de máquinas virtuales pasaría a través de él, y por lo tanto en caso de fallos en el firewall no sería posible acceder a los servicios o sistemas alojados en las máquinas virtuales, es similar al proceso que realizan los balanceadores de carga, en el cual uno de los nodos denominado nodo primario o nodo master permanece activo y el otro permanece en modo pasivo y solo entraría en funcionamiento en el momento en que el nodo master falle.

Una vez elegida la herramienta teniendo en cuenta las características que eran necesarias para cumplir con el objetivo propuesto, se procedió con su implementación.

Imagen 11: “Esquema general de implementación del firewall”

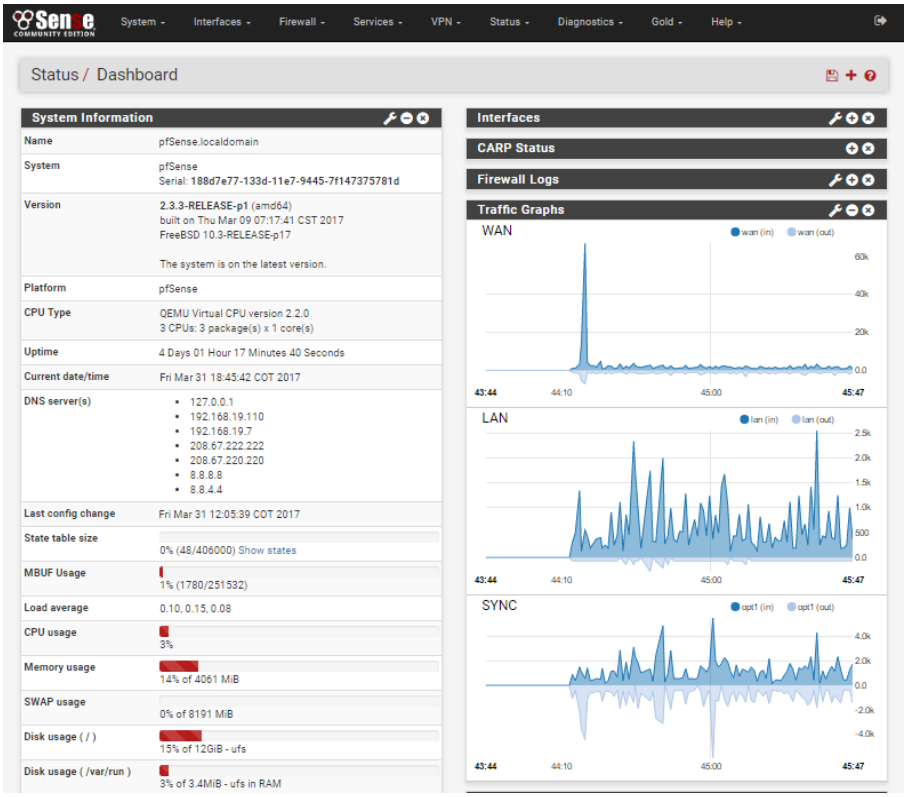



En la imagen anterior se muestra un esquema general de la implementación del firewall, para ello es necesario hacer uso de tres interfaces de red; una será utilizada como red WAN o red externa que tendrá acceso a internet otra será utilizada como red LAN o red de comunicación entre las máquinas virtuales y la otra será la red de sincronización entre los firewalls necesaria para implementar carp failover o alta disponibilidad en modo activo pasivo.

Para este caso particular teniendo en cuenta que el objetivo final sería brindarles protección a las máquinas virtuales, el papel de la red WAN lo haría la red LAN de los servidores físicos.

Una vez instalado el firewall PfSense, nos presenta una interfaz web personalizable la cual facilita el monitoreo, la administración e instalación de diferentes herramientas.

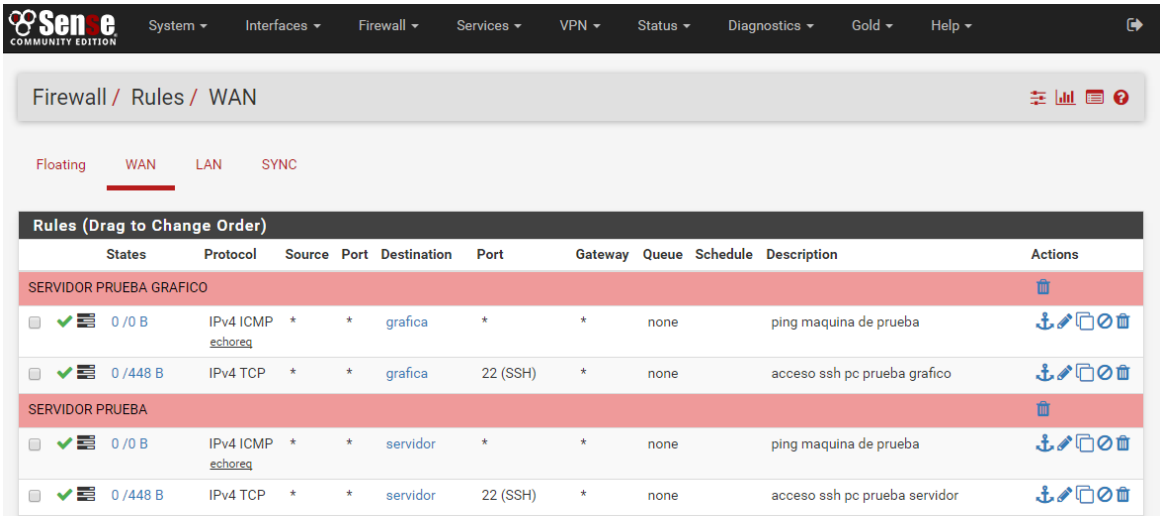
Imagen 12: “Interfaz web del firewall PfSense”




En la imagen anterior se muestra la interfaz web del firewall PfSense, la cual facilita la administración de la red de las máquinas virtuales, en donde podemos observar en una columna las características del hardware junto con especificaciones técnicas y en la otra columna encontramos información del estado del nodo principal, además de graficas del tráfico de red, logs del firewall y estado de las interfaces de red.

El funcionamiento del firewall se basa en reglas, mediante las cuales se rechaza, deniega o acepta la comunicación o tráfico entre redes e ip; adicional a esto PfSense cuenta con paquetes o herramientas que complementan y aumentan el control y la seguridad de la infraestructura.

Imagen 13: “interfaz del menú reglas”



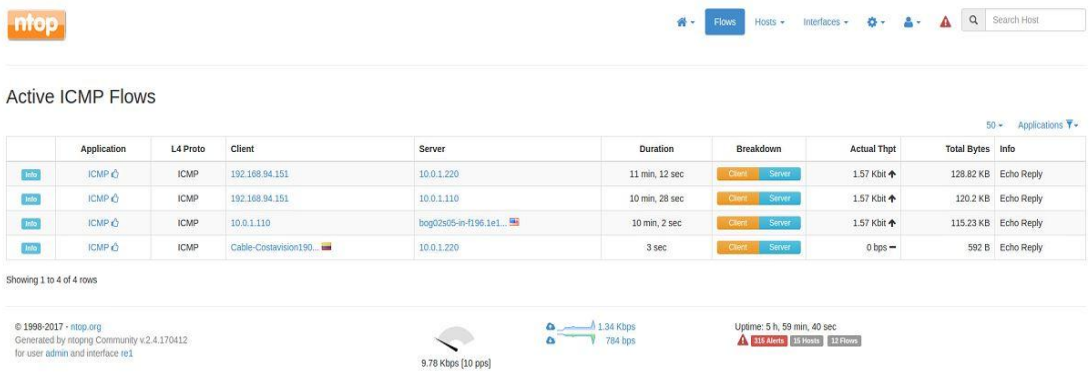

En la imagen anterior se muestra la interfaz del menú reglas en la cual se identifican los submenús floating, WAN, LAN y sync que hacen referencia a las interfaces o segmentos a los cuales se les puede aplicar reglas.

Resaltamos el submenú WAN y LAN, en el primero podemos configurar reglas de acceso y comunicación hacia los servidores o máquinas virtuales que se encuentran en la red LAN y en el segundo podemos configurar las reglas de acceso y comunicación entre los servidores que se encuentran en la red LAN así como también la salida hacia la red WAN y redes externas.

### 5.1.1 Alertas y Monitoreo de Red

**Ntop-ng**

Imagen 14: “Panel de monitoreo Ntop-ng”




Al prestar servicios computacionales más específicamente infraestructura, la administración en general del contenido, accesos y demás operaciones de los servidores adjudicados a usuarios y estudiantes de la comunidad universitaria no es realizada por los administradores del grupo CONUSS, por lo tanto, es importante poder monitorear la red para poder tomar medidas pertinentes en caso de ser necesario.

Como se mencionó anteriormente se utilizó ntop-ng como complemento del firewall para monitorear esta misma, la herramienta permite registrar todo el tráfico entrante y saliente para así poder tener control completo sobre la red mediante una interfaz gráfica.

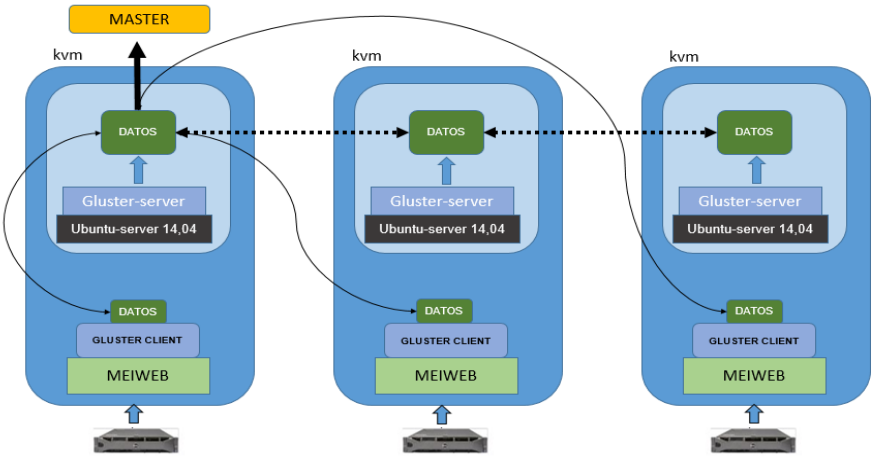
En la imagen anterior se evidencia uno de los menús de la interfaz gráfica de la herramienta ntop-ng, en el momento de la imagen se hacía un monitoreo por filtro, dando prioridad al protocolo icmp (ping), estas pruebas de monitoreo fueron

realizadas de manera controlada y con servidores virtuales propios del grupo CONUSS

## 5.2 IMPLEMENTACIÓN Y PRUEBAS CON GLUSTERFS

### 5.2.1 Modelo Implementado

Imagen 15: “Arquitectura del modelo de replicación de datos con GlusterFS”

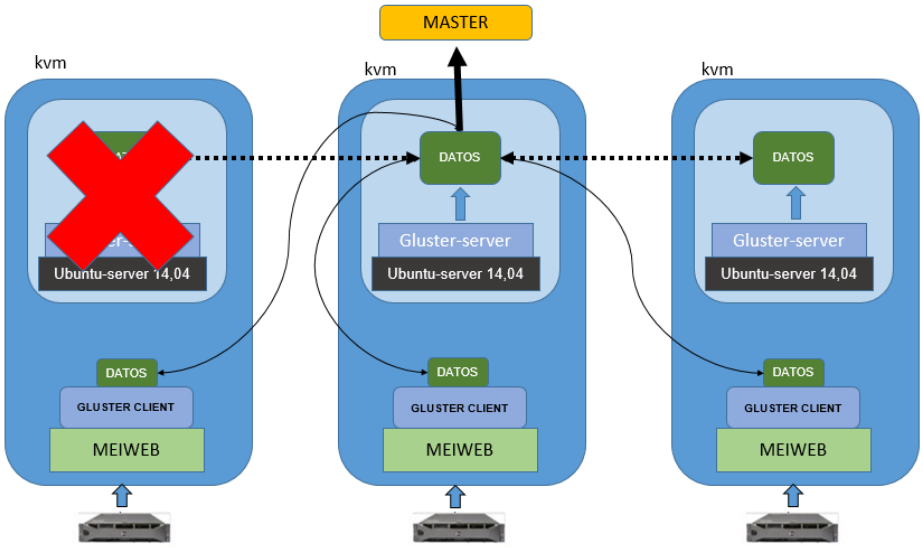



En la imagen anterior se puede ver el esquema que se implementó para realizar las pruebas con GlusterFS, en el cual se utilizó el portal Meiweb en tres nodos o servidores virtuales, tres nodos con Ubuntu server 14.04, que funcionan como servidores y se distribuyeron en los tres servidores físicos de producción con los que cuenta el grupo GID-CONUSS en este momento.

Una vez configurado todo el modelo, se hicieron pruebas de creación, modificación y eliminación de datos en cada uno de los tres nodos clientes Meiweb, para observar el comportamiento de la replicación de los mismos obteniendo un resultado óptimo en cuanto a velocidad e integridad y consistencia.

### 5.2.2 Tolerancia a Fallos

Imagen 16: ”Arquitectura del modelo de replicación de datos con glusterfs, tolerancia a fallos”




Teniendo en cuenta que existen diferentes factores que la afectan la infraestructura directamente tales como fallos en la luz, red, problemas en los servidores físicos o daños en los discos de las máquinas virtuales producto de apagones, era necesario plantear un modelo que permitiera un alto porcentaje de disponibilidad de los servicios prestados aprovechando al máximo las prestaciones y recursos de los servidores físicos.

Por esto, se realizaron pruebas creando escenarios simulando apagones, daños en el nodo principal del servidor GlusterFS, daños en los clientes, así también, encendido tardío con intensión, de un cliente después de haber realizado una prueba de copia y verificando que éste último al iniciar se sincronizará con los demás sin generar inconsistencia.

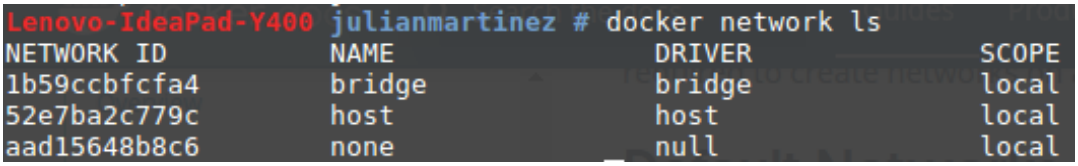
## 5.3 DOCKER

Al querer trabajar con Docker, sólo es necesario disponer de un host equipado con alguna distribución de Windows, Mac o Linux. Los más recomendado es que sea una distribución de Linux, dado que Docker tiene una estrecha integración con el kernel de Linux. Esto hace que la eficiencia sea bastante alta a un bajo costo. Docker no es (actualmente) un reemplazo para las máquinas virtuales para Windows, OS X, etc. Al ejecutar contenedores Docker en una máquina que no sea Linux, se ejecutarán dentro de una máquina virtual a través de boot2docker.

En este caso en particular se usó Ubuntu14. Luego de esto se instalan los componentes necesarios para poder usar Docker.

Al instalarse Docker, este crea tres redes automáticamente, las cuales son:

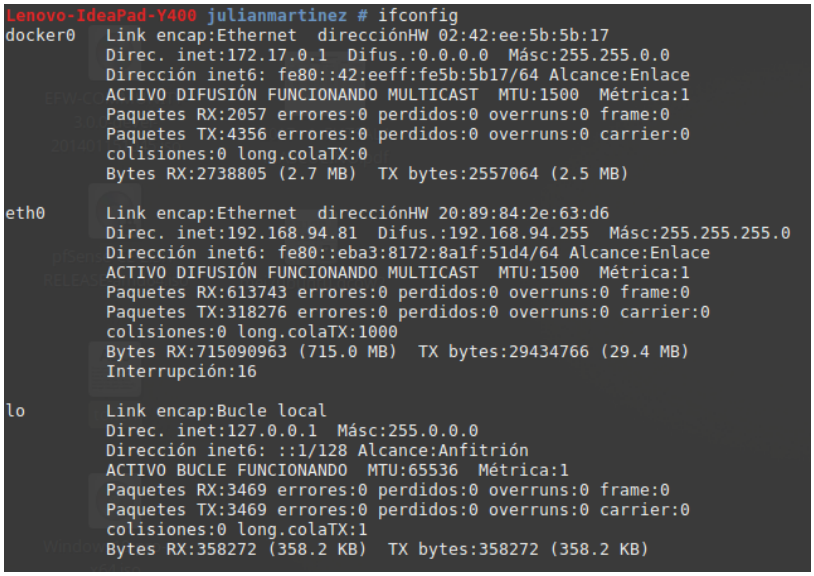
Imagen 17: “Redes creadas por Docker”




Estas tres redes están integradas en Docker, cuando se crea un contenedor, hay una opción para especificar la red a la que queremos que pertenezca dicho contenedor.

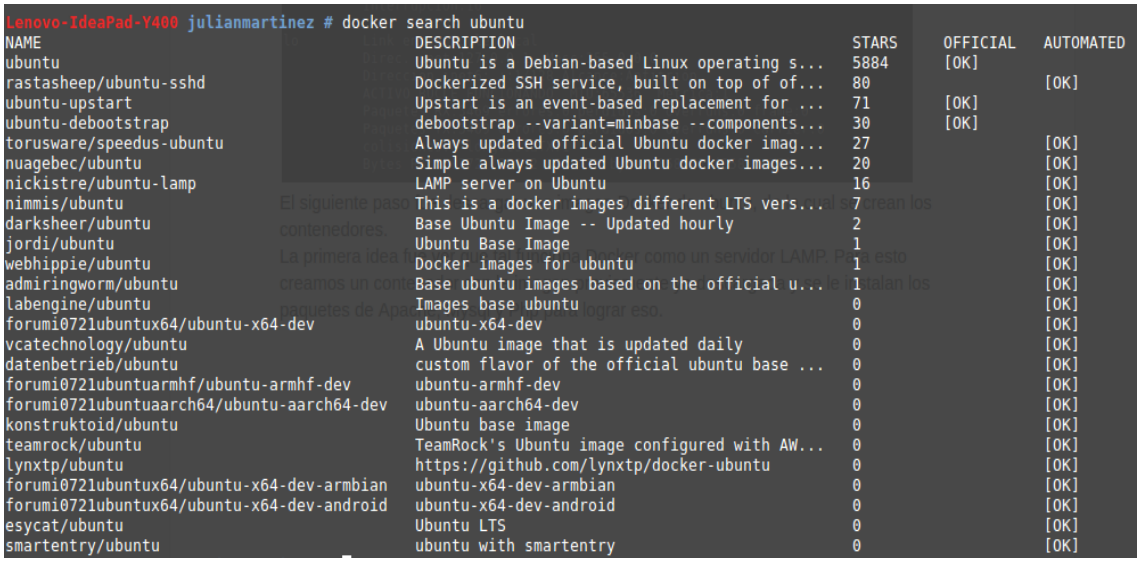
La red bridge representa la red docker0 que está presente en todas las creaciones de contenedores, a menos que especifiquemos lo contrario. El demonio de Docker conecta los contenedores a esta red y lo hace de manera predeterminada. Esto se puede ver revisando la pila de redes del host en el que está instalado Docker.

Imagen 18: “Redes de nuestro sistema”




El siguiente paso fue buscar una imagen Docker de Ubuntu, de la cual se crean los contenedores.

Imagen 19: “Imágenes disponibles de Docker”



La primera idea fue ver que tal funciona Docker como un servidor LAMP*.

Para esto se creó un contenedor con la imagen previamente descargada.

Acá ya se empieza a ver una ventaja importante de Docker sobre las máquinas virtuales. Crear un contenedor, iniciarlo y trabajar en el solo es cuestión de segundos, basta con tener una imagen base de Docker con la cual se quiere trabajar y lanzar los comandos correctos para estar trabajando en dicho contenedor. Mientras que con una máquina virtual es completamente diferente, ya que antes de empezar la máquina, hay que instalar el sistema operativo sobre el cual se quiere trabajar, lo que conlleva a una configuración tediosa y por ende a un gasto de tiempo mucho mayor al de un contenedor Docker.

Al momento de ingresar al contenedor, aparece lo siguiente:

Imagen 20: “Path del contenedor”

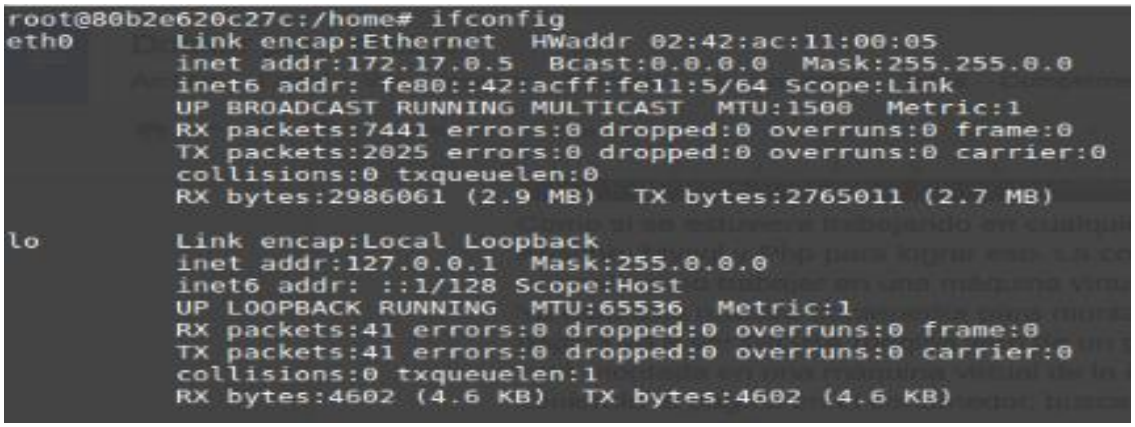

Como si se estuviera trabajando en cualquier host de Ubuntu se instalan los paquetes de Apache, Mysql y Php. La configuración de todos estos paquetes se hace de manera convencional, como trabajar en una máquina virtual o en un servidor físico.

Una vez hecho esto, el contenedor se encuentra listo para montar una página web. Para esta prueba se utilizó la página GIGBA (página web de un grupo de investigación de geología), que ya está montada en una máquina virtual, sobre un sistema operativo Ubuntu server 14.04 de la infraestructura.

* Linux, apache, mysql, php

Teniendo la página en el contenedor, se busca la IP de este.

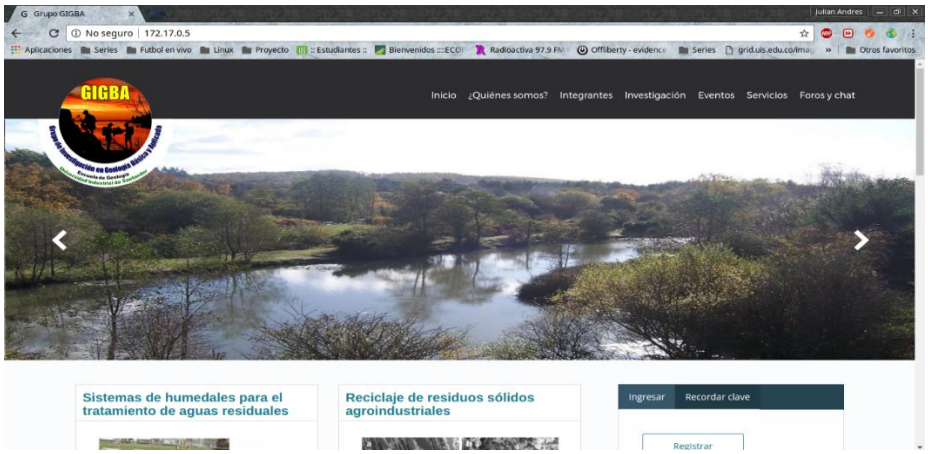
Imagen 21: “Redes del contenedor”




Se observa que el contenedor tiene una IP de la red predeterminada, llamada *docker0* mencionada anteriormente.

Esta IP se ingresa al navegador para verificar que todo esté funcionando de manera correcta.

Imagen 22: “Página web corriendo en el contenedor”

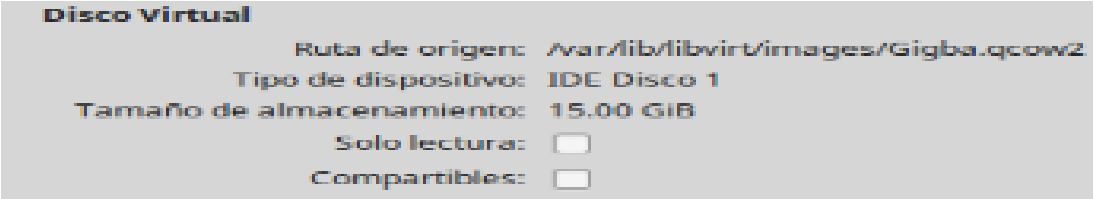


En la imagen anterior se puede observar que la pagina carga de manera correcta.

Una de las ventajas más importantes del uso de esta tecnología, es el ahorro de espacio en disco.

Este ahorro es bastante significativo ya que con el contenedor solo se usa un 18% de espacio en comparación al utilizado por la máquina virtual con esta misma página web

Imagen 23: “Espacio usado por la máquina virtual”




Cabe destacar que, si la máquina tiene 15 GB, es porque así se necesita para este caso en particular, sin embargo, los requisitos mínimos recomendados para un Ubuntu server 14.04 que sirve como servidor LAMP es de 10 GB. SI este hubiera sido el caso, también un gran ahorro en espacio en disco.

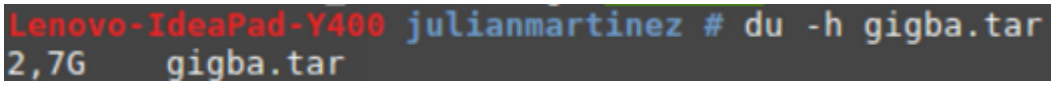
Para observar el tamaño de la imagen Docker de GIGBA, se exporta en un formato .tar

Imagen 24: “Comando para exportar imagen Docker”

```
Lenovo-IdeaPad-Y400 julianmartinez # docker export gigba > gigba.tar
```

La exportación se demora unos minutos dependiendo del peso de la misma.

Imagen 25: “Tamaño archivo .tar”



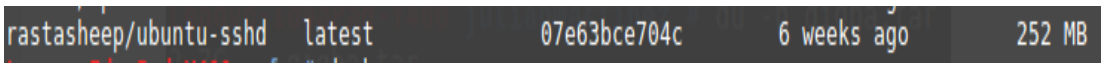

Queda evidenciado el gran ahorro de espacio que proporciona trabajar con contenedores Docker.

Este no es el único beneficio que Docker ofrece, con esta imagen también se pueden generar todas las copias que se quiera de dicha máquina, es decir puede funcionar como una plantilla, haciendo también que el ahorro no sea solo de espacio, sino que también sea de tiempo. Se ahorra el tiempo de espera a la hora de clonar una máquina o de instalar una distribución de Linux completa en esta misma. Otro gran beneficio que se puede tener con Docker es su portabilidad, esta imagen se puede llevar a cualquier entorno y ponerla a funcionar, siempre y cuando tenga Docker instalado.

Otra prueba que se realizó fue montar las bases de datos distribuidas (tres bases de datos), algo que con máquinas virtuales se demoraría horas, con los contenedores Docker es posible realizarse es mucho menos tiempo.

Se puede usar la misma imagen Docker de Ubuntu que se usó para el servidor LAMP cuyo nombre es *rastasheep/ubuntu-sshd*. A partir de dicha imagen se crea el nuevo contenedor para la base de datos.

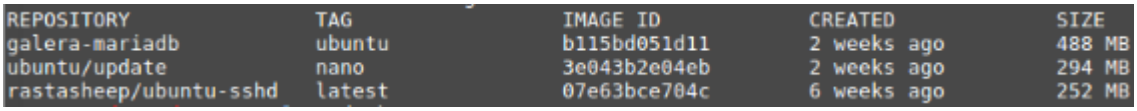
Imagen 26: “Imágenes Docker en nuestro sistema”



Los paquetes de MariaDB y Galera se instalan en el nuevo contenedor. A partir de ahí se puede hacer uso de las facilidades de Docker, ya que con este contenedor se crea otra imagen Docker para usarla como plantilla para los otros dos contenedores. Es así como se puede ahorrar una gran cantidad de tiempo en configuraciones tediosas.

A la nueva imagen se le dio el nombre de *galera-mariadb*.

Imagen 27: “Imágenes Docker de nuestro sistema”



| REPOSITORY | TAG | IMAGE ID | CREATED | SIZE |
|---|---|---|---|---|
| galera-mariadb | ubuntu | b115bd051d11 | 2 weeks ago | 488 MB |
| ubuntu/update | nano | 3e043b2e04eb | 2 weeks ago | 294 MB |
| rastasheep/ubuntu-sshd | latest | 07e63bce704c | 6 weeks ago | 252 MB |

Con esta nueva imagen se crean rápidamente los dos nuevos contenedores.

Si se llega a borrar algún contenedor, toda la información que este contiene, se elimina junto a él. Para combatir este problema, Docker ofrece la opción de gestionar volúmenes de datos. Estos volúmenes se montan directamente en el contenedor. Un contenedor puede tener varios volúmenes montados y un volumen puede ser usado por varios contenedores. Así que, si un contenedor se llega a borrar, no se borrará el volumen asociado a él.

Para el caso del servidor LAMP es recomendable crear dos volúmenes de datos para guardar el contenido del servidor *(/var/www/*) y los logs del mismo *(/var/log/apache2/*). En el caso de las bases de datos se crea un volumen para */var/lib/mysql/*.

Con esta investigación sobre Docker se concluye que, si se integra junto con otras herramientas para buscar la seguridad, alta disponibilidad y contenedores como servicio, podría ser una herramienta de gran ayuda para la infraestructura del

grupo GID-CONUSS, donde en un futuro se llegase a mudar parte de dicha infraestructura de máquinas virtuales a contenedores Docker ya que como se ha mencionado el ahorro de espacio y tiempo es bastante alto.

SI existe alguna máquina que maneje grandes cantidades de archivos lo más recomendable es no migrar a contenedores y seguir usando máquinas virtuales, ya que tienen acceso más directo a los recursos que la máquina proporciona.

Otra observación a tener en cuenta con esta investigación es que no existe un modelo predeterminado para una infraestructura, sino que simplemente debemos ver qué manera de virtualizar se adapta mejor a lo que necesitamos. Tampoco tiene que ser toda una infraestructura con contenedores o toda con máquinas virtuales, sino que se puede buscar la forma de acoplar las dos formas de virtualizar para así brindar el mejor rendimiento posible.

## 5.4 OPENSTACK

La puesta en marcha de esta herramienta para la administración de la nube privada además de ofrecernos la configuración y ejecución de todas sus funcionalidades por medio de línea de comandos, nos brinda también mediante una interfaz gráfica una forma ágil y amigable de gestionar y configurar los diferentes proyectos cloud que se requieran.

Esta interfaz es conocida como horizon e interactúa con todos los módulos de la infraestructura para así poder hacer uso de la funcionalidad de cada uno.

Imagen 28: “Inicio de sesión de Openstack”

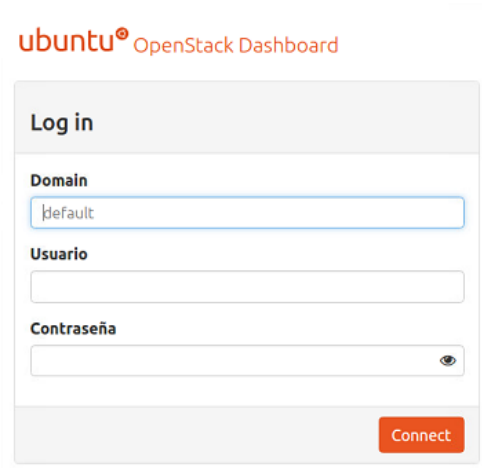
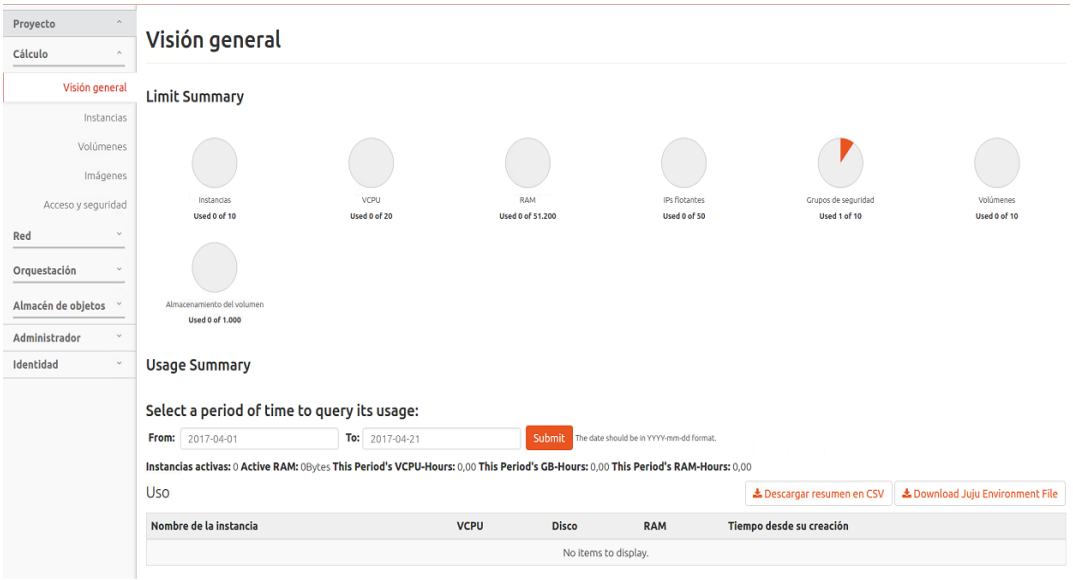



Imagen 29: “Vista general de la interfaz de Openstack”




En la imagen anterior se ve la vista general desde la sesión de administrador de la interfaz de openstack, desde esta vista el administrador puede controlar todo lo que sucede en la infraestructura, creación de tipos que permiten crear características que se les darán a las instancias de acuerdo a los requerimientos

que se precisen, creación de redes, instancias e imágenes que podrán ser utilizadas por los demás usuarios y proyectos.

**5.4.1 Topología de Red**. Una de las ventajas más importantes del módulo neutrón es la vista de topología de red, que nos muestra de manera gráfica como esta implementada la red, y de qué forma se conectan las instancias hacia el exterior o red pública.

Imagen 30: “Topología de red”

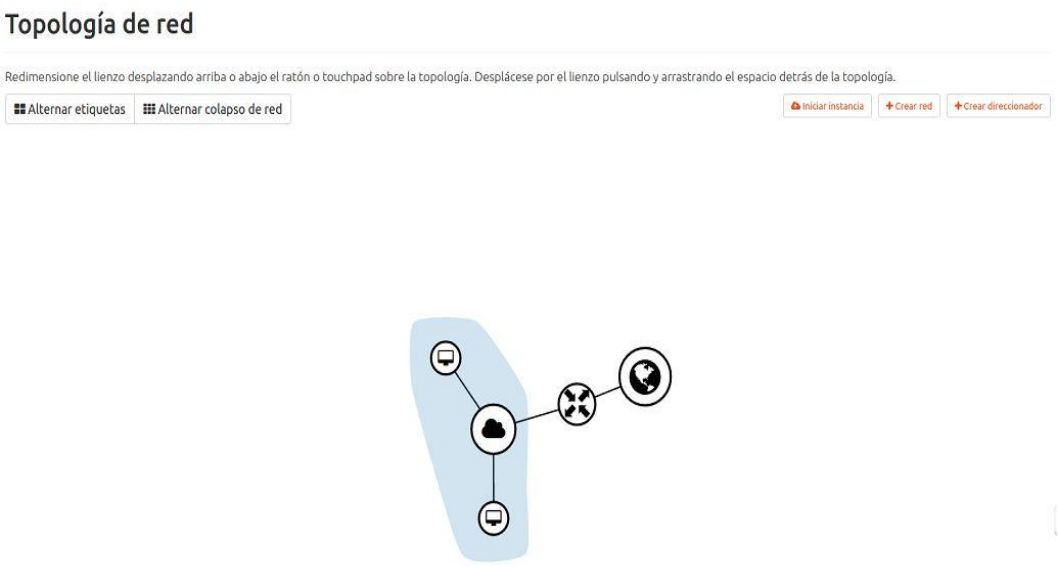



Imagen 31: “Redes existentes”

En las imágenes anteriores se puede ver como se distribuye la red en la infraestructura, en el momento de la imagen se contaba con dos redes, una publica que permite el acceso y salida a internet y una red interna en la que solo se comunican las maquinas o instancias que estén en ella.

Para que una instancia pueda acceder a internet o a la red pública, es necesaria la creación de un router virtual como se evidencia en la imagen, que conectará las dos redes y posteriormente se le asignará una IP flotante. Una vez creadas las redes se procede a la creación de las instancias.

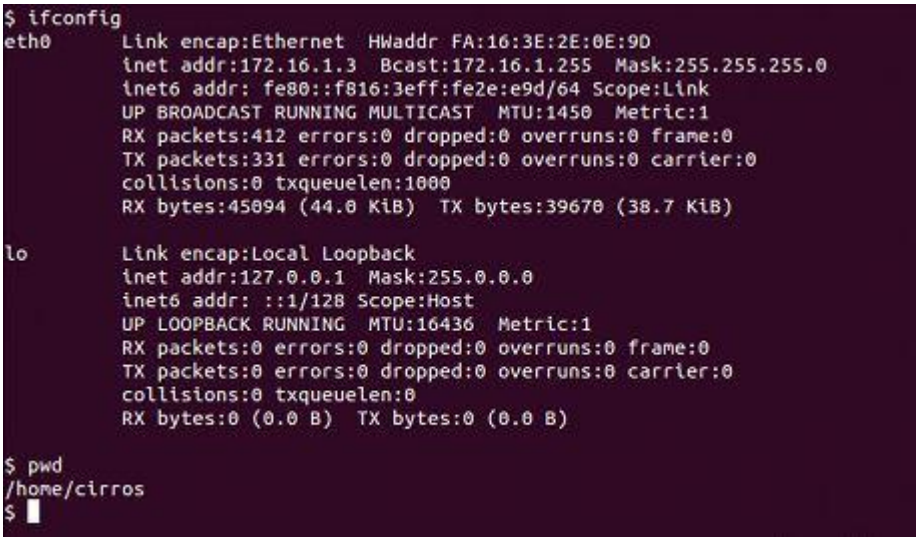
### 5.4.2 Instanciación de Imágenes

Imagen 32: “Instalación de imágenes”


En la imagen anterior se puede evidenciar la creación de instancias. Para su creación es necesario seguir una serie de procedimientos en los cuales se debe elegir una imagen para instanciar, una red internet y posteriormente la asignación de IP flotante para la salida a la red pública, selección de tipo o sabor que nos definirá los requerimientos de hardware que utilizará dicha instancia, la zona de disponibilidad que será el espacio donde se guardará la instancia en alguno de los

nodos compute y por último la asignación de volúmenes y demás requerimientos opcionales.

Imagen 33: “Datos de red de la instancia creada”




Como se mencionó anteriormente, una vez creada la imagen, es necesario asignarle una ip flotante para tener acceso al exterior.

En la ilustración anterior se observa mediante una conexión por ssh la configuración de red de una instancia generada, notándose la ip interna de la a la cual se le asignó una ip flotante para conexiones desde y hacia el exterior.

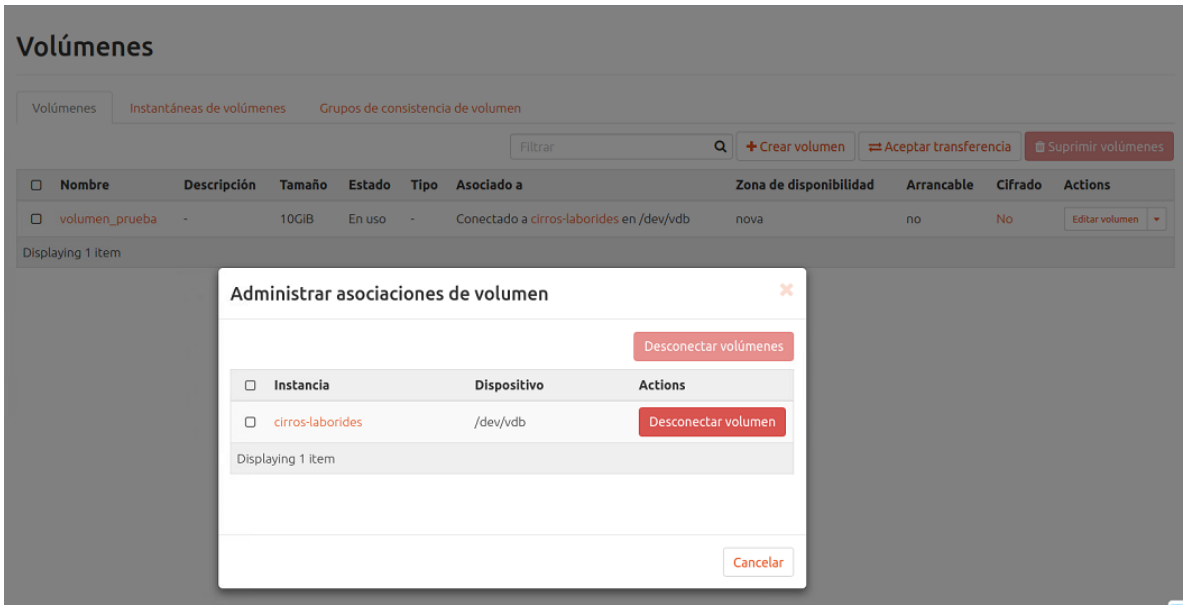
**5.4.3 Block Storage (cinder)**. Como se ha mencionado en apartados anteriores en este documento, al crear una instancia, se le asignan diferentes características de hardware que permiten su funcionalidad.

Algunos usuarios necesitan para sus servidores la utilización de espacio o disco adicional para diferentes propósitos.

Openstack, ofrece la posibilidad de agregar, editar o eliminar volúmenes en bloque como si fueran discos físicos que se le instalan a dicho servidor.

Existen varias maneras de realizar este procedimiento, ya sea por línea de comandos o por medio de la interfaz gráfica, se puede hacer al momento de crear la instancia o después.

Imagen 34: “Imagen asociación de volumen a una instancia”




En la imagen anterior se muestra uno de los apartados de horizon que permite asociar un volumen a una instancia previamente creada.

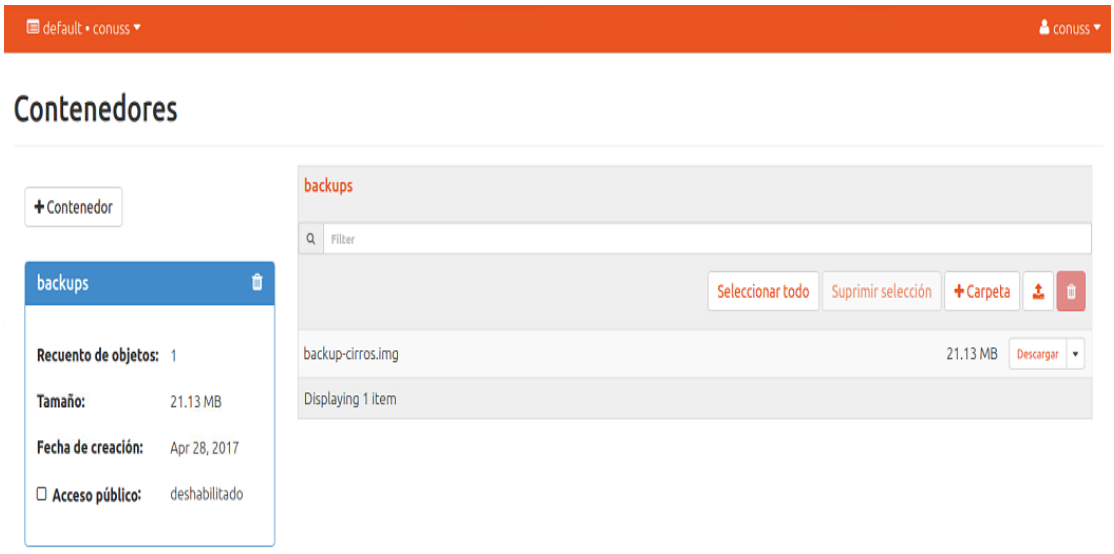
**5.4.4 Object Storage (Swift)**. Es importante al momento de prestar servicios poder recuperarse de fallos en un momento dado, una de las formas de afrontar de estas situaciones es mediante el respaldo de la información o backups.

Openstack, ofrece la posibilidad de hacer backups de las instacias y volúmenes creando snapshots o instantáneas de los mismos con una reducción de tamaño

considerable para permitir una manipulación de manera ágil en la red y Swift ofrece un servicio de almacenamiento de objetos mediante el uso de contenedores generando replicación de los datos almacenados, mediante su modelo de zonas, anillos y particiones, lo cual genera confianza al momento de guardar copias, ya que se reparten en diferentes nodos, pero de manera transparente para el usuario o administrador de la nube.

Una vez el administrador crea el backup, este se puede descargar en el formato img, el cual es utilizado para las instancias y posteriormente cargar en el contenedor del usuario propietario de la instancia a la cual se le hizo copia.

Imagen 35: “Imagen módulo de contenedores de almacenamiento”



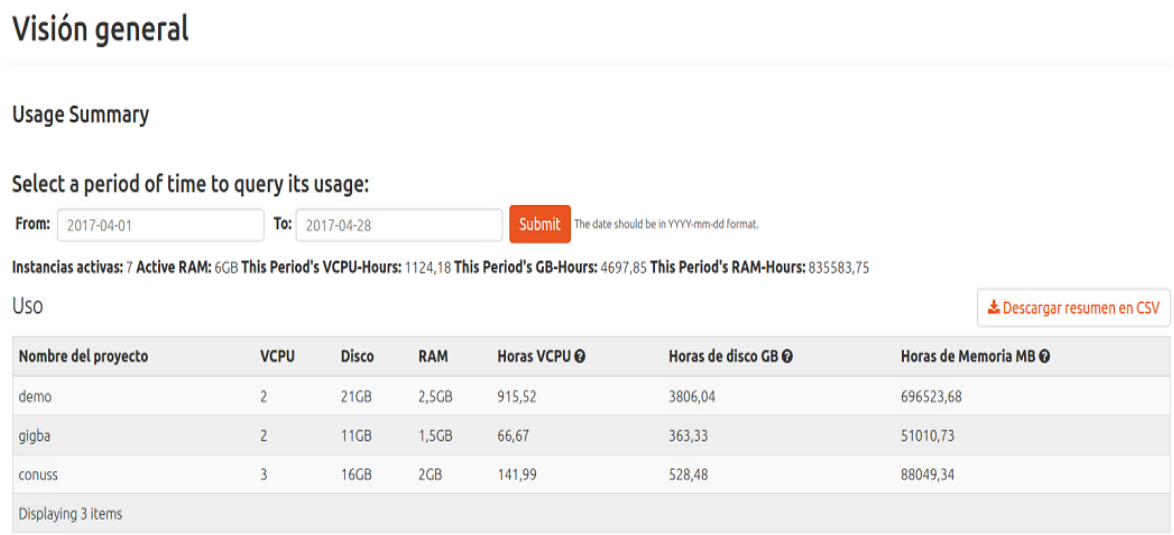

En la imagen anterior se observa un contenedor llamado backups, en el cual se encuentra almacenada una imagen backup de una instancia, para desplegar en este mismo escenario en caso de fallos o para migrarla a otra infraestructura.

Swift nos brinda la posibilidad por medio de los contenedores de generar un enlace de descarga de los datos en caso de que el usuario lo necesite.

**5.4.5 Telemetria (ceilometer)**. Este proyecto integrado de OpenStack, permite el uso de métricas para tener un control sobre los usos de recursos computacionales para generar facturación.

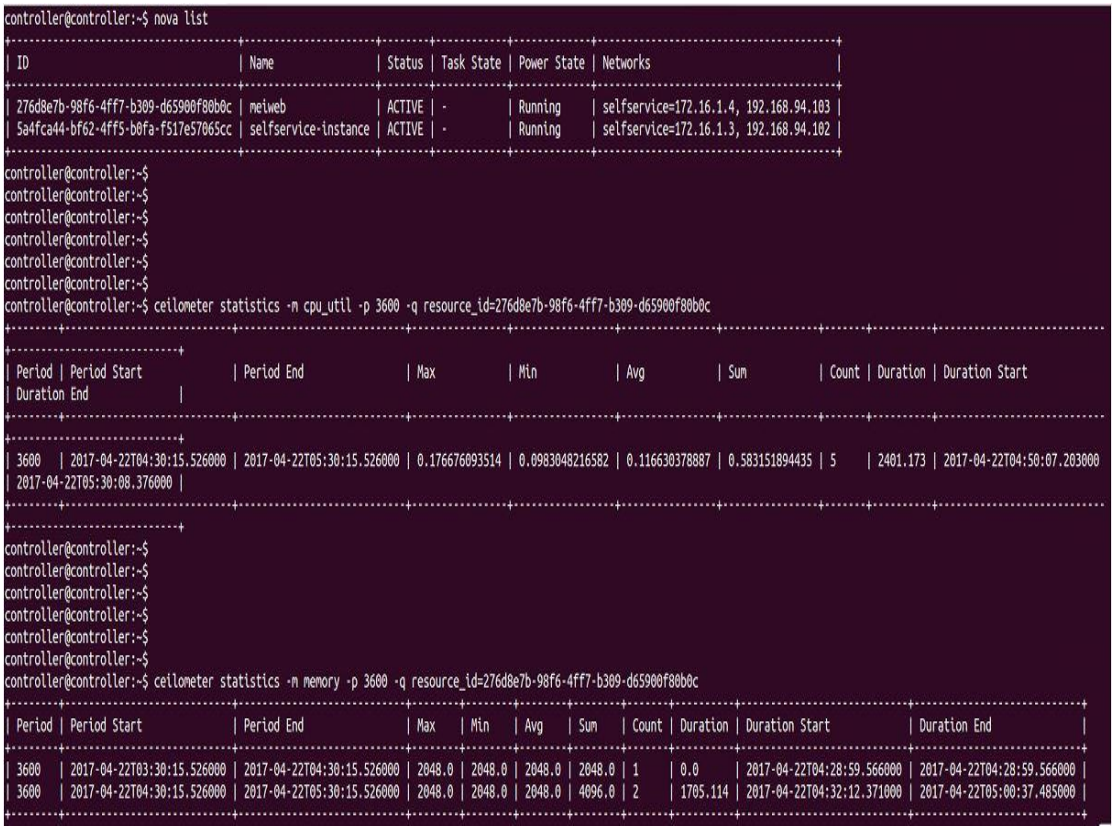
Este módulo brinda la posibilidad de medir de forma general el gasto de estos de cada usuario y también por línea de comandos nos permite ver de manera específica el tipo de recurso utilizado por una instancia particular.

Imagen 36: “Vista de general del módulo de telemetría”




En la imagen anterior se evidencia la vista general de la utilización de recursos, en este caso se tienen dos instancias.

Imagen 37: “Telemetría por línea de comandos”

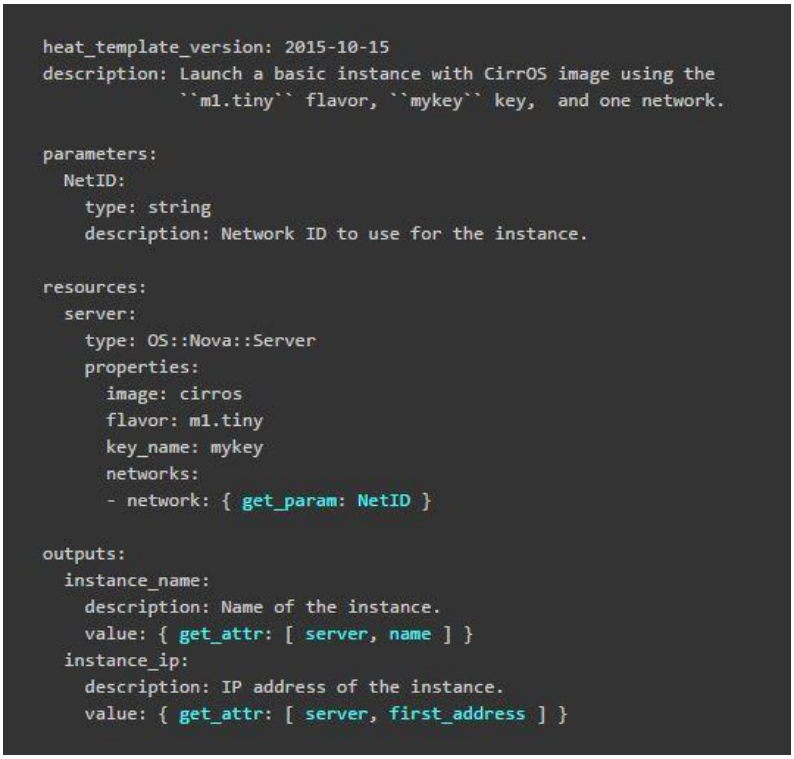



En la imagen anterior se puede ver la madición de una instancia particular, para ello se listan las instancias que están activas y posteriormente con el id de la instancia se general los resultados de las mediciones con el comando ceilometer.

En este caso se evidencia la medición de cpu y memoeria ram de 60 minutos de una instancia especifica.

### 5.4.6 Oquestación (heat)

Imagen 38: “Plantilla básica de orquestación”



```
heat_template_version: 2015-10-15
description: Launch a basic instance with CirrOS image using the
             ``m1.tiny`` flavor, ``mykey`` key,  and one network.

parameters:
  NetID:
    type: string
    description: Network ID to use for the instance.

resources:
  server:
    type: OS::Nova::Server
    properties:
      image: cirros
      flavor: m1.tiny
      key_name: mykey
      networks:
      - network: { get_param: NetID }

outputs:
  instance_name:
    description: Name of the instance.
    value: { get_attr: [ server, name ] }
  instance_ip:
    description: IP address of the instance.
    value: { get_attr: [ server, first_address ] }
```

Como se mencionó anteriormente la creación de instancias requieren una serie de procedimientos, selección de imagen, red interna, IP flotante para acceder a la red pública y a internet, selección de tipo o sabor que nos define los requerimientos de hardware para dicha instancia, así como también la definición de volúmenes en bloque si se requiere.

En la imagen anterior se muestra el formato de una plantilla de orquestación básica, en la cual al importar el archivo se generará automáticamente una instancia que tendrá las siguientes características.

- **Servidor**: selecciona la zona de disponibilidad para la creación de la instancia.

- **Imagen**: cirros.
- **Red**: se le asignará una IP de la red interna para posteriormente asignarle una IP flotante.
- **Tipo o sabor**: m1.titny, este tipo esta previamente creado en la sesión de administrador y disponible para su uso.
- **Key_name**: nos permitirá acceder a las llaves ssh para la conexión remota.

Esta plantilla, aunque básica con algunas modificaciones permitirá crear proyectos más completos con diferentes características de acuerdo a las necesidades de los usuarios.

## 5.5 DOCUMENTACIÓN ADMINISTRATIVA

Con el fin de mantener la infraestructura y asegurar su estabilidad se crearon manuales administrativos para facilitar que futuros administradores puedan ofrecer soporte de manera óptima a la misma.

# 6. CONCLUSIONES

Con la actualización e implementación de los nuevos módulos de Openstack se consiguió aprovechar mejor el uso de los recursos y brindar un servicio de computación en la nube con nuevas y mejores características que permiten el desarrollo de proyectos que van de acuerdo al crecimiento y escalabilidad de la infraestructura.

La implementación del firewall Pfsense brinda una mayor seguridad a la red de máquinas virtuales permitiendo tener un control total de la misma, evitando que agentes externos puedan vulnerar la integridad de los proyectos que están utilizando los servicios que ofrece el grupo, generando confianza en los usuarios que adquieren los servicios computacionales del mismo.

La utilización de ntop como paquete integrado con el firewall permitió hacer seguimiento continuo a las máquinas virtuales adquiridas por los usuarios y facilitó el monitoreo de la red para de esta forma tomar medidas pertinentes en casos que comprometan la seguridad de la infraestructura.

Se realizaron pruebas con un nuevo sistema de replicación de archivos llamado GlusterFS, comparándolo con DRBD que es el sistema de replicación actualmente implementado y se observó que este sistema de replicación se ajusta mejor a las necesidades de la infraestructura, permitiendo mayor escalabilidad.

Se comprobó la viabilidad que presenta el uso de contenedores Linux como complemento a la virtualización mostrando que puede aprovechar mejor los recursos mejorando la administración y despliegue de medianos o pequeños proyectos.

Existe gran variedad de herramientas de software libre como el firewall PfSense o el mismo Openstack que pueden reducir considerablemente los gastos en licencias. Al usar estas tecnologías, se le da al sistema un gran valor agregado y alto nivel de acoplamiento a los intereses y la demanda de las organizaciones.

# 7. RECOMENDACIONES

Continuar con el estudio de los contenedores Linux, puesto que con las pruebas realizadas se observó que esta tecnología puede hacer un mejor uso de los recursos de la infraestructura del grupo GID-CONUSS.

Investigar el uso de contenedores como servicio y su integración con OpenStack para agilizar el despliegue de proyectos.

Migrar el sistema de replicación de archivos. Pasar de DRBD a GlusterFS, para buscar una mayor escalabilidad en la infraestructura.

Generar un nuevo proyecto para actualizar la arquitectura del sistema operativo base de la plataforma.

Incluir nuevo hardware (servidores físicos) con el fin de implementar Openstack en alta disponibilidad con miras a ofrecer un servicio de manera más profesional en la que entidades externas se puedan interesar.

Continuar explorando Openstack, ya existe una versión más reciente a la implementada en este proyecto.

Sugerir un proyecto dedicado a Openstack, para poder profundizar en cada uno de sus módulos y así sacar provecho de la gran variedad de herramientas que estos ofrecen.

Implementar un sistema de copias de seguridad que permita garantizar la integridad y disponibilidad de las mismas de una manera más eficiente que la implementada hasta el momento.

Se sugiere seguir por la vía del software libre, siendo esta la característica principal que ha identificado al grupo GID-CONUSS desde su creación.

Promover la creación de un laboratorio de computación en la nube que permitan hacer pruebas controladas, buscando que se cree una asignatura en el campo de la computación en la nube para estudiantes de la escuela de ingeniería de sistemas.

## REFERENCIAS BIBLIOGRÁFICAS


[1] CARREÑO DÍAZ, Emmanuel. Modelo y prototipo de servicios de computación en la nube para estudiantes y profesores de la escuela de ingeniería de Sistemas e informática de la universidad industrial de Santander. Bucaramanga. Universidad industrial de Santander. Facultad de Ingenierías Físico-mecánicas. Escuela de ingeniería de sistemas e informática, 2012.

[2] Cloudbus.org. IEEE Task Force on Cluster Computing - High Availability. [online] Available at: http://www.cloudbus.org/~raj/tfcc/high-availability.html [Recuperado en: Nov. 2016].

[3] Doc.PfSense.org. PfSenseDocs. [online] Available at: https://doc.PfSense.org/index.php/Main_Page#Welcome_to_the_PfSense_Documentation_site [Recuperado en Dic - 2016].

[4] Docs.endian.com. Endian UTM 5.0 Reference Manual - Endian UTM 5.0 Reference Manual. [online] Available at: http://docs.endian.com/5.0/utm/index.html [Recuperado en Dic - 2016].

[5] Docs.openstack.org. (2016). OpenStack Docs: Identity service overview. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_identity.html [Recuperado en: Nov. 2016].

[6] Docs.openstack.org. (2016). OpenStack Docs: Image service. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/glance.html [Recuperado en: Nov. 2016].

[7] Firewalls - definición", Spi1.nisu.org. [En línea]. Disponible en: http://spi1.nisu.org/recop/al01/rmoreno/definicion.html [Recuperado en: Dic. 2016].

[8] Gluster Docs, Gluster.readthedocs.io, 2017. [En línea]. Disponible en: https://gluster.readthedocs.io/en/latest/. [Recuperado en: Ene- 2017].

[9] Gluster File System, Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Gluster_File_System. [Recuperado en: Ene- 2017].

[10] Jones, Why Docker?, via @codeship, 2016. [En línea]. Disponible en: https://blog.codeship.com/why-docker/. . [Recuperado en: Feb- 2017].

[11] LIZARAZO TORRES, John Edinson. Administración, mantenimiento, configuración y monitoreo de los equipos servidores del grupo gid-conuss con énfasis en el análisis y reestructuración de los modelos de alta disponibilidad y computación en la nube. Facultad de Ingenierías Físico-mecánicas. Escuela de ingeniería de sistemas e informática, 2013.

[12] LXC, Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/LXC. [Recuperado en Feb- 2017].

[13] Máquina virtual", Docker.com, 2016. [En línea]. Disponible en: https://www.docker.com/what-container [Recuperado en: Dic- 2016].

[14] Ntop, Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Ntop. [Recuperado en: Dic- 2016].

[15] OpenStack Docs: Block Storage service overview, Docs.openstack.org, 2016. [En línea]. Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_block_storage.html. [Recuperado en: Nov- 2016].

[16] OpenStack Docs: Compute service overview", Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_compute.html. [Recuperado en: Nov. 2016].

[17] OpenStack Docs: Mitaka, Docs.openstack.org, 2016. [En línea]. Disponible en: https://docs.openstack.org/mitaka/. [Recuperado en: Nov- 2016].

[18] OpenStack Docs: Networking (neutron) concepts", Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/neutron-concepts.html. [Recuperado en: Nov. 2016].

[19] OpenStack Docs: Orchestration service overview, Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_orchestration.html. [Recuperado en: Nov. 2016].

[20] P. Mell and T. Grance, “The NIST definition of cloud computing,” NIST Spec. Publ., vol. 145, p. 7, 2011. Available at: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

[21] M. Joy, Performance comparison between Linux containers and virtual machines, Conference Proceeding - 2015 International Conference on Advances in Computer Engineering and Applications, ICACEA 2015, p. 245, 2015.

[22] What is Docker, Docker. [En línea]. Disponible en: https://www.docker.com/what-docker. [Recuperado en: Feb- 2017].

## BIBLIOGRAFÍA


CARREÑO DÍAZ, Emmanuel. Modelo y prototipo de servicios de computación en la nube para estudiantes y profesores de la escuela de ingeniería de Sistemas e informática de la universidad industrial de Santander. Bucaramanga. Universidad industrial de Santander. Facultad de Ingenierías Físico-mecánicas. Escuela de ingeniería de sistemas e informática, 2012.

Cloudbus.org. IEEE Task Force on Cluster Computing - High Availability. [Online] Available at: http://www.cloudbus.org/~raj/tfcc/high-availability.html [Recuperado en: Nov. 2016].

Doc.PfSense.org. PfSenseDocs. [Online] Available at: https://doc.PfSense.org/index.php/Main_Page#Welcome_to_the_PfSense_Documentation_site [Recuperado en Dic - 2016].

Docs.endian.com. Endian UTM 5.0 Reference Manual — Endian UTM 5.0 Reference Manual. [online] Available at: http://docs.endian.com/5.0/utm/index.html [Recuperado en Dic - 2016].

Docs.openstack.org. (2016). OpenStack Docs: Identity service overview. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_identity.html [Recuperado en: Nov. 2016].

Docs.openstack.org. (2016). OpenStack Docs: Image service. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/glance.html [Recuperado en: Nov. 2016].

Firewalls - definición", Spi1.nisu.org. [En línea]. Disponible en: http://spi1.nisu.org/recop/al01/rmoreno/definicion.html [Recuperado en: Dic. 2016].

Gluster Docs, Gluster.readthedocs.io, 2017. [En línea]. Disponible en: https://gluster.readthedocs.io/en/latest/. [Recuperado en: Ene- 2017].

Gluster File System, Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Gluster_File_System. [Recuperado en: Ene- 2017].

Jones, Why Docker?, via @codeship, 2016. [En línea]. Disponible en: https://blog.codeship.com/why-docker/. . [Recuperado en: Feb- 2017].

LIZARAZO TORRES, John Edinson. Administración, mantenimiento, configuración y monitoreo de los equipos servidores del grupo gid-conuss con énfasis en el análisis y reestructuración de los modelos de alta disponibilidad y computación en la nube. Facultad de Ingenierías Físico-mecánicas. Escuela de ingeniería de sistemas e informática, 2013.

LXC, Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/LXC. [Recuperado en Feb- 2017].

M. Joy, Performance comparison between Linux containers and virtual machines, Conference Proceeding - 2015 International Conference on Advances in Computer Engineering and Applications, ICACEA 2015, p. 245, 2015.

Máquina virtual", Docker.com, 2016. [En línea]. Disponible en: https://www.docker.com/what-container [Recuperado en: Dic- 2016].

Ntop, Es.wikipedia.org, 2016. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Ntop. [Recuperado en: Dic- 2016].

OpenStack Docs: Block Storage service overview, Docs.openstack.org, 2016. [En línea]. Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_block_storage.html. [Recuperado en: Nov- 2016].

OpenStack Docs: Compute service overview", Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_compute.html. [Recuperado en: Nov. 2016].

OpenStack Docs: Mitaka, Docs.openstack.org, 2016. [En línea]. Disponible en: https://docs.openstack.org/mitaka/. [Recuperado en: Nov- 2016].

OpenStack Docs: Networking (neutron) concepts", Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/neutron-concepts.html. [Recuperado en: Nov. 2016].

OpenStack Docs: Orchestration service overview, Docs.openstack.org, 2016. [En línea] Disponible en: https://docs.openstack.org/mitaka/install-guide-ubuntu/common/get_started_orchestration.html. [Recuperado en: Nov. 2016].

P. Mell and T. Grance, “The NIST definition of cloud computing,” NIST Spec. Publ., vol. 145, p. 7, 2011. Available at: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

What is Docker, Docker. [En línea]. Disponible en: https://www.docker.com/what-docker. [Recuperado en: Feb- 2017].