UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática

Trabajo de Fin de Máster. Escuela de Ingeniería Informática.
Universidad de Las Palmas de Gran Canaria

Diseño e Instalación de una infraestructura de nube privada basada en Openstack.

Autor:

David Cuerva Trujillo

Tutores:

Carmelo Rubén García Rodríguez – Ciencias de la computación e Inteligencia Artificial
Alexis Quesada Arencibia – Ciencias de la computación e Inteligencia Artificial

Las Palmas de Gran Canaria a 01 de agosto de 2017

Trabajo de Fin de Máster de la Escuela de Ingeniería Informática de la Universidad de Las Palmas de Gran Canaria presentado por:

David Cuerva Trujillo

Título del Proyecto: Diseño e Instalación de una infraestructura de nube privada basada en OpenStack

Tutores: Carmelo Rubén García Rodríguez
Alexis Quesada Arencibia

Agradecimientos

A Michaela Wolf y Florentino Pérez.

A los tutores de mi trabajo de fin de Máster Carmelo Rubén García Rodríguez y Alexis Quesada Arencibia por el apoyo ofrecido en este proyecto.

A todo el profesorado que impartió el Experto en Virtualización y Computación en la nube ya que debido a sus enseñanzas he decidido emprender este proyecto de fin de Máster.

Gracias.
David Cuerva Trujillo

Tabla de Contenido

1. INTRODUCCIÓN

El Trabajo de Fin de Máster(TFM) realizado parte del Trabajo de Fin de Grado(TFG) donde se inició una aventura con OpenStack y la virtualización.

En el TFG se tuvo la posibilidad, mediante recursos virtuales, desplegar una nube privada OpenStack que proveía máquinas virtuales a usuarios.

La diferencia principal con el TFG es que este TFM se configura sobre una estructura física, concretamente sobre los recursos provistos por el Centro de Cálculo del Departamento de informática y sistemas por lo que, en vez de usar la estructura principal de dicha configuración de forma virtual, la tenemos de forma física.

Las principales complicaciones que se han tenido con respecto al TFG han sido las siguientes:

- La gestión de redes (Direcciones IP, Mascaras de Red y puertas de enlaces) . De forma virtual la gestión es casi inexistente, de forma física es trivial el uso y entendimiento de las mismas.

- Salidas a Internet. En el TFG al tener toda la infraestructura de forma virtual todas nuestras máquinas tienen interfaces de red teóricamente ilimitadas por lo tanto una de ellas es configurada automáticamente por el sistema como pasarela entre el ordenador anfitrión y la máquina virtual, por lo tanto, todas las máquinas virtuales tienen Internet por defecto. En el TFM tenemos un número muy limitado de Interfaces de Red (específicamente 2) por lo que no todas las máquinas fueron posible de conectar a Internet.

- Interfaces de red. De forma virtual el nº de interfaces de red es ilimitado. De forma física se está limitado a dos unidades en nuestro caso personal. Por ejemplo, en uno de nuestros servidores necesitábamos conectar 3 cables de red y teníamos dos interfaces.

- Limitación a 1 red. En el TFG se tenía 3 redes. Una para salir a internet. Una para gestión y otra para que las MV compartieran datos. De forma física debido a las limitaciones comentadas anteriormente se tuvo que gestionar todo en 1 sola red.

- Accesibilidad. En el TFG la accesibilidad a las máquinas era inmediata pues al ser virtualizado todo estaba accesible 24horas. En este caso el proyecto se somete a los horarios del centro de cálculo.

- Centos 7. En el TFG usamos CentOS 6.7 mientras que en este caso se ha usado CentOS 7.0 en el cual ha cambiado flagrantemente la filosofía del mismo (los archivos conocidos no están en el mismo sitio, las órdenes habituales han cambiado, etc …)

- En primer lugar, se intentó usar la versión de OpenStack Newton pero después debido a recurrentes problemas indocumentados y una vez consultado en el Foro de OpenStack, se decidió bajar a la versión MITAKA que es una versión estable.

Este TFM va a virtualizar recursos, tanto de cómputo como de almacenamiento para laboratorios docentes en los que se imparten asignaturas que requieren que el alumno posea privilegios de administrador y en las que las actividades de aprendizaje requieren modificar las configuraciones o alterar componentes software del sistema.

Aunque existen alternativas de pago como "**Amazon Web Services**" o "**VMWare Cloud**", este proyecto se ha desarrollado sin ningún coste en licencias de uso.

Actualmente, los laboratorios disponen de dispositivos con diferentes configuraciones, pero en este trabajo se ha tomado como tipo de laboratorio aquellos que necesitan configuraciones especiales para poder realizar sus prácticas y donde a cada usuario se le asigna una partición del disco duro adaptándola a sus necesidades.

El objetivo de este trabajo es tener una solución basada en software libre que proporcione recursos de cómputo, red y almacenamiento a todos los usuarios del laboratorio.

Por lo tanto, en este TFM se despliega una instalación básica de nube privada que proporciona máquinas virtuales para el alumnado y de esa manera éstos puedan hacer modificaciones al software del sistema logrando completar los objetivos de cada asignatura y sin riesgo de corromper otras máquinas.

Para proveer de una mayor comodidad, al administrador de la nube privada, se le provee de una interfaz sencilla mediante la cual puede hacer las siguientes acciones:

- Crear, eliminar, arrancar y parar máquinas virtuales.

- Administrar volúmenes lógicos para la instalación de máquinas virtuales.

- Crear plantillas de máquinas virtuales (*flavours*).

- Administrar cuentas de usuario.

2. ESTRUCTURA DE LA MEMORIA

Este trabajo se estructura en 12 apartados:

Se empieza con una breve introducción acerca del trabajo desarrollado, a continuación, se define la estructura de este trabajo, a continuación, exponemos a la planificación temporal de proyecto, en cuarto lugar, se da unas definiciones previas que sitúan al lector en el contexto del trabajo. Seguidamente se presenta un estudio sobre las herramientas existentes en el mercado.

El siguiente bloque temático de esta memoria está formado por los apartados seix, siete y ocho. En el punto seis se presentan los objetivos de este trabajo. En el séptimo apartado, se expone la justificación de las competencias específicas cubiertas. En el punto número ocho se enuncian las aportaciones conseguidas con el trabajo desarrollado.

La tecnología utilizada para desarrollar este proyecto se explica en el siguiente bloque temático, puntos nueve y diez. En noveno lugar se explica la herramienta utilizada en este proyecto en el apartado siguiente explicaremos en detalle cómo hemos desarrollado este proyecto.

Se enuncian las normativas y leyes a cumplir con respecto a la legislación vigente y la bibliografía del trabajo se incluye en el punto 11 de este trabajo de Fin de Master

Por último, los resultados, conclusiones y trabajos futuros se presentan en el punto 12

En el punto 13 se exponen las fuentes de información consultadas para la realización de este Trabajo.

3. PLANIFICACIÓN TEMPORAL

En este apartado se presenta la planificación temporal del proyecto y a continuación explicaremos las desviaciones que se han producido durante su ejecución.

La planificación inicial del trabajo fue la siguiente:

Análisis: 30 horas.
Toma de requisitos del proyecto.

En esta fase hemos realizado **finalmente** las siguientes tareas:

- Reunión con los tutores para estudiar la viabilidad y los requisitos del proyecto.
- Reunión con los responsables del Centro de Cálculo del edificio de Informática y Sistemas, Tutor del Trabajo y Autor del mismo con el objetivo de conocer los recursos que el centro pondrá a disposición para la correcta realización del trabajo.
- Elaboración de la documentación oficial a suministrar al centro.
- Análisis de la documentación realizada en el Trabajo de Fin de Grado
- Reciclaje de los aspectos básicos de virtualización, redes, etc…
- Estudio de la página web de *Openstack*(www.openstack.org) para tomar la decisión de que versión utilizar.

Para todo ello el tiempo invertido han sido unas 2 semanas de trabajo que estipulamos en unas 40 Horas

Diseño: 30 horas.
Re-estudio de los componentes de *f* – Diseño de la solución.

En esta fase hemos realizado **finalmente** las siguientes tareas:

- Estudio del componente *Nova-Compute* (4 horas).
- Estudio del componente *Neutron-Networking* (5 horas).
- Estudio del componente *Cinder-BlockStorage* (3 horas).
- Estudio del componente *KeyStone-Identity* (8 horas).

- Estudio del componente *Glance-ImageService* (3 horas).
- Estudio del componente *Head-Orchestration* (2 horas).
- Estudio del componente *Ceilometer-Stats* (2 horas).
- Estudio del componente *Horizon-Dashboard* (7 horas).
- Estudio del resto de componentes de forma general (2 horas).

Por tanto, el tiempo utilizado aproximadamente para completar esta fase del trabajo es de 36 Horas.

Instalación e Implementación: 150 horas.
Instalación y prueba de cada componente, creación de todos los componentes a utilizar.

En esta fase, podemos decir que hemos realizado las siguientes tareas:

- Instalación de toda la infraestructura hardware (servidores, redes, sistemas operativos).
- Configuración básica de los servidores.
- Pruebas sobre la red de la infraestructura.
- Instalación de la plataforma de virtualización *KVM*.
- Instalación y configuración de los componentes necesarios para la realización del trabajo con la plataforma *OpenStack*.
- Configuración del LVM exportando desde un servidor SAN.

Esta ha sido la fase en la que más tiempo hemos tenido que utilizar y aunque no se puede estimar un número de horas exacto si se puede decir que hemos estado alrededor de 75 días a una media de 2horas/día, por lo que podemos concluir que la estimación inicial fue acertada.

Pruebas: 60 horas.
Prueba sobre todos los componentes del sistema.

En esta fase, podemos decir que hemos realizado las siguientes tareas:

- Prueba de cada componente individualmente.

- Prueba de la plataforma de forma general.
- *Troubleshooting* de cada componente con cada error.

La mayoría del tiempo invertido en esta fase ha sido en depuración de errores.

Se ha cumplido con la expectativa de 60 horas.

Documentación: 30Horas
Documentación de todo el trabajo.

Para la documentación del trabajo hemos invertido mucho más de 30 horas, pero no podemos precisar exactamente el número. Estimamos unas 40 horas.

Por lo tanto, el balance final de tiempo utilizado en este trabajo se estima en 326 horas invertidas.

4. CONCEPTOS Y DEFINICIONES PREVIAS

4.1. ¿Qué es la virtualización?

Según la definición de TURBAN[11], la virtualización es crear, a través de software, una versión virtual sobre recursos tecnológicos.

También es definida como la abstracción de los recursos (CPU, Memoria, Dispositivos Periféricos y Conexiones de Red) de una computadora.

Consolidación

Debido a que los servidores físicos suelen funcionar usando en torno al 10-15% de su capacidad se genera un gran desperdicio de recursos.

Consolidar servidores es la capacidad de albergar dentro de un servidor físico un número finito de máquinas virtuales consiguiendo que el rendimiento de un servidor se incremente. Consiguiendo de esta forma un ahorro de costes de entorno a un 50-70%.

Como se puede ver en la figura 1 en este caso existe una consolidación de 10:1 debido a que conseguimos virtualizar 10 servidores virtualizados en 1 físico.

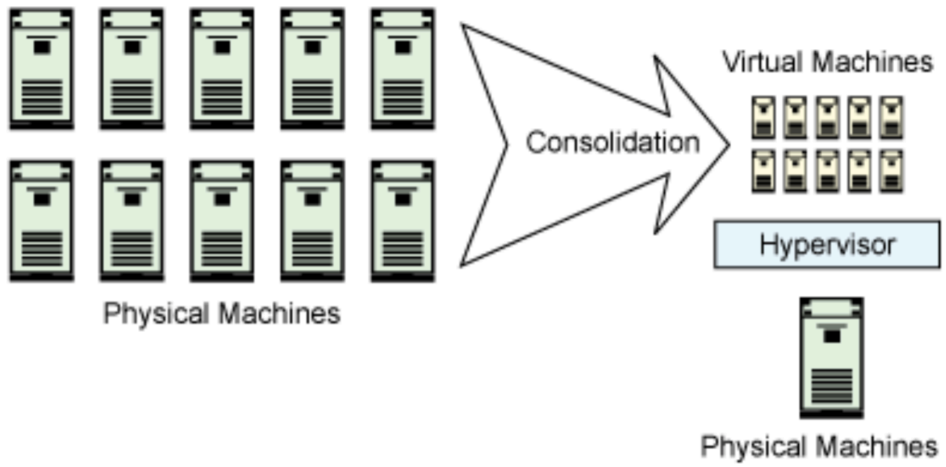


Figura 1[*1] - Consolidación de máquinas virtuales

Fuente: http://www.aitinc.net/products-services/network-support-maintenance/virtualization

¿Qué permite hacer la virtualización?

- ✓ Aprovechar mejor los recursos del sistema.
- ✓ Backups del sistema de forma rápida al estar toda la información en ficheros.
- ✓ Incorporar rápidamente nuevos recursos a los servidores virtualizados.
- ✓ Aislar fallos al tener cada servicio en una máquina virtual.
- ✓ Reducir el mantenimiento al centrar la administración.
- ✓ Reducir el consumo energético.
- ✓ Reducción de los costes de espacio.

Inconvenientes

- ✓ Rendimiento inferior de la máquina virtualizada con respecto a la máquina anfitrióna.
- ✓ Hardware que no sea posible ser virtualizado no se podrá usar.
- ✓ Rendimiento gráfico extremadamente limitado.

4.2. ¿Qué es la "Computación en la nube"?

Según el **NIST (NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY)**

"Computación en la nube es un modelo para permitir acceso vía red, ubicuo, cómodo y bajo demanda a un conjunto compartido de recursos de computación configurables (incluyendo redes, servidores, almacenamiento, aplicaciones y servicios) que pueden ser rápidamente aprovisionados y liberados con un esfuerzo de manejo mínimo o con una mínima interacción con el proveedor del servicio"

La página **WWW.COMPUTACIONENLANUBE.ORG** lo define como:
"un sistema informático basado en Internet y centros de datos remotos para gestionar servicios de información y aplicaciones"

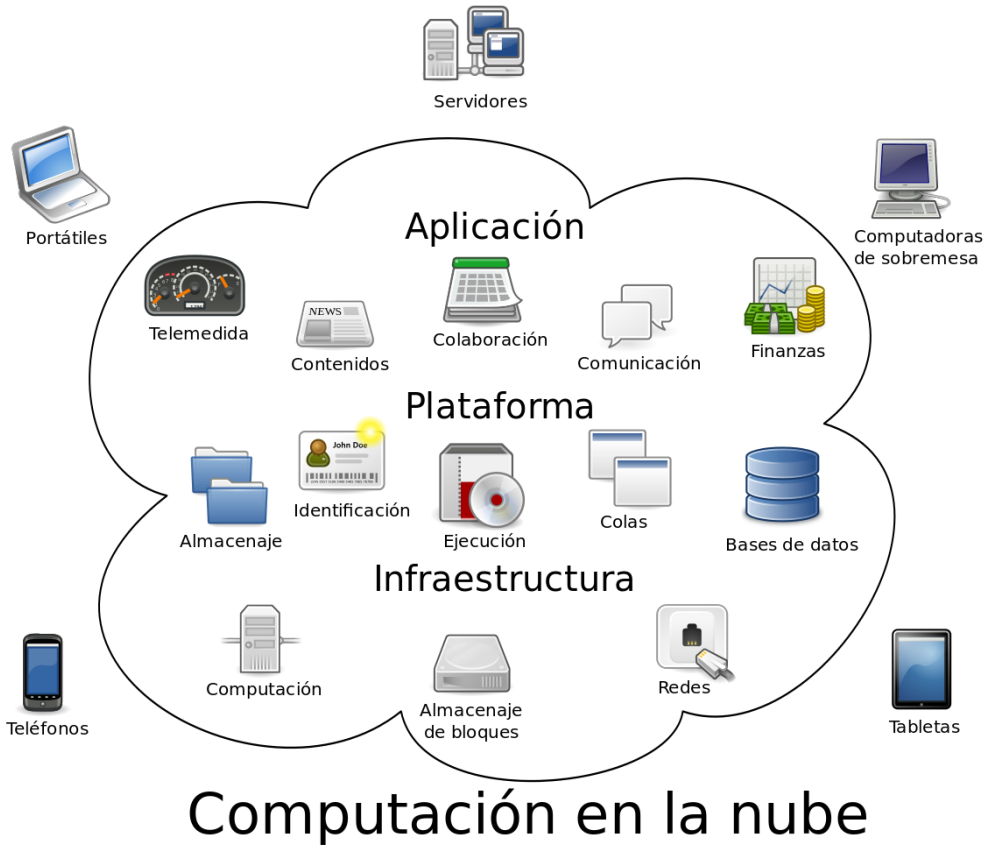


Figure 2[*2] - Modelo de computación en la nube

Fuente: https://es.wikipedia.org/wiki/Computaci%C3%B3n_en_la_nube#/media/File:Cloud_computing-es.svg

Beneficios de la computación en la nube.

- Servicios en Internet, por lo que se pueden prestar a distintos niveles: local, nacional e internacional.
- Instalación y mantenimiento externos que son responsabilidad del proveedor.
- Coste inferior al iniciar un negocio.
- Rapidez y fiabilidad en la implantación de un sistema mayor.
- Nivel de integración mayor.

Desventajas de la computación en la nube.

- Servicio Centralizado, dependencia sobre el proveedor.
- Dependencia sobre una conexión a internet.
- El correcto funcionamiento de todo el sistema requiere que el proveedor de servicios *cloud sea* solvente.
- La seguridad en general es menor en este tipo de servicios.
- La información se puede ver expuesta a terceros si la nube es pública y no tiene las medidas de seguridad adecuadas.

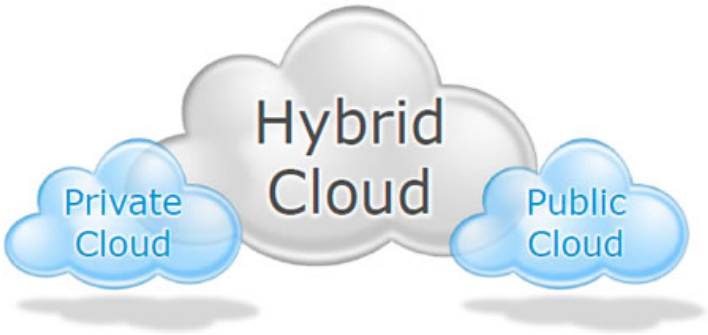


Figura 1 - Tipos de nube

Fuente: https://diarioti.com/computacion-en-nube-a-la-medida-de-la-empresa/69719

Como se puede observar en la figura3 existen 3 tipos de nubes que las describimos a continuación:

Nubes privadas:

Las nubes privadas son las nubes de mayor frecuencia, este tipo de nubes se encuentran dentro de las instalaciones de las empresas y normalmente son de uso exclusivo.

En general, en las empresas se usan para aprovisionar recursos de infraestructura ya sea de cómputo o de almacenamiento de manera rápida, esto no quiere decir que algunas empresas alojen una aplicación propia y la sirvan como un servicio.

Es una implementación recomendable para aquellas empresas que necesiten una alta protección de sus datos. En este tipo de nube la empresa es quien decide quien utiliza la nube y sus servicios.

Nubes públicas:

Las nubes públicas son el estándar de la computación, todos los servicios que se ofrecen al cliente de forma gratuita o mediante licencias de pago se encuentran alojados en servidores externos.

Nubes híbridas

Las nubes híbridas combinan recursos tanto de nube privada como de nube pública. Esto permite a una empresa mantener el control de acceso y aprovechar la nube pública únicamente cuando sea necesario.

Capas de la "Computación en la nube"

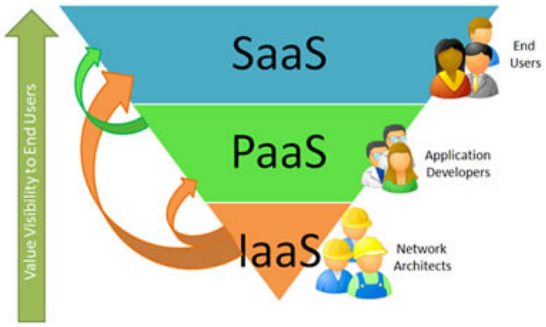


Figura 2 - Capas de la computación en la nube

Fuente: http://www.itsteziutlan.edu.mx/site2010/images/revistatec/cultura-informatica/2011/09/computacion-en-la-nube.jpg

Software como servicio (SaSS) – Software bajo demanda

Es lo más habitual entre los usuarios, la mayoría de las aplicaciones actualmente se encuentran en la nube donde cada usuario puede acceder a ellas a través de un explorador web.
El software como servicio (SaSS) es un modelo de software donde todo el soporte está almacenado en un servidor al cual se accede a través de internet y donde la compañía suministradora del producto se encarga del buen funcionamiento de este.

Gmail by Google BETA

Figure 3 - Logo de gmail

Dropbox

Figure 4 - Logo de dropbox

Microsoft® Office 365

Figure 5 - Microsoft Office 365

Plataforma como servicio (PaSS)

Es la capa de en medio, esta capa provee todo lo necesario para que los programadores puedan hacer prototipos, analizar, desarrollar, testear y documentar las aplicaciones para posteriormente poder ser utilizadas.

Figure 6 - Logo de heroku

Figure 7 - Logo de windows azure

Infraestructura como servicio (IaaS)

Es la capa base de la computación en la nube, esta capa provisiona los recursos necesarios (cómputo, red y almacenamiento) para que las capas superiores puedan utilizarlo y así poder crear aplicaciones al servicio del usuario final.

Figure 8 - Logo Amazon Webservices

Figure 9 - Logo rackspace

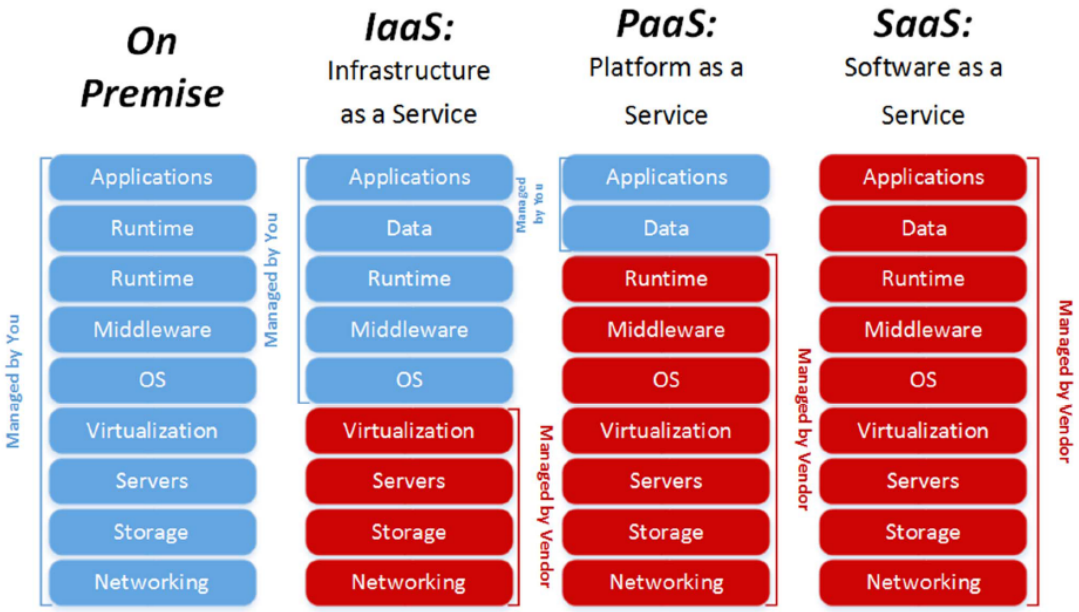


Figure 10 - Gestión de responsabilidades de usuario y del suministrador

Fuente: http://blogs.salleurl.edu/dc-wall/2014/02/26/de-la-virtualizacion-al-cloud-computing/

En la figura 12 se indica en azul lo que es suministrado por un usuario que maneja dicho servicio y en rojo la gestión que tiene que llevar a cabo sobre dichos componentes.

Por ejemplo, si nosotros contratamos Infraestructura como Servicio(PaaS) lo relativo a virtualización, servidores, almacenamiento y red lo deberemos gestionar nosotros.

4.3. ¿Qué es un Centro de Datos definido por Software?

Un centro de datos definido por software es un centro de datos informático en el que todos los elementos de la infraestructura (cómputo, redes, almacenamiento) se virtualizan entregándose como un servicio.

5. Situación Actual

En la actualidad, la mayoría de empresas se están mudando a lo que se denomina “la nube” motivadas principalmente por una razón económica o técnica.

Antiguamente cada empresa tenía lo que se denominaba “Departamento Informático”, este departamento se encargaba de la puesta a punto, la actualización de hardware/software y del mantenimiento periódico de todos los ordenadores.

En la actualidad muchas de las empresas están externalizando este servicio lo que implica que se está prescindiendo de equipos de mantenimiento informático.

Esto conlleva a la empresa a no tener que preocuparse de instalar nuevo software o de tener siempre a punto todo su patrimonio tecnológico.

A continuación, procederemos a analizar algunos de los mejores sistemas de infraestructura en la nube que existen; En primer lugar, se analiza al primer innovador en este campo ***Amazon*****,** en segundo lugar, analizaremos el producto más popular y más caro de Infraestructura como Servicio denominado ***VMWare***.

Por último, vamos a analizar 4 infraestructuras como servicio OpenSource que se llaman ***OpenStack***, ***CloudStack*****,** ***OpenNebula*** y ***Eucalyptus***

5.1. Amazon Web Services

Figure 11 - Logotipo de Amazon Webservices

Según la definición del propio AMAZON[6]:

Amazon Elastic Compute Cloud (Amazon EC2) es un servicio web que proporciona capacidad informática con tamaño modificable en la nube.

Está diseñado para facilitar a los desarrolladores recursos informáticos escalables basados en web con una sencilla interfaz web.

Amazon EC2 reduce el tiempo necesario para obtener y arrancar nuevas instancias de servidor en minutos, lo que permite escalar rápidamente la capacidad, ya sea aumentándola o reduciéndola, según cambien sus necesidades.

Amazon EC2 cambia el modelo económico de la informática, al permitir pagar sólo por la capacidad que utiliza realmente.

Amazon EC2 proporciona a los desarrolladores las herramientas necesarias para crear aplicaciones resistentes a errores y para aislarse de los casos de error más comunes.

Según AMAZON[6] las ventajas que ofrece son las siguientes:

- **Informática a escala web elástica**: Amazon EC2 permite aumentar o disminuir la capacidad en minutos controlado desde una interfaz sencilla.
- **Totalmente controlado**: Tener control total sobre las instancias ya que tenemos acceso de usuario administrador a todas ellas, y podemos interactuar con ellas como con cualquier otra máquina.
- **Servicios de alojamiento en la nube flexibles:** Tenemos la posibilidad de elegir entre varios tipos de instancia, sistemas operativos y paquetes de software.

- **Diseñado para utilizarse con otros servicios de AWS:** Amazon EC2 trabaja con *Amazon Simple Storage Service* (Amazon S3), Amazon Relational Database Service (Amazon RDS), *Amazon SimpleDB* y Amazon Simple Queue Service (Amazon SQS) para proporcionar una solución completa de informática, procesamiento de consultas y almacenamiento para una gran variedad de aplicaciones.

- **Fiabilidad:** Amazon EC2 ofrece un entorno muy fiable en el que las instancias de sustitución se pueden enviar con rapidez y anticipación. El servicio se ejecuta en los centros de datos y la infraestructura de red acreditados de Amazon.

- **Seguridad:** Amazon EC2 funciona junto con *AmazonVPC* para proporcionar una funcionalidad de red sólida y segura para sus recursos informáticos.

    - Sus instancias informáticas se ubican en una nube privada virtual (VPC) con el rango de IP que se especifique. Se deciden las instancias que se expondran en Internet y las que permanecen de forma privadas.

    - Los grupos de seguridad y las ACL de red permiten controlar el acceso entrante / saliente a la red a y desde sus instancias.

    - Se puede conectar infraestructura de TI a los recursos en VPC mediante conexiones VPN IPsec cifradas estándar del sector.

    - Aprovisionar recursos de EC2 como instancias dedicadas. Las instancias dedicadas son instancias de Amazon EC2 que se ejecutan en hardware dedicado a un único cliente para ofrecer más aislamiento.

- **Asequibilidad:** Amazon EC2 permite disfrutar de las ventajas financieras de la escala de Amazon.

Fuente: [*6] http://aws.amazon.com/es/ec2/

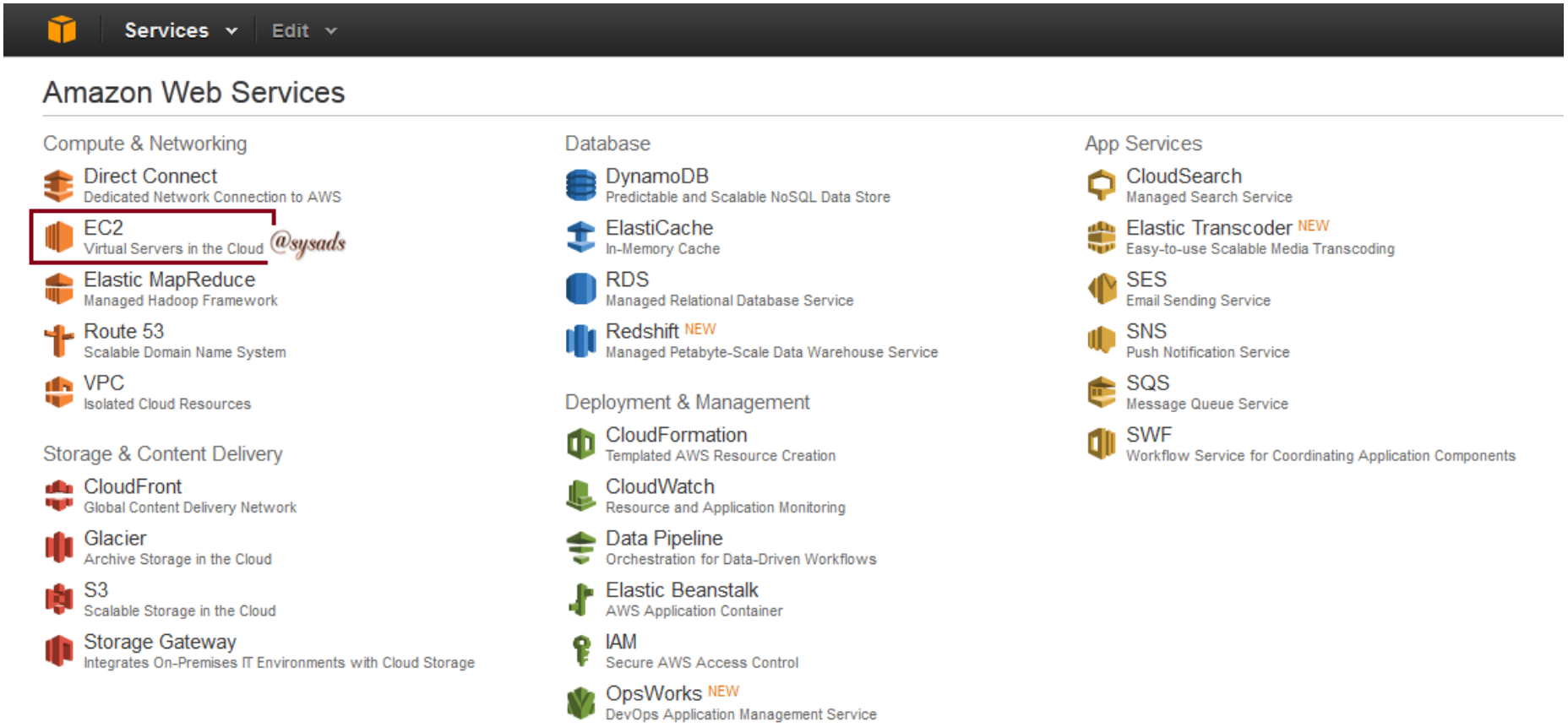


Figure 12 - Amazon Web Services Control Panel – Fuente: Amazon

5.2. VMware

Figure 13- Imagen corporativa VMWare

VMWare[10] (VM es el acrónimo Virtual Machine) es una compañía suministradora de servicios de virtualización por software donde las máquinas virtuales proporcionan un ambiente de ejecución similar, a todos los efectos, a un computador físico.

Virtualización de servidores

Debido a que la arquitectura de los **servidores actuales no permite la ejecución de más de un sistema operativo simultáneamente**, la virtualización de servidores desbloquea la arquitectura tradicional uno a uno de los servidores x86. Para hacerlo, el sistema operativo y las aplicaciones se **abstraen del hardware físico**, con lo que se consigue un entorno de servidor más rentable, ágil y sencillo.

Mediante la virtualización de servidores, en un único servidor se pueden ejecutar varios sistemas operativos como máquinas virtuales, todos con acceso a los recursos informáticos del servidor subyacente.

La mayoría de los servidores operan sólo al 15 % de su capacidad. Esto, además de ser muy poco eficiente, da lugar a la complejidad y proliferación de servidores con la virtualización de servidores y con el software de VMWare se logra un aumento hasta el 80% de la utilización de los recursos de servidor, lo que conlleva a las empresas a un ahorro de hasta el 50 % en costes operativos y de capital donde cada servidor puede tener una proporción de consolidación de servidores de 10:1 como mínimo.

Virtualización de redes.

La virtualización de redes es la reproducción completa en software de una red física.

Las redes virtuales cuentan con las mismas características y garantías que una red física.

Sin embargo, ofrecen los beneficios operativos y la independencia del hardware de la virtualización: aprovisionamiento rápido, implementación sin interrupciones, mantenimiento automatizado y compatibilidad con aplicaciones nuevas y heredadas.

Virtualización de escritorios.

Implementar escritorios como servicio gestionado ofrece la posibilidad de responder con más rapidez a los cambios y a las oportunidades del mercado.

Al ofrecer las aplicaciones y los escritorios de forma virtualizada se consigue reducir costes y a la vez aumentar el servicio a las sucursales.

Las soluciones de escritorio de VMware son escalables, coherentes, totalmente seguras y muy disponibles para garantizar un tiempo máximo de actividad y productividad. Esto permite:

- Optimizar la implementación y la gestión proporcionando los escritorios como servicio.
- Proporcionar acceso remoto seguro a los trabajadores a distancia y a los empleados eventuales sin sacrificar el rendimiento.

Virtualización de aplicaciones

Debido a que cada vez más, las organizaciones están virtualizando muchas de sus plataformas y aplicaciones como bases de datos, correo electrónico, middleware, inteligencia empresarial, etc.

Con objeto de mantener los niveles de calidad de servicio y los acuerdos que garantizan la calidad para estas aplicaciones empresariales en entornos virtuales, las organizaciones de tecnologías de la información se centran tanto en los componentes de virtualización del proyecto como en la gestión y supervisión de las aplicaciones empresariales virtualizadas.

Además, deben mantener las directrices corporativas sobre continuidad del negocio y recuperación ante desastres, las aplicaciones virtualizadas funcionan mejor además proporcionando alta disponibilidad, recuperación ante desastres, rapidez, agilidad y preparación para la nube.

Con la solución de virtualización de aplicaciones de primer nivel de VMware que tiene como base *VMware vCloud Suite* se puede mejorar la calidad de los servicios de tecnologías de la información que proporciona, a la vez que simplifica la infraestructura, maximiza la eficiencia y elimina el exceso de aprovisionamiento.

Virtualización de almacenamiento

La virtualización del almacenamiento forma parte de la capa de almacenamiento definido por software, esta debe proporcionar mejoras de rendimiento y eficiencia sin necesidad de comprar más hardware de almacenamiento.

Debe ser posible un aprovisionamiento rápido, de forma que el almacenamiento sea de alto rendimiento y que se aproveche eficazmente el espacio además de que se pueda crear con la misma rapidez con la que ya se puede crear una máquina virtual.

Debe ofrecer un modelo de gestión del almacenamiento centrado en la máquina virtual que sea intuitivo para los administradores virtuales, quienes se van a encargar de más tareas de gestión del almacenamiento en los entornos virtuales.

Por último, debe integrarse con la plataforma del hipervisor para aprovechar flujos de trabajo nativos ya conocidos.

La virtualización del almacenamiento de VMware es una combinación de prestaciones que proporciona una capa de abstracción que permite abordar, gestionar y optimizar los recursos de almacenamiento físicos en una implementación de virtualización.

La tecnología de virtualización del almacenamiento proporciona una manera esencialmente mejor de gestionar los recursos de almacenamiento para la infraestructura virtual.

Proporciona a su organización la capacidad de:

- Mejorar significativamente la utilización y flexibilidad de los recursos de almacenamiento.
- Simplificar la aplicación de parches al sistema operativo y los requisitos de controladores, con independencia de la topología de almacenamiento.
- Aumentar el tiempo de actividad de las aplicaciones y simplificar las operaciones cotidianas.
- Aprovechar y complementar la infraestructura de almacenamiento existente.

Fuente[*7]: http://www.vmware.com/es/virtualization

5.2.1. VMware vSphere

Diseñado para organizaciones que desean optimizar los activos de infraestructura tecnológica existentes y demorar las costosas expansiones del centro de datos, *VMware® vSphere® Standard Edition* proporciona una solución de consolidación básica de las aplicaciones, a fin de reducir drásticamente los costes de hardware, además de acelerar la implementación de las aplicaciones sin tener que programar ningún tiempo de inactividad.

VMware vSphere es la plataforma de virtualización líder del sector para construir infraestructuras en la nube.

Permite a los usuarios ejecutar aplicaciones críticas para el negocio con confianza y responder con mayor rapidez a las necesidades empresariales. **vSphere** acelera el cambio hacia la computación en la nube para los centros de datos.

Ventajas

- Se obtiene una mayor eficiencia gracias a la automatización. pudiendo conseguir índices de consolidación de 10:1 y una mejora de hardware del 15% hasta un 80% o más sin haber sacrificado el rendimiento.
- Disminuir los gastos de propiedad en hasta un 70% y los costes operativos en un 30%, por cada aplicación que se ejecute en vSphere.
- Escalabilidad y disponibilidad: se puede responder rápidamente a las necesidades empresariales en constante cambio sin sacrificar la seguridad ni el Control.
- Se tiene libertad de elección al usar una plataforma común basada en estándares.

Servicios de Infraestructura

- La arquitectura de hipervisor *VMware vSphere ESXi™* proporciona una sólida capa de virtualización probada para entornos de producción y de alto rendimiento.

- Permite que varias máquinas virtuales compartan recursos de hardware con un rendimiento que puede igualar al nativo.

- *vSphere Virtual Symmetric Multiprocessing* permite utilizar máquinas virtuales ultrapotentes con hasta ocho CPU virtuales.

- El hardware virtual de *VMware* es capaz de admitir 1 TB de RAM y gran variedad de hardware de próxima generación como los procesadores de gráficos 3D o los dispositivos USB 3.0.

- *vSphere Storage Virtual Machine File System* permite que las máquinas virtuales puedan acceder a los dispositivos de almacenamiento compartido (Fiber Channel, iSCSI, etc.).

- *vSphere Storage API* proporciona integración con las soluciones de protección de datos de terceros compatibles.

- *vSphere Storage Thin Provisioning* proporciona asignación dinámica de la capacidad de almacenamiento compartido. De este modo, el departamento de tecnologías de la información puede implementar una estrategia de almacenamiento por niveles y reducir al mismo tiempo el gasto en almacenamiento hasta un 50%.

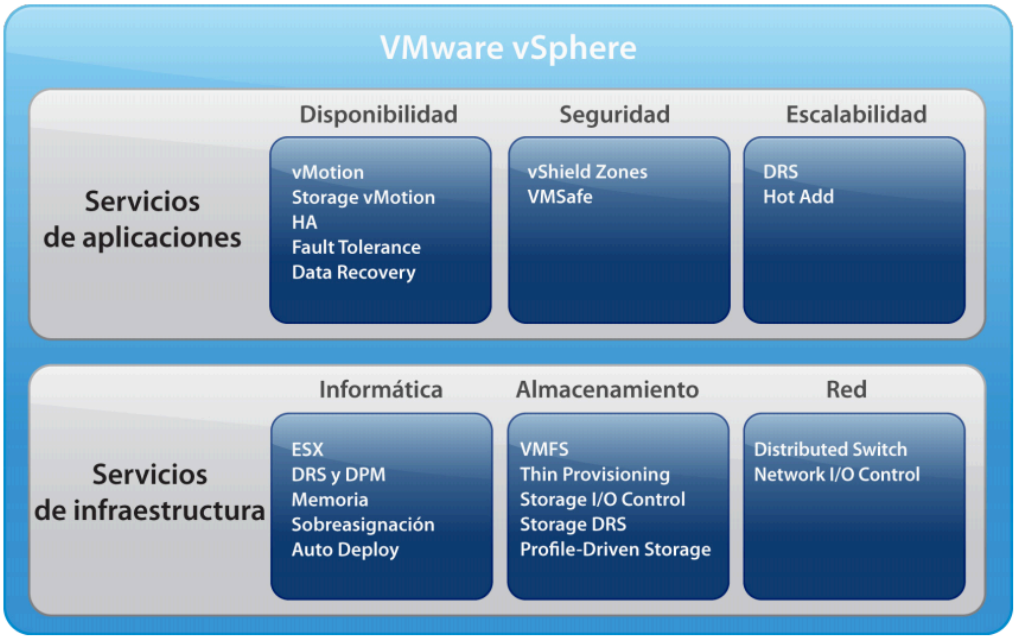
Servicios de aplicaciones

- VMware vSphere vMotion permite la migración en caliente de máquinas virtuales entre servidores sin interrupción alguna para los usuarios ni pérdidas de servicio. De esta forma se elimina la necesidad de programar tiempo de inactividad de las aplicaciones para el mantenimiento de servidores.

- VMware High Availability (HA) es la solución automatizada más rentable para el reinicio de todas las aplicaciones en cuestión de minutos en caso de un fallo de hardware o del sistema operativo.

- La conexión en caliente permite conectar o desconectar dispositivos virtuales de almacenamiento o de red de las máquinas virtuales sin interrupciones ni tiempo de inactividad.

- La ampliación en caliente de discos virtuales permite añadir almacenamiento virtual a las máquinas virtuales sin interrupciones ni tiempo de inactividad.



Fuente[*7]:https://www.vmware.com/files/es/pdf/VMware-vSphere-Standard-Edition-Datasheet.pdf

5.2.2. VMware View

VMWare View, utiliza **VSphere** como plataforma para virtualizar las máquinas virtuales que ofrece a sus usuarios.

Estas máquinas son gestionadas por un intermediario llamado **View Manager**, que se encarga de autenticar a los usuarios y asignar la máquina virtual que le corresponde a cada usuario.

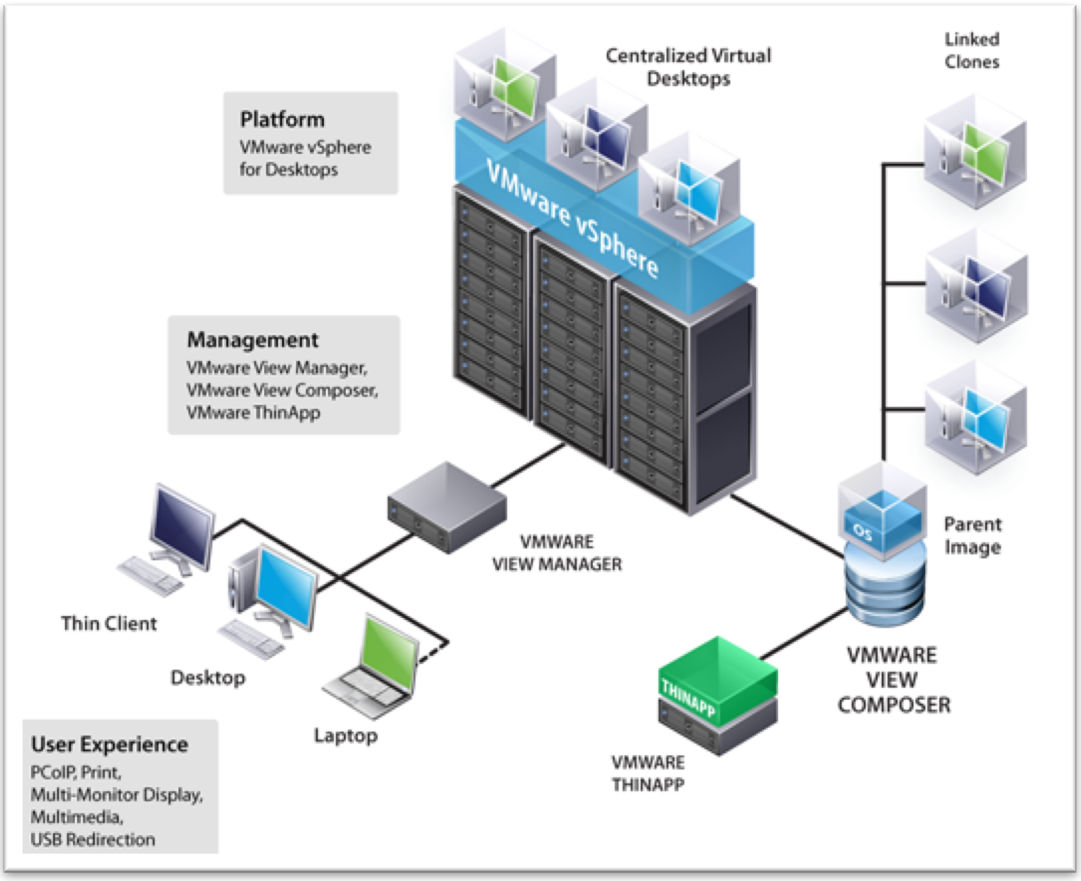


Figura 14 - Esquema de VM View Manager

Fuente[*7]: **https://www.vmware.com/files/es/pdf/VMware-vSphere-Standard-Edition-Datasheet.pdf**

5.3. OpenStack

OpenStack[1] es un "sistema operativo en la nube" que controla gran número de máquinas de cómputo, almacenamiento y recursos de red de un centro de datos, todo ello gestionado por software a través de un panel con interfaz web o mediante órdenes, que permite a los administradores controlar los recursos asignados a sus usuarios. Además, OpenStack es un proyecto de código abierto para centros de datos que dan servicios IaaS.

Persigue una implementación sencilla que sea escalable y rica en funcionalidades.

Este a su vez está dividido en muchos paquetes de software donde todos se comunican entre ellos, por lo tanto, OpenStack no es un paquete de software sino un integrador de paquetes.

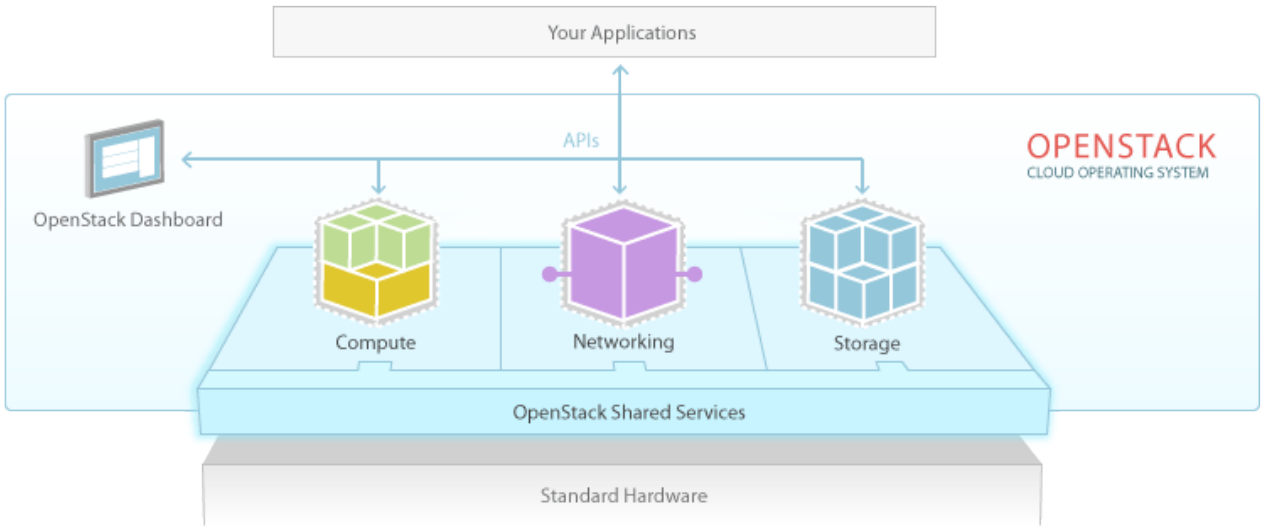


Figure 15 – Sistema distribuido de OpenStack

Fuente: www.openstack.org

NOTA:
RDO (No es una abreviación o acrónimo) es una comunidad de personas que usan y desarrollan Openstack a través de Red Hat Enterprise Linux, Fedora y distribuciones derivadas de esta como CentOS, SciLinux y muchas más.

Principios de OpenStack

- Es OpenSource y sus repositorios de código fuente son públicos.
- Proceso de diseño abierto, es decir, una comunidad externa puede proponer un añadido.
- Todos los procesos de desarrollo deben estar documentados y ser transparentes.
- Orientado para adoptar estándares abiertos.
- Diseño modular que permite flexibilidad mediante el uso de APIs.

Componentes o Servicios

- ✓ Horizon – Panel de gestión vía web.
- ✓ Nova – Gestor de máquinas virtuales.
- ✓ Cinder – Almacenamiento de volúmenes.
- ✓ Keystone – Autenticación y autorización.
- ✓ Swift – Almacenamiento de objetos.
- ✓ Glance – Gestión de imágenes para las instancias.
- ✓ Neutron – Gestión de redes virtuales.

5.4. Apache CloudStack

Apache CloudStack es un proyecto de alto nivel de la Fundación de Software Apache. El proyecto desarrolla software de código abierto y nubes con infraestructura como servicio (IaaS).

Este proyecto provee las herramientas necesarias para entregar nubes públicas de manera fiable y escalable.

Entre sus características se encuentran las siguientes:

- Trabaja con anfitriones bajo tecnología Xen, KVM, Hyper-V o Vmware ESXi con Vsphere.
- Provee una interfaz sencilla para el manejo de toda la operativa de un IaaS por medio de una interfaz visual o de comandos.
- Tiene una API Nativa.
- Compatibilidad con Amazon S3 y EC2.
- Maneja almacenamiento para instancias basado en los hipervisores además de plantillas, capturas de instancias e Imágenes ISO como almacenamiento secundario.
- Servicios de red desde la capa 2 (Capa de datos) hasta la capa 7 (Capa de Servicios) dando servicios como DHCP, NAT, cortafuegos y VPNs.
- Está separada en 3 partes: Red, cómputo y almacenamiento.
- Soporta multi-tenencia, es decir, un proyecto puede ser accedido por varios usuarios concurrentemente.

Fuente[*26]: https://cloudstack.apache.org/about.html

5.5. OpenNebula

Este grupo con sede en la Universidad Complutense de Madrid es un equipo de investigación enfocado en la computación distribuida, virtualización y plataformas de IaaS. En el ámbito de las comunidades de software libre existentes que han ido sumándose al proyecto se encuentran actualmente: Ubuntu, Debian, OpenSuse y CentOS. Las mismas han ido integrando en sus repositorios los paquetes del sistema.

El sistema OpenNebula al ser completamente abierto permite una completa inter-operatividad con los componentes de las infraestructuras actuales existentes. Esto previene posibles presiones de proveedores terceros existentes en la industria actual de la nube.

OpenNebula no depende exclusivamente de ningún tipo concreto de hipervisor (gestores de virtualización), igualmente no requiere de unos determinados recursos pudiendo adaptarse a infraestructuras ya existentes del volumen y capacidades que se deseen.

Dentro de los hipervisores soportados dentro de la estructura de OpenNebula podremos identificar soluciones libres como Xen, KVM, QEMU/KVM.

Igualmente soporta libvirt como capa de abstracción entre KVM/XEN. En el apartado de hipervisores no libres soporta VMWARE en todos sus productos (VMWare ESXI, ESX, Server)
OpenNebula proporciona a los diferentes usuarios de la plataforma las siguientes herramientas:

- Gestión de recursos flexible y despliegue de máquinas pre-configuradas.
- Gestión de usuarios (uso, facturación, provisionamiento..).
- Gestión de perfiles de seguridad.
- Gestión de redes (subsistema propio de redes con independencia entre las máquinas virtuales completa, VLANS, Bondings,...).
- Gestión de almacenamiento (en red, local, en la nube).
- Gestión de Alta Disponibilidad y Clusters.

***OpenNebula nació en el DSA (Distributed Systems Architecture Research Group) http://dsa-research.org**

- Gestión de Zonas (posibilidad de ubicar máquinas en diferentes espacios)
- Gestión de Virtual Centers (Virtual Data Center)
- Gestión de Nubes Híbridas (Amazon EC2)
- Gestión de Markets (Apps a través de todas las instancias de OpenNebula y su distribución)
- Servicios de Monitorización
- API para integración
- Sistema de Hooks (Ejecución de ficheros personalizados)

Este conjunto de funcionalidades convierte a OpenNebula en un completo gestor de VDC (Virtual Data Center) que permite gestionar desde la capa más básica de red y almacenamiento hasta los procesos de gestión de usuarios, tiempos de uso y explotación de los recursos además de su escalado bajo demanda.

Fuente[*9]:
https://opennebula.org/about/project/
http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=820:introduccion-a-la-nube-open-nebula-como-caso-de-exito&catid=108:blog-cenatic&Itemid=150

5.6. Eucalyptus

Eucalyptus (eucalipto) es una infraestructura (plataforma) de código abierto para la implementación de computación en nube privada en clústers de ordenadores.

Su nombre hace referencia al acrónimo "Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems" que puede traducirse como "Utilidad de arquitectura informática elástica para confiar sus programas a sistemas funcionales". *Eucalyptus* puede instalarse fácilmente en la mayoría de distribuciones GNU/Linux.

También puede usar gran variedad de tencologías de virtualización de hardware incluyendo hipervisores VMware, Xen y KVM para implementar las abstracciones de nube que soporta.

Eucalyptus implementa nubes de tipo privado e híbrido, de estilo de Infraestructura como Servicio.

Eucalyptus incluye las siguientes funciones:

- Compatibilidad con la API Amazon Web Services.
- Instalación y desarrollo con recursos de gestión de clústers de ordenadores Rocks Linux, desde código o paquetes DEB y RPM.
- Comunicación segura entre los procesos internos vía SOAP y WS-Security.
- Recursos para administración básica.
- Capacidad de configurar múltiples clústeres de servidores como una sola "cloud".
- Soporte para máquinas virtuales Linux y Windows.
- Direcciones IP elásticas y grupos de seguridad.
- Gestión de usuarios y grupos.
- Informes de contabilidad.
- Políticas programables y configurables.

Fuente[*10]https://docs.eucalyptus.com/eucalyptus/latest/#install-guide/intro.html

5.7. Comparativa de plataformas de código abierto

En esta sección se muestra la única comparativa encontrada que **se hizo en el año 2012** sobre las plataformas de código abierto siguientes: CloudStack, OpenStack, OpenNebula y Eucalyptus.

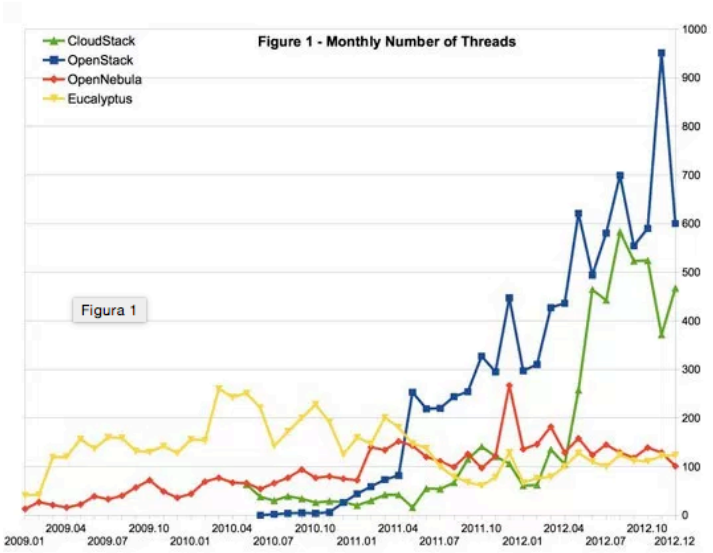


Figura 18 – Número mensual de temas en foros

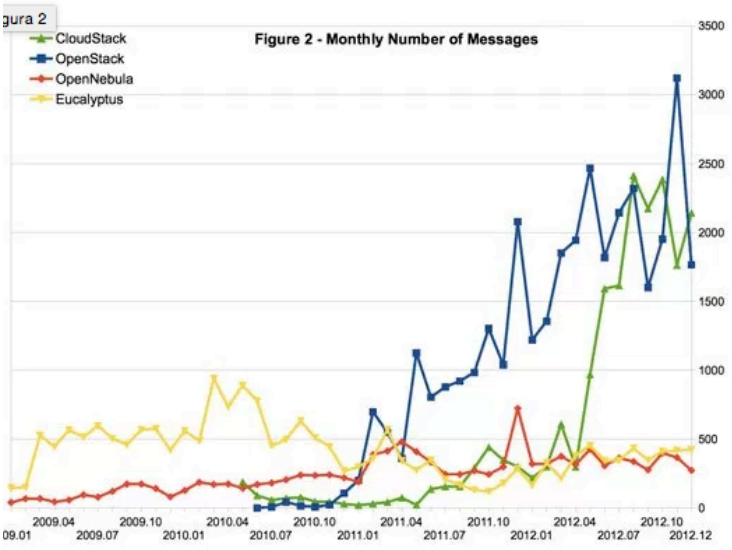


Figura 19 – Número mensual de mensajes en foros

Las Figuras 18 y 19 representan el número mensual de temas (threads) y mensajes (topics).

Se puede observar que el volumen de las discusiones OpenStack y CloudStack relacionados es mucho mayor que la de Eucalyptus y OpenNebula.

En el pasado el proyecto OpenStack tenía una relación de participación mucho más alto que los otros. Sin embargo, durante los últimos 6 meses, la tasa de participación de CloudStack y Eucalyptus esta creciendo de manera constante, mientras que la tasa de participación de OpenStack está disminuyendo gradualmente.

Actualmente CloudStack y Eucalyptus tienen las más altas tasas de participación, mientras OpenStack y OpenNebula tienen tasas de participación relativamente bajas.

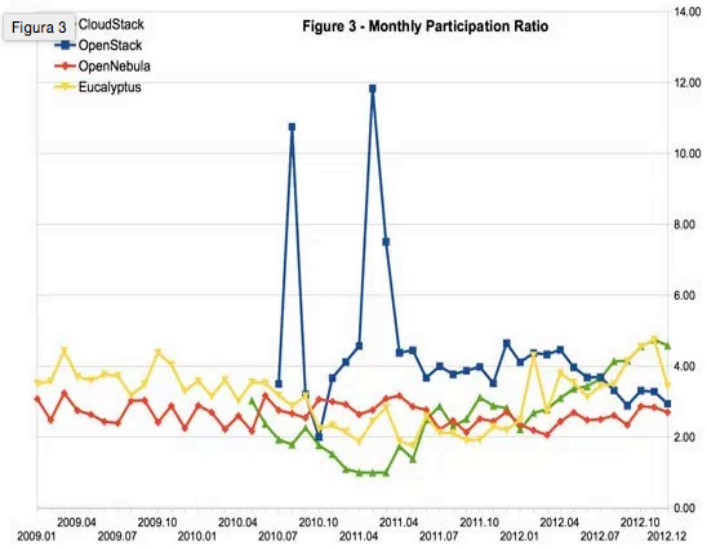


Figura 20 - Esquema Lógico y de comunica 1

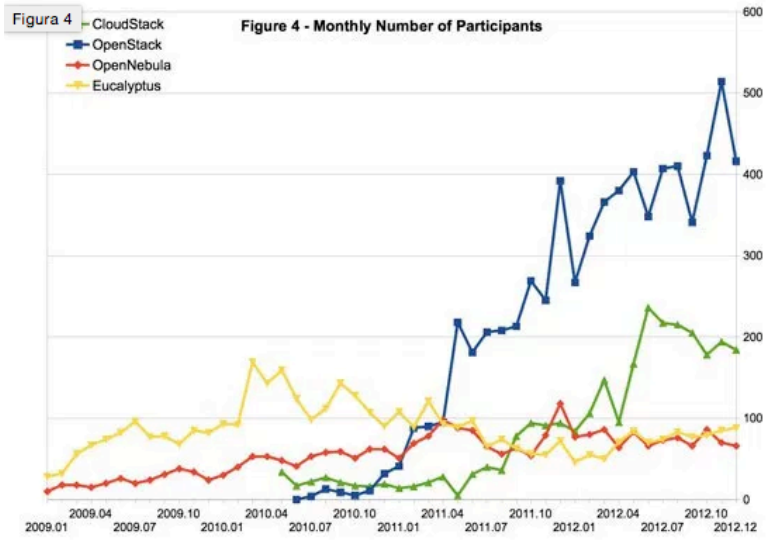


Figura 21 - Esquema Lógico y de comunica 2

Se puede observar que los participantes activos del CloudStack y OpenStack son mucho mayores que OpenNebula y Eucalyptus.

Sin embargo, durante los últimos 3 meses de la gráfica, el número de participantes, tanto para CloudStack y OpenStack han disminuido ligeramente.

Cabe señalar que, aunque el número de participantes activos de CloudStack es algo menor que OpenStack, pero el volumen de la discusión (en términos de número mensual de hilos y mensajes) de los dos proyectos están en el mismo nivel. Esto indica que los miembros activos del club CloudStack hablan más que los del club de OpenStack (en promedio).

Fuente[*11]: https://www.revistacloudcomputing.com

6. *Objetivos*

Los objetivos de este TFM los podemos separar en dos partes bien diferenciadas:

Objetivos formativos:

- Mejorar los conocimientos fundamentales de la virtualización y de la computación en la nube en un sistema real.
- Consolidar y aumentar los conocimientos acerca de las tecnologías de la virtualización en un ambiente real.
- Aplicar conocimientos teóricos a un problema real.
- Aplicar los conocimientos adquiridos en los estudios de Máster en ingeniería informática en el diseño, instalación y despliegue de un sistema informático destinado a proporcionar recursos de cómputo y almacenamiento.
- Adquirir el conocimiento necesario sobre leyes y normativas aplicables al ámbito de este Trabajo de Fin de Máster

Objetivos tecnológicos:

El objetivo tecnológico fundamental es el de diseñar y configurar una nube privada, basada en la plataforma OpenStack con el objetivo de proporcionar instancias (Máquinas Virtuales) y recursos virtuales a laboratorios docentes.

Para ello desarrollaremos los siguientes objetivos:

- Desarrollar el catálogo de requisitos del sistema a construir.
- Elaborar el diseño de la solución.
- Instalar y configurar todos los elementos básicos de cómputo y almacenamiento.
- Crear los distintos elementos básicos de la infraestructura en la nube, tales como usuarios, proyectos, roles, imágenes y volúmenes de almacenamiento.

7. JUSTIFICACIÓN DE LAS COMPETENCIAS ESPECÍFICAS

Competencias Genéricas y transversales

ULPGC1. Liderar equipos y organizaciones, promoviendo el libre intercambio de ideas y experiencias, la búsqueda de soluciones originales y el compromiso permanente con la excelencia.

No se puede decir que haya liderado un equipo debido a que implícitamente mi proyecto ha sido individual, pero si puedo concluir este objetivo ya que he tenido colaboradores en el centro de cálculo del departamento de informática y sistemas con los cuales me he sentado a debatir y hemos desarrollado soluciones a problemas que nos han surgido.

ULPGC2. Impulsar responsablemente todas las formas de conocimiento y de acción que puedan contribuir al enriquecimiento del capital económico, social y cultural de la sociedad en la que desarrolla su práctica profesional y en la que ejerce sus derechos y deberes de ciudadanía.

El objetivo del trabajo es precisamente la competencia ULPGC2. Donde fomentamos las formas de conocimiento a través de laboratorios virtuales los cuales contribuyen al enriquecimiento de la sociedad.

Competencias del real decreto 1393/2007 para el Máster

RDM1. Que los estudiantes sepan aplicar los conocimientos adquiridos y su capacidad de resolución de problemas en entornos nuevos o poco conocidos dentro de contextos más amplios (o multidisciplinares) relacionados con su área de estudio

El Trabajo de Fin de Máster parte del Máster en Ingeniería Informática en el cual no se estudia en principio nada sobre virtualización, a partir de mi especialización es cuando decido aplicar mis estudios a este proyecto, solucionando un problema aportando una solución.

RDM2. Que los estudiantes sean capaces de integrar conocimientos y enfrentarse a la complejidad de formular juicios a partir de una información que, siendo incompleta o limitada, incluya reflexiones sobre las responsabilidades sociales y éticas vinculadas a la aplicación de sus conocimientos y juicios

Al realizar la primera reunión con el Centro de Cálculo con el Departamento de Informática y Sistemas de la Universidad de Las Palmas de Gran Canaria la información aún no estaba madurada por lo que teníamos información incompleta y

limitada, en el transcurso del trabajo fuimos solucionando los inconvenientes según iban llegando por lo que considero este objetivo cumplido.

RDM3. Que los estudiantes sepan comunicar sus conclusiones y los conocimientos y razones últimas que las sustentan a públicos especializados y no especializados de un modo claro y sin ambigüedades

Con la realización de esta memoria queda cubierta esta justificación.

RDM4. Que los estudiantes posean las habilidades de aprendizaje que les permitan continuar estudiando de un modo que habrá de ser en gran medida auto dirigido o autónomo.

En mi proyecto muchas de las partes se han desarrollado de forma autodidacta y autónoma por lo que queda cubierto esta justificación.

Competencias Específicas

C01 Capacidad para proyectar, calcular y diseñar productos, procesos e instalaciones en todos los ámbitos de la ingeniería informática.

El objetivo del trabajo en sí cubre esta justificación.

C02 Capacidad para la dirección de obras e instalaciones de sistemas informáticos, cumpliendo la normativa vigente y asegurando la calidad del servicio.

El trabajo tiene como objetivo crear una infraestructura para proveer un servicio siempre cumpliendo con la normativa vigente.

C03 Capacidad para dirigir, planificar y supervisar equipos multidisciplinares.

Se ha dirigido este proyecto, generando una planificación y colaborando con personal de distintas áreas de la informática.

C04 Capacidad para el modelado matemático, cálculo y simulación en centros tecnológicos y de ingeniería de empresa, particularmente en tareas de investigación, desarrollo e innovación en todos los ámbitos relacionados con la Ingeniería Informática.

Este aspecto no se ha cubierto pues el trabajo no lo requería

C05 Capacidad para la elaboración, planificación estratégica, dirección, coordinación y gestión técnica y económica de proyectos en todos los ámbitos de la Ingeniería Informática siguiendo criterios de calidad y medioambientales.

El trabajo requiere de elaboración, planificación, coordinación y gestión técnica por lo que se considera cubierto.

C06 Capacidad para la dirección general, dirección técnica y dirección de proyectos de investigación, desarrollo e innovación, en empresas y centros tecnológicos, en el ámbito de la Ingeniería Informática.

Este aspecto no se ha cubierto pues el trabajo no lo requería

C07 Capacidad para la puesta en marcha, dirección y gestión de procesos de fabricación de equipos informáticos, con garantía de la seguridad para las personas y bienes, la calidad final de los productos y su homologación.

Este aspecto no se ha cubierto pues el trabajo no lo requería

C08 Capacidad para la aplicación de los conocimientos adquiridos y de resolver problemas en entornos nuevos o poco conocidos dentro de contextos más amplios y mulitidisciplinares, siendo capaces de integrar estos conocimientos.

La virtualización es un mundo novedoso por lo que creemos justificar esta competencia.

C09 Capacidad para comprender y aplicar la responsabilidad ética, la legislación y la deontología profesional de la actividad de la profesión de Ingeniero en Informática.

Con el apartado de legislación y normativa cubrimos este apartado.

C10 Capacidad para aplicar los principios de la economía y de la gestión de recursos

Se han gestionado recursos informáticos, de forma económica no se han gestionado ya que no hemos manejado recursos económicos.

8. APORTACIONES

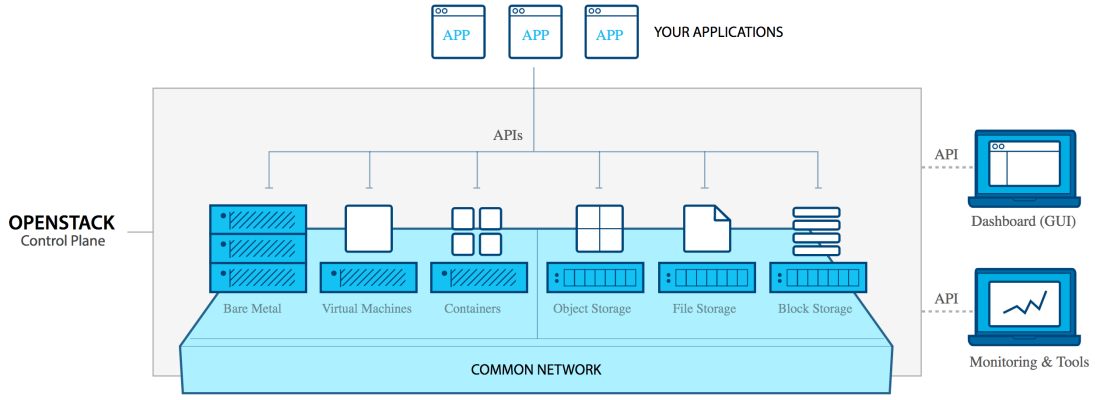
Este trabajo sirve como un estudio o **paso previo** a la virtualización de los laboratorios docentes, donde el objetivo, es tener una **idea más precisa** acerca de los **recursos, costes y tareas a acometer** para virtualizar los laboratorios docentes:

- La virtualización de los laboratorios **generará una mayor seguridad** frente a fallos ocasionados por las modificaciones practicadas por el alumnado en las asignaturas cursadas.
- Con la virtualización de los laboratorios **la disponibilidad del mismo** se incrementará ya que se podrá acceder a las máquinas desde el exterior de las dependencias del laboratorio (siempre y cuando estén en la misma red) a una máquina similar a las instaladas en los laboratorios.
- **Reducción del coste** (electricidad, actualización de terminales, mantenimiento, etc.…).
- **Mayor rendimiento del servidor** incrementando el % de uso de CPU al incrementar la consolidación.
- **Gasto menor en compra de nuevo material**. Cada centro temporalmente renueva los dispositivos (renovando material antiguo), con este trabajo logramos una reducción de costes mejorando el servidor central para dar mayor/mejor servicio.

Es decir, se proporciona una alternativa al uso de los laboratorios y simultáneamente un ahorro de costos a la administración.

9. OPENSTACK

9.1. ¿Qué es?



Openstack es una colección de proyectos con licencia *Open Source* que suministra software para la construcción de nubes, es un proyecto que se inició en el año 2010 por la empresa *Rackspace Cloud* y por la agencia espacial norteamericana NASA.

En la actualidad existen al menos 150 empresas afiliadas al proyecto, entre las que se encuentran AMD, Intel, Red Hat, IBM, DELL, HP, Cisco y muchas más.

9.2. Características.

Desde un punto de vista global OpenStack está construido para ser un "sistema operativo para el despliegue de nubes masivamente escalables".

Para lograr el objetivo cada uno de los servicios, módulos o proyectos se intercomunican entre sí a través de un módulo de paso de mensajes como puede ser **Rabbit QM**.

Actualmente *OpenStack* tiene en activo cinco proyectos relacionados:

- ***OpenStack Compute (Nova)*** que nos suministra máquinas virtuales y todo tipo de recursos de cómputo.

- ***OpenStack Object Storage (Swift)*** que proporciona recursos de almacenamiento.
- ***OpenStack Image Service (Glance)*** que suministra entre otras cosas las imágenes que serán utilizadas por las máquinas virtuales.
- ***OpenStack Networking (Networking)*** *que proporciona soporte en la creación de redes.*
- ***OpenStack Volume Service (Cinder)*** *con el cual podemos añadir volúmenes a nuestras instancias.*

Al ser una plataforma *Open Source* cualquier organización puede desplegar una nube a gran escala tanto para uso privado como para uso público.

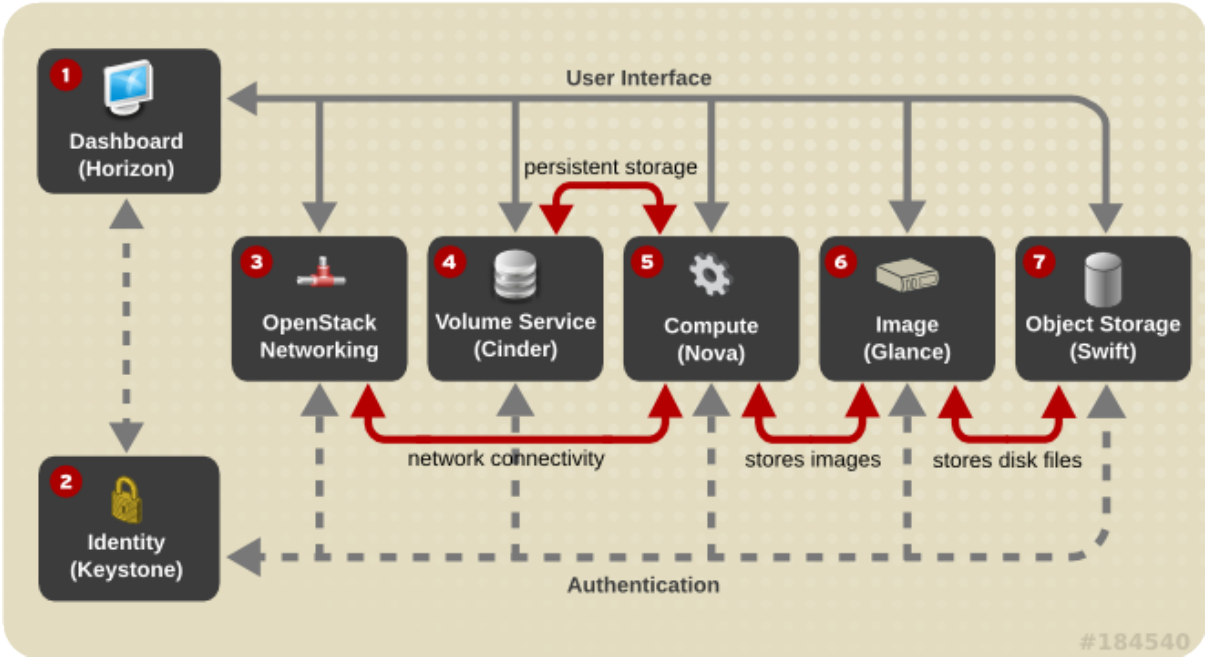


Figura 22 - Relaciones entre proyectos OpenStack

Fuente[*8]: www.openstack.com

9.3. *Arquitectura Lógica de OpenStack*

Aunque desde un punto de vista lógico OpenStack es inmensamente podemos reseñar de una forma resumida 3 aspectos clave:

- Los usuarios finales interactúan a través de una interfaz visual.
- Todos los servicios son previamente autenticados a través del servicio *keystone.*
- Todos los servicios interactúan entre sí a través de APIs públicas.

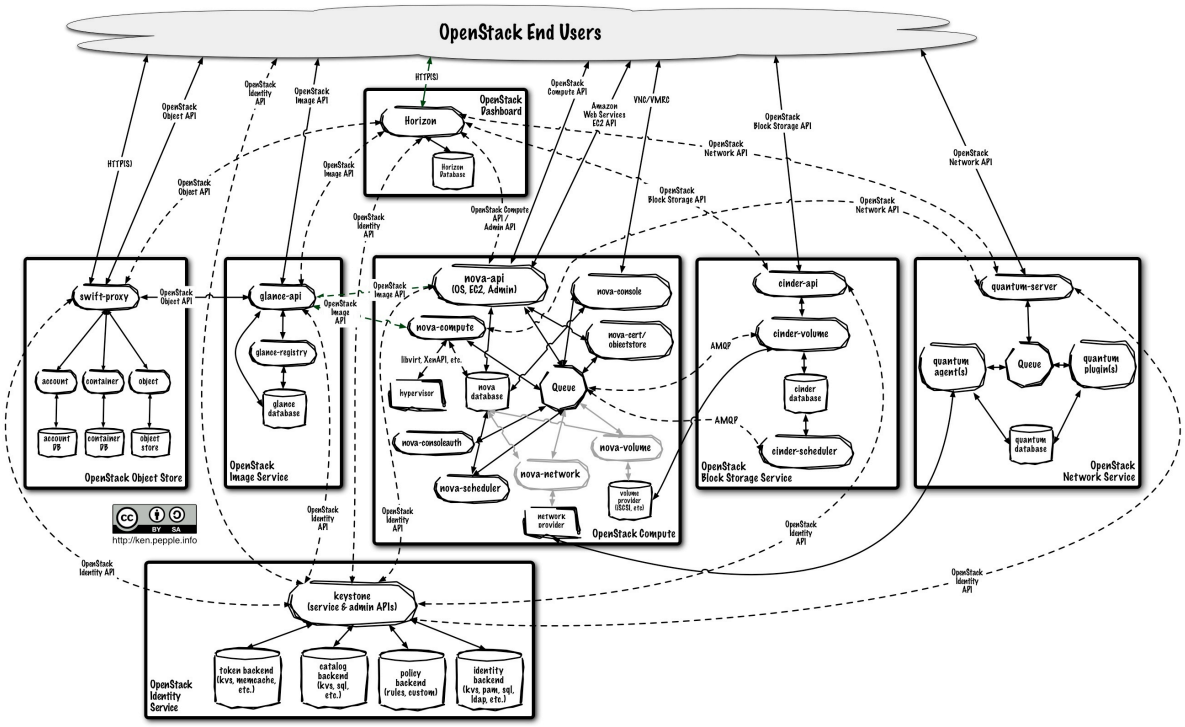


Figura 23 - Esquema Lógico y de comunicacióna

9.4. Componentes

OpenStack es una herramienta que está compuesta de una serie de componentes diferenciados en proyectos, cada proyecto tiene su propio encargado y se comunica con los demás por medio de una API externa y un software encargado de enviar mensajes a otros módulos. A continuación, procederemos a enumerar todos los paquetes que componen OpenStack al momento de escritura de este trabajo:

9.4.1. Dashboard (Horizon)

Una interfaz de control web para la gestión de los servicios de OpenStack donde se puede lanzar máquinas virtuales, asignar direcciones IP, configurar usuarios, proyectos , etc …

¿Qué permite?

- Crear y lanzar instancias de máquinas virtuales.
- Gestionar redes y establecer controles de acceso.
- Crear proyectos, usuarios, roles y en general controlar los usuarios.
- Crear imágenes de Sistemas Operativos.
- Crear sabores (plantillas de máquinas físicas).
- Crear avisos y visualizar estadísticas.

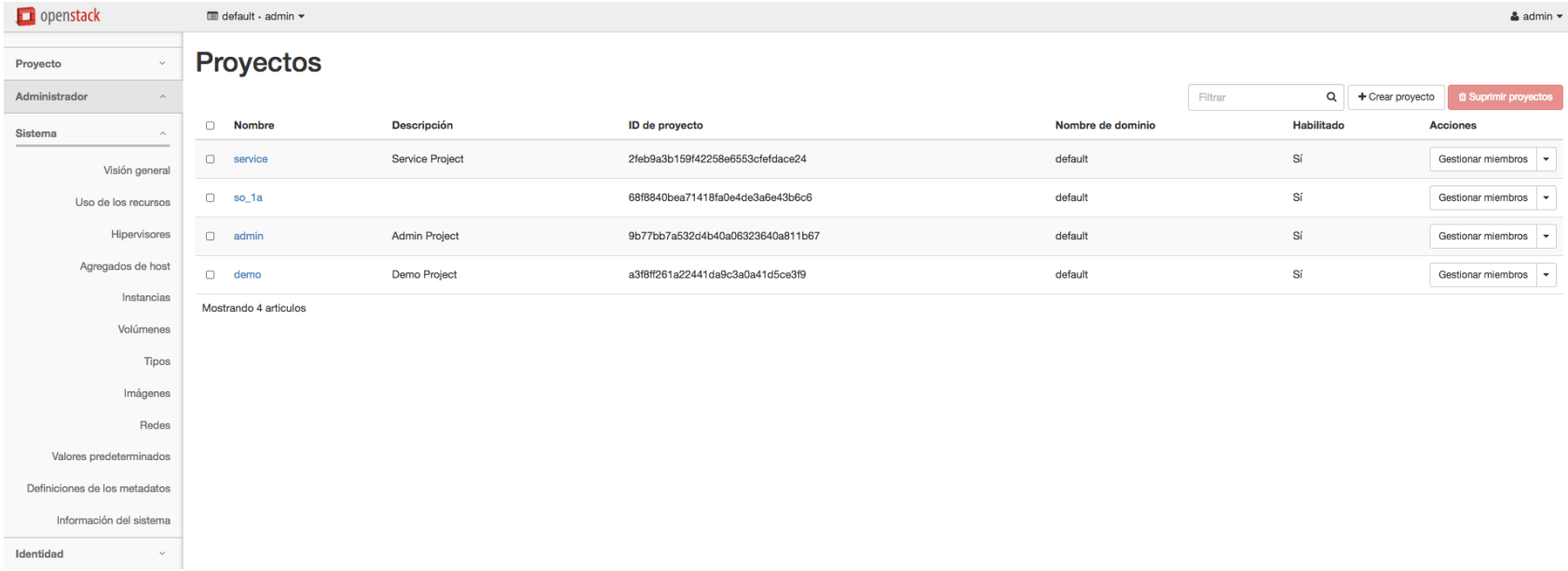


Figure 16 - Captura de pantalla de la interfaz de OpenStack

9.4.2. *Identity (Keystone)*

Un servicio de autenticación centralizado que proporciona autorización de usos de los servicios, y sus direcciones de acceso (*endpoints*). Gestiona usuarios, contenedores de recursos y roles.

Keystone permite la integración de los servicios de OpenStack de forma centralizada. Cada función que keystone integra en su base de datos se conecta a un servicio diferente.

Características

- Es utilizado por todos los componentes de OpenStack.
- El servicio soporta múltiples formas de autentificación, incluyendo nombre de usuario y contraseña; sistemas basados en *tokens*; y los basados en AWS style (*Amazon Web Services*).
- El servicio de identidad también ofrece un catálogo central de servicios y direcciones de acceso a los mismos (*endpoints*), ejecutándose en una nube OpenStack particular.
- Actúa como un directorio de servicios para otros sistemas OpenStack.

¿Qué permite?

- Autenticar y autorizar a los usuarios de OpenStack además de realizar un seguimiento de los usuarios y sus actividades permitidas.

9.4.3. Networking (Neutrón)

El objetivo del proyecto Neutron es proporcionar red como servicio (NaaS) entre las interfaces de red gestionadas por Nova(Compute). Esto permite una mayor flexibilidad a la hora de crear nuevas interfaces de red e interconexionar máquinas. Admite plugins para trabajar con diferentes tecnologías.

Características

- Se comunica principalmente con el servicio Nova (Compute) para suministrar servicios de red y conectividad a sus instancias.
- Sus plug-ins permiten acomodarse a diferente hardware y software, permitiendo gran variedad de arquitecturas.
- Ofrece una gran flexibilidad en modelos de redes, el administrador puede cambiar la topología de las redes y adaptarlas a sus necesidades.
- Las direcciones IPs pueden ser estáticas o dinámicas (*floating IPs*), permitiendo estas últimas enrutar el tráfico dinámicamente.

¿Qué permite?

- Nos suministra la conectividad en red como un servicio entre los dispositivos permitiendo la comunicación de los servicios Openstack.
- Los usuarios pueden crear redes, controlar el tráfico, y conectar máquinas virtuales y dispositivos a una o más redes.

9.4.4. Volume Cinder (Cinder)

Servicio que maneja volúmenes de almacenamiento de bloques persistente para máquinas virtuales. Es equivalente al servicio *Elastic Block Storage (EBS)* de *Amazon*.

Características

- Interactúa con el servicio Nova (Compute) para proporcionar volúmenes a las máquinas virtuales.
- Proporciona una herramienta de trabajo con la cual crear, listar, y borrar capturas de máquinas virtuales.

¿Qué permite?

- Montar y desmontar volúmenes para asociarlos a máquinas virtuales.
- Crear, listar, modificar y borrar volúmenes.

9.4.5. Compute (Nova)

Maneja el ciclo de vida de las instancias (MVs) que se ejecutan en los nodos. Este servicio permite la creación, ejecución, eliminación de las instancias bajo demanda.

Características

- El servicio *Compute* es el corazón de *OpenStack*, es el controlador todo el sistema IaaS y permite manejar un sistema de computación en la nube.
- Sus módulos principales están hechos en Python.
- El servicio dialoga con el servicio Identity para autenticarse.
- El servicio Image le sirve imágenes.
- El servicio Dashboard para la interfaz web de los usuarios y del administrador utiliza el módulo de compute para crear máquinas virtuales.
- El acceso a las imágenes está limitado por proyecto en OpenStack llamado *tenant* y por usuario.
- Compute ha sido desarrollado para ejecutarse horizontalmente sobre un hardware estandar y puede descargar imágenes para lanzar instancias cuando son requeridas.

¿Qué permite?

- Crear máquinas virtuales por demanda. Compute es el planificador para que se ejecuten sobre un conjunto de nodos de cómputo.

9.4.6. Image (Glance)

El servicio Glance actúa como un registro para almacenar las imágenes virtuales de disco.

Los usuarios pueden añadir nuevas imágenes o realizar una instantánea (snapshot) para copiar el estado de una máquina virtual para ser usado como una copia de seguridad o como una imagen plantilla para nuevas máquinas virtuales.

Las imágenes registradas pueden almacenarse en el servicio *Object Storage*, y también en otros lugares (por ejemplo, en un sistema de ficheros o en un servidor web externo).

Se soportan los siguientes formatos para las imágenes:

- RAW (formato sin estructura).
- aki/ami/ari (formatos de *Amazon Web Services*).
- iso (formato para discos ópticos como CDROM).
- qcow2 (Qemu/KVM, soporta Copy on Write).
- vhd (Hyper-V, común para máquinas virtuales monitorizadas por VMWare, Xen, Microsoft, VirtualBox, y otros).
- vdi (Qemu/VirtualBox).
- vmdk (VMWare).

9.4.7. Object Storage (Swift)

El servicio *Object Storage* proporciona almacenamiento de objetos en contenedores virtuales, lo que permite a los usuarios almacenar y recuperar archivos.

Al ser una arquitectura distribuida nos proporciona tolerancia frente a fallos y además nos posibilita el escalado horizontal.

Object Storage utiliza el concepto de:

- **Storage réplicas** es utilizado para mantener el estado de los objetos en caso de interrupción. Se recomienda un mínimo de tres replicas.

- **Storage zones** es utilizado para hospedar las réplicas. Las zonas aseguran que cada réplica de un objeto dado puede ser almacenado separadamente. Una zona puede representar un disco individual o un array de discos, un servidor, todos los servidores en un rack o incluso un centro de datos entero.

- **Storage regions** es básicamente un grupo de zonas que comparten una localización.

9.4.8. Telemetry (Ceilometer)

El servicio de telemetría proporciona datos de uso a nivel de usuario de los recursos de la nube. Se utiliza para la facturación al cliente, la supervisión del sistema o alertas.

Los datos pueden recogerse mediante notificaciones enviadas por los servicios de OpenStack (por ejemplo, los eventos de uso se emiten desde Nova(*Compute*)) o interrogando a la infraestructura (por ejemplo, con libvirt).

Telemetría incluye un demonio que se comunica a través de un sistema de mensajería de confianza con el sistema de autenticación para recopilar y agregar datos.

El servicio posee un sistema de plugins que hace que sea fácil agregar nuevos monitores de medición.

9.4.9. Orchestation (Heat)

El servicio Orchestration proporciona un instrumento de orquestación basado en plantillas para los servicios que se prestan en la nube OpenStack.

Se utiliza para crear y administrar los recursos de infraestructura de la nube como almacenamiento, redes, instancias y aplicaciones como un entorno repetible de ejecución.

Características

- El servicio ofrece acceso a todos los servicios básicos de OpenStack a través de una única plantilla modular, con capacidades adicionales de orquestación, como auto escalado y alta disponibilidad básica.
- Una única plantilla da acceso a todas las APIs.
- Las plantillas pueden definirse recursivamente y ser rehusadas. Esto hace que la infraestructura de la nube pueda ser definida y reutilizada de una forma modular.
- La implementación de recursos se hace mediante plugins, lo cual permite utilizar recursos particulares.

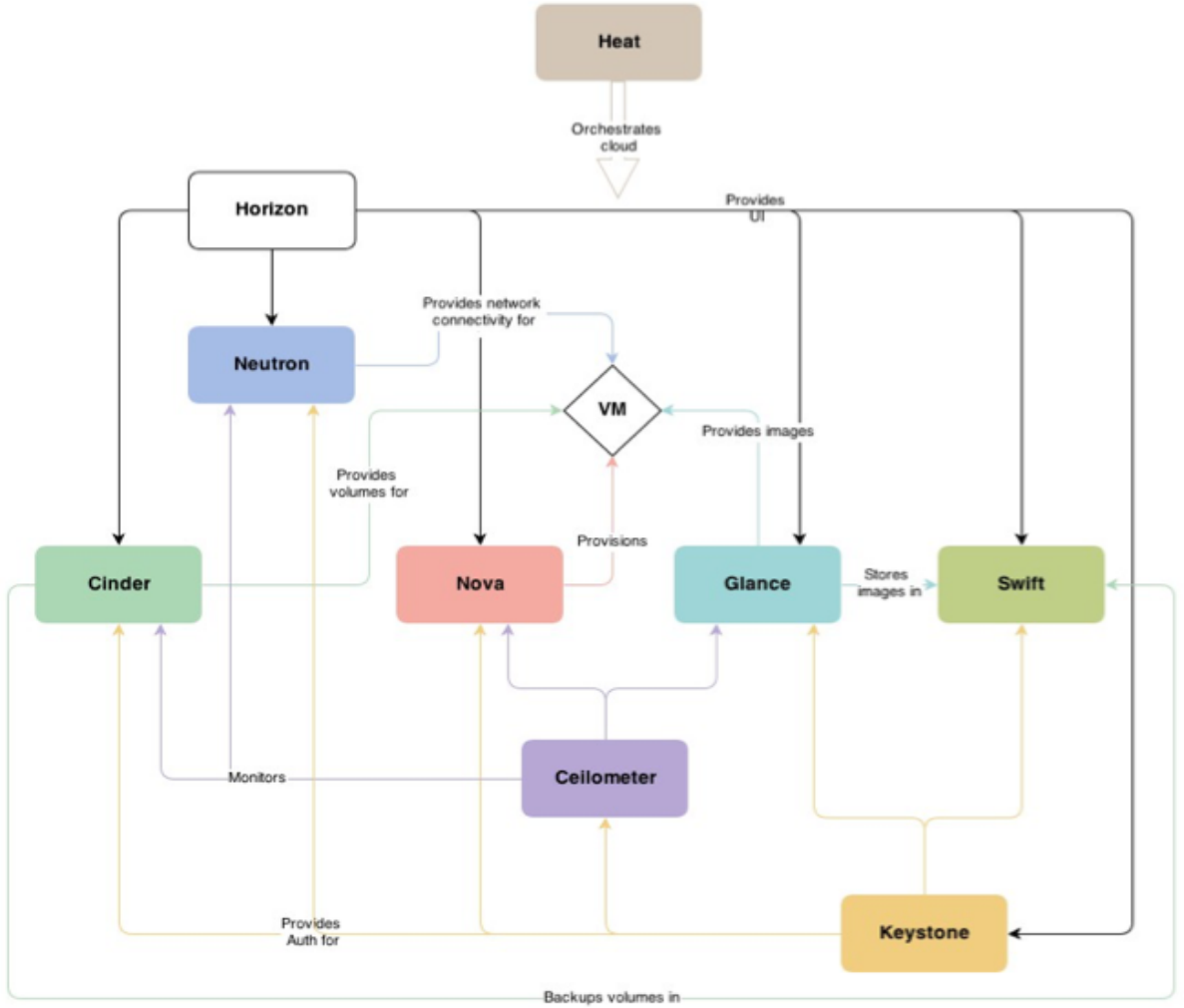


Figure 17-Arquitectura de OpenStac

10. Instalación y Configuración de OpenStack.

¿De donde se parte?

El proyecto inicia con una reunión inicial con los miembros responsables del centro de cálculo del edificio de informática y matemáticas de la Escuela de Ingeniería Informática.

En esta reunión se establecen de forma teórica los recursos tecnológicos requeridos por la plataforma, fundamentalmente hardware, que se destinaran al trabajo cuyas especificaciones se indican a continuación:

4 servidores con las siguientes especificaciones:

- 4 núcleos.
- 8 Gb de Memoria RAM.
- 3 tarjetas ethernet cada uno.
- 160Gb de Disco Duro.

Además, para generar espacio de disco para las máquinas virtuales de destinará un servidor conectado a una cabina SAN para almacenamiento donde la única especificación que tenemos es que vamos a tener un volumen lógico de 500Gb

La siguiente figura representa el estado en el que nos encontramos los servidores pre-instalados listos para ser usados.

Una vez llegado el momento vimos que las especificaciones teóricas no se podían cumplir y lo que pudimos obtener fueron 4 servidores con las siguientes características:

4 servidores con las siguientes especificaciones:

- 4 núcleos.
- 4 Gb de Memoria RAM.
- 2 tarjetas ethernet cada uno.
- 160Gb de Disco Duro.

Contamos también con un switch que interconecta a todos los servidores en una red interna de tipo 192.168.100.X/24.

Vamos a usar una arquitectura distribuida de 4 servidores donde 1 servidor será el que maneje toda la logística de la arquitectura (control), 2 servidores serán los que proveerán de recursos de cómputo(compute) y por último 1 servidor que provea espacio de disco bajo demanda (block1).

Después de un análisis exhaustivo de las necesidades del proyecto y de diversas pruebas preliminares con el hardware y la plataforma OpenStack se llega a las siguientes consideraciones:

1. Debido a varios experimentos y problemas con la configuración del NAT al final decidimos pedir otra IP Pública temporal para el servidor de cómputo.

2. Es necesario que un servidor destinado a recursos de cómputo tenga al menos 8Gb de Memoria RAM como requisito de *OpenStack*. Por este motivo se decide prescindir de uno de los servidores y emplear la RAM de dicho servidor para que uno de los servidores cumpla este requisito mínimo.

3. El servidor destinado a generar espacio de disco a demanda y al tener solamente 2 interfaces de red (una para la red interna y otra para la cabina SAN) no puede estar conectado directamente a internet. Debido a que esto no es un requisito indispensable no incurre en un problema mayor.

4. Las interfaces Ethernet de los servidores quedan de la siguiente manera:

    a. Servidor 1 (Control): Va a dirigir todas las operaciones de la infraestructura

        i. Interfaz A => Internet

        ii. Interfaz B => Red Interna

    b. Servidor 2 (Compute): Va a proveer máquinas virtuales

        i. Interfaz A => Libre

        ii. Interfaz B => Red Interna

    c. Servidor 3 (Block1): Encargado de proveer espacio de disco bajo demanda.

        i. Interfaz A => Red Interna

        ii. Interfaz B => Cabina SAN

5. En un principio se instaló Centos 6.7 pero después de varias semanas comprobamos que es un requisito de OpenStack versión Mitaka es usar Centos 7 lo que era una tarea de aprendizaje añadida tanto para mí como

para los miembros del CDIS debido a amplias diferencias en órdenes con Centos 6.7 (CentOS 7.0 tiene una nueva filosofía)

6. Con respecto a la cabina también tuvimos algunos problemas para conectar, en primer lugar, debido a que la cabina daba fallos y en segundo lugar debido a un nuevo método para autenticar que no estaba documentado.

Tras esta fase de pruebas de la infraestructura hardware y software, que se alargo durante un periodo de 2 meses, comenzamos la siguiente fase del proyecto empleando para ello los siguientes recursos:

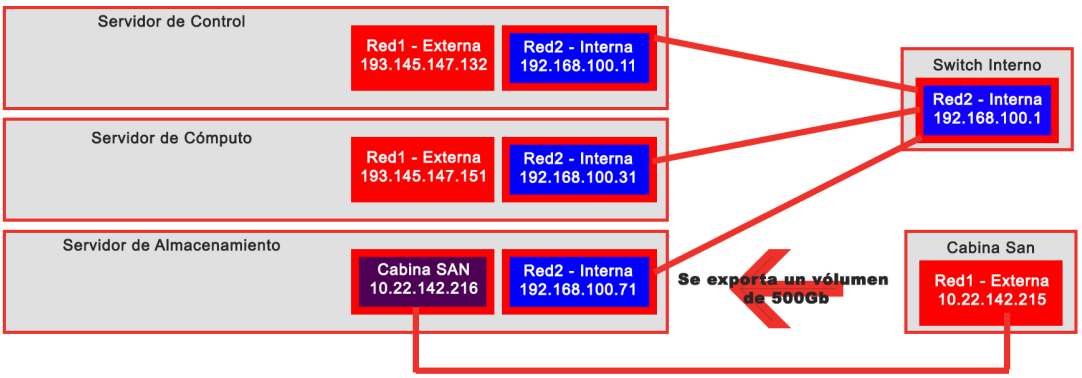


Figura 18 - Esquema configurado

Como muestra la figura21 el sistema configurado finalmente se compone de 3 servidores, una red interna y una conexión a una cabina SAN:

- Un servidor anfitrión con sistema operativo **CentOS 7.0** y con núcleo de virtualización "**KVM**" con el objetivo de ser la máquina de control
- Un servidor anfitrión con sistema operativo **CentOS 7.0** y con núcleo de virtualización "**KVM**" con el objetivo de ser la máquina de cómputo.
- Un servidor anfitrión con sistema operativo **CentOS 7.0** y con núcleo de virtualización "**KVM**" con el objetivo de ser la máquina para proveer de espacio de disco.

Las especificaciones de las máquinas finalmente son las siguientes:

- **Control**:
    - Un procesador de 4 núcleos,
    - 4Gb de Memoria RAM
    - Un espacio de disco duro de 250Gb.
    - Además, tiene 2 interfaces de RED
        - Internet
        - Red Interna

    El sistema operativo que ejecuta el Servidor es CENTOS 7.0 64bits
- **Cómputo**:
    - Un procesador de 4 núcleos,
    - 8Gb de Memoria RAM
    - Un espacio de disco duro de 250Gb.
    - Además, tiene 2 interfaces de RED
        - Internet
        - Red Interna

    El sistema operativo que ejecuta el Servidor es CENTOS 7.0 64bits
- **Block1**:
    - Un procesador de 4 núcleos,
    - 4Gb de Memoria RAM
    - Un espacio de disco duro de 250Gb.
    - Además, tiene 2 interfaces de RED
        - Internet
        - Red Interna

El sistema operativo que ejecuta el Servidor es CENTOS 7.0 64bits

La configuración de red será la siguiente:

Red Interna (192.168.100.0/24) será la red empleada por los nodos de OpenStack para el Control y administración de los servicios, es decir, será la red de gestión, administración y la que gestionará el correcto movimiento de datos entre máquinas de la arquitectura.

Internet es la red que conecta las máquinas virtuales al exterior y por lo tanto a internet para poder instalar los paquetes de OpenStack y para que los usuarios se puedan conectar por SSH y ejecutar los diferentes servicios ofrecidos por la infraestructura

10.1. Instalación de la plataforma KVM en todas las máquinas de la Infraestructura

Para instalar la plataforma KVM hemos hecho lo siguiente:

Primero verificamos que el procesador soporta virtualización por hardware con la orden:

```
# grep -E '(vmxlsvm)' /proc/cpuinfo
```

Para comprobar si soporta virtualización en la salida por pantalla deben salir las palabras **vmx** o **svm,** de lo contrario la CPU no soporta virtualización por hardware

a) Con la orden

```
# yum install qemu-kvm qemu-img virt-manager libvirt libvirt-python libvirt-client virt-install virt-viewer bridge-utils
```

Instalamos todas las librerías necesarias para el correcto funcionamiento de KVM.

b) Arrancamos el demonio libvirt y hacemos que el sistema lo inicie al arranque.

```
# systemctl start libvirtd
# chkconfig libvirt on
```

c) Actualizamos la instalación y reiniciamos la máquina con las siguientes órdenes:

```
# yum upgrade
# reboot
```

10.2. Configuración de red de la máquina Controlller

Para que la máquina pueda atender todas las peticiones de los usuarios se tienen que habilitar varias interfaces de red, para ello se modifican los archivos situados en ***/etc/sysconfig/network-scripts***.

Cuando el sistema arranca utilizará estos archivos para saber que interfaces debe modificar, activar y configurarlas convenientemente, estos archivos tienen comúnmente el nombre ifcfg-enp[0-9]s[0-9].

ifcfg-enp5s0

```
TYPE=Ethernet
BOOTPROTO="static"
DEFROUTE=yes
NAME=enp5s0
UUID=33e36221-0471-4232-bc99-
773cc26e10e5
DEVICE=enp5s0
NM_CONTROLLED=no
ONBOOT=yes
IPADDR=193.145.147.132
PREFIX0=26
GATEWAY=193.145.147.129
```

ifcfg-enp9s0

```
TYPE=Ethernet
BOOTPROTO="static"
NAME=enp9s0
UUID=c94203f3-aeb0-4451-b3d2-
310a3e428dfc
DEVICE=enp9s0
ONBOOT=yes
NM_CONTROLLED=no
DEFROUTE=no
IPADDR=192.168.100.11
NETMASK=255.255.255.0
GATEWAY=192.168.100.1
```

El significado de los parámetros se explica a continuación:

- DEVICE es el nombre de la interfaz.
- HWADD es la dirección MAC de la tarjeta de red.
- TYPE es el tipo de interfaz a usar.
- ONBOOT establece si la interfaz se activa al inicio del sistema.
- NM_CONTROLLED indica si NETWORK MANANAGER es el encargado controlar la red.
- BOOTPROTO es el protocolo de red a usar (DHCP, ESTÁTICO).
- DEFROUTE para definir que se emplee la ruta por defecto.
- NETMASK establece la máscara de red.

10.3. Configuración de red de la máquina de Cómputo

Para que la máquina pueda atender todas las peticiones de los usuarios se tienen que habilitar varias interfaces de red, para ello se modifican los archivos situados en ***/etc/sysconfig/network-scripts***.

Cuando el sistema arranca utilizará estos archivos para saber que interfaces debe modificar, activar y configurarlas convenientemente, estos archivos tienen comúnmente el nombre ifcfg-enp[0-9]s[0-9].

ifcfg-enp5s0
BOOTPROTO="static"
DEFROUTE=yes
NAME=enp5s0
DEVICE=enp5s0
ONBOOT=yes
NM_CONTROLLED=no
IPADDR=193.145.147.151
PREFIX0=26
GATEWAY=193.145.147.129

ifcfg-enp9s0
TYPE=Ethernet
BOOTPROTO="static"
DEFROUTE=yes
NAME=enp9s0
DEVICE=enp9s0
ONBOOT=yes
IPADDR=192.168.100.31
NETMASK=255.255.255.0
GATEWAY=192.168.100.1

10.4. Configuración de red de la máquina Block1

Para que la máquina pueda atender todas las peticiones de los usuarios se tienen que habilitar varias interfaces de red, para ello se modifican los archivos situados en ***/etc/sysconfig/network-scripts***.

Cuando el sistema arranca utilizará estos archivos para saber que interfaces debe modificar, activar y configurarlas convenientemente, estos archivos tienen comúnmente el nombre ifcfg-enp[0-9]s[0-9].

ifcfg-enp5s0

```
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=no
NAME=enp5s0
UUID=cc78e769-d78b-415c-97a9-
bcacf506e70a
DEVICE=enp5s0
ONBOOT=yes
IPADDR=10.22.146.216
NETMASK=255.255.255.0
GATEWAY=10.22.146.1
```

ifcfg-enp9s0

```
TYPE=Ethernet
BOOTPROTO=static
DEFROUTE=yes
NAME=enp9s0
UUID=b8544e71-2493-434d-aee0-
fbe0b1056c09
DEVICE=enp9s0
ONBOOT=yes
IPADDR=192.168.100.71
NETMASK=255.255.255.0
GATEWAY=192.168.100.11
```

10.5. Verificación de la conectividad entre redes

Para verificar que las redes tienen conectividad entre máquinas virtuales y hacia el exterior se comprueba por medio de la herramienta PING.

10.6. Comprobaciones desde la máquina Control

En primer lugar, se verifica si la máquina Control tiene conectividad con la máquina Computo, con Nbloque1 y con internet. Para ello se ejecutan las siguientes órdenes:

```
# Ping compute1
# Ping block1
# Ping www.ulpgc.es
```

Conectividad con máquina cómputo

```
PING compute1 (192.168.100.11) 56(84) bytes of data.
64 bytes from compute1 (192.168.100.31): icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from compute1 (192.168.100.31): icmp_seq=2 ttl=64 time=0.018 ms
64 bytes from compute1 (192.168.100.31): icmp_seq=3 ttl=64 time=0.016 ms
```

Conectividad con máquina almacenamiento

```
PING block1 (192.168.100.71) 56(84) bytes of data.
64 bytes from block1 (192.168.100.71): icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from block1 (192.168.100.71): icmp_seq=2 ttl=64 time=0.018 ms
64 bytes from block1 (192.168.100.71): icmp_seq=3 ttl=64 time=0.016 ms
```

Conectividad con exterior

```
PING cluster-web-ldap.ulpgc.es (193.145.138.12) 56(84) bytes of data.
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=1 ttl=58
time=0.947 ms
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=2 ttl=58
time=0.962 ms
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=3 ttl=58
time=0.912 ms
```

10.7. Comprobaciones desde la máquina virtual Computo

En primer lugar, se comprueba si la máquina Computo tiene conectividad con la máquina Control, con Nbloque1 y con internet. Para ello se ejecutan las siguientes órdenes:

```
Conectividad de computo con máquina control

# Ping compute1

# Ping block1

# Ping www.ulpgc.es
```

```
PING controller (192.168.100.11) 56(84) bytes of data.
64 bytes from controller (192.168.100.11): icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from controller (192.168.100.11): icmp_seq=2 ttl=64 time=0.018 ms
64 bytes from controller (192.168.100.11): icmp_seq=3 ttl=64 time=0.016 ms
```

Conectividad de computo con máquina control

```
PING block1 (192.168.100.71) 56(84) bytes of data.
64 bytes from block1 (192.168.100.71): icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from block1 (192.168.100.71): icmp_seq=2 ttl=64 time=0.018 ms
64 bytes from block1 (192.168.100.71): icmp_seq=3 ttl=64 time=0.016 ms
```

Conectividad de computo con exterior

```
PING cluster-web-ldap.ulpgc.es (193.145.138.12) 56(84) bytes of data.
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=1 ttl=58
time=0.947 ms
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=2 ttl=58
time=0.962 ms
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=3 ttl=58
time=0.912 ms
```

10.8. Comprobaciones desde la máquina virtual NBloque1

En primer lugar, se testea si la máquina Nbloque1 tiene conectividad con la máquina Control, con Computo y con internet. A continuación se detallan las órdenes utilizadas para etas comprobaciones:

```
# Ping compute1
# Ping block1
# Ping www.ulpgc.es
```

Conectividad de control con almacenamiento

```
PING controller (192.168.100.11) 56(84) bytes of data.
64 bytes from controller (192.168.100.11): icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from controller (192.168.100.11): icmp_seq=2 ttl=64 time=0.018 ms
64 bytes from controller (192.168.100.11): icmp_seq=3 ttl=64 time=0.016 ms
```

Conectividad de computo con almacenamiento

```
PING compute1 (192.168.100.31) 56(84) bytes of data.
64 bytes from compute1 (192.168.100.31): icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from compute1 (192.168.100.31): icmp_seq=2 ttl=64 time=0.018 ms
64 bytes from compute1 (192.168.100.31): icmp_seq=3 ttl=64 time=0.016 ms
```

Conectividad de computo con exterior

```
PING cluster-web-ldap.ulpgc.es (193.145.138.12) 56(84) bytes of data.
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=1 ttl=58
time=0.947 ms
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=2 ttl=58
time=0.962 ms
64 bytes from cluster-web-ldap.ulpgc.es (193.145.138.12): icmp_seq=3 ttl=58
time=0.912 ms
```

10.9. Configuración del servicio Chrony (Sincronización del Reloj)

Network Time Protocol (NTP) es un protocolo de Internet para sincronizar los relojes de los sistemas informáticos a través del enrutamiento de paquetes en redes con latencia variable. NTP utiliza UDP como su capa de transporte, usando el puerto 123.

Chrony es un programa basado en NTP que sincroniza relojes del sistema con respecto a un servidor externo.

10.9.1. Instalación y configuración en la máquina virtual Control

a) Se instala el servicio Chrony y se arranca.

```
# yum install chrony
```

b) Se edita el archivo **/etc/chrony.conf** para añadir el servidor NTP que necesitemos usar.

c) Se habilita a otros nodos conectarse a nuestro nodo de control para sincronizarse con la directiva "allow".

d) Se establece que el demonio se inicie al iniciar el SO.

```
# systemctl enable chronyd.service
```

e) Se inicia el demonio.

```
# systemctl start chronyd.service
```

f) El servicio ntp usa el puerto udp 123 por lo que se tiene que abrir ese puerto en el cortafuego con las siguientes órdenes:

```
# iptables -A INPUT -m state --state NEW -m udp -p udp --
dport 123 -j ACCEPT
# service iptables save
# service iptables restart
```

10.9.2. Instalación y configuración en la máquina Computo y NBloque1

a) Se instala el servicio NTP y lo arrancamos.

```
# yum install chrony
```

b) El nodo Computo sincronizará a través del nodo Control por lo tanto se edita el archivo ubicado en ***/etc/chrony.conf*** y se añade la línea

```
# server controller iburst
```

a) Se establece el demonio se inicie al iniciar el SO.

```
# systemctl enable chronyd.service
```

b) Se inicia el demonio.

```
# systemctl start chronyd.service
```

10.9.3. Habilitando el repositorio de Openstack Mitaka

c) Se habilita a través de la orden yum.

```
# yum install centos-release-openstack-mitaka
```

c) Se actualiza el sistema con la orden siguiente.

```
# yum upgrade
```

d) Se instala el cliente.

```
# yum install python-openstackclient
```

10.10. Instalación del servidor de base de datos MYSQL en Control

a) Se instala la base de datos MySql así como sus dependencias.

```
# yum install mariadb mariadb-server python2-PyMySQL
```

b) Se edita el fichero ***/etc/my.cnf/openstack.cnf*** y se modifica añadiendo lo siguiente:

```
[mysqld]
...
bind-address = 192.168.100.11
default-storage-engine = innodb
 innodb_file_per_table max_connections = 4096
 collation-server = utf8_general_ci
 character-set-server = utf8
```

En primer lugar, se asocia la base de datos al servidor correspondiente, en segundo lugar se establece el motor de base de datos junto con todos los parámetros correspondientes.

c) Se inicia el servicio de base de datos y también se inicia la configuración con:

```
# systemctl enable mariadb.service
# systemctl start mariadb.service
# mysql_install_db
```

d) A través de la interfaz de texto se configura la base de datos.

En este caso se ha configurado de la siguiente manera:
Usuarios Anónimos => NO
Deshabilitar el Login del Root de forma Remota => SI
Borrar base de datos de Test => SI
Refrescar Privilegios => SI

10.11. Instalación base de datos NoSQL

A continuación, se instala el motor de base de datos NoSQL para que el servicio de telemetría pueda almacenar la información

a) Se instala todo el sistema de plugins de centOS.

```
# yum install mongodb-server mongodb
```

b) Se configura el archivo /etc/mongod.conf.

bind_ip = 10.0.0.11

smallfiles = true

c) Se agregan las utilidades de openstack.

```
# systemctl enable mongod.services
# systemctl start mongod.service
```

10.12. Instalación y configuración del servidor de mensajes RabbitMQ en Control

a) Se instala el servicio Qpid y se arranca al inicio con el sistema operativo.

```
# yum install rabbitmq-server
# systemctl enable rabbitmq-server.service
# systemctl start rabbitmq-server.service
# rabbitmqctl add_user openstack RABBIT_PASS
# rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

10.13. Instalación y configuración del servicio memcached

Memcached es empleado para el almacenamiento en caché de datos u objetos en la memoria RAM, reduciendo así las necesidades de acceso a un origen de datos externo (como una base de datos o una API).

Memcached tiene versiones para Linux, Windows y MacOS y se distribuye bajo licencia de software libre permisiva.

Se instala el servicio memcached y se arranca al inicio con el sistema operativo.

```
# yum install memcached python-memcached
# systemctl enable memcached.service
# systemctl start memcached.service
```

10.14. Configuración del servicio de identificación (Keystone) en Control

a) Se instala el servicio *keystone* en el nodo Control junto con la dependencia python-keystoneclient.

```
# yum install openstack-keystone httpd mod_wsgi
```

b) Como el servicio *keystone* necesita configurar el fichero **/etc/keystone/keystone.conf,** se configura una base de datos para guardar información, necesitando establecer la localización, usuario y password de la base de datos.

c) Se crea la tabla que utilizará el servicio *keystone.*

```
# su -s /bin/sh -c "keystone-manage db_sync" keystone
```

d) Se inician las llaves Fernet.

```
# keystone-manage fernet_setup --keystone-user keystone --
keystone-group keystone
```

e) Se configura el servidor APACHE HTTP editando el fichero **/etc/httpd/conf/httpd.conf** y es creado el archivo de configuración **/etc/httpd/conf.d/wsgi-keystone.conf**

f) Se arranca el servicio *keystone* para que se ejecute al comienzo junto con el sistema operativo.

```
# systemctl enable httpd.service
# systemctl start httpd.service
```

g) Se crean los servicios y los API endpoints.

```
# Openstack service créate --name keystone --description
"OpenStack Identity" identity
# openstack endpoint create --region RegionOne identity
public http://controller:5000/v3
# openstack endpoint create --region RegionOne identity
internal http://controller:5000/v3
# openstack endpoint create --region RegionOne identity
admin                          http://controller:35357/v3
```

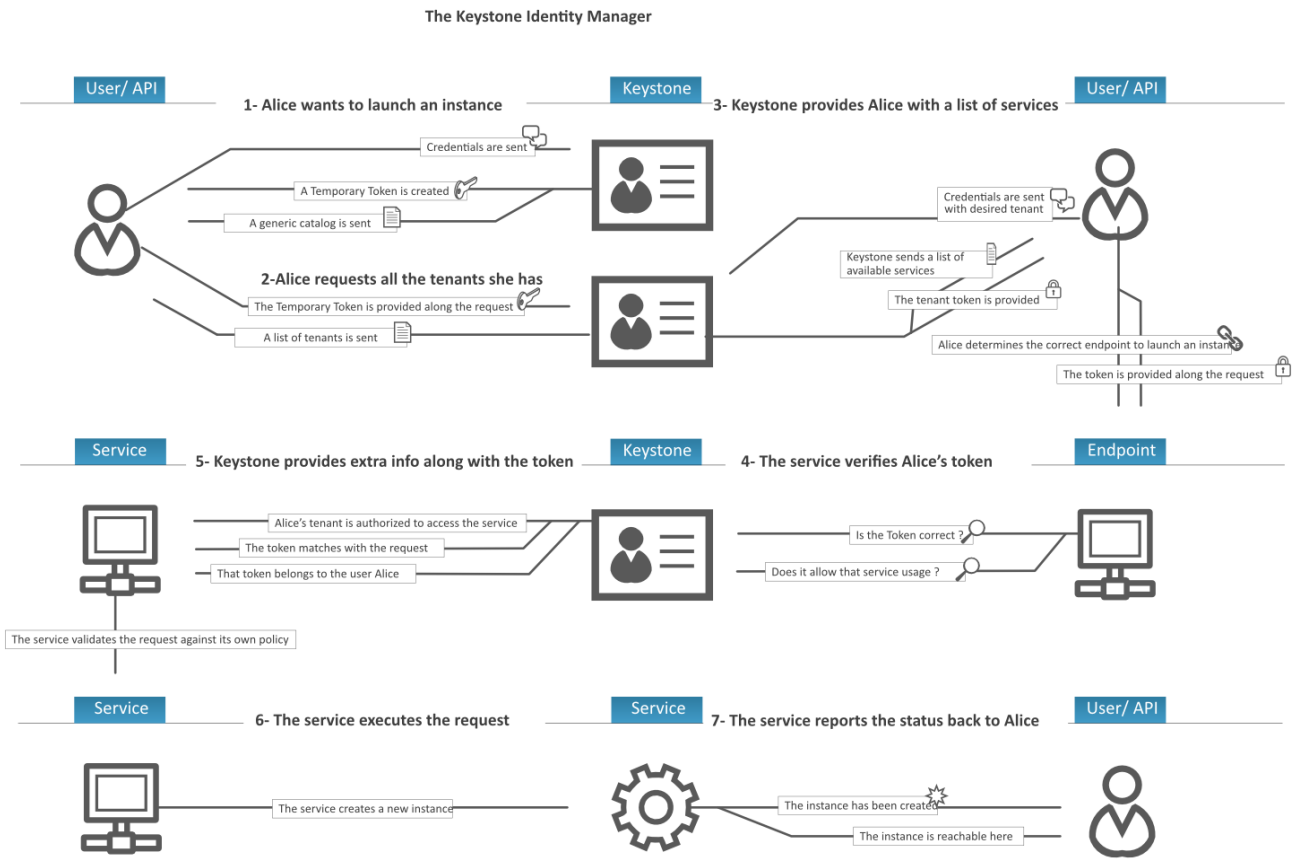


Figure 19 - Algoritmo de validación de keystone (http://www.dbigcloud.com/cloud-computing/170-openstack-desde-cero-keystone.html)

10.15. Configuración del servicio de imágenes (Glance) en Control

a) Se crea la base de datos que usará el servicio de imágenes *glance*.

```
# mysql -u root -p

# CREATE DATABASE glance;

# GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' \
IDENTIFIED BY 'GLANCE_DBPASS';

# GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' \
IDENTIFIED BY ' GLANCE_DBPASS'';
```

b) Se crean las credenciales para el servicio *glance* con respecto al servicio *identity.* Para ello deberemos de tener privilegios de administrador.

```
# openstack user create --domain default --password-prompt
glance

# openstack role add --project service --user glance admin

# openstack service create --name glance --description
"OpenStack Image" image
```

c) Se crea el *endpoint* del servicio *identity.*

```
# openstack endpoint create --region RegionOne image public
http://controller:9292

# openstack endpoint create --region RegionOne image
internal http://controller:9292

# openstack endpoint create --region RegionOne image admin
http://controller:9292
```

d) Se instala el servicio *Glance.*

```
# yum install openstack-glance
```

e) Se edita el fichero ***/etc/glance/glance-api.conf*** *y se adjunta el archivo final a este documento.*

f) Se edita el fichero ***/etc/glance/glance-registry.conf*** y se adjunta *el archivo final a este documento.*

g) Se configura la tabla en la base de datos del servicio glance con la siguiente orden:

```
# su -s /bin/sh -c "glance-manage db_sync" glance
```

h) Se reinicia del demonio y se configura para que al iniciarse el sistema permanezcan los mismos parámetros.

```
# systemctl enable openstack-glance-api.service openstack-glance-registry.service
# systemctl start openstack-glance-api.service openstack-glance-registry.service4
```

10.15.2. Creación de la imagen CirrOS

Para la creación de imágenes, en el proyecto Glance es necesario ejecutar la orden glance image-create.

A continuación, se detallan algunos de sus parámetros:

- name: nombre de la imagen.
- disk-format: formato del disco que contiene la imagen (ami,ari,aki,vhd,,raw,qcow2,vdi y iso).
- container-format: el formato de la propia imagen (ami,ari,aki,bare y ovf).
- owner: el proyecto que albergará la imagen.
- size: Tamaño de la imagen.
- min-disk: tamaño mínimo de disco para arrancar la imagen.
- min-ram: tamaño mínimo de memoria RAM
- File: fichero local que contiene la imagen.
- is-public: imagen accesible a todos los proyectos.
- is-protected: imagen protegida contra el borrado.

La creación de la imagen CirrOs se debe a que es una distribución que pesa muy poco, necesitando de muy poca memoria RAM y que nos viene muy bien para hacer pruebas iniciales con las máquinas virtuales.

Para la creación de la imagen cirrOS se han seguido los siguientes pasos:

a) Se descarga la imagen del repositorio fedora con la siguiente orden:

```
# wget http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x86_64-disk.img
```

b) Se crea la imagen en el proyecto *Glance* con la orden:

```
# openstack image create "cirros" --file cirros-0.3.4-x86_64-disk.img --disk-format qcow2 --container-format bare --public
```

10.15.3. Creación de la imagen Fedora

Se ha creado una imagen Fedora debido a que es una distribución para uso cotidiano, que únicamente necesita de 1GB de memoria RAM y 10Gb de espacio de disco, por lo que la hace atractiva para el catálogo de imágenes.

Para la creación de la imagen Fedora se han seguido los siguientes pasos:

a) Se descarga la imagen del repositorio fedora con la siguiente orden:

```
# wget
http://download.fedoraproject.org/pub/fedora/linux/updates/20/Images/x86_64/Fedora-x86_64-20-20140407-sda.qcow2
```

b) Se crea la imagen en el proyecto Glance con la orden:

```
# openstack image create " Fedora " --file Fedora.img --disk-format qcow2 --container-format bare --public
```

10.15.4. Creación de la imagen CentOS

La distribución CentOS es derivada de Red Hat Enterprise Linux (RHEL) por lo que al ser el proyecto OpenStack creado por la comunidad Red Hat es de obligación incluir esta versión en el catálogo de imágenes.

Otra razón de peso para incluir esta imagen es que es una distribución de uso muy frecuente en servidores, así como en el ámbito estudiantil, por lo que se considera que debe estar entre el catálogo de imágenes a ofrecer al usuario.

Para ello se necesitan unos requisitos mínimos que se describen a continuación:

Instalación mínima

- Memoria RAM = 64Mb
- Disco Duro = 1014Mb – 2Gb (Recomendado)

Instalación con escritorio.

- Memoria RAM = 2Gb
- Disco Duro = 20Gb – 40Gb (Recomendados)

Por lo tanto, procedemos a crear las diferentes imágenes:

a) Se crean las imágenes de las distintas versiones en el proyecto Glance con la orden:

```
# Openstack image create ‘CentOS 7 x86_64' --disk-format qcow2 --container-format bare --is-public true --copy-from http://cloud.fedoraproject.org/fedora-20.x86_64.qcow2
```

```
# Openstack image create --name ‘CentOS 6.5 x86_64' --disk-format qcow2 --container-format bare --is-public true --copy-from http://cloud.centos.org/centos/6.5/images/CentOS-6-x86_64-GenericCloud-20140929_01.qcow2
```

10.15.5. Creación de una imagen personalizada de CentOS

También se ha investigado la creación manual de imágenes, es decir, personalizar una imagen y ofrecerla a los usuarios.

El procedimiento usado para crear una imagen manualmente ha sido el siguiente:

a) Se crea una máquina virtual que contenga una distribución de centOS mínima previamente bajándonos la imagen CentOS de sus repositorios.

b) Se configura el paquete cloud-init con las siguientes órdenes:

```
# yum install http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
#yum install cloud-init
```

c) Se deshabilita la ruta ZEROCONF para que la instancia pueda acceder a los metadatos con la orden:

```
# echo "NOZEROCONF=yes" >> /etc/sysconfig/network
```

d) Se borra todo lo referente a la dirección MAC de la tarjeta de red ya que cada vez que OpenStack lanza una instancia tendra una *MAC address* diferente con la orden:

```
# virt-sysprep –d centos-7
```

.

10.16. Configuración del servicio de computo (Nova) en Control

a) Se crea la base de datos que usará nova

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

b) Se crea el usuario nova para autenticar en el servicio keystone:

```
# openstack user create --domain default --password-prompt nova
```

c) Se añade el Role de Administrador al usuario nova:

```
# openstack role add --project service --user nova admin
```

d) Se crean los API endpoints:

```
# openstack endpoint create --region RegionOne compute public http://controller:8774/v2.1/%\(tenant_id\)s
# openstack endpoint create --region RegionOne compute internal http://controller:8774/v2.1/%\(tenant_id\)s
# openstack endpoint create --region RegionOne compute admin http://controller:8774/v2.1/%\(tenant_id\)s
```

e) Se instalan los paquetes necesarios para el nodo de Control:

```
# yum install openstack-nova-api openstack-nova-conductor openstack-nova-console openstack-nova-novncproxy openstack-nova-scheduler
```

f) Se configura el archivo **/etc/nova/nova.conf** y se adjunta como documento anexo a este archivo.

g) Se rellena y se crean las tablas:

```
# su -s /bin/sh -c "nova-manage api_db sync" nova
# su -s /bin/sh -c "nova-manage db sync" nova
```

h) Se arrancan todos los servicios de Computo:

```
# systemctl enable openstack-nova-api.service openstack-nova-consoleauth.service openstack-nova-scheduler.service openstack-nova-conductor.service openstack-nova-novncproxy.service
# systemctl start openstack-nova-api.service openstack-nova-consoleauth.service openstack-nova-scheduler.service openstack-nova-conductor.service openstack-nova-novncproxy.service
```

10.17. Configuración del servicio de computo (Nova) en computo

a) Se instalan los paquetes necesarios para el nodo de Control:

```
# yum install openstack-nova-compute
```

b) Se edita el archivo ***/etc/nova/nova.conf*** *y se adjunta a este archivo.*

c) Se inicia el servicio de Computo y sus dependencias.

```
# systemctl enable libvirtd.service openstack-nova-compute.service
# systemctl start libvirtd.service openstack-nova-compute.service
```

10.18. Configuración del servicio de red (Legacy Service) en Control

a) Se configura la base de datos:

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

b) Se crea el usuario neutron para autenticar con el proyecto Keystone:

```
# openstack user create --domain default --password-prompt neutron
```

c) Se asocia el usuario al rol de administrador con la orden:

```
# openstack role add --project service --user neutron adminº
```

d) Se crea el servicio y las API endpoints con las siguientes órdenes:

```
# openstack service create --name neutron --description "OpenStack Networking" network
# openstack endpoint create --region RegionOne network public http://controller:9696
# openstack endpoint create --region RegionOne network internal http://controller:9696
# openstack endpoint create --region RegionOne network admin http://controller:9696
```

e) Se instalan y se procede a configurar los siguientes servicios con la orden:

```
# yum install openstack-neutron openstack-neutron-ml2 openstack-neutron-linuxbridge ebtables
```

f) Se configura el archivo **/etc/neutron/neutron.conf** y se adjunta el archivo final a este documento.

g) Se el archivo **/etc/neutron/plugins/ml2/ml2_conf.ini** y se adjunta el archivo final a este documento.

h) Se configura el archivo **/etc/neutron/plugins/ml2/linuxbridge_agent.ini** y se adjunta el archivo final a este documento.

i) Se configura el archivo **/etc/neutron/dhcp_agent.ini** y se adjunta el archivo final a este documento.

j) Se configura el archivo **/etc/neutron/metadata_agent.ini** y se adjunta el archivo final a este documento.

k) Se configura el archivo **/etc/nova/nova.conf** y se adjunta el archivo final a este documento.

l) Se finaliza la configuración creando un enlace simbólico y rellenando las tablas de la base de datos con las siguientes órdenes:

```
# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini
```

```
# su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf --config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade head" neutro
```

m) Por último, se establece el arranque de los demonio al iniciarse el sistema con:

```
# systemctl restart openstack-nova-api.service
```

```
# systemctl enable neutron-server.service neutron-linuxbridge-agent.service neutron-dhcp-agent.service neutron-metadata-agent.service
```

```
#systemctl start neutron-server.service neutron-linuxbridge-agent.service neutron-dhcp-agent.service neutron-metadata-agent.service
```

10.19. Configuración del servicio de red (Legacy Service) en Computo

a) Se instalan las librerias necesarias con:

```
# yum install openstack-neutron-linuxbridge ebtables ipset
```

b) Se configura el archivo ***/etc/neutron/neutron.conf*** y se adjunta el archivo final a este documento

c) Se configura **/etc/neutron/plugins/ml2/ linuxbridge_agent.ini** y se adjunta el archivo final a este documento

d) Se configura el archivo **/etc/nova/nova.conf** y se adjunta el archivo final a este documento

e) Se reinicia el servicio, sus dependencias y se hace que el servicio arranque con el sistema.

```
# systemctl restart openstack-nova-compute.service
# systemctl enable neutron-linuxbridge-agent.service
# systemctl start neutron-linuxbridge-agent.service
```

10.20. Configuración del servicio de dashboard (Dashboard Service) en Control

a. Se instala el módulo Dashboard junto con las dependencias:

```
# yum install openstack-dashboard
```

b. Se configura el archivo ***/etc/openstack-dashboard/local_settings*** *y se adjunta el archivo a este documento.*

c. Se inicia el servicio:

```
# systemctl restart httpd.service memcached.service
```

d. Se comprueba que la URL **http://controller/dashboard** funciona.

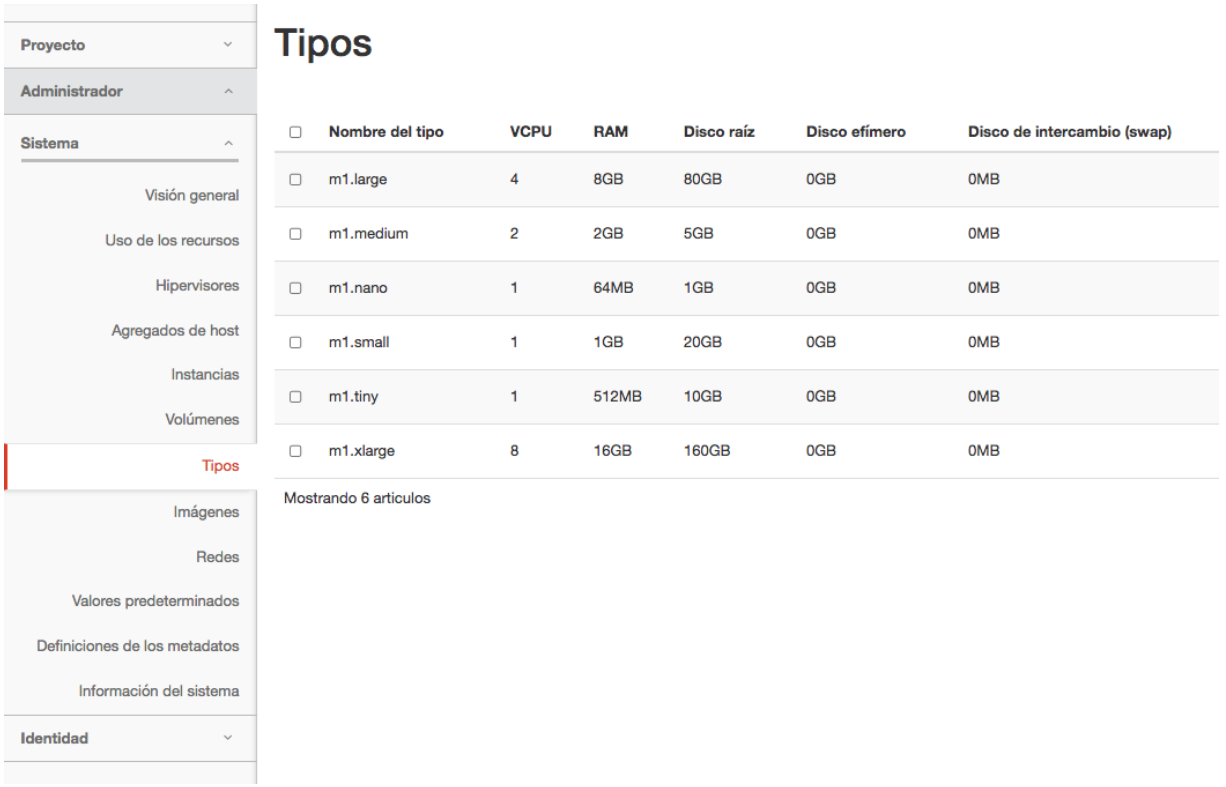


Ilustración 1 - Interfaz Visual OpenStack

10.21 Configuración del servicio de bloques (Cinder) en Control

a) Se crea la base de datos que utilizará el Cinder y le se concende al usuario permisos de administrador.

```
# mysql -u root –p

# mysql> CREATE DATABASE cinder;

# mysql> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' \ IDENTIFIED BY 'CINDER_DBPASS';

# mysql> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' \
IDENTIFIED BY 'CINDER_DBPASS';
```

a) Se crea el usuario para que Cinder se pueda autenticar en el servicio keystone.

```
# openstack user create --domain default --password-prompt cinder
```

b) Se asocia el usuario al rol de administrador:

```
# openstack role add --project service --user cinder admin
```

c) Se crean los servicios y los API Endpoints del proyecto CINDER con:

```
# openstack service create --name cinder --description "OpenStack Block Storage" volume

# openstack service create --name cinderv2 --description "OpenStack Block Storage" volumev2

# openstack endpoint create --region RegionOne \ volume public http://controller:8776/v1/%\(tenant_id\)s

# openstack endpoint create --region RegionOne \ volume internal http://controller:8776/v1/%\(tenant_id\)s

# openstack endpoint create --region RegionOne \ volume admin http://controller:8776/v1/%\(tenant_id\)s

# openstack endpoint create --region RegionOne \ volumev2 public http://controller:8776/v2/%\(tenant_id\)s

# openstack endpoint create --region RegionOne \ volumev2 internal http://controller:8776/v2/%\(tenant_id\)s

# openstack endpoint create --region RegionOne \ volumev2 admin http://controller:8776/v2/%\(tenant_id\)s
```

d) Para instalar el servicio Cinder se utiliza la siguiente orden:

```
# yum install openstack-cinder
```

e) Se configura el archivo ***/etc/cinder/cinder.conf*** y se añade el mismo a este documento

f) Se crean las tablas para el servicio Cinder:

```
# su -s /bin/sh -c "cinder-manage db sync" cinder
```

g) Se configura el archivo ***/etc/nova/nova.conf*** y se anexa a este documento.

h) Se inician los servicios y todos los módulos asociados:

```
# systemctl restart openstack-nova-api.service
# systemctl enable openstack-cinder-api.service openstack-cinder-scheduler.service
# systemctl start openstack-cinder-api.service openstack-cinder-scheduler.service
```

10.22 Configuración del servicio de bloques (Cinder) en NBloque1

a) Para configurar el servicio de bloques se tiene que utilizar LVM (Logical Volume Manager) para ello se instala la librería para trabajar con ello con la órden:

```
# yum install lvm2
```

```
# systemctl enable lvm2-lvmetad.service
```

```
# systemctl start lvm2-lvmetad.service
```

b) Se crean los volúmenes lógicos y los físicos en un disco duro secundario en `/dev/sdb:`

```
# pvcreate /dev/sdc
```

```
# vgcreate cinder-volumes /dev/sdc
```

c) Se añade una entrada para filtrar los dispositivos que usan las máquinas virtuales en el fichero ***/etc/lvm/lvm.conf.***

```
devices {
     ...
        filter = [ "a/sda1/", "a/sdc/", "r/.*/"]
     ...
}
```

d) Se instala el servicio *Cinder* en NBloque1:

```
# yum install openstack-cinder targetcli python-keystone
```

e) Se configura el archivo ***/etc/cinder/cinder.conf*** y se anexa el mismo a este documento.

f) Se arranca el servicio:

```
# systemctl enable openstack-cinder-volume.service
target.service
```

```
# systemctl start openstack-cinder-volume.service
target.service
```

10.23 Configuración del servicio de telemetría (Telemetry Service) en Control

Pre-requisitos:

Para guardar toda la información se necesita usar la base de datos NOSQL instalada con anterioridad. Por lo tanto se configura el acceso con:

```
# mongo --host controller --eval ' db = db.getSiblingDB("ceilometer"); db.createUser({user: "ceilometer", pwd: "CEILOMETER_DBPASS", roles: [ "readWrite", "dbAdmin" ]})'
```

Se realizan los siguientes pasos para configurar openstack:

a) Se crea el usuario CEILOMETER con:

```
# openstack user create --domain default --password-prompt ceilometer
```

b) Se asocia el usuario al rol de administrador:

```
# openstack role add --project service --user ceilometer admin
```

c) Se crea los servicios y los API endpoints con:

```
# openstack service create --name ceilometer \ --description "Telemetry" metering
# openstack endpoint create --region RegionOne \ metering public http://controller:8777
# openstack endpoint create --region RegionOne \ metering internal http://controller:8777
# openstack endpoint create --region RegionOne \ metering admin http://controller:8777
```

d) Se instalan las librerías en el nodo de control con la orden:

```
# yum install openstack-ceilometer-api openstack-ceilometer-collector openstack-ceilometer-notification openstack-ceilometer-central python-ceilometerclient
```

e) Se configura el archivo ***/etc/ceilometer/ceilometer.conf*** y se anexa a este documento.

f) Se inician los demonios correspondientes con las órdenes:

```
# systemctl enable openstack-ceilometer-api.service
openstack-ceilometer-notification.service openstack-
ceilometer-central.service openstack-ceilometer-
collector.service

# systemctl start openstack-ceilometer-api.service \
openstack-ceilometer-notification.service openstack-
ceilometer-central.service openstack-ceilometer-
collector.service
```

10.24 Telemetry Service - Estadísticas sobre PROYECTO GLANCE

a) Se configura el archivo ***/etc/glance/glance-api.conf*** *y se anexa el fichero a este documento*

b) Configuramos el archivo ***/etc/glance/glance-registry.conf*** *y se anexa el fichero a este documento*

c) Se reinicia el demonio:

```
# systemctl restart openstack-glance-api.service openstack-glance-registry.service
```

10.25 Telemetry Service - Estadísticas sobre PROYECTO COMPUTE

a) Se instala la librería adecuada para ello con la orden:

```
# yum install openstack-ceilometer-compute python-ceilometerclient python-pecan
```

b) Configuramos el archivo ***/etc/ceilometer/ceilometer.conf*** y se anexa a este documento el fichero final.

c) Configuramos el archivo ***/etc/nova/nova.conf*** y se anexa a este documento el fichero final.

d) Se reinicia el demonio

```
# systemctl enable openstack-ceilometer-compute.service
systemctl start # openstack-ceilometer-compute.service

# systemctl restart openstack-nova-compute.service
```

a) Se configura el archivo ***/etc/cinder/cinder.conf*** *y se anexa a este documento el fichero final.*

b) Se reinicia el demonio:

```
# systemctl restart openstack-cinder-api.service openstack-cinder-scheduler.service
# systemctl restart openstack-cinder-volume.service
```

10.27 Telemetry Service – Estadísticas sobre Proyecto ALARMING

Prerrequisitos:

a. Se crea la base de datos (en este caso MySQL) y se le da privilegios :

```
# mysql –u root –p
# CREATE DATABASE aodh;
# GRANT ALL PRIVILEGES ON aodh.* TO ‘aodh’@’localhost’ \
IDENTIFIED BY ‘AODH_DBPASS’;
# GRANT ALL PRIVILEGES ON aodh.* TO ‘aodh’@’%’ \ IDENTIFIED BY
‘AODH_DBPASS’;
```

b. Se configura Openstack para que acepte el nuevo módulo:

```
# openstack user create –domain default \ --password-
prompt aodh

# openstack role add –project service –user aodh admin
```

c. Se crea el servicio y los API endpoints:

```
# openstack service create –name aodh \ --description
“Telemetry” alarming

# openstack endpoint create –region RegionOne \ alarming
public http://controller:8042

# openstack endpoint create –region RegionOne \ alarming
internal http://controller:8042

# openstack endpoint create –region RegionOne \ alarming admin
http://controller:8042
```

d. Se configura el archivo ***/etc/aodh/aodh.conf*** y se anexa como parte de este documento.

e. Se rellena la base de datos con la orden:

```
# su –s /bin/sh –c “aodh-dbsync” aodh
```

f. Se inician los servicios:

```
# systemctl enable openstack-aodh-api.service openstack-
aodh-evaluator.service openstack-aodh-notifier.service
openstack-aodh-listener.service

# systemctl start openstack-aodh-api.service openstack-aodh-
evaluator.service openstack-aodh-notifier.service openstack-
aodh-listener.service
```

11. Leyes aplicables al ámbito del Trabajo de Fin de Máster

En este apartado se procede a enumerar y explicar las leyes o normativas que se tienen que aplicar en un centro de datos:

11.1 Ley Orgánica de Protección de Datos

Según la AGENCIA ESPAÑOLA DE PROTECCIÓN DE DATOS (AEPD) la relación con el prestador de servicios estará sometida a la Ley Orgánica de Protección de Datos de carácter personal

Fuente:

https://www.agpd.es/portalwebAGPD/canaldocumentacion/publicaciones/common/Guias/GUIA_Cloud.pdf

Si el prestador de servicios va a prestar los servicios a administraciones públicas además deberá cumplir el **Esquema Nacional de Seguridad (ENS)**, este determina la política de seguridad que se ha de aplicar en la utilización de los medios electrónicos.

Además del **Esquema Nacional de Interoperabilidad (ENI)**, que establece los principios y directrices en el intercambio y conservación de la información electrónica por parte de la administración pública.

¿En que afecta la LOPD a nuestro trabajo?
En el trabajo presente se verá afectado en relación a los datos de carácter privado que tenga la infraestructura.

Por ejemplo, si tenemos una máquina virtual que almacene en una base de datos características raciales, religión etc.… no será lo mismo que si tenemos un simple servidor para una página web informativa.

En el caso albergar datos privados de administraciones públicas como nóminas, salarios, empleados, documentos de catastro etc.… nos veremos regidos por el Esquema Nacional de Seguridad.

11.2 Ley de Servicios de la Sociedad de la Información y Comercio Electrónico (LSSII)

Esta ley se debe cumplir en nuestro proyecto siempre y cuando el prestador de servicios actúe como intermediario en la transacción de contenidos sean o no remunerados.

Por lo tanto, como el usuario está percibiendo un servicio por vía electrónica este trabajo está sujeto a dicha ley.

12 Resultados, conclusiones y trabajos futuros.

12.1 Resultados.

Al finalizar este Trabajo de Fin de Máster podemos concluir como resultados los siguientes ejemplos creados a partir de la infraestructura desarrollada teniendo en cuenta la limitación del servidor de cómputo a 8Gb de Memoria RAM:

Ejemplo 1 – Proyecto LabCirrOS

Tal y como muestra la Figura 20 tenemos un laboratorio docente con 10 máquinas virtuales con sistema operativo *CirrOS* cada uno con 1 núcleo, 512Mb de RAM y 1Gb de Disco Duro.

Nombre de la instancia	Nombre de la imagen	Dirección IP	Tamaño	Par de claves	Estado	Zona de disponibilidad	Tarea	Estado de energía	Tiempo desde su creación	Acciones
p-10	cirros	192.168.100.203	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-9	cirros	192.168.100.206	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-8	cirros	192.168.100.205	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-7	cirros	192.168.100.202	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-6	cirros	192.168.100.201	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-5	cirros	192.168.100.204	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-4	cirros	192.168.100.225	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-3	cirros	192.168.100.200	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-2	cirros	192.168.100.224	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
p-1	cirros	192.168.100.223	m1.tiny	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea

Mostrando 10 articulos

Figura 20 – Laboratorio docente para prácticas con sistema operativo CirrOS

Ejemplo 2 – Proyecto LabCentOS

Tal y como muestra la Figura 21 tenemos un laboratorio docente con 2 máquinas virtuales con sistema operativo CentOS cada uno con 2 núcleos, 2Gb de RAM y 20Gb de Disco Duro.

Nombre de la instancia	Nombre de la imagen	Dirección IP	Tamaño	Par de claves	Estado	Zona de disponibilidad	Tarea	Estado de energía	Tiempo desde su creación	Acciones
centos-2	Centos6	192.168.100.212	m1.medium	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
centos-1	Centos6	192.168.100.211	m1.medium	mykey	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea

Mostrando 2 articulos

Figura 21 – Laboratorio docente para prácticas con sistema operativo CentOS

Ejemplo 3 – Proyecto LabFedora

Tal y como muestra la Figura 22 tenemos un laboratorio docente con 2 máquinas virtuales con sistema operativo Fedora cada uno con 1 núcleo, 1Gb de RAM y 20Gb de Disco Duro.

	Nombre de la instancia	Nombre de la imagen	Dirección IP	Tamaño	Par de claves	Estado	Zona de disponibilidad	Tarea	Estado de energía	Tiempo desde su creación	Acciones
☐	fedora-2	Fedora	192.168.100.200	m1.small	Fedora	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea
☐	fedora-1	Fedora	192.168.100.222	m1.small	Fedora	Activa	nova	Ninguno	Ejecutando	0 minutos	Crear instantánea

Mostrando 2 articulos

Figura 22 – Laboratorio docente para prácticas con sistema operativo Fedora

Ejemplo 4 – Servidor + Balanceador de Carga + Motor de BBDD

Tal y como muestra la Figura 23 tenemos un esquema de infraestructura web donde tenemos 2 servidores web (1 principal y otro de respaldo), un balanceador de carga y un motor de base de datos MYSQL

La infraestructura no está configurada según el ejemplo pues no es objetivo de este TFM aprender a configurar una infraestructura del estilo.

También hemos elegido a propósito diferentes sistemas operativos para mostrar que tenemos libertad de elección, incluso adquiriendo un sistema operativo personalizado y configurarlo como una imagen *cloud*.

	Nombre de la instancia	Nombre de la imagen	Dirección IP	Tamaño	Par de claves	Estado
☐	ServidorApacheRespaldo	Ubuntu16.04	192.168.100.208	m1.medium	mykey	Activa
☐	MySql	Fedora	192.168.100.206	m1.tiny	Fedora	Activa
☐	LoadBalancer	cirros	192.168.100.205	m1.tiny	mykey	Activa
☐	ServidorApache	Ubuntu16.04	192.168.100.204	m1.small	mykey	Activa

Mostrando 4 articulos

Figura 23 – Laboratorio docente para prácticas con sistema operativo Fedora

12.2 Conclusiones

El primer contacto fue con el Trabajo de Fin de Grado y se pudo comprobar que **crear y mantener una infraestructura de este tipo no es algo trivial** y que se necesita tener los conceptos muy claros.

Con respecto a este segundo contacto se ha podido comprobar que adaptar un sistema distribuido a una plataforma como es la de la escuela en la cual dependemos de otros no es sencillo al tener restricciones debido a que parte de la infraestructura utilizada se emplea en el día a día de la actividad docente que se realiza en los laboratorios.

Se ha podido realizar una instalación y configuración de *OpenStack* tratándose de una plataforma novedosa y que crece exponencialmente en uso en el campo de la computación en la nube.

En su estado final, el trabajo se compone de 3 máquinas físicas (Control, cómputo y nbloque1) interconectadas que **pueden ofrecer servicios de infraestructura como servicio**.

Se ha estudiado una **plataforma viable** que se puede implementar en los laboratorios docentes donde se puede **acceder a máquinas análogas a las utilizadas por los alumnos en las prácticas de forma remota, ininterrumpida, sin restricciones de horario** teóricamente.

A través de una interfaz web se puede ofrecer **diversas opciones de cómputo** para usuarios, así como diferentes tamaños de memorias a elegir dependiendo del trabajo a acometer.

Además, se ofrecen **diferentes imágenes** que arrancan **diferentes sistemas operativos** (CirrOS, Fedora, Centos e imágenes personalizadas) según la preferencia del usuario.

En este primer contacto **no se ha podido sacar conclusiones de rendimiento** interno de cada máquina, pero si en general a través del módulo de telemetría, pudiéndose comprobar como **las instancias se ejecutaban simultáneamente**.

---Conclusiones Personales
La realización de este trabajo ha sido una grata **experiencia, creando, experimentando y manteniendo una nube privada** a pesar de la **abrumadora cantidad de documentación** que proporciona *OpenStack* y que se ha tenido que ir filtrando para desdeñar lo adecuado para este proyecto.

En este Trabajo de Fin de Master y en particular en la rama del Máster en Ingeniería Informática en su especialidad en computación en la nube puedo decir que con respecto a las otras ramas un ingeniero en esta especialidad debe tener conocimientos generales de las otras ramas y específicos en virtualización.

En este proyecto he tenido que utilizar conocimientos además de virtualización, sobre redes, sobre administración de sistemas, sobre programación, etc...

Se puede afirmar que el trabajo previo de investigación realizado es un trabajo adicional ya que cada nube es diferente, **cada una de ellas tiene sus propias y específicas condiciones de red, cómputo y almacenamiento**.

Iniciarse en la plataforma *OpenStack* no es sumamente complicado una vez se han adquirido previamente conocimientos básicos de computación en la nube.

Se puede decir **que la plataforma *OpenStack* sobre el papel es válida para implementar una infraestructura que dé servicio** a los laboratorios docentes **proporcionando a cada alumno una máquina virtual análoga** a las usadas actualmente, la cual, podrá usar a su conveniencia con el fin de lograr los objetivos de cada asignatura

12.3 Trabajos futuros

Se propone como trabajo futuro mejorar la mantenibilidad del sistema y la documentación exhaustiva del mismo ya que solucionar un simple error en este tipo de sistema puede ser una ardua tarea si no se tienen experiencia y conocimiento en este campo.

Otro trabajo que se podría realizar en ampliar la infraestructura de la nuble privada desarrollada incorporando más nodos de cómputo con el fin de poder soportar un laboratorio de tamaño medio (25 puestos de trabajo).

La creación de tareas personalizadas para la escuela también, tales como: Implementación de un laboratorio con un solo clic, control de cambios sobre máquinas virtuales para evaluación del alumnado, actualizaciones de laboratorios automatizadas sería una buena vía de exploración. (Inclusión de nuevos módulos personalizados para el mejor y más cómodo control del sistema).

12.4 Propuesta de Implementación de Nube para servicios en Asignatura con 25 puestos de trabajo.

Las necesidades para esos 25 puestos serían las siguientes:

- Memoria RAM: 50Gb
- Nº de núcleos: 25
- Espacio de disco= 250Gb
- 3 redes (data, management y una red externa)

Por lo que para acometer esta tarea se necesita:

- **1 servidor de Control**:
    - 2 interfaces de red
    - Procesador con al menos 4 núcleos.
    - 2Gb de memoria RAM
    - 250gb de disco duro
- Debido a que se necesita 50Gb de memoria RAM y 25 núcleos se va a necesitar **7 servidores de cómputo** que tengan cada uno 4 núcleos y 8Gb de RAM.
- 1 nodo de almacenamiento de bloques que contenga 250Gb de espacio de disco.

Para la red se necesita lo siguiente:

- **SWITCH** central que lleve un cable a cada terminal y otro cable al switch donde está toda la estructura implementada con esto se logra que ya estuvieran todo en la misma red y por lo tanto tuvieran conectividad desde la misma red.
- **Imagen personalizada de la asignatura en cuestión.**

Con respecto a las licencias a utilizar siempre y cuando se utilicen **como sistema operativo los de tipo código abierto no se necesita ninguna**.

13 Fuentes de información

1) http://www.aitinc.net

2) https://es.wikipedia.org/wiki/Computaci%C3%B3n_en_la_nube#/media/File:Cloud_computing-es.svg

3) https://diarioti.com

4) http://blogs.salleurl.edu

5) http://www.itsteziutlan.edu.mx

6) http://aws.amazon.com/es/ec2/

7) http://www.vmware.com/es/virtualization

8) www.openstack.org

9) https://opennebula.org/about/project/

10) https://docs.eucalyptus.com/eucalyptus/latest/#install-guide/intro.html

11) https://www.revistacloudcomputing.com

12) http://www.dbigcloud.com/cloud-computing/170-openstack-desde-cero-keystone.html

13) http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=820:introduccion-a-la-nube-open-nebula-como-caso-de-exito&catid=108:blog-cenatic&Itemid=150

14) *"OpenStack Installation Guide. For Red Hat, CentOs and Fedora. Havana."* http://docs.openstack.org/havana/install-guide/install/yum/content/. OpenStack Foundation, 2014.

15) *"OpenStack End User Guide"*. http://docs.openstack.org/user- guide/content/. OpenStack Foundation, 2014.

16) *"OpenStack Virtual Machine Image Guide"*. http://docs.openstack.org/image-guide/content/. OpenStack Foundation, 2014.

17) Kevin Jackson. *"OpenStack Cloud Computing CookBook"*. Packt Publishing, 2012.

18) Tom Fifield, Diane Fleming, Anne Gentle, Lorin Hochstein, Jonathan Proulx, Everett Toews & Joe Topjian. *"OpenStack Operations Guide. Set up and manage your OpenStack Cloud"*. O'Reilly Edt, 2014.

19) Greg Schulz. "Cloud and Virtual data Storage Networking". CRC Press, 2012.

20) Barrie Sosinsky. ¿Qué es la nube? Anaya, 2012.

21) http://es.wikipedia.org/wiki/Computaci%C3%B3n_en_la_nube

22) http://aws.amazon.com/es/what-is-cloud-computing/?nc2=h_l2_cc/

23) http://www.vmware.com/es/virtualization

24) Turban, E; King, D; Lee, J; Viehland, D (2008). «Chapter 19: Building E-Commerce Applications and Infrastructure». *Electronic Commerce A Managerial Perspective* (5th edición). Prentice-Hall. p. 27.

25) "The NIST Definition of Cloud Computing". National Institute of Standards and Technology. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

26) https://cloudstack.apache.org/about.html

14 ANEXOS

Estos anexos se presentan como dudas que han surgido durante el Trabajo de Fin de Máster y aunque no se encuentran en el propio trabajo dada la curiosidad hemos investigado el proceso y por lo tanto se dejaran aquí planteadas las dudas y sus "posibles" soluciones:

14.1 Anexo 1 – Escalabilidad de Máquinas de Cómputo

¿Qué pasa si queremos tener un cierto número de máquinas virtuales, pero con el hardware actual no podemos?

Openstack en este caso nos da la posibilidad de aumentar el número de máquinas de cómputo.

Aunque dentro del mismo Openstack no existe demasiada documentación sobre esto, teóricamente la forma de escalar las máquinas de computo sería configurarlas de la misma manera que se ha configurado la de este Trabajo y la misma arquitectura se auto descubriría.

14.2 Anexo 2 - Acceso a Máquinas Virtuales

¿Como puedo acceder de forma remota a cada máquina virtual?

Cada máquina virtual tendría que tener un IP que fuera pública. Lo cual **tenemos una restricción** de IPs públicas es nuestra universidad.

Con respecto y específicamente para nuestro TFM la accesibilidad será de manera INTERNA, es decir, para acceder a las máquinas virtuales hay que acceder desde la red interna y cada máquina física tendrá que estar en la red 192.168.100.X

Tenemos una red 192.168.100.0 con máscara 255.255.255.0 es decir 255 direcciones IP que se organizan de la siguiente manera:

- 192.168.100.1 => IP de la red
- 192.168.100.11 => IP de la máquina de control
- 192.168.100.31 => Ip de la máquina de cómputo
- 192.168.100.71 => Ip de la máquina de bloques
- 192.168.100.200 a 192.168.100.225 => IPs de reparto del DHCP para máquinas virtuales.
- 192.168.100.255 => IP de Broadcast

Con esta infraestructura tendríamos 255-5-26= 224 IPs libres por lo que podrían haber 224 PCs conectados en red lo cual en un laboratorio de 25 terminales daría para 9 laboratorios.

Para escalar en número de terminales conectados en mi opinión la solución sería ampliar la máscara de red de /24 a una con más espacio.

Por ejemplo, a un /22 lo cual nos permitiría 2^10=1024IPs, lo que nos permitiría tener 1024 - 31 = 993 IPs libres.

De forma estructural los 3 nodos están conectados a un switch ese switch tendría que enviar un cable al switch de cada laboratorio y este a su vez a cada PC.