ESCUELA TÉCNICA SUPERIOR DE INGENIEROS
INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

GESTIÓN DE RED EN ENTORNOS CLOUD: DEMOSTRADOR SOBRE OPENSTACK Y SU INTEGRACIÓN CON LA PLATAFORMA NAGIOS

(Network Management in cloud enviroments: Demonstrator about Openstack and its integration with Nagios platform)

Para acceder al Titulo de

Graduado en
Ingeniería de Tecnologías de Telecomunicación

Autor: Pablo Rueda Ortega

Julio - 2020

E.T.S. DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACION

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por: Pablo Rueda Ortega
Director del TFG: José Angel Irastorza Teja y Alberto E. García Gutiérrez
Título: "Gestión de red en entornos Cloud: Demostrador sobre OpenStack y su integración con la plataforma Nagios"
Title: "Network Management in cloud enviroments: Demonstrator about Openstack and its integration with Nagios platform **"**

Presentado a examen el día:

para acceder al Título de

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente (Apellidos, Nombre): Roberto Sanz Gil
Secretario (Apellidos, Nombre): José Angel Irastorza Teja
Vocal (Apellidos, Nombre): Jesús Ibáñez Díaz

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente

Fdo.: El Secretario

Fdo.: El Vocal

Fdo.: El Director del TFG
(sólo si es distinto del Secretario)

Vº Bº del Subdirector

Trabajo Fin de Grado Nº
(a asignar por Secretaría)

AGRADECIMIENTOS

Quisiera agradecer a mis directores del trabajo, *Jose Ángel Irastorza* y *Alberto Eloy Garcia*, todo el apoyo brindado durante la realización del proyecto durante todo este tiempo. También a todos los profesores que me han impartido clase durante esta etapa en *ETSIIT* y por supuesto a mi familia y amigos sin los cuáles no estaría hoy aquí.

RESUMEN

A lo largo de los años se ha experimentado una evolución exponencial en todos los aspectos y esto también se traduce en el área tecnológica, donde ha aparecido una nueva herramienta o posibilidad de operar conocida como “nube” que permite liberar de carga física a muchas de las tareas que se tenían prácticamente como cotidianas. En este proyecto se abordará esta tecnología con gran potencial por exprimir y también se realizará la implementación de una red cloud en un entorno de laboratorio, integrándola diversas herramientas que ayuden a aumentar el potencial y las posibilidades que tiene la misma, con el objetivo de observar las diferentes posibilidades que tiene esta tecnología y ver que futuros usos se pueden obtener de ella. Por lo tanto se abordará generalmente toda la columna de esta tecnología, entrando específicamente en aspectos concretos que serán necesarios a la hora de realizar la implementación ya comentada.

Palabras Clave: Nube, Openstack, Cloud computing, Gestión de red.

ABSTRACT

Over the years, it has been experienced a huge evolution on every account and also in technological áreas, where it has appeared a new application known as “cloud” which allows to decrease physical load to many assignments. In this project will be treated this great technology and also will be performed the implementation of a cloud network in a lab environment, integrating it any software such a network management software to increase its potential in order to check the different possibilities this technology has and see any future uses that can get from it. Therefore, it will be addressed, generally all the key aspects of this technology, entering specifically in specific aspects that will be necessary at the time of making the implementation of the network.

Keywords: Cloud, Openstack, Cloud computing, Network management.

Índice

Índice de Figuras

Capítulo 1

Introducción

El avance tecnológico acontecido los últimos años ha supuesto que el nivel de redes existentes así cómo los dispositivos hardware que las componen hayan aumentado exponencialmente en número.

Esto supone que el siguiente paso a dar sea, la carga de espacio físico y mantenimiento entre otras cosas, de estos componentes físicos tenga que ser reducida, para así optimizar la economía empresarial, y también con el avance producido en los dispositivos móviles, ha creado que la sociedad tenga la capacidad de disponer en su dispositivo personal una gran cantidad de información que antes sería impensable.

Estos avances se dan lugar gracias a la creación de la nube, o a las redes cloud, que permiten liberar de esa carga física al usuario final, permitiendo poder ahorrar dinero en mantenimiento de equipo, en el caso de la empresa, o disponer de toda tu información, accesible en cualquier momento y desde cualquier dispositivo con Internet en el caso de una persona.

1.1. Motivación

Al ser esta tecnología, precoz en términos tecnológicos porque no lleva mucho tiempo en auge, se ha logrado asentar debido a sus grandes beneficios. Pero su mayor asentamiento ha sido a nivel de usuario, como las tecnologías que se usan prácticamente a diario ya sean *Dropbox* o *Google Drive* por mencionar dos servicios usados prácticamente por gran parte de la población con acceso a Internet.

Esto supone que surge la necesidad de enfocarse mas en potenciar aún mas su asentamiento en el dimensionamiento de redes a una escala mayor, integrándola con las herramientas de gestión de red, para exprimir lo máximo posible esta tecnología y obtener la mayor cantidad de beneficios que aporta.

Surge así la idea de buscar topologías de red alternativas dentro de estas redes cloud, que permitan un mayor entendimiento de las mismas y posibiliten tener la mayor información a la hora de gestionar la red.

1.2. Objetivos

Partiendo de la experiencia obtenida a partir del trabajo anterior [1] y empleada en este mismo proyecto, se pretende usar Openstack como proveedor Cloud, en este caso de IaaS, ofreciendo la infraestructura de red como servicio, e implementar una topología poco habitual de red.

El objetivo fundamental es obtener una red cloud pensada para ser usada en un entorno de laboratorio, una red que no exija mucha carga computacional, y poder trabajar con ella y modificarla según las necesidades que se tengan. Esta red debe ser estable y robusta y que sea fiable de que el funcionamiento es óptimo. Como resultado, la configuración final deberá tener una máquina virtual que haga las veces de router ya que esto implica un gran beneficio a la hora de integrar un software de gestión de red para monitorear la red y obtener información útil.

1.3. Estructura del proyecto

Este documento está organizado en 4 capítulos mediante los cuales se da el mayor enfoque posible, tanto desde el punto de vista práctico como teórico, de todos los aspectos tratados a lo largo del trabajo. Tras este breve capítulo de **introducción**, el segundo capítulo expone los **Conceptos teóricos** donde se hace una revisión del estado del arte del concepto de Cloud y Cloud computing, de los diversos tipos de nube que existen y sus beneficios así como las desventajas de usar esta tecnología. Se define también los diferentes proveedores cloud que hay, indicando brevemente sus características, y entrando mas de lleno en Openstack, en los componentes que lo forman y que hace cada uno de ellos. La implementación de la red cloud haciendo uso de los diferentes módulos explicados anteriormente para poder crear la topología deseada, es descrita en el capítulo tres, dónde también se explica como hacer uso del software de Openstack tanto desde la interfaz web como desde el Terminal. Por último, se discuten las principales conclusiones y se proponen ideas para extender la aplicación de este proyecto.

Capítulo 2

Conceptos teóricos

Antes de comenzar con la parte práctica del trabajo, se hace necesario introducir algunos aspectos teóricos relevantes a la hora de llevar a cabo la implementación que se tratara posteriormente.

2.1. Cloud Computing

A la hora de introducir la base teórica de este proyecto, es necesario hacer una distinción entre el cloud mas conocido comúnmente y el cloud computing.

El cloud que la gran mayoría de las personas conocen, esta referido al Cloud Storage, es decir, al almacenamiento de datos fuera de nuestros dispositivos. Este término, nube en inglés, tiene su origen en los años 60, y se le atribuye a Joseph Carl Robnett Licklider, que tenia una visión muy avanzada acerca de las redes y que uno de sus objetivos era que cualquier persona pudiese estar interconectado para poder acceder a datos o programas desde cualquier lugar. Por lo tanto, el cloud storage, es un sistema que permite alojar datos en Internet. Una vez alojados en la nube, cualquier persona con acceso a dichos datos, ya sea el usuario que los alojo o cualquier persona a la que le diesen permisos, podrá acceder a dichos datos desde cualquier dispositivo que pueda conectarse a Internet.

Estas ventajas, añadidas a que es un sistema de almacenamiento de datos mas barato que los sistemas tradicionales, que son físicos, concluyen en que el almacenamiento en la nube, esta sustituyendo a los sistemas de almacenamiento tradicionales rápidamente. Si en 1960 fue cuando se atribuye el origen del termino "cloud", en 1960 John McCarthy fue el primero en afirmar que la tecnología de las computadoras llevaría a que las aplicaciones llegaran a venderse como un servicio, pero no fue hasta mas de 30 años después cuando la primera empresa introduce la entrega de aplicaciones via web. En 1999 se establece el primer servicio empresarial que ofrece aplicaciones a través de un sitio web, lo que después se denominó cloud computing. Cloud computing es ofrecer servicios a través de una red, que comúnmente suele ser Internet, es decir, los clientes no tienen alojados las aplicaciones o servicios que se les ofrecen sino que los adquieren a través de la nube, diferenciándose en modelos antiguos de proveedores de servicios, en que el cloud computing, tiene las ventajas de la nube, es decir, estos servicios son accesibles desde cualquier lugar, siempre que se tenga conectividad a Internet. Es una tecnología que permite el acceso remoto a software, almacenamiento y procesamiento de datos mediante Internet. El cloud computing ofrece al individuo la capacidad de unos recursos de computación, que tendrán un buen mantenimiento, estarán dotados de una alta seguridad, serán de fácil acceso y bajo demanda, es decir, escalables. Lo que a las empresas las dota de flexibilidad ya que sus datos son accesibles desde cualquier lugar. Como se detalla en [2], el cloud computing simplifica mucho las tecnologías de computación y es accesible sin necesidad de unos conocimientos realmente avanzados, aunque aún está en sus primeras fases de todo el potencial que puede llegar a tener.

2.1.1. Ventajas y Desventajas

2.1.1.1. Ventajas

Esta tecnología tienes varias características, que hacen que sea una opción real a considerar frente a otros sistemas tradicionales. Dichas características son:

- Accesibilidad a la hora de contratar servicios, ya que no se necesita una infraestructura para usarlos, sino que el proveedor te lo ofrece, y el es el que tiene esta infraestructura, el cliente solo lo usa.
- Bajo coste, debido a que se prescinde, como cliente, de servidores y mas hardware, que antes era necesario mantenerlo, configurarlo y actualizarlo, mientras que al contratar un servicio cloud, el responsable de estar tareas es el proveedor, el cliente no se preocupa y no gasta dinero en eso. También se ahorra en equipamiento, ya que la empresa proveedora es la encargada de actualizar los equipos no el cliente.
- Escalabilidad y elasticidad a la hora de tener una infraestructura, ya que el servicio cloud, te permite tener la infraestructura necesaria en cualquier momento, sin costes añadidos y el cambio es inmediato, en función de las necesidades del momento.
- Gran seguridad, debido a que los proveedores encargados de securizar los sistemas son empresas muy grandes las cuales tienen unos niveles de seguridad muy altos, el cliente solo tendrá que securizar los servicios que aloje en los sistemas que el proveedor le proporciona.
- Multiplataforma, ya que como se ha comentado antes, teniendo conexión a Internet, el servicio es accesible desde cualquier dispositivo.

2.1.1.2. Desventajas

Pero también tienen desventajas en su uso, aunque son minimas en comparación a las facilidades que dan estos servicios. Entre ellas destacan:

- Es necesaria una conexión a Internet para trabajar en la nube, ya que sin ella el usuario no podrá acceder a los servicios que haya adquirido.
- Los servidores de la nube pueden estar en cualquier parte, por lo que si hay algún problema legal, no se sabe que legislación regularía dicho problema y si esta protegería al usuario final.
- La información no esta en manos de los usuarios lo que genera discusión sobre a quien pertenece.

2.1.2. Clasificacion de Servicios Cloud

Como se detalla anteriormente, los proveedores de servicios cloud pueden ofertar servicios totalmente diferentes, y que se detallarán cada uno de ellos a continuación, ya que desde el punto de vista del cliente, la implementación es muy diferente en función de que servicio contrate y cuál satisfaga sus necesidades.

- **IAAS (Infraestructure as a Service):** En este tipo de servicio se contrata, como dice su nombre, la infraestructura, es decir, se contrata capacidad de almacenamiento y computacional. Se adquiere servicios de red, servidores, almacenamiento, pero abstraídos, es decir, el cliente no mira por el tipo de hardware o de servidor que van a proporcionarle, el cliente solicita un servicio, y se preocupa de que ese servicio, se realice correctamente sin preocuparse del tipo de hardware que se usa para que sea eficiente. De la gestión, mantenimiento y actualización de dicho hardware, virtualizado, se encarga el proveedor del servicio.

- **PAAS (Platform as a Service):** El proveedor ofrece al cliente una plataforma sobre la cual poder trabajar sobre una máquina o contenedor. El cliente tendrá un sitio donde trabajar sin importar el sistema operativo. El proveedor proporciona un servidor de aplicaciones, en el que se pueden desarrollar y ejecutar aplicaciones. Se ofrece al cliente lo necesario para desarrollar una aplicación y ponerla a punto. El mantenimiento del sistema operativo, la red, los servidores y la seguridad serán tareas del proveedor. Este servicio estaría enfocado hacia desarrolladores .

– **SAAS (Software as a Service):** Este modelo, se centra en facilitar el acceso a una aplicación de software al cliente mediante una interfaz o navegador web. Sería el ejemplo más comúnmente conocido, y mas usado por la gran mayoría de publico. El usuario adquiere el software que puede usar de forma gratuita o pagar por tener unas mejores prestaciones. Se ejecuta una aplicación, pero los servicios estarán en la nube. El caso de Dropbox seria un ejemplo de este tipo de servicios, en los que el usuario vía web, accede a un software en la nube donde almacenar archivos.

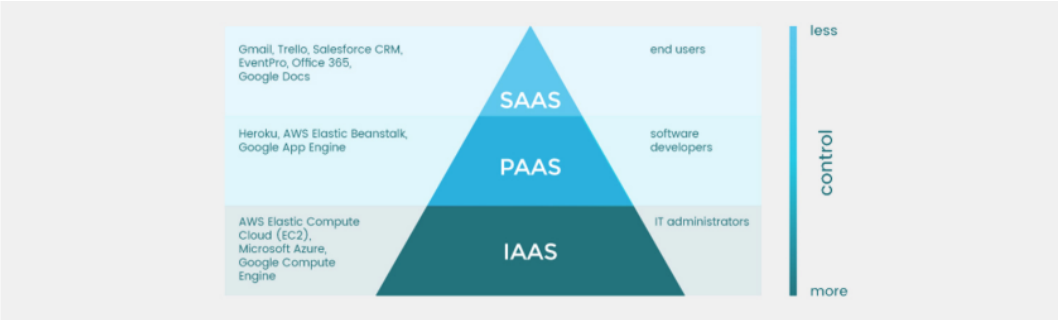

Figura 2.1: Servicios Cloud

2.1.3. Clasificacion de Tipos de Despliegue

Al igual que hay diferentes tipos de servicios ofertados por los proveedores cloud, también hay diferentes tipos de despliegue o tipos de redes cloud segun su ámbito.

- **NUBE PUBLICA:** En la que se basa el modelo de cloud, en la cual los servicios se ponen a disposición del usuario a través de Internet. En este tipo de nube, se ofrece poco margen de personalización, pero se gana en rapidez de entrega, ya que simplemente registrándote ya puedes usar los servicios contratados.

- **NUBE PRIVADA:** Este concepto de nube consiste en una infraestructura en la nube que se implementa en una empresa. Este tipo de nube requiere una gran inversión que la hace mas costosa que la pública, pero permite usar seguridad avanzada en ella, y se tiene una gran disponibilidad y tolerancia a fallos.

- **NUBE HIBRIDA:** Combina los servicios de las nubes publicas y privadas. Se asignan las tareas a una u otra nube según la necesidad. Se contrata un cloud privado, contratando a su vez servicios públicos para que sean accesibles de una forma mas rápida. Este tipo de nubes es recomendable para comercio electrónico, donde el proceso de pedidos se haría de forma publica y los datos de las cuentas de los clientes se gestionarían en la parte privada.

2.1.4. Proveedores Cloud

Una vez se han introducido las bases del cloud computing así como los servicios que se pueden obtener y los tipos de despliegues que se pueden llevar a cabo, es necesario conocer en este momento que proveedores cloud existen, cuales son los servicios que ofertan y las caracterísiticas de los mismos para conocer cual se ajusta mas a las necesidades del cliente.

Un proveedor cloud es una compañía externa que ofrece servicios de plataforma, infraestructura, aplicaciones o almacenamiento basados en la nube. Estos proveedores son una parte vital de la implementacion de un servicio cloud, porque aparte de que sera la empresa que proporcione dicho servicio, el cliente gran parte del tiempo que gastará en poner en marcha el servicio, lo empleará en estar en contacto con su proveedor. Por eso la elección del proveedor cloud correcto para el proyecto se antoja decisivo. Entre los proveedores cloud que hay, destacan los siguientes:

- **AWS (Amazon Web Services:** Amazon Web Services es una plataforma segura de servicios en la nube que ofrece potencia de cómputo, almacenamiento de bases de datos, entrega de contenido y otras funcionalidades para ayudar a las empresas a ajustar su escala y crecer. La nube de AWS proporciona un amplio conjunto de servicios de infraestructura, como potencia de cómputo, opciones de almacenamiento, redes y bases de datos, que se ofrecen como una utilidad: bajo demanda, disponibles en cuestión de segundos y pagando solo por lo que utiliza.

- **Google Cloud Platform:** Google ha hecho un esfuerzo por organizar conjuntamente distintos productos que hasta hace relativamente poco tiempo ofrecía por separado, dando al usuario un mismo punto de entrada a tareas administrativas, de facturación o de soporte de los diferentes servicios, Con la finalidad de recuperar el terreno perdido frente a otros proveedores como Amazon, por su entrada tardía en un sector clave dentro del negocio del cloud computing como es el IaaS.

- **Microsoft Azure:** Azure es una nube pública de pago por uso que te permite compilar, implementar y administrar rápidamente aplicaciones en una red global de datacenters de Microsoft. Presta servicios de infraestructura, desarrollo, pruebas para aplicaciones, copiad de seguridad y administración de usuarios. Por lo que a pesar de ser una plataforma muy reciente, se puede observar que tiene una enorme capacidad.

- **Eucalyptus:** Es una infraestructura open source 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". También puede usar gran variedad de tecnologías de virtualización de hardware incluyendo hipervisores VMware, Xen y KVM para implementar las abstracciones de nube que soporta. Actualmente posee una interfaz orientada al usuario que es compatible con los servicios pero la plataforma está modularizada para poder utilizar un conjunto de interfaces diferentes simultáneamente.

- **Openstack:** Openstack será el proveedor cloud elegido para la realización de este proyecto, y lo analizaremos en profundidad, tanto globalmente como cada uno de sus principales componentes en el siguiente capitulo.

2.2. Openstack

Openstack es un software de gestión de clouds, un proyecto de cloud computing basado en IAAS que a partir de componentes opensource, software libre, proporciona infraestructura escalable y elástica para poder implementar nubes tanto públicas como privadas, como se explica en [3]. Es modular y totalmente adaptable, lo que quiere decir que se instala por componentes y se pueden adaptar las instalaciones a las necesidades del usuario. Es el software libre de gestión de cloud mas potente que existe actualmente y cuenta con distintas distribuciones, cada una desarrollada por una compañía tales como IBM, Mirantis, RedHAt, HP, etc. . . Como se comenta anteriormente, se puede instalar por componentes de forma totalmente manual y personalizada por lo que no habrá dos instalaciones de OpenStack iguales.

Openstack, se divide en diferentes proyectos, que gestionaran cada uno de sus módulos o componentes. Dichos componentes, se encargan de gestionar las diferentes áreas del servicio que ofrece este software. Para gestionar cada uno de los componentes de Openstack, se encuentran dos formas de realizarlo, una mediante interfaz web accediendo al Dashboard y existe la opción de gestionar los componentes a través de la consola. Entre los componentes de Openstack destacan los siguientes:

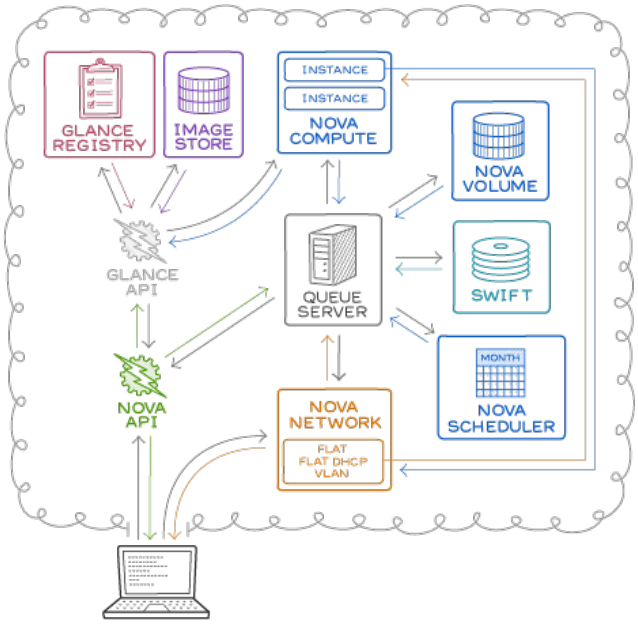

Figura 2.2: Componentes OpenStack

2.2.1. Horizon (Dashboard)

Horizon es el componente que se encarga de gestionar la interfaz web de Openstack, para que el acceso a la infraestructura sea mucho mas intuitivo y sencillo para administradores y usuarios. Desde el Dashboard se pueden gestionar la mayoría de elementos de una implementación cloud en Openstack, tanto a nivel de Administración como a nivel de Proyecto.

Las principales características de Horizon son las siguientes:

- Da soporte al resto de servicios de Openstack. Proporciona una API estable y robusta para el resto de proyectos de Openstack.
- Es extensible, y cualquiera puede añadir nuevos componentes. Cuando se quiera añadir un nuevo contenido, se tendrá que crear un nuevo objeto dashboard y conectarlo a los que ya existen,
- Es muy manejable, tal y como se menciona con anterioridad tiene un diseño muy limpio y sencillo de entender.
- Proporciona una serie de clases y plantillas reutilizables que mantienen la consistencia entre las aplicaciones.
- Es muy estable y mantiene compatibilidad con versiones anteriores.

2.2.2. Nova (Compute)

Gestiona todo lo que tiene que ver con los demás componentes. Es el componente más importante de Openstack y también el mas complejo. Puede estar distribuido en distintas ubicaciones y es el componente principal encargado de controlar todos los hipervisores y desde donde se lanzaran las instancias. Para lanzar dichas instancias son necesarios una serie de requisitos, como son:

- Una imagen
- una red
- Un sabor, que es la cantidad de recursos de computo, almacenamiento y memoria con los que se lanzará la instancia. El sabor que se le da a una imagen, se podrá cambiar posteriormente de que se haya lanzado la instancia.
- Opcionalmente podemos tener:
  - Pares de claves
  - Volumen, para guardar lo que hagas en cada instancia, ya que si no le asignas un volumen a la instancia, al apagarla perderas todo lo que hayas guardado.

Los procesos principales del componente nova son los siguientes:

- **nova-api:** acepta y responde a llamadas API del usuario final. También inicia tareas como la ejecución de instancias virtuales.
- **nova-compute:** se encarga principalmente de la creación y borrado de instancias de máquinas virtuales. Básicamente se encarga de aceptar las peticiones y ejecutar los comandos.
- **nova-volume:** se encarga de gestionar la creación, conexión y desconexión de volúmenes de almacenamiento. Aunque, el componente de Openstack Cinder, se encargara de sustituir a este proceso.
- **nova-network:** acepta peticiones de tareas de red y las encola para ejecutarlas sobre la infraestructura de red.
- **nova-schedule:** se encarga de aceptar las peticiones de las máquinas virtuales y decide en que host se ejecutarán.

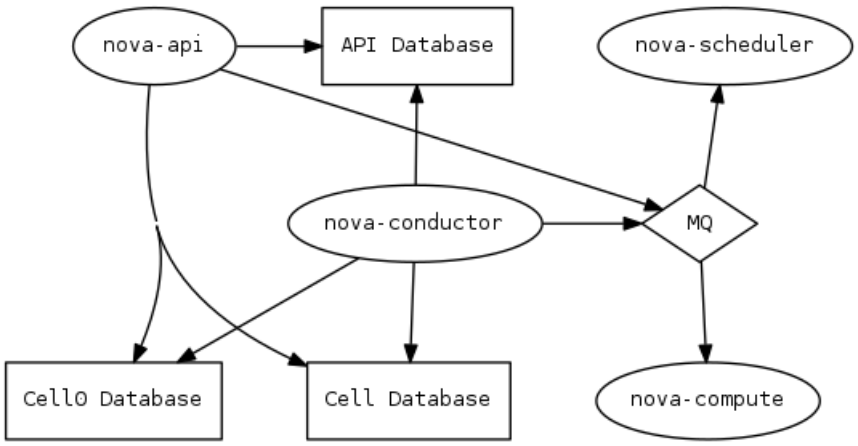

Figura 2.3: Componente Nova

2.2.3. Glance (Imagenes)

Es el componente que gestiona las imágenes de Openstack, controla las imágenes que se ejecutaran cuando se vaya a lanzar una instancia. Proporciona el registro, descubrimiento y entrega de imágenes de máquinas virtuales. Estas máquinas virtuales serán las instancias que se lanzarán. Las imágenes serán la base de la máquina virtual o instancia lanzada. Cuando accedemos a Openstack, en el apartado de Imágenes se encuentran ya unas imágenes almacenadas que podremos usar para lanzar una instancia rápidamente. Serán imágenes muy pequeñas y básicas, pero muy funcionales para un primer despliegue y encuentro con el software. Las imágenes pueden encontrarse en varios estados una vez que las creamos:

- **queued:** los datos de la imagen aun no se han añadido al servicio.
- **saving:** cuando se están añadiendo dichos datos.
- **active:** imagen completamente disponible para su uso. Los datos se añadieron completamente.
- **killed:** en la carga de datos hubo un error y no es usable.
- **deleted:** se han borrado los datos de la imagen y se procederá a su eliminación mas adelante.
- **pending_delete:** aun no se han borrado por completo los datos y puede ser recuperable.

Cuando se crea una imagen en Openstack, deberemos indicar el formato en el que se desea. Hay varios tipos de formato de disco, algunos de los que soporta Glance serian los siguientes:

- **vhd:** formato de archivo que representa una unidad de disco duro virtual. Es usado en Virtualbox por ejemplo.
- **vdi:** formato de disco soportado por VirtualBox y QEMU.
- **iso:** imagen que contiene información en formato ISO, como por ejemplo el contenido de un CDROM o DVD.
- **qcow2:** es un formato utilizado por QEMU y que puede expandirse dinámicamente.

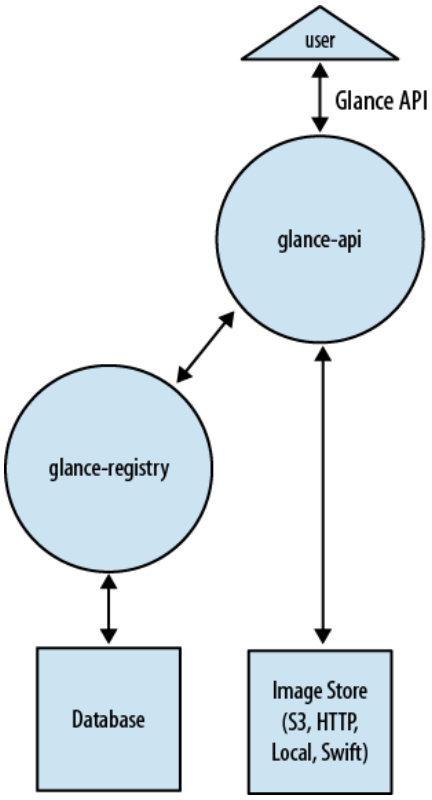

Figura 2.4: Componente Glance

2.2.4. Neutron (Red)

Gestiona todo lo que tiene que ver con la red de Openstack y las controla. Es un módulo también bastante complejo, basado en SDN, abstrae la red del cliente y la gestiona íntegramente a través de software. Es el componente de Openstack preparado para ofrecer Networking como servicio (NaaS) y permite trabajar con plugins de proveedores, estos plugins controlarán a unos agentes que son los encargados de gestionar la red en el nodo de computo en el que se encuentren instalados. Permite crear servicios de red como VPN, Firewalls o Load Balancing entre otras. Los tipos de redes que se pueden crear desde este componente se podrían dividir en dos grandes tipos, dependiendo de quien puede administrarlas y gestionarlas:

- Redes de Proveedor (Provider Network): Son las redes administradas únicamente por el administrador de Openstack. Pueden gestionar redes externas, conectarse a Internet, etc. . .
- Redes de Proyecto (Tenant Network): Son las redes administradas por cualquier miembro de un proyecto y son únicas y exclusivas a ese proyecto. Solo pueden crear redes internas para que las instancias se vean dentro de esas redes en cada proyecto. El cliente podrá crear una red a través de la cual se podrán ver las instancias conectadas a dicha red, pero no podrán ver a instancias conectadas a redes distintas.

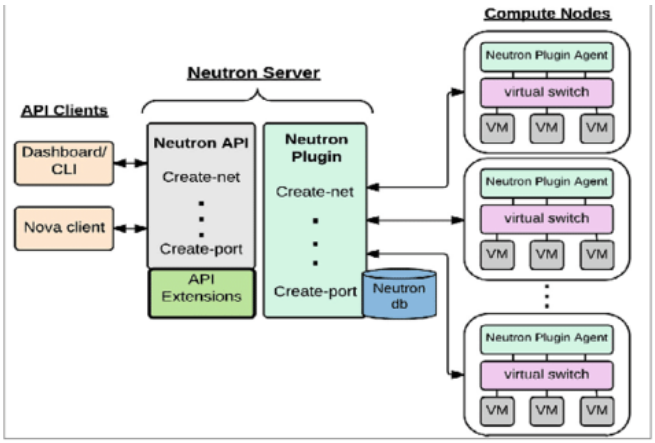

Figura 2.5: Componente Neutron

A su vez, las redes que se creen mediante el componente de Neutron, como hemos visto pueden ser Provider o Tenant Network. Estas redes a su vez pueden de un tipo de red concreto en funcion de la configuración que queramos. Los tipos que Neutron soporta son Flat, LAN, VLAN, GRE y VXLAN.

Este componente proporciona al cliente ciertos servicios que podra gestionar libremente, entre los que destacan los siguientes:

- Crear grupos de seguridad para controlar por que puertos se pueden acceder a las instancias.
- Crear redes y subredes.
- Crear routers.

2.2.5. Keystone (Gestion de Usuarios)

Gestiona todo lo relacionado con permisos de usuarios, grupos y sus roles en los proyecto. Principalmente, Keystone tiene dos funciones que son las siguientes:

- Gestionar usuarios, manteniendo la información de estos mismos así como los permisos que tienen sobre la infraestructura.
- Proporciona un catalogo de servicios, que están disponibles, y la ubicación de las APIs para poder usar dichos servicios.

Keystone también gestiona no solo los permisos que tienen los usuarios dentro de un proyecto o servicio, sino también los permisos que tienen ciertos componentes para operar dentro de otros componentes. Estas acciones entre los diferentes servicios las gestiona a través de tokens, que son unos conjuntos de bits, que indicaran si la acción se podrá llevar a cabo o no.

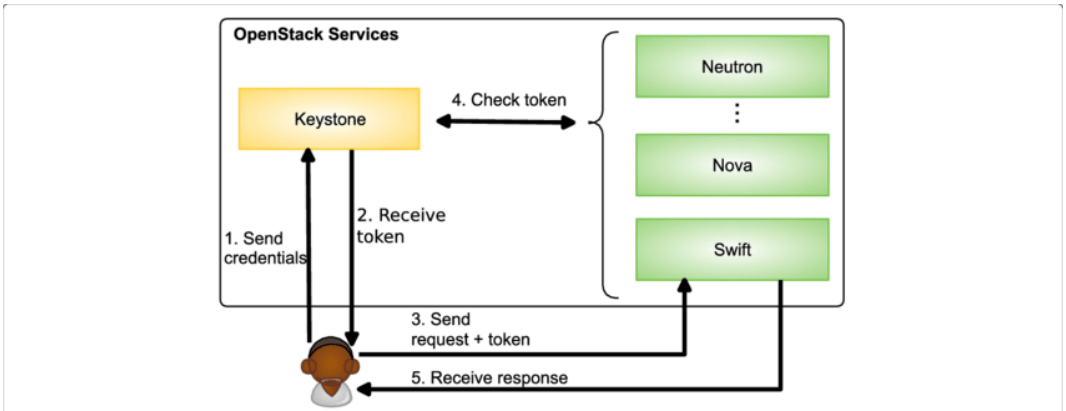

Figura 2.6: Componente Keystone

Keystone tiene varias partes fundamentales a controlar, que serían las siguientes:

- Usuarios y Grupos: son las identidades digitales de personas, sistemas o servicios que usan Openstack. Keystone se encarga de validar que la petición es realizada por el usuario correcto y cuando se loguean, a ese usuario se le asignan permisos para acceder a los recursos. Estos permisos son denominados "tokens". Los grupos serían asociaciones de varios usuarios.

- Token: es un texto utilizado para el acceso a recursos. Cada token describe a que recursos se puede acceder.

- Roles: serán los poderes o permisos que tendrán los usuarios o grupos dentro de un proyecto, para realizar tareas sobre él. En Keystone, el token asignado a un usuario incluye en el los roles asignados a dicho usuario o grupo. Son acumulativos, lo que quiere decir que si un usuario tiene el rol de administrador dentro de un proyecto, y esta en un grupo que tiene el rol de miembro dentro de ese proyecto, al usuario se le dotara de ambos roles, administrador y miembro. Si no se tiene un rol definido dentro del proyecto, el usuario no podrá acceder a este mismo.

2.2.6. Cinder (Almacenamiento)

Antes de hablar de Cinder, se hace necesario sitinguir entre los dos tipos de almacenamiento que existen en Openstack.

- Almacenamiento efímero (Nova): este almacenamiento esta vinculado al ciclo de vida de una maquina virtual concreta. Es decir, cuando se apaga la instancia, se suprime el dispositivo de almacenamiento y se borra todo lo que se ha realizado en ese tiempo.
- Almacenamiento persistente:
  - Almacenamiento de bloque (Cinder)
  - Almacenamiento de objetos (Swift)
  - Almacenamiento compartido de archivos (Manila)

Cinder es el componente de Openstack que gestiona el almacenamiento persistente de bloque. Apareció por primera vez en la versión de Openstack Folsom y es una copia del componente nova-volume, por lo que se puede usar cualquiera de los dos. Utiliza LVM y iSCSI para que las máquinas virtuales accedan a recursos de almacenamiento según sus necesidades.

Este almacenamiento de bloque se puede compartir pero no a la vez.

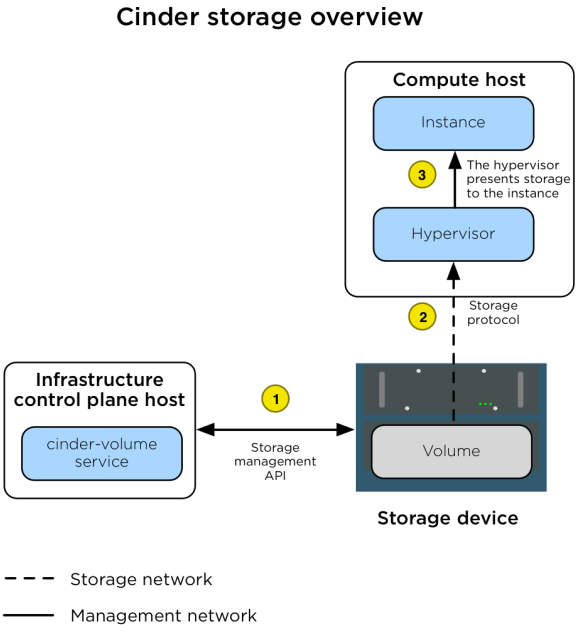

Figura 2.7: Componente Cinder

Los volúmenes de almacenamiento se crearan dentro del software de Openstack, según las necesidades que se requieran y se asignara dicho volumen a una instancia. Pero si queremos asignarle ese volumen a otra instancia, debemos asegurarnos de que esta libre, es decir, que no esta asignado previamente a otra. Por lo tanto, los volúmenes se pueden usar por varias instancias, pero no a la vez, no puede estar asignado a dos instancias al mismo tiempo. Para tener un volumen en dos instancias a la vez, se tiene MANILA.

2.2.7. Swift (Gestion de Objetos)

Como se ha visto anteriormente en el apartado de Cinder, Swift es el módulo encargado de la gestión de objetos y contenedores de Openstack. Como objetos, se entiende a entidades únicas que almacenan información. Swift tiene una alta consistencia de datos, lo que significa que en cada nodo de Openstack guarda 3 replicas de cada objeto. Cada objeto almacenado en Swift tiene una dirección URL, que son distintas unas de otras y propias de dicho objeto. Esta URL tendrá el siguiente formato:

https://storage.example.com/v1/AUTH_acct/cont/obj

Donde:

- storage.example.com es la dirección IP del modulo de CINDER.
- v1 es la versión de la API
- AUTH_acct es la cuenta en Swift, que estas cuentas son los proyectos.
- Cont será el contenedor donde se aloja dicho objeto, que además gestionan sus permisos.
- Obj es el objeto al que pertenece dicha URL

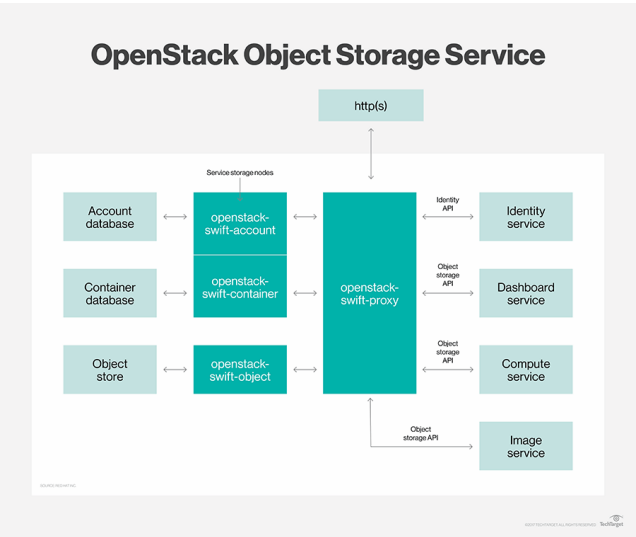

Figura 2.8: Componente Swift

2.3. Gestion en la Nube

La gestión de red consiste en la monitorización, análisis y control de los recursos de una red para que la implementación de dicha red cumpla los requisitos y necesidades que se requerían previamente a su instalación, y se realiza mediante tareas de despliegue, integración y coordinación de hardware, software y elementos humanos.

Es muy importante la estrategia de gestión, ya que si está bien implementada permite al usuario mantener el control sobre el entorno cloud, que es dinámico y escalable. Dicha estrategia permite la consecución de varios objetivos. Primeramente, la gestión en la nube proporciona la capacidad de autoservicio, lo que elimina el proceso tradicional de gestión de los recursos IT. El usuario puede acceder a la nube, sea del tipo que sea, publica, privada o hibrida y monitorizar el estado de cada componente, el estado de las instancias lanzadas y ver el gasto de recursos que tiene cada una de ellas, y por ejemplo conocer de esta forma las instancias menos utilizadas pararlas para reducir el gasto computacional. En implementaciones para entornos de laboratorio pequeños, la monitorización del estado de cada componente de Openstack y de cada componente de la topología implementada sería lo mas importante. En una implementación en un entorno empresarial, la gestión cloud también permite automatizar el flujo de trabajo, lo que permite crear y administrar las instancias lanzadas sin necesidad de una intervención humana, gracias a convertir las políticas de negocio en pasos realizables. Esta automatización, también permite que las empresas cumplan sus objetivos de implementación y cumplimiento, mediante la generación de informes, ya que las herramientas de gestión generan informes y alertas, por ejemplo cuando se vaya superar la capacidad total del sistema.

Por otra parte, la gestión cloud también permite conocer las cargas de trabajo y la experiencia del usuario. En una nube privada, las organizaciones pueden garantizar que su infraestructura funciona correctamente. En las nubes públicas, las métricas de rendimiento para la latencia o el tiempo de inactividad ayudan a garantizar el cumplimiento con los acuerdos de nivel de servicio del proveedor de la nube pública. Gracias a estas métricas pueden decidir si es mas viable cambiar de proveedor o continuar con el actual.

Gracias a la gestión de red, se consigue optimizar y mejorar el uso de los recursos existentes, lo que conlleva una mejora en la calidad de la red, que es importante para la mejora en el rendimiento de las aplicaciones cloud.

Sin embargo la gestión de red tiene algún inconveniente, como puede ser la falta de integración entre los sistemas de gestión, lo que en un entorno cloud, al tener elementos de red tales como servidores, hardware de red y demás componentes mucho mas concentrados que en un entorno tradicional, provoca la limitación de la mejora de calidad de las redes, tal y como se comenta en [2]. Los problemas que tiene la gestión de red, se vuelven mas graves en un entorno cloud, lo que conduce a que la gestión de red cloud pierda fiabilidad y confianza, porque hay que diferenciar entre la gestión en una red cloud y la tradicional, ya que la red tradicional implica gestionar cada dispositivo y los enlaces que los interconectan, por el contrario, en una red cloud los componentes están virtualizados y los enlaces también, por lo que hace que la gestión de las redes virtuales sea muy importante ya que errores que sean frecuentes pueden desencadenar la caída de toda la red. Al igual que con los proveedores de servicios cloud, para realizar la gestión de red en un entorno cloud se tienen diversas herramientas muy distintas unas de otras. Hay varios tipos de herramientas que ayudan a realizar una gestión y un mantenimiento de la red adecuados. Los distintos tipos de herramientas que existen son los siguientes:

- **Herramientas de configuracion:** estas herramientas proporcionan un control de los recursos que se necesitan para la provision de servicios. Dichas herramientas también pueden dar información sobre la correcta configuración de estos recursos y las relaciones entre estos. Se diferencian dos tipos de herramientas, las nativas o las de terceros.
  - **nativas:** conllevan a ser mas dependiente del proveedor de nube publica, ya que cuando se usan dos o mas nubes publicas, la herramienta nativa no funcionara correctamente en ambas, si son de proveedores distintos. Un ejemplo de una herramienta nativa de configuración seria **AWS Config**, de Amazon Web Service. Esta herramienta monitorea y registra continuamente la configuración de los recursos de AWS, pudiendo automatizar la revisión de las configuraciones registradas en referencia a las que se deseaban en un inicio. Entre los beneficios que aporta, destacarían los siguientes:
    - Monitoreo constante: se puede monitorear de una forma continua y dejar registrados todos los cambios en la configuración de los recursos.
  - **terceros:** Las de terceros, corrigen ese problema y pueden operar con varios proveedores cloud pero la capacidad de estas herramientas es menor. Para garantizar la multioperabilidad tienen que renunciar a ciertas funciones que las herramientas nativas si poseen. Un ejemplo de una herramienta de configuración de terceros seria **Puppet**. Puppet es una herramienta de código abierto diseñada para administrar la configuración en sistemas Unix y Windows. En esta herramienta, el usuario mediante el lenguaje Puppet, describe los recursos del sistema y los estados de cada uno de ellos.

– **Herramientas de monitorizacion:** las herramientas de monitoreo son esenciales para asegurar el funcionamiento de sistemas informáticos y para evitar cualquier fallo en la red. También al aportar mucha información sobre el uso de recursos dentro de la red, resulta de gran ayuda a la hora de optimizar la red. Los principales beneficios que hay a la hora de usar este tipo de herramientas serian:

  - Optimización de la instalación: con el uso de estas herramientas, se puede acceder de una forma muy fácil a un análisis completo y general de la instalación, lo que facilita detectar cualquier tipo de fallo o anomalia.
  - Tráfico intruso: este tipo de software permite también detectar todo tipo de trafico malintencionado dentro de la red monitoreada.
  - Logs: una de las grandes ventajas es, que estas herramientas generan logs y analizan el rendimiento de los componentes de la red lo que también facilita analizar el rendimiento de la misma y considerar posibles futuros cambios que generen una optimización y una mejora del rendimiento y la eficiencia.

  Existe un gran numero de programas para realizar una monitorización de la red, y es importante conocer cuales son los mas relevantes y sus características, para elegir el que mas se amolde a las necesidades que existan.

  - **Pandora FMS:** es uno de los softwares más conocidos. Esta herramienta se puede integrar con los dispositivos móviles para monitorizarlos.
  - **Zabiix:** Su configuración es muy sencilla y, además, el interfaz gráfico resulta muy intuitivo.
  - **Monitis:** es una de las mejores opciones para pequeños entornos. Incluye la monitorización de transacciones web y permite monitorizar sistemas de aplicaciones en la nube como Amazon.
  - **Nagios:** esta es la herramienta que se ha escogido para la monitorización de la red implementada en este proyecto. Como punto a favor tiene que se pueden descargar multitud de plugins e incluso construirlos manualmente para monitorizar aspectos muy concretos de la red, e incluso integrarlo en una red cloud. A continuación, se profundizara mas en esta herramienta.

2.3.1. Nagios

Nagios es un sistema de monitorización de redes de código abierto ampliamente utilizado, que monitoriza tanto el hardware como el software que se especifiquen en la red que se este visualiazndo, alertando cuando el comportamiento de los mismos no sea el deseado.

Su función principal es la de comprobar que, el equipo que está siendo monitoreado por él, responda las peticiones para evaluar su estado y en caso de que su comportamiento no sea el esperado Nagios notifique al administrador de la red, via correo electronico entre otros medios, esta notificacion es configurable, identificando 5 estados para los equipos y servicios que son “Up” “Down” “Warning” “Flapping” y “Critical”.

Entre las principales funcionalidades de Nagios destacan las siguientes:

- Las alertas que Nagios envia cuando los componentes críticos de infraestructura fallan y se recuperan, proporcionando a los administradores avisos de eventos importantes. Las alertas se pueden enviar por correo electrónico, SMS o script personalizado.
- Los informes de Nagios proporcionan un historial de interrupciones, eventos, notificaciones y respuesta de alerta para una revisión posterior.
- Programación de tiempo de inactividad: El tiempo de inactividad programado evita las alertas durante las operaciones de mantenimiento y actualización programadas.
- Los gráficos y los informes de planificación de tendencias y capacidades le permiten identificar las actualizaciones de infraestructura necesarias antes de que se produzcan fallos.
- Los controladores de eventos pueden reiniciar automáticamente las aplicaciones, servidores, dispositivos y servicios fallidos cuando se detectan problemas.

2.4. Software de Virtualizacion

Con virtualizacion se entiende como la creación a traves de software, de entornos simulados desde un solo sistema fisico, tal y como se explica en [4]. Este software se llama Hypervisor o Maquina Virtual (VM) y se conecta directamente con el hardware y crea una abstraccion entre el hardware del host y el sistema operativo de la maquina virtual. Las maquinas virtuales dependeran de la capacidad del Hypervisor de abstraer dichos recursos del host y de la capacidad de distribuirlos adecuadamente por todas ellas. Es decir, el hypervisor divide los recursos del host en uno o varios entornos de ejecucion. A la hora de realizar este trabajo, se decide realizarlo mediante un software de virtualizacion, debido a que este método tiene muchas ventajas a la hora de implementar una topología cloud y realizar pruebas de configuracion con seguridad, debido a que se manipulan muchos archivos de configuracion en lo que a conectividad de red y tarjetas de red se refiere. Además de que se aprovechan las innumerables ventajas que tiene la virtualizacion, entre las que destacan las siguientes:

- La virtualización permite consolidar múltiples recursos de TI resultando en ahorros de costes y en mayor eficiencia.
- Utilizando la virtualización para consolidar, se puede ahorrar de manera significativa en consumo total de energía y en dinero.
- las máquinas virtuales son totalmente independientes, entre sí y con el hypervisor. Por tanto un fallo en una aplicación o en una máquina virtual afectará únicamente a esa máquina virtual. El resto de máquinas virtuales y el hypervisor seguirán funcionando normalmente.
- podemos crear las máquinas virtuales con las características de CPU, memoria, disco y red que necesitemos, sin necesidad de adquirir un equipo concreto con esas caracteristicas.

Para realizar esta virtualizacion, existen varios softwares que lo permiten como son el caso de:

- **Vmware:** El software de VMware puede funcionar en Windows, Linux, y en la plataforma macOS que corre en procesadores Intel. VMware es una de las herramientas de virtualización más potente que puedes utilizar.
- **QEMU:** tiene capacidades de virtualización dentro de un sistema operativo, ya sea GNU/Linux, Windows, o cualquiera de los sistemas operativos admitidos.
- **Windows Virtual PC:** es un software gestor de virtualización desarrollado por Connectix y comprado por Microsoft para crear equipos virtuales.
- **Virtual Box:** este software es el elegido para la realización del proyecto y se abordará a continuación con mas detalle.

2.4.1. VirtualBox

Para este proyecto se elige este software, porque es el que mayor familiaridad tenía y aparte por su facilidad de uso y accesibilidad. Ya que otros por ejemplo VMWare tiene varias de sus funciones restringidas al programa de pago y en cambio VirtualBox te ofrece todos los servicios de forma gratuita. VirtualBox es muy simple pero también muy poderoso. Puede ejecutarse desde cualquier lugar, de pequeños sistemas integrados o máquinas de escritorio a implementaciones de centros de datos e incluso entornos en la nube. Una vez concluido el proyecto, VirtualBox nos da la opcion de poder clonar la maquina virtual sobre la que se ha realizado y asi tener la opcion de tener toda la implementación lista para usar en otros dispositivos [4].

Capítulo 3

Implementación

Teniendo en consideración todos los conceptos teóricos explicados anteriormente, se procede a continuación a presentar la forma de uso de cada uno en la implementación del proyecto.

3.1. Entorno de desarrollo

3.1.1. Configuración del entorno de virtualización

Hay que recordar que la implementación de este proyecto se realizará en un entorno virtualizado gracias al software VirtualBox [4]. Se necesita configurar un anfitrión en dicho software que sea el que soporte el entorno cloud que se va a desplegar y desde el cuál se podrá acceder a cada una de las características de dicho entorno.

La máquina desde donde se lanzará esta máquina virtual, es recomendable que cuente con al menos 16GB de RAM para que el entorno cloud vaya fluido desde el mismo equipo y no surjan problemas. En el caso particular de este proyecto, se cuenta con un equipo de 8GB de RAM, por lo que a la hora de configurar el entorno cloud, se tendrá una experiencia menos fluida. La máquina virtual se crear a a partir de una imagen Linux, concretamente una de la distribución de *Debian, Ubuntu 16.04.4*, asignándola 6GB de RAM y 120GB de disco duro. Estos aspectos técnicos pueden variar en función del equipo, pero no es recomendable asignar menos memoria RAM porque puede ocasionar problemas a la hora de lanzar instancias desde el software de OpenStack [6].

Cuando creemos la máquina, hay que asegurarse de que el adaptador de red sea el *Adaptador Puente* y que el dispositivo elegido sea el correcto.

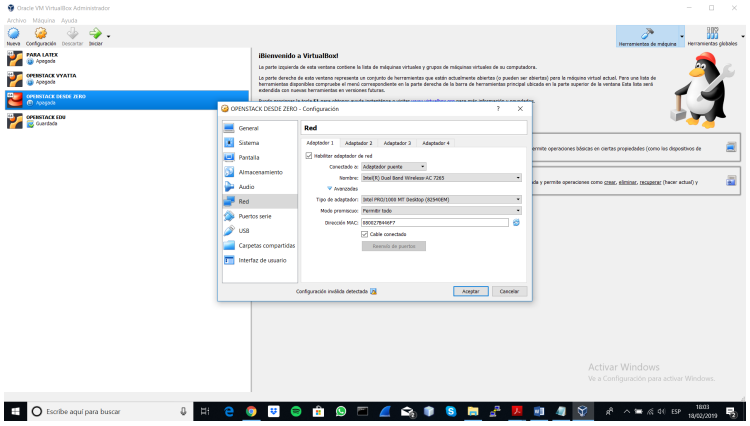

Figura 3.1: Entorno de desarrollo.

3.1.2. Configuración de la máquina virtual

Cuando se tienen los aspectos hardware listos, se inicia la máquina y hay que comprobar que la configuración de red es la correcta, para ello hay que entrar desde el terminal y con permisos de superusuario en el archivo de configuración de la tarjeta de red, que se localiza en la ruta: */etc/sysconfig/network-scripts/ifcfg-enp0s3*, en el caso de que la tarjeta de red sea la enp0s3. Una vez se tiene instalado el entorno virtualizado, se deberán instalar los repositorios de RDO que es la distribuidora de la versión de Openstack que usaremos, que son necesarios para la correcta instalación del mismo.

```
root@openstack:~$ sudo yum install -y \
centos-release-openstack-rocky
```

Para finalizar, hay que actualizar los paquetes de la máquina, desde el terminal:

```
root@openstack:~$ sudo yum update -y
```

Una vez el entorno virtualizado esta correctamente actualizado, hay que instalar la versión de OpenStack que se va a usar. En este proyecto en particular se usará la versión de Packstack, desarrollada por RDO Project, [5]:

```
root@openstack:~$ sudo yum install -y openstack-packstack
```

Solo queda ejecutar la versión que se ha descargado. Para que como red externa se use una red externa ya existente, en este caso la red a la que esta conectada el host, se deberá ejecutar Openstack Packstack con el siguiente comando:

```
root@openstack:~$ packstack --allinone --provision-demo=n \
--os-neutron-ovs-bridge-mappings=extnet:br-ex \
--os-neutron-ml2-type-drivers=vxlan,flat
```

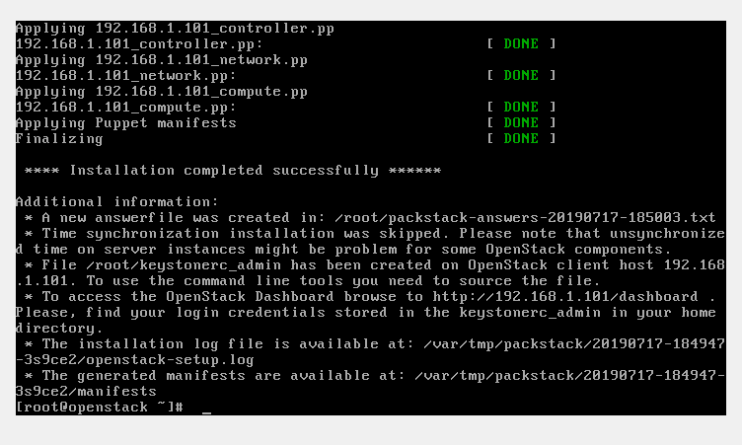

Figura 3.2: Instalacion Packstack.

En este comando, se especifica que no se cree el proyecto demo cuando se inicie el software, y como se ha comentado anteriormente, especifica que el bridge que conecte con la red externa sea la interfaz de red br-ex que se tendrá que configurar posteriormente desde el directorio /etc/sysconfig/network-scripts/. En este directorio se creará el archivo de configuración de la tarjeta de red br-ex, que será la que actúe como puente entre el host y el software de Openstack. En este directorio habrá que hacer dos cosas, primero crear el archivo de configuración de la tarjeta que hará de bridge y luego modificar la configuración de la tarjeta que nos viene por defecto.

Para crear el archivo de configuración de la tarjeta br-ex, se hará uso del comando vi con la siguiente instrucción:

```
root@openstack:~$ sudo vi \
/etc/sysconfig/network-scripts/ifcfg-br-ex
```

En el editor vi que se abrirá, habrá que escribir las siguientes instrucciones para tener la tarjeta correctamente configurada:

Figura 3.3: Configuracion de la tarjeta br-ex.

En el caso del proyecto, se usan esas direcciones tanto para la IP del host como para el Gateway. A la hora de configurar la tarjeta de red, habrá que poner unas ips que se correspondan con el caso particular de cada caso. Como se comenta anteriormente, también habrá que modificar la configuración de la tarjeta de red que se usa por defecto, esto se realizará del mismo modo que hemos configurado la br-ex, a partir del comando vi. El nombre del archivo de configuración de la tarjeta de red a modificar, en este caso particular será ifcfg-enp0s3 y se deberá modificar para que quede de la siguiente forma:

Figura 3.4: Configuracion de la tarjeta enp0s3.

Para que estos cambios tengan efecto, hay que reiniciar el servicio de red del sistema, que se puede realizar de dos formas:

```
root@openstack:~$ reboot
```

```
root@openstack:~$ sudo service network restart
```

Cuando se reinicia y se aplican los cambios, hay que crear desde dentro de Openstack varios elementos como son la red y la subred externas y el router software que se conectará a dicha red, y que empezará a dar forma a la topología de red que se implementará en el proyecto [7]. Estos elementos los crearemos desde la consola o el CLI, y para ello primero hay que loguearse o cargar las credenciales para tener acceso a las funciones de Openstack desde el host. Para ello, cuando se ejecuta el Packstack se crea un archivo en el que están contenidas todas las credenciales llamado *keystonerc_admin*. Para loguearse con permisos de administrador, hay que ejecutar el siguiente comando:

```
root@openstack:~$ . keystonerc_admin
```

Y se concederán permisos de administrador para poder crear diversos elementos necesarios para el proyecto desde la consola. Una vez se esta logueado con permisos de administrador, hay que crear la red externa, que ya se ha asociado previamente a la hora de ejecutar Packstack. Para crear la red externa a partir de la consola, hay que ejecutar el siguiente comando:

```
root@openstack ~(keystone_admin):# neutron net-create \
external_network --provider:network_type flat \
--provider:physical_network extnet --router:external
```

El nombre de extnet, es el nombre que se le ha dado previamente a la red externa a la hora de lanzar Packstack. Una vez se tiene la red externa, hace falta proporcionarla una subred en donde se elegirá un pool de asignaciones de IPs, es decir un rango de @IPs que serán las que se usarán para dar a elementos conectados en esta subred. Para crear dicha subred habrá que ejecutar:

```
root@openstack ~(keystone_admin):# neutron subnet-create --name \
public_subnet --enable_dhcp=False \
--allocation-pool=start=192.168.1.10,end=192.168.1.20 \
--gateway=192.168.1.1 external_network 192.168.1.0/24
```

Lo último que queda por realizar antes de entrar a la creación de la topología de red y entrar mas detalladamente en el software de Openstack, es asignar un router a la subred externa que se acaba de crear y establecer el Gateway del router hacia la red externa. Para ello primero hay que crear el router y después hacer la asignación pertinente:

```
root@openstack ~(keystone_admin):# neutron router-create router1
```

```
root@openstack ~(keystone_admin):#  neutron router-gateway-set \
router1 external_network
```

Una vez se tiene el router creado y asignado correctamente, el entorno en el cual se va a ejecutar e implementar la topología del proyecto en cuestión. Se daría por finalizado.

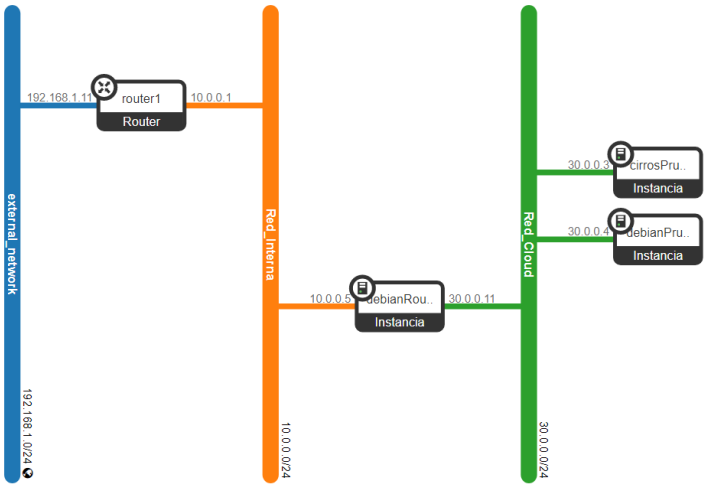

Figura 3.5: Topología final.

En este punto hay que definir la topología a implementar y realizarla mediante los comandos de Openstack. Hay una particularidad y es que para realizar cambios, ya sean crear, modificar o eliminar elementos dentro de Openstack, se puede realizar de dos formas, ya sea a través del CLI, la consola de comandos, o a través del Dashboard, mediante interfaz web.

A continuación se detallará como crear la topología deseada, paso a paso y con ambos métodos, tanto mediante Dashboard como mediante el CLI.

3.2. Creación de la nube privada

Cuando se entre a Openstack se verá la red externa y el router conectado a ella ya creados, como se ha detallado previamente. Lo que queda por hacer es dar forma al resto de la red, viendo las diferentes formas que hay para hacerlo y todas las posibilidades que nos proporciona Openstack. Por ello primeramente hay que saber que topología de red se quiere implementar para tener claro que hay que hacer. En este proyecto el objetivo es tener una topología de red compuesta por la red externa que dará conectividad al exterior, que estará conectada a un router. Ambos elementos ya creados. La otra parte de la topología estará compuesta por dos redes internas, conectadas entre ellas mediante una instancia que hará de router. Esto se hace para que posteriormente, y tal y como se detallará, cuando se integre con el software Nagios, al tener una instancia ejerciendo de router, se podrá obtener más información del router, que si se usa un router software predeterminado de Openstack, del que no se puede obtener mas información que saber si esta encendido o apagado. Por lo tanto, tendremos una red externa, conectada a un router que irá conectado a una red interna. Dicha red interna estará conectada a una instancia, que se recomienda que sea de una distribución Debian y que será la instancia que hará las veces de router, por lo que dicha instancia tendrá otro puerto conectado a una segunda red interna, que será en la cuál se desplegaran todas las instancias de nuestra red. Una vez se tiene clara la topología, hay que desplegarla. Para ello y como se ha comentado anteriormente, hay dos opciones, desde el Dashboard de Openstack o desde el CLI, que es la consola. Primero se detallará la forma para realizarlo desde el Dashboard y seguidamente esos mismos pasos, se mostrarán para realizarlo desde el CLI. Para entrar al Dashboard, lo primero es loguearse, y se solicitará un nombre de usuario y una contraseña. Estos datos se nos proporcionan en el archivo keystonerc_admin que se nos descarga cuando instalamos cualquier distribución de Openstack. Una vez logueados, primero se empezará por desplegar las diferentes redes internas con sus respectivas subredes y rango de IP.

Una vez dentro veremos unos paneles a la izquierda y en la zona superior. En la parte superior se podrá elegir que usuario usaremos, y se recomienda que este marcado como admin. En los de la izquierda se podrá navegar entre las diferentes secciones, del Proyecto, Administrador, etc..

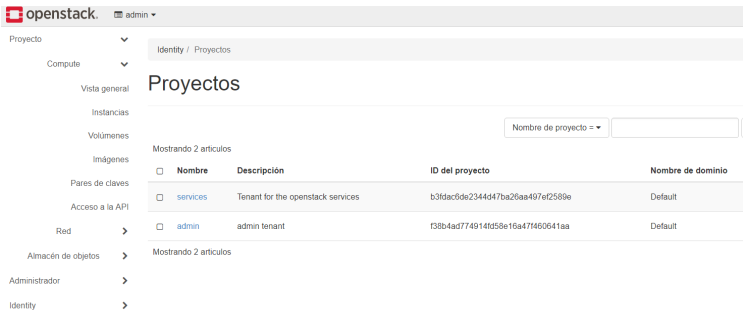

Figura 3.6: Panel Inicial Dashboard

3.2.1. Creación de las redes internas

Para crear las dos redes interna, también llamadas Tenant Network, hay que ir al panel de Proyecto y ahí entrar en Redes. Nos aparecerá en la zona derecha una opción que será "crear red" donde se podrá elegir el nombre de la red entre otras cosas. Aquí también se permitirá también crear la subred asociada a dicha red. Al crear dicha subred, habrá que elegir el nombre de la misma asi como su direccionamiento IP, es decir el rango de direcciones IP que estarán contenidas en la red que se ha creado y también la dirección IP del Gateway. Esto se hará para ambas redes, por ejemplo una red llamada Red_Interna cuyo direccionamiento sea el 10.0.0.0/24 con Gateway de salida 10.0.0.1 y otra llamada Red_Cloud con direccionamiento 30.0.0.0/24 con Gateway 30.0.0.1.

Figura 3.7: Redes Internas

Para realizar esto, la creación de redes y subredes desde el CLI, la consola de Openstack, lo que habrá que hacer primeramente, y que habrá que hacer siempre que se quiera entrar al CLI será loguearse en el proyecto en el que se desea trabajar, en este caso, se trabajará en el proyecto admin, y para loguearse habrá que exportar las variables del archivo keystonerc_admin que como se ha mencionado anteriormente, es un archivo que se crea automáticamente al ejecutar la versión de Openstack. Para exportar dichas variables se hace igual que se ha detallado antes, con el comando:

```
root@openstack:~$  . keystonerc_admin
```

Una vez se esta logueado dentro del CLI, con el comando openstack, es con el que se podrán realizar todas las tareas de creación y administración dentro del software. En este caso concreto, en el apartado de las redes, que lo lleva el componente de Neutron, se recomienda primeramente ejecutar el comando:

```
root@openstack ~(keystone_admin):#  openstack network -h
```

Que mostrará todas las opciones que se tienen para administrar las redes en Openstack. En este caso concreto los que mas interesan serían:

```
root@openstack ~(keystone_admin):#  openstack network create
```

```
root@openstack ~(keystone_admin):#  openstack subnet create
```

Donde si se añade el prefijo -h se mostrarán los diferentes parámetros que se pueden configurar a la hora de crear la red. Estarán prácticamente los mismos que se visualizan desde el Dashboard aunque desde el CLI puede haber alguna opción mas. Por lo tanto para crear las dos redes que tendrá este proyecto, se deberá ejecutar para cada una de ellas los siguientes comandos:

```
root@openstack ~(keystone_admin):#  openstack network create \
Red_Interna --project admin
```

```
root@openstack ~(keystone_admin):#  openstack subnet create \
Subred_Interna --network Red_Interna --subnet \
--range 10.0.0.0/24
```

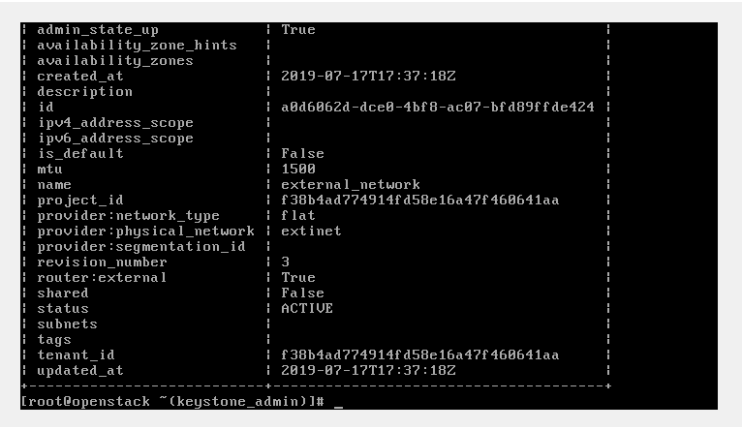

Figura 3.8: Creación Red interna

En el primer comando se creará la red, donde sólo se definirá el nombre de la misma y el proyecto al que pertenecerá y en el segundo comando se creará la subred asociada a dicha red y se elegirá también el direccionamiento de la misma. Habrá que tener cuidado porque en el segundo comando hay parámetros específicos de cada red, como son el nombre de la red a la que pertenecen (–network nombre_red) y el direccionamiento (–subnet-range x.x.x.x/y) que deberán ser distintos en cada red que se cree. Lo que se cree a través del CLI, se actualizará si se accede posteriormente al Dashboard y viceversa. Después de crear estas dos redes a través del CLI deberían aparecer con los parámetros definidos desde el Dashboard.

3.2.2. Preparación de los componentes para lanzar las instancias

Una vez se tienen todas las redes de nuestra topología creadas, llega el momento de lanzar las instancias. Para lanzar las instancias, primero hay que crear ciertos requerimientos que necesita una instancia para funcionar correctamente, como son la imagen, el volumen, los grupos de seguridad etc.

3.2.2.1. Creación de la imagen

Para subir la imagen a Openstack a través del Dashboard, se tendrá que descargar la imagen en la máquina que haga de host. En el caso concreto de este proyecto se ha usado una imagen Debian para lanzar las instancias. Si no se va a requerir un funcionamiento muy activo de las instancias, se puede usar la imagen de cirros que viene instalada en Openstack por defecto, pero es una imagen pequeña que puede ser algo limitada. Para subir una imagen que se ha descargado previamente, habrá que ir al apartado de Imagen en Proyecto si se quiere que la imagen sea solo accesible desde el Proyecto en el que se trabajará. Si se quiere que esa imagen sea accesible desde mas proyectos, habrá que ir a Administrador >Imagen y desde ahí subirla. Para subirla se hace desde el botón a crear imagen y se nos abrirá un submenú en el que se podrá determinar el nombre que llevará dicha imagen, se recomienda dar un nombre sencillo, debian por ejemplo, también se elegirá la forma a través de la cuál se importará la imagen, que en este caso concreto será desde un archivo, y se buscará dicho archivo y se cargará. En este submenú se podrá elegir también el formato de la imagen, que vienen en la página donde se descarga la imagen, aunque es recomendable usar el formato qcow2. También si es necesario se podrán establecer los requerimientos mínimos de la imagen, si es que necesita unos requerimientos mínimos para ejecutarse.

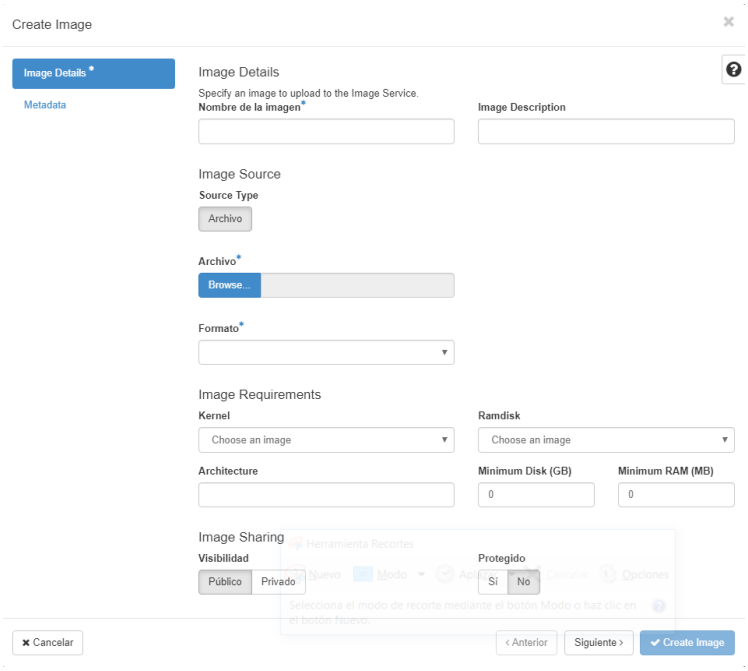

Figura 3.9: Creación imagen Dashboard

A la hora de crear una imagen desde el CLI, primero hay que descargar la imagen en el equipo o máquina donde se ha instalado el software de Openstack. Esto se puede hacer de varias formas, dos de ellas podrían ser las siguientes:

```
root@openstack:~$  curl -o nombre_imagen.img ruta_enlace
```

```
root@openstack:~$  wget ruta_enlace_imagen
```

Una vez se ha descargado la imagen en el equipo, se podrá importar a Openstack gracias al módulo de Glance que se ha explicado previamente. Como se explica antes, si no nos hemos logueado previamente, hay que hacerlo exportando las credenciales como se detalla mas arriba. Al igual que para crear las redes, es recomendable ver todas las opciones que tenemos a la hora de administrar las imágenes desde el CLI, que se hace ejecutando el siguiente comando:

```
root@openstack ~(keystone_admin):~#   openstack image -h
```

```
root@openstack ~(keystone_admin):~#   openstack image create -h
```

En el caso concreto de subir la imagen previamente descargada, se podría hacer con ejecutando el comando de la siguiente forma, adaptándolo a las necesidades que requiera la imagen y el proyecto:

```
root@openstack ~(keystone_admin):~#   openstack image create \
debian --file /root/debian.img --min-ram 64 \
--min-disk 1 --disk-format qcow2
```

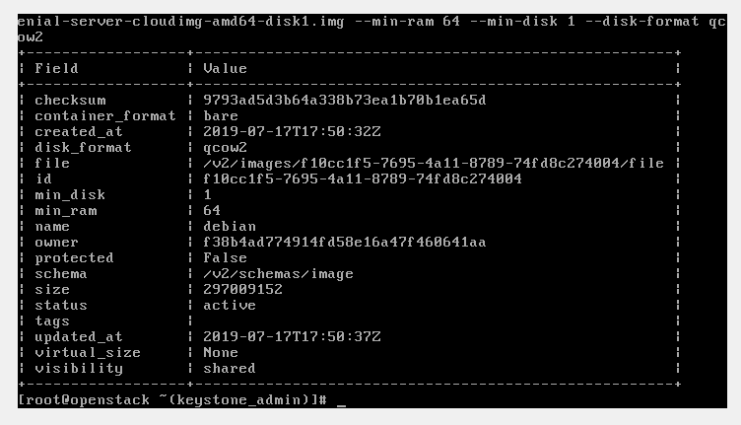

Figura 3.10: Creación imagen CLI

Que subiría a Openstack una imagen que hemos descargado previamente en el directorio /root con el nombre de debian.img, la crea con unos requisitos mínimos de ram (64MB) y de disco duro (1GB) y la crea en formato QCOW2. Es importante detallar que si la imagen ya se ha importado previamente desde el Dashboard, al importar la misma imagen desde el CLI posteriormente, originará un error. También es interesante la opción que hay en el CLI, a través de la cuál se puede descargar una imagen importada en Openstack, directamente al equipo o máquina, lo que se realizaría de la siguiente forma:

```
root@openstack ~(keystone_admin):~#   openstack imagen save \
nombre_imagen --file /ruta_descarga_imagen/nombre
```

3.2.2.2. Creación del sabor

Una vez tenemos la imagen, otro aspecto importante que es necesario crear o configurar, para lanzar las instancias posteriormente correctamente, se deberá crear un sabor que se ajuste a las necesidades que se desean para las instancias o usar uno de los sabores que vienen creados por defecto si ese se adecua a las necesidades. Un sabor como se ha explicado anteriormente es la cantidad de recursos con los que se lanza una instancia, y dependiendo de los recursos de los que se dispongan en el equipo que soporta el software y las máquinas virtuales de Openstack y de los que se hayan asignado a dicha maquina virtual puede ser que si se usa un sabor con mucho recurso computacional de error y no se puedan lanzar las instancias. Para este proyecto en el que se asignan 6GB de RAM a la maquina virtual que ejecuta OpenStack, se usará un sabor de estas características: Para crear dicho sabor desde el Dashboard hay que ir al menú de Administrador, y ahí seleccionar el submenú de Sabores. En esta sección aparecerá un botón "crear sabor" desde donde el cuál se podrá crear el sabor a las necesidades requeridas. Habrá que poner un nombre a dicho sabor, y como parámetros importantes los Mb de RAM y los Gb de disco duro.

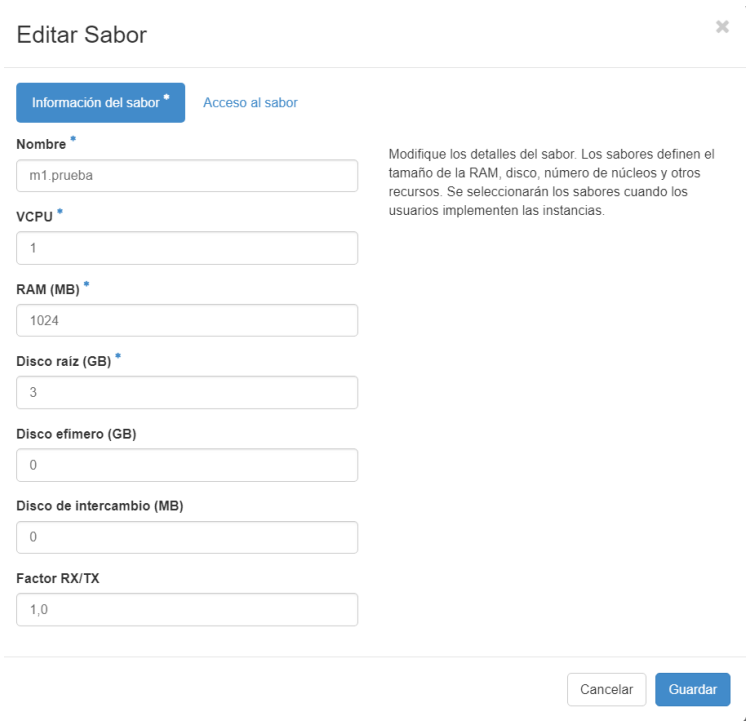

Figura 3.11: Creación sabor Dashboard

Desde el CLI, como se explica antes, habrá primero que loguearse si no se ha hecho previamente. Una vez logueados en el proyecto, es interesante conocer que opciones se tienen desde el CLI para administrar los sabores, que se visualizarán ejecutando:

```
root@openstack ~(keystone_admin):~#   openstack flavor -h
```

```
root@openstack ~(keystone_admin):~#   openstack flavor create -h
```

El segundo comando nos indicará las opciones y parámetros que se pueden definir a la hora de crear el sabor. Un ejemplo del comando para crear el sabor podría ser el siguiente:

```
root@openstack ~(keystone_admin):~#   openstack flavor create \
m1.sabor --vcpus 1 --ram 128 --disk 1 --id 8
```

Hay que destacar que el apartado de –ram se pone en MB y –disk en GB.

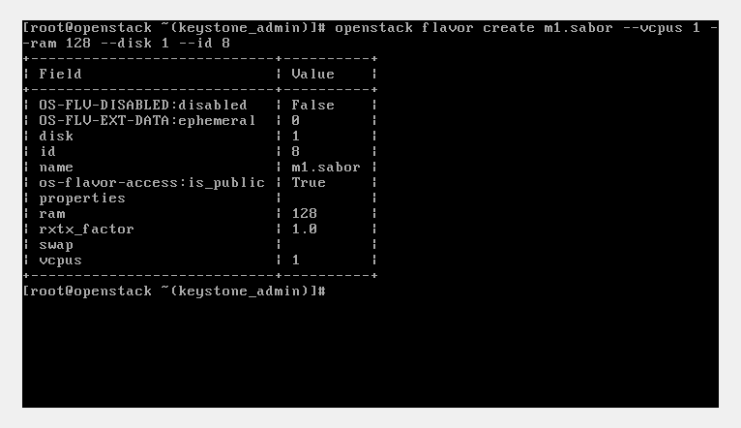

Figura 3.12: Creación sabor CLI

3.2.2.3. Creación de los Grupos de Seguridad

Otro aspecto importante que es necesario crear o configurar, para lanzar las instancias posteriormente correctamente y sin problemas es tener configurados los Grupos de Seguridad correctamente. Los Grupos de Seguridad son los que marcarán que conexiones se podrán realizar con la instancia, es decir, definen el tráfico entrante y saliente permitido en cada una de las instancias. Por defecto viene un grupo de seguridad creado llamado "default" pero es un grupo muy estricto, ya que bloquea todo el tráfico entrante a la instancia. Es por ello que se recomienda crear un Grupo desde cero. El apartado de los grupos de seguridad, se encuentra dentro del menú del Proyecto, en el apartado dedicado a la red. Ahí se mostrará una opción para crear un grupo de seguridad desde cero. En este caso concreto, se creará un Grupo de Seguridad que sea contrario al que viene por defecto, le llamaremos "permisivo" y será un Grupo de Seguridad que permita todo el tráfico entrante. Esta es una característica que a la hora de crear un entorno profesional, no es recomendable porque tener todo el tráfico entrante permitido puede generar problemas de seguridad, pero en este proyecto al trabajar con entornos de prueba no existe dicho problema. Para crear el grupo de seguridad que permita todo el tráfico entrante, se deberán definir las reglas de: "ALL ICMP, ALL TCP, ALL UDP,..." y algún caso particular del proyecto como por ejemplo la regla de SSH y HTTP, y cuando se elijan, no habrá que especificar ninguna IP, se dejarán todas las IPs como vienen por defecto, la 0.0.0.0.

Figura 3.13: Grupo Seguridad

A la hora de gestionar los grupos de seguridad, desde el Command Line es un poco más complejo y es que se necesitan conocer los puertos a través de los cuáles se quiere activar la comunicación. El procedimiento es idéntico al resto de configuraciones, donde se recomienda primeramente, ver que opciones existen para la gestión de los grupos de seguridad desde el CLI, haciendo uso del comando:

```
root@openstack ~(keystone_admin):~#   openstack security groups \
--h
```

Y repetir el sufijo "–h" en la opción deseada para observer los diferentes parametros que se requieren para ejecutar la orden correctamente. Por ejemplo para crear un nuevo grupo de seguridad y permitir todo el trafico tcp entrante, se ejecutarían las siguientes instrucciones:

```
root@openstack ~(keystone_admin):~#   openstack security group \
create permisivo
```

Donde permisivo sería el nombre del grupo de seguridad nuevo

```
root@openstack ~(keystone_admin):~#   openstack security group \
rule create --proto tcp permisivo
```

3.2.2.4. Creación de las IPs Flotantes

Una vez definidos los Grupos de Seguridad, quedaría crear las IPs flotantes. En este proyecto solo hace falta crear una. Haría falta crear una IP flotante por cada instancia que se desee que tenga conexión directa con el exterior, es decir con la red externa. La IP flotante es la IP con la que la instancia se conecta con dicha red, es decir, se la dotaría de una dirección IP adicional para tener conexión externa. Estas IPs pertenecen a la red externa y para crearlas habría que ir al menú de Proyecto y una vez ahí seleccionar el apartado de IP‘s flotantes y dar a “Asignar IP al Proyecto” en la parte superior derecha. Aparecerá un submenú en el que se podrá elegir entre las redes externas que se disponen dentro del Proyecto, si es que se dispone de mas de una, y se nos dará una IP perteneciente a dicha red, pero también se podrá editar esa IP y elegir una que se desee en concreto, pero siempre perteneciente a la red externa.

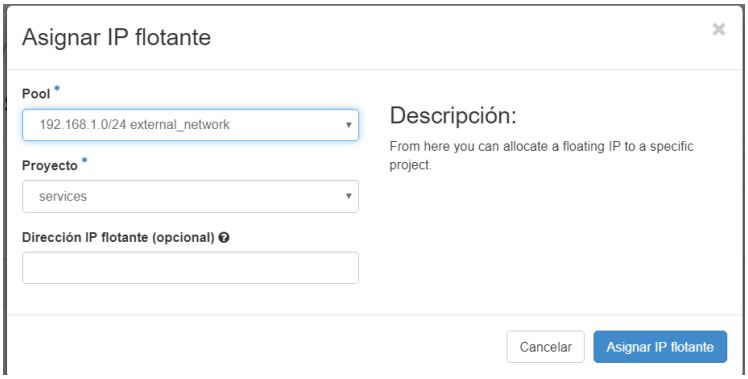

Figura 3.14: Creacion IP Flotante

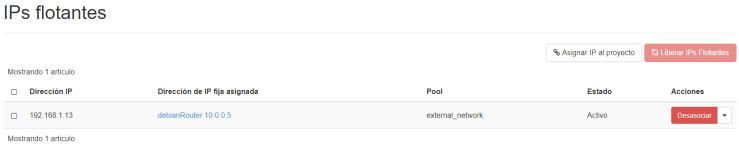

Figura 3.15: IP Flotante asignada

3.2.3. Lanzamiento de la instancia

Una vez se tienen todos los componentes necesarios para lanzar una instancia creados, se procede al lanzamiento de la misma o las mismas dependiendo de la topología. En este proyecto como se ha comentado anteriormente, se lanzará una instancia que hará las funciones de router y que habrá que configurar posteriormente, y luego se lanzará una o varias instancias, según se desee para comprobar que la instancia hace las veces de router y hay conectividad. En el caso particular de este proyecto, como se van a usar imágenes de Debian, de las cuáles no se conoce la contraseña para loguearse posteriormente, la única forma de hacerlo es a través de una conexión ssh, por lo que tanto al lanzarlas por el Dashboard como por el CLI, se deberán asignar un par de claves ssh a la instancia, y esas claves tenerlas almacenadas en el equipo desde el que se va a realizar el túnel ssh. De este aspecto se hablará a continuación. A la hora de lanzar ambos tipos de instancias, la que hará de router y las demás, el procedimiento es igual tanto por Dashboard como por CLI. Lo único que cambiará será el sabor que se elegirá para cada una, si en la instancia que hará las veces de router necesita una capacidad mayor que las demás. También posteriormente a dicha instancia habrá que asignarla una IP flotante para que tenga conectividad con la red externa.

Como se menciona anteriormente, se necesitará crear un par de claves ssh para cada instancia para poder acceder a ellas posteriormente mediante un túnel ssh, ya que en muchas imágenes no se dispone de la clave de acceso, como es el caso de las imágenes Debian que serán las usadas en este proyecto.

3.2.3.1. Lanzamiento de instancias a través del Dashboard

A la hora de lanzar la instancia mediante el Dashboard, se recomienda crear el par de claves antes de hacerlo, aunque es posible crearlas durante el lanzamiento de la misma. Para crear dicho par de claves, en el menú de Proyecto se accede al apartado de Red y una vez allí entramos al submenú Pares de Claves y allí se podrán crear. Para crearlas únicamente se tendrá que elegir el nombre que le asignaremos a las mismas dentro de Openstack, por lo que es recomendable elegir un nombre sencillo y que sea fácil saber a que instancia pertenece, como por ejemplo routerkey.pem. Cuando se crean las claves, se preguntará al usuario si desea guardar un archivo .pem, que se recomienda aceptar y guardarlas en una ubicación conocida porque posteriormente no será posible desde el Dashboard descargar dichas claves.

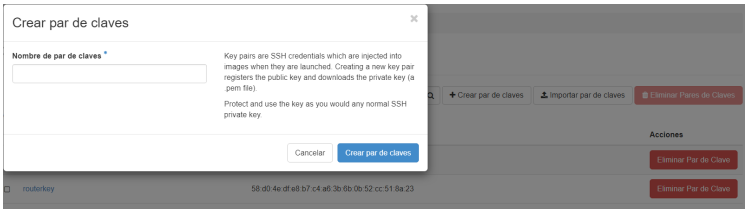

Figura 3.16: Creacion de claves Dashboard

Una vez se tiene el par de claves creadas, el proceso para lanzar la instancia es igual para cualquier caso. Se deberá acceder al menú Proyecto y una vez allí entrar al menú Compute. Accediendo al submenú Instancias, a la derecha aparecerá la opción de “Crear Instancia” y si accedemos a ella aparecerá una ventana donde se podrán especificar los diferentes aspectos que tendrá la instancia a lanzar. Se definirá el nombre de la instancia, donde se recomienda poner un nombre que diferencie las distintas instancias, para reconocer su función dentro de la topología de red desplegada, y si se accede desde el CLI conocer cada instancia por el nombre. En el caso particular de este proyecto, por ejemplo para la instancia que hará de router, al lanzarse a partir de una imagen debían, se llamará debían_router.

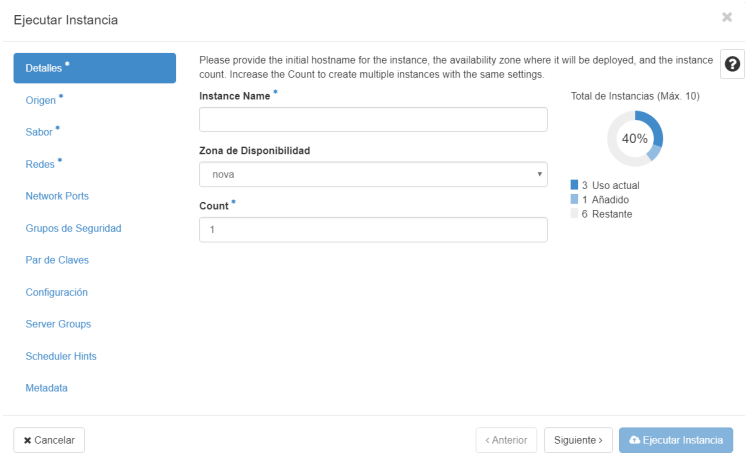

Figura 3.17: Lanzamiento de la instancia: eleccion del nombre

Otro aspecto que se deberá elegir será la imagen a partir de la cual se lanzará la instancia. Nos aparecerá en la parte inferior una lista con todas las imágenes que se tienen disponibles y se deberá seleccionar la indicada pulsando en la flecha que se tiene en la parte derecha de cada imagen. Posteriormente se definirá el sabor. Es posible que ciertos sabores tengan un triángulo de advertencia, esto significa que, en la imagen escogida anteriormente, a la hora de crearla se definieron unos requisitos mínimos y dichos sabores no cumplen esos requisitos mínimos.

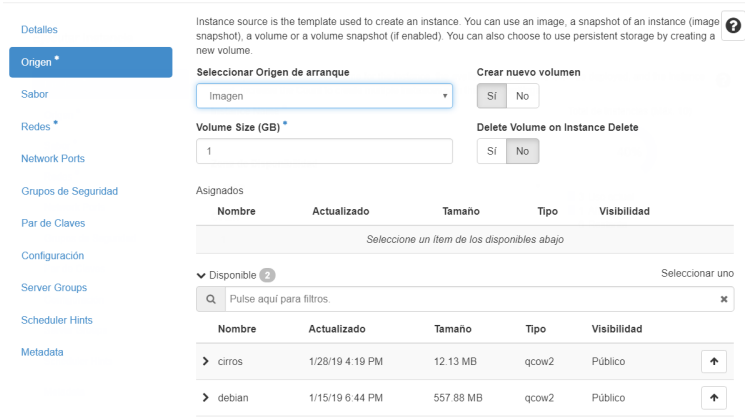

Figura 3.18: Lanzamiento de la instancia: eleccion de la imagen

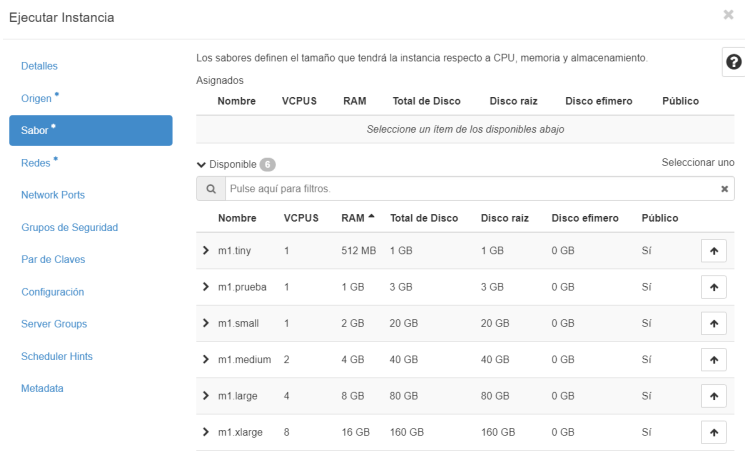

Figura 3.19: Lanzamiento de la instancia: eleccion de la sabor

Una vez elegido el sabor correspondiente, habrá que seleccionar la red o redes a las que estará conectada la instancia. Aquí hay distinción entre la instancia que haga de router y las demás,ya que la instancia que haga de router deberá estar conectada a dos redes, y las demás solo a una, en el caso particular de la topología de red de este proyecto. A la hora de escoger las dos redes, para que quede la topología más estética, se recomienda que la primera red seleccionada sea la que la instancia tenga a la izquierda, porque la primera red seleccionada se le asignará al primer puerto de la instancia que es el que tiene a la izquierda. Esta red será la red que en este proyecto se ha denominado Red_Interna, que es la que esta conectada al router software. En los casos de las otras instancias al sólo haber una red a la que se conectarán, no se tendrá este problema. La instancia que hace de router, como segunda red a la que conectarse, será la llamada Red_Cloud, que es a la que se conectarán también las demás instancias.

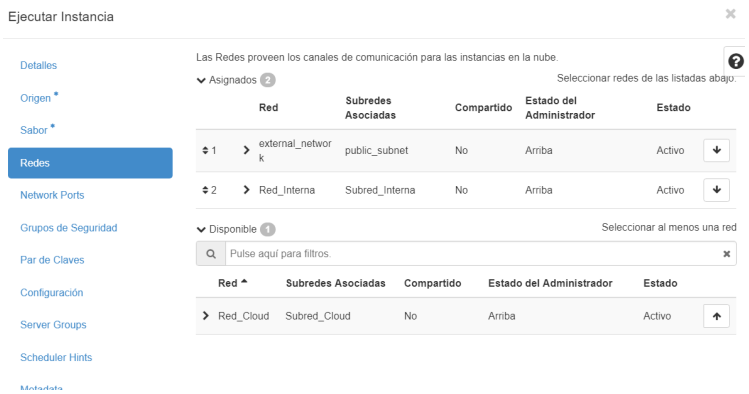

Figura 3.20: Lanzamiento de la instancia: eleccion de redes

Para finalizar con el lanzamiento de la instancia, habrá que seleccionar el Grupo de Seguridad de la misma. Vendrá el grupo “default” seleccionado por defecto, por lo que se deberá dar a la flecha de la derecha para bajarle a la lista de todos los grupos de seguridad, y ahí seleccionar el Grupo de Seguridad que se ha creado anteriormente, que se denominó “permisivo”.

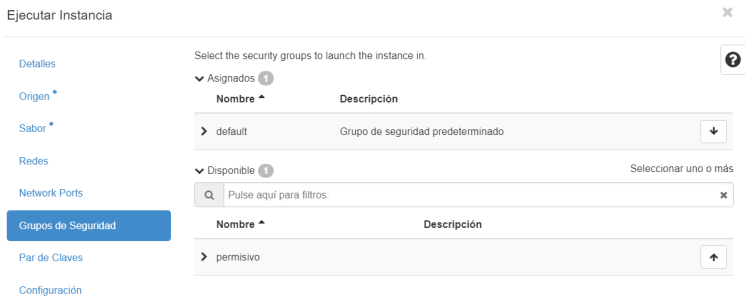

Figura 3.21: Lanzamiento de la instancia: eleccion de grupo de seguridad

Por útimo se deberá elegir el par de claves que se asignará a dicha instancia y una vez hecho, finalizamos el proceso y se lanzará la instancia. Las instancias de la red, las que no se comportarán como un router, se lanzarán de la misma forma con la salvedad de que sólo se asignarán a la Red Cloud. El resto del proceso es idéntico. Un error común que suele aparecer una vez se está lanzando la instancia, es que aparezca un mensaje de que no hay suficientes servidores o host disponibles y se aborta el lanzamiento de la misma. Esto suele suceder porque el sabor asignado a la instancia es computacionalmente muy pesado para los Gb de RAM que se han definido para la Máquina Virtual desde la que se está hosteando el software Openstack.

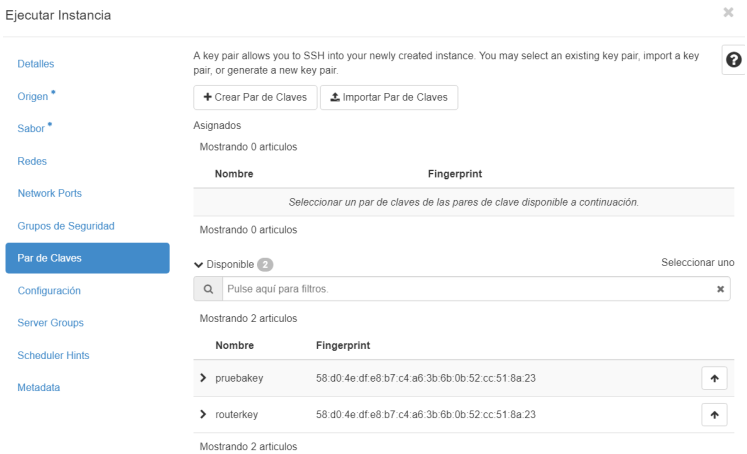

Figura 3.22: Lanzamiento de la instancia: eleccion de claves

3.2.3.2. Creación de túnel SSH a la instancia desde el equipo anfitrión.

Una vez tenemos las instancias lanzadas por Dashboard, es momento de tratar la conexión mediante un túnel SSH hacia ellas. Para ello se hará uso del software Putty, que permite establecer una conexión SSH con una clave elegida por el usuario. Lo primero que hay que realizar es ir a PuttyGen y darle a cargar para usar así un archivo .pem ya existente para generar la clave. Para cargar la clave existente se deberá ir a la ubicación que el usuario ha elegido que sea en la cuál se guardasen los pares de claves creados y allí seleccionar el archivo deseado. Una vez se ha cargado el archivo, se podrá elegir un comentario e incluso una "passphrase" para la contraseña si así se desea. Guardando la clave privada en una ubicación segura, se terminaría el proceso en Puttygen [8].

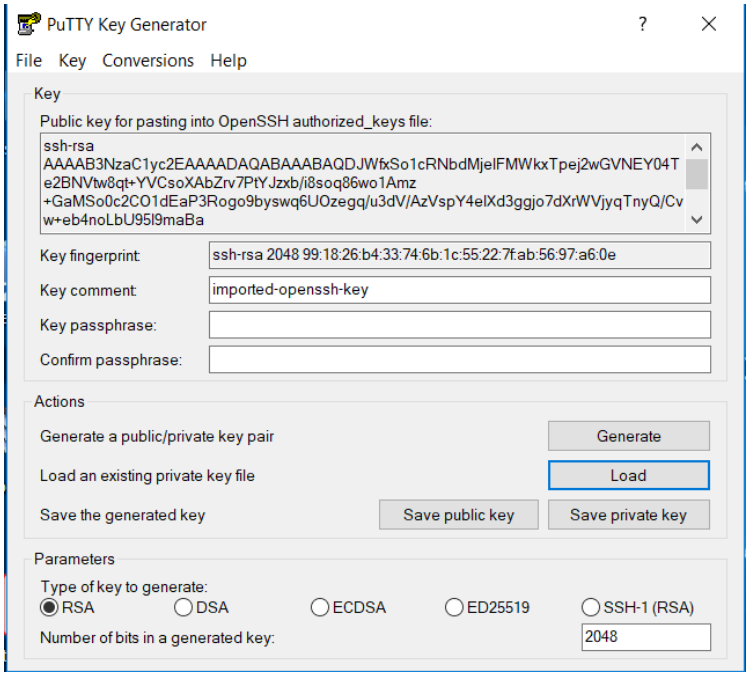

Figura 3.23: Generacion clave en Puttygen

Ahora queda seleccionar la clave que hemos guardado en Putty para asociarla con la instancia a la que se desea acceder. Para ello, hay que acceder a Putty y en los menús de la izquierda hay que desplegar el de SSH y una vez ahí entrar al submenú Auth. En este submenú se mostrará una opción para cargar una clave privada para la autenticación, y es en este cuadro de diálogo donde deberemos buscar y cargar la clave privada descargada. Una vez se ha asociado la clave correcta, hay que volver a los menús de la izquierda y en Session indicar los parámetros necesarios para conectar con la instancia, esto es, en la parte de Host Name se deberá escribir el nombre de la instancia seguido de la ip flotante asignada a esta misma.

Por ejemplo una dirección válida sería “debianRouter@192.168.1.113”. Si toda la información es correcta, se podrá iniciar la conexión después de dar permisos a la aplicación porque en la primera conexión al no haber sido registrada previamente, saldrán unos mensajes de aviso.

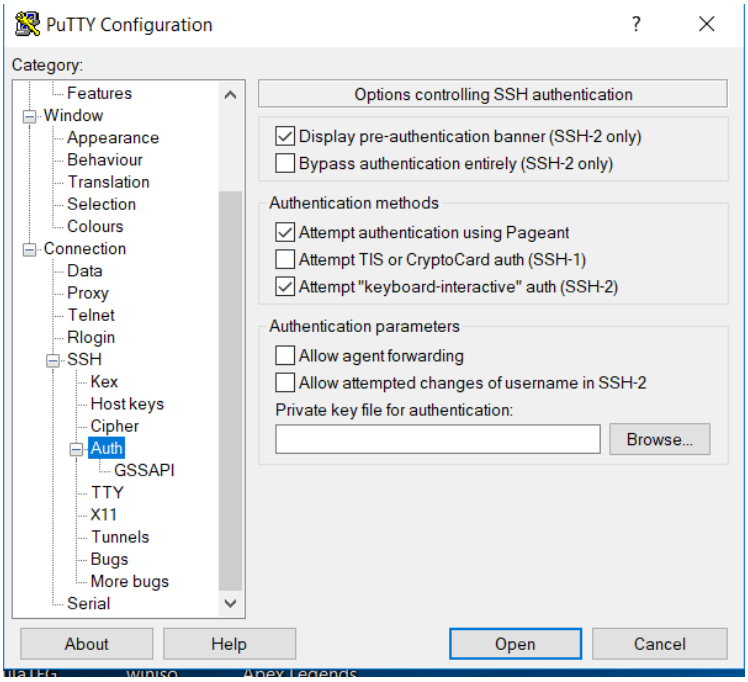

Figura 3.24: Configuracion Putty

Esta forma de acceder es solo válida para la instancia router que es la instancia que tiene asignada una Ip Flotante, las otras instancias que darán forma a la red, no son accesibles por este método, desde el Host, sino que habrá que acceder a ellas mediante la consola de la instancia que haga de router, realizando un SSH, o también será posible acceder desde la misma consola de Openstack si es que se conocen las credenciales de acceso de las imágenes usadas.

3.2.3.3. Lanzamiento de instancias a través del CLI

Para lanzar y conectarse a las instancias desde el CLI, el proceso será más rápido que desde el Dashboard pero menos visual. Al igual que desde el Dashboard, se tiene que crear un par de claves ssh con los que conectarse posteriormente a las instancias. Este proceso es sencillo, pero hay que saber en todo momento donde se guarda dicha clave y como se llama, porque una vez lanzada una instancia asignada a dicha clave, solo se podrá acceder a ella si se usa la clave correcta, en otro caso será imposible y se debería eliminar y volver a lanzar de nuevo la instancia. Para crear el par de claves desde el CLI, primero habrá que dirigirse al directorio en el cuál se guardan las claves ssh y después, se deberá crear el par de claves ssh, como se haría en cualquier otro caso. Para ello se deberá lanzar los siguientes comandos:

```
root@openstack:~$    cd ~/.ssh
```

```
root@openstack .ssh(keystone_admin):~$    ssh-keygen -t rsa
```

Al lanzarlo, se mostrará un mensaje en el que se pedirá introducir la ubicación en la cuál se quiere que se alojen las claves privada y pública. La consola te da una ruta por defecto, concretamente /home/user/ .ssh/clave donde clave será el nombre que se le asignará a el par de claves, tanto a la privada como a la pública, pero esta última con la extension .pub. La ruta por defecto será la que se elegirá en este proyecto para alojar dicho par de claves. Una vez se elige la ubicación, aparecerá un cuadro de texto en el cuál podemos elegir una contraseña escrita, pero se puede omitir si se deja el campo de texto vacío.

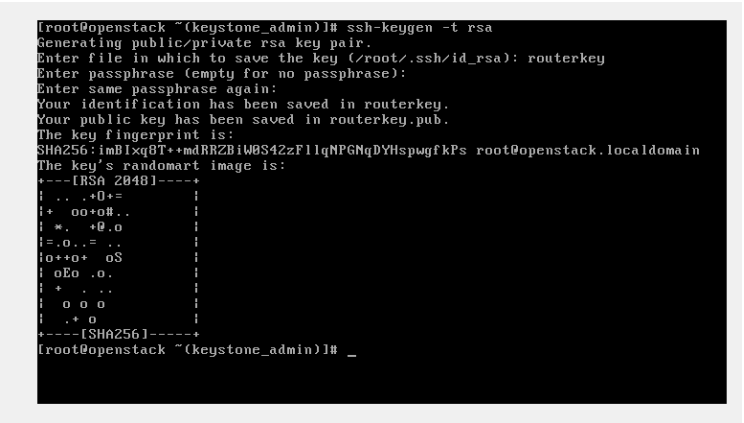

Figura 3.25: Creacion claves CLI

En este punto se tendría el par de claves en el host. Se podría acceder mediante un túnel ssh con la instancia con la cuál se tiene conexión, es decir, la instancia que hará en un futuro de router. Para acceder mediante ssh a una instancia, a la hora de lanzarla hay que asignarle el par de claves correspondiente, por lo que en este instante, se tienen todos los requisitos necesarios para lanzar las instancias de una forma óptima. Al igual que desde el Dashboard, desde el CLI todas las instancias se lanzaran de igual manera, con la salvedad de que habrá que especificar los requerimientos que se desean para cada una de ellas. Para lanzar las instancias, previamente hay que loguearse si no se ha hecho previamente, con las credenciales del archivo keystonerc_admin y después habrá que hacerlo ejecutando un comando parecido al siguiente, que será un ejemplo para lanzar la instancia que hará de router:

```
root@openstack ~(keystone_admin):~$   openstack server create --h
```

```
root@openstack ~(keystone_admin):~$   openstack server create \
nombre_instancia --nic net-id=id_red --flavor nombre_sabor \
--image nombre_imagen --key-name nombre_clave_publica \
--security-group grupo_seguridad
```

Donde el net-id es el identificador de la red a la que se quiere que se asocie la instancia, que se puede visualizar ejecutando ***openstack network list****.*

Cuando se ejecuta el comando, saldrá un recuadro en el que se mostrarán todos los aspectos que se han elegido para dicha instancia.

Field	Value
OS-DCF:diskConfig	MANUAL
OS-EXT-AZ:availability_zone	
OS-EXT-SRV-ATTR:host	None
OS-EXT-SRV-ATTR:hypervisor_hostname	None
OS-EXT-SRV-ATTR:instance_name	
OS-EXT-STS:power_state	NOSTATE
OS-EXT-STS:task_state	scheduling
OS-EXT-STS:vm_state	building
OS-SRV-USG:launched_at	None
OS-SRV-USG:terminated_at	None
accessIPv4	
accessIPv6	
addresses	
adminPass	UXNB5xjJYvTt
config_drive	
created	2019-07-17T18:01:14Z
flavor	m1.sabor (8)
hostId	
id	1f0a08d8-d98b-4a2f-a392-53aaa1f5f1c4
image	debian (286c6a71-d4f6-449f-b8a2-f2ec2c4481f7)
key_name	routerkey
name	debian_router

Figura 3.26: Lanzamiento instancia CLI

En el caso específico de la instancia que hace de router, habrá que asignarla a otra red, la que la conectará con las demás instancias. Esto se puede hacer, a la hora de ejecutar el comando, añadir otro net-id=id_red2, y de esta forma la instancia quedaría asignada a las dos redes. Por último para que pueda ser accesible desde la red externa, hay que proporcinarle una ip flotante, y para ello habrá primero que crear una ip flotante y posteriormente asignársela a la instancia. Para ello habrá que ejecutar los siguientes comandos:

```
root@openstack ~(keystone_admin):~$   openstack floating ip \
create nombre_red_externa
```

Este comando crea la ip flotante y muestra la direccion ip

```
root@openstack ~(keystone_admin):~$   openstack server add \
floating ip nombre_instancia ip_flotante
```

Esa ip es la que se introduce en este comando

Una vez tenemos asignada la ip flotante a la instancia, ya podemos acceder a ella a través de un túnel SSH desde el Host.

3.2.3.4. Conexión mediante túnel SSH desde el host

Para ello, se deberá cambiar el directorio al mismo en el cuál se guardan las claves ssh, en el cuál estarán el par de claves con los cuáles se lanzó dicha instancia. Una vez en el directorio, se ejecutará el túnel SSH y se podrá acceder a la instancia desde el Host.

```
root@openstack:~$    cd ~/.ssh
```

```
root@openstack .ssh(keystone_admin):~$    ssh -I id_rsa \
debian@ip_flotante
```

Donde id_rsa será la clave con la que se lanza la instancia debian@ip_flotante será el nombre@ip_flotante asignada

Si se ha seguido todo el proceso correctamente, deberá cambiar el Terminal y se habrá accedido a la instancia que en este caso hará de router.

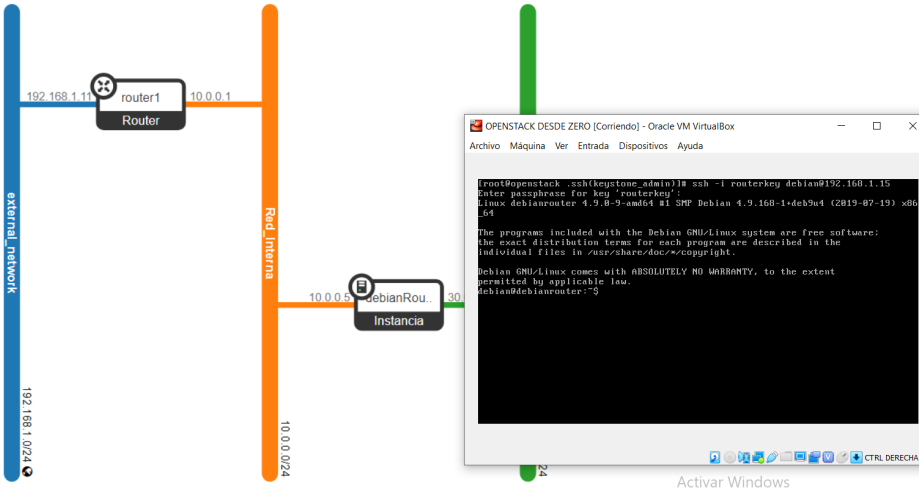

Figura 3.27: Creacion del tunel SSH

El acceso a la instancia, como se ha explicado anteriormente, deberá hacerse desde la consola del Host, a menos que se disponga del user y password de la imagen desde la que se ha creado dicha instancia. Si no se dispone de dicha información la única manera de acceder será mediante la tecnología SSH.

3.3. Configuración de las instancias

3.3.1. Configurar instancia para que actúe como un router

Neutron es un proyecto de Openstack que porporciona redes como un servicio, es decir "Networking as a Service" a los demás componentes de Openstack. Este componente, permite al usuario crear o usar cualquier elemento de red, ya sean redes, subredes, routers, etc. Los routers de los que dispone Openstack, gestionados por Neutron, son routers software y los usuarios no tienen acceso directo a ellos. Esto genera un problema a la hora de gestonar una red dimensionada por Openstack, ya que la única información que se mostrará de los routers que esten en la topología de red, será si están up o down. Por lo que se hace interesante, poder usar una máquina virtual, es decir una instancia, que realice la función de router e interconecte las diferentes redes, ya que el usuario si dispone de más información de las instancias como se explica en [9]. Para poder tener una instancia que haga la función de router, necesitaremos que dicha instancia se conecte a una red externa, desde la que se accederá desde el host a la red cloud, y a una red interna, que será a la que se acceda a través de la instancia. Como se ha explicado y realizado anteriormente, la instancia que hará de router, al necesitar conectarse con la red externa, necesita una IP flotante mediante la cuál salir al exterior de la red cloud. Cuando se tiene todo listo, hay que loguearse en la instancia para configurar una serie de parámetros para permitir la comunicación de datos. Una vez se ha logueado en la instancia, lo primero que se realizará será ver la tabla de rutas de dicha instancia, mediante la ejecución del comando:

```
debian@debianrouter:~$   sudo ip r
```

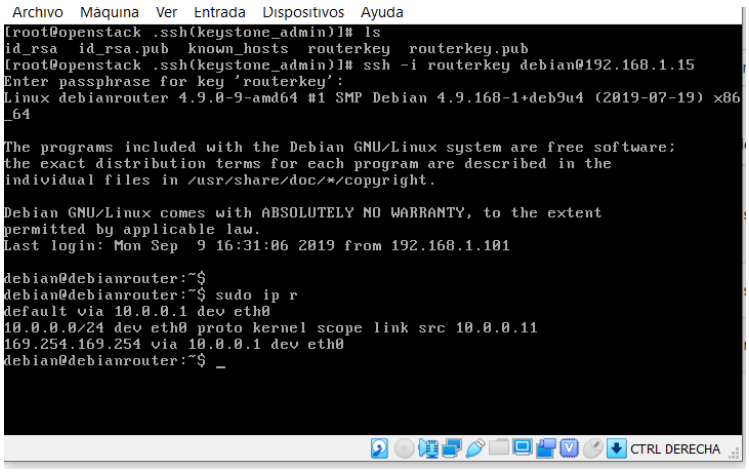

Figura 3.28: Comando sudo ip r

Se mostrará la tabla de rutas que indicará las conexiones de dicha instancia. Además aparecerá una linea adicional, al final de la table de rutas, con un aspecto semejante al siguiente, "169.254.169.254 vía 10.0.0.1 dev eth0", que aparece debido a que en el agente DHCP de Neutron, viene una opción activada por defecto que es el "enable_isolated_metadata", que permite acceder al servidor de metadatos desde una red interna. Como en el caso de este proyecto, esta instancia es capaz de acceder a dicho servidor de metadatos a través de la interfaz eth0 esta última regla no es necesaria, por lo que se deberá reconfigurar. También cambiaremos la configuración de la red a una configuración estática.

Para ello:

```
debian@debianrouter:~$   sudo ifdown eth1
```

Desactivamos la interfaz que queremos modificar

Una vez desactivada la interfaz, se deberá ir al archive de configuración de dicha interfaz y hacer los siguientes cambios. Este archivo de configuración en este caso concreto en el que se usa una imagen Debian, estará en la ruta /etc/network/interfaces. Aquí añadiremos en la línea de la interfaz correspondiente los siguientes aspectos (con la dirección IP correspondiente en este caso concreto):

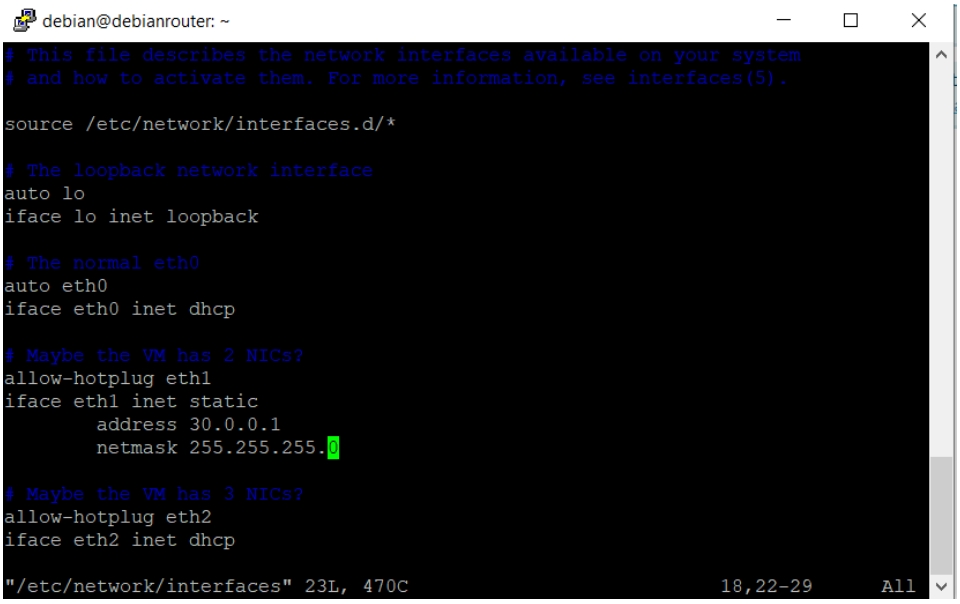

Figura 3.29: Interfaces de red

Cuando se ha realizado la modificación en el archive de configuración, y se ha guardado correctamente, se deberá volver a activar la interfaz para que los cambios resulten efecto. Para ello, se volverá a visualizar la tabla de rutas para ver que dichos cambios se han ejecutado correctamente.

Se observa que la línea innecesaria ha desaparecido y que las direcciones IP se corresponden con las que se tienen en el proyeto. Ahora habra que activar el bit de forwarding, para que la instancia pueda ejercer de router. Para ello, se deberán ejecutar los siguientes comandos:

```
debian@debianrouter:~$   sudo sed -i \
'/^#net.ipv4.ip\_forward/s/^#//g' /etc/sysctl.conf
```

```
debian@debianrouter:~$  sudo sysctl -p /etc/sysctl.conf
```

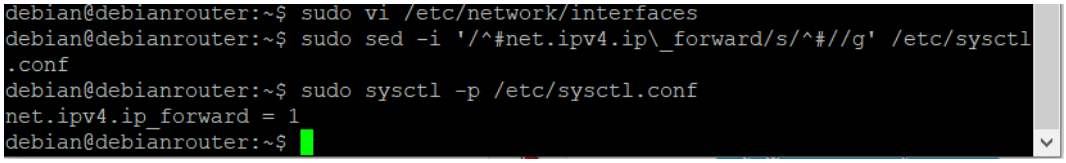

Figura 3.30: Comando bit forwarding

Una vez ejecutados, la configuración en la instancia se ha finalizado. En este punto, toca modificar las instancias lanzadas al otro lado de la red interna, que serán las instancias a las cuáles podremos acceder desde la instancia router y por ende desde el host.

3.3.1.1. Configuración de las instancias internas

Como se ha explicado, se desea acceder a las instancias de la red interna desde el exterior, es decir, desde el host. Estas instancias al no poder disponer de una IP flotante, se hace necesario un salto intermedio para poder llegar a ellas, y este salto se realiza mediante la instancia router configurada anteriormente. Por ello hay que configurar la table de rutas de dichas instancias para que tengan conectividad con el exterior. Para configurar estas instancias, primeramente habrá que acceder a ellas a través de la instancia router, mediante otro túnel SSH, es decir hacer un túnel SSH dentro de otro túnel SSH.

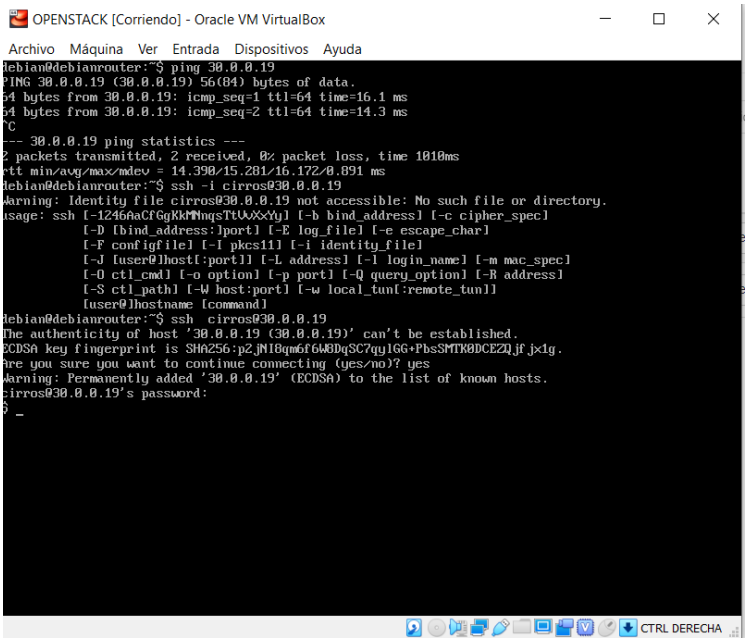

Figura 3.31: SSH instancia cloud

Una vez se ha accedido a dicha instancia, se procede a configurarla.

```
cirros@30.0.0.19:~$  sudo ip r
```

Visualizando la tabla de rutas de estas instancias, se observa que no tienen una regla de enrutamiento por defecto, ya que no fue definida cuando se creó la subred. Por lo que se deberá marcar como gateway de dichas instancias a la instancia router. Para ello se ejecutará el siguiente comando:

```
cirros@30.0.0.19:~$  sudo ip r add default via 30.0.0.1
```

Figura 3.32: Definición del gateway de la instancia interna

Por ultimo, para conectar la red interna al router software de Openstack, el que interconecta la instancia router con la red externa, es necesario definir una regla de enrutamiento en el router software que le permita conectarse a la red interna a través de la instancia router:

```
root@openstack:~$  neutron router-update \
3e98276c-f107-48a5-97de-f144e10df3df --routes type=dict \
list=true destination=30.0.0.0/24,nexthop=10.0.0.5
```

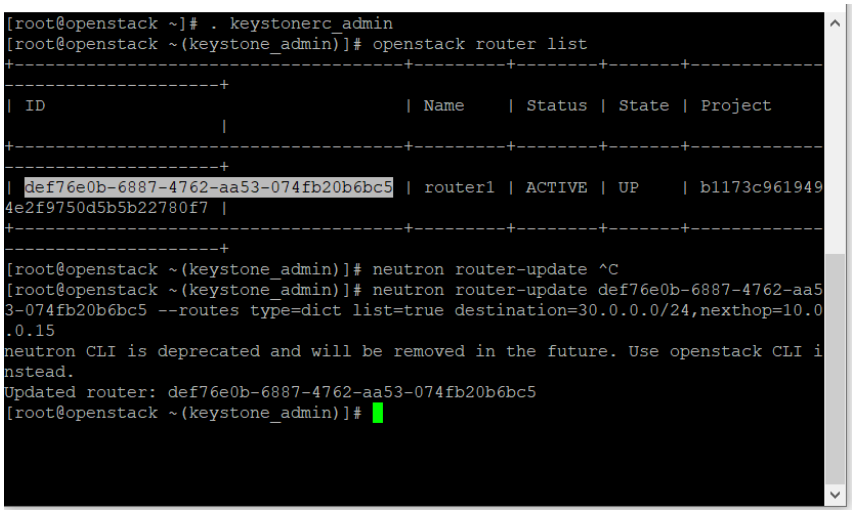

Figura 3.33: Regla enrutamiento para el router

Una vez ejecutado el comando y creada la regla, surge un problema y es que el router software no responde al ping enviado por la instancia interna. Para ver donde falla el ping, una manera fácil de reaizarlo es hacer un tcpdump en la instancia router a su interfaz conectada a la red interna, para ver donde está ell problema:

```
debian@debianrouter:~$  sudo tcpdump -i eth1 icmp
```

El ping se observa que llega a la instancia router a través de eth1 y lo reenvía al router software, que gracias a la regla definida puede contestar a través de la instancia router. La respuesta llega a la instancia router pero nunca llega a la instancia interna debido a que Openstack tiene unas reglas de anti-spoofing (contra la suplantación de identidad) en las tablas de rutas que se añaden automáticamente a todas las instancias. Esta política anti-spoofing no permite usar instancias como routers. Esta regla se puede ver, ejecutando el siguiente comando:

```
root@openstack:~$  sudo iptables -L neutron-openvswi-sebf586a5-a \
-n -v
```

Donde "ebf586a5-a" es el identificador de Puerto que se corresponde con la interfaz eth1. Openstack define para cada instancia lanzada una cadena con la forma neutrón-openvswi-s(identificador de puerto) para cada puerto, por el que todo el tráfico saliente de IP diferente a la de la instancia va por ahí. Para obtener el identificador de puerto, se podrá visualizar ejecutando:

```
root@openstack:~$  openstack port list
```

Una vez ejecutado, aparecerá una cadena larga de caracteres para cada puerto. El identificador de puerto con el que se debe de quedar para ejecutar el comando anterior serán los 10 primeros dígitos del uuid que se muestra. Como se ha comentado anteriormente, la regla anti-spoofing, no permite usar instancias como routers, pero Neutron permite añadir direcciones IP concretas a un puerto gracias a la regla "alowed_address_pairs", y se consigue añadir una tabla de rutas en la regla de anti-spoofing que permitirá que se reconozca a la instancia router como un router, permitiéndola reenviar datos.

```
root@openstack:~$  neutron port-update \
ebf586a5-a31a-4ec5-a550-76e530da1260 --allowed-address-pairs \
type=dict list=true \
mac_address=fa:16:3e:42:a1:c4,ip_address=10.0.0.0/24
```

Si volvemos a observar la regla contra la suplantación de identidad, se podrá observar que aparece una línea adicional que antes no se observaba.

```
root@openstack:~$  iptables -L neutron-openvswi-sebf586a5-a -n -v
```

Una vez hecho esto, se puede comprobar que ahora sí, el ping vuelve a la instancia interna, solventando el problema originado anteriormente.

```
debian@debianrouter:~$  ping -c 1 10.0.0.1
```

3.4. Instalación y configuración de Nagios

La gran desventaja de las redes cloud, es que hoy por hoy carecen de herramientas completas de gestión de red, nativas de la nube, es decir, hay que integrar una herramienta no pensada para la gestión de la nube en ella. Un gran problema que surge es que muchos elementos como por ejemplo los routers o encaminadores que proporcionan los proveedores cloud, son elementos software, por lo que estas herramientas de red no pueden dar apenas información sobre el envío de paquetes que se realiza a traves de estos elementos.

En este trabajo se proporciona una red, pensada para que sea una máquina virtual la que haga de router y por tanto estas herramientas de gestión de red, como por ejemplo Nagios, puedan ser capaces de proporcionar mas información clave e importante para el usuario.

Por ello como parte final, se integrará la herramienta Nagios, en dicha red para observar que elementos se pueden monitorear y que información se da acerca de los mismos.

Esta herramienta, se puede configurar que se instale de forma automática con la instalación de la distribución que se instale de Openstack, modificando el archivo de configuración que se recibe al descargar el instalador, en el cuál se encuentra una opción que será "install_nagios=n". Modificando la n por una y, al ejecutar el instalador se instalará también Nagios.

En el caso concreto de este proyecto, no se activó dicha opción para poder instalar desde cero la herramienta de monitoreo, con los paquetes y plugins que se deseaban [10] Para instalar Nagios en Centos, primero habrá que instalar unos paquetes necesarios, para ello:

```
root@openstack:~$ yum -y install httpd php gcc glibc glibc-common \
wget perl gd gd-devel unzip zip
```

Una vez instalados, hay que crear un usuario y un grupo de usuarios que tengan permiso para poder ejecutar los comandos externos a través de la interfaz web:

```
root@openstack:~$ useradd nagios
```

```
root@openstack:~$ groupadd nagcmd
```

```
root@openstack:~$ usermod -a -G nagcmd nagios
```

```
root@openstack:~$ usermod -a -G nagcmd apache
```

Con esto se tiene todo listo para descargar e instalar Nagios Server para empezar a monitorear los equipos. Para hacerlo, habrá que ejecutar:

```
root@openstack:~$ cd ~
```

```
root@openstack ~:~$ curl -L -O \
https://assets.nagios.com/downloads/nagioscore/nagios-4.4.3.tar.gz
```

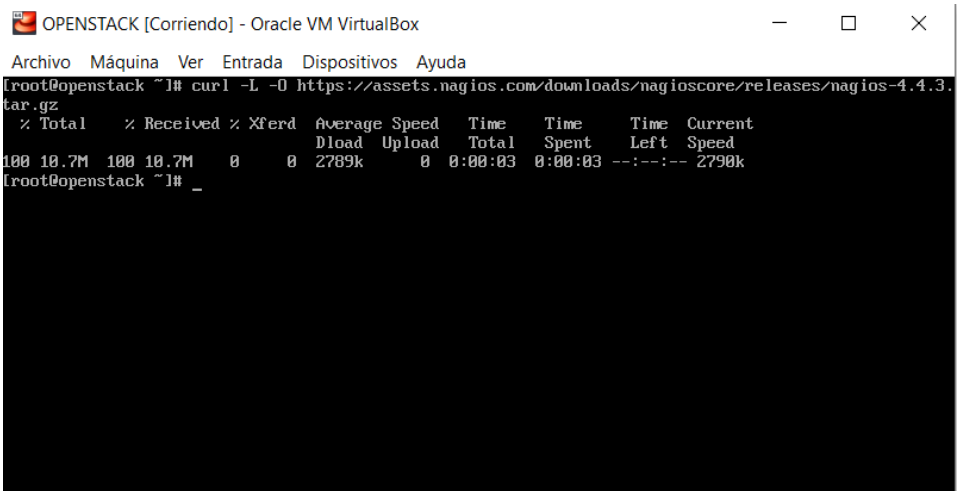

Figura 3.34: Instalación Nagios Server

Una vez descargado, hay que descomprimirlo y compilarlo:

```
root@openstack ~:~$ tar -zxvf nagios-4.4.3.tar.gz
```

```
root@openstack ~:~$ cd nagios-4.4.3
```

```
root@openstack ~/nagios-4.4.3:~$ \
./configure --with-nagios-group=nagios --with-command-group=nagcmd
```

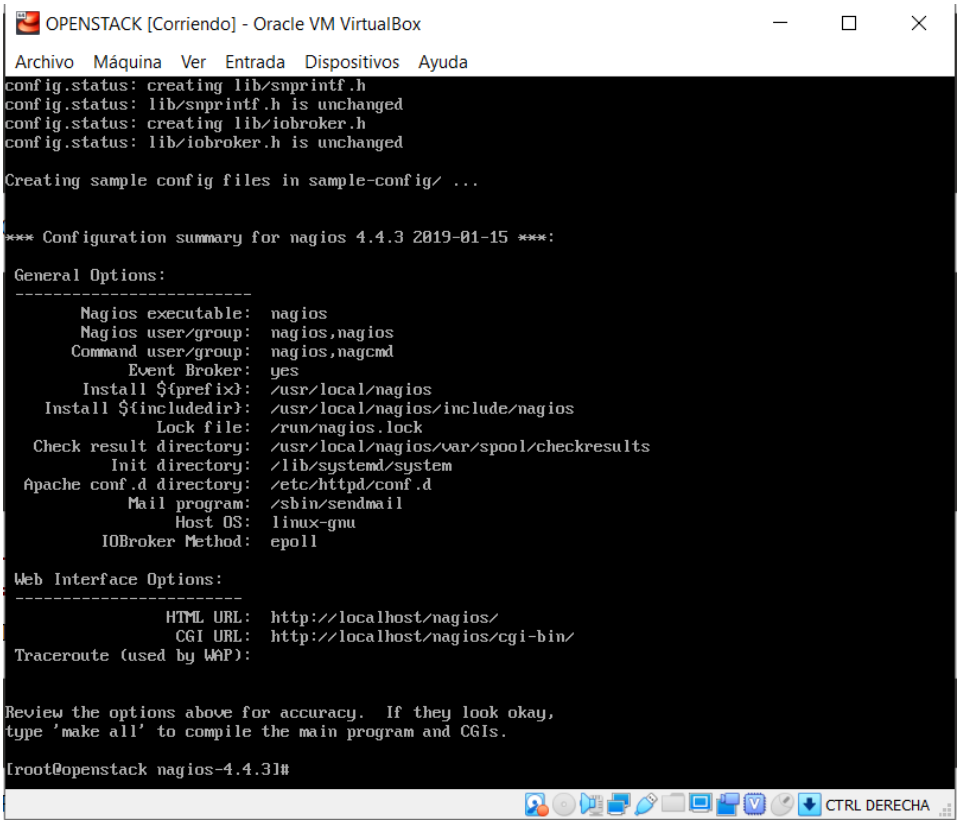

Figura 3.35: Compilación Nagios

Una vez compilado, habrá que instalarlo:

```
root@openstack nagios-4.4.3:~$ make all
```

```
root@openstack nagios-4.4.3:~$ make install
```

```
root@openstack nagios-4.4.3:~$ make install-init
```

```
root@openstack nagios-4.4.3:~$ make install-config
```

```
root@openstack nagios-4.4.3:~$ make install-commandmode
```

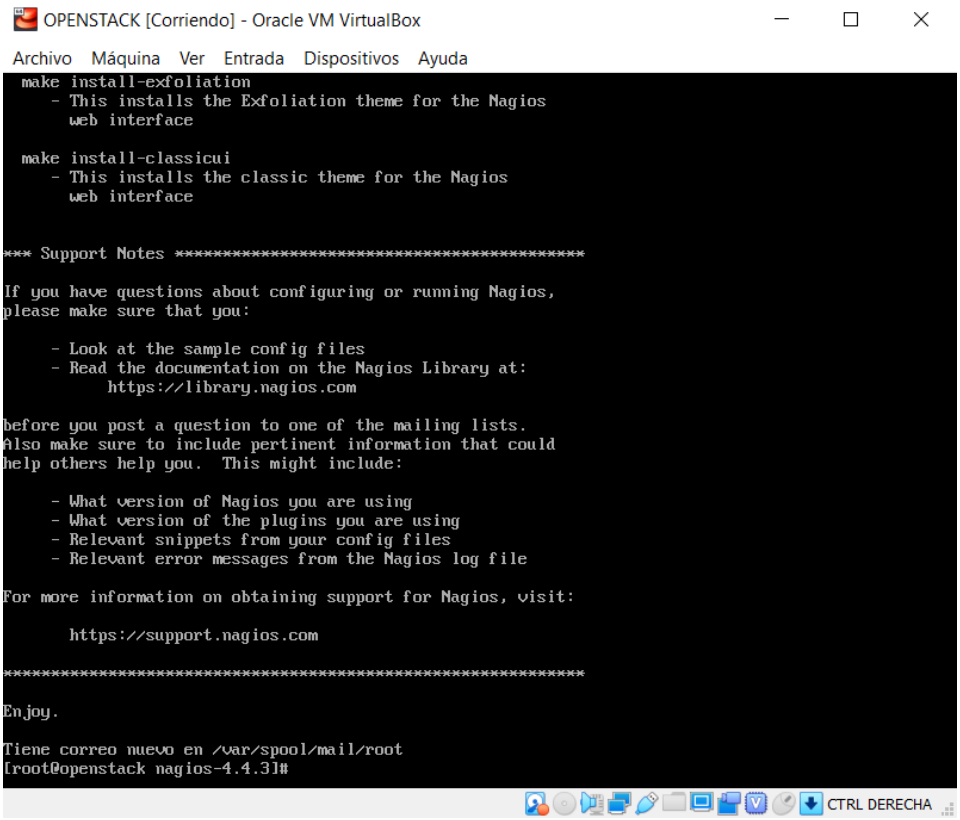

Figura 3.36: Instalacion Nagios

En este punto se tendría Nagios completamente instalado. Por lo que se va a instalar la configuración web, para ello:

```
root@openstack:~$ make install-webconf
```

Y para acceder a la interfaz web hay que crear un usuario, que se llamará "nagiosadmin". Para ello se ejecutará el siguiente comando, que nos pedirá también una contraseña.

```
root@openstack:~$ htpasswd -c \
/usr/local/nagios/etc/htpasswd.users nagiosadmin
```

Una vez hecho, hay que reiniciar el servicio apache para que los cambios tengan efecto. Como estamos trabajando sobre Centos 7, se haría de la siguiente manera:

```
root@openstack:~$  systemctl restart httpd
```

```
root@openstack:~$ systemctl enable httpd
```

En el mismo directorio donde se ha descargado e instalado Nagios, hay que descargar los plugins de Nagios, ya que si no se instalan estos plugins, al acceder via web a Nagios, se mostrarán varios errores en los dispositivos a monitorear. Para descargarlos e instalarlos, hay que seguir el mismo procedimiento que con Nagios, primero se descarga y se descomprime, y después en la carpeta que se genera, se compila e instala:

```
root@openstack:~$ wget \
https://nagios-plugins.org/download/nagios-plugins-2.2.1.tar.gz
```

```
root@openstack:~$ tar -zxvf nagios-plugins-2.2.1.tar.gz
```

```
root@openstack:~$ cd /nagios-plugins-2.2.1/
```

Una vez se ha descargado y descomprimido, se procede a compilarlo e instalarlo:

```
root@openstack /nagios-plugins-2.2.1:~$ ./configure \
--with-nagios-user=nagios --with-nagios-group=nagios
```

```
root@openstack /nagios-plugins-2.2.1:~$ make
```

```
root@openstack /nagios-plugins-2.2.1:~$ make install
```

Una vez se tiene todo completamente instalado, es turno de configurar el servidor de Nagios. Se deberán modificar varios archivos de configuración manualmente, para que se monitoree los dispositivos y elementos que se desean como se explica en [11]. Lo primero será activar la configuración de los servidores, es decir de las máquinas a monitorear. Para ello se deberá acceder al archivo de configuración de Nagios, ubicado en la siguiente ruta: /usr/local/nagios/etc y ahí se encontrará el archivo nagios.cfg, en el que se deberá descomentar (quitar el #) la línea que marca la ruta del directorio que almacenará el archivo de configuración que se usará para monitorear las máquinas.

cfg_dir=/usr/local/nagios/etc/servers

Una vez se ha descomentado y guardado, habrá que crear dicho directorio, ya que no está creado. Para ello se ejecutará:

```
root@openstack:~$ sudo mkdir /usr/local/nagios/etc/servers
```

Se creará un directorio en el cuál se deberán alojar los archivos de configuración de las máquinas remotas que se deseen monitorear. Paso que se realizará posteriormente. Primero se editará la configuración para que se detecte el router software que usa Openstack. Por el momento al ser un router software, no se puede dar mas información que si responde a ping, es decir, si esta UP o DOWN dicho router. Para configurarlo, se deberá editar el archivo de configuración que viene ya creado en la ruta: /usr/local/nagios/etc y se deberá definir el dispositivo, con su dirección IP y también los servicios que se desean monitorear. Un ejemplo de la definición del router sería:

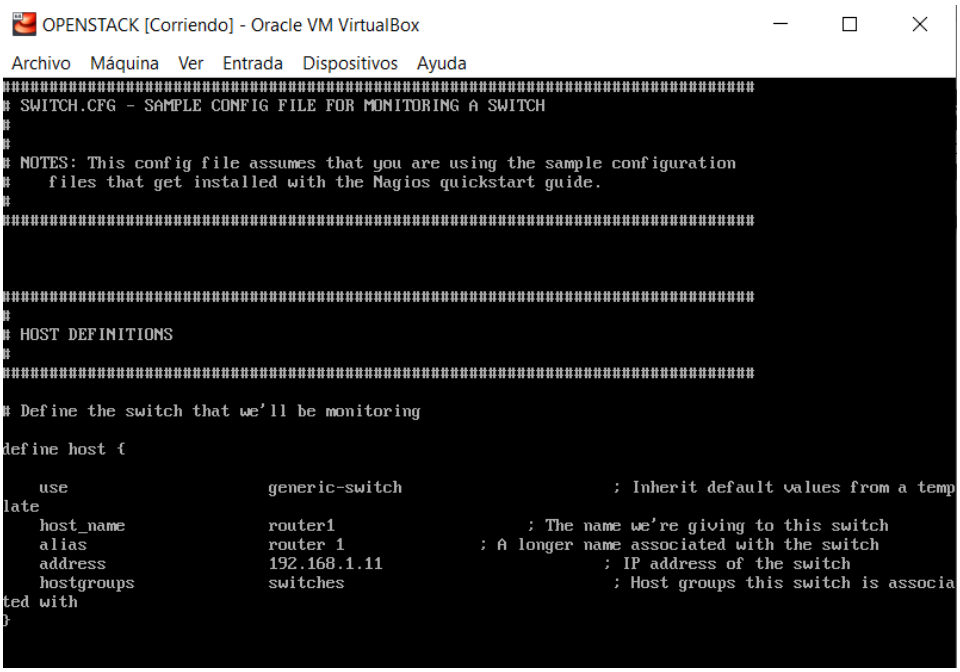

Figura 3.37: Definicion router

Y un ejemplo de un servicio a monitorear, por ejemplo el estado del PING:

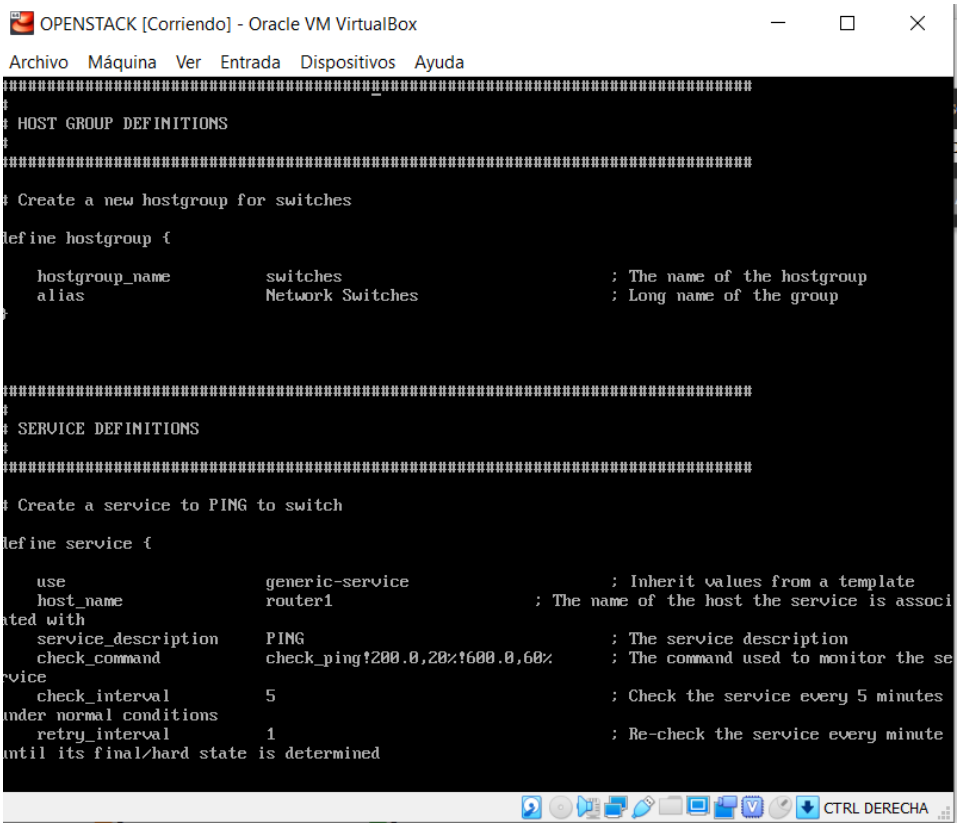

Figura 3.38: Definicion servicio PING

Una vez configurado, se puede acceder a Nagios via web y se observará que en el apartado Hosts aparecen dos elementos, el localhost que viene ya configurado por defecto y del que no se ha modificado nada, y el router donde se nos indicará su estado y en el apartado Services se observarán las características de cada uno que se están monitoreando.

Para monitorear equipos remotos, hay que instalar un plugin de Nagios, el NRPE (), en cada equipo que se desea monitorear. En el host habrá que modificar la configuración de Nagios para que detecte este plugin y entonces se podrán monitorear los diferentes dispositivos remotos.

Primero, se instalarán los plugins en cada uno de los equipos remotos que se quieran monitorear. Para instalar los plugins de Nagios y el NRPE Add-on, habrá que ejecutar las siguientes instrucciones:

```
debian@debianrouter:~$ rpm -ivh \
https://dl.fedoraproject.org/epel/epel-release-latest-7.noarch.rpm
```

```
debian@debianrouter:~$ apt update
```

```
debian@debianrouter:~$ apt install -y nagios-nrpe-server \
nagios-plugins
```

Una vez se ha terminado de instalar, habrá que configurar NRPE para que permita comunicación con el servidor de Nagios, es decir con el host. Para ello, habrá que modificar el archivo de configuración de NRPE, nrpe.cfg, y en la línea de allowed host, añadir la ip del host.

allowed_hosts=127.0.0.1, **192.168.1.101**

Este archivo de configuración, contiene los comandos mas básicos para chequear diferentes aspectos del equipo remoto y como ejecutarlos. Por ejemplo:

command[check_total_procs]=/usr/lib/nagios/plugins/check_procs -w 150 -c 200

Este comando, medirá la cantidad total de procesos que se están ejecutando y si son mas de 150 devolverá un estado de aviso (w) y si son mas de 200 uno de crítico(c). La salida de este comando podría ser las siguiente:

PROCS WARNING: 190 processes | procs=190;150;200;0;

Cuando se ha terminado de configurar el equipo remoto, hay que reiniciar el servicio NRPE para que los cambios tengan efecto.

```
debian@debianrouter:~$ systemctl restart nagios-nrpe-server
```

Queda por último configurar el host, instalando en el también NRPE, para poder detectar desde el mismo, el estado de los equipos remotos. Como en el caso concreto de este proyecto, se ha usado para el host, una imagen Centos, hay que realizar un paso previo a la instalación de los plugins y NRPE, que es configurar los repositorios EPEL ya que por defecto, estos no contienen NRPE y los plugins de Nagios. Para ello habrá que instalar los paquetes con los repositorios de la siguiente manera:

```
root@openstack:~$ yum install -y epel-release
```

Una vez se tiene instalado, se puede proceder a instalar sin problemas NRPE, de la siguiente forma:

```
root@openstack:~$ yum -y install nagios-plugins-nrpe
```

Hay que tener en cuenta que los comandos para instalarlo en el host y en los equipos remotos, pueden variar si se usan sistemas operativos diferentes. Cuando ya se tienen instalados, hay que crear una definición del comando de nrpe en el archivo de configuración de los comandos de Nagios, commands.cfg que se encuentra en el subdirectorio objects dentro de la carpeta de Nagios. En dicho archivo de configuración habría que añadir al final unas líneas como las siguientes:

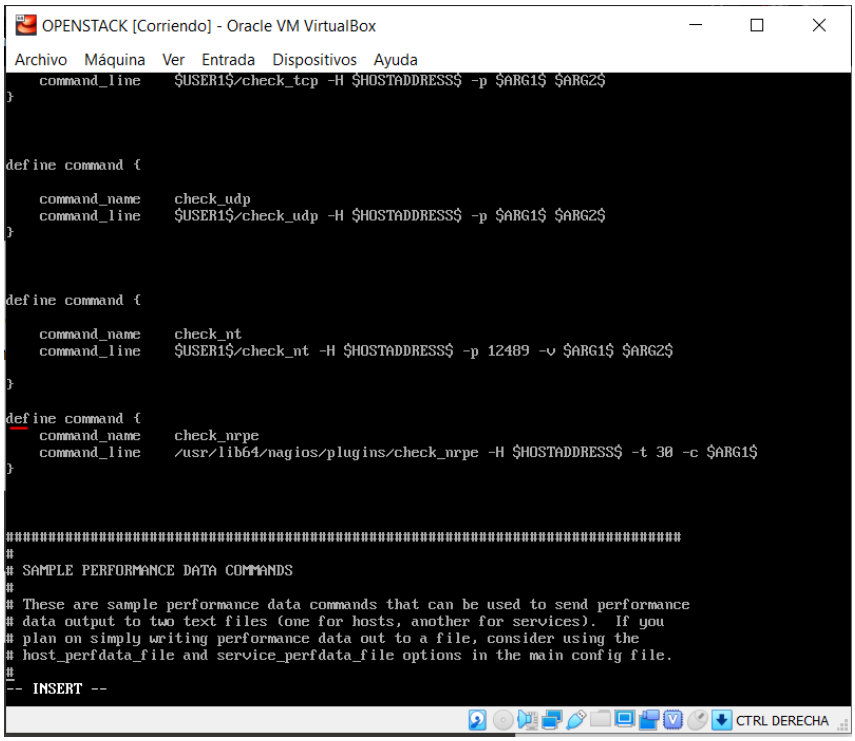

Figura 3.39: Definicion NRPE

En este punto, es cuando hay que realizar el paso mencionado anteriormente, que es el de crear el archivo de configuración de cada uno de los equipos remotos que se van a monitorear, y alojarlos en la carpeta servers creada con anterioridad. En este caso, dichos archivos de configuración se crearán con el edit vi pero se puede usar cualquier otro. Dichos archivos deberán tener primero la definición del host a monitorear, posteriormente una definición del grupo al que pertenece dicho host y por último todos los servicios que se desean monitorear y están soportados por NRPE, como se ha visto en la instalación de este plugin en el equipo remoto. En la definición del host, habría que indicar la plantilla que se usará como base, así como el nombre de dicho equipo, que a ser posible deberá ser igual al nombre del archivo de configuración, y también habrá que incluir su dirección IP. En este caso se indicará la IP flotante que Nagios asigna a la máquina virtual. Quedaría algo de la siguiente manera:

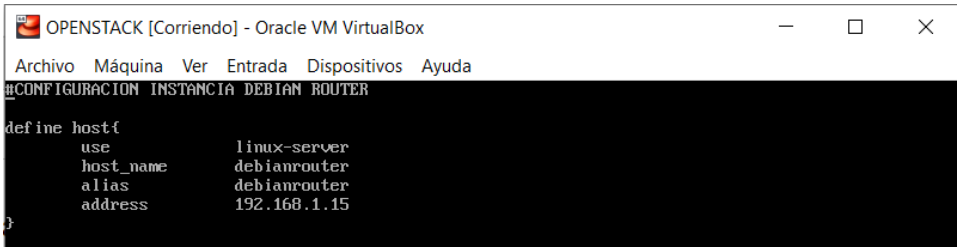

Figura 3.40: Definicion host

Para definir el grupo al que pertenece este host, se añadirían unas líneas como las siguientes:

Figura 3.41: Definicion hostgroup

Y ya faltaría por último añadir los servicios que se quieren monitorear. La sintaxis de dichos servicios sería similar en todos, cambiando la descripción del servicio que sería un comentario propio y lo importante el comando de NRPE que usaría para ello. Un ejemplo para monitorear el uso de la memoria SWAP, el total de procesos o la carga actual seria:

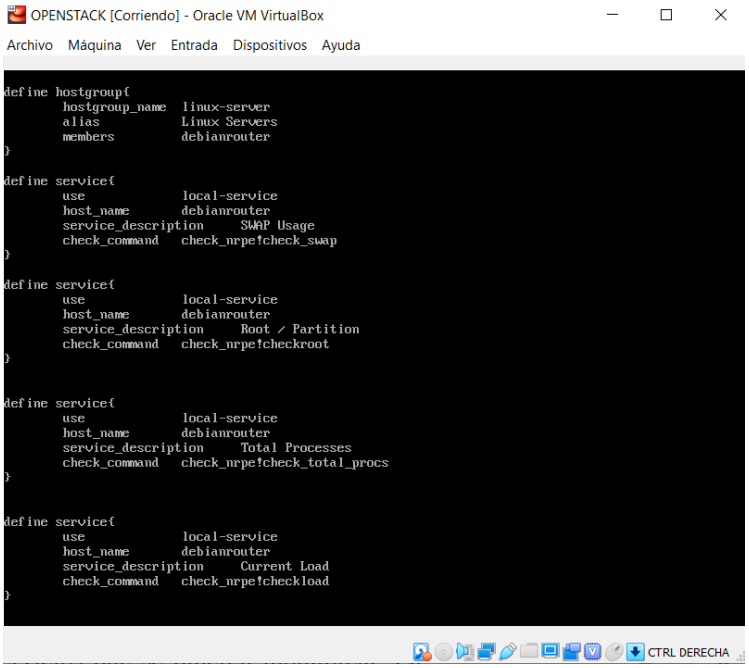

Figura 3.42: Definicion servicios

Esta sintaxis es la que debería de seguir cualquier servicio, cambiando el service_description y la parte en negrita del check_command que es la que identifica que tipo de comando usará para monitorear dicho servicio.

Capítulo 4

Conclusiones y líneas futuras

4.1. Conclusiones

Las aplicaciones y soluciones cloud tienen una marcada tendencia hacia el rápido crecimiento y en un futuro cercano a asentarse como el modelo empresarial global. Es una tecnología en continuo desarrollo y tal y como se detalla en este proyecto, es una tecnología que aporta grandes ventajas al desarrollo empresarial lo que generará su asentamiento en un futuro cercano como la principal solución para las redes de las empresas. Aunque principalmente se orientó al entorno empresarial, ha sido en el uso a nivel usuario cuando ha conseguido una mayor repercusión a la hora de facilitar el acceso de datos en cualquier momento y lugar. Esto es debido a que aún existen grandes reticencias acerca de implementar esta tecnología al completo en una empresa por el desconocimiento de ciertos aspectos legales acerca de la misma. En este trabajo se aborda la implementación de un entorno de laboratorio, escalable a un entorno empresarial con unos recursos más potentes, a partir de uno de los proveedores de servicios Cloud más intuitivos, y se observa la facilidad que existe a la hora de desarrollar una topología de red e implementarla con las necesidades deseadas junto con la integración de un software de gestión de red aplicado a la red cloud, para poder realizar un monitoreo de la misma y llevar un control a tiempo real y más exhaustivo de dicha implementación. En resumen, la tecnología cloud aporta una nueva forma y una evolución de la implementación de las redes, facilitando el proceso ya que elimina muchos elementos físicos y mejorando la adaptabilidad y actualización de las mismas gracias a ser completamente elementos software.

4.2. Líneas futuras

La gran desventaja de las redes cloud, es que hoy por hoy carecen de herramientas completas de gestión de red, nativas de la nube, es decir, hay que integrar una herramienta no pensada para la gestión de la nube en ella como se ha realizado en este trabajo al integrar el software de gestión de red de Nagios a una red Openstack. Zabbix es un desarrollo completo, no un fork de Nagios, y tiene una mejor visión de la monitorización, cubriendo aparte rendimiento y no solo estados de los elementos, que es una de las principales desventajas de Nagios. Como punto a favor es que tiene una interfaz web para la gestión, evitando así el complejo de ficheros de texto entrelazados y procesos manuales de los que se surte Nagios. Añadiendo que la configuración en Nagios se basa en ficheros de texto y con cada cambio que se realice hay que reiniciar. Por otro lado, el uso de Zabbix también necesita de mas recursos que Nagios al tener una base de datos centralizada. Es por eso que se plantea a partir de este proyecto, integrarlo en un servidor para obtener así mas recursos que en una computadora personal y poder integrar Zabbix para la monitorización y gestión de red en lugar de Nagios, ya que es un software muy antiguo y que a nivel empresarial en IT está prácticamente en desuso y sería una buena manera de conocer un sistema, que es más usado por las empresas como CIC o GFI, que en sus sistemas cloud ya sea con una tecnología de Openstack o AWS por ejemplo, tienen Zabbix integrado. Se conseguiría una visión diferente y mas actualizada del uso de esta tecnología a nivel empresarial.

Bibliografía

[1] Eduardo López Martínez. Estudio y evolución de la gestión de red hacia los entornos cloud. Trabajo fin de grado, Universidad de Cantabria, 2017.

[2] J. Panneerselvam, L. Liu, R. Hill, Y Zhan, and W Liu. An investigation of the Effect of Cloud Computing on Network Management. *IEEE Conference*, 2012.

[3] G. Suciu, S. Halunga, A. Ochian, and V Suciu. Network Management and Monitoring for Cloud Systems. *ECAI Conference*, 2014.

[4] VirtualBox Documentation, https://www.virtualbox.org/manual/UserManual.html.

[5] Packstack Installation, https://rdoproject.org/install/packstack,.

[6] Manuel Serrano. *EBOOK Install Openstack on Centos*. Virtualiza desde Zero, 2017.

[7] Networking wtih external network, http://rdoproject.org/networking/neutron-withexisting-external-network/.

[8] Keypair Generator Puttygen, https://puttygen.com.

[9] Playing around with Openstack: Using an instance as a router, https://albertomolina.wordpress.com/2015/11/22/playing-around-with-openstack-using-an-instance-as-a-router/.

[10] Nagios core support, https://support.nagios.com/kb/article/nagios-core-installing-nagios-core-from-source-96.html.

[11] Nagios Documentation object definitions, https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/3/en/objectdefinitions.html.

Lista de Acrónimos

IaaS Infraestructure as a Service (Infraestructura como servicio).

PaaS Platform as a Service (Plataforma como servicio).

SaaS Software as a Service (Software como servicio).

SDN Software Defined Network (Redes definidas pr software).

VPN Virtual Private Network (Red privada virtual).

VLAN Virtual Local Area Network (Red de Área Local Virtual).

CLI Command Line Interface (Interfaz de linea de comandos).

RAM Random Access Memory (Memoria de acceso virtual).

SSH Secure Shell.

NRPE Nagios Remote Plugin Executor (Plugin de ejecucion remota de Nagios).