

UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA

UNAN – León

Facultad De Ciencias y Tecnología

Ingeniería en Telemática



Monografía para optar al título de

Ingeniero en Telemática

Implementación de una nube privada multi-nodo basada en OpenStack utilizando la versión Stein sobre CentOS 7, empleando PackStack RDO para su instalación.

Presentado por:

Br. Claudia Gabriela Cáceres Pérez.

Br. Rodolfo Eleuterio Cortedano Salgado.

Br. Jerson Alexander Pastrán Flores.

Tutor:

M.Sc. Denis Espinoza Hernández.

León, junio 2020

¡A la libertad por la Universidad!

Agradecimientos

Primero que nada, agradecer a Dios por darme la oportunidad de seguir adelante en cualquier situación y permitirme llegar hasta esta etapa final de mi formación profesional.

A mis formadores, personas con gran sabiduría quienes se han esforzado a ayudarme a llegar al punto en el cual me encuentro.

Quiero agradecer a mi familia en general por el gran esfuerzo que hacen a diario por darme todo lo necesario y por sus consejos que hacen que sea una mejor persona cada día y así mismo poder llegar hasta donde estoy en estos momentos.

A todos aquellos que con sus pequeñas palabras de apoyo y ánimo pudieron hacer que este trabajo salga a la luz, en especial al M.Sc. Denis Espinoza tutor de esta tesis que con su gran sabiduría y paciencia supo cómo orientarme de forma teórica, práctica y metodológicamente de manera correcta.

A mis padres Dolores Pérez y Alejandro Cáceres por su apoyo incondicional en la parte moral y económica, también por ser los principales promotores de mis sueños, gracias a ellos por cada confiar y creer en mí y en mis expectativas, agradezco a ambos por desear y anhelar siempre lo mejor para mi vida, gracias por sus palabras que me guiaron durante mi vida.

Gracias a la vida por este nuevo triunfo y a las demás personas que me apoyaron y creyeron en esta realización de tesis.

Br. Claudia Gabriela Cáceres Pérez.

Primeramente, tengo que agradecer a Dios por todas las bendiciones que me ha regalado en la vida, por darme sabiduría y las fuerzas para seguir adelante, porque sin su ayuda nada es posible.

A mis padres por ayudarme, apoyarme económicamente y estar siempre a mí lado, confiando en mí.

A mi tutor M.Sc. Dennis Leopoldo Espinoza, por todo su apoyo, sus orientaciones, su tiempo y enseñanzas que me ha brindado, así también a todos los docentes que me ayudaron a lo largo de carrera.

A mis compañeros y amigos que tuve a lo largo de la carrera que de alguna manera ayudaron en la culminación de este trabajo brindándome apoyo y palabras de aliento.

Br. Rodolfo Eleuterio Cortedano Salgado.

Primeramente, a Dios por ser mi guía y acompañarme en el transcurso de mi vida, brindándome paciencia y sabiduría para culminar con éxito mis metas propuestas.

A mis padres, que han sido el pilar fundamental de mi vida, por haberme apoyado incondicionalmente, pese a las adversidades e inconvenientes que se presentaron y ser mi mayor inspiración que, a través de su amor, paciencia, buenos valores, ayudan a trazar mi camino.

A mis maestros, que compartieron conmigo su tiempo y sus conocimientos.

Al Prof. Marvin Somarriba, por todo el apoyo para a culminación de esta tesis y por sus enseñanzas a lo largo de mi carrera.

A nuestro tutor, M.Sc. Dennis Leopoldo Espinoza, principal colaborador durante todo este proceso, quien con su dirección, conocimiento, enseñanza y bondad permitió el desarrollo de este trabajo.

Br. Jerson Alexander Pastrán Flores.

Dedicatorias

Primeramente, a Dios, por haberme dado esta gran oportunidad de ser alguien en la vida, por la fuerza y fortaleza que me ha brindado para culminar nuestra carrera universitaria.

A mis padres por todo el amor, cariño y apoyo que me brindaron a lo largo de este camino y que siempre estuvieron ahí para alentarme a la realización de todas las metas que me he propuesto.

A nuestro tutor por compartir de forma desinteresada sus conocimientos y sugerencias en la elaboración del presente trabajo.

Br. Claudia Gabriela Cáceres Pérez.

A Dios, por haberme dado la vida y permitirme llegar a finalizar mis estudios.

A mi familia, por la orientación y el esfuerzo incondicional que me brindaron para la conclusión de mis estudios.

A mis maestros, que supieron guiarme con sus conocimientos a luchar por culminar mi carrera.

A nuestro tutor, M.Sc Dennis Leopoldo Espinoza, que confió en mí y en mis compañeras y emprendió el camino final con nosotros.

Br. Rodolfo Eleuterio Cortedano Salgado.

A Dios, por ser el inspirador y darme fuerzas para continuar en este proceso de obtener uno de los anhelos mas deseados.

A mis padres, por su amor, trabajo y sacrificio en todos estos años, gracias a ustedes he logrado llegar hasta aquí y convertirnos en lo que somos.

A todas las personas que nos han apoyado y han hecho que el trabajo se realice con éxito en especial a aquellos que nos abrieron las puertas y compartieron sus conocimientos.

Br. Jerson Alexander Pastrán Flores.

Resumen

Con el propósito de proporcionar al lector los conocimientos necesarios para el despliegue de una nube privada se ha llevado a cabo un minucioso estudio de las diferentes soluciones de infraestructura que proporciona el mercado actual, buscando aquella o aquellas que mejor se adecuen a nuestras exigencias. Esto implicó tener que distinguir las diferentes formas en las que se puede presentar esta solución de infraestructura, teniendo que saber seleccionar aquella que se considere más adecuada en base a los objetivos que se hayan marcado. Una vez que se logró alcanzar una posible solución, tuvimos que verificar por medio de un conjunto de pruebas que efectivamente es una solución válida. Además, esto nos permitió conocer más acerca de la plataforma junto con la administración que requiere, entre otros aspectos de igual importancia.

Tradicionalmente las empresas, han ofrecido sus servicios en un conjunto de infraestructuras y plataformas. Estos se componen de hardware físico en términos de computación, almacenamiento y red junto con software en términos de gestión de las plataformas. Con este compendio entre hardware y software, la empresa uniría las piezas y obtendría una solución adaptada a las necesidades requeridas.

Con el surgimiento de la virtualización y por consiguiente del Cloud, el mundo tecnológico ha cambiado de manera sustancial, principalmente en la forma en que se crean y se entregan los diferentes servicios. El Cloud, que tiene sus bases en la virtualización, es un paradigma que permite ofrecer servicios de computación a través de una red (usualmente Internet). Teniendo en cuenta lo antes comentado, es claro que el Cloud tiene y va a seguir teniendo gran valor en los próximos años. Viendo su potencial, se realizó un despliegue con OpenStack, un software libre y de código abierto distribuido bajo los términos de la licencia Apache.

Índice

1	Introducción	1
2	Antecedentes	1
3	Planteamiento del problema	4
4	Justificación	5
4.1	Originalidad	5
4.2	Alcance.....	5
4.3	Producto	6
4.4	Impacto.....	6
5	Objetivos	7
5.1	Objetivo general	7
5.2	Objetivos específicos.....	7
6	Marco Teórico	8
6.1	Cloud Computing.....	8
6.2	¿Para qué sirve el Cloud Computing?.....	9
6.3	¿Cómo funciona el Cloud Computing?.....	9
6.3.1	Ventajas	10
6.3.2	Desventajas	11
6.4	Los tres modelos fundamentales en el campo de servicios en la nube.....	11
6.4.1	SAAS: El software como un servicio.....	12
6.4.2	IAAS: Infraestructura como un servicio.....	12
6.4.3	PAAS: Plataforma como un servicio	13
6.5	Tipos de nubes	13
6.5.1	Nube pública	14
6.5.2	Nube privada.....	14
6.5.3	Nube Híbrida.....	15
6.6	Tipos de Virtualización	16
6.6.1	Tipos fundamentales de Hipervisores.....	17
6.7	¿Por qué es interesante utilizar virtualización y qué lo hace posible?.....	17
6.7.1	Métodos de virtualización	18
6.8	OpenStack.....	20
6.8.1	Principales características de OpenStack.....	21

6.8.2	Módulos o componentes de OpenStack	21
6.8.3	OpenStack Stein.....	23
6.8.4	RDO.....	23
6.8.5	Métodos de instalación de OpenStack.....	24
7	Diseño Metodológico.....	25
7.1	Tipo de investigación.....	25
7.2	Etapas de la investigación.....	25
7.2.1	Etapa I. Etapa de investigación.....	25
7.2.2	Etapa II. Selección de las herramientas y versión	25
7.2.3	Etapa III. Configuración de la nube privada	25
7.2.4	Etapa IV. Prueba de las herramientas	26
7.2.5	Etapa V. Documentación	26
7.3	Materiales hardware y software.....	26
7.3.1	Software.....	27
7.3.2	Hardware	27
8	Resultados	28
8.1	Configuración de la nube privada.....	28
8.2	Prueba de las herramientas	37
9	Conclusiones	41
10	Recomendaciones.....	42
11	Bibliografía.....	43
12	Anexos.....	46
12.1	Anexo 1: Interfaces de OpenStack.....	46
12.2	Anexo 2: Archivo de respuesta.....	55

1 Introducción

En los últimos años se ha desarrollado lo que es el Cloud Computing (Computación en la nube), que sin lugar a duda es una nueva área dentro del campo de la tecnología y la información permitiendo almacenamiento, procesamiento y el uso de elementos de mayor nivel tales como sistemas operativos y aplicaciones sin tener la parte física presente. Un ejemplo son los servicios ofrecidos por Google, Dropbox, One Drive, Mega, de igual forma servicios de aulas virtuales tales como Moodle y Classroom. Estos sistemas han avanzado tanto que son autónomos para el usuario pudiendo crear también despliegues de red, infraestructuras completas, creación de máquinas, routers, etc.

Se puede tener acceso a la información o estos servicios, mediante una conexión a internet desde cualquier dispositivo móvil o fijo ubicado en cualquier lugar, sin la necesidad de tener que instalar programas en el ordenador, facilitando su uso y evitando cargar el ordenador del usuario, de igual forma reduciendo los costos de equipos. En si podemos tener servidores completos tanto como para usuarios o para empresas, sin tenerlos físicamente.

En la actualidad existen muchas empresas que brindan el servicio de cloud entre ellas están Google Cloud, Amazon Web Services (AWS), Microsoft Azure y DigitalOcean donde brindan una plataforma completa de paga, la cual está disponible para cualquier persona que desee hacer uso de los servicios que estas ofrecen, donde uno puede crear desde una máquina virtual hasta una infraestructura completa.

Con ello nace la necesidad de desarrollar una plataforma de software libre y código abierto que incluya almacenamiento y servicios completos de computación, en colaboración de la nasa y otras empresas dedicadas a estos servicios nace OpenStack, orientado para infraestructuras, software y plataformas en diferentes tipos de nubes, que se implementaran según la distribución que se elige, puede ser nubes públicas, privadas e híbridas.

2 Antecedentes

Para establecer los antecedentes de esta investigación se buscó en la Biblioteca Google Académico sobre el tema de Openstack.

Durante el proceso de búsqueda se encontraron varios estudios sobre el tema.

✓ **Creación de sistema cloud con OpenStack.**

Lo implementaron en Valencia, España, su autor es: Alejandro Carlos Osuna Fontan.

El objetivo general del tema fue: La implementación completa un sistema de Cloud Computing con la tecnología OpenStack.

En este trabajo se realizó la implementación completa de un sistema de Cloud Computing con la tecnología OpenStack. Iniciando con un pequeño estudio de las diferentes opciones libres de sistemas cloud. Se prosiguió un estudio cronológico del origen de OpenStack, qué empresas lo utilizan y con qué fin. Por otro lado, se analizó la arquitectura del sistema y los diferentes módulos que dispone. Además, se llevó a cabo una primera implantación básica del sistema con el objetivo de ofrecer una guía de implantación y una segunda implantación con módulos avanzados, donde se estudiaron las diferentes funciones que nos ofrece un sistema cloud. Finalmente se observaron las sobrecargas que aparecen a causa de la virtualización de los sistemas, sobrecarga en Escritura/Lectura de disco, de transferencia de red y ejecución de aplicaciones

✓ **Establecimiento de la integración entre OpenStack y las redes definidas por software en el periodo de diciembre del año 2016.**

Se realizó en Pereira, Colombia. Su autor es Miguel Antonio Gaitán Gallego.

El objetivo general fue: Establecer la integración entre OpenStack reléase Juno y las Redes Definidas por Software para permitir la gestión de las redes de una forma centralizada, flexible, automática y escalable.

Este trabajo mostró cómo se integra la plataforma de OpenStack y las SDN, ya que la primera carece de flexibilidad, despliegue y gestión de las redes; mediante este trabajo se expuso el controlador ODL como una solución a dicho problema. En la primera

sección del documento se toman los referentes teóricos de dichas herramientas y los antecedentes del trabajo, una segunda parte de carácter metodológico donde se muestra cómo se implementa y se configura un ambiente básico de la plataforma OpenStack; para este proceso se realizó la instalación y configuración de tres máquinas virtuales corriendo bajo VirtualBox.

✓ **Implementación de Prototipo de Tecnología de Cloud Computing para Servicios de Infraestructura (IaaS) en la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato.**

Se realizó en: Ambato, Ecuador. Sus autores son: Klever Renato Urvina, Barrionuevo Braulio Vinicio Ruiz Quispe.

El objetivo general fue: Implementar un prototipo de tecnología Cloud Computing para servicios de infraestructura (IaaS) en la facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato

El propósito del trabajo fue Implementar un Prototipo de Tecnología de Cloud Computing para Servicios de Infraestructura (IaaS) en la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, con la finalidad de ahorrar costos en la adquisición de equipos, así como también utilizar las herramientas IaaS para mejorar las prácticas de los estudiantes y docentes dentro de los laboratorios de la FISEI. Para la realización del trabajo se consideraron varios Sistemas Operativos con los cuales se puede implementar la tecnología IaaS como son: Mirantis, Windows Azure, OpenStack Kilo, OpenStack Liberty. Tomando en cuenta que la creación de la nube se realizó en equipos de cómputo convencionales y de características sofisticadas, sin la necesidad de adquisición de un servidor que puede llevar costos excesivos tanto en la compra como en su soporte, se procedió a la creación de IaaS bajo la licencia Apache, la misma que es de libre distribución, así también se consideró el Sistema Operativo CentOS 7 en su versión mínima para mejor facilidad y poca utilización de recursos tanto de hardware como de software.

3 Planteamiento del problema

Debido a que la computación en la nube es el auge en la actualidad y que las empresas necesitan de este conocimiento, se decidió abordar este tema en nuestro trabajo monográfico.

Un inconveniente que se encontró para poder realizar la implementación de una nube privada con OpenStack fue la poca información detallada que existe sobre el tema. Otro problema es la diversidad de formas que encontramos para realizar la instalación, lo cual fácilmente puede confundir a alguien nuevo en este tipo de servicios.

A lo anterior se puede sumar la gran cantidad de recursos físicos que requiere este tipo de instalación, los cuales no siempre están al alcance de las personas interesadas en este tema. Otro factor que también dificulta su instalación son los diversos conocimientos de redes TCP/IP y servicios que hacen falta para lograr la intercomunicación entre los diferentes nodos.

Pregunta general:

- ✓ ¿Cómo implementar una nube privada multinodo basada en OpenStack utilizando la versión Stein sobre CentOS 7 empleando PackStack RDO para su instalación?

Preguntas específicas:

- ¿Qué infraestructura de red se debe desplegar para realizar una instalación de OpenStack multinodo agrupando los componentes en diversas maquinas?
- ¿Cuáles son las configuraciones necesarias para permitir comunicación entre la red física y la nube privada?
- ¿Qué elementos es necesario crear para poder ofrecer la infraestructura como servicio en nuestra nube privada?

4 Justificación

Tomando como referencia los problemas antes expuestos, nace la idea de crear un documento que explique cómo implementar una nube privada multinodo basada en OpenStack utilizando la versión Stein sobre CentOS 7 empleando PackStack RDO.

4.1 Originalidad

No existen trabajos similares en nuestra universidad que traten de este tema, por lo cual esperamos que sirva de base para futuras investigaciones.

4.2 Alcance

A lo largo de nuestro trabajo se abordan los diferentes aspectos que fueron necesarios desarrollar para la implementación de una nube privada con OpenStack. Estos aspectos fueron:

- Selección de la infraestructura de red sobre la cual sería desplegada la nube, seleccionando los componentes que se instalarían e indicando en que nodo quedaría cada uno de ellos.
- Instalación de OpenStack multinodo, tomando en cuenta las características físicas de los equipos para una división apropiada de los roles y componentes en base a los recursos, así como las configuraciones de red necesarias para la comunicación de la nube privada con la red física.
- Creación y configuración de los elementos necesarios para brindar el servicio de nube privada a los usuarios finales, permitiéndoles crear catálogo de imágenes, routers, redes, instancias entre otros servicios.
- Validación del correcto funcionamiento de la nube privada permitiendo a usuarios finales desplegar sus propios servicios sobre ésta.

4.3 Producto

Se presentará una infraestructura de nube privada instalada de manera multi nodo donde se configura y se valida su correcto funcionamiento. A su vez se entregará un documento donde se explique la forma de instalación, configuración y creación de los elementos necesarios para ofrecer el servicio de nube privada a los usuarios finales.

4.4 Impacto

Este trabajo permitirá comprender la instalación de una nube privada multinodo con OpenStack y su correcta configuración, así como el comportamiento de los diferentes componentes. Con este material, el lector tendrá las bases suficientes para ahondar en este tipo de servicio y desplegar su propia nube haciendo uso de las tecnologías y herramientas aquí expuestas.

5 Objetivos

5.1 Objetivo general

Implementar una nube privada multinodo basada en OpenStack utilizando la versión Stein sobre CentOS 7 empleando PackStack RDO para su instalación.

5.2 Objetivos específicos

1. Seleccionar la infraestructura de red sobre la cual se desplegará OpenStack, indicando los componentes que serán instalados en cada nodo.
2. Instalar OpenStack multinodo de acuerdo a la capacidad de los equipos disponibles.
3. Configurar la interfaz de red en el nodo de red en modo bridge para que la nube privada tenga conectividad con la red física.
4. Crear los elementos necesarios en la nube privada para brindar el servicio a los usuarios finales.
5. Validar el correcto funcionamiento de la nube privada, lanzando un determinado número de instancias con algún servicio.

6 Marco Teórico

6.1 Cloud Computing

En los últimos años y con la aparición de nuevas tecnologías relacionadas con el mundo digital han surgido una serie de conceptos, generalmente en inglés, para definir los nuevos servicios que se ofrece tanto para usuarios como para empresas. Conceptos como Big Data o Cloud Computing están cada vez más extendidos en cualquier ámbito, ya sea doméstico o empresarial y muchos todavía no saben lo que estos términos significan.

Y es que, aunque a veces se piense que no somos usuarios de servicios con estas palabras tan extrañas lo cierto es que los usamos con mucha más frecuencia de lo que se cree y además de forma diaria. Si alguien se pregunta qué es Cloud Computing lo más frecuente es que no sepa qué responder, pero resulta de vital importancia asimilar y comprender estos conceptos para no quedarse atrás en esta nueva revolución digital.

En concreto el Cloud Computing consiste en ofrecer servicios de computación a través de la red, por medio del almacenamiento en la nube. El usuario de este tipo de servicios no tiene que instalar nada en su ordenador y sin embargo puede tener acceso a múltiples servicios con utilidades que van desde lo más cotidiano a lo más técnico.

Para entender este concepto de una forma más sencilla lo mejor es a través de un ejemplo que se entiende fácilmente: el correo electrónico. Todo el mundo usa, en mayor o en menor medida el correo electrónico, y sin saberlo está usando un servicio de los denominados Cloud Computing. Usamos el correo electrónico sin tener que instalar nada en el ordenador gracias al servicio de almacenamiento en la nube.

Del mismo modo ocurre con aplicaciones que sé, no hace falta descargar porque tienen su almacenamiento en la nube o muchos sistemas de gestión empresarial. Lo cierto es que esta tecnología está cada vez tomando un mayor protagonismo en nuestras vidas y que ha llegado para quedarse entre nosotros.

6.2 ¿Para qué sirve el Cloud Computing?

La principal utilidad de este tipo de tecnología radica en evitar que el ordenador del usuario asuma una fuerte carga de trabajo a la hora de ejecutar aplicaciones que el usuario necesite utilizar. Gracias al almacenamiento en la nube este trabajo lo realiza la gran red de computadoras que componen este sistema, agilizando el trabajo y haciéndolo fácilmente accesible y sin afectar al ordenador del usuario del servicio en cuestión.

Lo único que necesitaría el usuario en lugar de una aplicación instalada y ejecutándose en el ordenador es usar el navegador de internet como si de una web cualquiera se tratase. La nube se encarga del resto del trabajo, simplificando las tareas que el usuario tiene que llevar a cabo.

Las ventajas del almacenamiento en la nube están fuera de toda duda. A la ventaja de multiplicar los recursos a los que el usuario puede tener acceso de forma rápida y sencilla se une la ventaja de evitar el coste de las licencias de los programas y softwares para cada ordenador que necesite usar el programa en cuestión, algo que costaba grandes sumas de dinero a las empresas antes de aparecer esta tecnología.

6.3 ¿Cómo funciona el Cloud Computing?

En un sistema Cloud Computing se puede encontrar dos partes claramente diferenciadas del sistema. La primera es el frontend o interfaz del usuario y la segunda es el backend, que es el servicio en la nube en sí mismo.

La interfaz del usuario incluye la red informática que este use y la aplicación necesaria para acceder al servicio en la nube. Una interfaz sencilla para el usuario que está conectada al backend a través de internet. En la parte del backend se puede encontrar todos los sistemas que usa la nube para almacenar la información, los servidores que usa y los ordenadores del sistema.

Todo este engranaje se conecta a un servidor central que administra el funcionamiento de todo el sistema, supervisa el tráfico del mismo y se asegura de que las demandas de los clientes se realicen sin ningún tipo de inconveniente. Las reglas del sistema siguen

unos protocolos y usan un software middleware que facilita la conexión entre las computadoras en red para que se comuniquen entre sí. Un sistema muy complejo a simple vista pero que repercute en grandes beneficios para usuarios y empresas como son las comunicaciones unificadas.

La tecnología de Cloud Computing está evolucionando la forma en la que empresas y personas individuales realizan una gran multitud de tareas y servicios distintos. Encontrar toda la información de forma segura y rápida a través de una conexión a internet sin tener que descargar ningún programa adicional ni pagar diferentes licencias para cada ordenador es una evolución que empresas de todo el mundo está sabiendo aprovechar. (Gluppi, 2018)

6.3.1 Ventajas

- ✓ **Acceso desde cualquier ubicación geográfica a los datos y aplicaciones:** Solo es necesario contar con una buena conexión a internet para disponer de todos nuestros datos y aplicaciones almacenadas en la nube, independientemente del momento, del lugar en que nos encontremos y del tipo de dispositivo con el que se accede.
- ✓ **Libre mantenimiento por parte del usuario:** La empresa que presta el servicio de almacenamiento se hace cargo de todas las actividades de mantenimiento preventivo y correctivo tanto del hardware como del software.
- ✓ **Actualizaciones a últimas versiones:** La empresa proveedora del servicio se encarga de las actualizar de las aplicaciones informáticas, tanto las utilidades como del software para proteger la integridad de los datos.
- ✓ **Aplicaciones compartidas más económicas:** La utilización de las aplicaciones por varios usuarios, abarata su costo por usuario.
- ✓ **Reducción de la inversión en equipamiento informático del usuario:** Debido a la concentración de datos y aplicaciones en la nube, disminuyen los requerimientos técnicos de los dispositivos del usuario tanto a nivel de micro procesadores como de la capacidad de almacenamiento.
- ✓ **Sistema de almacenamiento escalable:** El espacio de almacenamiento contratado en la nube, es fácilmente modificable según el aumento o

disminución de las necesidades del cliente, sin tener que realizar mayores inversiones para la previsión de posibles soluciones futuras.

6.3.2 Desventajas

- ✓ **Falta de seguridad:** El control de acceso a los datos está en manos del proveedor de servicio y de los usuarios del servicio por lo que será conveniente prestar atención en la fortaleza de las claves generadas, realizar cambios frecuentes de ellas y no divulgarlas a otras personas.
- ✓ **Privacidad:** Nuestros archivos de datos se encuentran ubicados en un lugar remoto por lo que perdemos su control directo y este control pasa a estar a cargo de la empresa proveedora del servicio, por lo que se vuelve fundamental, contratar a una empresa de prestigio.
- ✓ **Acceso remoto:** El acceso a los datos solo es posible con una conexión fiable a Internet por lo que sin conexión no hay posibilidad de acceso a los datos y a las aplicaciones almacenadas en la nube.
- ✓ **Cobertura legal:** Hay situaciones en que no está del todo claro si se debe aplicar la ley de uno u otro país. Suele ocurrir que el cliente está radicado en un país y los servidores de la empresa que presta el servicio de almacenamiento en uno diferente, por lo que hay dudas sobre cuál es la ley que hay que aplicar.
- ✓ **Conflictos de propiedad intelectual:** La información de los clientes ya no se encuentra bajo su control, con lo que pueden llegar a surgir problemas sobre su propiedad. (AprenderCompartiendo, 2016)

6.4 Los tres modelos fundamentales en el campo de servicios en la nube.

Cuando nos referimos a desarrollar aplicaciones en la nube tenemos que puntualizar de qué manera lo vamos a hacer, dentro del concepto nube existen distintas formas de hacerlo. Entre estas distintas formas que puede adoptar la nube se encuentran: Software como servicio (SAAS), Infraestructura como servicio (IAAS) y Plataforma como un servicio (PAAS).

Los servicios en la nube van mucho más allá del alojamiento de archivos en línea para poder acceder a ellos a través de Internet. Actualmente, este concepto incluye toda una

gama de servicios que nos permiten acceder a aplicaciones, plataformas y hasta infraestructuras enteras, todas en línea.

Existen tres conceptos que deben estar en el punto de mira de cualquier gerente, incluso aunque su empresa no sea de ámbito tecnológico, ya que pueden llegar a ser herramientas fundamentales para generar valor y convertirse en el líder de su mercado.

6.4.1 SAAS: El software como un servicio

Los servicios de aplicaciones en la nube, o Software como servicio (SaaS), simbolizan el mercado más extendido en la nube y continúan creciendo vertiginosamente. SaaS es un modelo por el cual las aplicaciones de software son instaladas en línea y accesibles vía Internet desde un navegador web, sin necesidad de realizar descargas o instalaciones. En este modelo, la configuración del software, la instalación y el despliegue es realizado por el mismo proveedor del servicio prestado, lo que significa que las empresas no tienen por qué incidir en costos adicionales de hardware ni licencias de software.

En ocasiones, el proveedor impone un precio mensual o anual por usuario que se conecte a la aplicación, aunque existen otras formas de pago, como son: el consumo de CPU, el consumo de megas, el ancho de banda de datos desde o hacia la aplicación y/o el tráfico de datos desde o hacia la aplicación.

6.4.2 IAAS: Infraestructura como un servicio

La infraestructura como servicio se comporta de manera similar a como lo hace SaaS. Existe una diferencia clave y es que, en lugar de vender programas o licencias, los proveedores de este servicio ofrecen sus servidores para que otras empresas puedan usarlos a su antojo. Se trata de una infraestructura informática inmediata que se aprovisiona y administra a través de Internet. Permite reducir o escalar verticalmente los recursos con rapidez para ajustarlos a la demanda y se paga por uso.

IaaS evita el gasto y la complejidad que suponen la compra y administración de sus propios servidores físicos y otra infraestructura de centro de datos. Cada recurso se

ofrece como un componente de servicio aparte, y solo hay que alquilar un recurso concreto durante el tiempo que se necesite.

En definitiva, una empresa puede eludir la adquisición de servidores, espacio en un centro de datos o equipamiento de redes y comprar todas estas infraestructuras a un proveedor de nube. Este aprovisionamiento de recursos se hace a través de la web.

6.4.3 PAAS: Plataforma como un servicio

Este concepto de plataforma como Servicio proporciona un entorno que permite a los desarrolladores crear aplicaciones y servicios que funcionan a través de Internet.

Con Plataforma como un servicio nos referimos a la combinación del hardware y software que requieren los desarrolladores y diseñadores web a la hora de poner en marcha sus proyectos tecnológicos y aplicaciones.

Los servicios de plataforma en la nube (PaaS), se utilizan para aplicaciones y otros desarrollos, mientras proporcionan componentes de nube al software. Lo que obtenemos con PaaS es la posibilidad de construir, desarrollar o personalizar aplicaciones. PaaS hace que el desarrollo, la prueba y el despliegue de aplicaciones sean más rápido, sencillo y rentable.

Algunas de las ventajas que ofrece este servicio son: la reducción del tiempo de programación, agregar más funcionalidad de desarrollo sin incorporar más personal, desarrollar para varias plataformas con mayor facilidad, usar herramientas sofisticadas a un precio asequible o administrar el ciclo de vida de las aplicaciones con eficacia. (Technology, 2017)

6.5 Tipos de nubes

La nube se está convirtiendo en una opción obvia para las organizaciones modernas que buscan escalar sus capacidades informáticas. Aquí veremos los 3 principales tipos de nubes que actualmente se ofrecen:

6.5.1 Nube pública

Una nube pública es quizás la más simple de todas las implementaciones en la nube. El proveedor de la nube distribuye los recursos, servicios y plataformas de computación, desarrollados a partir de hardware de propiedad y gestiona la administración a múltiples clientes a través de una red. Las nubes públicas siempre tienen una arquitectura multitenant, lo que significa que existen múltiples instancias virtuales en una sola instancia de software.

Las nubes públicas pueden complementarse con otros servicios compartidos tales como servicios de balanceo y aceleración de carga, servicios de backup o de seguridad perimetral. Compartir recursos permite un importante ahorro de costes respecto al modelo de Cloud Privado. Sin embargo, también son más vulnerables que las nubes privadas.

Características

- ✓ Mayor escala, costes más bajos.
- ✓ Difícil integración con Legacy.
- ✓ Mayor nivel de autogestión: los servicios, aplicaciones y almacenamiento están disponibles a través de internet.
- ✓ El mejor nivel de eficiencia en entornos compartidos.
- ✓ Más vulnerable: incertidumbre en calidad y seguridad.
- ✓ Modular y escalable.

6.5.2 Nube privada

Hablamos de nube privada cuando los servicios e infraestructura se mantienen en una red privada. Es decir, la empresa dispone de un entorno de nube exclusiva. Estas nubes ofrecen mayor nivel de seguridad y control que la nube pública, pero requieren que la compañía siga comprando y manteniendo todo el software y la infraestructura, la cual cosa aumenta el incremento de los costes IT. El Cloud Privado podría compararse con la data center internos de algunas empresas, diseñados para responder a una demanda concreta. Mediante la virtualización podemos añadir a las características de la

data center los beneficios de la nube, tales como la agilidad en la provisión o cierto nivel de elasticidad.

Las soluciones de Cloud Privado generan una sensación de mayor seguridad para los clientes que disponen de este tipo de despliegues, al no compartir recursos con otros usuarios. La elección del proveedor dependerá mucho de los recursos tecnológicos que más se adapten a las necesidades técnicas o económicas de la empresa, así como las tecnologías que permiten integraciones con otros servicios de infraestructura ya existentes (backup, balanceadores, red, etc.).

Características

- ✓ Mejora la operación interna.
- ✓ Mejoras modestas en costes.
- ✓ Escalabilidad y flexibilidad limitadas.
- ✓ Riesgo de obsolescencia.
- ✓ Gestión limitada de picos de demanda.
- ✓ Proyectos a medida.

6.5.3 Nube Híbrida

Las nubes híbridas disfrutan de las ventajas de la nube pública y privada. Una nube híbrida es una combinación de 1 o más nubes públicas y privadas organizadas por software de administración y automatización que permite que las cargas de trabajo, los recursos, las plataformas y las aplicaciones migren entre los entornos. El inconveniente es que debe realizar un seguimiento de múltiples plataformas de seguridad diferentes y asegurarse de que todos los aspectos de su negocio puedan comunicarse entre sí.

Su estructura compleja requiere la coordinación de una infraestructura propia con otra gestionada por otro entorno, así como una buena conectividad entre las dos plataformas, gracias a su versatilidad y a la experiencia que pueden aportar algunos integradores.

Características

- ✓ **Integra la nube privada (interna) con servicios de nube pública (externa):** Integración con los sistemas ya existentes, evitando el conflicto con el sistema heredado manteniendo las configuraciones de computación, red y almacenamiento.
- ✓ **Escalabilidad:** Permite responder a las necesidades de la empresa de forma inmediata y crecer en base a los recursos tecnológicos. De esta manera la organización no tiene que preocuparse por los requerimientos tecnológicos para ampliar sus posibilidades.
- ✓ **Personalización:** Adaptándose a la estrategia en función de las necesidades particulares de la empresa en cada momento.
- ✓ **Seguridad:** Respondiendo a diferentes requisitos de seguridad, en particular, la gestión de datos críticos.
- ✓ **Orquestación y automatización:** En armonía unos con otros, de forma automatizada. La sincronización entre nube pública y privada debe permitir el mantenimiento de cada elemento de negocio en su entorno más eficiente. (Ilimit, 2018)

6.6 Tipos de Virtualización

Para todos aquellos que nos centramos en el mundo Linux, existe algo que es indispensable, y es el particionamiento del disco duro. De esta forma, podemos tener, en una partición, nuestra distribución 'fija' y en la otra ir probando las diferentes novedades que puedan ir surgiendo. Sin embargo, no todos están a favor de esta opción, puesto que es verdad que cuando más 'sufre' el disco duro es durante una instalación, y el ir instalando distribución tras distribución con el paso de un par de años le pesará a dicha pieza de nuestro equipo.

Los diferentes tipos de virtualización conforman un tipo de tecnología que está transformando rápidamente el panorama de TI y que ha cambiado la manera en que las empresas utilizan los ordenadores.

La virtualización es fundamental en Cloud Computing (Computación en la nube), ya que reduce la utilización de hardware, ahorra energía y costes, y hace posible ejecutar múltiples aplicaciones y varios sistemas operativos en el mismo servidor. A su vez, aumenta la utilización, la eficiencia y la flexibilidad del hardware existente en la data center.

Como alternativa tenemos la virtualización, que es la creación de una versión virtual de algún recurso tecnológico (con la intermediación de un software), por ejemplo, un dispositivo de almacenamiento, una plataforma de hardware, etc...

Los Hipervisores (o VMM Virtual Machine Monitors) son una pieza clave en este puzzle, ya que permiten que diferentes SO's, tareas y configuraciones de software coexistan en una misma máquina, abstrayendo los recursos físicos de esta para las distintas máquinas virtuales. La independencia entre estas diferentes máquinas virtualizadas está garantizada y proporcionan una interfaz única para el hardware.

6.6.1 Tipos fundamentales de Hipervisores

- ✓ **Nativo o Bare-Metal:** Donde el hipervisor es una capa entre el hardware y el sistema operativo. Aquí al sistema operativo se le denomina Dominio de Control y funciona sobre el hipervisor, y los invitados son los llamados Dominios Lógicos.
- ✓ **Hosted:** El hipervisor es una capa de software que funciona sobre el sistema operativo anfitrión, El invitado o cliente podrá ver el sistema operativo que ya esté funcionando, no podrá usar una capa con sistema operativo propio.

6.7 ¿Por qué es interesante utilizar virtualización y qué lo hace posible?

Como hemos dicho, la virtualización en Cloud Computing (Computación en la nube) ofrece varios beneficios, como el ahorro de tiempo y energía, la reducción de costes y la minimización de riesgos:

- ✓ Proporciona capacidad para administrar recursos de manera efectiva.
- ✓ Aumenta la productividad, ya que proporciona acceso remoto seguro.

- ✓ Proporciona prevención ante una posible pérdida de datos.

Para hacer posible la virtualización se utiliza un software conocido como hipervisor, monitor de máquina virtual o administrador de virtualización. Este software se sitúa entre el hardware y el sistema operativo y asigna la cantidad de acceso que tienen las aplicaciones y los sistemas operativos con el procesador y otros recursos de hardware.

6.7.1 Métodos de virtualización

Los métodos de virtualización que podemos usar según nuestras necesidades y capacidad de nuestro equipo son:

- ✓ **Emulación:** Con la emulación se consigue simular la máquina al completo, hardware incluido. Permite a los huéspedes de diferentes arquitecturas hardware funcionar dentro de un mismo entorno virtualizado. Comúnmente es utilizado para desarrollar software sin que el hardware esté disponible físicamente. La única pega de este tipo de virtualización es que, al compartirse el hardware de la máquina, el rendimiento general de esta baja considerablemente.
- ✓ **Virtualización completa:** A diferencia de la emulación los sistemas operativos a virtualizar están diseñados para funcionar en la misma arquitectura que el anfitrión. Ideal si se combina con hardware CMT, Intel VT, AMD-V, ya que estas CPU's controlan el acceso a instrucciones de Virtualización. La ventaja de este sistema es la flexibilidad puesto que permite virtualizar diferentes sistemas operativos de distintas plataformas, y por contras tenemos el rendimiento con CPU's distintas a las mencionadas y el hecho de que no se puedan emular otras arquitecturas.
- ✓ **Paravirtualización:** Este método permite virtualizar por software diferentes sistemas operativos, cada uno como si fuese un equipo independiente. El único requisito es que el sistema virtualizado (o invitado) sea soportado de forma nativa por la API (Interfaz de Programación de Aplicaciones), ya que lo que básicamente ofrece es una versión virtualizada 'modificada' del equipo anfitrión, ya que lo que se verá en el equipo invitado será la misma arquitectura del anfitrión. Las ventajas de este sistema de virtualización son el rendimiento que

llega a ofrecer incluso con CPU's convencionales, la escalabilidad, facilidad de gestión y el aislamiento entre usuarios. Sin embargo, todos los sistemas deberán tener la misma arquitectura y tendremos que estar modificando los sistemas invitados a cada cuánto.

- ✓ **Virtualización Ligera:** También denominada 'virtualización a nivel de sistema operativo', lo que hace es virtualizar servidores en la capa del sistema operativo (Kernel) y crea particiones aisladas o entornos virtuales en un único servidor físico e instancia de sistema operativo para maximizar los esfuerzos de administración de hardware, software y gestión de datos. El hipervisor tiene una capa base (normalmente un Kernel) que se carga directamente en el servidor creado en el Kernel del anfitrión. La siguiente capa superior muestra todo el hardware virtualizado asignado a la máquina virtual, y una vez cargada esta ya se puede cargar el sistema operativo y finalmente las aplicaciones con las que se trabajarán. Es el método que más ventajas implementa, ya que carga muy rápido los sistemas virtualizado, ofrece un rendimiento muy similar al de una instalación nativa, la capa de virtualización es ligera por lo que también será muy rápida de gestionar por el procesador, etc.

Luego, existen también otros tipos de virtualizaciones, llamémoslas 'menores', que básicamente son aplicaciones que emulan otras aplicaciones, bibliotecas, etc... Por mencionar tres de las más conocidas:

- ✓ **Virtualización de Bibliotecas:** Wine es un subconjunto de la API de Win32 para poder ejecutar aplicaciones Windows bajo otros sistemas operativos, como Linux.
- ✓ **Virtualización de aplicación:** Java Virtual Machine, es el mejor ejemplo de entorno de ejecución virtual, que cuenta con una API para la ejecución en diferentes plataformas.
- ✓ **Virtualización de escritorio:** Donde se implementa el escritorio como servicio al que accedemos remotamente para trabajar con nuestro disco duro. SunVDI nos podría servir de ejemplo. (Webinars, Introducción a la Virtualización, 2013)

6.8 OpenStack

OpenStack es uno de los proyectos de código abierto más emocionantes y rápidos del mundo en este momento. En un corto período de tiempo, el proyecto ha ganado una masa crítica de partidarios y está progresando rápidamente. Durante los últimos tres años, Red Hat ha estado completamente comprometido con el proyecto OpenStack, y ahora es uno de los principales contribuyentes al proyecto.

Openstack es la suma de varios componentes que tienen una función específica y que puedes instalar de manera separada o conjunta según la distribución que elijas. Es totalmente modular, puedes elegir que instalar y que no. Ha crecido gracias a la ayuda de compañías muy importantes que están invirtiendo mucho dinero. IBM, DELL, Red Hat, Mirantis y no para de crecer.

Las empresas quieren vender la infraestructura y el soporte, es decir la mano de obra, es decir los técnicos. Inicialmente fue un proyecto colaborativo entre la NASA y RackSpace para combinar las plataformas de almacenamiento y el Cloud Computing; que ha desembocado en un sistema colaborativo de desarrolladores y expertos en IT, para producir una plataforma de código abierto que ofrezca servicio de nubes públicas, privadas o híbridas.

Para que lo vayamos entendiendo, se trata de una serie de diferentes proyectos interrelacionados que nos van sirviendo diferentes servicios en una estructura en la nube. Nos permitirá a los usuarios desplegar máquinas virtuales, así como otro tipo de instancias con las que podremos llevar a cabo diferentes tareas sin salir de esta estructura en la nube, por lo que la accesibilidad desde cualquier equipo en cualquier parte está asegurada.

Además de esta accesibilidad, el hecho de que se trate de un software de código abierto nos dice que habrá cientos o miles de desarrolladores que mantendrán actualizado y añadirán funciones constantemente, así como si tenemos los conocimientos necesarios, nosotros mismo podremos acceder al código fuente y realizar las modificaciones que deseemos en la estructura del mismo.

6.8.1 Principales características de OpenStack

- ✓ **Pago por uso:** Empezamos por esta porque el dinero siempre va por delante en las grandes empresas y OpenStack facilita que solo pagues lo que usas y el tiempo que lo usas. Tanto si es un cloud (nube) público como si es privado.
- ✓ **Autónomo para el usuario:** El administrador no necesita intervenir cuando un usuario necesita desplegar instancias. Lo puede hacer todo de manera autónoma y sencilla. Montarlo y dejarlo funcionando es otra historia.
- ✓ **Escalable:** Usa lo que necesites. De hecho, más que escalable OpenStack lo define como elástico. Los recursos que necesites te los da en cuestión de segundos de manera que pueden aumentar o disminuir según te convenga, lo que va unido al concepto de pago por uso.
- ✓ **Código abierto:** No sólo significa gratis, significa que cualquiera puede aportar y consultar el código. Tiene una licencia Apache 2.0 y no tiene versión de pago, aunque hay empresas que si cobran por el soporte o el uso de su sistema operativo (Por ejemplo, Red Hat OpenStack). (cero, 2017)

6.8.2 Módulos o componentes de OpenStack

Como buen software Open Source, el modularidad es una base importantísima en este caso, tanto que OpenStack se puede subdividir en los varios servicios que componen el núcleo de la solución general. Estos, son comunes en la mayoría (por no decir en todas) de las distribuciones o en las instalaciones en las que venga pre-instalado, y son mantenidos oficialmente por la comunidad de desarrollo.

- ✓ **Nova:** Es el proyecto OpenStack que proporciona una manera de aprovisionar instancias de proceso (también conocidos como servidores virtuales). Nova admite la creación de máquinas virtuales, servidores baremetal (mediante el uso de ironic) y tiene soporte limitado para contenedores del sistema. Nova se ejecuta como un conjunto de demonios encima de los servidores Linux existentes para proporcionar ese servicio. (OpenStack.org, 2020)
- ✓ **Keystone:** Es un servicio de OpenStack que proporciona autenticación de cliente de API, detección de servicios y autorización multiinquilino distribuida

mediante la implementación de la API de identidad de OpenStack. (OpenStack.org, 2019)

- ✓ **Ceilometer:** Es un servicio de recopilación de datos que proporciona la capacidad de normalizar y transformar datos en todos los componentes principales actuales de OpenStack con el trabajo en curso para admitir futuros componentes de OpenStack.

Ceilometer es un componente del proyecto de telemetría. Sus datos se pueden utilizar para proporcionar capacidades de facturación, seguimiento de recursos y alarma de clientes en todos los componentes principales de OpenStack. (OpenStack.org, 2019)

- ✓ **Cinder:** Es el servicio OpenStack Block Storage para proporcionar volúmenes a máquinas virtuales Nova, hosts de bare metal, contenedores y mucho más. (OpenStack.org, 2020)

- ✓ **Glance:** El proyecto Image service (glance) proporciona un servicio donde los usuarios pueden cargar y detectar activos de datos que están destinados a usarse con otros servicios. Esto incluye actualmente imágenes y definiciones de metadatos. (OpenStack.org, 2019)

- ✓ **Horizon:** Es la implementación canónica de OpenStack's Dashboard, que proporciona una interfaz de usuario basada en web para servicios de OpenStack como Nova, Swift, Keystone, etc. (OpenStack.org, 2019)

- ✓ **Neutron:** Es un proyecto de OpenStack para proporcionar "conectividad de red como servicio" entre dispositivos de interfaz (por ejemplo, vNICs) gestionados por otros servicios de OpenStack (por ejemplo, nova). (OpenStack.org, 2020)

- ✓ **Swift:** Es un almacén de objetos/blobs altamente disponible, distribuido y eventualmente consistente. Las organizaciones pueden usar Swift para almacenar muchos datos de forma eficiente, segura y barata. (OpenStack.org, 2020)

- ✓ **Aodh:** El proyecto de servicio Alarmante (aodh) proporciona un servicio que permite desencadenar acciones basadas en reglas definidas con datos de métricas o eventos recopilados por Ceilometer o Gnocchi. (OpenStack.org, 2018)

- ✓ **Ironic:** Es un proyecto de OpenStack que aprovisiona máquinas bare metal. Se puede utilizar de forma independiente o como parte de OpenStack Cloud, y se integra con los servicios OpenStack Identity (keystone), Compute (nova), Network (neutron), Image (glance) y Object (swift). (OpenStack.org, 2019)
- ✓ **Heat:** Es un servicio para organizar aplicaciones en la nube compuestas mediante un formato de plantilla declarativa a través de una API REST nativa de OpenStack. (OpenStack.org, 2019)

6.8.3 OpenStack Stein

La comunidad OpenStack lanzó Stein, la versión 19 del software de infraestructura de nube de código abierto más ampliamente implementado. OpenStack Stein mejora la gestión de bare metal y de red, al tiempo que refuerza la funcionalidad de los contenedores.

La 19a versión de OpenStack también incluye actualizaciones de redes para casos de uso de computación perimetral y NFV y mejoras en la administración y el seguimiento de recursos.

Entre las docenas de mejoras proporcionadas en Stein, tres aspectos destacados son:

- ✓ Fortalecimiento de la funcionalidad de contenedores.
- ✓ Actualizaciones de redes para admitir casos de uso de 5G, computación perimetral y virtualización de funciones de red (NFV).
- ✓ Mejoras en la gestión y el seguimiento de recursos. (OpenStack.org, s.f.)

6.8.4 RDO

RDO es una comunidad que ofrece la última versión de OpenStack en Red Hat Enterprise Linux, CentOS, Scientific Linux y otras plataformas basadas en RHEL, y en Fedora. Esta distribución de OpenStack apoyada por RDO incluye el instalador Pack Stack, lo que hace que sea muy fácil arrancar y hacer crecer su instalación RDO.

Si bien inicialmente RDO consistía en los proyectos centrales de OpenStack, desde la reorganización de Big Tent, esa lista ha crecido. Los proyectos empaquetados en RDO dependen de lo que los usuarios del proyecto indiquen que necesitan y de lo que la comunidad intensifica para empaquetar. (Hat, 2016)

6.8.5 Métodos de instalación de OpenStack

1. **DevStack:** Es una serie de scripts extensibles que se utilizan para abrir rápidamente un entorno OpenStack completo basado en las últimas versiones de todo, desde git-master. Se utiliza de forma interactiva como entorno de desarrollo y como base para gran parte de las pruebas funcionales del proyecto OpenStack.
2. **PackStack:** Es una herramienta que incorpora RDO para realizar el despliegue de OpenStack lanzando recetas de puppet pre configuradas y un solo archivo de configuración que es capaz de realizar la instalación en uno o varios nodos.
3. **Manual:** Se realiza mediante el uso de comandos para la instalación y configuración de cada uno de los componentes, logrando así la comunicación entre ellos y entre los diferentes nodos en los que se desea desplegar OpenStack de forma manual.

7 Diseño Metodológico

7.1 Tipo de investigación

Proyectiva: Es un tipo de estudio que consiste en buscar soluciones a distintos problemas, analizando de forma integral todos sus aspectos y proponiendo nuevas acciones que mejoren una situación de manera práctica y funcional.

Este tipo de investigación propone modelos que generen soluciones a necesidades concretas de tipo social, organizacional, ambiental o de algún área especial del conocimiento, con miras al futuro de cada contexto y mediante su análisis situacional.

7.2 Etapas de la investigación

7.2.1 Etapa I. Etapa de investigación

En esta etapa indagamos los pasos necesarios para la implementación de una nube privada y se recopiló la Información acerca del tema, haciendo búsquedas en libros, sitios web, etc.

- 1) Investigación:** En esta sub-etapa se recopiló la Información acerca del tema, haciendo búsquedas en libros, sitios web, etc.
- 2) Identificación:** En esta sub-etapa se delimita el problema haciendo uso de la información recopilada en la sub-etapa anterior.

7.2.2 Etapa II. Selección de las herramientas y versión

Luego de identificar y delimitar el problema, en esta etapa se ideó una estrategia para darle solución, para lo cual se probaron diversas herramientas en un entorno controlado. Después de identificar el problema en esta etapa se seleccionó una versión para la instalación de la nube privada y la forma en la que se realizara la instalación, esto se hizo a través de un análisis de las distintas formas de instalación determinando la forma más óptima tomando en cuentas las variantes que presentan cada versión del software a implementar.

7.2.3 Etapa III. Configuración de la nube privada

Aquí se procedió a configurar las herramientas previamente evaluadas:

1) Infraestructura de red sobre la cual se desplegará la nube privada

Se realizó la selección de los componentes de la plataforma que se van a instalar y la manera en la que estos se agruparon en cada nodo.

2) Instalación de la nube privada

Se instaló la nube privada sobre la infraestructura de red de acuerdo a la capacidad de los equipos, realizando una instalación multinodo utilizando OpenStack Stein sobre CentOS 7 empleando PackStack RDO.

3) Interfaz de red en el nodo de red

Se realizó la configuración de la interfaz de red en el nodo de red en modo bridge para que la nube privada pudiera tener conectividad con la red física y así poder acceder a las instancias dentro de la nube.

4) Creación y configuración de elementos básicos

Se crearon los elementos necesarios como fueron red externa, catálogo de imágenes, usuarios, proyectos y sabores para poder ofrecer el servicio de nube privada para el usuario final.

7.2.4 Etapa IV. Prueba de las herramientas

En esta etapa se procedió a realizar pruebas donde los diferentes usuarios crearon su propia infraestructura con diferentes servicios lo que permitió validar el correcto funcionamiento de la nube.

7.2.5 Etapa V. Documentación

Redacción del informe final: En este acápite logramos redactar los resultados de todo el proceso que llevo nuestro proyecto, para llegar a definir las conclusiones de éste y así lograr la finalización de nuestro informe.

7.3 Materiales hardware y software

Para la realización de este proceso, fue necesario el empleo de los siguientes materiales hardware y software:

7.3.1 Software

Software	Versión
CentOS	Linux 7 (Core)
OpenStack	Stein
PackStack	Stein

7.3.2 Hardware

Nodos	CPU	Memoria RAM	Disco Duro
Controlador	Intel(R) Core(TM) i3-2100 CPU @ 3.10 GHz	12 GB DDR3	500 GB
Computo	Intel(R) Core(TM) i3-2100 CPU @ 3.10 GHz	4 GB DDR3	500 GB
Red	Intel(R) Core(TM) i3-2100 CPU @ 3.10 GHz	4 GB DDR3	500 GB

Equipos de Red	Modelo	Puertos	Características
Router Xiaomi	Wireless 11n	1 WAN, 2 LAN, USB Power-in.	2.4 GHz Wifi, 2,4GHz ROM, 16M flash, 64M DDR2.
Switch Cisco Catalyst	WS-C3560-8PC-S	8 Ethernet, 1 dual-purpose, 1 SFP.	Capa 2 y 3. Soporta PoE IEEE 802.3af y Cisco pre-estandar. Enrutamiento básico estático y dinámico.
5 Cables Ethernet	Categoría 5/Rj45		Ancho banda de hasta 100 MHz. Velocidad 10 o 100 Mb Máximo 100 metros.

8 Resultados

Las imágenes que se verán a continuación, son el resultado de aplicar las herramientas antes mencionadas.

8.1 Configuración de la nube privada

Se realizó la selección de la infraestructura donde se instaló la plataforma y la manera en que se agrupó cada componente, de acuerdo a las características de los equipos que se dispuso, seleccionando más componentes para el nodo controlador porque éste poseía más recursos hardware que los demás (Ver Figura 1).

Podemos observar en la imagen la manera en la que se distribuyeron los componentes en la infraestructura de red de acuerdo a los recursos de los equipos disponibles. En el nodo controlador se instaló la mayoría de los componentes, los cuales son Nova, Ceilometer, Swift, AODH, Cinder y Glance, instalando también en el nodo de cómputo el componente de Nova-Compute y en el nodo de red el componente de Neutron.

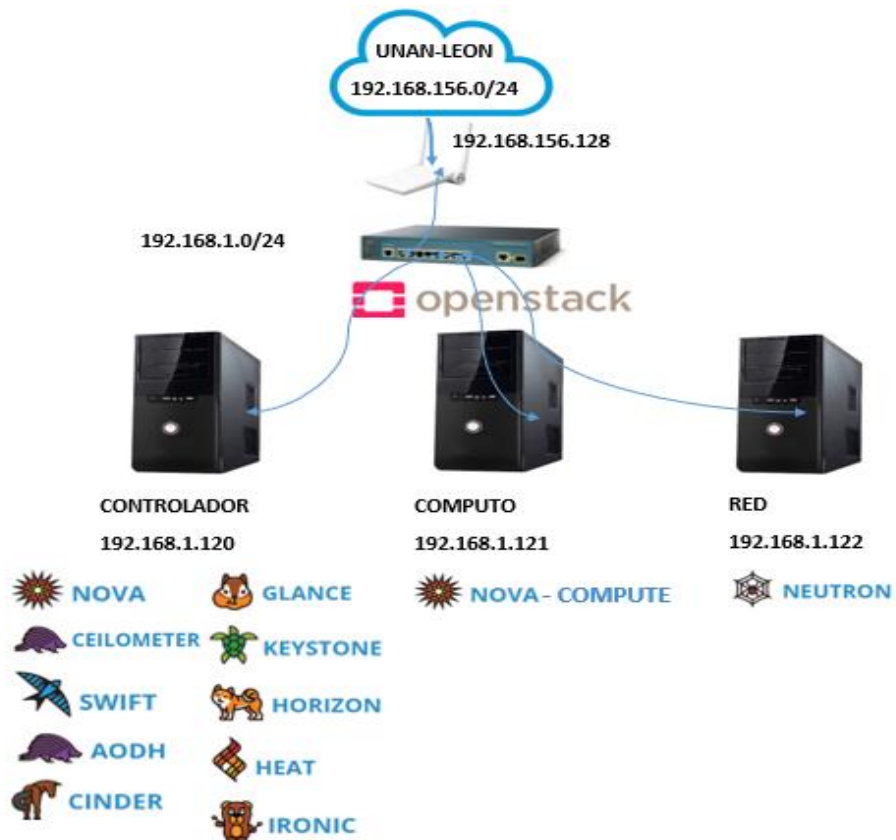


Figura 1: Infraestructura de red de la nube

Instalación de la nube privada en la infraestructura.

Primeramente, vemos como se instaló CentOS 7 en cada uno de los equipos, en la cual se estableció el hostname de estos a: Controlador, Cómputo y Red, respectivamente.

Una vez

Instalados los 3 sistemas se configuró la interfaz de red de todos los nodos, quedando de la siguiente manera (Ver figura 2,3,4).

Nodo Controlador:

```
TYPE="Ethernet"
BOOTPROTO="static"
DEFROUTE="yes"
PEERDNS="no"
NAME="enp2s0"
DEVICE="enp2s0"
ONBOOT="yes"
IPADDR="192.168.1.120"
NETMASK="255.255.255.0"
GATEWAY="192.168.1.1"
NM_CONTROLLED="no"
```

Figura 2: Interfaz de red del nodo controlador.

Nodo Cómputo:

```
TYPE="Ethernet"
BOOTPROTO="static"
DEFROUTE="yes"
PEERDNS="no"
NAME="enp2s0"
DEVICE="enp2s0"
ONBOOT="yes"
IPADDR="192.168.1.121"
NETMASK="255.255.255.0"
GATEWAY="192.168.1.1"
NM_CONTROLLED="no"
```

Figura 3: Interfaz de red del nodo controlador.

Nodo de red:

```
TYPE="Ethernet"
BOOTPROTO="static"
DEFROUTE="yes"
PEERDNS="no"
NAME="enp2s0"
DEVICE="enp2s0"
ONBOOT="yes"
IPADDR="192.168.1.122"
NETMASK="255.255.255.0"
GATEWAY="192.168.1.1"
NM_CONTROLLED="no"
```

Figura 4: Interfaz de red del nodo de red.

Posteriormente se reinició el demonio de red. (Ver figura 5)

```
root@red# systemctl restart network
```

Figura 5: Reinicio del demonio de red en nodo de red

Configuración de los DNS en cada uno de los nodos en el archivo /etc/resolv.conf. (Ver figura 6).

```
nameserver 192.168.156.1
```

Figura 6: Configuración de los DNS

Configuración de las traducciones en el archivo /etc/hosts para una comunicación a través de nombres entre los nodos. (Ver figura 7).

```
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.1.120 controlador
192.168.1.121 computo
192.168.1.122 red
```

Figura 7: Configuración de hosts para generar comunicación entre nodos.

Deshabilitamos los siguientes servicios. (Ver figura 8).

```
systemctl stop NetworkManager
systemctl disable NetworkManager
systemctl stop firewalld
systemctl disable firewalld
```

Figura 8: Servicios deshabilitados.

Habilitamos el servicio de red (Ver figura 9)

```
systemctl start network
systemctl enable network
```

Figura 9: Servicio de red habilitado.

Editamos el archivo /etc/sysconfig/selinux para deshabilitar Selinux. (Ver figura 10)

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#   enforcing - SELinux security policy is enforced.
#   permissive - Selinux prints warnings instead of enforcing.
#   disabled - No Selinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#   targeted - Targeted processes are protected,
#   minimum - Modification of targeted policy. Only selected processes are
protected.
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Figura 10: Selinux deshabilitado.

Editamos el archivo `/etc/yum.repos.d/CentOS-Base.repo` para habilitar el repositorio de CentOS Plus. (Ver figura 11)

```
# unless you are manually picking other mirrors.
#
# If the mirrorlist= does not work for you, as a fall back you can try the
# remarked out baseurl= line instead.
#
#
[base]
name=CentOS-$releasever - Base
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo
=os&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#released updates
[updates]
name=CentOS-$releasever - Updates
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo
=updates&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/updates/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#additional packages that may be useful
[extras]
name=CentOS-$releasever - Extras
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo
=extras&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/extras/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#additional packages that extend functionality of existing packages
[centosplus]
name=CentOS-$releasever - Plus
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo
=centosplus&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/centosplus/$basearch/
gpgcheck=1
enabled=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
```

Figura 11: Repositorio de CentOS Plus habilitado.

Actualizamos las listas de paquetes de los repositorios (Ver figura 12)

```
# yum update -y
```

Figura 12: Actualizar repositorios.

Se buscaron los repositorios de OpenStack (Ver figura 13)

```
# yum search openstack
centos-release-openstack-train.noarch
centos-release-openstack-stein.noarch
centos-release-openstack-rocky.noarch
centos-release-openstack-queens.noarch
```

Figura 13: Lista de repositorios actualizada.

Se instala el repositorio en su versión más reciente. (Ver figura 14)

```
# yum install -y centos-release-openstack-stein.noarch
```

Figura 14: Instalamos repositorio Stein.

Realizamos nuevamente la actualización de listas de paquetes de los repositorios. (Ver figura 15)

```
# yum update -y
```

Figura 15: Actualizando repositorios.

La siguiente sección de comandos se ejecuta en el nodo controlador:

Instalación de PackStack. (Ver figura 16)

```
root@controlador# yum install -y openstack-packstack
```

Figura 16: Instalacion de los puppet.

Se genera el archivo de respuesta para el despliegue de Openstack. (Ver figura 17)

```
root@controlador# packstack --gen-answer-file=/root/answer.txt
```

Figura 17: Generar el archivo de respuesta.

Abrimos el archivo de respuesta. (Ver figura 18)

```
root@controlador# vi answer.txt
```

Figura 18: Archivo de respuesta

Dentro del archivo de respuesta editamos las siguientes líneas. (Ver figura 19)

```
CONFIG_CONTROLLED_HOST=192.168.1.120
CONFIG_COMPUTE_HOSTS=192.168.1.121
CONFIG_NETWORK_HOSTS=192.168.1.122
CONFIG_NTP_SERVERS=0.pool.ntp.org,1.pool.ntp.org,2.pool.ntp.org
CONFIG_CINDER_VOLUMES_SIZE=60G
CONFIG_PROVISION_DEMO=n
```

Figura 19: Editamos las siguientes líneas del archivo.

El archivo de respuesta coloca como dirección IP predeterminada para la instalación de todos los componentes, la dirección IP de la máquina en que se genera dicho archivo (IP del Nodo Controlador para nuestro caso). Ya que en nuestro trabajo se realizó una instalación multinodo, fue necesario cambiar las direcciones IP's de los hosts de cómputo y red en el archivo de respuesta a como se observa en las 3 primeras líneas.

Una vez modificado el fichero de respuesta adecuadamente, procedemos a realizar la instalación ejecutando el siguiente comando. (Ver figura 20)

```
root@controlador# packstack --answer-file=/root/answer.txt
```

Figura 20: Ejecución del archivo de respuesta.

El proceso de instalación tomará algún tiempo, el cual dependerá de las capacidades de los equipos y el ancho de banda de la red. En nuestro caso este proceso duró aproximadamente 40 minutos. Al finalizar la instalación deberíamos de visualizar el siguiente mensaje en el nodo controlador. (Ver figura 21)

```
**** Installation completed successfully ****
```

Figura 21: Instalación exitosa.

Se configuró la interfaz de red en el nodo de red en modo bridge quedando de la siguiente manera. (Ver figura 22)

```
TYPE="OVSPort"  
DEVICETYPE="ovs"  
OVS_BRIDGE="br-ex"  
NAME="enp2s0"  
DEVICE="enp2s0"  
ONBOOT="yes"
```

Figura22: Interfaz en nodo de red de modo Bridge

Se creó un nuevo archivo en el directorio `/etc/sysconfig/network-scripts/` con el nombre `ifcfg-br ex`. (Ver figura 23)

```
TYPE="OVSBridge"  
BOOTPROTO="none"  
DEFROUTE="yes"  
PEERDNS="no"  
NAME="br-ex"  
DEVICE="br-ex"  
ONBOOT="yes"  
IPADDR="192.168.1.122"  
NETMASK="255.255.255.0"  
GATEWAY="192.168.1.1"  
NM_CONTROLLED="no"
```

Figura23: Bridge de la interfaz.

Una vez hecha esta configuración se reinició el demonio de red (Ver figura 24).

```
root@red# systemctl restart network
```

Figura24: Reinicio del demonio de red

Se crearon los elementos necesarios para poder ofrecer el servicio de nube privada al usuario final, los cuales son: red externa, catálogo de imágenes, usuarios, proyectos y sabores.

Se creó la red externa y a su vez la subred de ésta, para conectar la nube privada con la red física y así el usuario pueda conectarse a la instancia que tiene en la nube. Se reservó un rango del direccionamiento de la red física (192.168.1.140-192.168.1.160), de donde se tomarán las IP's a ser empleadas por el usuario en la creación de Router's para la comunicación de la red externa con su red privada. (Ver figura 25)

Redes

Proyecto	Nombre de la red	Subredes asociadas	Agentes DHCP	Compartido	Externa	Estado	Estado de administración	Zonas de Disponibilidad	Acciones
admin	Red-Externa	Subred-Externa 192.168.1.0/24	0	Si	Si	Activo	ARRIBA	-	Editar red

Figura 25: Creacion de la red externa

Se creó un catálogo de imágenes el cual estará disponible para el usuario con la imagen Debian 10, la que se le definieron como recursos mínimos 5 GB de Disco 512 MB de RAM y 1 VCPU. (Ver figura 26)

Imágenes

Propietario	Nombre	Tipo	Estado	Visibilidad	Protegido	Formato de disco	Tamaño	Acciones
admin	Debian 10	Imagen	Activo	Público	No	QCOW2	547.89 MB	Iniciar

Figura 26: Imagen Debian

Como podemos observar se crearon usuarios los cuales van a poder acceder a la nube con las credenciales brindados por el administrador. (Ver figura 27)

<input type="checkbox"/>	heat_admin	-	heat_admin@localhost	13b7d8df52c54dbc82b97f9fd7f0e259	Sí	-	Editar
<input type="checkbox"/>	cinder	-	cinder@localhost	1850e5fc8f6f4ebbbdc6f53ba0b55860	Sí	Default	Editar
<input type="checkbox"/>	swift	-	swift@localhost	1a1df1ebb2bf4e4080810a1693b640b0	Sí	Default	Editar
<input type="checkbox"/>	Claudia Caceres	-	caceresclaudia780@gmail.com	1e08e85f202840d6972ae9bfa84258a5	Sí	Default	Editar
<input type="checkbox"/>	gnocchi	-	gnocchi@localhost	36052476d2424617851cf64660cc5b36	Sí	Default	Editar
<input type="checkbox"/>	ceilometer	-	ceilometer@localhost	3d622fa89950488eb8ef2683a62ece71	Sí	Default	Editar
<input type="checkbox"/>	heat-cfn	-	heat-cfn@localhost	3fb6cc46627c481cab2abae0dd801395	Sí	Default	Editar
<input type="checkbox"/>	heat	-	heat@localhost	4fbd6444a6684e119d502fb232fd2f85	Sí	Default	Editar
<input type="checkbox"/>	Jerson Pastran	-	jerlexander03@gmail.com	68faf92762d34473b6fa24678b7a8091	Sí	Default	Editar
<input type="checkbox"/>	neutron	-	neutron@localhost	8bbbe63e17224ef2a10152a684a822f3	Sí	Default	Editar
<input type="checkbox"/>	glance	-	glance@localhost	baa5bba6acc40a6b2aed653805f6f96	Sí	Default	Editar
<input type="checkbox"/>	Rodolfo Cortedano	-	cortedano.rodolfo@gmail.com	bca8b9008d05408db024be458b1ffbac	Sí	Default	Editar
<input type="checkbox"/>	nova	-	nova@localhost	c587e3c3bdfc4afa918840648fea4aea	Sí	Default	Editar

Figura 27: Usuarios creados por el administrador.

Se realizó la creación de un proyecto para cada usuario, en el cual ellos van a desplegar la infraestructura que deseen dentro de la nube privada. (Ver figura 28)

Proyectos

Nombre de proyecto =

Mostrando 5 artículos

<input type="checkbox"/>	Nombre	Descripción	ID del proyecto	Nombre de dominio	Habilitado	Acciones
<input type="checkbox"/>	Proyecto-CC		17e2955fce084851bfedab26a62317dd	Default	Sí	Administrar Miembros
<input type="checkbox"/>	services	Tenant for the openstack services	250bd436812042bdb5a4bb4fcd9d622	Default	Sí	Administrar Miembros
<input type="checkbox"/>	Proyecto-RC		37317c9fc84b4ca5a06f39c0790077e1	Default	Sí	Administrar Miembros
<input type="checkbox"/>	admin	admin tenant	71cc8207f4194ccca56dedc4ea6cc449	Default	Sí	Administrar Miembros
<input type="checkbox"/>	Proyecto-JP		c08291eca4644d53afc856face3d40d6	Default	Sí	Administrar Miembros

Mostrando 5 artículos

Figura 28: Proyectos creados por el administrador.

Como no se contaba con grandes recursos de cómputo se creó un sabor específico para instancias que se quieran ejecutar con Debian. Esto se hizo debido a que los sabores existentes desperdiciaban demasiados recursos al ejecutar una instancia con un sistema bastante liviano. (Ver figura 29)

Mostrando 6 artículos

<input type="checkbox"/>	Nombre del sabor	VCPU	RAM	Disco raíz	Disco efimero	Disco de intercambio (swap)	Factor RX/TX	ID	Público	Metadatos	Acciones
<input type="checkbox"/>	Debian 10	1	512MB	5GB	0GB	0MB	1,0	9625a408-1f95-4ba1-9680-151c7c87b2b8	Sí	no	Actualizar metadatos ▾
<input type="checkbox"/>	m1.large	4	8GB	80GB	0GB	0MB	1,0	4	Sí	no	Actualizar metadatos ▾
<input type="checkbox"/>	m1.medium	2	4GB	40GB	0GB	0MB	1,0	3	Sí	no	Actualizar metadatos ▾
<input type="checkbox"/>	m1.small	1	2GB	20GB	0GB	0MB	1,0	2	Sí	no	Actualizar metadatos ▾
<input type="checkbox"/>	m1.tiny	1	512MB	1GB	0GB	0MB	1,0	1	Sí	no	Actualizar metadatos ▾
<input type="checkbox"/>	m1.xlarge	8	16GB	160GB	0GB	0MB	1,0	5	Sí	no	Actualizar metadatos ▾

Figura 29: Creacion del sabor para Debian.

8.2 Prueba de las herramientas

Se realizaron pruebas donde los diferentes usuarios crearon su propia infraestructura en la nube con diferentes servicios lo que permitió validar el correcto funcionamiento.

Este grafico pertenece al usuario Rodolfo Cortedano y al proyecto Proyecto-RC. Podemos observar que el usuario creó un router el cual se conecta a la red externa creada por el administrador y en el otro extremo se conectan dos redes privadas creadas por el usuario. En esta red privada el usuario ejecutó dos instancias de la imagen Debian agregada en el catálogo, ambas con un servicio web. (Ver figura 30)

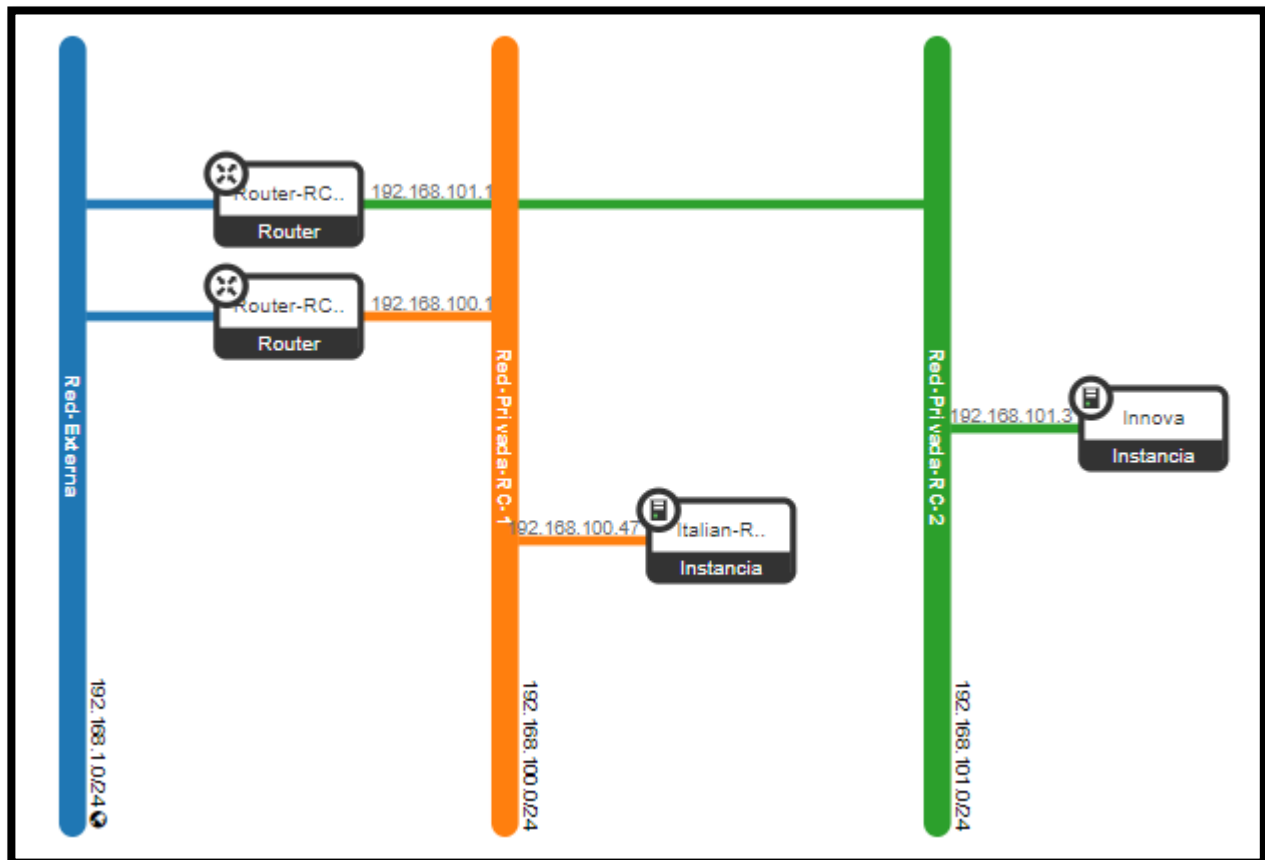


Figura 30: Infraestructura en la nube del usuario Rodolfo Cortedano y su proyecto.

Esta topología fue creada por el usuario Jerson Pastrán, el cual tiene asignado el proyecto Proyecto-JP. El cual creó un router que se conecta a la red externa y en el otro extremo se conecta una red privada creada por el usuario, conectando a dicha red dos instancias ambas con el servicio web. (Ver figura 31)

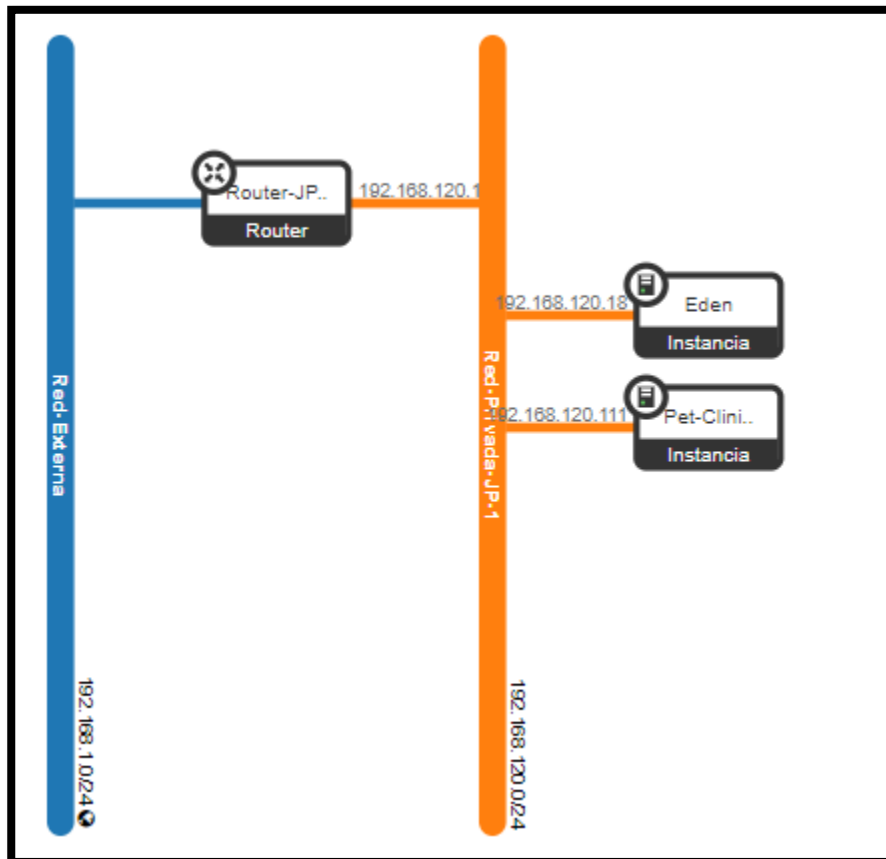


Figura 31: Infraestructura en la nube del usuario Jerson Pastrán y su proyecto.

En la imagen siguiente podemos observar una topología que pertenece al usuario Claudia Caceres que se le asignó el proyecto Proyecto-CC a diferencia de los anteriores este ejecuta una sola instancia con el servicio DNS conectada a la red privada que creó respectivamente. (Ver figura 32)

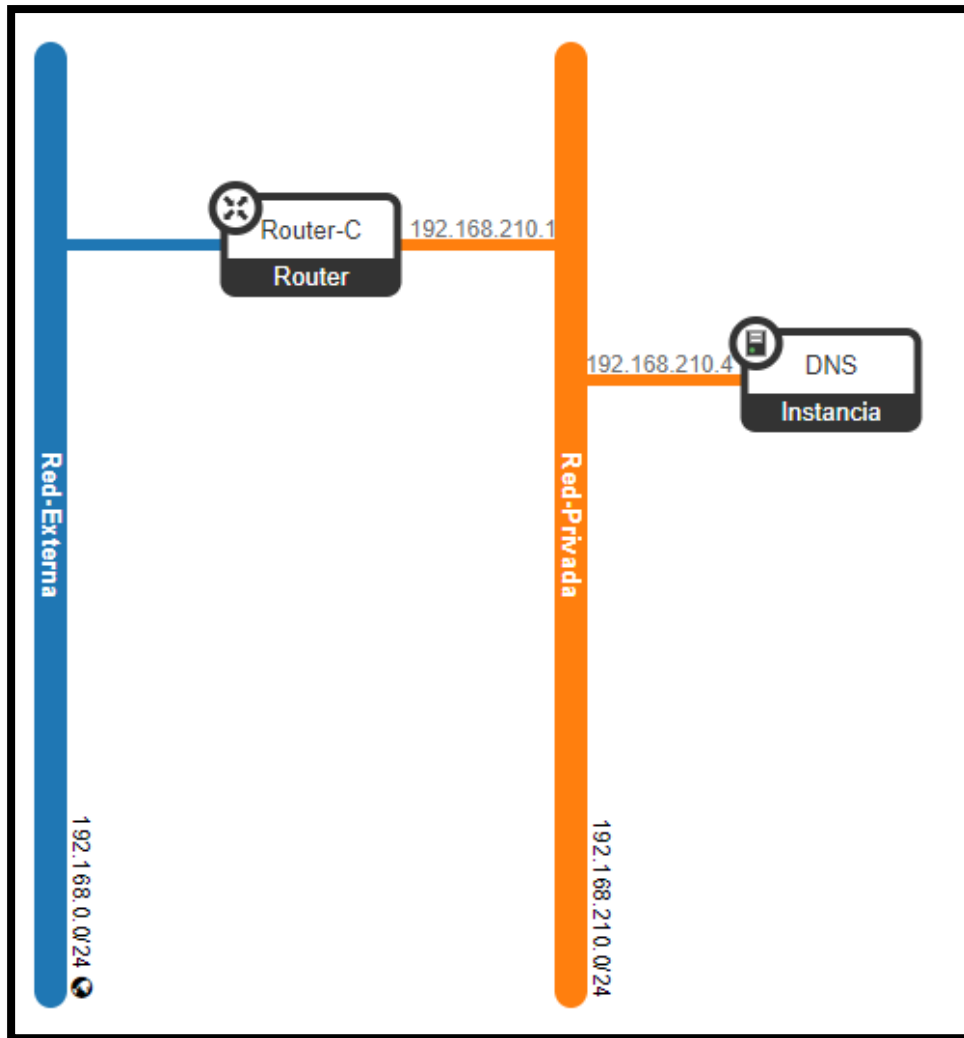


Figura 32: Infraestructura en la nube del usuario Claudia Caceres y su proyecto.

Como administrador podemos observar las instancias ejecutadas por todos los usuarios mostrando el proyecto al que pertenecen, el host en el que se ejecutan, el nombre, la dirección IP de la red privada a la que pertenecen y la IP flotante que tienen reservada, el sabor que está ejecutando la instancia y otros datos que ayudan a ver los detalles de las mismas. (Ver figura 33).

Instancias

Mostrando 5 artículos

<input type="checkbox"/>	Proyecto	Host	Nombre	Nombre de la imagen	Dirección IP	Sabor	Estado	Tarea	Estado	Age	Acciones	
<input type="checkbox"/>	Proyecto-CC	computo	DNS	-	192.168.110.74, 192.168.1.160	Debian 10	Activo		Ninguno	Corriendo	3 días	Rescue Instance ▾
<input type="checkbox"/>	Proyecto-JP	computo	Eden	-	192.168.120.18, 192.168.1.146	Debian 10	Activo		Ninguno	Corriendo	3 días	Rescue Instance ▾
<input type="checkbox"/>	Proyecto-JP	computo	Pet-Clinic	-	192.168.120.111, 192.168.1.140	Debian 10	Activo		Ninguno	Corriendo	3 días	Rescue Instance ▾
<input type="checkbox"/>	Proyecto-RC	computo	Innova	-	192.168.101.3, 192.168.1.144	Debian 10	Activo		Ninguno	Corriendo	3 días	Rescue Instance ▾
<input type="checkbox"/>	Proyecto-RC	computo	Italian-Restaurant	-	192.168.100.47, 192.168.1.155	Debian 10	Activo		Ninguno	Corriendo	3 días	Rescue Instance ▾

Mostrando 5 artículos

Figura 33: Instancias ejecutadas por todos los usuarios.

9 Conclusiones

Al culminar nuestro trabajo de tesis se llegó a las siguientes conclusiones:

- ✓ Se logró desplegar una infraestructura de red compuesta por 3 máquinas físicas (controlador, cómputo y red) sobre la cual se instaló OpenStack multinodo, agrupando en ellas los diversos componentes.
- ✓ Se realizaron las configuraciones para permitir comunicación entre la red física y la nube privada. Así mismo se crearon los elementos necesarios para poder ofrecer la infraestructura como servicio.
- ✓ Por último, se realizaron las pruebas de funcionamiento de la nube privada creando usuarios y asignándoles un proyecto en el cuál desplegaron su propia infraestructura.

Por todo lo anterior podemos concluir que:

Logramos implementar una nube privada multinodo basada en OpenStack utilizando la versión Stein sobre CentOS 7 empleando PackStack RDO para su instalación.

10 Recomendaciones

En base a los resultados obtenidos realizamos las siguientes recomendaciones:

Recomendaciones de uso o implementación:

- ✓ Capacitar al usuario final como manejar la plataforma y los elementos necesarios que debe de crear para formar su infraestructura y así la correcta ejecución de una instancia.
- ✓ Mantener el número de instancias limitadas a los recursos de hardware del nodo de cómputo.

Mejoras o trabajos futuros:

- ✓ Disponer siempre de material actualizado, cuando se realice una actualización de la plataforma y si la nueva versión es compatible con los requerimientos físicos de los equipos ya que la documentación varía de acuerdo a la versión.
- ✓ Proveer más nodos de cómputo si se desea ejecutar gran número de instancias, ya que el nodo de cómputo es el que brinda los recursos de hardware para las instancias.

11 Bibliografía

- AprenderCompartiendo. (15 de 06 de 2016). *Almacenando en la Nube, ventajas y desventajas*. Recuperado el 08 de 04 de 2019, de <https://aprendercompartiendo.com/la-nube-ventajas-desventajas/>
- cero, V. d. (10 de 03 de 2017). *¿Qué es Openstack y por qué deberías saber de su existencia?* Recuperado el 08 de 04 de 2019, de <https://www.tuyu.es/modelos-servicios-en-la-nube/>
- Gluppi. (30 de 04 de 2018). *QUÉ ES CLOUD COMPUTING, PARA QUÉ SIRVE Y CÓMO FUNCIONA*. Recuperado el 08 de 04 de 2019, de <https://gluppi.com/que-es-cloud-computing/>
- Hat, R. (16 de 03 de 2016). *RDO*. Recuperado el 08 de 04 de 2019, de <https://www.rdoproject.org/rdo/>
- llimit. (19 de 11 de 2018). *Diferencias entre Nube Pública, Privada e Híbrida*. Recuperado el 08 de 04 de 2019, de <https://blog.ilimit.com/es/diferencias-entre-nube-publica-privada-e-hibrida>
- OpenStack. (s.f.). Recuperado el 08 de 04 de 2019, de <https://docs.openstack.org/devstack/latest/>
- OpenStack.org. (31 de 10 de 2018). *Welcome to Aodh's documentation!* Recuperado el 23 de 06 de 2020, de https://docs.openstack.org/aodh/latest/?_ga=2.47261927.1121178038.1592935435-61574442.1585243550
- OpenStack.org. (28 de 06 de 2019). *Horizon: The OpenStack Dashboard Project*. Recuperado el 23 de 06 de 2020, de https://docs.openstack.org/horizon/latest/?_ga=2.119277897.1121178038.1592935435-61574442.1585243550
- OpenStack.org. (22 de 07 de 2019). *Keystone, the OpenStack Identity Service*. Recuperado el 23 de 06 de 2020, de

https://docs.openstack.org/keystone/latest/?_ga=2.117402822.1121178038.1592935435-61574442.1585243550

OpenStack.org. (19 de 09 de 2019). *Welcome to Ceilometer's documentation!*

Recuperado el 23 de 06 de 2020, de

https://docs.openstack.org/ceilometer/latest/?_ga=2.110651973.1121178038.1592935435-61574442.1585243550

OpenStack.org. (08 de 08 de 2019). *Welcome to Glance's documentation!* Recuperado el 23 de 06 de 2020, de

https://docs.openstack.org/glance/latest/?_ga=2.50973929.1121178038.1592935435-61574442.1585243550

OpenStack.org. (06 de 09 de 2019). *Welcome to Ironic's documentation!* Recuperado el 23 de 06 de 2020, de

https://docs.openstack.org/ironic/latest/?_ga=2.43771493.1121178038.1592935435-61574442.1585243550

OpenStack.org. (20 de 03 de 2019). *Welcome to the Heat documentation!* Recuperado el 23 de 06 de 2020, de

https://docs.openstack.org/heat/latest/?_ga=2.106613955.1121178038.1592935435-61574442.1585243550

OpenStack.org. (03 de 03 de 2020). *OpenStack Block Storage (Cinder) documentation.* Recuperado el 23 de 06 de 2020, de

https://docs.openstack.org/cinder/latest/?_ga=2.72510803.1121178038.1592935435-61574442.1585243550

OpenStack.org. (11 de 03 de 2020). *OpenStack Compute (nova).* Recuperado el 23 de 06 de 2020, de

https://docs.openstack.org/nova/latest/?_ga=2.76613973.1121178038.1592935435-61574442.1585243550

OpenStack.org. (08 de 01 de 2020). *Welcome to Neutron's documentation!* Recuperado el 23 de 06 de 2020, de

https://docs.openstack.org/neutron/latest/?_ga=2.76704341.1121178038.1592935435-61574442.1585243550

OpenStack.org. (12 de 05 de 2020). *Welcome to Swift's documentation!* Recuperado el 23 de 06 de 2020, de https://docs.openstack.org/swift/latest/?_ga=2.106161475.1121178038.1592935435-61574442.1585243550

OpenStack.org. (s.f.). *Software*. Recuperado el 23 de 06 de 2020, de <https://www.openstack.org/software/stein/>

Technology, T. (04 de 09 de 2017). *Los tres modelos fundamentales en el campo de servicios en la nube*. Recuperado el 08 de 04 de 2019, de <https://www.tuyu.es/modelos-servicios-en-la-nube/>

Webinars, O. (02 de 10 de 2013). *Introducción a la Virtualización*. Recuperado el 08 de 04 de 2019, de <https://openwebinars.net/blog/introduccion-la-virtualizacion/>

Webinars, O. (29 de 04 de 2018). *Software de código abierto para la creación de nubes privadas y públicas*. Recuperado el 08 de 04 de 2019, de <https://www.openstack.org/>

12 Anexos

12.1 Anexo 1: Interfaces de OpenStack

Interfaz de Horizon

Figura 34: En esta interfaz tanto el usuario final como el administrador hacen uso de ella para ingresar a la nube privada usando sus credenciales.



Figura 34: Interfaz de Inicio.

Interfaz de creación de red y subred.

Figura 35: El usuario decidirá el nombre de la red y si desea crear la subred.

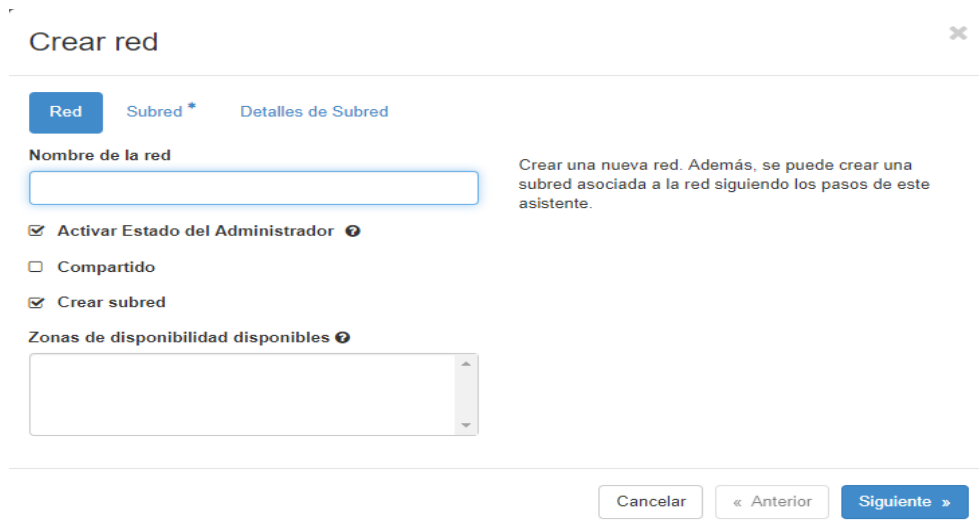


Figura 35: Creación de la red privada.

Figura 36: Al marcar el usuario la opción de crear subred se habilitará la interfaz para ingresar los datos de la misma.

Crear red

Red **Subred *** Detalles de Subred

Nombre de subred

Direcciones de red *

Versión de IP

IPv4

IP de la puerta de enlace

Deshabilitar puerta de enlace

Crea una subred asociada a la red. Es necesario añadir una "dirección de red" y una "IP de la puerta de enlace" válidos. Si no añade una "IP de la puerta de enlace", el primer valor de la red se asignará por defecto. Si no quiere puerta de enlace, seleccione "Deshabilitar puerta de enlace". La configuración avanzada está disponible haciendo click en la pestaña "Detalles de subred".

Cancelar « Anterior **Siguiente >>**

Figura 36: Interfaz de subred.

Figura 37: En la siguiente interfaz se ingresan los últimos detalles de la subred.

Crear red

Red Subred * **Detalles de Subred**

Habilitar DHCP

Especificar atributos adicionales para la subred.

Pools de asignación

Servidores DNS

Rutas de host

Cancelar « Anterior **Crear**

Figura 37: Detalles para la subred.

Interfaz de creación del router.

Figura 38: El usuario ingresará los detalles para la creación del router y seleccionar la red, la cual estará conectada a dicho router.

Crear router

Nombre del router

Descripción:

Crea un enrutador con parámetros especificados.

Habilitar SNAT sólo tendrá efecto si se ha establecido una red externa.

Activar Estado del Administrador

Red externa

Seleccionar red

Habilitar SNAT

Cancelar Crear router

Figura 38: Interfaz para crear un nuevo router.

Interfaz de creación de grupos de seguridad.

Figura 39: Se debe de agregar el nombre y la descripción del grupo.

Crear grupo de seguridad

Nombre *

Descripción:

Los grupos de seguridad son conjuntos de reglas de filtrado que se aplican en las interfaces de red de una MV. Después de crear un grupo de seguridad se pueden añadir las reglas.

Crear grupo de seguridad

Figura 39: Interfaz para crear un nuevo grupo de seguridad.

Interfaz para agregar reglas al grupo de seguridad.

Figura 40: Se debe de seleccionar la regla, la dirección y el puerto como campos obligatorios para crear la regla de seguridad.

Agregar regla ✕

Regla *

Descripción ⓘ

Dirección

Puerto abierto *

Puerto* ⓘ

Remoto* ⓘ

CIDR ⓘ

Descripción:
Las reglas definen el tráfico permitido a las instancias asociadas al grupo de seguridad. Una regla de un grupo de seguridad contiene tres partes principales:
Regla: Puede especificar una plantilla de reglas deseada o usar reglas TCP, UDP e ICMP personalizadas.
Puerto abierto/Rango de puertos Para las reglas de TCP y UDP puede optar por abrir un solo puerto o un rango de ellos. La opción "Rango de puertos" le proporcionará el espacio para especificar tanto el puerto de comienzo como de final del rango. Para las reglas de ICMP por el contrario debe especificar el tipo y código ICMP en los espacios proporcionados.
Remoto: Debe especificar el origen del tráfico a permitir a través de esta regla. Lo puede hacer bien con el formato de un bloque de direcciones IP (CIDR) o especificando un grupo de origen (Grupo de Seguridad). Al seleccionar un grupo de seguridad como origen, se permitirá que cualquier instancia de ese grupo de seguridad pueda acceder a cualquier otra instancia a través de esta regla.

Figura 40: Crear una nueva regla de seguridad.

Interfaz para crear sabor.

Figura 41: Se deben de ingresar los detalles para la creación de un nuevo sabor que se adapte a los recursos necesarios para una nueva imagen sin desperdiciar recursos.

Crear Sabor ✕

Información del sabor * Acceso al sabor

Nombre *

ID ⓘ

VCPU *

RAM (MB) *

Disco raíz (GB) *

Disco efimero (GB)

Disco de intercambio (MB)

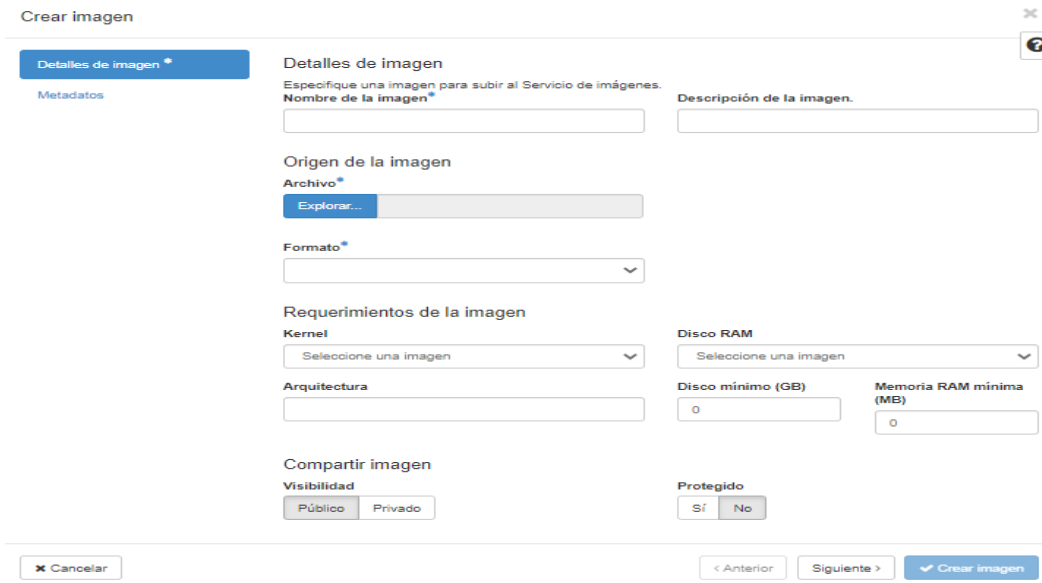
Factor RX/TX

Los sabores definen los tamaños de memoria RAM, disco, número de cores y otros recursos que pueden ser seleccionados por los usuarios al desplegar instancias.

. Figura 43: Interfaz para crear un nuevo sabor.

Interfaz para subir imagen.

Figura 42: Se ingresan los datos necesarios y se selecciona el origen y formato.



. Figura 42: Interfaz para crear una nueva imagen.

Interfaz para ejecutar una instancia.

Figura 43: El usuario debe de completar y seleccionar los campos necesarios, teniendo en cuenta la creación de algunos elementos antes de disponer a la creación de la instancia.



Figura 43: Interfaz para ejecutar/lanzar una instancia.

Interfaz para crear par de claves.

Figura 44: Se debe de proporcionar el nombre para el par de claves, seguido de esto se descargará automáticamente la llave privada.

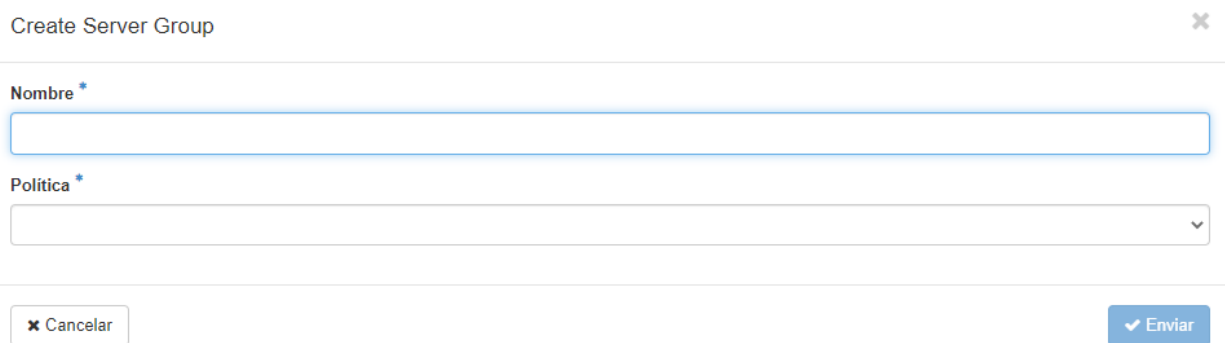


The screenshot shows a modal window titled "Crear Par de Claves" with a close button (X) in the top right corner. Below the title bar, there is a text input field labeled "Nombre de Par de Claves*" with a help icon (question mark) to its right. Below this is a dropdown menu labeled "Key Type*". At the bottom of the modal, there are two buttons: "Cancelar" with a close icon (X) and "Crear Par de Claves" with a plus icon (+).

Figura 44: Interfaz para crear par de claves.

Interfaz para crear un grupo de servidores.

Figura 45: Se ingresará el nombre y se seleccionará la política que tendrá el grupo de servidores.



The screenshot shows a modal window titled "Create Server Group" with a close button (X) in the top right corner. Below the title bar, there is a text input field labeled "Nombre*" and a dropdown menu labeled "Politica*". At the bottom of the modal, there are two buttons: "Cancelar" with a close icon (X) and "Enviar" with a checkmark icon (✓).

Figura 45: Crear grupo de servidores.

Interfaz para la creación de usuarios.

Figura 46: Se ingresan los datos necesarios para la creación de un nuevo usuario, así como su contraseña.

Crear usuario ✕

ID de dominio

Nombre de dominio

Usuario *

Descripción

Correo electrónico

Contraseña *

Confirme la contraseña *

Proyecto principal

Rol

Habilitado

Figura 46: Interfaz para crear usuarios.

Interfaz para la creación de proyectos.

Figura 47: Se proporcionará el nombre del proyecto, así como los usuarios miembros del mismo.

Crear proyecto

Información del proyecto * Miembros del proyecto Grupos de proyecto

ID de dominio default

Nombre de dominio Default

Nombre *

Descripción

Habilitado

Cancelar Crear proyecto

Figura 47: Crear proyecto.

Interfaz para asignar una IP flotante.

Figura 48: Se deberá de seleccionar el pool de direcciones de la red externa, donde se reservará la IP y de manera opcional proporcionar una descripción.

Asignar IP flotante

Pool * Red-Externa

Descripción

DNS Domain

Nombre DNS

Descripción: Asignar una IP flotante desde un pool de IPs flotantes.

Cuotas del proyecto

IP flotante 0 de 50 Usada

Cancelar Asignar IP

Figura 48: Interfaz para asignar IP flotante.

12.2 Anexo 2: Archivo de respuesta

En este archivo editamos los parámetros necesarios para realizar la instalación de acuerdo a los nodos donde se van a instalar los componentes, los elementos que se decidan instalar y las contraseñas que deseamos asignar, si no dejamos las que vienen por defecto. (Ver figura 49)

```
[general]
# Path to a public key to install on servers. If a usable key has not
# been installed on the remote servers, the user is prompted for a
# password and this key is installed so the password will not be
# required again.
CONFIG_SSH_KEY=/root/.ssh/id_rsa.pub

# Default password to be used everywhere (overridden by passwords set
# for individual services or users).
CONFIG_DEFAULT_PASSWORD=

# The amount of service workers/threads to use for each service.
# Useful to tweak when you have memory constraints. Defaults to the
# amount of cores on the system.
CONFIG_SERVICE_WORKERS=%{::processorcount}

# Specify 'y' to install MariaDB. ['y', 'n']
CONFIG_MARIADB_INSTALL=y

# Specify 'y' to install OpenStack Image Service (glance). ['y', 'n']
CONFIG_GLANCE_INSTALL=y

# Specify 'y' to install OpenStack Block Storage (cinder). ['y', 'n']
CONFIG_CINDER_INSTALL=y

# Specify 'y' to install OpenStack Shared File System (manila). ['y',
# 'n']
CONFIG_MANILA_INSTALL=n

# Specify 'y' to install OpenStack Compute (nova). ['y', 'n']
CONFIG_NOVA_INSTALL=y

# Specify 'y' to install OpenStack Networking (neutron) ['y']
CONFIG_NEUTRON_INSTALL=y

# Specify 'y' to install OpenStack Dashboard (horizon). ['y', 'n']
CONFIG_HORIZON_INSTALL=y

# Specify 'y' to install OpenStack Object Storage (swift). ['y', 'n']
CONFIG_SWIFT_INSTALL=y

# Specify 'y' to install OpenStack Metering (ceilometer). Note this
# will also automatically install gnocchi service and configures it as
# the metrics backend. ['y', 'n']
CONFIG_CEILOMETER_INSTALL=y
```

```

# Specify 'y' to install OpenStack Telemetry Alarming (Aodh). Note
# Aodh requires Ceilometer to be installed as well. ['y', 'n']
CONFIG_AODH_INSTALL=y

# Specify 'y' to install OpenStack Events Service (panko). ['y', 'n']
CONFIG_PANKO_INSTALL=n

# Specify 'y' to install OpenStack Data Processing (sahara). In case
# of sahara installation packstack also installs heat. ['y', 'n']
CONFIG_SAHARA_INSTALL=n

# Specify 'y' to install OpenStack Orchestration (heat). ['y', 'n']
CONFIG_HEAT_INSTALL=y

# Specify 'y' to install OpenStack Container Infrastructure
# Management Service (magnum). ['y', 'n']
CONFIG_MAGNUM_INSTALL=n

# Specify 'y' to install OpenStack Database (trove) ['y', 'n']
CONFIG_TROVE_INSTALL=n

# Specify 'y' to install OpenStack Bare Metal Provisioning (ironic).
# ['y', 'n']
CONFIG_IRONIC_INSTALL=n

# Specify 'y' to install the OpenStack Client packages (command-line
# tools). An admin "rc" file will also be installed. ['y', 'n']
CONFIG_CLIENT_INSTALL=y

# Comma-separated list of NTP servers. Leave plain if Packstack
# should not install ntpd on instances.
CONFIG_NTP_SERVERS=

# Comma-separated list of servers to be excluded from the
# installation. This is helpful if you are running Packstack a second
# time with the same answer file and do not want Packstack to
# overwrite these server's configurations. Leave empty if you do not
# need to exclude any servers.
EXCLUDE_SERVERS=0.pool.ntp.org,1.pool.ntp.org,2.pool.ntp.org

# Specify 'y' if you want to run OpenStack services in debug mode;
# otherwise, specify 'n'. ['y', 'n']
CONFIG_DEBUG_MODE=n

# Server on which to install OpenStack services specific to the
# controller role (for example, API servers or dashboard).
CONFIG_CONTROLLER_HOST=192.168.1.120

# List the servers on which to install the Compute service.
CONFIG_COMPUTE_HOSTS=192.168.1.121

# List of servers on which to install the network service such as
# Compute networking (nova network) or OpenStack Networking (neutron).
CONFIG_NETWORK_HOSTS=192.168.1.122

```

```

# Specify 'y' if you want to use VMware vCenter as hypervisor and
# storage; otherwise, specify 'n'. ['y', 'n']
CONFIG_VMWARE_BACKEND=n

# Specify 'y' if you want to use unsupported parameters. This should
# be used only if you know what you are doing. Issues caused by using
# unsupported options will not be fixed before the next major release.
# ['y', 'n']
CONFIG_UNSUPPORTED=n

# Specify 'y' if you want to use subnet addresses (in CIDR format)
# instead of interface names in following options:
# CONFIG_NEUTRON_OVS_BRIDGE_IFACES,
# CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS, CONFIG_NEUTRON_OVS_TUNNEL_IF.
# This is useful for cases when interface names are not same on all
# installation hosts.
CONFIG_USE_SUBNETS=n

# IP address of the VMware vCenter server.
CONFIG_VCENTER_HOST=

# User name for VMware vCenter server authentication.
CONFIG_VCENTER_USER=

# Password for VMware vCenter server authentication.
CONFIG_VCENTER_PASSWORD=

# Comma separated list of names of the VMware vCenter clusters. Note:
# if multiple clusters are specified each one is mapped to one
# compute, otherwise all computes are mapped to same cluster.
CONFIG_VCENTER_CLUSTER_NAMES=

# (Unsupported!) Server on which to install OpenStack services
# specific to storage servers such as Image or Block Storage services.
CONFIG_STORAGE_HOST=192.168.1.120

# (Unsupported!) Server on which to install OpenStack services
# specific to OpenStack Data Processing (sahara).
CONFIG_SAHARA_HOST=192.168.1.120

# Comma-separated list of URLs for any additional yum repositories,
# to use for installation.
CONFIG_REPO=

# Specify 'y' to enable the RDO testing repository. ['y', 'n']
CONFIG_ENABLE_RDO_TESTING=n

# To subscribe each server with Red Hat Subscription Manager, include
# this with CONFIG_RH_PW.
CONFIG_RH_USER=

# To subscribe each server to receive updates from a Satellite
# server, provide the URL of the Satellite server. You must also
# provide a user name (CONFIG_SATELLITE_USERNAME) and password

```

```
# (CONFIG_SATELLITE_PASSWORD) or an access key (CONFIG_SATELLITE_AKEY)
# for authentication.
CONFIG_SATELLITE_URL=

# Specify a Satellite 6 Server to register to. If not specified,
# Packstack will register the system to the Red Hat server. When this
# option is specified, you also need to set the Satellite 6
# organization (CONFIG_RH_SAT6_ORG) and an activation key
# (CONFIG_RH_SAT6_KEY).
CONFIG_RH_SAT6_SERVER=

# To subscribe each server with Red Hat Subscription Manager, include
# this with CONFIG_RH_USER.
CONFIG_RH_PW=

# Specify 'y' to enable RHEL optional repositories. ['y', 'n']
CONFIG_RH_OPTIONAL=y

# HTTP proxy to use with Red Hat Subscription Manager.
CONFIG_RH_PROXY=

# Specify a Satellite 6 Server organization to use when registering
# the system.
CONFIG_RH_SAT6_ORG=

# Specify a Satellite 6 Server activation key to use when registering
# the system.
CONFIG_RH_SAT6_KEY=

# Port to use for Red Hat Subscription Manager's HTTP proxy.
CONFIG_RH_PROXY_PORT=

# User name to use for Red Hat Subscription Manager's HTTP proxy.
CONFIG_RH_PROXY_USER=

# Password to use for Red Hat Subscription Manager's HTTP proxy.
CONFIG_RH_PROXY_PW=

# User name to authenticate with the RHN Satellite server; if you
# intend to use an access key for Satellite authentication, leave this
# blank.
CONFIG_SATELLITE_USER=

# Password to authenticate with the RHN Satellite server; if you
# intend to use an access key for Satellite authentication, leave this
# blank.
CONFIG_SATELLITE_PW=

# Access key for the Satellite server; if you intend to use a user
# name and password for Satellite authentication, leave this blank.
CONFIG_SATELLITE_AKEY=

# Certificate path or URL of the certificate authority to verify that
# the connection with the Satellite server is secure. If you are not
# using Satellite in your deployment, leave this blank.
```

```

CONFIG_SATELLITE_CACERT=

# Profile name that should be used as an identifier for the system in
# RHN Satellite (if required).
CONFIG_SATELLITE_PROFILE=

# Comma-separated list of flags passed to the rhnreg_ks command.
# Valid flags are: novirtinfo, norhnsd, nopackages ['novirtinfo',
# 'norhnsd', 'nopackages']
CONFIG_SATELLITE_FLAGS=

# HTTP proxy to use when connecting to the RHN Satellite server (if
# required).
CONFIG_SATELLITE_PROXY=

# User name to authenticate with the Satellite-server HTTP proxy.
CONFIG_SATELLITE_PROXY_USER=

# User password to authenticate with the Satellite-server HTTP proxy.
CONFIG_SATELLITE_PROXY_PW=

# Specify filepath for CA cert file. If CONFIG_SSL_CACERT_SELFSIGN is
# set to 'n' it has to be preexisting file.
CONFIG_SSL_CACERT_FILE=/etc/pki/tls/certs/selfcert.crt

# Specify filepath for CA cert key file. If
# CONFIG_SSL_CACERT_SELFSIGN is set to 'n' it has to be preexisting
# file.
CONFIG_SSL_CACERT_KEY_FILE=/etc/pki/tls/private/selfkey.key

# Enter the path to use to store generated SSL certificates in.
CONFIG_SSL_CERT_DIR=~/.packstackca/

# Specify 'y' if you want Packstack to pregenerate the CA
# Certificate.
CONFIG_SSL_CACERT_SELFSIGN=y

# Enter the ssl certificates subject country.
CONFIG_SSL_CERT_SUBJECT_C=--

# Enter the ssl certificates subject state.
CONFIG_SSL_CERT_SUBJECT_ST=State

# Enter the ssl certificates subject location.
CONFIG_SSL_CERT_SUBJECT_L=City

# Enter the ssl certificates subject organization.
CONFIG_SSL_CERT_SUBJECT_O=openstack

# Enter the ssl certificates subject organizational unit.
CONFIG_SSL_CERT_SUBJECT_OU=packstack

# Enter the ssl certificates subject common name.
CONFIG_SSL_CERT_SUBJECT_CN=controlador

```

```
CONFIG_SSL_CERT_SUBJECT_MAIL=admin@controlador

# Service to be used as the AMQP broker. Allowed values are: rabbitmq
# ['rabbitmq']
CONFIG_AMQP_BACKEND=rabbitmq

# IP address of the server on which to install the AMQP service.
CONFIG_AMQP_HOST=192.168.1.120

# Specify 'y' to enable SSL for the AMQP service. ['y', 'n']
CONFIG_AMQP_ENABLE_SSL=n

# Specify 'y' to enable authentication for the AMQP service. ['y',
# 'n']
CONFIG_AMQP_ENABLE_AUTH=n

# Password for the NSS certificate database of the AMQP service.
CONFIG_AMQP_NSS_CERTDB_PW=PW_PLACEHOLDER

# User for AMQP authentication.
CONFIG_AMQP_AUTH_USER=amqp_user

# Password for AMQP authentication.
CONFIG_AMQP_AUTH_PASSWORD=PW_PLACEHOLDER

# IP address of the server on which to install MariaDB. If a MariaDB
# installation was not specified in CONFIG_MARIADB_INSTALL, specify
# the IP address of an existing database server (a MariaDB cluster can
# also be specified).
CONFIG_MARIADB_HOST=192.168.1.120

# User name for the MariaDB administrative user.
CONFIG_MARIADB_USER=root

# Password for the MariaDB administrative user.
CONFIG_MARIADB_PW=a8777db008c94e23

# Password to use for the Identity service (keystone) to access the
# database.
CONFIG_KEYSTONE_DB_PW=5fc01326092e4c02

# Enter y if cron job to rotate Fernet tokens should be created.
CONFIG_KEYSTONE_FERNET_TOKEN_ROTATE_ENABLE=True

# Default region name to use when creating tenants in the Identity
# service.
CONFIG_KEYSTONE_REGION=RegionOne

# Token to use for the Identity service API.
CONFIG_KEYSTONE_ADMIN_TOKEN=1d59203ea7ed455c932334abd77ec01b

# Email address for the Identity service 'admin' user. Defaults to
CONFIG_KEYSTONE_ADMIN_EMAIL=root@localhost

# User name for the Identity service 'admin' user. Defaults to
```

```
# 'admin'.
CONFIG_KEYSTONE_ADMIN_USERNAME=admin

# Password to use for the Identity service 'admin' user.
CONFIG_KEYSTONE_ADMIN_PW=bfaf40877d504424

# Password to use for the Identity service 'demo' user.
CONFIG_KEYSTONE_DEMO_PW=158073c856cb4810

# Identity service API version string. ['v2.0', 'v3']
CONFIG_KEYSTONE_API_VERSION=v3

# Identity service token format (FERNET). Since Rocky, only FERNET is
# supported. ['FERNET']
CONFIG_KEYSTONE_TOKEN_FORMAT=FERNET

# Type of Identity service backend (sql or ldap). ['sql', 'ldap']
CONFIG_KEYSTONE_IDENTITY_BACKEND=sql

# URL for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_URL=ldap://192.168.1.120

# User DN for the Identity service LDAP backend. Used to bind to the
# LDAP server if the LDAP server does not allow anonymous
# authentication.
CONFIG_KEYSTONE_LDAP_USER_DN=

# User DN password for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_PASSWORD=

# Base suffix for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_SUFFIX=

# Query scope for the Identity service LDAP backend. Use 'one' for
# onelevel/singleLevel or 'sub' for subtree/wholeSubtree ('base' is
# not actually used by the Identity service and is therefore
# deprecated). ['base', 'one', 'sub']
CONFIG_KEYSTONE_LDAP_QUERY_SCOPE=one

# Query page size for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_PAGE_SIZE=-1

# User subtree for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_SUBTREE=

# User query filter for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_FILTER=

# User object class for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_OBJECTCLASS=

# User ID attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_ID_ATTRIBUTE=

# User name attribute for the Identity service LDAP backend.
```

```

CONFIG_KEYSTONE_LDAP_USER_NAME_ATTRIBUTE=

# User email address attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_MAIL_ATTRIBUTE=

# User-enabled attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE=

# Bit mask integer applied to user-enabled attribute for the Identity
# service LDAP backend. Indicate the bit that the enabled value is
# stored in if the LDAP server represents "enabled" as a bit on an
# integer rather than a boolean. A value of "0" indicates the mask is
# not used (default). If this is not set to "0", the typical value is
# "2", typically used when
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE = userAccountControl".
CONFIG_KEYSTONE_LDAP_USER_ENABLED_MASK=-1

# Value of enabled attribute which indicates user is enabled for the
# Identity service LDAP backend. This should match an appropriate
# integer value if the LDAP server uses non-boolean (bitmask) values
# to indicate whether a user is enabled or disabled. If this is not
# set as 'y', the typical value is "512". This is typically used when
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_ATTRIBUTE = userAccountControl".
CONFIG_KEYSTONE_LDAP_USER_ENABLED_DEFAULT=TRUE

# Specify 'y' if users are disabled (not enabled) in the Identity
# service LDAP backend (inverts boolean-enabled values). Some LDAP
# servers use a boolean lock attribute where "y" means an account is
# disabled. Setting this to 'y' allows these lock attributes to be
# used. This setting will have no effect if
# "CONFIG_KEYSTONE_LDAP_USER_ENABLED_MASK" is in use. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ENABLED_INVERT=n

# Comma-separated list of attributes stripped from LDAP user entry
# upon update.
CONFIG_KEYSTONE_LDAP_USER_ATTRIBUTE_IGNORE=

# Identity service LDAP attribute mapped to default_project_id for
# users.
CONFIG_KEYSTONE_LDAP_USER_DEFAULT_PROJECT_ID_ATTRIBUTE=

# Specify 'y' if you want to be able to create Identity service users
# through the Identity service interface; specify 'n' if you will
# create directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_CREATE=n

# Specify 'y' if you want to be able to update Identity service users
# through the Identity service interface; specify 'n' if you will
# update directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_UPDATE=n

# Specify 'y' if you want to be able to delete Identity service users
# through the Identity service interface; specify 'n' if you will
# delete directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_USER_ALLOW_DELETE=n

```



```

# Identity service LDAP attribute mapped to password.
CONFIG_KEYSTONE_LDAP_USER_PASS_ATTRIBUTE=

# DN of the group entry to hold enabled LDAP users when using enabled
# emulation.
CONFIG_KEYSTONE_LDAP_USER_ENABLED_EMULATION_DN=

# List of additional LDAP attributes for mapping additional attribute
# mappings for users. The attribute-mapping format is
# <ldap_attr>:<user_attr>, where ldap_attr is the attribute in the
# LDAP entry and user_attr is the Identity API attribute.
CONFIG_KEYSTONE_LDAP_USER_ADDITIONAL_ATTRIBUTE_MAPPING=

# Group subtree for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_SUBTREE=

# Group query filter for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_FILTER=

# Group object class for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_OBJECTCLASS=

# Group ID attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_ID_ATTRIBUTE=

# Group name attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_NAME_ATTRIBUTE=

# Group member attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_MEMBER_ATTRIBUTE=

# Group description attribute for the Identity service LDAP backend.
CONFIG_KEYSTONE_LDAP_GROUP_DESC_ATTRIBUTE=

# Comma-separated list of attributes stripped from LDAP group entry
# upon update.
CONFIG_KEYSTONE_LDAP_GROUP_ATTRIBUTE_IGNORE=

# Specify 'y' if you want to be able to create Identity service
# groups through the Identity service interface; specify 'n' if you
# will create directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_CREATE=n

# Specify 'y' if you want to be able to update Identity service
# groups through the Identity service interface; specify 'n' if you
# will update directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_UPDATE=n

# Specify 'y' if you want to be able to delete Identity service
# groups through the Identity service interface; specify 'n' if you
# will delete directly in the LDAP backend. ['n', 'y']
CONFIG_KEYSTONE_LDAP_GROUP_ALLOW_DELETE=n

# List of additional LDAP attributes used for mapping additional

```

```

# attribute mappings for groups. The attribute=mapping format is
# <ldap_attr>:<group_attr>, where ldap_attr is the attribute in the
# LDAP entry and group_attr is the Identity API attribute.
CONFIG_KEYSTONE_LDAP_GROUP_ADDITIONAL_ATTRIBUTE_MAPPING=

# Specify 'y' if the Identity service LDAP backend should use TLS.
# ['n', 'y']
CONFIG_KEYSTONE_LDAP_USE_TLS=n

# CA certificate directory for Identity service LDAP backend (if TLS
# is used).
CONFIG_KEYSTONE_LDAP_TLS_CACERTDIR=

# CA certificate file for Identity service LDAP backend (if TLS is
# used).
CONFIG_KEYSTONE_LDAP_TLS_CACERTFILE=

# Certificate-checking strictness level for Identity service LDAP
# backend; valid options are: never, allow, demand. ['never', 'allow',
# 'demand']
CONFIG_KEYSTONE_LDAP_TLS_REQ_CERT=demand

# Password to use for the Image service (glance) to access the
# database.
CONFIG_GLANCE_DB_PW=36c76c4d682643d6

# Password to use for the Image service to authenticate with the
# Identity service.
CONFIG_GLANCE_KS_PW=bcaf0daccd024224

# Storage backend for the Image service (controls how the Image
# service stores disk images). Valid options are: file or swift
# (Object Storage). The Object Storage service must be enabled to use
# it as a working backend; otherwise, Packstack falls back to 'file'.
# ['file', 'swift']
CONFIG_GLANCE_BACKEND=file

# Password to use for the Block Storage service (cinder) to access
# the database.
CONFIG_CINDER_DB_PW=40df91640def4f69

# Enter y if cron job for removing soft deleted DB rows should be
# created.
CONFIG_CINDER_DB_PURGE_ENABLE=True

# Password to use for the Block Storage service to authenticate with
# the Identity service.
CONFIG_CINDER_KS_PW=28e0ab6305e84e72

# Storage backend to use for the Block Storage service; valid options
# are: lvm, gluster, nfs, vmdk, netapp, solidfire. ['lvm', 'gluster',
# 'nfs', 'vmdk', 'netapp', 'solidfire']
CONFIG_CINDER_BACKEND=lvm

# Specify 'y' to create the Block Storage volumes group. That is,

```

```

# Packstack creates a raw disk image in /var/lib/cinder, and mounts it
# using a loopback device. This should only be used for testing on a
# proof-of-concept installation of the Block Storage service (a file-
# backed volume group is not suitable for production usage). ['y',
# 'n']
CONFIG_CINDER_VOLUMES_CREATE=y

# Specify a custom name for the lvm cinder volume group
CONFIG_CINDER_VOLUME_NAME=cinder-volumes

# Size of Block Storage volumes group. Actual volume size will be
# extended with 3% more space for VG metadata. Remember that the size
# of the volume group will restrict the amount of disk space that you
# can expose to Compute instances, and that the specified amount must
# be available on the device used for /var/lib/cinder.
CONFIG_CINDER_VOLUMES_SIZE=60G

# A single or comma-separated list of Red Hat Storage (gluster)
# volume shares to mount. Example: 'ip-address:/vol-name', 'domain
# :/vol-name'
CONFIG_CINDER_GLUSTER_MOUNTS=

# A single or comma-separated list of NFS exports to mount. Example:
# 'ip-address:/export-name'
CONFIG_CINDER_NFS_MOUNTS=

# Administrative user account name used to access the NetApp storage
# system or proxy server.
CONFIG_CINDER_NETAPP_LOGIN=

# Password for the NetApp administrative user account specified in
# the CONFIG_CINDER_NETAPP_LOGIN parameter.
CONFIG_CINDER_NETAPP_PASSWORD=

# Hostname (or IP address) for the NetApp storage system or proxy
# server.
CONFIG_CINDER_NETAPP_HOSTNAME=

# The TCP port to use for communication with the storage system or
# proxy. If not specified, Data ONTAP drivers will use 80 for HTTP and
# 443 for HTTPS; E-Series will use 8080 for HTTP and 8443 for HTTPS.
# Defaults to 80.
CONFIG_CINDER_NETAPP_SERVER_PORT=80

# Storage family type used on the NetApp storage system; valid
# options are ontap_7mode for using Data ONTAP operating in 7-Mode,
# ontap_cluster for using clustered Data ONTAP, or E-Series for NetApp
# E-Series. Defaults to ontap_cluster. ['ontap_7mode',
# 'ontap_cluster', 'eseries']
CONFIG_CINDER_NETAPP_STORAGE_FAMILY=ontap_cluster

# The transport protocol used when communicating with the NetApp
# storage system or proxy server. Valid values are http or https.
# Defaults to 'http'. ['http', 'https']
CONFIG_CINDER_NETAPP_TRANSPORT_TYPE=http

```

```

# Storage protocol to be used on the data path with the NetApp
# storage system; valid options are iscsi, fc, nfs. Defaults to nfs.
# ['iscsi', 'fc', 'nfs']
CONFIG_CINDER_NETAPP_STORAGE_PROTOCOL=nfs

# Quantity to be multiplied by the requested volume size to ensure
# enough space is available on the virtual storage server (Vserver) to
# fulfill the volume creation request. Defaults to 1.0.
CONFIG_CINDER_NETAPP_SIZE_MULTIPLIER=1.0

# Time period (in minutes) that is allowed to elapse after the image
# is last accessed, before it is deleted from the NFS image cache.
# When a cache-cleaning cycle begins, images in the cache that have
# not been accessed in the last M minutes, where M is the value of
# this parameter, are deleted from the cache to create free space on
# the NFS share. Defaults to 720.
CONFIG_CINDER_NETAPP_EXPIRY_THRES_MINUTES=720

# If the percentage of available space for an NFS share has dropped
# below the value specified by this parameter, the NFS image cache is
# cleaned. Defaults to 20.
CONFIG_CINDER_NETAPP_THRES_AVL_SIZE_PERC_START=20

# When the percentage of available space on an NFS share has reached
# the percentage specified by this parameter, the driver stops
# clearing files from the NFS image cache that have not been accessed
# in the last M minutes, where M is the value of the
# CONFIG_CINDER_NETAPP_EXPIRY_THRES_MINUTES parameter. Defaults to 60.
CONFIG_CINDER_NETAPP_THRES_AVL_SIZE_PERC_STOP=60

# Single or comma-separated list of NetApp NFS shares for Block
# Storage to use. Format: ip-address:/export-name. Defaults to ''.
CONFIG_CINDER_NETAPP_NFS_SHARES=

# File with the list of available NFS shares. Defaults to
# '/etc/cinder/shares.conf'.
CONFIG_CINDER_NETAPP_NFS_SHARES_CONFIG=/etc/cinder/shares.conf

# This parameter is only utilized when the storage protocol is
# configured to use iSCSI or FC. This parameter is used to restrict
# provisioning to the specified controller volumes. Specify the value
# of this parameter to be a comma separated list of NetApp controller
# volume names to be used for provisioning. Defaults to ''.
CONFIG_CINDER_NETAPP_VOLUME_LIST=

# The vFiler unit on which provisioning of block storage volumes will
# be done. This parameter is only used by the driver when connecting
# to an instance with a storage family of Data ONTAP operating in
# 7-Mode Only use this parameter when utilizing the MultiStore feature
# on the NetApp storage system. Defaults to ''.
CONFIG_CINDER_NETAPP_VFILER=

# The name of the config.conf stanza for a Data ONTAP (7-mode) HA
# partner. This option is only used by the driver when connecting to

```

```

# an instance with a storage family of Data ONTAP operating in 7-Mode,
# and it is required if the storage protocol selected is FC. Defaults
# to ''.
CONFIG_CINDER_NETAPP_PARTNER_BACKEND_NAME=

# This option specifies the virtual storage server (Vserver) name on
# the storage cluster on which provisioning of block storage volumes
# should occur. Defaults to ''.
CONFIG_CINDER_NETAPP_VSERVER=

# Restricts provisioning to the specified controllers. Value must be
# a comma-separated list of controller hostnames or IP addresses to be
# used for provisioning. This option is only utilized when the storage
# family is configured to use E-Series. Defaults to ''.
CONFIG_CINDER_NETAPP_CONTROLLER_IPS=

# Password for the NetApp E-Series storage array. Defaults to ''.
CONFIG_CINDER_NETAPP_SA_PASSWORD=

# This option is used to define how the controllers in the E-Series
# storage array will work with the particular operating system on the
# hosts that are connected to it. Defaults to 'linux_dm_mp'
CONFIG_CINDER_NETAPP_ESERIES_HOST_TYPE=linux_dm_mp

# Path to the NetApp E-Series proxy application on a proxy server.
# The value is combined with the value of the
# CONFIG_CINDER_NETAPP_TRANSPORT_TYPE, CONFIG_CINDER_NETAPP_HOSTNAME,
# and CONFIG_CINDER_NETAPP_HOSTNAME options to create the URL used by
# the driver to connect to the proxy application. Defaults to
# '/devmgr/v2'.
CONFIG_CINDER_NETAPP_WEBSERVICE_PATH=/devmgr/v2

# Restricts provisioning to the specified storage pools. Only dynamic
# disk pools are currently supported. The value must be a comma-
# separated list of disk pool names to be used for provisioning.
# Defaults to ''.
CONFIG_CINDER_NETAPP_STORAGE_POOLS=

# Cluster admin account name used to access the SolidFire storage
# system.
CONFIG_CINDER_SOLIDFIRE_LOGIN=

# Password for the SolidFire cluster admin user account specified in
# the CONFIG_CINDER_SOLIDFIRE_LOGIN parameter.
CONFIG_CINDER_SOLIDFIRE_PASSWORD=

# Hostname (or IP address) for the SolidFire storage system's MVIP.
CONFIG_CINDER_SOLIDFIRE_HOSTNAME=

# Password to use for OpenStack Bare Metal Provisioning (ironic) to
# access the database.
CONFIG_IRONIC_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Bare Metal Provisioning to
# authenticate with the Identity service.

```

```

CONFIG_IRONIC_KS_PW=PW_PLACEHOLDER

# Enter y if cron job for removing soft deleted DB rows should be
# created.
CONFIG_NOVA_DB_PURGE_ENABLE=True

# Password to use for the Compute service (nova) to access the
# database.
CONFIG_NOVA_DB_PW=32f0aae2669b40a2

# Password to use for the Compute service to authenticate with the
# Identity service.
CONFIG_NOVA_KS_PW=c8eafe21ea334b81

# Whether or not Packstack should manage a default initial set of
# Nova flavors. Defaults to 'y'.
CONFIG_NOVA_MANAGE_FLAVORS=y

# Overcommitment ratio for virtual to physical CPUs. Specify 1.0 to
# disable CPU overcommitment.
CONFIG_NOVA_SCHED_CPU_ALLOC_RATIO=16.0

# Overcommitment ratio for virtual to physical RAM. Specify 1.0 to
# disable RAM overcommitment.
CONFIG_NOVA_SCHED_RAM_ALLOC_RATIO=1.5

# Protocol used for instance migration. Valid options are: ssh and
# tcp. Note that the tcp protocol is not encrypted, so it is insecure.
# ['ssh', 'tcp']
CONFIG_NOVA_COMPUTE_MIGRATE_PROTOCOL=ssh

# PEM encoded certificate to be used for ssl on the https server,
# leave blank if one should be generated, this certificate should not
# require a passphrase. If CONFIG_HORIZON_SSL is set to 'n' this
# parameter is ignored.
CONFIG_VNC_SSL_CERT=

# SSL keyfile corresponding to the certificate if one was entered. If
# CONFIG_HORIZON_SSL is set to 'n' this parameter is ignored.
CONFIG_VNC_SSL_KEY=

# Enter the PCI passthrough array of hash in JSON style for
# controller eg. [{"vendor_id":"1234", "product_id":"5678",
# "name":"default"}, {...}]
CONFIG_NOVA_PCI_ALIAS=

# Enter the PCI passthrough whitelist array of hash in JSON style for
# controller eg. [{"vendor_id":"1234", "product_id":"5678",
# "name":"default"}, {...}]
CONFIG_NOVA_PCI_PASSTHROUGH_WHITELIST=

# The hypervisor driver to use with Nova. Can be either 'qemu' or
# 'kvm'. Defaults to 'qemu' on virtual machines and 'kvm' on bare
# metal hardware. For nested KVM set it explicitly to 'kvm'.
CONFIG_NOVA_LIBVIRT_VIRT_TYPE=%{:default_hypervisor}

```

```

# Password to use for OpenStack Networking (neutron) to authenticate
# with the Identity service.
CONFIG_NEUTRON_KS_PW=6ab576c22fae4c51

# The password to use for OpenStack Networking to access the
# database.
CONFIG_NEUTRON_DB_PW=4d19fcdc131e434c

# The name of the Open vSwitch bridge (or empty for linuxbridge) for
# the OpenStack Networking L3 agent to use for external traffic.
# Specify 'provider' if you intend to use a provider network to handle
# external traffic.
CONFIG_NEUTRON_L3_EXT_BRIDGE=br-ex

# Password for the OpenStack Networking metadata agent.
CONFIG_NEUTRON_METADATA_PW=833e5f63ac1f40ac

# Specify 'y' to install OpenStack Networking's Load-Balancing-
# as-a-Service (LBaaS). ['y', 'n']
CONFIG_LBAAS_INSTALL=n

# Specify 'y' to install OpenStack Networking's L3 Metering agent
# ['y', 'n']
CONFIG_NEUTRON_METERING_AGENT_INSTALL=y

# Specify 'y' to configure OpenStack Networking's Firewall-
# as-a-Service (FWaaS). ['y', 'n']
CONFIG_NEUTRON_FWAAS=n

# Specify 'y' to configure OpenStack Networking's VPN-as-a-Service
# (VPNaaS). ['y', 'n']
CONFIG_NEUTRON_VPNAAS=n

# Comma-separated list of network-type driver entry points to be
# loaded from the neutron.ml2.type_drivers namespace. ['local',
# 'flat', 'vlan', 'gre', 'vxlan', 'geneve']
CONFIG_NEUTRON_ML2_TYPE_DRIVERS=geneve,flat

# Comma-separated, ordered list of network types to allocate as
# tenant networks. The 'local' value is only useful for single-box
# testing and provides no connectivity between hosts. ['local',
# 'vlan', 'gre', 'vxlan', 'geneve']
CONFIG_NEUTRON_ML2_TENANT_NETWORK_TYPES=geneve

# Comma-separated ordered list of networking mechanism driver entry
# points to be loaded from the neutron.ml2.mechanism_drivers
# namespace. ['logger', 'test', 'linuxbridge', 'openvswitch',
# 'hyperv', 'ncs', 'arista', 'cisco_nexus', 'mlnx', 'l2population',
# 'sriovnicswitch', 'ovn']
CONFIG_NEUTRON_ML2_MECHANISM_DRIVERS=ovn

# Comma-separated list of physical_network names with which flat
# networks can be created. Use * to allow flat networks with arbitrary
# physical_network names.

```

```

CONFIG_NEUTRON_ML2_FLAT_NETWORKS=*

# Comma-separated list of <physical_network>:<vlan_min>:<vlan_max> or
# <physical_network> specifying physical_network names usable for VLAN
# provider and tenant networks, as well as ranges of VLAN tags on each
# available for allocation to tenant networks.
CONFIG_NEUTRON_ML2_VLAN_RANGES=

# Comma-separated list of <tun_min>:<tun_max> tuples enumerating
# ranges of GRE tunnel IDs that are available for tenant-network
# allocation. A tuple must be an array with tun_max +1 - tun_min >
# 1000000.
CONFIG_NEUTRON_ML2_TUNNEL_ID_RANGES=

# Comma-separated list of addresses for VXLAN multicast group. If
# left empty, disables VXLAN from sending allocate broadcast traffic
# (disables multicast VXLAN mode). Should be a Multicast IP (v4 or v6)
# address.
CONFIG_NEUTRON_ML2_VXLAN_GROUP=

# Comma-separated list of <vni_min>:<vni_max> tuples enumerating
# ranges of VXLAN VNI IDs that are available for tenant network
# allocation. Minimum value is 0 and maximum value is 16777215.
CONFIG_NEUTRON_ML2_VNI_RANGES=10:100

# Name of the L2 agent to be used with OpenStack Networking.
# ['linuxbridge', 'openvswitch', 'ovn']
CONFIG_NEUTRON_L2_AGENT=ovn

# Comma-separated list of interface mappings for the OpenStack
# Networking ML2 SRIOV agent. Each tuple in the list must be in the
# format <physical_network>:<net_interface>. Example:
# physnet1:eth1,physnet2:eth2,physnet3:eth3.
CONFIG_NEUTRON_ML2_SRIOV_INTERFACE_MAPPINGS=

# Comma-separated list of interface mappings for the OpenStack
# Networking linuxbridge plugin. Each tuple in the list must be in the
# format <physical_network>:<net_interface>. Example:
# physnet1:eth1,physnet2:eth2,physnet3:eth3.
CONFIG_NEUTRON_LB_INTERFACE_MAPPINGS=

# Comma-separated list of bridge mappings for the OpenStack
# Networking Open vSwitch plugin. Each tuple in the list must be in
# the format <physical_network>:<ovs_bridge>. Example: physnet1:br-
# eth1,physnet2:br-eth2,physnet3:br-eth3
CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=extnet:br-ex

# Comma-separated list of colon-separated Open vSwitch
# <bridge>:<interface> pairs. The interface will be added to the
# associated bridge. If you desire the bridge to be persistent a value
# must be added to this directive, also
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS must be set in order to create
# the proper port. This can be achieved from the command line by
# issuing the following command: packstack --allinone --os-neutron-
# ovs-bridge-mappings=ext-net:br-ex --os-neutron-ovs-bridge-interfaces

```



```

# =br-ex:eth0
CONFIG_NEUTRON_OVS_BRIDGE_IFACES=

# Comma-separated list of Open vSwitch bridges that must be created
# and connected to interfaces in compute nodes when flat or vlan type
# drivers are enabled. These bridges must exist in
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS and
# CONFIG_NEUTRON_OVS_BRIDGE_IFACES. Example: --os-neutron-ovs-bridges-
# compute=br-vlan --os-neutron-ovs-bridge-mappings="extnet:br-
# ex,physnet1:br-vlan" --os-neutron-ovs-bridge-interfaces="br-ex:eth1
# ,br-vlan:eth2"
CONFIG_NEUTRON_OVS_BRIDGES_COMPUTE=

# Name of physical network used for external network when enabling
# CONFIG_PROVISION_DEMO. Name must be one of the included in
# CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS. Example: --os-neutron-ovs-
# bridge-mappings="extnet:br-ex,physnet1:br-vlan" --os-neutron-ovs-
# bridge-interfaces="br-ex:eth1,br-vlan:eth2" --os-neutron-ovs-
# external-physnet="extnet"
CONFIG_NEUTRON_OVS_EXTERNAL_PHYSNET=extnet

# Interface for the Open vSwitch tunnel. Packstack overrides the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface (for example, eth1).
CONFIG_NEUTRON_OVS_TUNNEL_IF=

# Comma-separated list of subnets (for example,
# 192.168.10.0/24,192.168.11.0/24) used for sending tunneling packets.
# This is used to configure IP filtering to accept tunneling packets
# from these subnets instead of specific IP addresses of peer nodes.
# This is useful when you add existing nodes to EXCLUDE_SERVERS
# because, in this case, packstack cannot modify the IP filtering of
# the existing nodes.
CONFIG_NEUTRON_OVS_TUNNEL_SUBNETS=

# VXLAN UDP port.
CONFIG_NEUTRON_OVS_VXLAN_UDP_PORT=4789

# Comma-separated list of bridge mappings for the OpenStack
# Networking Open Virtual Network plugin. Each tuple in the list must
# be in the format <physical_network>:<ovs_bridge>. Example: physnet1
# :br-eth1,physnet2:br-eth2,physnet3:br-eth3
CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS=extnet:br-ex

# Comma-separated list of colon-separated Open vSwitch
# <bridge>:<interface> pairs. The interface will be added to the
# associated bridge. If you desire the bridge to be persistent a value
# must be added to this directive, also
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS must be set in order to create
# the proper port. This can be achieved from the command line by
# issuing the following command: packstack --allinone --os-neutron-
# ovn-bridge-mappings=ext-net:br-ex --os-neutron-ovn-bridge-interfaces
# =br-ex:eth0
CONFIG_NEUTRON_OVN_BRIDGE_IFACES=

```

```

# Comma-separated list of Open vSwitch bridges that must be created
# and connected to interfaces in compute nodes when flat or vlan type
# drivers are enabled. These bridges must exist in
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS and
# CONFIG_NEUTRON_OVN_BRIDGE_IFACES. Example: --os-neutron-ovn-bridges-
# compute=br-vlan --os-neutron-ovn-bridge-mappings="extnet:br-
# ex,physnet1:br-vlan" --os-neutron-ovn-bridge-interfaces="br-ex:eth1
# ,br-vlan:eth2"
CONFIG_NEUTRON_OVN_BRIDGES_COMPUTE=

# Name of physical network used for external network when enabling
# CONFIG_PROVISION_DEMO. Name must be one of the included in
# CONFIG_NEUTRON_OVN_BRIDGE_MAPPINGS. Example: --os-neutron-ovn-
# bridge-mappings="extnet:br-ex,physnet1:br-vlan" --os-neutron-ovn-
# bridge-interfaces="br-ex:eth1,br-vlan:eth2" --os-neutron-ovn-
# external-physnet="extnet"
CONFIG_NEUTRON_OVN_EXTERNAL_PHYSNET=extnet

# Interface for the Open vSwitch tunnel. Packstack overrides the IP
# address used for tunnels on this hypervisor to the IP found on the
# specified interface (for example, eth1).
CONFIG_NEUTRON_OVN_TUNNEL_IF=

# Comma-separated list of subnets (for example,
# 192.168.10.0/24,192.168.11.0/24) used for sending tunneling packets.
# This is used to configure IP filtering to accept tunneling packets
# from these subnets instead of specific IP addresses of peer nodes.
# This is useful when you add existing nodes to EXCLUDE_SERVERS
# because, in this case, packstack cannot modify the IP filtering of
# the existing nodes.
CONFIG_NEUTRON_OVN_TUNNEL_SUBNETS=

# Password to use for the OpenStack File Share service (manila) to
# access the database.
CONFIG_MANILA_DB_PW=PW_PLACEHOLDER

# Password to use for the OpenStack File Share service (manila) to
# authenticate with the Identity service.
CONFIG_MANILA_KS_PW=PW_PLACEHOLDER

# Backend for the OpenStack File Share service (manila); valid
# options are: generic, netapp, glusternative, or glusternfs.
# ['generic', 'netapp', 'glusternative', 'glusternfs']
CONFIG_MANILA_BACKEND=generic

# Denotes whether the driver should handle the responsibility of
# managing share servers. This must be set to false if the driver is
# to operate without managing share servers. Defaults to 'false'
# ['true', 'false']
CONFIG_MANILA_NETAPP_DRV_HANDLES_SHARE_SERVERS=false

# The transport protocol used when communicating with the storage
# system or proxy server. Valid values are 'http' and 'https'.
# Defaults to 'https'. ['https', 'http']
CONFIG_MANILA_NETAPP_TRANSPORT_TYPE=https

```

```

# Administrative user account name used to access the NetApp storage
# system. Defaults to ''.
CONFIG_MANILA_NETAPP_LOGIN=admin

# Password for the NetApp administrative user account specified in
# the CONFIG_MANILA_NETAPP_LOGIN parameter. Defaults to ''.
CONFIG_MANILA_NETAPP_PASSWORD=

# Hostname (or IP address) for the NetApp storage system or proxy
# server. Defaults to ''.
CONFIG_MANILA_NETAPP_SERVER_HOSTNAME=

# The storage family type used on the storage system; valid values
# are ontap_cluster for clustered Data ONTAP. Defaults to
# 'ontap_cluster'. ['ontap_cluster']
CONFIG_MANILA_NETAPP_STORAGE_FAMILY=ontap_cluster

# The TCP port to use for communication with the storage system or
# proxy server. If not specified, Data ONTAP drivers will use 80 for
# HTTP and 443 for HTTPS. Defaults to '443'.
CONFIG_MANILA_NETAPP_SERVER_PORT=443

# Pattern for searching available aggregates for NetApp provisioning.
# Defaults to '(.*)'.
CONFIG_MANILA_NETAPP_AGGREGATE_NAME_SEARCH_PATTERN=(.*)

# Name of aggregate on which to create the NetApp root volume. This
# option only applies when the option
# CONFIG_MANILA_NETAPP_DRV_HANDLES_SHARE_SERVERS is set to True.
CONFIG_MANILA_NETAPP_ROOT_VOLUME_AGGREGATE=

# NetApp root volume name. Defaults to 'root'.
CONFIG_MANILA_NETAPP_ROOT_VOLUME_NAME=root

# This option specifies the storage virtual machine (previously
# called a Vserver) name on the storage cluster on which provisioning
# of shared file systems should occur. This option only applies when
# the option driver_handles_share_servers is set to False. Defaults to
# ''.
CONFIG_MANILA_NETAPP_VSERVER=

# Denotes whether the driver should handle the responsibility of
# managing share servers. This must be set to false if the driver is
# to operate without managing share servers. Defaults to 'true'.
# ['true', 'false']
CONFIG_MANILA_GENERIC_DRV_HANDLES_SHARE_SERVERS=true

# Volume name template for Manila service. Defaults to 'manila-
# share-%s'.
CONFIG_MANILA_GENERIC_VOLUME_NAME_TEMPLATE=manila-share-%s

# Share mount path for Manila service. Defaults to '/shares'.
CONFIG_MANILA_GENERIC_SHARE_MOUNT_PATH=/shares

```

```

# Location of disk image for Manila service instance. Defaults to '
CONFIG_MANILA_SERVICE_IMAGE_LOCATION=https://www.dropbox.com/s/vi5oeh10q1qkckh/ubuntu
_1204_nfs_cifs.qcow2

# User in Manila service instance.
CONFIG_MANILA_SERVICE_INSTANCE_USER=ubuntu

# Password to service instance user.
CONFIG_MANILA_SERVICE_INSTANCE_PASSWORD=ubuntu

# Type of networking that the backend will use. A more detailed
# description of each option is available in the Manila docs. Defaults
# to 'neutron'. ['neutron', 'nova-network', 'standalone']
CONFIG_MANILA_NETWORK_TYPE=neutron

# Gateway IPv4 address that should be used. Required. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_GATEWAY=

# Network mask that will be used. Can be either decimal like '24' or
# binary like '255.255.255.0'. Required. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_NETMASK=

# Set it if network has segmentation (VLAN, VXLAN, etc). It will be
# assigned to share-network and share drivers will be able to use this
# for network interfaces within provisioned share servers. Optional.
# Example: 1001. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_SEG_ID=

# Can be IP address, range of IP addresses or list of addresses or
# ranges. Contains addresses from IP network that are allowed to be
# used. If empty, then will be assumed that all host addresses from
# network can be used. Optional. Examples: 10.0.0.10 or
# 10.0.0.10-10.0.0.20 or
# 10.0.0.10-10.0.0.20,10.0.0.30-10.0.0.40,10.0.0.50. Defaults to ''.
CONFIG_MANILA_NETWORK_STANDALONE_IP_RANGE=

# IP version of network. Optional. Defaults to '4'. ['4', '6']
CONFIG_MANILA_NETWORK_STANDALONE_IP_VERSION=4

# List of GlusterFS servers that can be used to create shares. Each
# GlusterFS server should be of the form [remoteuser@]<volserver>, and
# they are assumed to belong to distinct Gluster clusters.
CONFIG_MANILA_GLUSTERFS_SERVERS=

# Path of Manila host's private SSH key file.
CONFIG_MANILA_GLUSTERFS_NATIVE_PATH_TO_PRIVATE_KEY=

# Regular expression template used to filter GlusterFS volumes for
# share creation. The regex template can optionally (ie. with support
# of the GlusterFS backend) contain the #{size} parameter which
# matches an integer (sequence of digits) in which case the value
# shall be interpreted as size of the volume in GB. Examples: "manila-
# share-volume-d+${", "manila-share-volume-#{size}G-d+${"; with matching
# volume names, respectively: "manila-share-volume-12", "manila-share-
# volume-3G-13". In latter example, the number that matches "#{size}",

```

```

# that is, 3, is an indication that the size of volume is 3G.
CONFIG_MANILA_GLUsterFS_VOLUME_PATTERN=

# Specifies the GlusterFS volume to be mounted on the Manila host.
# For e.g: [remoteuser@]<volserver>:/<volid>
CONFIG_MANILA_GLUsterFS_TARGET=

# Base directory containing mount points for Gluster volumes.
CONFIG_MANILA_GLUsterFS_MOUNT_POINT_BASE=

# Type of NFS server that mediate access to the Gluster volumes
# (Gluster or Ganesha).
CONFIG_MANILA_GLUsterFS_NFS_SERVER_TYPE=gluster

# Path of Manila host's private SSH key file.
CONFIG_MANILA_GLUsterFS_PATH_TO_PRIVATE_KEY=

# Remote Ganesha server node's IP address.
CONFIG_MANILA_GLUsterFS_GANESHA_SERVER_IP=

# Specify 'y' to set up Horizon communication over https. ['y', 'n']
CONFIG_HORIZON_SSL=n

# Secret key to use for Horizon Secret Encryption Key.
CONFIG_HORIZON_SECRET_KEY=726da2ba67fe429cbc9ec3c0eb944497

# PEM-encoded certificate to be used for SSL connections on the https
# server. To generate a certificate, leave blank.
CONFIG_HORIZON_SSL_CERT=

# SSL keyfile corresponding to the certificate if one was specified.
# The certificate should not require a passphrase.
CONFIG_HORIZON_SSL_KEY=

CONFIG_HORIZON_SSL_CACERT=

# Password to use for the Object Storage service to authenticate with
# the Identity service.
CONFIG_SWIFT_KS_PW=5edb32e5d0f14d78

# Comma-separated list of devices to use as storage device for Object
# Storage. Each entry must take the format /path/to/dev (for example,
# specifying /dev/vdb installs /dev/vdb as the Object Storage storage
# device; Packstack does not create the filesystem, you must do this
# first). If left empty, Packstack creates a loopback device for test
# setup.
CONFIG_SWIFT_STORAGES=

# Number of Object Storage storage zones; this number MUST be no
# larger than the number of configured storage devices.
CONFIG_SWIFT_STORAGE_ZONES=1

# Number of Object Storage storage replicas; this number MUST be no
# larger than the number of configured storage zones.
CONFIG_SWIFT_STORAGE_REPLICAS=1

```

```
# File system type for storage nodes. ['xfs', 'ext4']
CONFIG_SWIFT_STORAGE_FSTYPE=ext4

# Custom seed number to use for swift_hash_path_suffix in
# /etc/swift/swift.conf. If you do not provide a value, a seed number
# is automatically generated.
CONFIG_SWIFT_HASH=8e59cd8cf2bc415e

# Size of the Object Storage loopback file storage device.
CONFIG_SWIFT_STORAGE_SIZE=2G

# Password used by Orchestration service user to authenticate against
# the database.
CONFIG_HEAT_DB_PW=80744e05c4844853

# Encryption key to use for authentication in the Orchestration
# database (16, 24, or 32 chars).
CONFIG_HEAT_AUTH_ENC_KEY=ebc3ff7d30fd46b4

# Password to use for the Orchestration service to authenticate with
# the Identity service.
CONFIG_HEAT_KS_PW=a25d421e9e934a62

# Specify 'y' to install the Orchestration CloudFormation API. ['y',
# 'n']
CONFIG_HEAT_CFN_INSTALL=y

# Name of the Identity domain for Orchestration.
CONFIG_HEAT_DOMAIN=heat

# Name of the Identity domain administrative user for Orchestration.
CONFIG_HEAT_DOMAIN_ADMIN=heat_admin

# Password for the Identity domain administrative user for
# Orchestration.
CONFIG_HEAT_DOMAIN_PASSWORD=a2604e28ebd742bb

# Specify 'y' to provision for demo usage and testing. ['y', 'n']
CONFIG_PROVISION_DEMO=n

# Specify 'y' to configure the OpenStack Integration Test Suite
# (tempest) for testing. The test suite requires OpenStack Networking
# to be installed. ['y', 'n']
CONFIG_PROVISION_TEMPEST=n

# CIDR network address for the floating IP subnet.
CONFIG_PROVISION_DEMO_FLOATRANGE=172.24.4.0/24

# Allocation pools in the floating IP subnet.
CONFIG_PROVISION_DEMO_ALLOCATION_POOLS=[]

# The name to be assigned to the demo image in Glance (default
# "cirros").
CONFIG_PROVISION_IMAGE_NAME=cirros
```

```
# A URL or local file location for an image to download and provision
# in Glance (defaults to a URL for a recent "cirros" image).
CONFIG_PROVISION_IMAGE_URL=http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-
x86_64-disk.img

# Format for the demo image (default "qcow2").
CONFIG_PROVISION_IMAGE_FORMAT=qcow2

# Properties of the demo image (none by default).
CONFIG_PROVISION_IMAGE_PROPERTIES=

# User to use when connecting to instances booted from the demo
# image.
CONFIG_PROVISION_IMAGE_SSH_USER=cirros

# Name of the uec image created in Glance used in tempest tests
# (default "cirros-uec").
CONFIG_PROVISION_UEC_IMAGE_NAME=cirros-uec

# URL of the kernel image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_KERNEL_URL=http://download.cirros-cloud.net/0.3.5/cirros-
0.3.5-x86_64-kernel

# URL of the ramdisk image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_RAMDISK_URL=http://download.cirros-cloud.net/0.3.5/cirros-
0.3.5-x86_64-initramfs

# URL of the disk image copied to Glance image for uec image
# (defaults to a URL for a recent "cirros" uec image).
CONFIG_PROVISION_UEC_IMAGE_DISK_URL=http://download.cirros-cloud.net/0.3.5/cirros-
0.3.5-x86_64-disk.img

CONFIG_TEMPEST_HOST=

# Name of the Integration Test Suite provisioning user. If you do not
# provide a user name, Tempest is configured in a standalone mode.
CONFIG_PROVISION_TEMPEST_USER=

# Password to use for the Integration Test Suite provisioning user.
CONFIG_PROVISION_TEMPEST_USER_PW=PW_PLACEHOLDER

# CIDR network address for the floating IP subnet.
CONFIG_PROVISION_TEMPEST_FLOATRANGE=172.24.4.0/24

# Primary flavor name to use in Tempest.
CONFIG_PROVISION_TEMPEST_FLAVOR_NAME=m1.nano

# Primary flavor's disk quota in Gb.
CONFIG_PROVISION_TEMPEST_FLAVOR_DISK=1

# Primary flavor's ram in Mb.
CONFIG_PROVISION_TEMPEST_FLAVOR_RAM=128
```

```
# Primary flavor's vcpus number.
CONFIG_PROVISION_TEMPEST_FLAVOR_VCPUS=1

# Alternative flavor name to use in Tempest.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_NAME=m1.micro

# Alternative flavor's disk quota in Gb.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_DISK=1

# Alternative flavor's ram in Mb.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_RAM=128

# Alternative flavor's vcpus number.
CONFIG_PROVISION_TEMPEST_FLAVOR_ALT_VCPUS=1

# Specify 'y' to run Tempest smoke test as last step of installation.
CONFIG_RUN_TEMPEST=n

# Test suites to run, example: "smoke dashboard TelemetryAlarming".
# Optional, defaults to "smoke".
CONFIG_RUN_TEMPEST_TESTS=smoke

# Specify 'y' to configure the Open vSwitch external bridge for an
# all-in-one deployment (the L3 external bridge acts as the gateway
# for virtual machines). ['y', 'n']
CONFIG_PROVISION_OVS_BRIDGE=y

# Password to use for Gnocchi to access the database.
CONFIG_GNOCCHI_DB_PW=0aaadca5dc414623

# Password to use for Gnocchi to authenticate with the Identity
# service.
CONFIG_GNOCCHI_KS_PW=dba1bc4a677542bc

# Secret key for signing Telemetry service (ceilometer) messages.
CONFIG_CEILOMETER_SECRET=068099e93acd4e27

# Password to use for Telemetry to authenticate with the Identity
# service.
CONFIG_CEILOMETER_KS_PW=0d2cf023d9bd4369

# Ceilometer service name. ['httpd', 'ceilometer']
CONFIG_CEILOMETER_SERVICE_NAME=httpd

# Backend driver for Telemetry's group membership coordination.
# ['redis', 'none']
CONFIG_CEILOMETER_COORDINATION_BACKEND=redis

# Whether to enable ceilometer middleware in swift proxy. By default
# this should be false to avoid unnecessary load.
CONFIG_ENABLE_CEILOMETER_MIDDLEWARE=n

# IP address of the server on which to install the Redis server.
CONFIG_REDIS_HOST=192.168.1.120
```



```
# Port on which the Redis server listens.
CONFIG_REDIS_PORT=6379

# Password to use for Telemetry Alarming to authenticate with the
# Identity service.
CONFIG_AODH_KS_PW=5f80c0a7864347f6

# Password to use for Telemetry Alarming (AODH) to access the
# database.
CONFIG_AODH_DB_PW=24aa816f81d54898

# Password to use for Panko to access the database.
CONFIG_PANKO_DB_PW=PW_PLACEHOLDER

# Password to use for Panko to authenticate with the Identity
# service.
CONFIG_PANKO_KS_PW=PW_PLACEHOLDER

# Password to use for OpenStack Database-as-a-Service (trove) to
# access the database.
CONFIG_TROVE_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Database-as-a-Service to authenticate
# with the Identity service.
CONFIG_TROVE_KS_PW=PW_PLACEHOLDER

# User name to use when OpenStack Database-as-a-Service connects to
# the Compute service.
CONFIG_TROVE_NOVA_USER=trove

# Tenant to use when OpenStack Database-as-a-Service connects to the
# Compute service.
CONFIG_TROVE_NOVA_TENANT=services

# Password to use when OpenStack Database-as-a-Service connects to
# the Compute service.
CONFIG_TROVE_NOVA_PW=PW_PLACEHOLDER

# Password to use for OpenStack Data Processing (sahara) to access
# the database.
CONFIG_SAHARA_DB_PW=PW_PLACEHOLDER

# Password to use for OpenStack Data Processing to authenticate with
# the Identity service.
CONFIG_SAHARA_KS_PW=PW_PLACEHOLDER

# Password to use for the Magnum to access the database.
CONFIG_MAGNUM_DB_PW=PW_PLACEHOLDER

# Password to use for the Magnum to authenticate with the Identity
# service.
CONFIG_MAGNUM_KS_PW=PW_PLACEHOLDER
```

Figura 49: Archivo de respuesta.