Grado en Ingeniería Informática

Trabajo Final de Grado

Configuración e Implementación de una Infraestructura Cloud Computing Privada

Supervisor:
Jaume Matamoros Rosa

Autor:
Pedro Segarra Cabedo

Tutor académico:
Rafael Mayo Gual

Fecha de lectura: 28 de Octubre de 2014
Curso académico 2013/2014

Resumen

Cloud computing (computación en la nube) es un paradigma de computación que permite ofrecer recursos de computación a través de la red. Estos recursos se ofrecen bajo demanda y pueden ser suministrados con una mínima interacción y gestión por parte del proveedor del servicio.

La computación en la nube posee algunas características esenciales, destacando la elasticidad, es decir el rápido escalado de los recursos del sistema de forma horizontal. Además todos los elementos que pueden ser "virtuales" se ofrecen como servicios.

Existen tres modelos de servicios (Software como servicio (SaaS), Plataforma como Servicio (PaaS) e Infraestructura como Servicio (IaaS)).

Otra característica destacada de la computación en la nube es que consta de cuatro tipos de despliegues, Cloud Privada, Cloud Pública, Cloud Comunitaria y Cloud Híbrida.

El Trabajo Final de Grado cubre el estudio de una Infraestructura de Cloud computing Privada. En concreto se ha realizado la configuración e implementación del sistema bajo la plataforma open source "OpenStack". Dicha plataforma permite configurar e implementar una Infraestructura como Servicio en la nube (IaaS) sobre un hardware básico.

OpenStack maneja modularmente todos los servicios de la plataforma cloud computing (cómputo, almacenamiento y red), permitiendo disponibilidad, escalabilidad y fiabilidad de la infraestructura.

Palabras Clave

OpenStack, Nube, Cómputo, Red, Almacenamiento, Software como Servicio, Plataforma como Servicio, Infraestructura como Servicio.

Keywords

OpenStack, Cloud, Compute, Networking, Storage, Software as a Service, Platform as a Service, Infrastructure as a Service.

Índice

Índice de Tablas

Índice de Figuras

Capítulo 1

1. Introducción

1.1. Contexto y motivación del proyecto

1.1.1. Contextualización del proyecto

Cloud computing [1] (computación en la nube) es un paradigma de computación que permite ofrecer recursos de computación a través de la red. Estos recursos (redes, almacenamiento, cómputo, servidores, aplicaciones, servicios) se ofrecen bajo demanda y pueden ser suministrados con una mínima interacción y gestión por parte del proveedor del servicio.

La computación en la nube se compone de cinco características esenciales, tres modelos de servicios y cuatro tipos de despliegues.

Características esenciales:

- Servicio disponible de forma **automática** y bajo **demanda**.
- Acceso a través de la **red**. Ofrece servicio a múltiples plataformas heterogéneas (móviles, servidores, tablets, portátiles...)
- Modelo **multi-tenant**. Se comparten los recursos con otros usuarios, garantizándose el aislamiento y seguridad entre los usuarios.
- **Elasticidad**. Escalar y reducir el sistema rápidamente.
- Optimización de los recursos. **Pago por uso**

Modelos de Servicios:

- **Software como Servicio** (SaaS). Aplicación como servicio en la nube. El usuario utiliza una aplicación a través de la red en lugar de tenerla instalada en el propio equipo.

  Ejemplo: Google Apps, DropBox.

- **Plataforma como Servicio** (PaaS). Una empresa ofrece servicios a terceros proporcionando sencillez de mantenimiento, ya que la empresa es la encargada de gestionar la instalación, configuración y mantenimiento de la plataforma. Está limitada por el software del proveedor.

La utilización de PaaS suele ser realizada por un desarrollador o pequeña empresa que desea tener una aplicación web escalable y en pago por uso. No se disponen de los conocimientos técnicos adecuados para realizar la implantación de un cloud por sí mismo o no se dispone de tiempo para dedicar a formación para alcanzarlo.

Ejemplos de PaaS: Google App Engine, Windows Azure, Heroku, OpenShift.

- **Infraestructura como Servicio** (IasS). Se ofrece un control completo sobre la plataforma. Es utilizada por Administradores de Sistemas, que son los encargados de la configuración, actualización y mantenimiento de la plataforma. Se dispone de procesamiento, almacenamiento, capacidad de red y otros recursos computacionales a medida.

  Ejemplos de IaaS: Amazon Web Services, Rackspace Cloud Servers, Google Compute Engine.

La figura 1 muestra gráficamente los modelos de servicios en la nube, mostrando a que usuarios va dirigido cada modelo y las principales empresas que destacan en cada modelo se servicio.

Figura 1.- Comparativa modelos de servicios, con usuario asociado y empresa. Fuente: Denoe [2]

Tipos de despliegues:

- **Cloud Privada**. La infraestructura en la nube está preparada para el uso exclusivo de una sola organización. Puede ser administrada por la empresa o por terceros y puede existir dentro o fuera de las instalaciones de la empresa.
- **Cloud Pública**. La infraestructura en la nube está preparada para el uso abierto al público en general. Una empresa ofrece servicios a terceros encargándose de toda la gestión del Cloud.
- **Cloud Comunitaria**. La infraestructura en la nube está preparada para el uso exclusivo de una determinada comunidad de consumidores.
- **Cloud Híbrida**. La infraestructura de la nube es una composición de dos o más nubes (privadas, comunitarias o públicas).

En la figura 2 se resumen todas las características del cloud computing vistas anteriormente.

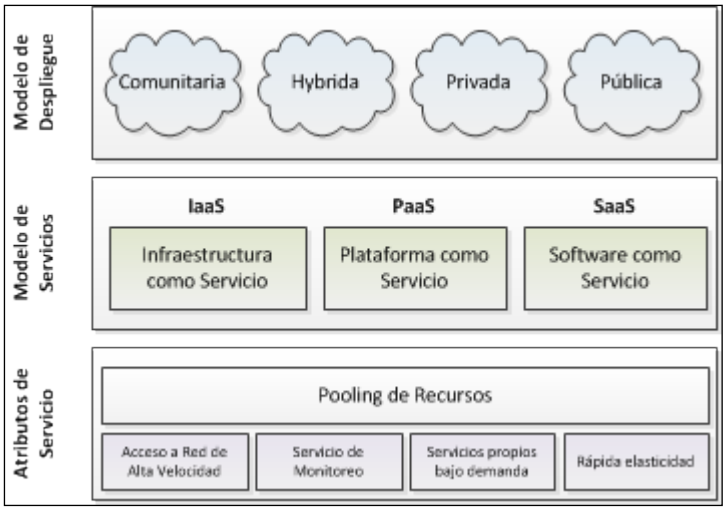


Figura 2.- Resumen de las características de Cloud computing. Fuente: Mentevisual [3]

El TFG (Trabajo Final de Grado), está basado en la configuración e implementación de una Infraestructura de Cloud Computing Privada. Es decir de los tipos de despliegue vistos anteriormente, se corresponde al de Cloud Privada, la infraestructura está preparada para uso exclusivo de una sola organización. El modelo de Servicio se basa en el de Infraestructura como Servicio (IaaS), ofreciendo un control total de la plataforma al administrador. El despliegue de esta infraestructura se ha realizado a través de la plataforma OpenStack; que utilizando

software libre, realiza la construcción de una Infraestructura como Servicio (IaaS) sobre hardware básico.

1.1.2. Motivaciones del proyecto

Una de las motivaciones del proyecto ha consistido en dar a conocer la funcionalidad y el alcance que ofrece OpenStack como proyecto de computación en la nube; además de entender la diferencia entre la implementación de una infraestructura clásica de máquinas físicas o virtuales frente a la infraestructura en la nube.

En el enfoque tradicional de implementación con máquina física existen muchos recursos desaprovechados, en contraposición con el enfoque de computación en la nube, donde los recursos se comparten globalmente, permitiendo elasticidad, reducción de costes y nuevos medios de almacenamiento.

En la figura 3 se muestra las principales ventajas de la computación en la nube .

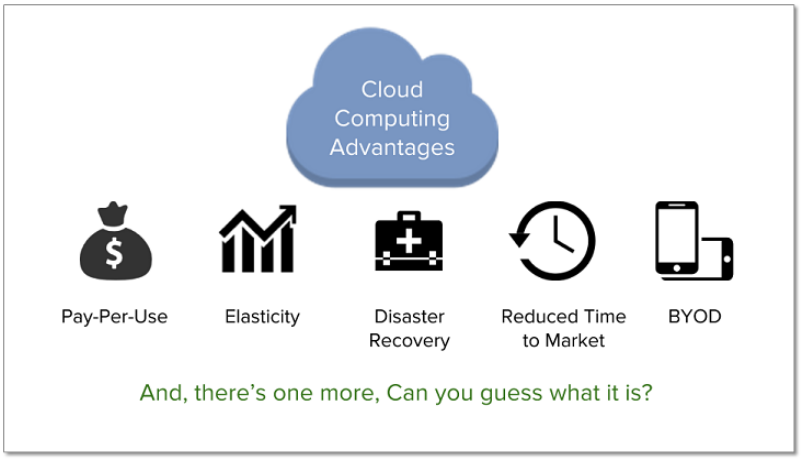


Figura 3.- Ventajas computación en la nube. Fuente:blog.flux7.com [4]

Otra de las motivaciones es trabajar con una plataforma Cloud Computing "*open source*", ya que OpenStack es una plataforma de cloud computing cuyo desarrollo se basa en Licencia Apache 2.0, por lo tanto es libre, y no hace uso de la versión "Enterprise", es decir no tiene funcionalidades de pago. Tanto el proceso de diseño y desarrollo es abierto; además se puede acceder al código fuente, descargarlo, modificarlo y realizar aportaciones.

Adicionalmente, el proyecto OpenStack está implementado bajo la metodología de desarrollo ágil y el desarrollo e implementación se basan en estándares abiertos.

1.2. Objetivos del proyecto

1.2.1. Objetivo principal del proyecto

El objetivo principal del proyecto ha consistido en la realización de un estudio de las posibilidades que ofrece la plataforma OpenStack [5]. Para ello se ha realizado un recorrido por las diferentes variantes de OpenStack. De esta forma nos facilita un aprendizaje progresivo sobre esta infraestructura de Cloud Computing. Se han desarrollado implementaciones mediante: TryStack, DevStack y OpenStack.

En los siguientes apartados de la memoria se explicará la diferencia entre las diferentes implementaciones de OpenStack y el uso que se puede dar a cada una de ellas.

1.2.2. Objetivos específicos del proyecto.

Una vez se ha conocido la infraestructura OpenStack, se ven claramente varios frentes de trabajo sobre esta plataforma, por un lado existe la gestión de la plataforma, saber cómo manejar las máquinas virtuales, sacarle rendimiento a la infraestructura; y por otro lado la implementación de la infraestructura necesaria para manejar el número de máquinas virtuales requerido. Un tercer frente se ha basado en la implementación de las aplicaciones o servicios que el cliente, el instituto en el caso del proyecto requiere.

En concreto, se requiere la implementación rápida de servicios tales como Moodle, Wordpress, Joomla, Owncloud y un repositorio para almacenar grandes volúmenes de información como vídeos, imágenes, exámenes escaneados para uso por parte del profesorado.

Existe también la necesidad de generar entornos personalizados para los alumnos, que éstos puedan implementar instalaciones de sistemas operativos, puedan hacer uso de aplicaciones específicas. Además todo ello se debe realizar de forma automática, rápida y escalable.

Capítulo 2

2. Descripción del proyecto.

2.1. Introducción

Como se ha comentado en los apartados anteriores el proyecto consiste en la configuración e implementación de una infraestructura Cloud Computing Privada. En concreto trata sobre el estudio, configuración e implantación de la Plataforma OpenStack como Infraestructura como Servicio (IaaS).

La elección de la Infraestructura como Servicio (IaaS) es debido a que de este modo tendremos el mayor control sobre la plataforma. Como se ha comentado en la definición de los modelos de servicios y en los tipos de despliegues, la infraestructura IaaS es la que proporciona control completo sobre la plataforma y el objetivo de este proyecto es la realización de esta implementación. El tipo de despliegue es privado ya que se va a realizar un uso exclusivo en la organización, en este caso por parte del instituto.

2.2. Justificación de la selección como plataforma IaaS-OpenStack

Cuando se aborda un proyecto de Infraestructura como Servicio (IaaS), enseguida surge la idea de realizar el despligue sobre los proveedores de servicio de pago más importantes como son Amazon Web Services, Rackspace Cloud Servers, Microsoft Windows Azure o el recién incorporado Google Compute Engine. Pero la filosofía del proyecto es que se realice mediante una plataforma cloud de software libre.

Además de OpenStack existen otros proveedores como Ecualyptus, OpenNebula, CloudStack que también son plataformas "open source", pero se ha decantado en realizar la implantación sobre OpenStack debido a los conocimientos previos sobre esta plataforma, la compatilidad con Amazon Web Services (AWS), y la apuesta que están realizando las principales empresas en el desarrollo de OpenStack.

En la figura 4 se muestran los principales vendedores que dan apoyo a OpenStack. En concreto se muestra el top 3 de los vendedores en algunas de las diferentes categorías (routers, switches, blades, almacenamiento, hipervisores, arquitectura, sistema operativos linux).

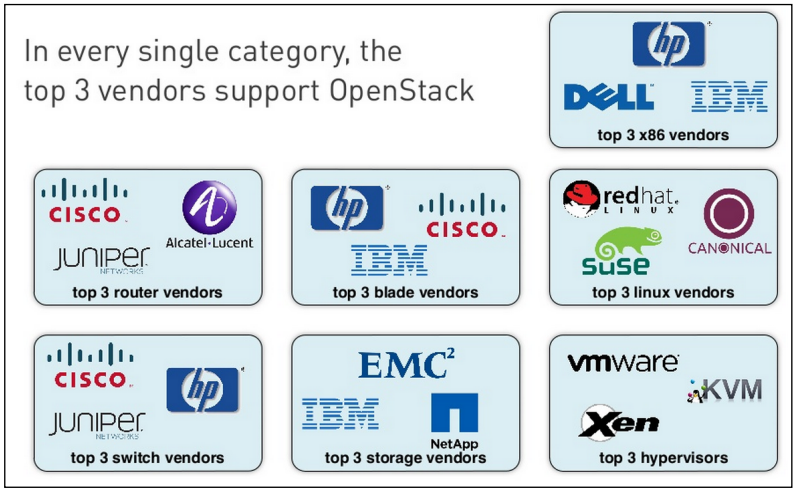


Figura 4. Principales vendedores en diferentes categorías. Fuente: CloudScaling [6]

2.3. Lugar de implantación y alcance del proyecto

2.3.1. Lugar de implantación del proyecto.

La implantación del proyecto se ha realizado en el Instituto de Enseñanza Secundaria "Ies Broch i Llop" de Vilareal. Como hemos visto en el apartado "*1.2 Objetivos del proyecto*", el objetivo principal del proyecto es el estudio de la plataforma OpenStack.

La coordinación de TI del departamento de Informática del "Broch i Llop" está interesado en conocer la plataforma OpenStack. El centro necesita que se realice un estudio de la infraestructura OpenStack para ver si cubre las necesidades que actualmente demanda el profesorado.

Estas necesitades están entre las siguientes: ampliar la capacidad de almacenamiento de moodle, tener acceso a blogs personalizados y de fácil uso, alojar vídeos, imágenes, exámenes escaneados en un repositorio y que el acceso sea lo más sencillo posible.

Por otro lado, la creación de máquinas o espacios específicos para que el alumno conozca los diferentes sistemas de blogs; puedan crear páginas webs en un entorno cerrado y centralizado, conozcan como configurar diferentes sistemas operativos, tengan acceso a un repositorio seguro para descargar imágenes, vídeos filtrados, etc.

2.3.2. Alcance del proyecto

2.3.2.1. Introducción a los servicios de OpenStack

En los apartados posteriores de la memoria se verán las principales características de OpenStack, y veremos que OpenStack está diseñado para ser masivamente escalable horizontalmente, lo que permite que todos los servicios sean distribuidos ampliamente. En la Figura 5 se muestran los principales servicios de OpenStack: cómputo (Compute), almacenamiento (Storage), y recursos de red (Networking), además todos ellos se pueden gestionar a través del cuadro de mando (Horizon), que es una aplicación web para gestión de la infraestructura.

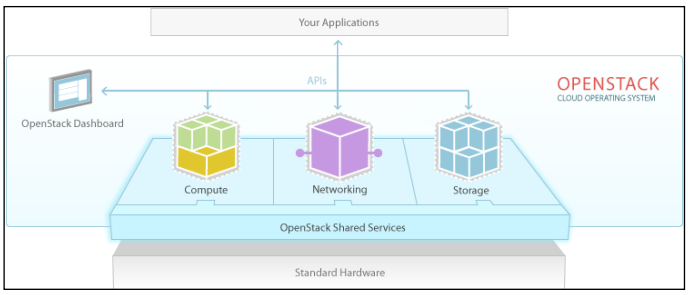


Figura 5. Funcionalidades de OpenStack. Fuente:OpenStack [5]

2.3.2.2. Descripción de las variantes de implementación de OpenStack

Cuando se empieza a entender las funcionalidades y posibilidades que brinda OpenStack, lo primero que nos damos cuenta, es la falta de recursos disponibles en nuestra organización para abordar el proyecto, es decir; no vamos a poder implementar una plataforma elástica, aunque aumentemos un poco la RAM de nuestro ordenador o conectemos un disco duro. Es por ello que la opción de utilizar la plataforma TryStack (podemos acceder de forma online a una implementación de OpenStack para realizar pruebas, sin poseer la infraestructura) es muy importante, ya que nos proporciona de forma gratuita la posibilidad de tener una infraestructura escalable, para un prototipado inicial de nuestro entorno.

TryStack no sirve para establecer un entorno en producción ya que sólo permite tener la infraestructura disponible 24 horas; pero nos ayuda en la gestión de la infraestructura, configuración de las instancias, volúmenes, etc.

También se va a trabajar sobre la plataforma DevStack. Esta plataforma nos permite realizar un montaje rápido de OpenStack, aproximadamente en 5 minutos, sobre una máquina virtual o sobre una máquina física. De esta forma se aprenderá como realizar la gestión de la plataforma y se podrá probar e interactuar sobre el código de OpenStack. Es una plataforma que está pensada para el desarrollador.

Como últimos pasos de aprendizaje sobre la implementación de la plataforma, está la automatización de la instalación de OpenStack, que se puede realizar mediante herramientas como Vagrant y Ansible, y para finalizar podemos realizar la implementación de OpenStack directamente sobre la máquina física o apoyándose en máquinas virtuales.

Esta última implementación, la de configurar la máquina paso a paso siguiendo la guía de instalación de OpenStack [7] , es la que aporta más conocimiento a bajo nivel de la infraestructura y nos acerca a entender la infraestructura OpenStack. Es por ello, que el aprendizaje de las diferentes implementaciones no ha seguido un orden estricto. Sino que se ha ido probando sobre TryStack que aporta un rápido conocimiento sobre gestión y alternando sobre la instalación paso a paso, pruebas con DevStack y automatizaciones.

2.3.2.3. Instalación de aplicaciones específicas.

Una vez conocido OpenStack, el objetivo principal del proyecto es proveer al Instituto de una plataforma ágil para una implementación rápida de aplicaciones y servicios destacando la instalación de Moodle, Wordpress, Owncloud. Así como la generación rápida de máquinas para que los alumnos aprendan con un sistema operativo específico , con aplicaciones concretas, con implementaciones de ISO; para que puedan acceder a una plataforma a medida en un breve intervalo de tiempo.

Para realizar todas estas implementaciones se ha requerido del estudio de las guías de instalación, de arquitectura de OpenStack. Existen también otras guías más específicas sobre Alta disponibilidad y Seguridad que no se tratan en este proyecto, pero son muy interesantes por si se quiere implementar un proyecto que requiera de estas características específicas.

Por otro lado se ha realizado el estudio de las guías de instalación de las aplicaciones Apache, Nginx, Mysql, Moodle, Wordpress, OwnCloud, Joomla, y de diferentes sistemas operativos para realizar correctamente su instalación.

Capítulo 3

3. Planificación del proyecto

3.1. Definición de tareas

3.1.1. Introducción

El primer paso para establecer la planificación del proyecto, consiste en definir o establecer la lista de tareas necesarias para cubrir las necesidades demandadas por la coordinación TI del centro, en base a las peticiones del profesorado y para mejorar la impartición de las clases a los alumnos.

Para lograr la implantación real en el centro de una infraestructura OpenStack, se necesitaría una infraestructura bastante amplia, y separar cada recurso (cómputo, red, almacenamiento) en un servidor específico, pero dada la naturaleza de investigación y estudio de este proyecto, las pruebas se realizan en un único equipo con 8GB de Ram , dos procesadores y 1TB de disco duro. Aunque un requisito que debe cumplir el equipo es que permita la virtualización VT-x.

Se comentará en el apartado "*3.3 recursos del proyecto*" la infraestructura que se requeriría para una implantación acorde a las necesidades reales del centro.

3.1.2. Listado de tareas iniciales del proyecto

A continuación, en la tabla 1 se muestra un desglose de las tareas propuestas inicialmente para alcanzar los objetivos propuestos del proyecto.

Tarea	Descripción
1	Adquisición de una máquina para realizar el prototipado de la implementación IaaS con OpenStack. (Conseguir como mínimo 8GB de Ram para conseguir realizar simulaciones de OpenStack razonables).
2	Instalación de sistemas operativos adecuados para la nube. A elegir o decidir entre Ubuntu 12.04 LTS, Fedora 20, CentOs/RHEL 6.5, OpenSuse o Debian.
3	Testeo de la plataforma con TryStack (Entorno de pruebas).
4	Instalación de una infraestructura para el desarrollo con DevStack sobre máquina virtual y sobre hardware.

5	Instalación de OpenStack mediante Vagrant y Ansible sobre máquinas virtuales.
6	Implementación de OpenStack paso a paso sobre máquinas virtuales
7	Estudio de los diferentes módulos o servicios de OpenStack y ver la funcionalidad que ofrecen
8	Instalación de instancias con la aplicación, Wordpress, Moodle, Owncloud. Instalación de instancias con diferentes sistemas operativos (con ISO, Windows, específicos).
9	Gestión de OpenStack sobre interfaz Web Horizon.
10	Estudio de la API de OpenStack mediante la línea de comandos. Estudio de python como gestión de la API (tareas automatizadas).

Tabla 1. Listado de tareas iniciales del proyecto.

3.2. Planificación temporal de las tareas

3.2.1. Introducción

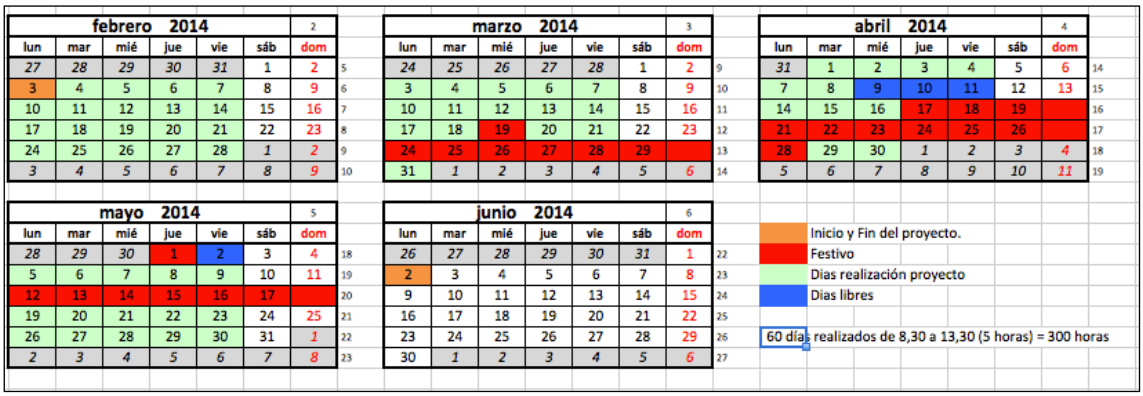
En la primera reunión entre el tutor, supervisor y el alumno, se estimó una dedicación de 6 horas diarias al proyecto. Lo que hacía que la fecha de finalización del mismo fuera aproximadamente sobre el 14 de Abril de 2014.

Pero la dedicación diaría establecida inicialmente de 6 horas se redujo a 5 horas, ya que de esta manera, se podía llegar a las clases de la Uji tranquilamente. Al reducir las horas en el centro, la planificación ha tenido en cuenta las fiestas locales de Castellón y Vilareal, y las fiestas de Semana Santa. Además se ha concedio al alumno algunos días libres.

Con esta nueva planificación horaria, el proyecto ha finalizado el día 2 de Junio de 2014, con la última reunión entre el tutor, supervisor y alumno. En el siguiente apartado se detalla el calendario del proyecto, con las fechas de realización del mismo, la fecha de inicio y fin del proyecto, las fechas de vacaciones y los días libres.

3.2.2. Calendario del proyecto

En la figura 6 se muestra el calendario de realización del proyecto. En total han sido destinadas 300 horas repartidas en 60 días.



febrero 2014							2
lun	mar	mié	jue	vie	sáb	dom	
27	28	29	30	31	1	2	5
3	4	5	6	7	8	9	6
10	11	12	13	14	15	16	7
17	18	19	20	21	22	23	8
24	25	26	27	28	1	2	9
3	4	5	6	7	8	9	10

marzo 2014							3
lun	mar	mié	jue	vie	sáb	dom	
24	25	26	27	28	1	2	9
3	4	5	6	7	8	9	10
10	11	12	13	14	15	16	11
17	18	19	20	21	22	23	12
24	25	26	27	28	29		13
31	1	2	3	4	5	6	14

abril 2014							4
lun	mar	mié	jue	vie	sáb	dom	
31	1	2	3	4	5	6	14
7	8	9	10	11	12	13	15
14	15	16	17	18	19		16
21	22	23	24	25	26		17
28	29	30	1	2	3	4	18
5	6	7	8	9	10	11	19

mayo 2014							5
lun	mar	mié	jue	vie	sáb	dom	
28	29	30	1	2	3	4	18
5	6	7	8	9	10	11	19
12	13	14	15	16	17		20
19	20	21	22	23	24	25	21
26	27	28	29	30	31	1	22
2	3	4	5	6	7	8	23

junio 2014							6
lun	mar	mié	jue	vie	sáb	dom	
26	27	28	29	30	31	1	22
2	3	4	5	6	7	8	23
9	10	11	12	13	14	15	24
16	17	18	19	20	21	22	25
23	24	25	26	27	28	29	26
30	1	2	3	4	5	6	27

Inicio y Fin del proyecto.
Festivo
Dias realización proyecto
Dias libres

60 días realizados de 8,30 a 13,30 (5 horas) = 300 horas

Figura 6.- Calendario de ejecución del proyecto.

3.2.3. Descripción de Tareas y temporalización.

La tabla 2 muestra la planificación establecida en el proyecto divididas por tareas, con su descripción, las horas empleadas y las fechas entre las que se ha realizado la actividad.

Tareas	Descripción	Horas	Fechas realización
Reunión Inicial	Se establece la primera reunión entre el tutor, el supervisor y el alumno para abordar la elaboración del proyecto.	2 horas	3 de Febrero
Planificación y Estudio	Planificación del proyecto. Estudio de guías de implementación, configuración y acceso sobre plataforma TryStack, DevStack y OpenStack	45 horas	4 al 14 de Febrero
Adquisición de la máquina y Configuración del Sistema	Recepción de la máquina para realizar el proyecto. Instalación de Sistemas Operativos adecuados para la nube. Pruebas sobre Ubuntu 12.04 LTS y Fedora 20.	25 horas	Del 17 al 21 de Febrero
Entrega de Propuesta Técnica	Hito 1: Entrega de Propuesta Técnica	--	Incluido en fase Planificación

Prototipado con TryStack	Aprender el funcionamiento de OpenStack a través de TryStack. Gestión de los módulos y servicios de OpenStack (TryStack) con la versión Havana. Aprendizaje de la interfaz Web Horizon. Implementación de máquinas virtuales con LAMP (Linux +Apache+Mysql+PHP). Estudio LEMP (Servidor Nginx) Instalación de Wordpress, Moodle, OwnCloud. Pruebas de acceso a las instancias por SSH y por Web.	50 horas	Del 24 al 28 de Febrero. Del 3 al 7 de Marzo.
Implementación de DevStack	Instalación de VirtualBox para el montaje de virtualización. Implementación de DevStack como plataforma de desarrollo OpenStack sobre Máquina virtual. Implementación de DevStack (desarrollo) sobre máquina física. Gestión de DevStack con Interfaz Web Horizon. Estudio de adaptar Horizon al cliente. (logotipo, estructura,etc..) Utilización de la API de OpenStack por línea de comandos.	50 horas	Del 10 al 14 de Marzo. Del 17 al 18 de Marzo. Del 20 al 21 de Marzo. 31 de Marzo.
Automatización de Instalación de OpenStack	Automatización de la Instalación de OpenStack con Vagrant y Ansible**.** Uso de los Repositorios Git Implementación de OpenStack sobre máquinas virtuales. Montajes de diferentes nodos: Controller, Compute, Network. Gestión de Instancias, Volúmenes, Almacenamiento de Objetos.	55 horas	Del 1 al 4 de Abril. Del 7 al 8 de Abril. Del 14 al 16 de Abril**.** Del 29 al 30 de Abril.

Implementación de OpenStack módulo a módulo (servicios)	Implementación de OpenStack sobre máquinas virtuales. (Diferentes nodos) Detalle de la implementación de los diferentes módulos y funcionalidades. Importancia de Keystone, Cola de mensajes, base de datos mysql.	50 horas	Del 5 al 9 de Mayo. Del 19 al 23 de Mayo.
Pruebas y análisis de las Aplicaciones instaladas y Gestión de la plataforma OpenStack	Prueba de las aplicaciones (Moodle, Wordpress,OwnCloud) Acceso a los diferentes sistemas Operativos, pruebas de ISO. Análisis de la Gestión Centralizada a través de Horizon	25 horas	Del 26 al 30 de Mayo
Reunión Final	Reunión final con tutor, supervisor y alumno. Explicación y demostración del proyecto realizado.	2 horas	2 de Junio
Entrega Hito 2	Hito 2: Finalización de la estancia	--	

Tabla 2.- Planificación de las tareas del proyecto.

3.2.4. Diagrama de Gannt

La planificación se puede ver mediante el diagrama de Gannt. En la figura 7 se puede ver una captura de la planificación.

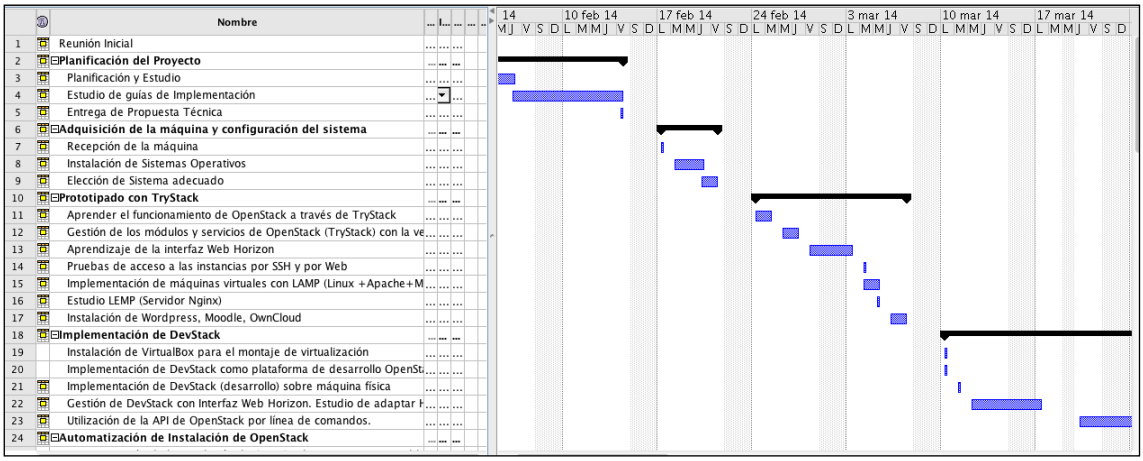


Figura 7.- Una captura de Febrero a Marzo del diagrama de Gannt de Planificación.

La planificación de las tareas se puede ver mejor en la tabla 3. Donde se muestra el nombre de las tareas, y el inicio y fin de las mismas.

		Nombre	Duracion	Inicio	Terminado
1		Reunión Inicial	0,25 days	3/02/14 11:30	3/02/14 13:30
2		**⊟Planificación del Proyecto**	**5,625 day...**	**4/02/14 8:30**	**14/02/14 13:30**
3		Planificación y Estudio	1,875 days?	4/02/14 8:30	6/02/14 13:30
4		Estudio de guías de Implementación	4,375 days?	6/02/14 8:30	14/02/14 13:30
5		Entrega de Propuesta Técnica	0,625 days?	14/02/14 8:30	14/02/14 13:30
6		**⊟Adquisición de la máquina y configuración del sistema**	**3,125 day...**	**17/02/14 8:30**	**21/02/14 13:30**
7		Recepción de la máquina	0,625 days?	17/02/14 8:30	17/02/14 13:30
8		Instalación de Sistemas Operativos	1,875 days?	18/02/14 8:30	20/02/14 13:30
9		Elección de Sistema adecuado	1,25 days?	20/02/14 8:30	21/02/14 13:30
10		**⊟Prototipado con TryStack**	**6,25 days?**	**24/02/14 8:30**	**7/03/14 13:30**
11		Aprender el funcionamiento de OpenStack a través de TryStack	1,25 days?	24/02/14 8:30	25/02/14 13:30
12		Gestión de los módulos y servicios de OpenStack (TryStack) con la ...	1,25 days?	26/02/14 8:30	27/02/14 13:30
13		Aprendizaje de la interfaz Web Horizon	1,25 days?	28/02/14 8:30	3/03/14 13:30
14		Pruebas de acceso a las instancias por SSH y por Web	0,625 days?	4/03/14 8:30	4/03/14 13:30
15		Implementación de máquinas virtuales con LAMP (Linux + Apache...	1,25 days?	4/03/14 8:30	5/03/14 13:30
16		Estudio LEMP (Servidor Nginx)	0,625 days?	5/03/14 8:30	5/03/14 13:30
17		Instalación de Wordpress, Moodle, OwnCloud	1,25 days?	6/03/14 8:30	7/03/14 13:30
18		**⊟Implementación de DevStack**	**6,312 day...**	**10/03/14 8:30**	**31/03/14 13:30**
19		Instalación de VirtualBox para el montaje de virtualización	0,625 days?	10/03/14 8:30	10/03/14 13:30
20		Implementación de DevStack como plataforma de desarrollo Open...	0,625 days?	10/03/14 8:30	10/03/14 13:30
21		Implementación de DevStack (desarrollo) sobre máquina física	0,625 days?	11/03/14 8:30	11/03/14 13:30
22		Gestión de DevStack con Interfaz Web Horizon. Estudio de adapta...	2,5 days?	12/03/14 8:30	17/03/14 13:30
23		Utilización de la API de OpenStack por línea de comandos.	1,938 days?	20/03/14 8:30	31/03/14 13:30
24		**⊟Automatización de Instalación de OpenStack**	**12,5 days?**	**1/04/14 8:30**	**22/05/14 13:30**
25		Automatización de la Instalación de OpenStack con Vagrant y Ansi...	2,5 days?	1/04/14 8:30	4/04/14 13:30
26		Uso de los Repositorios Git	0,625 days?	1/04/14 8:30	1/04/14 13:30
27		Implementación de OpenStack sobre máquinas virtuales	2,5 days?	3/04/14 8:30	8/04/14 13:30
28		Montajes de diferentes nodos: Controller, Compute, Network	4,375 days?	3/04/14 8:30	16/04/14 13:30
29		Gestión de Instancias, Volúmenes, Almacenamiento de Objetos	6,875 days?	29/04/14 8:30	22/05/14 13:30
30		**⊓Implementación de OpenStack módulo a módulo (servicios)**	**6,25 days?**	**5/05/14 8:30**	**23/05/14 13:30**
31		Implementación de OpenStack sobre máquinas virtuales. (Diferent...	2,5 days?	5/05/14 8:30	8/05/14 13:30
32		Detalle de la implementación de los diferentes módulos y funciona...	3,125 days?	8/05/14 8:30	21/05/14 13:30
33		Importancia de Keystone, Cola de mensajes, base de datos mysql.	1,25 days?	22/05/14 8:30	23/05/14 13:30
34		**⊞Pruebas y análisis de las Aplicaciones instaladas y Gestión de l...**	**3,125 day...**	**26/05/14 8:30**	**30/05/14 13:30**
Configuración e Implementación de una Infraestructura Cloud Computing Privada - pagina1					

Tabla 3.- Temporalización de la planificación del proyecto.

3.3. Estimación de recursos del proyecto

3.3.1. Introducción

Como se ha explicado en el apartado *"3.1 Definición de tareas"*, el proyecto se ha planteado como un estudio de la infraestructura OpenStack, no en una implementación real de la infraestructura necesaria para el Instituto.

No obstante para que nos hagamos una idea de los recursos necesarios para un proyecto de esta índole, podemos estimar los recursos que se requerirían si realmente se pusiera en marcha la implementación de OpenStack con las necesidades del Instituto.

3.3.2. Cálculos de memoria, disco, cómputo para el proyecto.

En el apartado *"4.2 Diseño de la Arquitectura OpenStack ",* se explica con detalle las posibles arquitecturas de OpenStack, no obstante, para acometer un proyecto OpenStack se van a necesitar por lo menos tres nodos (equipos donde se ejecuten los servicios OpenStack necesarios).

El nodo **controlador**, será el encargado de la gestión del cloud (autenticación, API, gestión), el nodo de **computación**, donde se ejecutan las instancias (se necesita de mucha memoria RAM y procesadores potentes). Por último necesitamos un nodo de **almacenamiento** donde guardar las imágenes, las instantáneas y volúmenes, además de guardar los vídeos, y escaneados de exámenes.

La plantilla del Instituto "Broch i Llop" consta de 90 profesores. Uno de los requerimientos es que puedan acceder a una máquina Moodle y Owncloud, con unas prestaciones adecuadas, con un tamaño de memoria de 8GB, dos procesadores, y un alojamiento de 30GB por usuario, ya que la capacidad de almacenamiento es importante para el profesorado. Por lo tanto 30GB de almacenamiento multiplicado por 90 profesores hace un total de 2700GB, es decir 2,7 TB.

También se necesita almacenar los vídeos, imágenes y exámenes escaneados, además de toda la infraestructura de volúmenes, instantáneas. Podría requerir otros 3TB de Disco.

Por último deberán ejecutarse unas 30 instancias a la vez, 15 alumnos por 2 instancias y la instancia de Moodle y OwnCloud. En total 31 instancias.

Resumiendo haría falta 30 instancias de 2GB de RAM, 1 de Moodle de 8GB de RAM. Además de 1 procesador por 30 instancias y 2 procesadores para moodle.

En total

RAM (30*2 + 16GB) = 66 GB

Disco 2,7TB +3TB = 5,7 TB

Procesador 30 + 2 =32 CPU

3.3.3. Hardware estimado

Basándonos en el estudio del Proyecto de innovación [8] Infraestructura para el cloud, y en los cálculos que se muestran en el apartado "*4.2.4.2. Escalabilidad de la nube"*, se necesitarán tres nodos que podrían tener las siguientes características:

Nodo *Controlador*: 4GB de RAM , 1 procesador x86_64, 2 interfaz de red de 1Gbps, 30GB de disco duro.
Nodo de *Cómputo*: 64GB de RAM, 2 procesadores , 2 interfaces de red de 1Gbps, 2 discos duros de 350GB.
Nodo de *Almacenamiento*: 16GB de RAM, 2 procesadores, 2 interfaces de red de 1Gbps y 3 discos duros de 2TB.

3.3.4. Adquisición de Equipo para el proyecto

Realmente los cálculos y necesidades de hardware comentados en los dos apartados anteriores no se contemplaron en ningún momento en el proyecto, dada la naturaleza de investigación del mismo.

De esta manera se dispuso de un ordenador con las siguientes características, 8GB de RAM, arquitectura x86_64, dos procesadores, 1TB de disco duro y soporte para virtualización VT-x de Intel.

Con este equipo se ha realizado todo el trabajo de virtualización, realizando la estructura sobre máquinas virtuales, emulando los nodos de control, cómputo, networking y almacenamiento.

Capítulo 4

4. Conceptos, diseño y gestión de OpenStack

4.1. Conceptos básicos sobre OpenStack

4.1.1. Introducción

OpenStack se define como una plataforma de cloud computing implementada a través de software libre para el despliegue de nubes públicas y privadas. Está desarrollada con la idea de ser sencilla de implementar, escalable y con muchas prestaciones. OpenStack proporciona una solución de Infraestructura como servicio (IaaS) mediante un conjunto de servicios interrelacionados. Siendo sus principales servicios el cómputo, los servicios de red y el almacenamiento.

OpenStack está constituido por una fundación, la OpenStack Foundation [9] que establece una serie de principios. Estos principios son los siguientes:

- Establecer una licencia de software libre Apache 2.0.
- Se ha optado por un desarrollo todo abierto, donde se aceptan opiniones y aportaciones de cualquiera. También se presentan y acuerdan muchos aspectos en los "*summits*" que se realizan en diferentes ciudades. La mejor forma de estar informado sobre los cambios y actividades de OpenStack es seguirlos en su cuenta de twitter: @OpenStack.
- Todo el código fuente está disponible en su repositorio de github [10], y utilizan metodologías de desarrollo ágil.
- El código fuente de OpenStack está escrito en lenguaje python, aunque también existen SDK en Java y se están incorporando otros lenguajes para su desarrollo.

4.1.2. Descripción de los servicios de OpenStack

OpenStack no es un solo producto, es un conjunto de servicios que se pueden combinar en función de las características y necesidades de cada implementación. Algunos servicios son fundamentales y otros son complementarios y opcionales.

En la figura 8 se muestran los servicios de OpenStack con el "nombre" asignado por la plataforma.

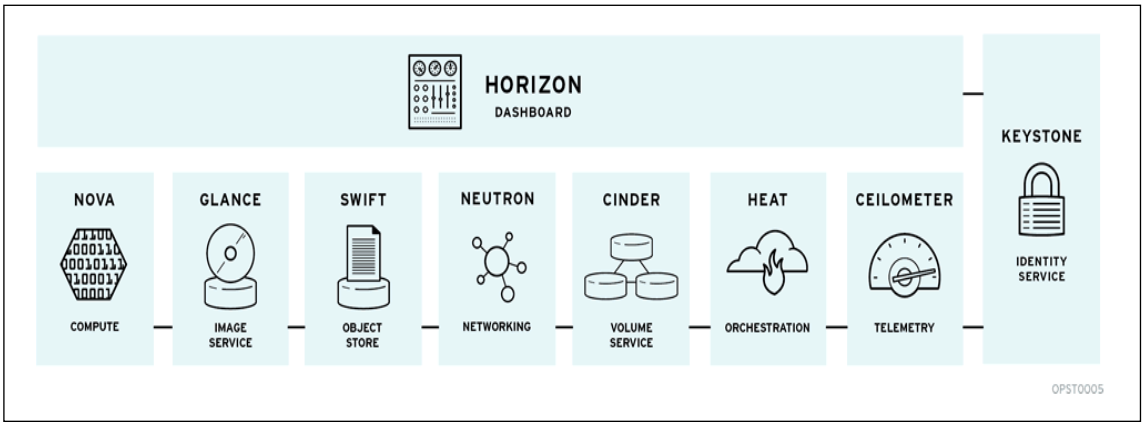


Figura 8. Diagrama de Servicios de la plataforma OpenStack. Fuente: Redhat [11]

A continuación, en la tabla 4 se describen cada uno de los servicios con su funcionalidad en la plataforma OpenStack.

Servicio	Nombre proyecto	Descripción
Dashboard	**Horizon**	Front End basado en web donde se pueden gestionar todos los servicios y plataforma de OpenStack de forma gráfica.
Compute	**Nova**	Almacena y recupera discos virtuales ("Imágenes") y los metadatos asociados en la imagen ("Glance").
Network	**Neutron**	Ofrece la red virtual para Compute .
Almacenamiento		
Block Storage	**Cinder**	Proporciona los volúmenes de almacenamiento para Compute.
Object Store	**Swift**	Es el almacen de objetos. Se puede acceder a través de la API o URL.
Servicios Compartidos		
Identity	**Keystone**	Todos los servicios se autentican contra Keystone.
Image Service	**Glance**	Se pueden almacenar los archivos de disco virtual reales en el almacén de objetos ("Swift")
Telemetry	**Celiometer**	Monitorización de la plataforma
Servicios de Alto nivel		
Orchestration	**Heat**	Crea recurso en la nube con un lenguaje llamado HOT

Database Service	**Trove**	Proviene de una funcionalidad de servicio de base de datos en la nube tanto para bbdd relaciones como no relacionales

Tabla 4.- Descripción Servicios. Fuente: OpenStack-Training Guides[12]

4.1.3. Características de las versiones de OpenStack

OpenStack ha pasado por diferentes versiones hasta llegar a la última vigente en el momento de redactar esta memoria. Actualmente la versión tiene por nombre IceHouse. El proyecto se ha ido realizando entre la versión Havana (todavía está estable) y la versión IceHouse.

La forma de nombrar a las versiones [13] es incrementar una letra en el alfabeto, así se sabrá si es una versión anterior o posterior cuando se consulta la documentación.

La tabla 5 muestra las diferentes versiones que ha habido de OpenStack, en dicha tabla se muestra las incorporaciones y cambios que ha habido en los servicios por cada versión.

Nombre de Versión	Fecha Creación	Componentes incluidos
Austin	21 de Octubre de 2010	Nova, Swift
Bexar	3 de Febrero de 2011	Nova, Glance, Swift
Cactus	15 de Abril de 2011	Nova, Glance, Swift
Diablo	22 de Septiembre de 2011	Nova, Glance, Swift
Essex	5 de Abril de 2012	Nova, Glance, Swift, Horizon, Keystone
Folsom	27 de Septiembre de 2012	Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder
Grizzly	4 de Abril de 2013	Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder
Havana	17 de Octubre de 2013	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder
IceHouse	Abril de 2014	Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder
Juno	--	--

Tabla 5.- Versiones de OpenStack. Fuente: OpenStack - Training guides [12]

4.1.4. Terminología de los elementos Cloud Computing en OpenStack

Una vez visto los componentes o servicios de los que consta OpenStack, también hay que conocer la terminología de los elementos de cloud computing para poder gestionar la plataforma correctamente, ya sea mediante el cuadro de mandos Horizon, mediante la línea de comandos o mediante python haciendo uso de la API como veremos posteriormente.

4.1.4.1. Gestionado por el servicio Compute (Nova)

Imagen: Sistema Operativo previamente configurado que se almacena en Glance. Esta imagen es la que se clona, se instancia. Las imágenes soportan diferentes formatos. Dichos formatos deben ser adecuados para el manejo del hipervisor.

Los diferentes formatos soportados por las imágenes son los siguientes, mostrados en la tabla 6:

Formatos	
AMI(el formato de AWS)	QCOW2(Qemu/KVM)
VHD (Hyper-V)	VMDK(VMware)
VDI (VirtualBox)	OVF(VMWare,others)
Raw	

Tabla 6.- Formatos de imágenes soportados por OpenStack.

Instancia: Es una máquina virtual que el usuario va a usar. Va a tener IP, nombre, claves SSH propias para el usuario. La instancia puede estar ejecutándose, en pausa o parada. También se puede borrar, perdiendo su contenido.

Sabor (flavor): Posee las características de la máquina virtual de la instancia. Define el número de CPUs virtuales, los cores, la RAM, la capacidad de disco duro, si es disco raíz o disco externo. Los sabores están preconfigurados, pero el administrador del sistema puede crear nuevos sabores.

Para la administración del sistema se pueden crear diferentes perfiles para el acceso a la plataforma. Existe un usuario administrador con permisos totales y luego se pueden crear usuarios específicos por proyectos (**Tenants**).

Instantánea: Es un snap shot de una imagen o de un volumen. Es como una copia de seguridad. De esta manera podemos duplicar las imágenes y los volúmenes y utilizarlos en instancias. Este procedimiento no sustituye al

mecanismo de copias de seguridad que deben seguir realizándose en el sistema. Lo tenemos que ver como un mecanismo de réplica de imagenes o volúmenes.

IP Fija: Es la dirección IP de la instancia. Se asigna automáticamente por DHCP en la creación de la instancia y tiene la misma duración que la vida de la instancia.

IP Flotante: Es una IP que se asocia a una instancia. Se realiza esta asociación para acceder desde fuera de la instancia, se realiza un NAT. Pueden existir instancias que no tengan una IP flotante, porque no interese que se acceda desde fuera, ya que se puede querer que forme parte de la red interna solamente.

Grupo de Seguridad: Es un conjunto de reglas de cortafuegos que controlan el acceso a las instancias mediante las IP Flotantes. Puede configurarse varios grupos de seguridad. Realmente por debajo está realizando iptables.

Par de claves SSH: Las imágenes que descargamos desde el repositorio de OpenStack o desde el repositorio de los fabricantes no llevan contraseña por defecto o no es conocida. Al instanciar estas imágenes necesitamos un mecanismo de acceso seguro, y éste se realiza desde el par de claves. Si no poseemos nosotros una clave privada, existe un mecanismo en OpenStack para la generación de la clave pública y la privada. La pública se queda en el sistema y la privada la descargamos a nuestra máquina para el acceso. Se comentará más adelante un ejemplo de acceso por SSH.

4.1.4.2. Gestionado por el servicio Block Storage (Cinder)

Volumen: Es un dispositivo de bloque que se puede asociar o desasociar a una instancia cuando se desee. El servicio que maneja el volumen en OpenStack se llama Cinder. No se puede asociar un volumen a más de una instancia, y tampoco se puede modificar el volumen en caliente. Pero podemos crear instancias a partir de volúmenes y que estos volúmenes dispongan de los aplicaciones que queramos.

4.1.4.3. Gestionado por el servicio Object Storage (Swift)

Contenedor: Sirve para organizar objetos. Los contenedores no se pueden anidar.

Objeto: Son los elementos que se almacenan, también se almacenan los metadatos. Se manejan grandes volúmenes de datos a través del servicio swift de OpenStack. Es el equivalente a S3 en AWS.

4.1.4.4. Gestionado por el servicio Networking (Neutron)

Red: Dominio aislado de capa2. Equivalente a VLAN.

Subred: Bloques de direcciones de IPs que se asignan a una máquina virtual (Instancias).

Router: Sirve para conectar redes.

Puerto: Puerto Virtual del switch o del router.

4.1.5. Consideraciones sobre las nociones básicas de OpenStack

Una vez conocida toda la terminología anterior se va a poder entender de forma adecuada los siguientes apartados de la memoria, en concreto los que tratan de la creación de un proyecto (*tenant*) usando máquinas virtuales (*instancias*) que se basan en imágenes,volúmenes o instantáneas y que poseen unas determinadas características (número de CPU, memoria, disco,etc...) que lo marca el *Sabor*.

Todas estas máquinas virtuales serán las que pueda acceder el usuario, ya sea a su red interna mediante las *IP fijas* o a la red externa mediante la *IP flotante*. Las instancias (máquinas virtuales) están integradas en una red y su acceso se realiza gracias al mecanismo de *Grupos de Seguridad* y *Par de claves*.

Existe también la posibilidad de manejar grandes volúmenes de datos a través del *almacenamiento de objetos* o guardar datos en bloque a través del dispositivo de *Volúmenes*.

4.2. Diseño de la Arquitectura OpenStack

4.2.1. Introducción

La puesta en marcha de una máquina virtual (una instancia) implica muchas interacciones entre varios servicios. La figura 9 ofrece la arquitectura conceptual de un entorno típico de OpenStack.

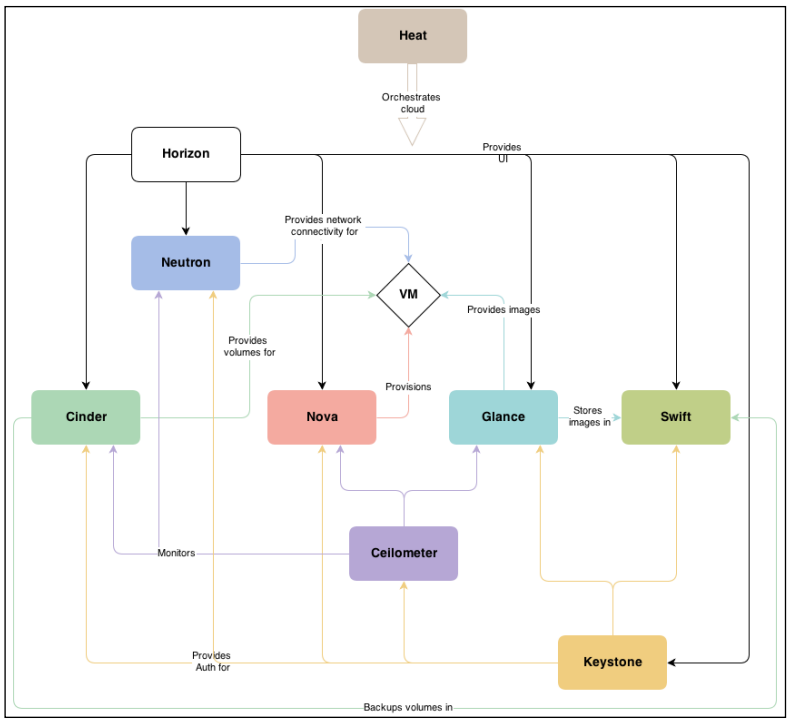


Figura 9.- Arquitectura Conceptual de OpenStack. Fuente:Openstack [5]

OpenStack es altamente configurable para satisfacer las diferentes necesidades en función de los requisitos de cómputo, redes y almacenamiento.

Los servicios vistos anteriormente deben estar disponibles en el sistema. Existen diferentes formas de agrupar estos servicios para dar consistencia a la infraestructura OpenStack. Estos servicios se configuran o instalan en nodos (las máquinas reales o virtuales) donde van a ejecutar los servicios.

Existen varias agrupaciones de estos servicios para hacer más escalable y distribuida la infraestructura. Para explicar alguna de las agrupaciones nos hemos apoyado en la guía de Instalación de OpenStack sobre Ubuntu [7]. En los siguientes apartados se explican algunas de sus arquitecturas.

4.2.2. Descripción de la arquitectura Three-node.

La arquitectura *"Three-node architecture with OpenStack Networking (neutron)"* se muestra en la figura 10.

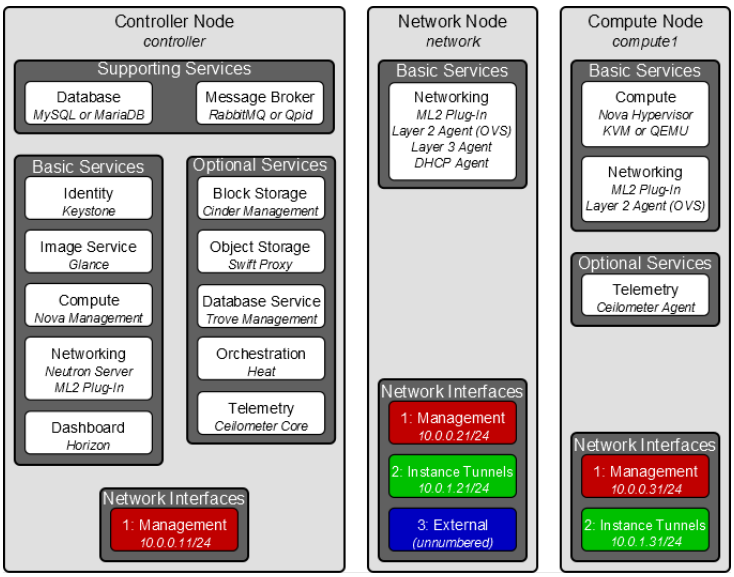


Figura 10.- Three-node architecture with OpenStack Neutron. Fuente:OpenStack [5]

Podemos considerar los nodos como las máquinas físicas donde se ejecutan los servicios de OpenStack, en concreto con esta arquitectura de 3 nodos, los servicios se reparten de la siguiente manera.

El nodo *controller* ejecuta el servicio Identity Service, gestión del Compute y del Networking, y el servicio Horizon. Soporta servicios para database, message broker y servicio NTP.

Opcionalmente el nodo controlador puede ejecutar servicios de Block Storage, Object Storage, Database Service, Orchestatrion y Telemetry.

El nodo *network* ejecuta servicios como el Networking plug-in, para proveer y operar con los tenant networks. También para dar soporte a los redes virtuales y a los túneles. Incluye también routing, NAT y DHCP. También incluye conectividad para proporcionar internet a las máquinas virtuales o instancias.

El nodo *compute* ejecuta el componente hipervisor de Compute que opera con los tenants de las máquinas virtuales o instancias. Por defecto Compute usa KVM como hipervisor. Además el nodo *compute* ejecuta el plug-in de Networking para operar con la red de los tenants e implementar la seguridad en la configuración del los Grupos de Seguridad.

Opcionalmente el nodo de *compute* también puede ejecutar el agente Telemetry. Siempre se pueden añadir más nodos a la arquitectura para hacerla escalable.

4.2.3. Descripción de la arquitectura Two-node.

La arquitectura *"Two-node architecture with legacy networking (nova-network)"* se muestra en la figura 11.

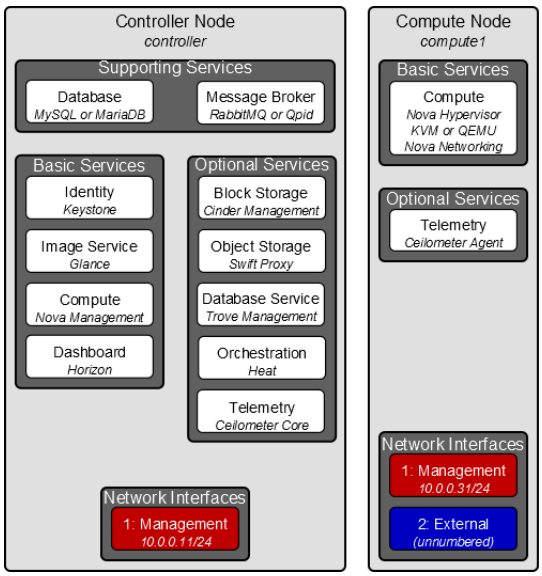


Figura 11. Two-node architecture with legacy networking(nova-network). Fuente:OpenStack[5]

El nodo controlador ejecuta el servicio Identity Service, la gestión del compute, Horizon, Image Service. Da soporte a database, message broker y NTP. Opcionalmente ejecuta Block Storage, Object Storage, Database Service, Orchestation y Telemetry.

El nodo compute ejecuta el hipervisor de Compute que opera con los tenants de las máquinas virtuales o instancias. Por defecto Compute usa KVM como hipervisor.

4.2.4. Consideraciones sobre los ejemplos de arquitecturas y escalabilidad de la nube

4.2.4.1. Consideraciones sobre los ejemplos de arquitecturas

En los apartados anteriores se han mostrado dos ejemplos de arquitecturas, estos ejemplos están extraídos de la guía de Instalación de OpenStack, y se muestran en dicha guía para el aprendizaje básico de los servicios de OpenStack.

OpenStack es una infraestructura muy amplia, y estas dos arquitecturas se muestran en la guía a modo de aprendizaje (son las dos funcionales), pero de una forma más amena y sencilla podemos ir aprendiendo el funcionamiento, instalación y cómo funciona cada uno de los servicios de OpenStack.

En el "*apartado 5.3 implementación de OpenStack paso a paso*", se hace uso de la arquitectura *"Two-node"* para comprender la estructura interna que maneja OpenStack.

4.2.4.2. Escalabilidad de la nube

En el capítulo 3 de la guía *OpenStack Operations Guide* [14], tratan el tema de la escalabilidad del sistema.

"La determinación de la escalabilidad de la nube y cómo mejorarla es un ejercicio con muchas variables para equilibrar. Ninguna solución cumple con los objetivos de escalabilidad de todos los sistemas. Sin embargo, es útil para el seguimiento de un número de métricas. "

El punto de partida para la mayoría de los sistemas es el número de núcleos de la nube. Mediante la aplicación de algunas escalas, se puede recopilar información sobre el número de máquinas virtuales que se pueden esperar en ejecución en el sistema y la cantidad de almacenamiento que se requiere.

Por ejemplo teniendo 10 núcleos físicos y un nodo físico hasta 48 GB de RAM; si las instancias se ejecutan con el sabor medio, es decir con 2 núcleos virtuales, 4GB de RAM y 50 GB de almacenamiento. Se pueden tener ejecutando 18 máquinas virtuales, y se requiere un almacenamiento de 80TB.

El cálculo para hallar las instancias o máquinas virtuales se realiza de la siguiente manera. Como el ratio de RAM esta en 1.5, el calculo es el siguiente:

Cálculo: 48GB *1.5 =72GB

y

72GB/4GB de RAM = 18 instancias

El cálculo del tamaño de almacenamiento es el producto del número de máquinas virtuales por tamaño. En el ejemplo 18*50GB = 900 GB.

Además, como el ratio de los núcleos está en 16:1, es decir una CPU física equivale a 16 virtuales, podríamos tener instancias hasta de 8 núcleos, ya que siguiendo el cálculo.

Cálculo: (10 núcleos * 16) / 18 = 8 núcleos posibles

4.3. Gestión de OpenStack

4.3.1.- Introducción

La gestión de OpenStack se puede realizar mediante la línea de comandos o con el lenguaje de programación python a través de la API de la plataforma; o interactuándo con una Interfaz Web desarrollada con Django, llamada Horizon.

La plataforma TryStack resulta la más adecuada para aprender a manejar y gestionar OpenStack a través del Dashboard Horizon, de una manera simple, rápida y sencilla. Haciendo que la curva de aprendizaje de OpenStack disminuya considerablemente.

Como ya se ha comentado en el "*apartado 2 Descripción del proyecto*", TryStack es una plataforma muy útil para el prototipado del proyecto. Nos permite crear un proyecto sin preocuparnos de adquirir los recursos del sistema, ya que este entorno nos cede los recursos para el testeo. La cesión tiene una duración de 24 horas y limitado a tres instancias, pero es un tiempo más que suficiente para realizar las tareas de test.

A continuación se detallan las tareas realizadas en la plataforma TryStack, para facilitar su aprendizaje y desarrollar los objetivos solicitados en el proyecto.

4.4. Prototipado con TryStack

4.4.1. Registro en la plataforma TryStack

Lo primero que se debe realizar para acceder a TryStack [15], en concreto a la plataforma x86, es que se tiene que loguear a través de una cuenta de Facebook. Los administradores de TryStack comentan en su página web que están estudiando otras alternativas, pero de momento es la única forma de acceso.

El primer paso para conseguir acceso, es utilizar una cuenta de Facebook (en nuestro caso esta cuenta no tenía ningún "amigo" asociado), y por ello la inclusión al grupo de TryStack y concesión de acceso no fue aprobada desde la administración de TryStack.

Al no conceder el acceso desde esta cuenta, se volvió a reclamar el acceso al grupo de TryStack desde una cuenta de Facebook con un volumen más elevado de grupo de amigos, y entonces sí que se recibió la notificación de inclusión en el grupo y la concesión de acceso a TryStack. La figura 12 muestra la pantalla que explica el funcionamiento de la plataforma TryStack.

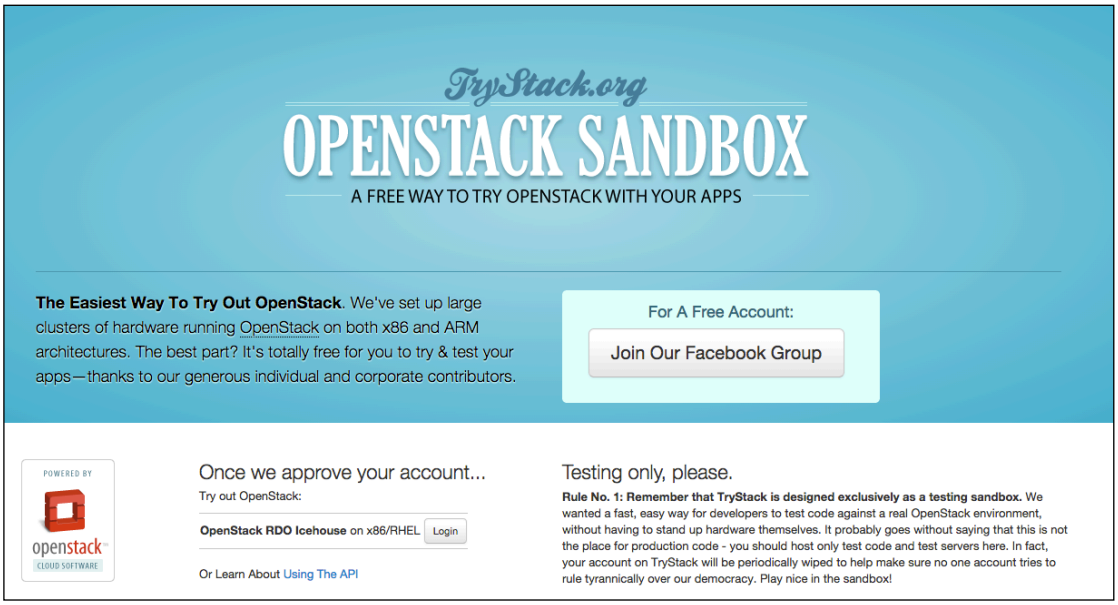


Figura 12.- Explicación de la plataforma TryStack de testeo. Fuente: TryStack [15]

4.4.2. Acceso a la plataforma TryStack

Una vez que se pulsa en el Login nos lleva a la página de acceso a través de Facebook. En la figura 13 se muestra el modo de acceso a TryStack además de un link sobre las preguntas habituales sobre la plataforma y un vídeo explicativo de la misma.

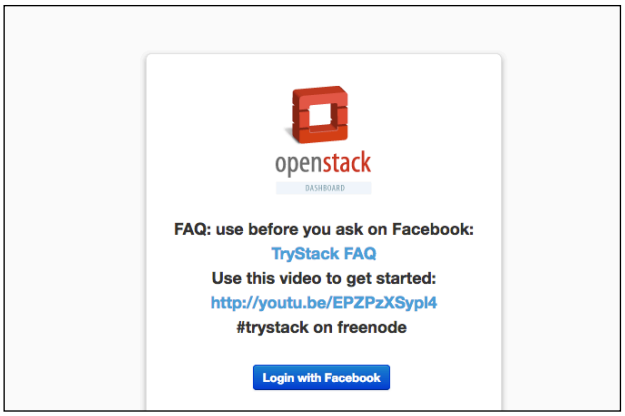


Figura 13.- Acceso a TryStack a través de la cuenta de Facebook. Fuente:TryStack [14]

Como dato curioso, el acceso a la plataforma desde el Instituto "Broch i Llop" , no se podía realizar directamente ya que la conexión a Internet a través del instituto está prohibida para la aplicación Facebook, ya que la conexión pasa a través de un firewall de Consellería de la Generalitat Valenciana, que impide el

acceso a Facebook, a Twitter y a otras páginas que consideran inadecuadas para los centros de educación.

Para poder utilizar TryStack se necesitó acceder a través del navegador Tor Browser [16], que permite el acceso anónimo por Internet, de esta manera se consigue saltar la restricción del firewall de Consellería. Por contra, al estar cifrado el acceso hace que la conexión sea más lenta, aunque de esta manera se consigue realizar el testeo de la plataforma.

En la figura 14 se muestra la pantalla inicial de la plataforma TryStack cuando se accede por primera vez. Muestra el componente Horizon de OpenStack. En la parte izquierda de la imagen se ven las posibilidades que nos brinda TryStack. En esta plataforma sólo nos deja manejar un proyecto (tenant), que es el que está configurado por defecto. Más adelante, se verá que con las plataformas DevStack y OpenStack si que se pueden manejar multiples proyectos y asignar diferentes usuarios a los mismos.

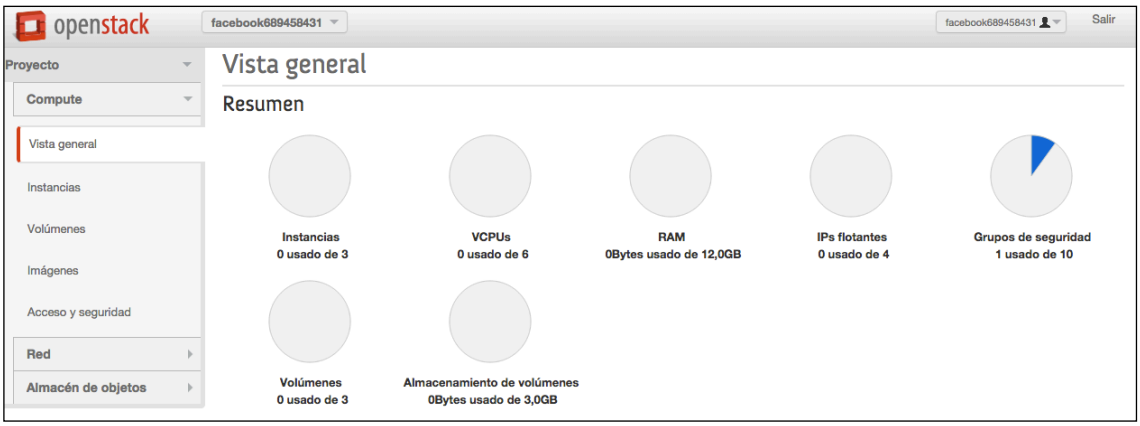


Figura 14.- Vista de Horizon (Dashboard).

En el lado izquierdo de la imagen se ven los diferentes servicios que se pueden usar. Compute (Instancias, Volúmenes, Imágenes, Acceso y Seguridad) , Red y Almacén de Objetos. Todos estos componentes y su uso se van a explicar a continuación.

4.4.3. Gestión de OpenStack a través de Horizon

4.4.3.1. Gestión de la Red

Lo primero que se debe crear para realizar una implementación de un proyecto, es la *Red* que se va a gestionar en nuestro proyecto. En la figura 15 se muestra una estructura de red, donde ya se ha incorporado un router.

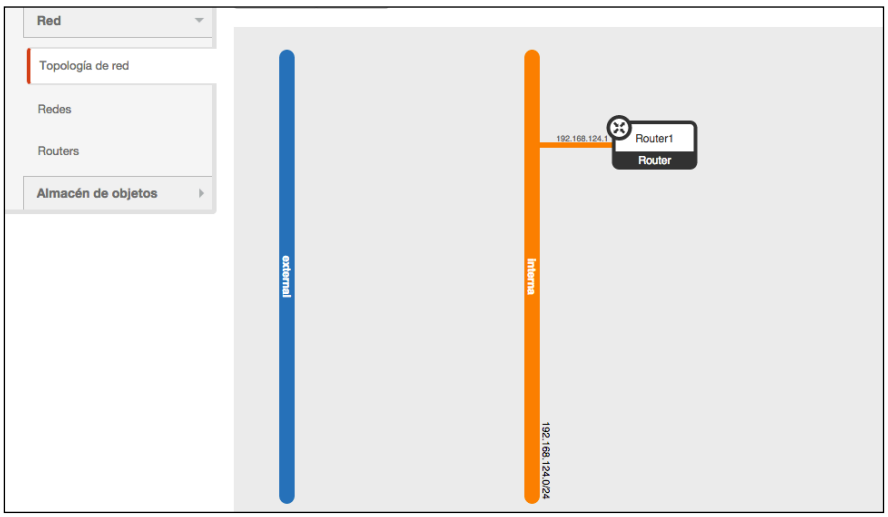


Figura 15.- Topología de red plataforma TryStack.

Dentro del servicio de **Red** se pueden ver tres apartados *Topología de red* donde se muestra las redes que poseemos. En la figura 15 se muestran dos redes, una red externa en azul, es la red a la que se puede acceder de forma externa mediante NAT - la que asociaremos la IP_flotante - y la red interna de la infraestructura.

En el apartado Redes, se pueden crear, borrar, editar la red o redes que se desee, como muestra la figura 16.

Figura 16.- Características de la red y crear nuevas redes.

Por ejemplo se puede crear una segunda red interna y darle un rango de Ips. La figura 17 muestra como se ha ido configurando la red.

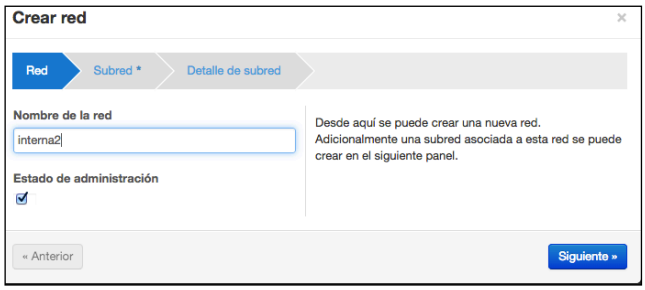


Figura 17.- Nombre de la segunda red interna.

En la figura 18, se elige un rango de Ip's para la segunda red interna. El rango elegido es 192.168.125.0 /24

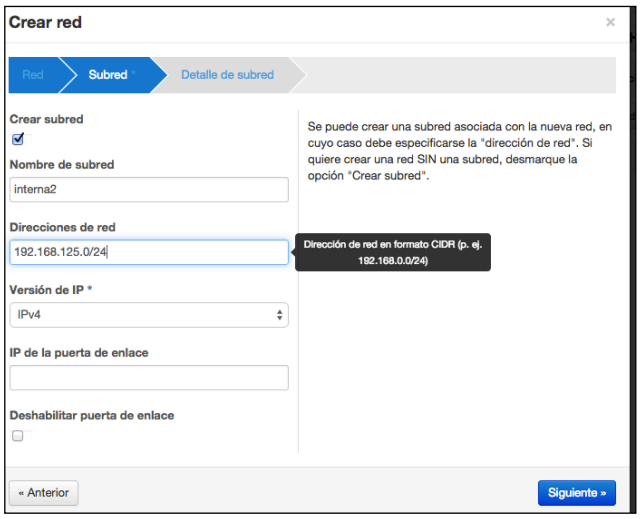


Figura 18.- Asignación de Rango de Ip's.

La figura 19 muestra el resumen de las redes, donde se observa la creación de la segunda red llamada interna2.

Figura 19.- Resumen de las redes con sus Ips asociadas.

En la figura 20 se observa como ha quedado la nueva topología de red una vez incorporada la segunda red a la infraestructura.

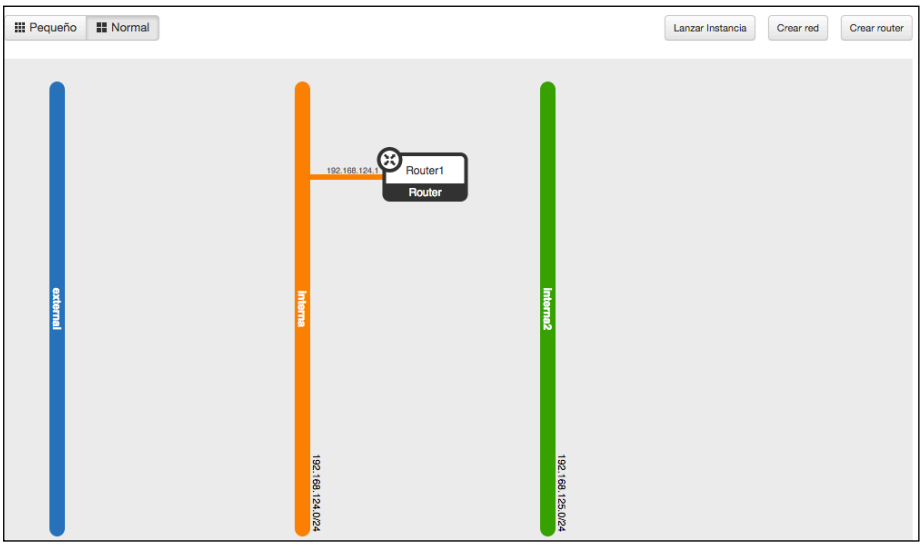


Figura 20.- Topología de red.

Para realizar el conexionado de la red externa con las **dos redes internas**, se debe configurar en las redes internas la conexión con el router y éste hay que conectarlo con la red externa. A continuación se explica como realizar esta configuración. Se observa también que desde la topología de red se pueden crear routers, lanzar instancias, crear redes.

Como primer paso para unir la red interna2 con la red externa se debe crear un router, tan sencillo como pulsar en el botón crear router como muestra la figura 21.

Figura 21. Creación del router (Router2).

Nota .- Anteriormente se había creado una red interna y un Router1 para la primera red, aunque se ha omitido la captura de pantallas de este montaje, ya que son los mismos pasos que se explican para la red interna2.

A continuación se le añade una interfaz al Router2 creado mediante el botón añadir interfaz. Aquí se selecciona que red se desea añadir, se puede elegir entre la red interna o interna2. Se une a la interna como queda reflejado en la figura 22.

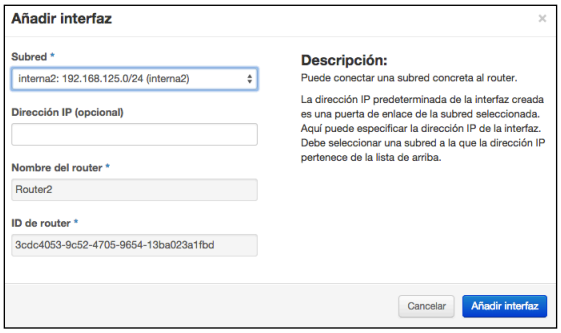


Figura 22.- Asociación del router a la red interna2.

De esta manera si se vuelva a observar la topología de red, nos aparecen las dos redes internas (interna e interna2) conectadas cada una a su respectivo router (figura 23).

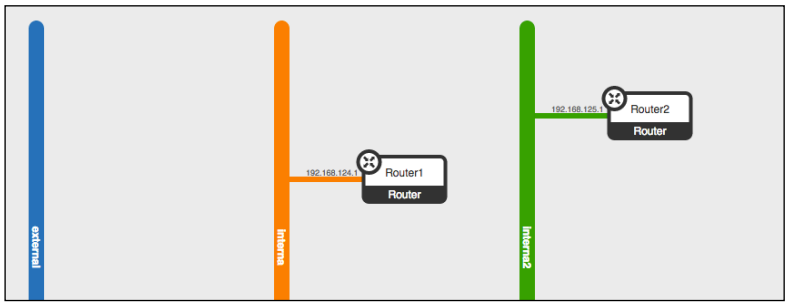


Figura 23.- Topología de red con las tres redes.

Ahora solo falta asociar la puerta de enlace a cada router para conectar con la red externa. La figura 24 muestra como asociar una puerta de enlace al router. Se asocia a la red externa. Este proceso se realizará tanto para el router1 como para el router2.

Figura 24. Estableciendo la puerta de enlace al Router2.

Quedando la topología de red como muestra la figura 25. Se observa que los dos Routers ya están unidos a la interfaz externa y que internamente estan unidos a las dos redes internas. El Router1 une con la ip 192.168.124.1 la primera red interna y con la puerta de enlace 192.168.125.1 se une el Router2 a la red interna2.

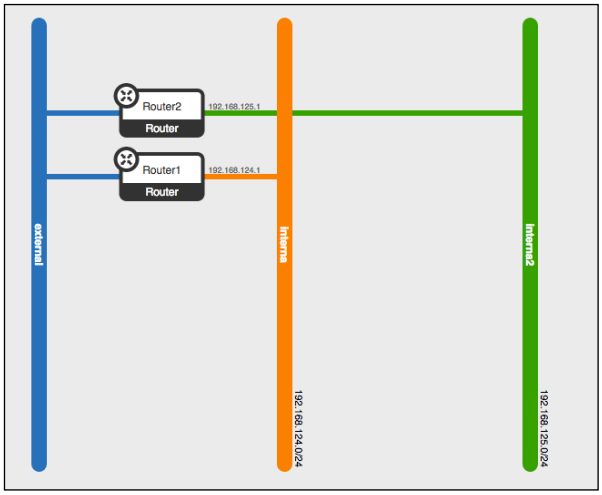


Figura 25.- Topología definitiva para el ejemplo de infraestructura.

Una vez creada la red, se deben crear las máquinas virtuales (las instancias) que actuarán en el entorno IaaS de OpenSTack.

4.4.3.2. Gestión de Instancias

La creación de instancias se puede crear desde el apartado Topología de red, pulsando el botón Lanzar Instancia o desde el apartado Compute, Instancias. Da igual desde donde se realice, ya que va a pedir los mismos pasos.

Lo primero que se debe realizar para crear una instancia es elegir la imagen a instanciar. Como se ha explicado anteriormente en el apartado "*4.1. Conceptos básicos de OpenStack*" en la parte de terminología, una imagen es un sistema operativo previamente configurado. Se pueden subir a la plataforma los diferentes sistemas operativos que se deseen. En el siguiente enlace de la guía de imágenes [17] de OpenStack se pueden descargar las imágenes adecuadas para la plataforma.

La figura 26 muestra las imágenes preconfiguradas que nos proporciona la plataforma OpenStack, no obstante se tiene la posibilidad de subir otras imágenes a la misma.

Figura 26.- Listado de imágenes presentes en la plataforma TryStack.

La creación de la instancia consiste en estos sencillos 6 pasos:

1. Elegir la imagen que se va a instanciar. Se va a elegir a modo de ejemplo la imagen Ubuntu 13.10, son imágenes que los fabricantes ya han dejado en un formato válido para trabajar en la nube, en este caso QCOW2.

2. Elegir el sabor que va a tener la instancia. En la plataforma TryStack no está permitido realizar una configuración a medida de los sabores, y debemos elegir entre las tres categorías disponibles (small, medium y large). En OpenStack si que permite definir más sabores. En las figura 27 se puede ver las características de cada sabor.

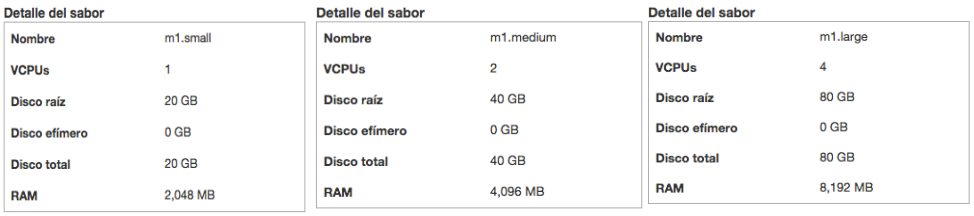


Figura 27.- Diferentes características de los sabores (small,medium,large).

3. Asignar el grupo de Seguridad con el que va a trabajar la instancia. En el apartado de Acceso y Seguridad se pueden crear diferentes grupos de seguridad, cada uno de ellos con unas reglas *ip tables* específicas. En la figura 28 se muestran las diferentes grupos creados.

Figura 28.- Grupos de Seguridad.

Mediante el botón de Administrar reglas se puede configurar de una forma sencilla el comportamiento de "ip tables" del sistema OpenStack. En la figura 29 se muestran algunas reglas configuradas en el Grupo de seguridad "Brochillop_Profesores".

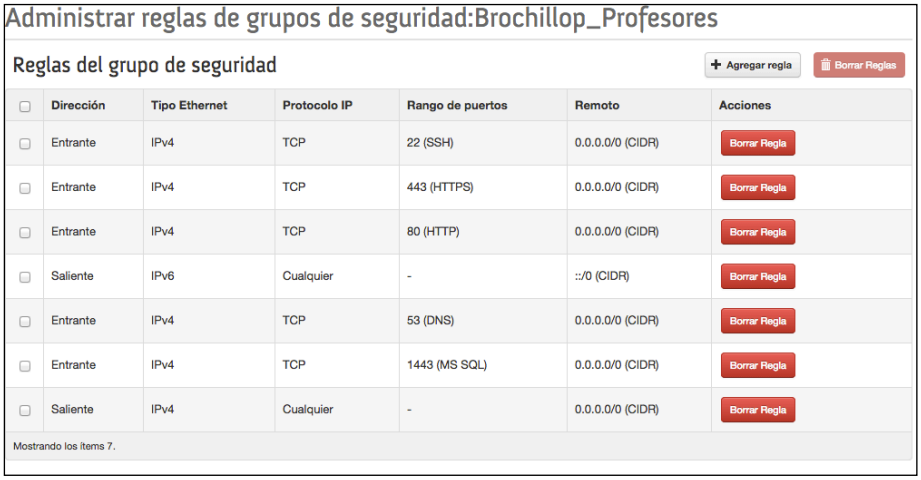


Figura 29.- Reglas de grupo Seguridad "Broch i Llop para profesorado"

En este grupo de reglas, se permite el acceso por SSH, HTTP, HTTPS, DNS y el acceso a una instancia con la aplicación Microsoft SQL Server que responde en el puerto 1443.

4. Asignar la clave RSA de acceso seguro a la instancia: Para asignar la clave de acceso seguro, primero hay que crearla. OpenStack dispone de un mecanismo sencillo de creación de claves públicas y privadas. La clave pública la guarda el sistema y la privada se la descarga el usuario. En la figura 30 se muestra la clave de seguridad creada y el botón de *crear par de claves*.

Figura 30.- Claves de seguridad del proyecto.

La figura 31 muestra como crear un par de claves nuevas.

Figura 31.- Creación de claves para acceso remoto por SSH.

5. Elegir la red interna en la que va a estar conectada la instancia.

6. Asignar una IP Flotante.

Estos son todos los pasos que hay que hacer para generar una instancia. En la figura 32 se muestra la pantalla completa del proceso de creación de instancias.

La zona que maneja las instancias es nova, la parte de cómputo de la infraestructura. También se le proporciona un nombre a la instancia, se elige un sabor, y se selecciona el origen de la instancia. En este caso se ha escogido la instancia a través de una imagen, pero también existe la posibilidad de crear una instancia a través de un volumen, o de una instantánea.

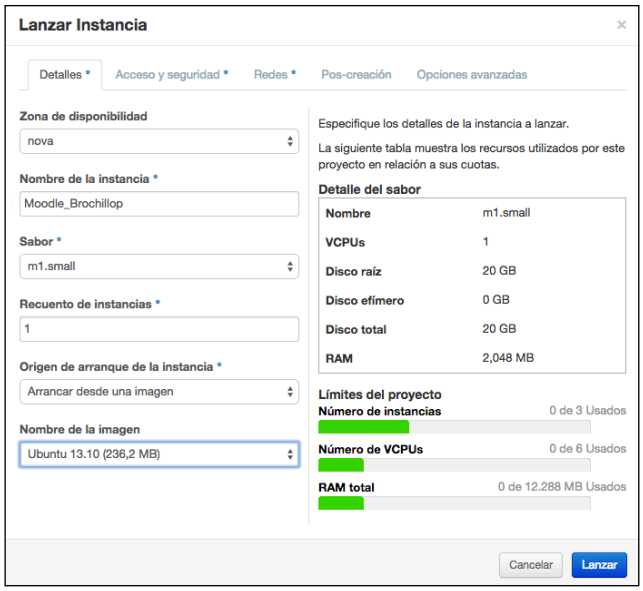


Figura 32.- Proceso de creación de Instancias con Horizon de OpenStack.

La figura 33 muestra la asociación de la clave de acceso segura mediante la pestaña Acceso y Seguridad. Y el grupo de seguridad elegido.

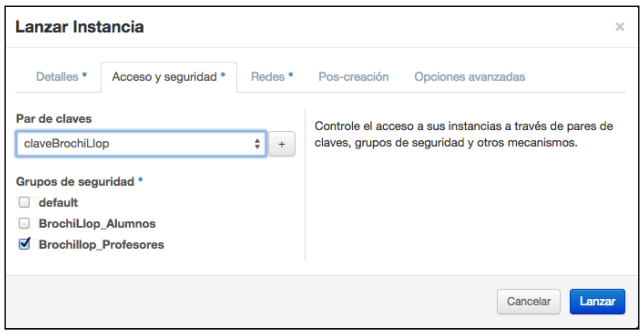


Figura 33.- Selección del par de claves y asignación del grupo de Seguridad.

A continuación se puede decidir unir la instancia a la red interna o a la red interna2. En la figura 34, se escoge unir la instancia a la red interna.

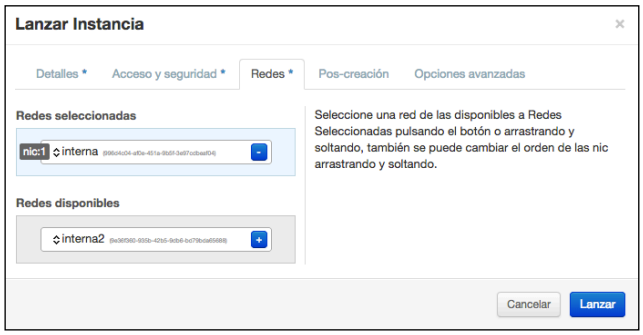


Figura 34.- Configurar la red para la instancia.

Una vez realizados todos los pasos, se pulsa el botón Lanzar instancia y se produce el proceso de creación de la misma como muestra la figura 35.

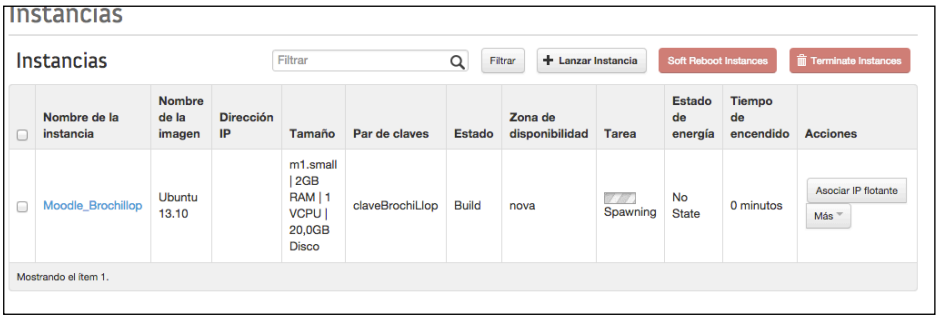


Figura 35.- Proceso de creación de la instancia.

Si se quiere que la instancia sea accedida desde el exterior, se debe asignar una IP_Flotante. En la figura 36 se ve que se puede pulsar el botón de Asociar IP Flotante para crearla, además de realizar más acciones con la instancia.

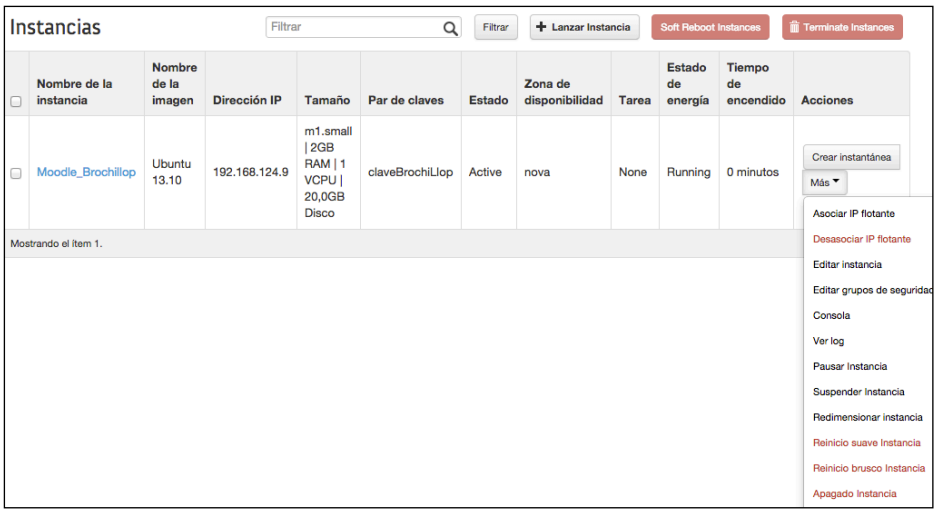


Figura 36.- Acciones posibles que se pueden realizar sobre la instancia.

Pulsando el botón Asociar IP flotante, se le asigna una IP a la instancia, en este caso como refleja la figura 37, se le ha asignado la IP Flotante 8.21.28.49 a la instancia para realizar el acceso remoto.

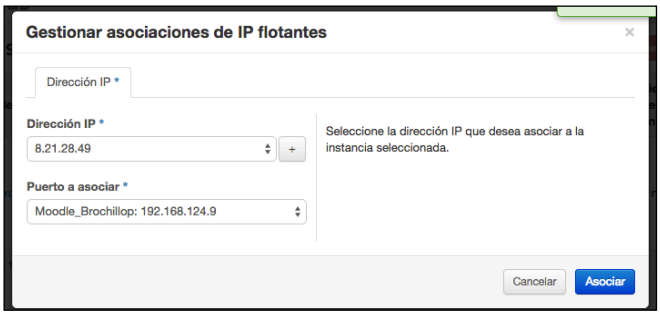


Figura 37.- Asociación de IP_flotante a la instancia.

4.4.3.3. Acceso a la instancia por SSH

Ahora ya se puede realizar el acceso a la instancia desde SSH y configurar las aplicaciones requeridas. En el caso del proyecto se va a mostrar como se accede remotamente y se instala la aplicación Moodle, Wordpress y OwnCloud para uso por parte del profesorado.

Para realizar el acceso, desde una consola del equipo local, se busca la clave privada que se ha descargado desde el sistema OpenStack de forma automática en

un directorio local al generar las claves, en mi caso se llama clavebrochillop.pem. Si intentamos el acceso como refleja la figura 38.

```
MacBook-Pro-de-admin-2:~ admin$ cd Downloads
MacBook-Pro-de-admin-2:Downloads admin$ ls *.pem
clave1.pem               clavebrochillop.pem     claveubuntutry.pem
MacBook-Pro-de-admin-2:Downloads admin$ ls -l clavebrochillop.pem
-rw-r-----@ 1 admin  staff  1671 31 ago 17:55 clavebrochillop.pem
MacBook-Pro-de-admin-2:Downloads admin$ ssh -i clavebrochillop.pem ubuntu@8.21.28.49
The authenticity of host '8.21.28.49 (8.21.28.49)' can't be established.
RSA key fingerprint is 8c:97:25:80:1e:b4:7b:6f:12:bc:3e:8b:dc:21:bf:d1.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '8.21.28.49' (RSA) to the list of known hosts.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0640 for 'clavebrochillop.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
bad permissions: ignore key: clavebrochillop.pem
Permission denied (publickey).
MacBook-Pro-de-admin-2:Downloads admin$
```

Figura 38.- Problemas de acceso por no tener los permisos adecuados.

Se aprecia que no se puede acceder al sistema , ya que la clave privada no tiene los permisos adecuados. Se soluciona este inconveniente quitando permisos para el usuario y el grupo y se accede a la máquina nuevamente de una forma correcta como se muestra en la figura 39.

```
MacBook-Pro-de-admin-2:Downloads admin$ chmod 600 clavebrochillop.pem
MacBook-Pro-de-admin-2:Downloads admin$ ssh -i clavebrochillop.pem ubuntu@8.21.28.49

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

ubuntu@moodle-brochillop:~$
```

Figura 39.- Acceso a la máquina Ubuntu por acceso remoto.

4.4.3.4. Instalación de aplicaciones específicas

Para poder instalar Wordpress, Moodle, se necesita tener en el sistema un entorno LAMP (Linux+Apache+Mysql+PHP). Se puede seguir la guía de instalación de LAMP [18] sobre Ubuntu.

Una vez instalado el componente Apache, se puede ver que realmente está resolviendo la IP y mostrando el servidor web. Si se modifica el fichero */var/www/index.html*, por ejemplo escribiendo en la página por defecto de Apache "Broch i Llop Works", en la figura 40 se puede observar que realmente se accede al servidor Apache por la IP 8.21.28.49

Figura 40.- Apache corriendo sobre IP_flotante asociada a la instancia.

También se puede comprobar que PHP está bien instalado como muestra la figura 41.

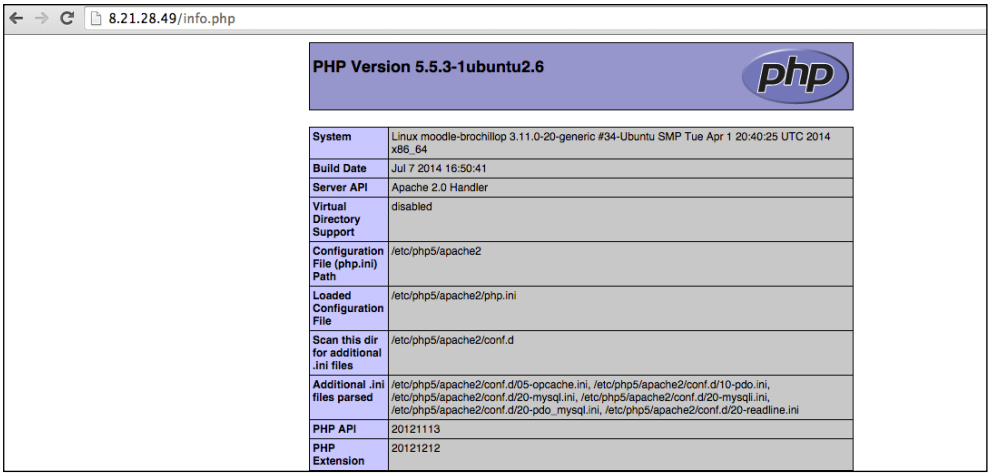


Figura 41.- PHP funcionando sobre IP_flotante.

Una vez se tiene LAMP funcionando, siguiendo la guía de instalación de Moodle [19], se baja la versión de moodle que se desee instalar mediante el comando *wget*, se descomprime y se crea el directorio moodledata. También se debe crear una base de datos para moodle. Se puede acceder a la consola de mysql con el comando "*mysql -u root -p*", y crear la base de datos moodle con el comando "*create database moodle;*" como muestra la figura 42.

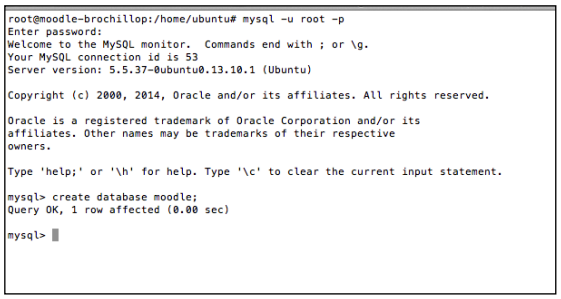


Figura 42.- Creación de la base de datos moodle.

Sólo hace falta seguir el asistente de configuración de moodle e ir configurando en base a la guía la instalación como refleja la figura 43.

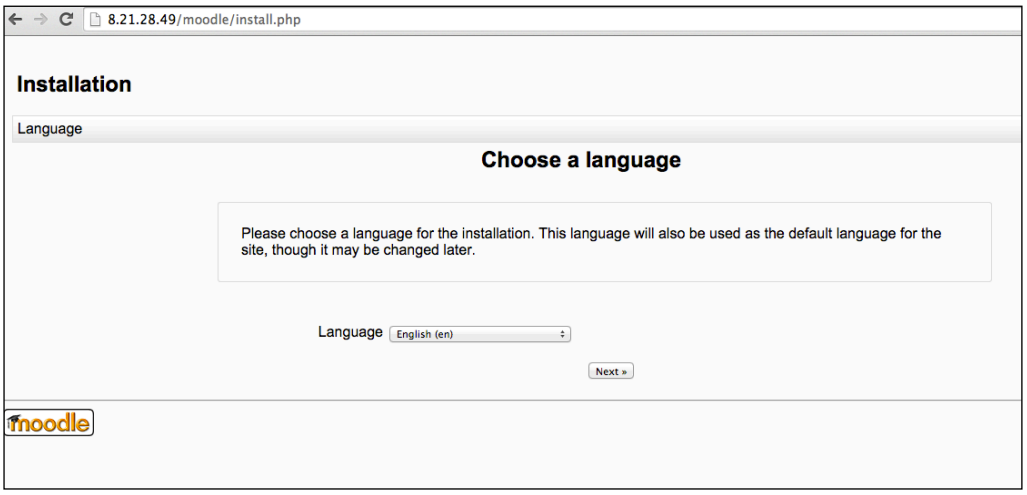


Figura 43.- Instalación de moodle.

En el proceso de instalación hay un momento que pide que se copie un script de configuración de *config.php*, simplemente se debe acceder a *"/var/www/moodle"* , crear un fichero *config.php* con el comando *"nano",* y copiar el contenido de la web.

También puede producirse un error de instalación, ya que igual no se poseen tolos los requerimientos *php-json*, simplemente instalando este módulo, la instalación continuará. Se puede instalar haciendo "*apt-get install php5-json*" y reiniciar el servicio de apache con el comando *" sudo /etc/init.d/apache2 restart".*

Una vez realizada la instalación ya se puede entrar en la aplicación Moodle y crear cursos, cargar como usuarios a alumnos del instituto, a profesores. Y realizar las tareas de gestión del entorno como muestra la figura 44.

Figura 44. Acceso al curso de Moodle.

Siguiendo la guía de instalación de Wordpress [20] se puede instalar esta aplicación tanto para uso del profesorado o para aprendizaje del alumno, en la figura 45 se muestra la pantalla de configuración inicial, después de haber seguido los pasos previos datos en la guía.

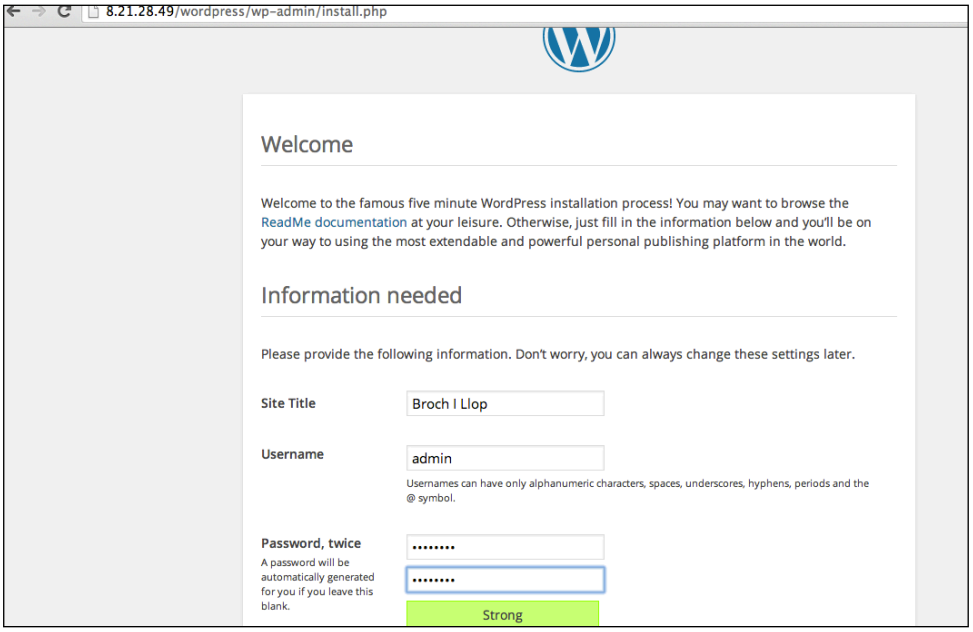


Figura 45. Proceso de instalación de Wordpress.

Una vez seguido el proceso de instalación, se puede entrar en la gestión propia de Wordpress, donde ya se pueden crear nuevos posts y tratar sobre el tema que se quiera de este CMS. La figura 46 muestra un ejemplo de un posible blog del instituto.

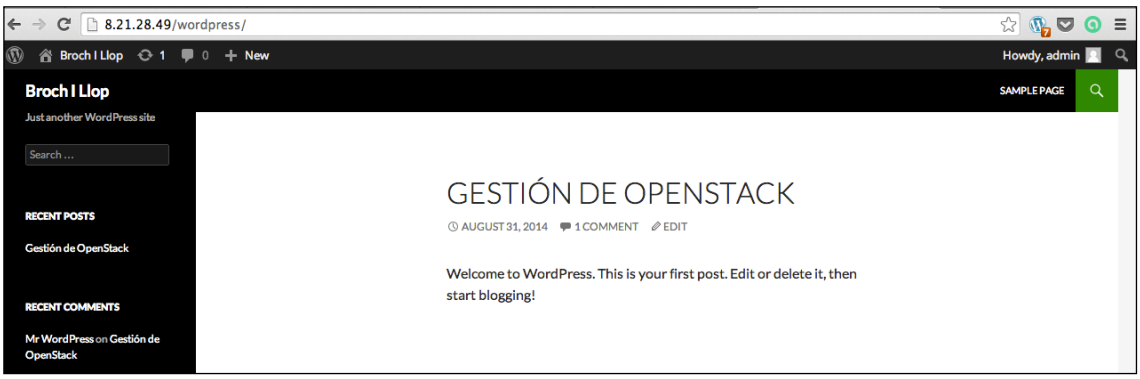


Figura 46. Ejemplo de aplicación Wordpress.

La tercera aplicación que falta por instalar es OwnCloud, se podría decir que es una aplicación parecida a DropBox, aunque dispone de mayores prestaciones. Para realizar la implementación de la misma se pueden seguir los pasos de instalación de la siguiente guía de instalación de OwnCloud [21] extraída como las anteriores de la documentación que ofrece el proveedor de servicios en la nube DigitalOcean [22].

Si se siguen los sencillos pasos de la guía de instalación, se obtiene la aplicación OwnCloud instalada como se muestra en la figura 47.

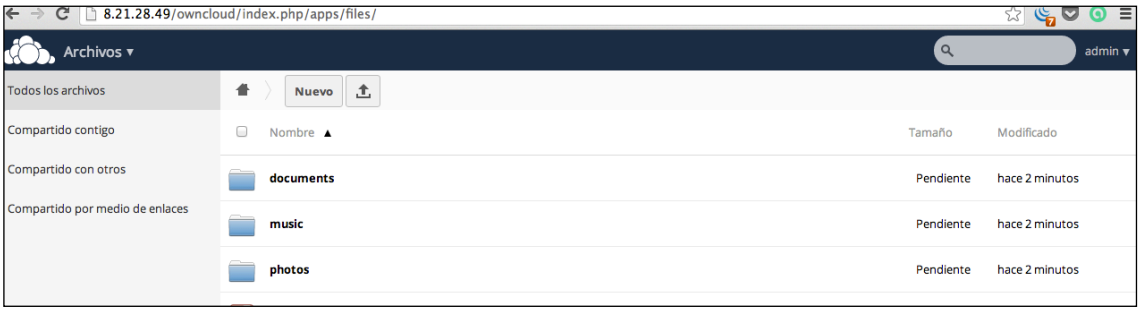


Figura 47.- Interfaz de OwnCloud.

4.4.3.5. Instantáneas de las instancias

Una vez instaladas estas tres aplicaciones, se puede ver que OpenStack ofrece más funcionalidades para realizar con las instancias. Por ejemplo se puede hacer una copia de seguridad de las mismas con las instantáneas o *"snap-shot".*

Si se pulsa sobre la instancia, crear instantánea, el sistema pedirá un nombre de la misma y nos generará la instantánea. A partir de aquí se puede tratar la instantánea como una imagen y crear una instancia a partir de ella. En la figura 48 se muestra la nueva imagen creada a partir de una instantánea.

Figura 48.- Imagen creada a partir de instantánea de una instancia.

Cuando se crea una imagen o una instantánea se puede indicar si la imagen es pública o privada. Si se quiere compartir entre proyectos o dejar solo disponible para el nuestro. También existe un campo protected, para no borrar el proyecto de forma accidental.

Se puede automatizar por línea de comandos o mediante script de python para que se hagan instántaneas y de este modo mantener copias del estado. Hay que recordar que aunque existen las instantáneas y que se pueden recurrir a ellas como copias de seguridad, este mecanismo no sustituye a las copias físicas de la infraestructura.

4.4.3.6. Gestión de Volúmenes

Siguiendo con la gestión del entorno OpenStack, en el apartado incluido también en cómputo del Dashboard Horizon, está el servicio referido a volúmenes.

Los volúmenes dependen del servicio OpenStack Block Storage (Cinder), que consiste en un almacenamiento de datos de forma persistente. Realmente es un dispositivo de bloque que se puede asociar y desasociar a una instancia cuando se desee. Además existe la posibilidad de crear instancias a partir de volúmenes, de esta forma al borrar la instancia, los datos que estaban creados en el volumen no se pierden. En la figura 49 se muestra el apartado referido a los volúmenes.

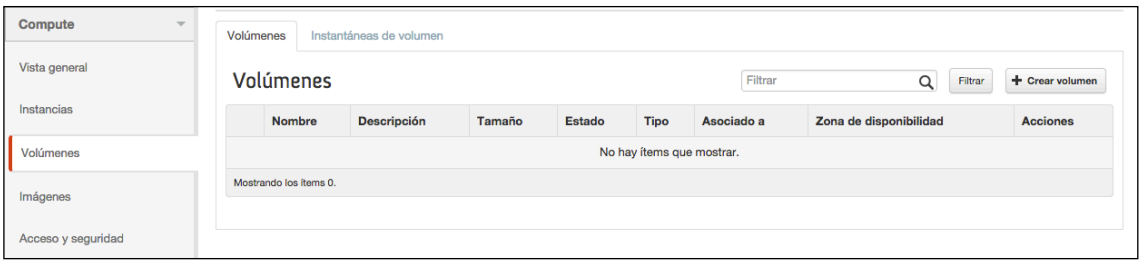


Figura 49.- Gestión de Volúmenes.

Los volúmenes se pueden redimensionar de tamaño y por lo tanto son muy útiles para guardar datos de las aplicaciones.

Para crear un volumen es tan sencillo como pulsar el botón de crear volumen y poner el nombre, el tamaño del volumen inicial, se puede elegir entre un volumen vacío, aquí lo que se hace es asociar un disco a la imagen, o crear un volumen a partir de una imagen o un volumen. Puede ser útil para usar imágenes ISO para que los alumnos instalen los sistemas operativos. La figura 50 muestra como se realiza la configuración de un volumen.

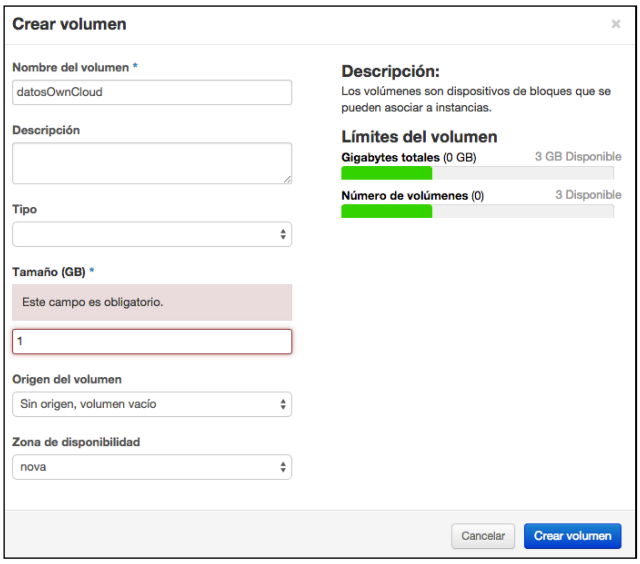


Figura 50.- Proceso de creación de volumen.

4.4.3.7. Gestión de Almacén de Objetos

Por último, podemos ver mediante Horizon la gestión del almacenamiento de objetos. Se realiza a través de OpenStack Object Storage (swift). Es un almacenamiento para grandes volúmenes, pensado para guardar vídeos, imágenes, copias de seguridad, escaneo de documentación, etc..

La figura 51 muestra un ejemplo de contenedores. En el contenedor hay creados dos contenedores vídeos y v1. Donde v1 tiene incluido un fichero llamado brochillop.mp4, v45 y video1.mp4.

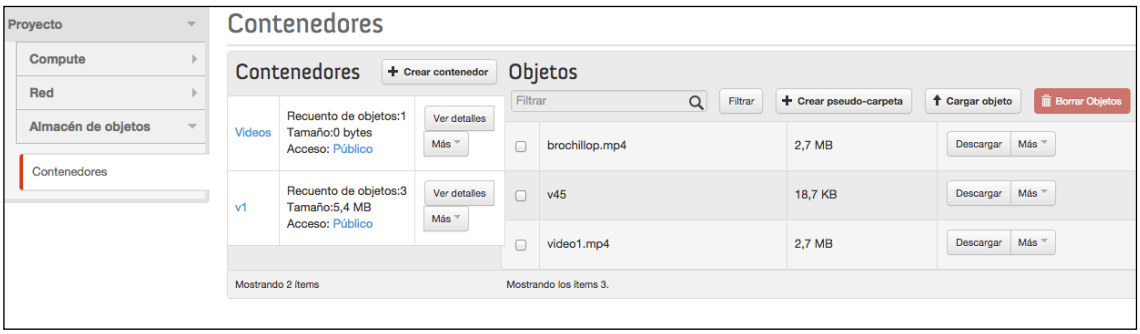


Figura 51. Contenedores para almacén de objetos.

Se puede acceder a la información de los contenedores a través de la URL. En la figura 52 se muestra como desde el apartado *Cómputo, Acceso y seguridad*, se puede acceder a la pestaña *Acceso a la API* donde están referenciadas todas las URLs de las APIs.

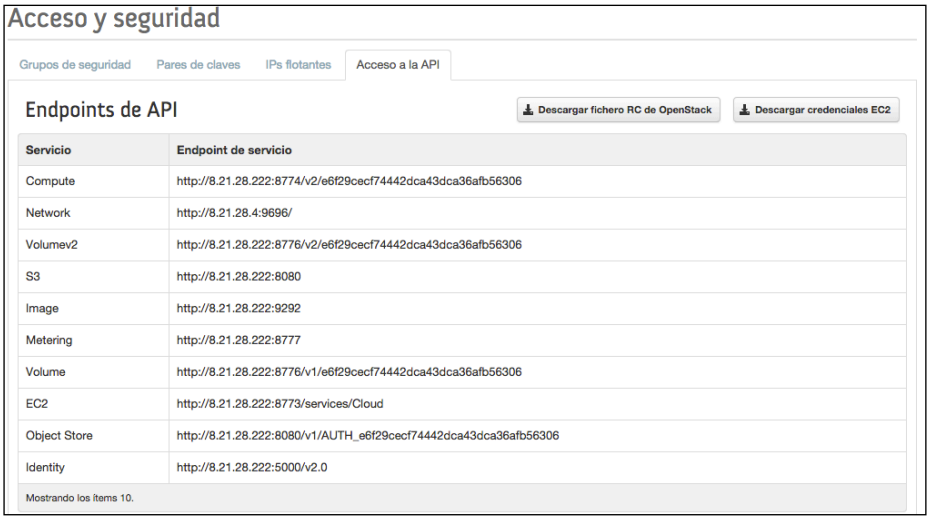


Figura 52.- Acceso a las API de OpenStack.

En concreto para el acceso al objeto v1 se realiza mediante la API referenciada en Object Store. Por ejemplo si se quiere ver el vídeo cargado en v1 que lleva por nombre brochillop.mp4, se debe escribir en la barra de direcciones del navegador la URL asociada al Object Store como recoge la figura 53.

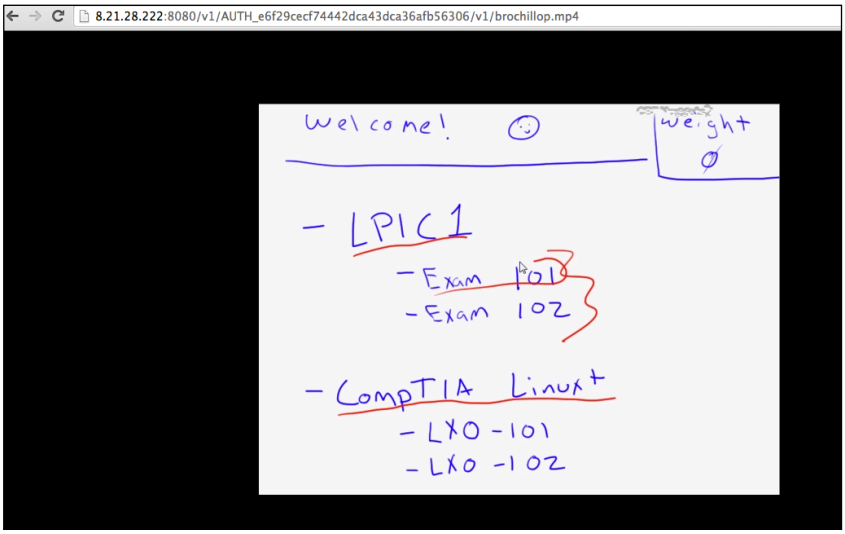


Figura 53.- Visualización del vídeo guardado en el contenedor v1.

Utilizando la funcionalidad de los contenedores se puede facilitar al profesorado el acceso a vídeos educativos que se hayan descargado anteriormente, por ejemplo con la aplicación DownloadHelper o vídeos que les presten las editoriales para acompañar a la docencia.

Algunos institutos no tienen la posibilidad de acceso a Internet, por tanto esta funcionalidad es bastante importante, porque la red OpenStack puede estar construida también como dos redes internas proporcionando acceso al material escolar.

Capítulo 5

5. Implementación de la Infraestructura OpenStack

5.1 Introducción

En el "*apartado 4.4 Prototipado con TryStack*" se ha visto como realizar la gestión de OpenStack gracias a la infraestructura configurada en la nube por el equipo de TryStack. Se ha aprendido a gestionar, ver las funcionalidades e instalación de las aplicaciones en este entorno de tipo test.

Para implementar una infraestructura OpenStack privada, se deben adquirir los equipos como se ha comentado en el apartado "*3.3 Estimación de los recursos del proyecto"* , o en nuestro caso dada la naturaleza de investigación del proyecto, se ha realizado la implementación de la misma en el equipo que nos ha facilitado el Instituto "Broch i Llop".

A continuación se va a ver como implementar OpenStack en una única máquina física o virtual mediante DevStack, en varias máquinas virtuales con agentes de automatización o instalando las máquinas y lanzando scripts. O realizando una instalación paso a paso de los servicios.

5.2. Implementación de DevStack

5.2.1. Instalación de DevStack sobre máquina física y virtual

Una vez adquirido el equipo con las características explicadas en el *"apartado 3.3 Estimación de los recursos del proyecto"*, se instala el sistema operativo Ubuntu 12.04 LTS.

La razón de instalar el sistema operativo Ubuntu 12.04 LTS como software base del ordenador para realizar las implementaciones tanto de la plataforma DevStack y OpenStack, se deben a que en el momento de la realización del proyecto, esta versión de la distribución Ubuntu, era la que soportada la infraestructura DevStack [23].

DevStack es una plataforma para aprender sobre OpenStack y para desarrollar implementaciones sobre el mismo. En la página web del proyecto DevStack , comentan los pasos que se deben seguir para su instalación, ya sea sobre una máquina física o sobre una máquina virtual. La opción de instalar DevStack sobre máquina física ofrece más rendimiento ya que la virtualización se

realiza sobre KVM. En el caso de una máquina virtual se tiene un menor rendimiento y las instancias se ejecutan con el emulador QEMU.

La figura 54 muestra estas dos opciones de instalación, y el link a las guías de implementación correspondiente.

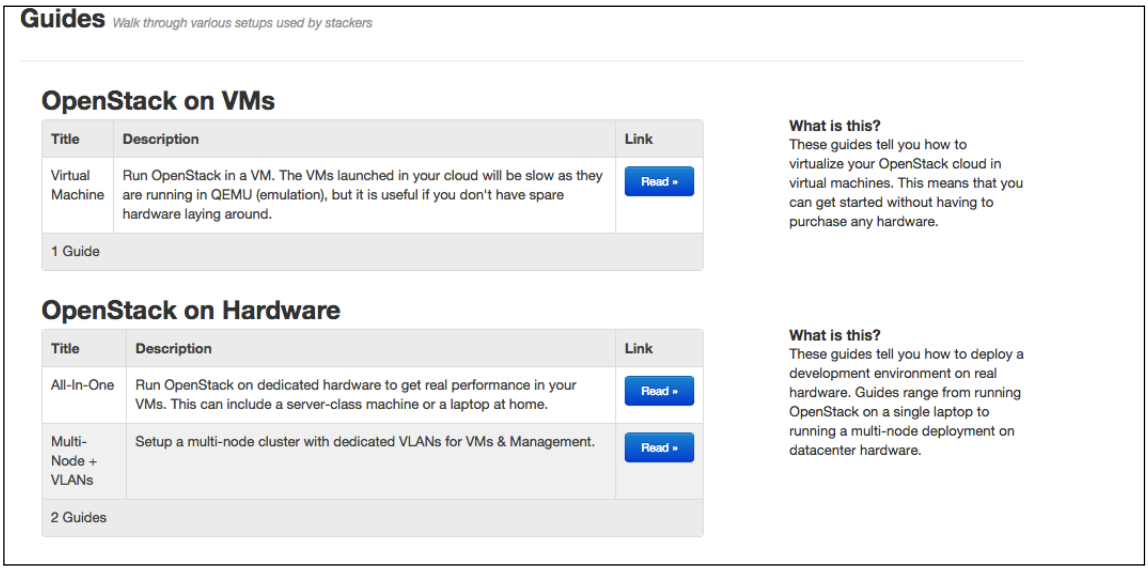


Figura 54.- Implementaciones DevStack sobre máquina física y virtual.Fuente:DevStack [22].

Para realizar el montaje de DevStack en máquina virtual, primero se ha tenido que instalar el software para virtualización *VirtualBox*. Resulta conveniente tener una máquina virtual preparada con el sistema operativo Ubuntu con entorno gráfico para usar en las instalaciones de DevStack y en las implementaciones de los nodos de OpenStack. Esta máquina debería ser clonada para realizar las instalaciones a partir de ella, de esta manera siempre tenemos un entorno límpio y podemos acudir a él en caso de errores en las instalaciones.

Respecto a la instalación de DevStack, tanto la instalación para máquina física[24], como la instalación para máquina virtual [25] es muy sencilla, y siguiendo las guías de instalación se logra tener el entorno implementado en un breve período de tiempo

Como la gestión de OpenStack mediante la interfaz web Horizon se ha mostrado en el *"apartado 4.4 Prototipado con TryStack"* ; con DevStack se va a mostrar lo que no se podía ver con la anterior plataforma de test. Con DevStack se va a enseñar la parte de gestión de Horizon referida a usuarios, proyectos y personalización del dashboard. Además de gestionar la línea de comandos para la creación de instancias, redes, volúmenes,etc.

5.2.2. Gestión de Usuarios y Proyectos a través de Horizon

5.2.2.1. Consideraciones sobre la instalación DevStack

Una vez se han seguido las guías de instalación, por ejemplo para la máquina virtual [24], y situado en el directorio *devstack,* al intentar realizar la instalación con el usuario root no dejará hacerla por temas de permisos. Se debe añadir un usuario para realizar la instalación, se puede hacer ejecutando el script que está en *"/devstack/tools/create-stack-user.sh".* Generará el usuario *stack* que sí que podrá lanzar la instalación.

El siguiente paso es realizar la conexión con el usuario *stack* con el comando *"su stack"* y ejecutar *"./stack.sh".* En este momento arranca el proceso de instalación que nos irá pidiendo contraseñas para los diferentes servicios. Si se deja alguna en blanco, el sistema las genera de forma aleatoria.

5.2.2.2. Acceso a DevStack desde Horizon

Para acceder a DevStack, se inicia el navegador y se escribe *localhost*, nos aparece la pantalla de login del entorno DevStack como muestra la figura 55.

Figura 55. -Login de acceso a DevStack.

Por defecto DevStack se configura con un usuario *"admin"* con rol de superusuario y un usuario *"demo"* con rol de usuario normal y las contraseñas se guardan en el fichero *local.conf* como muestra la figura 56.

```
 GNU nano 2.2.6          Archivo: local.conf

[[local|localrc]]
# Default passwords
ADMIN_PASSWORD=devstack
MYSQL_PASSWORD=devstack
RABBIT_PASSWORD=devstack
SERVICE_PASSWORD=devstack
SERVICE_TOKEN=devstack

SCREEN_LOGDIR=/opt/stack/logs

Q_PLUGIN="openvswitch"
ENABLE_TENANT_TUNNELS=True
```

Figura 56.- Fichero local.conf con las contraseñas de los servicios.

5.2.2.3. Gestión de proyectos y usuarios desde Horizon

Una vez se accede a la plataforma DevStack con admin, se observa que existen diferencias con la infraestructura TryStack. Aparece un nuevo menú llamado *"Panel de Identidad"*, ya que el usuario admin puede crear y administrar proyectos y usuarios. En la figura 57 se observan estos cambios respecto a un usuario sin privilegios.

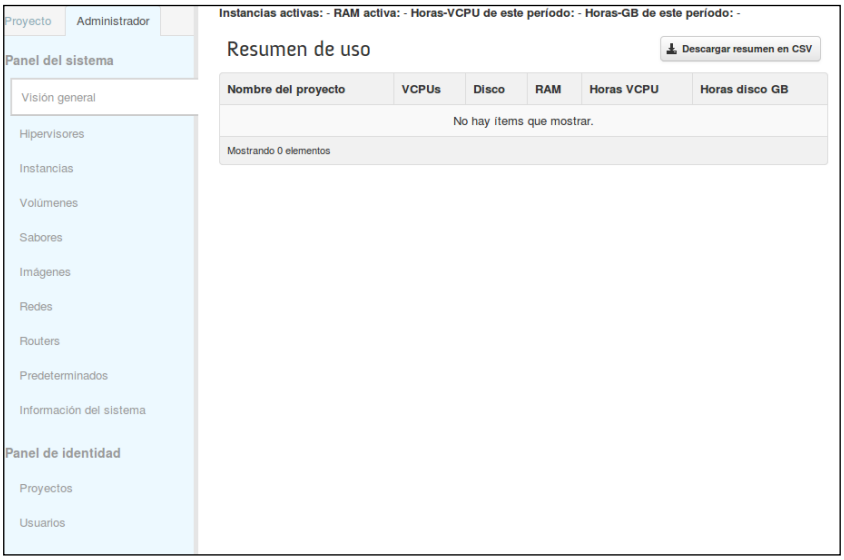


Figura 57.- Gestión de DevStack con nuevo panel de Identidad.

En este nuevo panel se pueden crear proyectos, borrar proyectos, editar el proyecto e introducir nuevos miembros. En la figura 58 se observa la creación de un nuevo proyecto llamado "Ies Broch i Llop".

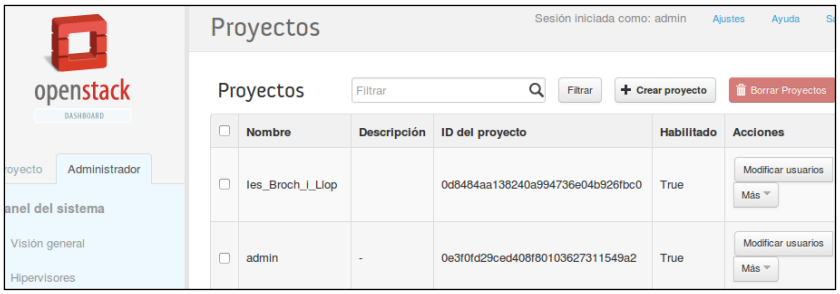


Figura 58.- Creación de un nuevo proyecto "Ies Broch i Llop"

Podemos administrar las cuotas para los usuarios encargados del proyecto. Es decir administrar el número de instancias, CPU, Volúmenes, instantáneas, RAM, Grupos de Seguridad, características de red,etc.. La figura 59 muestra un ejemplo de la administración de cuotas.

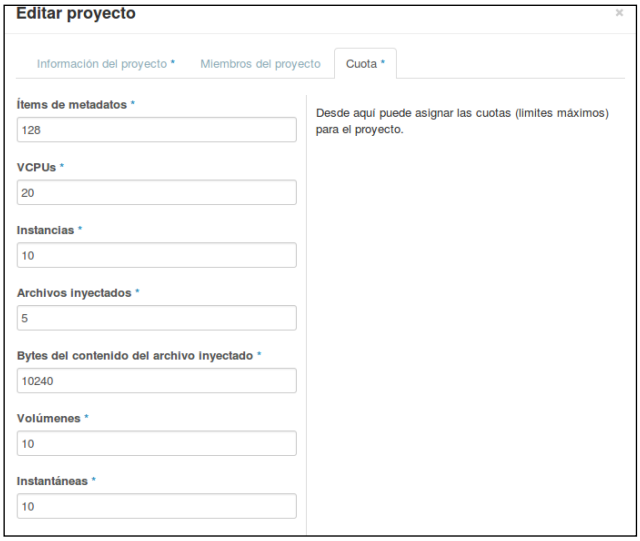


Figura 59.- Administración de cuotas por proyectos.

La gestión de usuarios también es muy sencilla. Se puede crear un nuevo usuario, nos pide características del mismo y contraseña, y se puede asignar este usuario a un proyecto. También un usuario puede ir asignado a más de un proyecto. Desde este apartado se pueden deshabilitar usuarios. En la figura 60 se muestra una captura de la creación de un usuario.

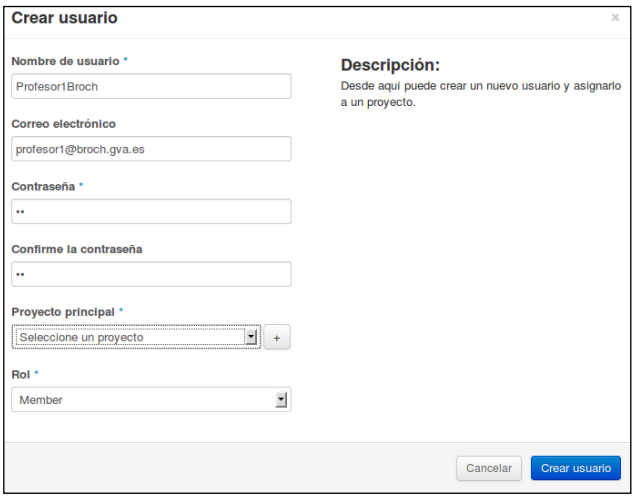


Figura 60.- Crear usuario y asociarlo a un proyecto.

En el entorno DevStack se pueden gestionar los servicios tal como se vió en el *"apartado 4.4 prototipado con TryStack"*, lo único que se debe de tener en cuenta es que cuando se termina de trabajar con el entorno y antes de reiniciar se debe ejecutar *"./unstack.sh"*. Una vez arrancado de nuevo el equipo se debe ejecutar *"./rejoin-stack.sh"* y se volverá a tener el entorno como antes. Si no se realizan estos pasos, se perderá toda la información del entorno DevStack.

Si lo que se desea es tener de nuevo un entorno totamente límpio, simplemente con ejecutar *"./clean.sh"* se tendrá de nuevo un entorno desde cero.

5.2.3. Personalización del dashboard Horizon

Horizon está desarrollado con el python framework-web Django. Se puede cambiar de apariencia totalmente siguiendo las guías de que proporciona OpenStack. Se puede, desde crear un dashboard, cambiar su estructura, cambiar los paneles, siguiendo la guía *Building on Horizon* [26] o símplemente cambiar logotipos, títulos del sitio, títulos del dashboard; que es lo que se requería en el Instituto. Con la guía *Customizing Horizon* [27] se pueden lograr estos cambios.

A continuación en la figura 61 se muestra el cambio en la pantalla de login.

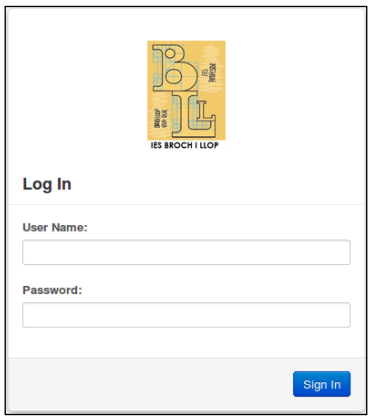


Figura 61.- Personalización de LogIn con logotipo del instituto

Se puede apreciar también, que se ha cambiado la leyenda del logo una vez accedido al componente Horizon en la figura 62.

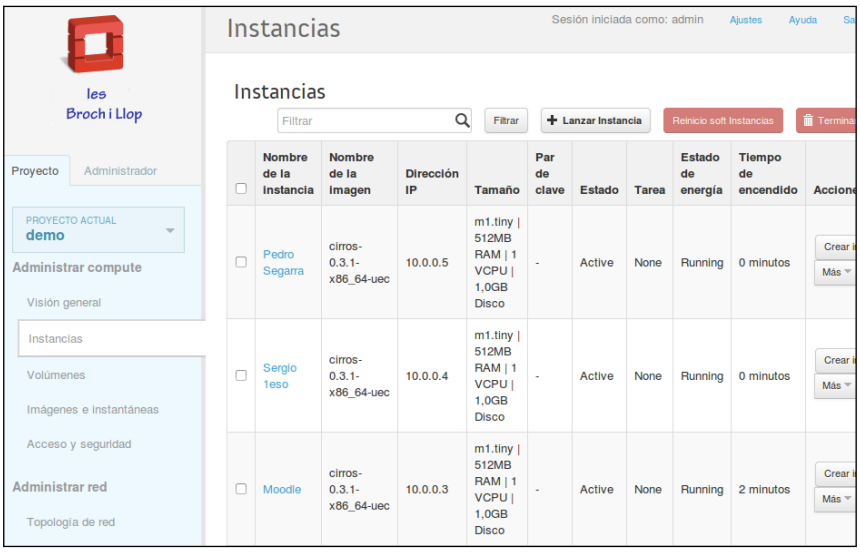


Figura 62.- Cambio de leyenda en gestión Horizon

5.2.4. Gestión de OpenStack a través de la línea de comandos.

5.2.4.1. Introducción a los command-line client

Uno de los aspectos interesantes de OpenStack es que se puede gestionar toda la infraestructura a través de la línea de comandos. De esta manera trabajando contra la API de OpenStack se pueden automatizar tareas, realizar una gestión más rápida del sistema, revisar los logs por si falla alguna instrucción,etc.

Cada servicio o componente de OpenStack posee su *"command-line client"* a través de la API del proyecto. Se está trabajando para unificar la línea de comandos a una única y no por servicio, pero de momento está separada por servicios. En la guía OpenStack end user guide [28] se detalla su uso.

Se puede utilizar la línea de comandos para administrar OpenStack desde cualquier ordenador, para ello se requiere ejecutar las credenciales de user y password. Se pueden descargar las credenciales a cualquier máquina y ejecutarlas con el comando *"source credencial.sh"*.

La tabla 7 muestra la lista de los *command-line client* por cada servicio de OpenStack , el paquete al que pertenece y la descripción.

Servicio	Cliente	Paquete	Descripción
Block Storage	**cinder**	**python-cinderclient**	**Crear y administrar volúmenes.**
Compute	**nova**	**python-novaclient**	**Crear y administrar imágenes, instancias y sabores.**
Database Service	**trove**	**python-troveclient**	**Crear y administrar bases de datos.**
Identity	**keystone**	**python-keystoneclient**	**Crear y administrar usuarios, proyectos, roles, credenciales y Create and manage users, tenants, roles, endpoints, and credentials.**
Image Service	**glance**	**python-glanceclient**	**Crear y administrar imágenes.**
Networking	**neutron**	**python-neutronclient**	**Configurar redes**
Object Storage	**swift**	**python-swiftclient**	**Recopilar estadísticas, actualización de los metadatos, subir, descargar y borrar objetos.**
Orchestration	**heat**	**python-heatclient**	**Detalles de procesos, recursos, eventos.**

Telemetry	**ceilometer**	**python-ceilometerclient**	**Crear y recoger mediciones**
Data Processing	**sahara**	**python-saharaclient**	**Crear y administrar Hadoop clusters.**

Tabla 7.- Command-line client

5.2.4.2. Instalación de los command-line client

Para poder usar el "command-line client" de cada uno de los servicios, antes se debe tener instalado el siguiente software que muestra la tabla 8. Si el equipo es un nodo *"por ejemplo de cómputo (nova)",* no hace falta instalar el *"cliente nova"* porque la propia instalación del servidor lo ha incorporado.

Prerrequisito	Descripción
Python 2.6 o posterior	No soporta Python 3
setuptoolspackage	Instalada por defecto en Mac OS X Los demás sistemas operativos lo pueden descargar de http://pypi.python.org/pypi/setuptools.
pip package	Se puede instalar pip package para cada sistema: **MacOs.** # easy_install pip **Microsoft Windows** Debemos tener puesto en el path C:/Python27\Scripts C:\>easy_install pip **Ubuntu** # apt-get install python-pip

Tabla 8.- Prerrequisitos de software.

La instalación del "command line client", es sencilla, por ejemplo para el servicio *nova* se realiza de la siguiente manera. Para los demás clientes, hay que cambiar por su correspondiente paquete, ver tabla 7.

Instalación de command-line

```
#pip install python-novaclient
```

Para actualizar clientes se hace de la siguiente forma:

```
# pip install --updgrade python-novaclient
```

Para borrar el cliente, con el siguiente comando:

```
# pip uninstall python-novaclient
```

Para poder usar el "command-line client" se deben tener permisos, se obtienen las credenciales de los servicios descargando de OpenStack el fichero *"openrc.sh".* Se puede generar en Horizon desde el apartado *Acceso y Seguridad* o crearlo manualmente con las contraseñas adecuadas. Luego se debe ejecutar *"source openrc.sh"* para transferir las credenciales.

Al ejecutar este comando nos pedirá el password del proyecto, es decir el password que pondríamos en la conexión de login al dashboard Horizon.

5.2.4.3. Uso de command-line client para gestión de OpenStack

A continuación se muestra como utilizar la línea de comandos para crear un proyecto, la carga de imágenes, crear instancias, etc.

El servicio *Identity Service* (Keystone) maneja la gestión de los proyectos (tenants). Por ejemplo podemos listar los proyectos con el comando:

```
keystone tenant-list
```

Para crear un proyecto con:

```
keystone tenant-create --name=BrochiLlop --enabled true
```

Una vez creado el proyecto se pueden ver sus características con *"nova quota-show",* por ejemplo, si este proyecto queremos que sea para un alumno, podemos limitar la creación de instancias a 2 y que la memoria Ram sea de 2GB con el siguiente comandos:

```
nova quota-update --ram 2048 --instances 2 id_delproyecto
```

Se puede ver como se han cambiado las características del proyecto en la figura 63, donde se muestran los comandos específicos sobre un *tenant*.

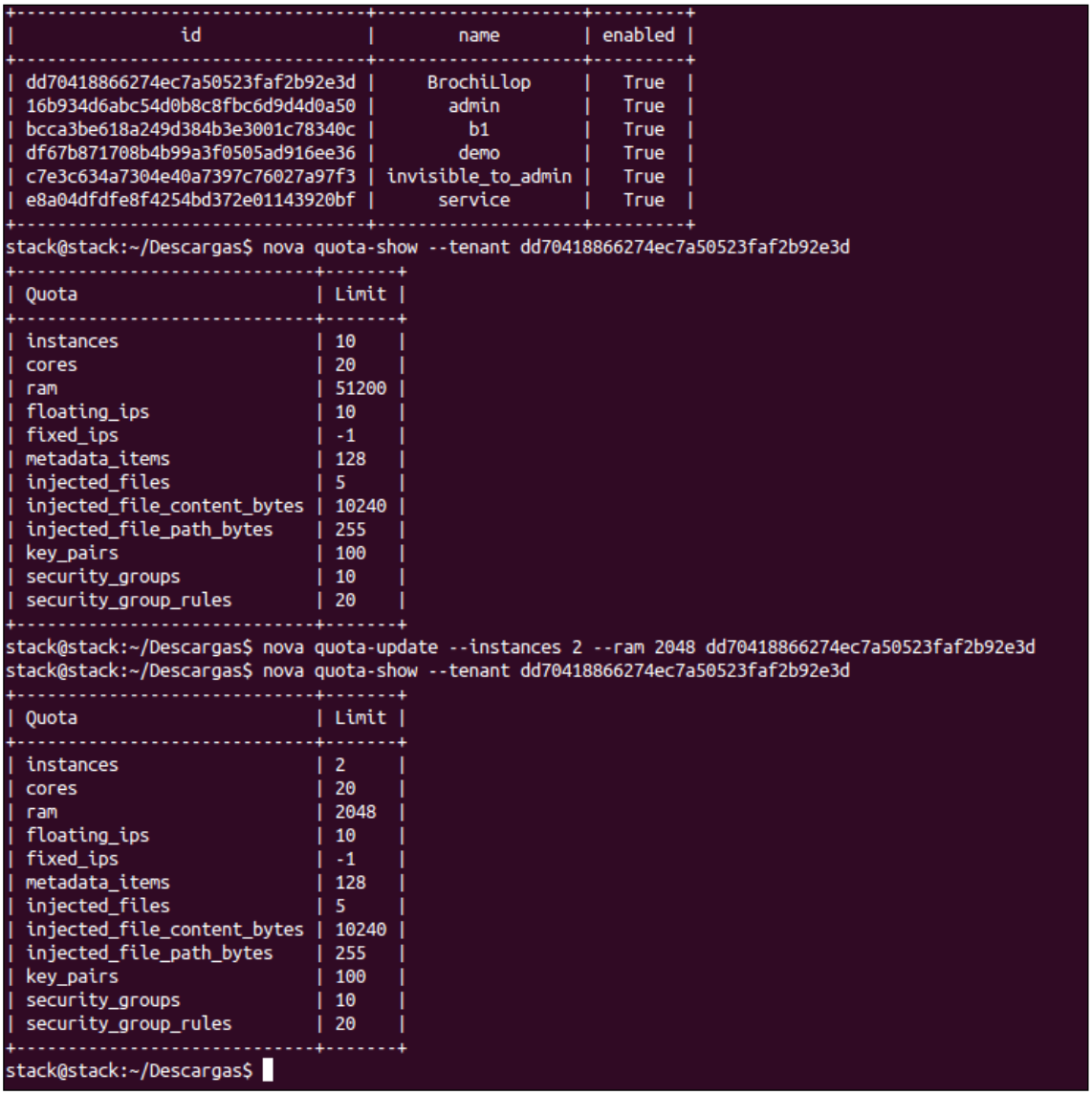


Figura 63.- Cambiar las características de instancias, memoria, cores, etc..

La misma gestión que se realizó con Horizon se puede hacer por línea de comandos, además se puede automatizar el proceso para crear proyectos para todos los alumnos del centro con las características adecuadas.

Por ejemplo se pueden ver que imágenes se tienen en el sistema con el comando *"nova image-list"*, las redes creadas con *"nova net-list"*, los sabores del sistema con *"nova flavor-list"*.

En la figura 64 se observan estos comandos lanzados sobre el entorno DevStack. Existe una guía de referencia a los comandos [29] donde se explica con detalle cada uno de los comandos asociados a los servicios, comentando los parámetros que se pueden pasar a cada uno de ellos.

```
stack@stack:~/Descargas$ nova image-list
+--------------------------------------+---------------------------------+--------+--------+
| ID                                   | Name                            | Status | Server |
+--------------------------------------+---------------------------------+--------+--------+
| 99bde870-4add-4d07-a951-8f410cb333d1 | cirros-0.3.1-x86_64-uec         | ACTIVE |        |
| 62f3124f-7aee-43d6-9fc3-fc249a58b602 | cirros-0.3.1-x86_64-uec-kernel  | ACTIVE |        |
| 50219a4e-edaa-477f-a52d-ea5ec5cd63c6 | cirros-0.3.1-x86_64-uec-ramdisk | ACTIVE |        |
+--------------------------------------+---------------------------------+--------+--------+
stack@stack:~/Descargas$ nova net-list
+--------------------------------------+-------------+------+
| ID                                   | Label       | CIDR |
+--------------------------------------+-------------+------+
| 05978d15-2a52-41f0-bdb5-63554d06355f | public      | -    |
| 912558cb-488d-4d2a-9108-0fe91eb3d49f | interna     | -    |
| 96617b32-116d-43b6-b79a-5d63ba1b5868 | ext-net     | -    |
| eff9edd7-9e0f-4fc4-8834-d4774e9dfd24 | interna-net | -    |
| f9de4df4-6623-4427-9bf2-24ae2dcbd279 | private     | -    |
+--------------------------------------+-------------+------+
stack@stack:~/Descargas$ nova flavor-list
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| 1  | m1.tiny   | 512       | 1    | 0         |      | 1     | 1.0         | True      |
| 2  | m1.small  | 2048      | 20   | 0         |      | 1     | 1.0         | True      |
| 3  | m1.medium | 4096      | 40   | 0         |      | 2     | 1.0         | True      |
| 4  | m1.large  | 8192      | 80   | 0         |      | 4     | 1.0         | True      |
| 5  | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1.0         | True      |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
stack@stack:~/Descargas$
```

Figura 64. Listado de imágenes, redes y sabores del proyecto BrochiLlop.

En la figura 65 se ve el comando para la carga de una imagen de Ubuntu 12.04 x86_64. Aquí se usa glance, que es el encargado de las imágenes.

```
stack@stack:~/Descargas$ nova image-list
+--------------------------------------+---------------------------------+--------+--------+
| ID                                   | Name                            | Status | Server |
+--------------------------------------+---------------------------------+--------+--------+
| 99bde870-4add-4d07-a951-8f410cb333d1 | cirros-0.3.1-x86_64-uec         | ACTIVE |        |
| 62f3124f-7aee-43d6-9fc3-fc249a58b602 | cirros-0.3.1-x86_64-uec-kernel  | ACTIVE |        |
| 50219a4e-edaa-477f-a52d-ea5ec5cd63c6 | cirros-0.3.1-x86_64-uec-ramdisk | ACTIVE |        |
+--------------------------------------+---------------------------------+--------+--------+
stack@stack:~/Descargas$ glance image-create --name 'Ubuntu 12.04 x86_64' --is-public=true --container-format=bare --disk-format=qcow2 -
-location http://uec-images.ubuntu.com/precise/current/precise-server-cloudimg-amd64-disk1.img
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| checksum         | None                                 |
| container_format | bare                                 |
| created_at       | 2014-09-05T04:13:34                  |
| deleted          | False                                |
| deleted_at       | None                                 |
| disk_format      | qcow2                                |
| id               | 8f9c2af3-edea-480c-8a34-6fbde8098f12 |
| is_public        | True                                 |
| min_disk         | 0                                    |
| min_ram          | 0                                    |
| name             | Ubuntu 12.04 x86_64                  |
| owner            | 16b934d6abc54d0b8c8fbc6d9d4d0a50     |
| protected        | False                                |
| size             | 261227008                            |
| status           | active                               |
| updated_at       | 2014-09-05T04:13:35                  |
+------------------+--------------------------------------+
stack@stack:~/Descargas$ nova image-list
+--------------------------------------+---------------------------------+--------+--------+
| ID                                   | Name                            | Status | Server |
+--------------------------------------+---------------------------------+--------+--------+
| 8f9c2af3-edea-480c-8a34-6fbde8098f12 | Ubuntu 12.04 x86_64             | ACTIVE |        |
| 99bde870-4add-4d07-a951-8f410cb333d1 | cirros-0.3.1-x86_64-uec         | ACTIVE |        |
| 62f3124f-7aee-43d6-9fc3-fc249a58b602 | cirros-0.3.1-x86_64-uec-kernel  | ACTIVE |        |
| 50219a4e-edaa-477f-a52d-ea5ec5cd63c6 | cirros-0.3.1-x86_64-uec-ramdisk | ACTIVE |        |
+--------------------------------------+---------------------------------+--------+--------+
stack@stack:~/Descargas$
```

Figura 65.- Carga de imagen al repositorio glance de Imágenes.

Como se ha comentado anteriormente se pueden cargar al sistema OpenStack gran cantidad de imágenes previamente configuradas por el fabricante para trabajar en la nube, aunque deben cumplir unas características de formato. En el siguiente enlace de OpenStack [17] se pueden obtener estas imágenes.

Para crear una instancia se realiza con el cliente nova. Por ejemplo la figura 66 muestra la creación de la instancia llamada MoodleBrochiLlop.

```
stack@stack:~/Descargas$ nova net-list
+--------------------------------------+---------+------+
| ID                                   | Label   | CIDR |
+--------------------------------------+---------+------+
| 05978d15-2a52-41f0-bdb5-63554d06355f | public  | -    |
| f9de4df4-6623-4427-9bf2-24ae2dcbd279 | private | -    |
+--------------------------------------+---------+------+
stack@stack:~/Descargas$ nova boot --flavor 1 --image 3eb09f82-33f9-4cfc-8fc5-e985eec65449 --key-name alumno --nic net
id=f9de4df4-6623-4427-9bf2-24ae2dcbd279 MoodleBrochiLlop
+--------------------------------------+----------------------------------------------+
| Property                             | Value                                        |
+--------------------------------------+----------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL                                       |
| OS-EXT-AZ:availability_zone          | nova                                         |
| OS-EXT-SRV-ATTR:host                 | -                                            |
| OS-EXT-SRV-ATTR:hypervisor_hostname  | -                                            |
| OS-EXT-SRV-ATTR:instance_name        | instance-00000001                            |
| OS-EXT-STS:power_state               | 0                                            |
| OS-EXT-STS:task_state                | scheduling                                   |
| OS-EXT-STS:vm_state                  | building                                     |
| OS-SRV-USG:launched_at               | -                                            |
| OS-SRV-USG:terminated_at             | -                                            |
| accessIPv4                           |                                              |
| accessIPv6                           |                                              |
| adminPass                            | gi2p8VrnC4S5                                 |
| config_drive                         |                                              |
| created                              | 2014-09-06T17:55:14Z                         |
| flavor                               | m1.tiny (1)                                  |
| hostId                               |                                              |
| id                                   | c80ac3ec-2fe2-4088-ab0c-503ad4a47ab9         |
| image                                | Ubuntu (3eb09f82-33f9-4cfc-8fc5-e985eec65449) |
| key_name                             | alumno                                       |
| metadata                             | {}                                           |
| name                                 | MoodleBrochiLlop                             |
| os-extended-volumes:volumes_attached | []                                           |
| progress                             | 0                                            |
| security_groups                      | default                                      |
| status                               | BUILD                                        |
| tenant_id                            | 16b934d6abc54d0b8c8fbc6d9d4d0a50             |
| updated                              | 2014-09-06T17:55:15Z                         |
| user_id                              | 8a279215b51c49ee834c5964e34f7a43             |
+--------------------------------------+----------------------------------------------+
stack@stack:~/Descargas$
```

Figura 66.- Creación de una instancia por línea de comandos.

5.3. Automatización de la infraestructura OpenStack

5.3.1. Introducción

Una vez se ha visto la gestión de OpenStack a través de la infraestructura Online TryStack, que nos facilita el aprendizaje de la administración de la plataforma; la gestión de DevStack a través de la línea de comandos y la adaptación de la apariencia de Horizon. El siguiente paso es ver como se puede automatizar la creación de máquinas virtuales, donde cada una de ellas gestione los nodos de OpenStack (Controller, Compute, Network y Storage).

Existen herramientas para realizar esta automatización. En concreto hay un repositorio en github [30] que facilita esta automatización basándose en las herramientas Ansible con Vagrant.

Con Ansible [31] se pueden desarrollar scripts de instalación de OpenStack basándonse en un lenguaje sencillo llamado YAML y con Vagrant [32] probarlos dentro de máquinas virtuales.

También se puede lograr una instalación robusta de OpenStack mediante máquinas virtuales siguiendo el repositorio [33] de Brian Brophy, que nos facilita la instalación gracias a unos scripts python de instalación de los nodos de compute, network y controller.

5.3.2. Instalación del Entorno con Ansible-Vagrant

Como explica la referencia [30] se deben instalar Vagrant, Ansible, python-netaddr y python-novaclient. En la figura 67 se muestran los pasos.

```
sudo pip install paramiko PyYAML Jinja2 netaddr python-novaclient
git clone git://github.com/ansible/ansible.git
cd ./ansible
source ./hacking/env-setup
```

Figura 67.- Instalación netaddr, python-novaclient, clonar ansible.

A continuación se obtiene la máquina virtual de Vagrant, como indica la figura 68.

```
vagrant box add precise64 http://files.vagrantup.com/precise64.box
```

Figura 68.- Descargar máquina virtual Vagrant.

Y clonamos el repositorio de Ansible preparado para OpenStack como muestra la figura 69.

```
git clone http://github.com/openstack-ansible/openstack-ansible.git
cd openstack-ansible
git submodule update --init
```

Figura 69.- Clonar Ansible, preparado para OpenStack.

Ahora ejecutando *make*, el sistema genera cuatro máquinas virtuales, con su sistema operativo Ubuntu, y con los nodos Controller, Compute, Network y Storage.

5.3.3. Explicación del entorno generado con Ansible-Vagrant

Al ejecutar *make*, el fichero *Makefile* realiza una serie de tareas, entre ellas, levantar las máquinas virtuales con la configuración de */testcases/standard,* instalar los servicios de OpenStack en las máquinas llamando a *openstack.yaml*. Además el fichero *ansible_hosts* indica las IPs a asignar a la infraestructura.

Con la "receta" *demo.yaml* se crea un proyecto de prueba para la plataforma totalmente funcional. En la figura 70 se muestra una parte de la estructura de directorios de la estructura ansible.

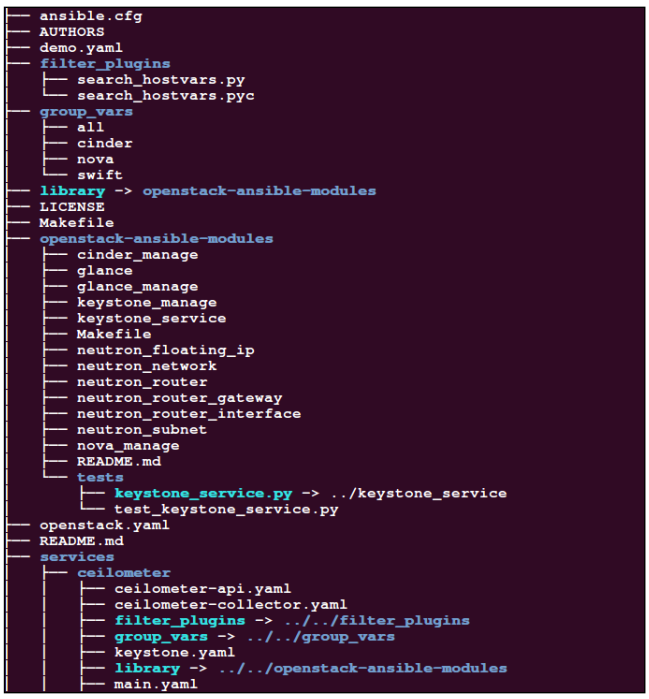


Figura 70.- Estructura directorio Ansible-OpenStack

Una vez el script ha terminado de ejecutarse tenemos cuatro máquinas virtuales (los cuatro nodos con la instalación de OpenStack). La instalación genera una estructura para la red de administración de esta manera (10.1.0.2 - controller, 10.1.0.3 - compute, 10.1.0.4 - neutron, 10.1.0.5 - storage).

La figura 71 muestra la estructura de red creada y la asignación de las IPs por defecto en la configuración.

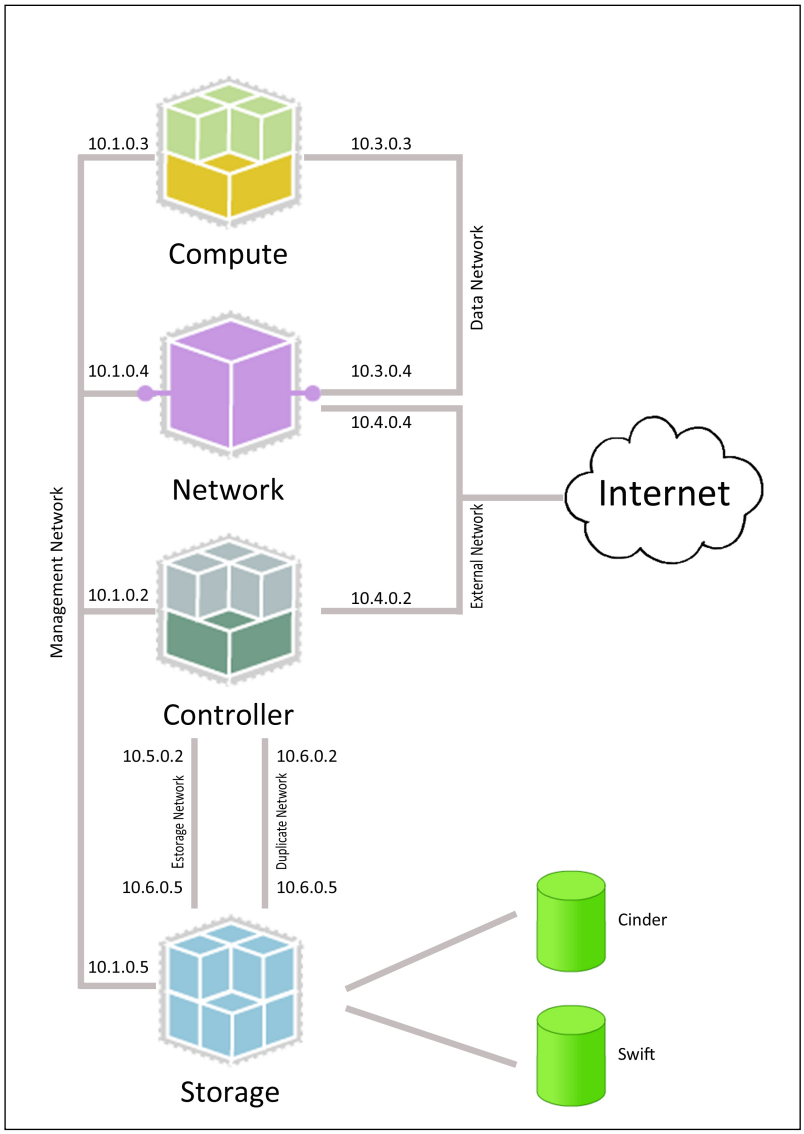


Figura 71.- División de la infraestructura por nodos.

5.3.4. Funcionalidad del entorno y adaptación al Instituto.

Con la automatización anterior se tiene un entorno totalmente funcional, y se pueden ajustar las "recetas" de Ansible para adaptar la instalación a la red del Instituto "Broch i Llop". De esta manera se puede implementar OpenStack en el Instituto, permitiendo que este entorno sirva como plataforma de acceso a las instancias de Moodle, Wordpress, Owncloud, vídeos, exámenes escaneados por parte del profesorado y por otro lado como entorno de formación para el alumno.

Las modificaciones que se tienen que realizar en la infraestructura Ansible para adaptar a la red del instituto *"red 10.2.0.0"* son las siguientes:

Modificar el fichero *ansible_hosts,* en concreto el apartado relacionado con la *api_network* y lo contenido en *[neutron:vars]*, a continuación en la figura 72 muestra la configuración por defecto de la plataforma y la figura 73 muestra los cambios realizados en la configuración para adaptarla a la estructura de red del Instituto. De esta forma se podrá acceder desde la red del instituto mediante la IP_flotante a las máquinas.

```
api_network = 10.2.0.0/16

external_network_name = public

[neutron:vars]
data_network = 10.3.0.0/16
external_network = 10.4.0.0/16
external_network_gateway_ip = 10.4.0.4
external_network_dns_nameservers = 8.8.8.8
external_network_allocation_pool_start = 10.4.10.100
external_network_allocation_pool_end = 10.4.250.250
```

Figura 72.- Configuración por defecto de la plataforma

```
api_network = 10.4.0.0/16

external_network_name = public

[neutron:vars]
data_network = 10.3.0.0/16

external_network = 10.2.0.0/16
external_network_gateway_ip = 10.2.0.4
external_network_dns_nameservers = 8.8.8.8
external_network_allocation_pool_start = 10.2.0.100
external_network_allocation_pool_end = 10.2.0.250
```

Figura 73.- Modificación de la api_network y la external_network.

Como la ip del Instituto es del rango de 10.2.0.0, se ha tenido que cambiar la red de *api_network* y en la red *external_network* poner la red del instituto. Lo único que se tiene que configurar un rango de la 100 a la 250 para que asigne IP_flotantes, y que éstas no interfieran con la red del instituto. El router DHCP del instituto se ha configurado para que asigne desde la 10.2.0.3 a la 10.2.0.99; quedando reservada la IP 10.2.0.2 para el router.

Además de los cambios anteriores, se debe configurar también el fichero Vagrantfile que es el que realiza la configuración de las máquinas virtuales. En concreto en la máquina "network" se cambia la *IP 10.4.0.4* por *10.2.0.4.* De esta manera se tiene la red adaptada a la del instituto. El fichero Vagrantfile guarda también el tamaño de memoria Ram de las máquinas. Aquí es conveniente aumentar la Ram a la máquina Compute, que pasa de tener 768MB a 4GB de Ram. La figura 74 muestra este cambio.

```
machine.vm.hostname = "compute"
machine.vm.provider :virtualbox do |v|
  v.customize ["modifyvm", :id, "--memory", 4096]
end
```

Figura 74.- Aumento de memoria en máquina compute.

Se puede aumentar del mismo modo el volumen de almacenamiento para cinder, que pasa de 4GB a 250GB y el almacenamiento de objetos que pasa de 1024MB a 600GB. La figura 75 muestra los cambios realizados.

```
machine.vm.hostname = "storage"
machine.vm.provider :virtualbox do |v|
  v.customize ["modifyvm", :id, "--memory", 512]
  v.customize ["createhd", "--filename", swift_file_to_disk, "--size", 256000]
  v.customize ["createhd", "--filename", cinder_file_to_disk, "--size", 614400]
  v.customize ["storageattach", :id, "--storagectl", "SATA Controller",
               "--port", 1, "--device", 0, "--type", "hdd",
               "--medium", swift_file_to_disk]
  v.customize ["storageattach", :id, "--storagectl", "SATA Controller",
```

Figura 75.- Cambio de tamaño de almacenamiento.

5.3.5. Consideraciones sobre la implementación Vagrant-Ansible

Una vez realizados todos estos cambios la infraestructura OpenStack es funcional en el Instituto, y se implementan todos las aplicaciones propuestas para el profesorado y la instalación de instancias para el alumno.

Sin embargo, la implementación Ansible-Vagrant, a pesar de ser funcional se ha sustituido por la implementación basada en el repositorio [33] por varios motivos, el primero es que sin *"causa aparente",* algunas veces no se ha podido acceder a Horizon a través de la web, sin embargo si que se podía gestionar la plataforma por la línea de comandos, pero no era el objetivo propuesto en el proyecto.

Un segundo motivo es que la creación de máquinas virtuales con Vagrant, deja a los nodos *"enjaulados",* aunque si que se puede acceder a los nodos mediante ssh, las máquinas virtuales están protegidas desde VirtualBox, sin poder gestionar estas máquinas de forma gráfica.

5.4. Implementación de OpenStack paso a paso

5.4.1. Introducción

Hemos visto como realizar la gestión de OpenStack mediante línea de comandos y mediante Horizon. También interactuar con un entorno online TryStack, donde no debíamos preocuparnos de la infraestructura, la instalación de DevStack en una única máquina virtual, y la instalación de una infraestructura funcional a través de varios nodos en diferentes máquinas virtuales con la automatización Ansible-Vagrant.

Para una implementación de OpenStack real sobre máquinas físicas para probar la infraestructura sin necesidad de requisitos de instancias y de aplicaciones; bastaría con adquirir el siguiente entorno:

- Controller: 1 procesador, 2GB de memoria y 5GB de almacenamiento.
- Network: 1 procesador, 512 MB de memoria y 5GB de almacenamiento.
- Compute: 1 procesador, 2GB de memoria y 10 GB de almacenamiento.

Como no se dispone de este entorno, se debe realizar la instalación en un entorno virtualizado. Es muy recomendable que se instale el sistema operativo de 64-bit, ya que las imágenes que se van a instanciar usan esta arquitectura y podrían fallar.

Se va a ver a través de la guía de instalación de Ubuntu [7], los servicios que se instalan en cada nodo y la estructura que se utiliza para hacer funcionales estos servicios. En concreto se va a ver la arquitectura two-node arquitecture with legacy networking (nova-network), reflejada en la figura 76.

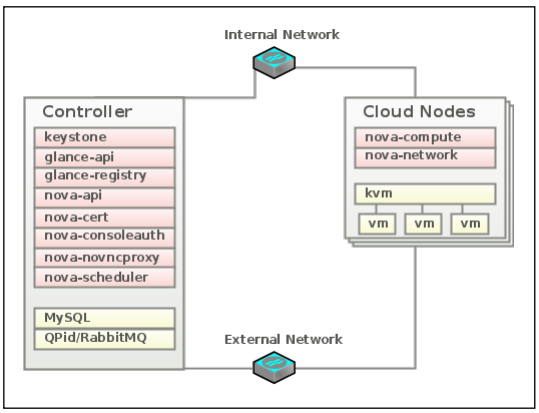


Figura 76.- Legacy networking (nova-network). Fuente: OpenStack

Dicha arquitectura maneja dos nodos, el nodo *Controller* que implementa los servicios básicos: *Keystone*, *Glance*, *Nova*, *Horizon*, *Mysql* y *RabbitMQ* y los opcionales *Cinder*, *Swift*, *Trove*, *Heat* y *Celiometer*. El nodo *Compute* que implementa *Nova Hypervisor* (*KVM*) y *Nova networking*.

5.4.2. Configuración básica del sistema

Como paso inicial de las máquinas, se deben configurar las tarjetas de red de cada nodo. Todos los nodos deben implementar el protocolo NTP. La figura 77 muestra la configuración básica de la red.

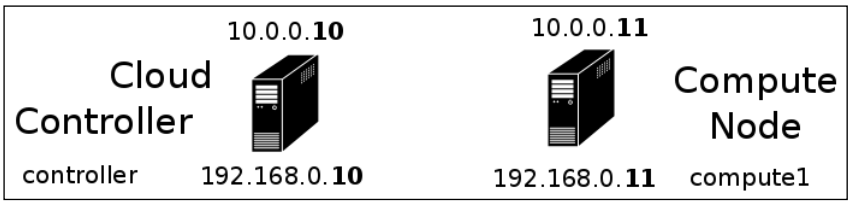


Figura 77.- Configuración de red Controller y Compute. Fuente OpenStack

Muchos de los servicios de OpenStack necesitan una base de datos para almacenar la información. En esta instalación la base de datos que se ha utilizado ha sido MySQL. En el nodo *Controller* se debe instalar MySQL database y MySQL Python Library. En el nodo *Compute* solamente MySQL Python Library.

Se instala en el *controller* con el comando:

```
#apt-get install python-mysqldb mysql-server
```

En la guía de referencia [7] se explica como se puede modificar el fichero */etc/mysql/my.cnf* para dar acceso a otros nodos al *controller* y modificar características específicas de MySQL como InnoDB, UTF-8, etc.

Se debe instalar en todas las máquinas (*Controller* y *Compute*), el paquete Openstack. Existe un repositorio en Ubuntu Cloud [34] para la descarga de cada versión. La instalación se realiza de la siguiente forma:

```
#apt-get install python-software-properties

#add-apt-repository cloud-archive:havana

#apt-get update && apt-get dist-upgrade

#reboot
```

En el nodo *Controller* se debe instalar el "messaging queue service", *apt-get install rabbitmq-server.* A partir de este momento ya se pueden instalar los servicios de OpenStack.

Es importante para que la conexión entre los nodos no falle que se escriba en el fichero de configuración de cada servicio la contraseña asignada de Rabbitmq, ya que sino el sistema no se comunicará correctamente.

5.4.3. Configuración de Identity Service (Keystone)

Tiene como función la gestión del usuario, el seguimiento de usuarios y permisos, y ofrecer el catálogo de servicios. El siguiente diagrama 78 muestra como funciona Keystone Identity Manager.

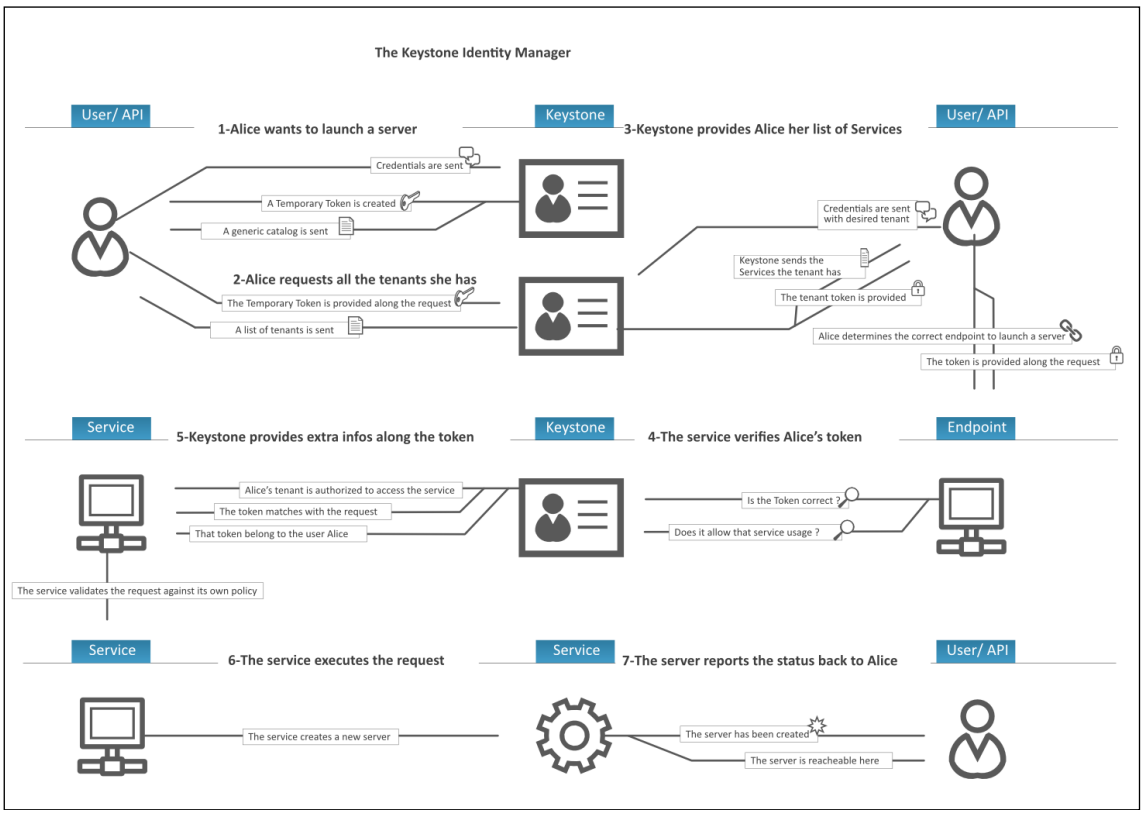


Figura 78.- Funcionamiento de Keystone Identity Manager. Fuente: OpenStack.

La instalación de Keystone se realiza con el siguiente comando *apt-get install keystone*, y usa MySQL para almacenar la información. Hay que modificar el fichero */etc/keystone/keystone.conf* para introducir la contraseña a establecer.

Además se debe crear la base de datos keystone con el siguiente comando:

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

Este servicio es el encargado de la autenticación, normalmente se suele hacer con user y password pero también se puede hacer a través de un token. Con las instrucciones siguientes se pasa el token y se dice donde se ejecuta el Identity Service.

```
# export OS_SERVICE_TOKEN=ADMIN_TOKEN
# export OS_SERVICE_ENDPOINT=http://controller:35357/v2.0
```

También se encarga de la autenticación de los proyectos -tenant-, de los usuarios y roles. Se configura con los siguientes comandos.

```
# keystone tenant-create --name=admin --description="Admin Tenant"
# keystone tenant-create --name=service --description="Service Tenant"
# keystone user-create --name=admin --pass=ADMIN_PASS
--email=admin@example.com
# keystone role-create --name=admin
# keystone user-role-add --user=admin --tenant=admin --role=admin
```

Keystone autentica a todos los servicios de la red, donde están localizados, por ello tanto el propio servicio Identity Service como los demás servicios se deben registrar y asociar a una API endpoint. A continuación se muestra como se registra el servicio keystone. Esta tarea se debe realizar para cada servicio.

```
# keystone service-create --name=keystone --type=identity --description="Keystone
Identity Service"

# keystone endpoint-create --service-id=identificador --
publicurl=http://controller:5000/v2.0
 --internalurl=http://controller:5000/v2.0
 --adminurl=http://controller:35357/v2.0
```

Para poder usar los diferentes servicios debemos tener la autorización, por ello se puede crear un fichero con los servicios exportados, por ejemplo guardarlos en *adminrc.sh* y ejecutar luego *source adminrc.sh*. El contenido de *adminrc.sh* puede ser similar al siguiente:

```
#export OS_USERNAME=admin
#export OS_PASSWORD=ADMIN_PASS
#export OS_TENANT_NAME=admin
#export OS_AUTH_URL=http://controller:35357/v2.0
```

5.4.4. Configuración de Image Service (Glance)

Image Service permite a los usuarios descubrir, registrar y recuperar imágenes de máquinas virtuales mediante el componente *glance-api*. Y con el componente *glance-registry*, almacena, registra y recupera información de metadatos sobre las imágenes.

Se denomina *Glance* al proyecto y utiliza una API REST para realizar consultas sobre los metadatos de las imágenes.

Se debe instalar en el nodo Controller con el comando *apt-get install glance python-glanceclient* . Además se deben editar los ficheros */etc/glance/glance-api.conf* y */etc/glance/glance-registry.conf* para modificar la contraseña.

Se debe crear la base de datos para el servicio *glance*, la tabla, el usuario, y realizar el proceso de autenticación contra *Keystone*. Todos estos pasos están perfectamente explicados en la guía de instalación de OpenStack sobre Ubuntu [7].

Las imágenes soportan diferentes formatos como se vió en el "*apartado 4.1.4.1. Gestionado por el servicio Compute (Nova)".* Se puede verificar el formato de las imágenes usando el comando file.

```
$ file cirros-0.3.1-x86_64-disk.img cirros-0.3.1-x86_64-disk.img: QEMU QCOW
Image (v2), 41126400 bytes
```

5.4.5. Configuración de Compute Service (Nova)

El servicio *Compute* es la parte principal del sistema IaaS. Interactúa con *Identity Service* para la autenticación, *Image Service* para las imágenes, y con el *Dashboard* para realizar la administración de la infraestructura.

Está dividido en áreas de funcionalidad:

API (*nova-api service*, responde a las llamadas del servicio de cómputo, y *nova-api-metadata service*, acepta llamadas de solicitud de metadatos por parte de las instancias).

Compute Core

nova-compute process, es un demonio para crear y terminar instancias a través del hipervisor.

nova-scheduler process, elige en que host se debe ejecutar una instancia.

nova-conductor module, interactúa con la base de datos.

Networking for VMs

nova-network, es un demonio que acepta tareas de red para manejar la red como interfaces, cambiar iptables. Esta funcionalidad se está migrando a Networking.

nova-dhcpbridge script. Se usa para seguimiento de ips. También se está migrando a Networking.

Console interface (*nova-consoleauth*, *nova-novncproxy*, *nova-xvpnvncproxy*, *nova-cert*, es un demonio que se ocupa de la autenticación, da acceso por vnc a las intancias y maneja certificados x509.

Image Management

nova-objectstore y *euca2ools client*, da soporte a S3 y EC2.

En definitiva, *Compute* es una colección de servicios que habilita el lanzamiento de instancias.

Para realizar su instalación se hace sobre el nodo *Controller* con el siguiente comando:

```
#apt-get install nova-novncproxy novnc nova-api nova-ajax-console-proxy nova-cert nova-conductor nova-consoleauth nova-doc nova-scheduler python-novaclient
```

Se debe establecer en el fichero de configuración de */etc/nova/nova.conf* la contraseña del servicio. Además se debe crear una base de datos para el servicio. Para ver todos los pasos de la instalación se puede consultar la referencia[7].

Es conveniente instalar otros nodos como nodo *Compute*. De esta manera se puede escalar horizontalmente de forma sencilla. En los otros nodos se debe instalar el hipervisor con el comando "*apt-get install nova-compute-kvm python-guestfs" .*

Si se está instalando el nodo compute sobre una máquina virtual, es conveniente ejecutar el comando *"egrep -c '(vmx|svm)' /proc/cpuinfo ".* De esta forma se puede saber si nuestro procesador soporta aceleración hardware para las máquinas virtuales. Si el resultado del comando es mayor que 0 es que soporta la aceleración y se podrá usar KVM, en caso contrario se hará simulación con QEMU.

Se puede modificar el apartado *[libvirt] libvirt=qemu* en */etc/nova/nova-compute.conf* para que funcione bajo la simulación.

5.4.6 Instalación de Networking Service (Neutron)

Este servicio es el encargado del manejo de la red del sistema (red, subnet, routers) y del manejo de los switches. Se debe instalar en el nodo *Controller*, y el primer paso es crear una base de datos "*neutron*". Una explicación detalla está en en la referencia [7].

En el nodo *controller* se modifica el fichero */etc/nova/nova.conf* con la API del network y se reinician los servicios. En el nodo de cómputo se incluye la interfaz externa para que reparta ips flotantes.

Se puede crear una red con el comando *nova network-create demo-net --bridge br100 --multi-host T --fixed-range-v4 192.168.0.24/29*

5.4.7. Añadir Dashboard (Horizon)

Horizon es la interfaz web que permite administrar los recursos y servicios de OpenStack de manera gráfica. Interactúa con OpenStack Compute a través de las OpenStack API.

Se debe instalar dashboard en el nodo que está instalado *Identity Service*, aquí es en el nodo *Controller*.

```
# apt-get install memcached libapache2-mod-wsgi openstack-dashboard
```

Los siguientes pasos de la configuración se pueden consultar en la referencia[7]. Aunque como todos los servicios, se debe crear una base de datos específica para este servicio.

Nota.- En la versión Havana se debía crear una base de datos dash para el servico dashboard. En la versión IceHouse ya no se requiere.

5.4.8. Añadir Block Storage Service (Cinder)

Es el encargado de la administración de los volúmenes, instantáneas de volúmenes y los tipos de volúmenes. Interactúa con el servicio Compute para proveer de volúmenes a las instancias.

Se va a instalar el servicio en el nodo Controller con el comando siguiente:

```
# apt-get install cinder-api cinder-scheduler
```

Como los demás servicios, se necesita crear una base de datos para su uso, y realizar una serie de modificaciones en los ficheros de configuración como explica la guía de referencia [7]. Se debe instalar un segundo nodo como Block Storage Service para proveer escalado.

5.4.9. Añadir Object Storage (Swift)

Se utiliza para el almacenamiento de objetos, es decir grandes volúmenes de información como vídeos, imágenes, etc.

Se puede instalar en un nodo para testeo, pero se debería instalar en múltiples nodos para dar redundancia y alta disponibilidad. Para instalarlo en un nodo se puede seguir la guía Swift All in One [35].

El comando para la instalación es *"apt-get install swift-account swift-container swift-object xfsprogs"* , en la guía [7] se explica detalladamente el proceso.

5.4.10 Consideraciones de la instalación paso a paso

Al igual que sucedía en la implementación de DevStack, para realizar la *instalación paso a paso* es conveniente tener una máquina virtual configurada en Virtual Box con el sistema operativo Ubuntu en modo gráfico. Esta máquina virtual puede ser utilizada como base, es decir tener un sistema siempre límpio, e ir clonando de ella, ya que la *instalación paso a paso* no es una tarea sencilla.

La *instalación paso a paso* está muy bien para ver como está construida la infraestructura por debajo, pero aunque se puede ser muy cuidadoso y seguir a detalle las instrucciones, se pueden encontrar problemas en el arranque de servicios, problemas con contraseñas, etc... Además es muy conveniente recurrir a la guía de instalación de la última versión de OpenStack, ya que han solucionado muchos de los problemas que los usuarios han ido teniendo con la instalación de la versión anterior.

También surgen muchos problemas con las interfaces de red, ya que la guía está indicada para realizar la instalación sobre máquinas físicas, y al tratar de realizarla sobre máquinas virtuales se debe tener en cuenta la configuración de las tarjetas en modo promíscuo, paravirtualización, bridge, túneles, etc.

La implementación de OpenStack *"paso a paso"* se utilizó en el proyecto para aprender la infraestructura a nivel de configuración y ver los diferentes servicios las bases de datos, ficheros "*.ini*", etc. y por ello se eligió por sencillez la arquitectura "two-nodes" que no implementa OpenStack Networking.

Dada la dificultad de instalar OpenStack paso a paso, además de consultar el foro de consultas técnicas [36] para ayudarnos a resolver los problemas que nos surjan a la hora de realizar las implementaciones, nos podemos apoyar en la automatización de instalaciones como la que se ha visto en el *"apartado 5.3. Automatización de la infraestructura OpenStack".* y en la implementación a través de scripts que se verá en el siguiente apartado, para facilitar la implementación de OpenStack.

5.5. Implementación de OpenStack a través de scripts

5.5.1. Introducción

Como se ha comentado anteriormente la implementación de OpenStack *paso a paso* no resulta sencilla, sobretodo en el tema de configuración de tarjetas de red, además la implementación automatizada Vagrant-Ansible no tenía un comportamiento estable en todos los reinicios de las máquinas virtuales, por eso la implementación final de la infraestructura OpenStack para el instituto se ha realizado adaptando la configuración de Brian Brophy [33].

5.5.2. Configuración del entorno OpenStack

La implementación del entorno OpenStack se ha realizado sobre tres nodos, el nodo *control*, *network* y *compute*. La figura 79 muestra la adaptación de la infraestructura al instituto.

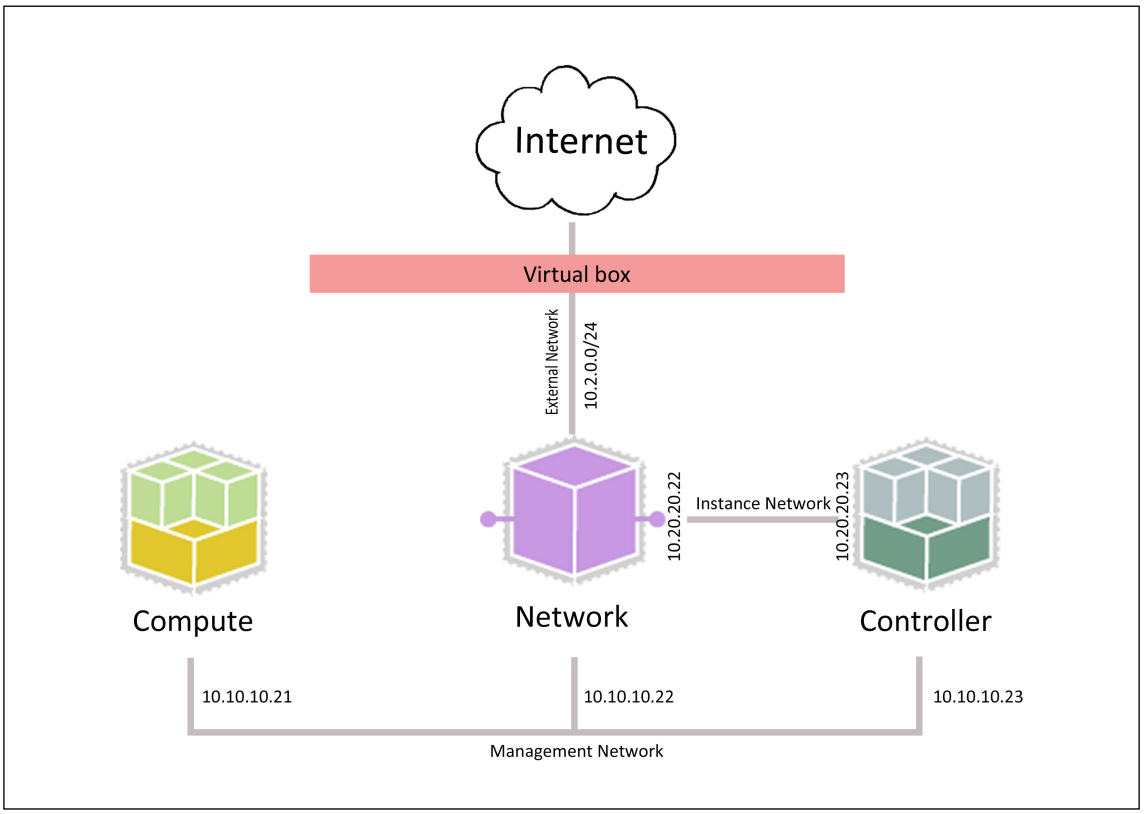


Figura 79.- Despliegue de la Infraestructura OpenStack.

5.5.2.1. Configuración e instalación del Nodo Controller

El nodo *Controller* se va a encargar de los siguientes servicios: NTP, RabbitMQ, MySQL, Keystone, Glance, Neutron, Nova, Cinder, Heat y Horizon.

La máquina virtual está configurada con las siguientes características: Ubuntu de 64-bit, memoria de 2GB, 1 procesador, disco duro de 20GB, y respecto a los adaptadores de red, eth0 configurado como Adaptador solo anfitrión de tipo Red paravirtualizada y en modo promíscuo (permitir todo). El adaptador eth1 configurado como NAT, y el modo promíscuo denegado. En total 2 adaptadores.

La configuración de la red se ha realizado de la siguiente manera, como recoge la imagen 80.

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 10.10.10.21
netmask 255.255.255.0

auto eth1
iface eth1 inet dhcp
```

Figura 80.- Configuración de la tarjeta de red del nodo Controller

Antes de configurar la red, resulta conveniente revisar la configuración de las interfaces de red de la máquina virtual, hay veces que el adaptador no empieza en eth0 sino en otro número, para corregirlo se puede ejecutar el comando:

```
rm -rf /etc/udev/rules.d/70-persistent-net.rules
```

Es interesante también revisar que la MAC de las interfaces de red corresponda con la MAC de los adaptadores configurados en VirtualBox para que no se produzcan errores posteriormente.

El siguiente paso a realizar, es descargar el repositorio como indica la referencia [33]. Se debe luego adaptar el fichero icehouse-install.ini a la dirección del instituto. Como muestra la figura 81.

```
[control]
# Management interface (ie 10.10.10.21/255.255.255.0)
network_interface_management = eth0
# Will be updated by control node installer
network_address_management = 10.10.10.21

[network]
# VM Instance interface (ie 10.20.20.22/255.255.255.0)
network_interface_instance = eth1
# External interface (ie 192.168.100.0/255.255.255.0)
network_interface_external = eth2
# NAT/Internet interface (ie connected to VirtualBox NAT/DHCP network)
network_interface_internet = eth3
# Within OpenStack, the network assigned as the external network
# An iptables rule will be added to /etc/rc.local to get VM/Instance Internet a$
#openstack_external_network = 192.168.100.0/24
openstack_external_network = 10.2.0.0./24
```

Figura 81.- Adaptación a la red del instituto

Una vez realizados estos cambios, se ejecuta el instalador del controlador con el comando.

```
python icehouse-setup-control-node.py
```

Cuando termina la instalación se tiene el nodo controlador instalado, se puede adaptar la apariencia de Horizon a la especificación del instituto.

5.5.2.2. Configuración e instalación del nodo Network

El nodo Network va a ejecutar los siguientes servicios: NTP y Neutron. La máquina virtual está configurada con las siguientes características: Ubuntu 64-bit, memoria de 2GB, 1 procesador, disco duro de 20GB, y respecto a los adaptadores de red, eth0,eth1,eth2 configurados como Adaptador solo anfitrión de tipo Red paravirtualizada y en modo promíscuo (permitir todo). El adaptador eth4 configurado como NAT, y el modo promíscuo denegado. En total 4 adaptadores.

Se debe instalar ethtool, ya que nos permite configurar la tarjeta de red y cambiar de modos, parámetros. La figura 82 muestra la configuración de la red del nodo Network.

```
auto eth0
iface eth0 inet static
address 10.10.10.22
netmask 255.255.255.0

auto eth1
iface eth1 inet static
address 10.20.20.22
netmask 255.255.255.0

auto eth2
iface eth2 inet manual
up ifconfig $IFACE 0.0.0.0 up
up ip link set $IFACE promisc on
down ip link set $IFACE promisc off
down ifconfigi $IFACE down
post-up ethtool -K $IFACE gro off

auto br-ex
iface br-ex inet static
address 10.2.0.2
netmask 255.255.255.0
post-up ethtool -K $IFACE gro off

auto eth3
iface eth3 inet dhcp
```

Figura 82.- Configuración de los adaptadores del nodo Network

Ahora se debe cambiar la ip externa en el fichero icehouse-install.ini, y se puede ejecutar el comando *python icehouse-setup-network-node.py*, que instalará el nodo Network. En este nodo hay que revisar que en */etc/rc.local* se haya añadido la regla iptables que realiza SNAT como muestra la figura 83.

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

iptables -t nat -A POSTROUTING -s 10.2.0.0/24 -j SNAT --to-source `ip -4 -o addr
 show eth3 | sed s/.*inets*// | cut -f1 -d/`
exit 0
~
~
```

Figura 83.- Regla iptables para acceso a la IP flotante.

5.5.2.3. Configuración e instalación del nodo Compute

El nodo compute ejecuta los siguientes servicios: NTP, Nova y Neutron.

La máquina virtual está configurada con las siguientes características: Ubuntu 64-bit, memoria de 3GB, 1 procesador, disco duro de 20GB, y respecto a los adaptadores de red, eth0,eth1 configurados como Adaptador solo anfitrión de tipo Red paravirtualizada y en modo promíscuo (permitir todo). El adaptador eth3 configurado como NAT, y el modo promíscuo denegado.

La configuración de la red se muestra en la figura 84.

```
auto eth0
iface eth0 inet static
address 10.10.10.23
netmask 255.255.255.0

auto eth1
iface eth1 inet static
address 10.20.20.23
netmask 255.255.255.0

auto eth2
iface eth2 inet dhcp
```

Figura 84.- Configuración de red del nodo compute.

A continuación se ejecuta el comando python icehouse-setup-network-node.py que instalará el nodo compute.

5.5.2.4. Consideraciones de la instalación a través de script

La instalación a través de script puede parecer sencilla a primera vista, pero el proceso tampoco es sencillo. Se debe tener mucho cuidado en la configuración de las tarjetas de red, ya que se pueden cometer fallos a la hora de escribir la configuración, por ejemplo con el nodo network, el sistema no genera errores, a pesar de estar mal escritos los comandos.

Es importante también configurar Virtualbox para que acceda al entorno generado, esta configuración se realiza en Archivo --> Preferencias de Virtualbox como se muestra en la figura 85.

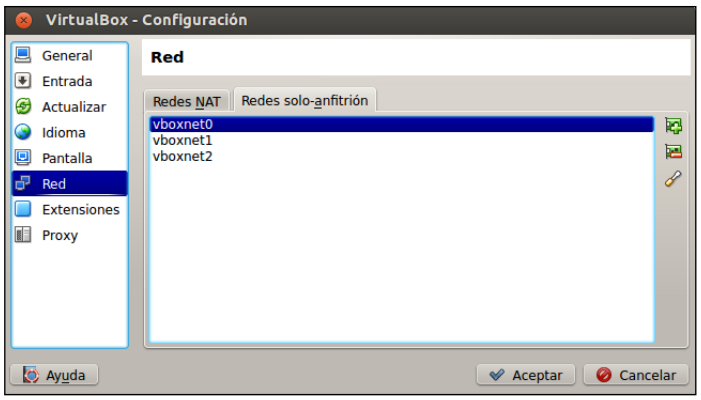


Figura 85.- Configuración de VirtualBox para acceso a las redes virtuales.

Además en cada una de las redes virtuales hay que configurar específicamente las ips apropiadas al rango de la red del entorno. En la figura 86 se muestra la configuración de vboxnet2 adaptada al instituto.

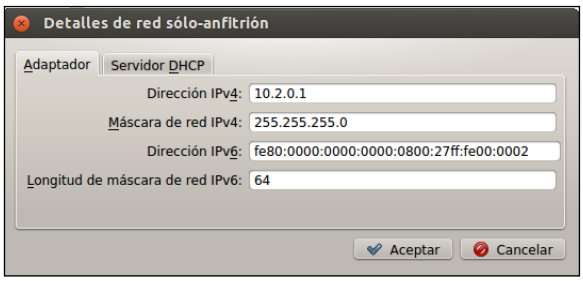


Figura 86.- Configuración de red virtual del instituto.

El proceso a veces puede fallar y dejar inacabada la ejecución de algún comando. También, se deben configurar correctamente el *hostname* y */etc/hosts*, ya que el sistema OpenStack a veces va a buscar *localhost* en vez de la IP.

Una vez instalado *OpenStack*, el manejo de la infraestructura es similar a los casos que hemos visto para *TryStack* y *DevStack*. La única consideración es a la hora de crear la red externa se debe elegir un rango de *10.2.0.100* a *10.2.0.254* para

no solapar con las direcciones del instituto que se asignan hasta la *10.2.0.99*. La figura 87 muestra los comandos para la creación de la red externa.

```
root@controller:/home/openstack# neutron subnet-create ext-net --name ext-subnet --allocation-pool start=1
0.2.0.100,end=10.2.0.254 --disable-dhcp --gateway 10.2.0.2 10.2.0.0/24
Created a new subnet:
+-------------------+--------------------------------------------+
| Field             | Value                                      |
+-------------------+--------------------------------------------+
| allocation_pools  | {"start": "10.2.0.100", "end": "10.2.0.254"} |
| cidr              | 10.2.0.0/24                                |
| dns_nameservers   |                                            |
| enable_dhcp       | False                                      |
| gateway_ip        | 10.2.0.2                                   |
| host_routes       |                                            |
| id                | 1c5b7b84-7156-4cdd-8235-61b0fa4c3c8c       |
| ip_version        | 4                                          |
| name              | ext-subnet                                 |
| network_id        | 402aa9ad-1da4-49d2-b63b-1cc6433a5259       |
| tenant_id         | fd886f449f5d4966b0878a7ee8574391           |
+-------------------+--------------------------------------------+
```

Figura 87.- Creación de la red Externa para acceso desde instituto

Cuando se consigue solucionar todos los inconvenientes, al final se tiene una instalación funcional, con las aplicaciones requeridas funcionando. La imagen 88 muestra una captura de las instancia generada de Moodle + Owncloud.

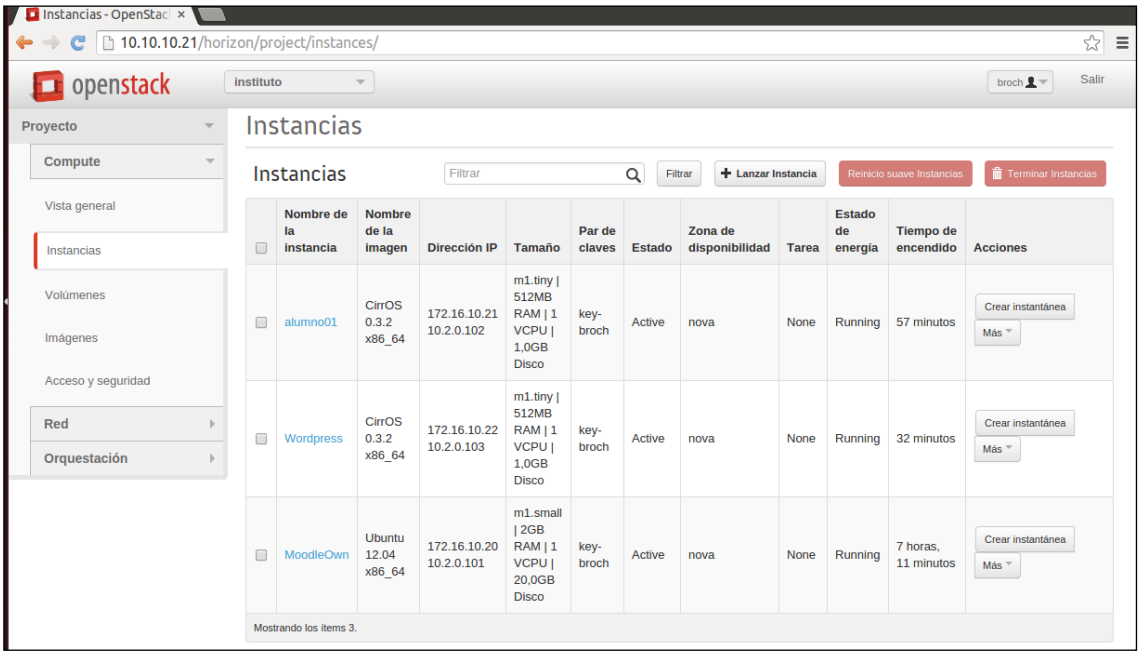


Figura 88.- Instancia instituto. IP's accesible desde las aulas

Capítulo 6

6. Análisis y pruebas de las instalaciones

6.1. Introducción

Una vez visto todas las variantes de implementación de la plataforma OpenStack que estaban planificadas en el proyecto (TryStack, DevStack, automatización e instalación *paso a paso*), se puede hacer un análisis de cada una de las variantes.

6.2. Análisis de TryStack

TryStack es ideal para realizar una rápida implementación de las aplicaciones, pruebas y aprendizaje de la plataforma. Como es totalmente funcional se puede "enseñar" rápidamente su uso y despliegue de las aplicaciones. Además sirve para aprender el modo de acceso por ssh, configurar servidores, gestión de Horizon, aprender como configurar la red, lanzar las imágenes, como interactúar con los volúmenes e instantáneas. Crear contenedores para guardar grandes volúmenes de información tales como vídeos, documentación, y acceder a través de la URL generada por la API de la aplicación.

El inconveniente de TryStack es que es un entorno de pruebas y no permite mantener las máquinas virtuales en uso más de 24 horas. Existen numerosos proveedores que ofrecen su servicio de cloud basado en OpenStack. Se puede contratar OpenStack a estos proveedores de servicios [37].

6.3. Análisis de DevStack

La implementación de DevStack requiere de algún conocimiento técnico para su instalación. Al principio puede resultar un poco liosa la instalación, decidir entre hacerlo en máquina virtual o física ya que cada una de ella requiere diferentes pasos. La documentación inicial no está muy clara, y hay que acudir a alguna guía más detallada.

Una vez se consigue tener acceso a la plataforma, hay que encontrar el usuario y contraseña en el fichero de configuración que está en los directorios de la instalación. Una vez obtenemos acceso a la plataforma, la gestión es similar a la de TryStack.

DevStack se utiliza para desarrollo y pruebas, ya que el entorno se instala de forma bastante rápida en un ordenador, y entre sus usos podemos cambiar la

apariencia de Horizon, seguir con el aprendizaje de la plataforma, esta vez, además de gestionarla a través de Horizon, se puede aprender a manejar todos los componentes a través de la línea de comandos.

Además podemos realizar todas las pruebas y cambios que se quieran a la plataforma, ya que ofrece un script para volver a dejar el entorno como se ha encontrado inicialmente.

6.4. Análisis de la Automatización

Existe muy buena documentación sobre OpenStack almacenada en los repositorios github. Muchos miembros de OpenStack están ayudando a la implementación de la infraestructura colgando instalaciones automatizadas de la plataforma, para entender el montaje sobre diferentes nodos (Controller, Compute, Storage, Networking).

Además de las referencias sobre automatización [30] y [33] que se han comentado en los apartados anteriores, existe un curso [38] sobre OpenStack en castellano donde los profesores explican como realizar la implementación de OpenStack y manejar la infraestructura de una manera sencilla.

La automatización realizada con la referencia [33] es la que se ha adaptado para la implantación de la infraestructura en el instituto.

6.5. Análisis de la Implementación paso a paso

La implementación de OpenStack sobre máquinas físicas no ha sido posible al no disponer de una infraestructura adecuada. Por ello la implementación de OpenStack se ha realizado sobre máquinas virtuales montadas sobre el equipo adquirido por el centro para la realización del proyecto.

No obstante el proyecto fué pensado como un estudio de la infraestructura, una investigación para ver como se podía implementar y gestionar una infraestructura cloud computing. Y como resultado realizar un despliegue de máquinas virtuales con las aplicaciones más demandadas por el profesorado y para el aprendizaje de los alumnos.

La instalación de OpenStack servicio a servicio, descubriendo que tecnología emplea por debajo la plataforma, ha servido para entender como funciona la gestión interna de OpenStack.

6.6. Pruebas de las instalaciones

En los apartados anteriores, concretamente en el apartado "*4.4. Prototipado con TryStack*" se enseña toda la gestión a través de Horizon (Gestión de redes, carga de imágenes, creación de instancias, asociación de Ip_flotante, acceso por SSH y configuración de las aplicaciones Moodle, Wordpress y Owncloud), creación de un volumen y manejo de objetos a través de la web.

En el apartado *"5.2.4. Gestión de OpenStack a través de la línea de comandos"*, se enseña como crear un proyecto, cambiar las características del mismo, ver las imágenes del sistema, las redes creadas, los sabores y cargar una imagen.

En el apartado "5.5 Creación de OpenStack a través de script" se muestra el entorno definitivo implantado en el instituto. En él se han generado las instancias de Moodle y Owncloud, Wordpress y una instancia para alumno. Hay que tener en cuenta que la máquina Compute, que es la que genera las instancias dispone de 3GB de Ram, y con esta configuración solamente es posible configurar un número limitado de instancias.

En la figura 89 se muestran algunas máquinas virtuales utilizadas para las pruebas e implementación del entorno OpenStack.

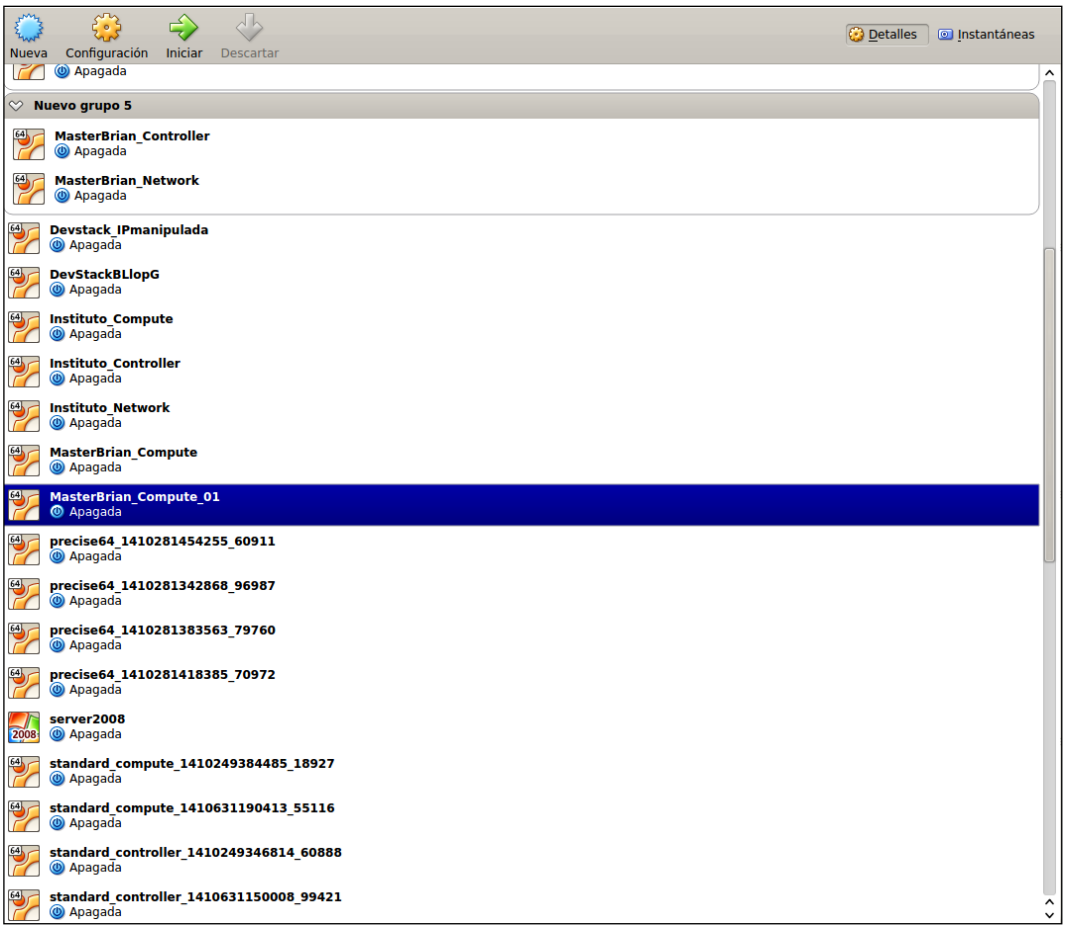


Figura 89.- Máquinas virtuales utilizadas en el proyecto.

Conclusiones

El proyecto está centrado en el proceso de aprendizaje de la plataforma OpenStack, en la gestión de la misma y en la instalación de unas aplicaciones básicas para el profesorado y el alumno.

Se ha logrado aprender la gestión de OpenStack tanto a nivel de gestión con Horizon, como la gestión a través de la línea de comandos. También se ha implementado una plataforma con máquinas virtuales para probar las aplicaciones Moodle, Wordpress, Owncloud, y visualización de vídeos a través de la red. Así como habilitar instancias con diferentes sistemas operativos para el aprendizaje del alumno.

La realización de la implementación a través de máquinas virtuales, ha permitido ver el funcionamiento de la infraestructura a nivel lógico, quedando pendiente ver el funcionamiento de la plataforma a nivel físico. De esta forma, no se ha visto como tratar el almacenamiento a través de SAN (Storage Area Network), ni la utilización de la infraestructura de red.

El aprendizaje de la infraestrura OpenStack puede complementarse con el estudio de otros módulos del mismo, como Telemetry para la monitorización de toda la plataforma, la implementación de OpenStack para alta disponibilidad y seguridad, así como "live migration", es decir paso de instancias de un nodo a otro.

Los fabricantes de hardware y de software están volcados con esta infraestructura que lleva apenas cuatro años de vida, y que desde su creación ya ha pasado por nueve versiones, lo que demuestra su rápida evolución.

Creo que es una buena apuesta seguir con el estudio de OpenStack, además de todo el conocimiento que se adquiere estudiando sus servicios, estos conocimientos son un requisito cada vez más demandado por las empresas de desarrollo en la nube.

Bibliografía

[1] NIST. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[2] Denoe. http://www.denoe.es/blog/

[3] MenteVisual. http://mentevisual.com.ec/cloudcomputing.htm

[4] blog.flux7.com.

http://blog.flux7.com/blogs/cloud-computing/5-cloud-advantages-you-didnt-think-about

[5] OpenStack. http://www.openstack.org

[6] CloudScaling. http://www.cloudscaling.com/

[7] Guía de Instalación de OpenStack sobre Ubuntu. (Havana/IceHouse).

http://docs.openstack.org/havana/install-guide/install/apt/content/

[8] Proyecto Innovación Cloud Computing.
http://www.gonzalonazareno.org/cloud/

[9] OpenStack Foundation. http://www.openstack.org/foundation/

[10] Github de OpenStack. https://github.com/openstack/

[11] RedHat. http://www.redhat.com/es/technologies/linux-platforms/openstack-platform

[12] OpenStack Training Guides.

http://docs.openstack.org/training-guides/training-guides.pdf

[13] Wiki de OpenStack. https://wiki.openstack.org/wiki/Release_Naming

[14] OpenStack Operations Guide by Tom Fifield, Anne Gentle, otros autores.

Editorial O'Reilly Media.

Primera edición:Marzo 2014 .

ISBN: 978-1-491-94695-4

[15] TryStack. http://www.trystack.org/

[16] Tor Project. Anonymity Online

https://www.torproject.org/projects/torbrowser.html.en

[17] Image Guide OpenStack

http://docs.openstack.org/image-guide/content/ch_obtaining_images.html

[18] Guía instalación LAMP de DigitalOcean.

https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu

[19] Guía de instalación de Moodle de DigitalOcean.

https://www.digitalocean.com/community/tutorials/how-to-install-moodle-on-a-vps-running-ubuntu-12-04-and-lamp

[20] Guía de instalación de Wordpress de DigitalOcean.

https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-on-ubuntu-12-04

[21] Guía de instalación de OwnCloud de DigitalOcean.

https://www.digitalocean.com/community/tutorials/how-to-setup-owncloud-5-on-ubuntu-12-10

[22] DigitalOcean. https://www.digitalocean.com/

[23] DevStack. http://devstack.org/

[24] Guía de instalación de DevStack en máquina virtual.
http://devstack.org/guides/single-vm.html

[25] Guía de instalación de DevStack en máquina física.
http://devstack.org/guides/single-machine.html

[26] Building on Horizon.
http://docs.openstack.org/developer/horizon/topics/tutorial.html

[27] Customizing Horizon.
http://docs.openstack.org/developer/horizon/topics/customizing.html#button-icons

[28] OpenStack End User Guide.

http://docs.openstack.org/user-guide/content/section_cli_overview.html

[29] Guía de Referencia a los command-client.

http://docs.openstack.org/cli-reference/content/

[30] Automatización con Ansible y Vagrant.

https://github.com/openstack-ansible/openstack-ansible

[31] Ansible. http://www.ansible.com/home

[32] Vagrant. https://www.vagrantup.com/

[33] Guía de instalación de Brian Brophy

https://github.com/BrianBrophy/OpenStack-Installation

[34] Ubuntu Cloud Archive. https://wiki.ubuntu.com/ServerTeam/CloudArchive

[35] Swift All in One.
http://docs.openstack.org/developer/swift/development_saio.html

[36] Foro de consultas técnicas sobre OpenStack. https://ask.openstack.org

[37] OpenStack Marketplace. http://www.openstack.org/marketplace/distros

[38] Curso OpenStack. http://iesgn.github.io/cloud/

[Libros Consultados]

Título: Deploying OpenStack

Autor: Ken Pepple

Editorial: O'Reilly

ISBN: 978-1-449-31105-6

Título: OpenStack Cloud Computing Cook-Book

Autor: Kevin Jackson

Editorial: Packet Publishing Ltd.

ISBN: 978-1-84951-732-4