DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE NUBE COMPUTACIONAL BASADOS EN LA PLATAFORMA DE CÓDIGO ABIERTO OPENSTACK.

Hernán Alfredo Hinojosa Rizzo
Email: hernan.hinojosa@gmail.com
Milton Patricio Ulloa Ibarra
Email: mpulloa@yahoo.com

Abstracto – Los Sistemas de Nubes Computacionales han ganado gran espacio e interés en los últimos tiempos debido a las múltiples aplicaciones en las que se encuentra instalada y promocionada, Sin embargo todavía resulta abstracta su comprensión y alcance. El presente documento tiene por objetivo el de describir el concepto de Nube Computacional bajo la perspectiva de Infraestructura como Servicio utilizando la plataforma de código abierto Openstack.

I. INTRODUCCIÓN

Dada la importancia que ha tomado el concepto de Nube Computacional y las oportunidades de negocio que se han generado a su alrededor, todas o casi todas las grandes empresas IT, tanto de hardware como de software, han demostrado su interés a través de estrategias a largo plazo para implementar plataformas bajo su propia marca, Así mismo la comunidad de desarrolladores independientes ha lanzado iniciativas bajo la licencia de código abierto Apache 2.0, siendo una de ellas Openstack.

La nube computacional o Cloud Computing se compone de dos términos; Nube o Cloud, que es el símbolo que se usa generalmente para representar de cierta manera a un sistema abstracto, como por ejemplo al Internet; y Computacional o Computing, que hace referencia a los términos relacionados con la informática, programas y almacenamiento de datos. Por tal motivo muchos autores denominan a la nube computacional como un paradigma que permite ofrecer servicios de computación a través de una red de datos tipo LAN, WAN o a través del Internet.

El Laboratorio de Tecnologías de la Información, integrado en el *National Institute of Standards and Technology* (NIST) del Departamento de Comercio del Gobierno Federal de los Estados Unidos, ha definido *Cloud Computing* de la siguiente forma:

"Cloud Computing es un modelo que permite el acceso bajo demanda y a través de la red a un conjunto de recursos compartidos y configurables (como redes, servidores, capacidad de almacenamiento, aplicaciones y servicios) que pueden ser rápidamente asignados y liberados con una mínima gestión por parte del proveedor del servicio"

La nube en sí misma facilita la entrada, procesamiento y salida de la información como un servicio. Los servicios de la nube incluyen el software, infraestructura y almacenamiento en la red basadas en la demanda del usuario.

Según el NIST, el modelo de la nube computacional tiene cinco características esenciales, así como también tres modelos de servicio y cuatro modelos de implementación:

1.4 Servicios a la carta.

El usuario puede acceder a capacidades de computación en la nube de forma automática a medida que las vaya necesitando sin necesidad de una interacción humana con su proveedor.

Esta característica incluye el tema tanto de logística de la implementación de los recursos así como el pago que corresponde su utilización.

1.5 Múltiples formas de acceder a la red.

Los recursos son accesibles a través de la red y por medio de mecanismos estándar que permiten su utilización por cualquier tipo de equipos heterogéneos por ejemplo desde los computadores de escritorio hasta teléfonos móviles o tabletas.

Esta característica denominada ubicuidad permite que múltiples sistemas puedan acceder a un mismo servicio desde cualquier ubicación física (siempre que cuenten con acceso a la dicha red), y es una de las principales bondades que aporta la Nube Computacional.

Con el objetivo de proporcionar dicha ubicuidad, los proveedores de la "nube" cuentan con la infraestructura física o virtual y el ancho de banda necesarios para dar cabida a los requisitos de los diferentes equipos que se subscriben. Además, los proveedores disponen de rutas redundantes y balanceadas en las redes de comunicaciones de acceso a sus servicios, lo cual ofrece mayor garantía en el equilibrio de la carga de datos por

esas rutas, reduciendo la posibilidad de que las redes se sobrecarguen y que los servicios ofrecidos se retrasen, fallen o se corten.

1.1 **Compartición de recursos.**
Los recursos computacionales tales como almacenamiento, memoria, ancho de banda, capacidad de procesamiento, máquinas virtuales, etc. de los proveedores son compartidos bajo el modelo multi-usuarios, a los que se van asignando capacidades tanto físicas como virtuales de manera dinámica según sus requerimientos. Los usuarios pueden desconocer la ubicación de los recursos a los que acceden y en algunos casos solo podrán seleccionar algunos aspectos tales como el Centro de cómputo donde se aloja la infraestructura.

Siendo la virtualización el método por el cual es posible una optimización respecto al aprovechamiento de los recursos instalados ya que permite que **las aplicaciones sean independientes del hardware** en el que se alojan, es decir que varias aplicaciones pueden ejecutarse en una misma máquina o una aplicación puede usar varios equipos al mismo tiempo, dependiendo de la necesidad del usuario.

Consideraciones referentes a la seguridad de la información de cada usuario son necesarias al momento de implementar entornos virtuales en los que los recursos de almacenamiento son compartidos, por lo que es necesario establecer controles adecuados de acceso y gestión segura de la información en cada uno de los niveles informáticos mediante el cifrado de los datos.

1.2 Rapidez y Elasticidad.
Por elasticidad de la Nube computacional se refiere a la capacidad de asignar y liberar los recursos computacionales físicos o virtuales en algunos casos hasta de manera automática y que su crecimiento y decrecimiento sea asociado con la demanda. Para el usuario, los recursos serán considerados como ilimitados y podrán ser asignados en cualquier cantidad que se encuentran siempre disponibles.

1.3 Servicio ponderado.
El controlador de la Nube Computacional tiene la capacidad y las herramientas para medir, a determinado nivel, el servicio y que sea efectivamente entregado a cada usuario, de menara que para el Proveedor y para el Usuario sea clara y conocida la información relacionada al consumo real de los recursos, lo que posibilita el pago por el uso efectivo de los servicios.

II. CONCEPTOS

Para comprender las bondades y capacidades que la nube computacional puede ofrecer es necesario dividirla en las siguientes capas o layers:

2.1 Software como Servicio (SaaS)

Corresponde a la capa más alta y consiste en la entrega de una aplicación completa como un servicio. En este caso el proveedor de SaaS dispone de una aplicación estándar desarrollada, operada y mantenida en algunos casos por él mismo y con la que brinda servicio a una masa de usuarios a través de la red, sin que estos tengan que instalar ningún programa adicional en sus equipos.
Los proveedores de SaaS son responsables de la disponibilidad y funcionalidad de sus servicios sin omitir las necesidades de los usuarios que finalmente son los que se beneficiarán del software.
Las actividades son gestionadas desde alguna ubicación central, en lugar de hacerlo desde cada lugar o equipo, permitiendo a los usuarios el acceso remoto a las aplicaciones mediante la Red. Igualmente, las actualizaciones son centralizadas, eliminando la necesidad de descargar parches por parte de los usuarios finales para sus equipos.

Ventajas
Ventajas comparativas con respecto a las aplicaciones tradicionales son las siguientes:

- *Inversión inicial:* Las implementaciones tradicionales tienen un alto costo relacionado con la adquisición de licencias, en el caso de SaaS el pago se realiza por el tiempo de uso de la aplicación.
- *Multiplataforma:* El acceso a las aplicaciones en SaaS se realiza mediante el navegador de Internet por lo tanto es independiente del sistema operativo o de la plataforma utilizada por el usuario.
- *Actualizaciones:* En este caso el proveedor de SaaS se encargará de gestionar los aspectos referentes a las mejoras en las aplicaciones, ofreciendo de tal manera al usuario acceso a las últimas versiones de los programas.
- *Logística:* Generalmente las aplicaciones tradicionales implican que exista un departamento de Sistemas o IT por detrás que gestione y opere la plataforma en la que se ejecutan, con SaaS se disminuye la gestión de las aplicaciones así como también el tiempo que se destina al soporte y mantenimiento, permitiendo al departamento de IT direccionar

mayor tiempo y recursos hacia el giro de negocio de la empresa.

Consideraciones

Al momento de elegir un proveedor de SaaS se recomienda considerar los siguientes aspectos:

Técnicos: Los enlaces de datos pueden resultar en un cuello de botella al momento de utilizar el SaaS, hay que realizar los cálculos previos para verificar que el enlace corporativo permitirá un uso adecuado. Así mismo otro tema a considerar es la personalización del SaaS, en algunos casos las organizaciones requieren de un programa específico para su necesidad, el SaaS personalizado puede resultar más costoso que una aplicación alojada localmente.

Económicos: Muy probablemente la inversión inicial al implementar el SaaS será menor que la inversión en una plataforma tradicional, sin embargo hay que analizar a largo plazo ya que SaaS necesariamente requiere pagos mensuales por el uso.

Legales: La sensibilidad de la información es un factor a considerar al momento de elegir el proveedor de SaaS, los acuerdos de servicio antes mencionados deben especificar aspectos relacionados a la seguridad de los datos de los usuarios.

2.2 Plataforma como Servicio (PaaS)

En orden descendente, PaaS (Platform as a Service) es la siguiente capa. Básicamente su objetivo se centra en un modelo en el que se proporciona un servicio de plataforma con todo lo necesario para dar soporte al ciclo de planteamiento, desarrollo y puesta en marcha de aplicaciones y servicios a través de la Red

El proveedor es el encargado de los recursos en caso de que la aplicación lo requiera, del rendimiento adecuado de la plataforma y de la seguridad de acceso. Para desarrollar los programas se requieren de bases de datos, herramientas de desarrollo y en ocasiones servidores y redes. Con PaaS el usuario únicamente se enfoca en desarrollar, depurar y probar ya que la herramienta necesaria para el desarrollo es proporcionada a través de la Red

Con el uso de PaaS, se separa del hardware al usuario, lo cual es interesante para muchos desarrolladores, y es probable que llegue a remplazar a las empresas de alojamiento tradicionales.

Incluso, también a los administradores de sistemas ya que no se necesita controlar ninguna infraestructura ni hay optimización posible más allá del código y sus algoritmos.

Ventajas.

Ventajas comparativas con respecto a las aplicaciones tradicionales son las siguientes:

Entorno de desarrollo y producción: En los procesos tradicionales se tienen al menos dos o tres entornos diferentes para la programación, pruebas y puesta en operación de un programa. En PaaS se puede conseguir que todo el proceso de desarrollo y operación se encuentren en la misma plataforma, lo que disminuye la posibilidad de errores por incompatibilidad de plataformas.

Consideraciones.

Al momento de elegir un proveedor de PaaS se recomienda considerar los siguientes aspectos:

Técnicos: La interoperabilidad de las plataformas de desarrollo y producción debe ser analizada detenidamente, en algunos casos PaaS solo ofrece plataformas basadas en lenguajes de programación de acceso libre por lo que restringe la gama de posibilidades de desarrollo de aplicaciones para el usuario.

Económicos: Se puede negociar con el proveedor de PaaS de un tiempo de prueba gratuito que posibilita al usuario de la oportunidad de adquirir la plataforma luego de verificar las bondades de la misma.

Legales: Se debe especificar un acuerdo en el que se detalle las condiciones de uso de la PaaS de tal forma que el usuario no haga uso indebido de la misma y el proveedor tarife un valor conocido por ambas partes tanto por espacio de almacenamiento así como de recursos computacionales (procesador, memoria y ancho de banda).

2.3 Infraestructura como Servicio (IaaS)

IaaS corresponde a la capa más baja. La idea básica es la de tercerización de equipos servidores en busca de espacio en disco, sistemas de base de datos, routers, switchs y/o tiempo de procesamiento en lugar de tener un servidor local y toda la infraestructura y personal necesarios para el soporte, mantenimiento y conectividad.

Las IaaS permiten desplazar al proveedor la mayor parte de los factores relacionados con la gestión de los equipos con el concerniente ahorro al pagar solo por lo consumido y olvidarse del manejo y administración de equipos así como su mantenimiento. Por otro lado, IaaS permite una escalabilidad automática o semiautomática, según los acuerdos preestablecidos, de forma que se puedan obtener más recursos según se requiera.

Para hacer una distinción respecto a las PaaS, las IaaS se presentan como una propuesta con mucho

más flexibilidad para el uso que el usuario la tenga en mente, pero también requieren mucho más involucramiento por parte del usuario en lo que a instalación, configuración y mantenimiento del software se refiere.

Ventajas
Ventajas comparativas con respecto a las aplicaciones tradicionales son las siguientes:
Rendimiento: el usuario se encarga de la gestión de los recursos asignados de acuerdo a sus necesidades incrementando el rendimiento de sus aplicaciones de acuerdo a la demanda interna, por otro lado el proveedor que gestione la plataforma de tal manera que los recursos son notablemente mejor aprovechados, beneficiando con un menor precio al usuario.
Infraestructura: Los equipos son propiedad del proveedor de IaaS por lo tanto la responsabilidad de mantenimiento y renovación ya no le corresponden al usuario. Sin embargo se debe considerar al momento de elegir un proveedor de IaaS que demuestre su planificación respecto a la permanencia del servicio en el transcurso del tiempo, lo que menos desea el usuario es que su proveedor desaparezca repentinamente.
Fiabilidad: La inversión que implica la redundancia de sistemas en muchos casos no es posible asumirla por parte del usuario, bajo la modalidad de IaaS es muy posible que las pequeñas organizaciones accedan a este tipo de beneficios incluidos en el servicio contratado por defecto.

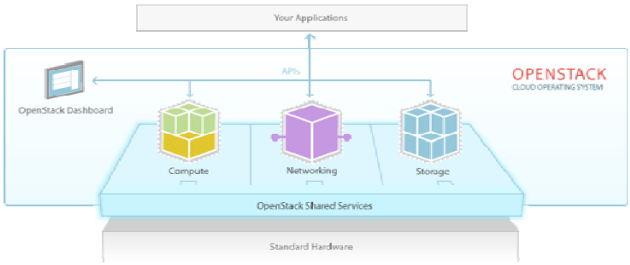
Consideraciones
Al momento de elegir un proveedor de IaaS se recomienda considerar los siguientes aspectos:
Técnicos: Puede ser un limitante la cantidad de Sistemas Operativos y aplicaciones tales como las de bases de datos o alojamiento web que el proveedor puede ofrecer. Dicha limitación será traspasada al usuario de tal manera que debe considerarse al momento de planificar alguna nueva implementación. Por otro lado como aspecto favorable está la posibilidad de que los usuarios puedan acceder a la infraestructura arrendada desde cualquier lugar a través de una conexión de datos fiable.
Económicas: IaaS ofrece al usuario varios modelos de pago entre los que se detalla; pago por consumo de recursos utilizados en un periodo de tiempo acordado, pago por recursos reservados de forma exclusiva o pago por recursos utilizados en un pico de carga.
Legales: Como se mencionó anteriormente, es importante que en el acuerdo de servicio se determine la duración del servicio a fin de evitar sorpresivos cierras durante el periodo de operación.

III. Openstack

Openstack comprende una colección de varios proyectos bajo licenciamiento Apache 2.0 cuyo desarrollo y mantenimiento está a cargo de los usuarios o comunidades inscritas para ello. Al momento de la realización del presente documento se encontraban adscritas al proyecto cerca de 175 compañías y 13700 personas, datos que reflejan la importancia que tiene esta tecnología tanto desde el punto de vista científico como desde su aspecto comercial, ya que si buscamos en el Internet, es muy fácil encontrar varias implementaciones comerciales de la aplicación de Openstack como núcleo y base de los servicios ofertados.

En sus inicios, el proyecto se encontraba a cargo de dos gigantes de la industria de aplicaciones tecnológicas, por un lado Rackspace, que ha liberado el código de sus servicios Cloud Files y Cloud Servers; y por otro NASA que ha colaborado con el código de su plataforma Nébula para potenciar su contraparte computacional. Ambos códigos forman núcleo inicial de OpenStack y han sido desarrollados utilizando Python, ofreciendo la posibilidad a usuarios o empresas de **crear sus propios servicios IaaS de Nube computacional**. Sin embargo durante el último año se han integrado a la dirección del proyecto otros gigantes como HP y Cisco para aportar sus experiencias en cada uno de sus campos de especialización. La directiva del proyecto se elige ahora por votación.

3.1 Componentes de Openstack
Openstack está formado por tres componentes principales; el componente de **infraestructura computacional**, el de **almacenamiento de objetos** e imágenes y finalmente del relacionado a los **servicios de red**. Todos ellos forman parte de las herramientas para crear y administrar los servicios en una nube de equipos computacionales, entre los que se incluye la ejecución de Instancias, la administración de los componentes de Red y los permisos de acceso a la misma.
Se entiende por Instancias a las Máquinas Virtuales o recursos computacionales administrados por Nova, creadas por imágenes almacenadas en Glance y elegidas de varias Recetas (flavors) predeterminados o personalizados.



Cada Componente cuenta con sus subcomponentes amparados bajo proyectos con diferentes nombres, A continuación un breve resumen de los componentes y su interacción con los demás:

- **Infraestructura Computacional – Nova:** Dirige, almacena y recupera los discos virtuales y recursos asociados a Glance y les asocia los recursos de infraestructura necesarios para su ejecución.
- **Servidor de Imágenes – Glance:** Almacena y recupera los discos virtuales activos en Swift
- Repositorio de Objetos – **Swift:** Servicio de Almacenamiento.
- **Panel de Control – Horizon:** Corresponde al aplicativo gráfico de interacción entre los servicios de Openstack y los usuarios.
- **Autenticación – Keystone**: Se utiliza para autenticar la ejecución de los diferentes servicios bajo la modalidad de ID y contraseña únicos para cada instancia.
- **Red – Neutron:** inicialmente denominado como **Quantum**, maneja las redes y las direcciones IP de la nube de tal manera que no se conviertan en un cuello de botella que influya negativamente en el desempeño de los otros componentes.
- **Telemetría – Ceilometer:** Genera la información relacionada a medición y monitoreo de la nube de tal manera que dicha información sea utilizada para medir el desempeño de la misma, así como también para procesos de facturación de los recursos utilizados.
- **Orquestación – Heat:** este componente se encarga de organizar y coordinar los diferentes estamentos de nube predefiniendo patrones de uso de acuerdo a las aplicaciones requeridas por el usuario.

Técnicamente, los componentes de Openstack han sido programados usando Phyton y la intercomunicación entre ellos se realiza mediante instrucciones API bajo estándares generales y que en conjunto forman un todo diseñado para proveer un Sistema Operativo para la Nube Computacional altamente escalable.

Los diferentes Accesos Programables que conformas los proyectos de Openstack se programan bajo la arquitectura REST. Para comprender las bondades de este API, nos referimos a su definición y alcance, siendo REST determinado de la siguiente manera: "***Transferencia de Estado Representacional*** *(Representational State Transfer) o* ***REST*** *es una técnica de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web.*"
Cada uno de los APIs permiten una intercomunicación entre los usuarios y los diferentes servicios o proyectos a través de un sistema de colas de mensajes siguiendo el protocolo de mensaje AMQP (Advanced Message Queuing Protocol), en este caso Openstack utiliza la aplicación de negociación de mensajes en código abierto denominada RabbitMQ, sin embargo ya se está en marcha el proyecto Marconi propio de Openstack y especializado en el manejo de mensajería y administración de solicitudes a gran escala.

3.2 Hipervisor

Openstack utiliza varios hipervisores entre los que se incluyen; KVM, LXC., QEMU (Quick Emulator), UML, VMWare ESX/ESXi, Xen, PowerVM, Hyper-V.
Probablemente el factor más importante al momento de elegir el hipervisor que será utilizado se basa en el nivel de conocimientos previos y experiencia que se cuente sobre alguno de ellos. Otros aspectos a considerar son las características y bondades de cada uno y la documentación así como el soporte técnico que haya disponible.
Para nuestro caso, vimos que KVM es el hipervisor recomendado por la comunidad de Openstack y que ha sido mayormente utilizado en las diferentes implementaciones que hemos encontrado, sin embargo también había aplicaciones con Xen y VMware y Hiper-V aunque la documentación no era muy descriptiva o utilizaban una versión de Openstack anterior a Grizzly o Havana.
KVM son las iniciales de Kernel-based Virtual Machine y corresponde a la categoría de hipervisores Tipo 1 ya que se integra en un sistema operativo existente. Así mismo se ha convertido en el hipervisor de virtualización oficial del kernel de Linux desde la versión 2.6.20 facilitando su implementación en otras versiones y aprovechando de los aportes de importantes compañías. Actualmente es utilizado por SUSE y por Ubuntu en sus versiones oficiales.
Proporciona de virtualización en arquitecturas x86 que cuentan con extensiones Intel VT (Virtualization Techonology) o AMD-V (Virtualization) motivo por el cuál previo a su implementación, es necesario verificar si el equipo

está capacitado para su ejecución mediante el siguiente comando:
 # Kvm-ok
Si la respuesta es:
INFO: /dev/kvm exists
KVM acceleration can be used

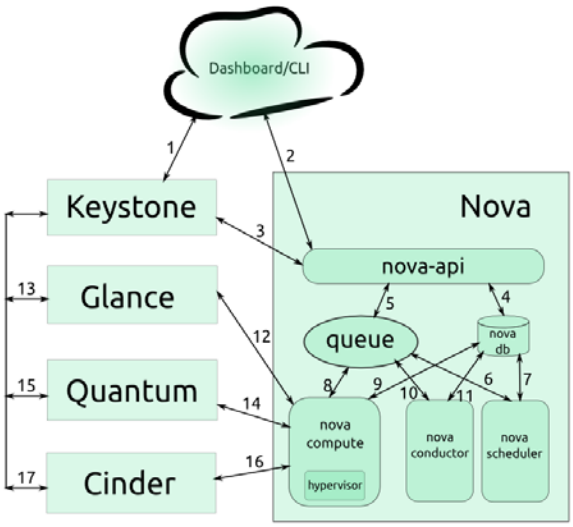
Significa que podemos continuar, caso contrario se debe verificar en el BIOS del equipo si es posible habilitar la virtualización del procesador.
Cuando se carga el módulo KVM, Linux se convierte en un hipervisor capaz de ejecutar las máquinas virtuales aisladas y las hospeda como procesos, por lo que cada una puede beneficiarse de todas las características del kernel de Linux, incluyendo las redes, seguridades y almacenamiento.
Así mismo todos los avances que se hagan al kernel de Linux como sistema Operativo, también se aplican a KVM, como por ejemplo (Tholeti, 2011):

- Las mejoras en el manejo de los núcleos de CPU y de la memoria RAM de un equipo por Linux, también serán trasladadas a las máquinas virtuales de KVM.
- Bajo KVM, la gestión de las máquinas virtuales se la mira como un proceso, por lo tanto si el kernel es mejorado en esa área, la gestión de KVM también se verá mejorada.
- Las imágenes de los discos de las máquinas virtuales se administran como cualquier archivo en Linux por lo tanto su almacenamiento puede realizarse bajo cualquier sistema soportado: NAS, iSCSI, SAN.
- KVM utiliza cualquier dispositivo que sea a su vez soportado por Linux.
- KVM también puede aprovecharse del modelo de seguridad de Linux, el programa Vagrant (http://www.vagrantup.com/) que crea "sandboxes" o contenedores que encierran a las máquinas virtuales basados en los lineamientos de seguridad que aíslan las máquinas virtuales entre ellas. (Jackson, 2013)

Linux y KVM utilizan las librerías **libvirt** (http://libvirt.org/) y **libguestfs** (http://libguestfs.org/) para estandarizar sus comunicaciones a través de sus APIs, base para la gestión de las máquinas virtuales y de las imágenes.
Cabe mencionar que libvirt puede gestionar también a otros hipervisores como Xen, VMWare ESX, OpenVZ, Hyper-V, lo que lo colabora con la versatilidad de Openstack.
Con respecto al manejo de imágenes, el formato nativo de KVM es QCOW2, el cual permite la realización de snapshots, compresión y cifrado de las máquinas virtuales en ejecución, esta características es utilizada por el componente Glance de Openstack para generar respaldos en línea. De esta manera KVM se relaciona con el proyecto de código abierto QEMU, un sistema de monitoreo de máquinas virtuales, ya que QCOW2 es un formato de archivo para imágenes de discos propio de QEMU con características avanzadas como la expansión dinámica, copy-on-write, snapshots, cifrado y compresión entre otros.
Así mismo QEMU se utiliza para proporcionar los dispositivos de entrada y salida que las máquinas virtuales utilizan.
Los hipervisores y las tecnologías de virtualización han ido creciendo y expandiendo su influencia en varios campos de las Tecnologías de la Información. Uno de las evoluciones lógicas ha sido su aplicación como elemento básico en el entorno de las nubes computacionales. Varios proyectos utilizando KVM como hipervisor, entre ellos están Openstack, CloudStack de Apache y OpenNebula.

3.3 Proceso de creación de las instancias

Las instancias se ejecutan en el nodo Computacional bajo la dirección del componente Nova y con el soporte del resto de componentes, cada uno de ellos con su correspondiente responsabilidad en el proceso. A continuación describimos los pasos seguidos en el ciclo de puesta en alta de una máquina virtual



1. El usuario autentica sus credenciales con Keystone vía RESTful API y le retorna un auth_token relacionado a la solicitud y que será utilizado por lo demás componentes

2. Horizon convierte la petición de la instancia en una orden de ejecución que se envía a nova-api
3. Nova-api recibe la solicitud y valida el auth_token y los permisos de acceso con Keystone, quien a su vez responde con la confirmación de los permisos y roles asignados
4. Nova-api interactúa con nova-database creando la base relacionado a la petición de instancia.
5. Nova-api envía una consulta a nova-scheduler para proceder con la petición.
6. Nova-scheduler recoge la petición de la cola
7. Nova-scheduler interactúa con nova-database para encontrar al anfitrión adecuado utilizando el procedimiento de ponderación de filtros y pesos. La respuesta incluye el identificador del anfitrión asignado (host_id) y de inmediato nova-scheduler envía la solicitud a nova-compute para que ejecute la instancia en dicho anfitrión.
8. Nova-compute recupera la petición de la cola de mensajes
9. Nova-compute envía la solicitud a nova-conductor para que busque la información tal como el identificador del anfitrión (host_id) y las características físicas de la instancia (flavor)
10. Nova-conductor recupera la solicitud de la cola de mensajes
11. Nova-conductor interactúa con nova-database para recuperar la información relacionada a la instancia
12. Nova-compute envía la autorización auth_token a glance-api y solicita la URI de la imagen de acuerdo al su identificador.
13. Glance-api valida el auth_toekn con Keystone
    Mientras tanto nova-compute recibe los metadatos de la imagen y envía una solicitud al Neutron-api para que se asigne la información de red a la instancia
14. Neutron-server valida el auth_token con Keystone y remite la información de red solicitada por nova-compute
15. Nova-compute remite el auth_token a Cinder solicitando la asignación de un disco virtual a la instancia.
16. Cinder-api valida el auth_token con Keystone mientras que nova-compute recupera la información del bloque de almacenamiento.

Nova-compute genera los datos para iniciar el driver del hipervisor así como también su puesta en marcha de la instancia en él, a través de libvirt o el API.

IV. Implementación y pruebas realizadas

Para la implementación de la plataforma se debe varios aspectos entre los que se puede detallar:

- Número de Usuarios simultáneos y ocasionales
- Requisitos de Máquinas Virtuales
- Costes de implementación
- Elección del proveedor de equipos
- Planificación de la infraestructura a largo plazo
- Complejidad y control de las actualizaciones

Otro factor a considerar es el modelo de implementación por el que se elegirá, en el caso de nuestro análisis encontramos dos tipos de opciones:

- Multi-Nodo
- Multi-Nodo con Alta disponibilidad

4.1 Multi modo con alta disponibilidad

Ya en entornos de producción donde el concepto de Alta Disponibilidad o High Availability (HA) es un requisito fundamental, se puede implementar Openstack con al menos dos o tres nodos de Control de tal manera que se reduzca los posibles puntos de fallo que afecten al desempeño de la nube. Este tipo de implementaciones son posibles gracias a que los servicios de Openstack están comunicados a través de los RESTful APIs mediante protocolo HTTP y en los mensajes basados en AMQP usando el protocolo RPC. Además de las ventajas propia de Openstack, la implementación HA incluye el uso de otras herramientas de código abierto disponibles, por ejemplo la solución Pacemaker (ClusterLabs, 2013) que es un administrador de recursos escalables y agrupados de alta disponibilidad, tomando la información desde el portal Web de Pacemaker podemos anotar las siguientes características sobre el software:

- Servicio de detección y recuperación de un equipo o aplicación que haya fallado.
- Suporta prácticamente cualquier configuración de redundancia.
- Cuenta con un sistema de estrategias configurables en caso de que muchas máquinas fallen a la vez.
- Controla la ejecución de aplicaciones en el mismo equipo

- Controla las aplicaciones que requieren ser ejecutadas en varios equipos simultáneamente.

Otra aplicación que se utiliza para las implementaciones HA es Galera de Codership, se trata de un programa que permite la clusterización o arreglo de las bases de datos generadas en MySQL, así mismo tomando la información desde el portal web del desarrollador podemos anotar las siguientes características que lo hacen útil para Openstack:

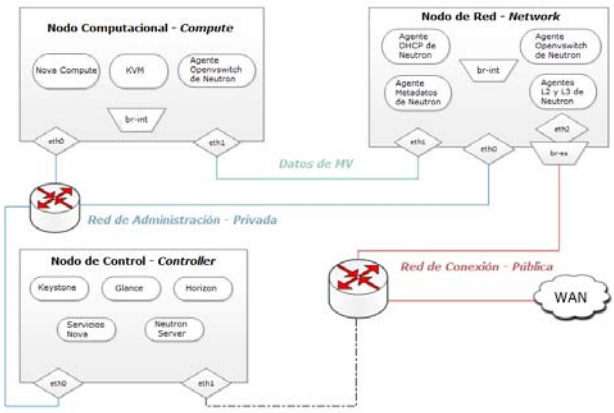
- Permite la replicación sincrónica de las bases de datos
- Posee una topología multi-master activa
- Puede leer y escribir en cualquier nodo de arreglo de bases de datos
- Control automático de los miembros, cualquier nodo que falle es retirado del arreglo.
- Inclusión de nodos relacionados automáticamente.

Conexiones directas basadas en MySQL nativo.

4.2 Multi modo sin alta disponibilidad

Este tipo de arquitectura resulta ideal para pequeñas y mediana empresas que quieren dar sus primeros pasos en el mundo "Cloud". El sistema multi-nodo pero sin Alta Disponibilidad se refiere a la implementación en la que cada nodo será único en su función y en los componentes que alberga, sin embargo permitirá un crecimiento uniforme tanto en almacenamiento como en procesamiento ya que se podrán agregar horizontalmente más equipos que operen como Nodos Computacionales con el objetivo de incrementar el poder de procesamiento y a su vez el número de máquinas virtuales que operan de manera simultánea.

El siguiente diagrama de la topología de red muestra este tipo de implementación:



Para realizar la implementación de la plataforma Openstack, hemos determinado el siguiente equipo:

Servidor Dell Power Edge 2950 para montar en rack de 2 U con las siguientes características:

- Dos procesadores Intel Xeon® de 4 núcleos de 2.66 GHz y 64 bits.
- 32 GB de RAM DDR-2 de 667 MHz para servidor
- 2 Puertos de red 1 Gbps.
- 4 Discos duros de 1 TB.
- 2 Fuentes de poder redundantes
- 1 Lector de DVD
- Controlador RAID de 6 unidades SATA
- Tarjeta iLO dedicada para administración

4.3 Descripción Lógica

Los disco duros están dispuestos en dos arreglos; el primero para el sistema operativo configurados en espejo. Los otros dos discos en mono nativo.

Dejando una capacidad total de 3 TB disponibles para el proyecto.

Para la implementación, hemos utilizado el sistema operativo Linux en la versión **Ubuntu Server 12.04.03 LTS para procesadores de 64 bits**.

La versión de Openstack puede ser instalada en un arreglo de nodos de procesamiento bajo las arquitecturas antes mencionadas o instalando en un solo nodo que a su vez alberga varios equipos virtualizados.

En nuestra investigación, encontramos que las plataformas de virtualización más utilizadas son Vagrant y VMware. Hay documentación disponible en el Internet con muchos detalles sobre los pasos a seguir para instalar Openstack en este tipo de ambientes virtualizados. Sin embargo no es recomendado para entornos de producción por el alto consumo de recursos inmersos solamente en poner en marcha la aplicación, sin embargo la ventaja fundamental es que permite realizar una implementación con más nodos de los que podemos utilizar en ambientes de hardware exclusivamente.

Para la elección de la implementación, nosotros consideramos que la realización de la implementación en ambientes virtuales puede reducir la capacidad que el equipo afectando a su vez el desempeño de las posibles aplicaciones que los estudiantes pueden requerir.

Por otro lado, la implementación de Openstack en varios nodos, mínimo tres como lo mencionamos anteriormente; nodo Controlador, nodo Computacional y nodo de Red resulta muy exigente para el hardware en el que se instala e incluye entre otras cosas a equipos de red, tales

como Switchs de capa L2/L3 con soporte para VLANs.
En el portal de Openstack hay varias recetas de instalación, así como otras en el internet. Para nuestro proyecto hemos optado por la instalación de la plataforma en dos equipos; el primer equipo que incluya al componente de Red y de Almacenamiento y un segundo equipo que contenga al componente computacional donde se albergará a las instancias.

4.4 Máquinas Virtuales

De acuerdo al hardware propuesto vamos a calcular el número de máquinas virtuales que podremos ejecutar de manera simultánea en el nodo Computacional. Para ello utilizamos la siguiente fórmula tomada del libro "Openstack *Operations Guide"* (Tom Fifield, Diane Fleming, 2013) :

$$\#núcleos = \frac{\#\,VM\ x\ Vel\ de\ VM}{Vel\ \ Procesador}$$

Donde:

- #núcleos corresponde a la cantidad de núcleos del Procesador
- # VM corresponde a las Máquinas Virtuales que operaran simultáneamente
- Vel de VM corresponde a la velocidad en GHz que asignaremos a cada instancia, en nuestro caso este valor será de 1,5 GHz

Resolviendo la ecuación:

$$\#VM = \frac{4\ x\ 2.66}{1.5}\ ;$$
$$\#VM = 7.01$$

Por lo tanto el número de instancias que se podrán ejecutar simultáneamente es de siete máquinas virtuales, considerando que no exista la posibilidad de sobre utilizar los recursos existentes. Sin embargo si se activa la opción de Hyper-threading o de Múltiples hilos en el servidor podemos incrementar el número de instancias en 1.3 veces la capacidad inicial obteniendo nueve instancias que se pueden ejecutar conjuntamente.
En el portal web de Openstack (http://www.openstack.com) se encuentra disponible la documentación que guía paso a paso la instalación Openstack sobre varios sistemas operativos; Ubuntu, Red Hat y SUSE. Cada uno de ellos describe los pasos seguidos para la instalación de todos los componentes de Openstack, sin embargo debido a que es muy posible cometer algunos errores de sintaxis durante el procedimiento, recomendamos utilizar paquetes instaladores previamente configurados tales como el proyecto FUEL patrocinado por la empresa Mirantis, disponible en http://www.mirantis.com que instala la versión más reciente disponible de Openstack utilizando la misma arquitectura y plataforma anteriormente descrita.

Luego de realizada la instalación, ingresamos a Openstack a través del Dashboard :

4.5 Actividades de laboratorio

Una vez instalada la plataforma, podemos realizar las siguientes actividades con el objetivo de familiarizarse con los comandos y el Panel de control Dashboard.

Actividad 1: Keystone 1
Objetivo: En este ejercicio vamos a configurar un equipo para que acceda a la implementación de Openstack
Requisitos: Openstack instalado y operativo, una cuenta de Administrador – Admin con acceso al componente Horizon
Procedimiento:
Usando el cliente de Línea de Comandos – CLI
Se asume que el estudiante está familiarizado con el uso de Terminal de Linux o cualquier otro cliente CLI
Instalar las variables de entorno
Antes de realizar cualquier instrucción a través del uso de los clientes API de Openstack, es necesario configurar las variables de entorno en el equipo. Las mismas que son necesarias para autenticar la sesión en el Nodo Controlador.
Los estudiantes podrán descargar el archivo openrc.sh desde el Dashboard de Horizon siguiendo la dirección IP del nodo de Control
A continuación vamos a crear usuarios utilizando los comandos en línea de keystone, por ejemplo:
Enlistamos los usuarios: keystone user-list
Y confirmamos con el detalle de usuarios mostrado en el Panel de Control.
De manera similar realizamos con los siguientes comandos:
Keystone user-create

Generar un usuario se lo realiza de la siguiente forma:

$ keystone user-create --name <nombre> --tenant-id <tenant-id> --pass <contraseña> --email <correo>

Verifique lo que ha generado en : Admin -> Proyectos -> Usuarios

Actividad 2: Administración y Cuentas
Objetivo: En esta práctica, vamos a revisar Keystone, la CLI y Administración de usuarios.
Procedimiento:
Configuración del sistema: Trabajo en OpenStack deployment. Una cuenta de administrador con acceso a horizon deployment.

Enlace a Horizon:
http://192.168.0.190/ (IP del Controlador)
Su usuario y password
Lista de Proyectos
$ keystone tenant-list
Tenga en cuenta que el Comando utiliza "tenant" en lugar de "proyecto".
Lista de usuarios: $ keystone user-list:
Agregar nuevo usuario al proyecto: $ keystone user-create
Vía Dashboard

- Admin -> Usuarios -> Crear usuario

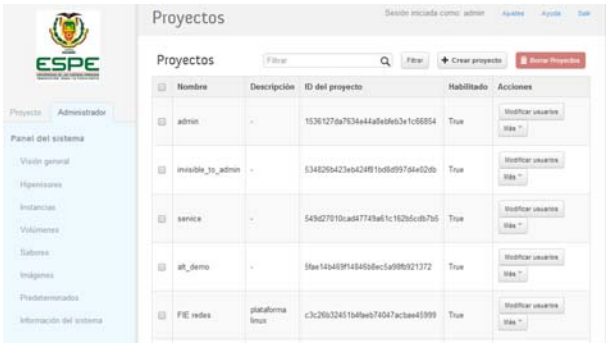
Ahora probamos a través de línea de comandos

- $ Keystone user-create --name<user-name> [--tenant-id<tenant-id>] [---pass<password>] [--email <email>] [--enable<true|false>]

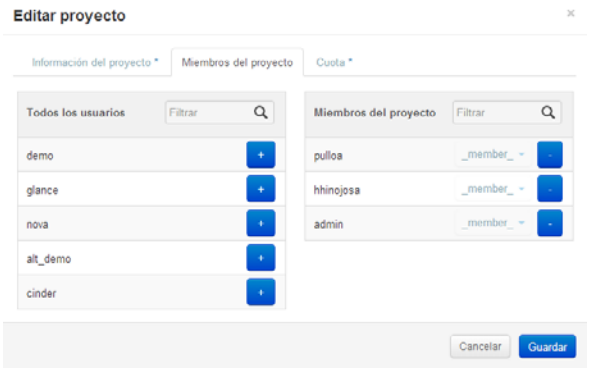
Agregar un usuario existente al proyecto Test
Vía Dashboard
Admin - > Proyectos - > Modificar usuarios

Ahora a través de línea de comandos

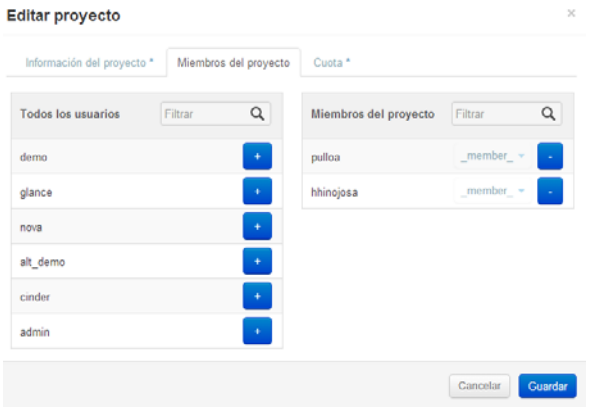
- $ Keystone user- role- add --user-id<user-id> --role-id <role-id> [--tenant-id <tenant-id>]



Vaya a Admin -> Proyectos -> Proyecto FIE redes y compruebe que el usuario está en el proyecto apropiado con el role que le corresponde



Inicie la sesión como el nuevo usuario y navegue a través del Dashboard
Confirme que el nuevo usuario tiene derechos de miembro, y no a los derechos de administrador.
Elimine el usuario de Project
Vía Dashboard eliminamos el usuario admin



A través de línea de comandos

$ Keystone user-role-remove --user-id <user-id> --role-id<role-id> [--tenant-id<tenant-id>]

Ejemplos
Lista de Proyectos Disponibles
$ keystone tenant-list

Mostrar Información específica de un proyecto
$ keystone tenant-get <tenant_id>
Ejemplo
$ keystone tenant-get
fa3700a42da8446f9d989dddbb2fbe0b

Crear un Nuevo Proyecto
$ keystone tenant-create --name <tenant-name>
Opciones
[--description <tenant-description>]
[--enabled <true|false>]

Actualizar Project Metadata
$ keystone tenant-update <tenant-id>
Opciones
[--name <tenant_name>]
[--description <tenant-description>]
[--enabled <true|false>]
Ejemplo
$ keystone tenant-update
fa3700a42da8446f9d989dddbb2fbe0b
--name Hhinojosa --description Prueba --enabled true

Enlistar los usuarios registrados
$ keystone user-list

Crear un Nuevo usuario de autenticación
$ keystone user-create --name<user-name>
Opciones
[--tenant-id <tenant-id>]
[--pass <password>] [--email <email>]
[--enabled <true|false>]
Ejemplo
$keystone user-create --name pulloa --pass patricio --email prueba@ejemplo.com --enabled true

Actualizar un usuario
$ keystone user-update <user-id>
Opciones
[--name <user-name>]
[--email <email>]
[--enabled <true|false>]
Ejemplo
$keystone user-update --name Hhinojosa --description Prueba -- enabled false
2a7081224a2a4aeab83bd6f188e1e753

Eliminar un usuario
$ keystone user-delete <user-id>
Ejemplo
$ keystone user-delete
2a7081224a2a4aeab83bd6f188e1e753

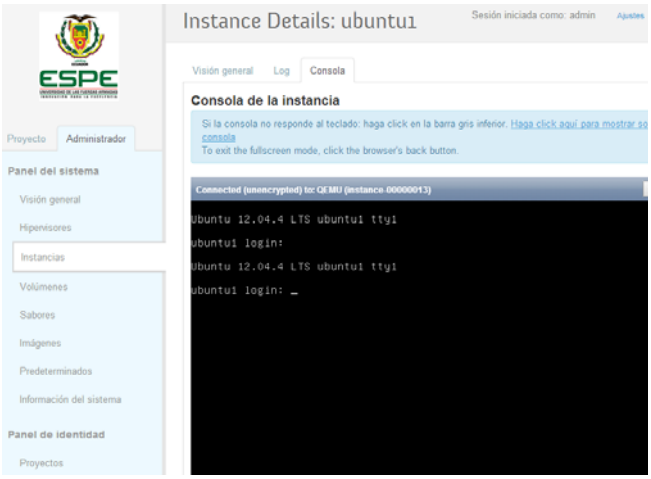
Práctica: Instancias y Migraciones
Objetivo: En esta práctica, vamos a crear las instancias desde el dashboard y la interfaz de línea de comandos (CLI). Vamos a migrar las máquinas virtuales a diferentes hosts.
Requisitos:
- Ambiente de trabajo OpenStack.
- Una cuenta de administrador con acceso a Horizon.
- Un computador configurado para acceder al ambiente de trabajo de OpenStack.

Enlace a Dashboards :
http://192.168.0.190/

Procedimiento:
Crear y migrar las instancias
En esta parte, vamos a crear instancias y migrar a diferentes hosts.

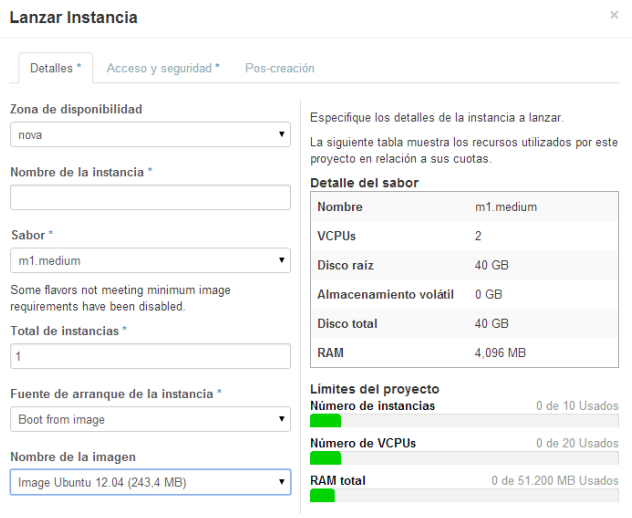
Crear un keypair
1 . Haga clic en Acceso y Seguridad
2 . Haga clic en el botón Crear keypair en la sección keypair
3 . Rellene el campo keypair name
4 . Lea la descripción del keypair para conocer su propósito
5 . Haga clic en Crear Keypair.. Tenga en cuenta que el keypair privada se ha descargado en su computadora.
Cambie los privilegios de ese archivo a 600 (por ejemplo chmod 600 keypair.pem)

Crear una instancia utilizando el Dashboard (Método # 1)
6 . Haga clic en Instancias en el panel Administrador
7 . Haga clic en el botón arranque de Instancia
8 . Seleccione la imagen como Instancia de fuente
9 . Seleccione Image Ubuntu 12.04 en el menú desplegable de imagen
10 . Escriba un nombre en el campo Nombre de instancia
11 . Elija un flavor (m1.small) en el menú desplegable
12 . En instancias de campo, tipo 1
13 . Haga clic en el enlace de acceso y seguridad en la parte superior de la ventana
14 . En la ventana, seleccione la keypair generadas en el paso 5.
15 . Haga clic en Iniciar. La instancia se generó y el estado pasará de Build a Activo.
16 . Seleccione la casilla junto a la nueva instancia y haga clic en el botón rojo Finalizar las veces.
17 .Haga clic en Finalizar instancias cuando la casilla de confirmación aparezca. Vamos a crear otra instancia en la siguiente sección.



Crear una instancia utilizando el Dashboard (Método # 2)

18 . Haga clic en Imágenes y Snapshots
19 . Haga clic en Iniciar al lado de la imagen Ubuntu 12.04 en la sección Imágenes
20 . Seleccione imagen como Instancia de fuente
21 . Seleccione Image Ubuntu 12.04 en el menú desplegable de imagen
22 . Escriba un nombre en el campo Nombre de instancia
23 . Elija un flavor (m1.small) en el menú desplegable
24 . En instancias de campo escoja, tipo 1
25 . Haga clic en el enlace de acceso y seguridad en la parte superior de la ventana
26 . En la ventana desplegable, seleccione la keypair generadas en el paso 5.
27 . Haga clic en Iniciar. La instancia se generó y el estado pasará de Build a Activo.

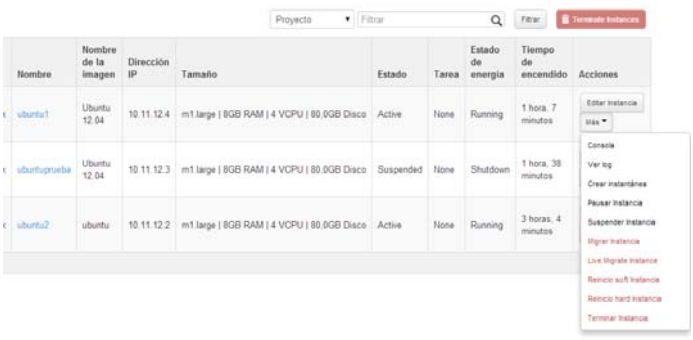


Crear una instancia (Command Line Interface)

28 . Autenticar, haga clic en Configuración en la esquina superior derecha del Dashboards
29 . Haga clic en OpenStack API
30 . Haga clic en Descargar archivo RC. Este archivo se descargará en su equipo.
31 . Escriba source openrc.sh para autenticarse.
32 . Para asegurarse de que se haya autenticado, hagamos una lista de las instancias. Escriba 'nova list'.
33 . Crear un keypair. Escriba ' nova keypair-add keypair_name> keypair_name.pem' el símbolo '>' anterior es una redirección. El comando creará la keypair y creara el archivo .pem que se utilizará cuando ingresemos por SSH a la instancia.
34 . Ahora vamos a lanzar la instancia. Escriba 'nova help boot' para ver una lista de opciones para el boot nova
35. Escriba 'nova boot –flavor m1.tiny --image<imagen ID> --key-name <keyname> <instance name>
Nota: Puede obtener el ID de imagen escribiendo ' nova image-list' y el nombre de clave, escriba ' nova keypair-list'
36. Cree una IP flotante por lo que la instancia puede ser de acceso público.
Escriba 'floating-ip-create'
37. Agregue la IP flotante a la instancia
Escriba ' nova add-floating-ip <Server name> <IP address>
La dirección IP es la IP flotante creada anteriormente. Escriba 'nova list' para obtener el nombre del servidor.
38. Conectarse mediante ssh a la instancia
ssh -i <file.pem> root @ floating_ip

Migre la instancia utilizando el Dashboard

39. Haga clic en el panel Proyecto
40. Haga clic en Instancias
41. En Acciones, haga clic en la flecha para abrir el menú desplegable
42. Seleccione Migrar
43. En el menú desplegable, seleccione un host y haga clic en Migrar.



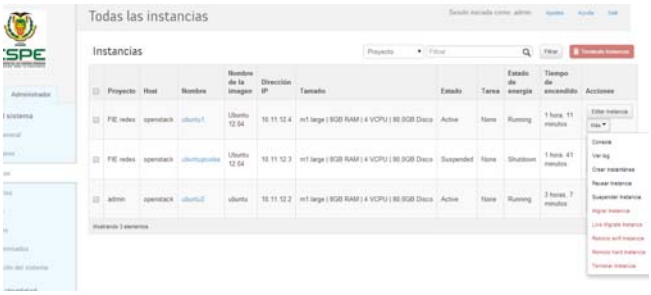
Migre la instancia mediante la CLI

44. Escriba ' nova host-list' para listar las

direcciones IP de host
45. Escriba 'nova-list' para listar el ID de instancia
46. Migre la instancia
Escriba 'nova gc-migrate --dest <destino host> <InstanceID>
Escriba 'nova list' para ver el cambio de estado de ejemplo para migrar

2. Crear instantáneas de instancias

Crear una instantánea desde el Dashboard
47. Haga clic en Instancias
48. Haga clic en botón de Crear instantánea



Crear una instantánea desde la CLI
49. Escriba 'nova list' para listar las instancias en ejecución
50. Escriba ' nova image-create <server> <snapshot_name>

V. CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

Bajo la propuesta descrita en los objetivos de nuestro proyecto, diseño e implementación de una nube computacional basados en Openstack, nuestras conclusiones son las siguientes:

Respecto al diseño:

- Openstack es un producto que tiene la flexibilidad necesaria para acoplarse a diferentes infraestructuras, en nuestro caso probamos la solución en varios ambientes de hardware y en todos los casos operó de acuerdo a lo esperado.
- En el diseño de la nube notamos que mínimo se deben considerar dos segmentos de red para su operación porque separa su red de administración de la de acceso público.

Respecto a la implementación.

- Openstack opera en plataformas de 64 bits exclusivamente, probamos con un nodo de 32 bits pero no fue posible agregarlo a la plataforma.
- Los costes de implementación de las pruebas se concentraron en el hardware ya que todas las aplicaciones utilizadas durante el proyecto son de Código Abierto.
- La implementación de la plataforma mediante el uso de del paquete FUEL resulta más sencilla de realizar y el soporte brindado por la comunidad hace que su mantenimiento sea constante.

Aspectos generales:

- La curva de aprendizaje para implementar es demasiado larga con respecto a otras soluciones de Código Abierto debido, fundamentalmente, a los continuos cambios en la documentación de soporte que brinda la Comunidad, en algunos casos llegamos a identificar que publicaban los manuales a diario.

5.2 RECOMENDACIONES

Siendo Openstack una solución soportada por un Comunidad con amplio espectro y fruto de nuestro contacto con soluciones de Código Abierto en general, hemos querido que nuestras recomendaciones hagan mención de la experiencia resultante de la investigación e implementación de la plataforma realizada por Nosotros, pero también agregando algunas observaciones realizadas por Miembros de la Comunidad que aportan y enriquecen las mismas. Recomendaciones realizadas bajo la óptica de los errores más frecuentes al momento de realizar una implementación cuyo objetivo apunta prevenirlos, identificarlos y de alguna forma a resolverlos.
Los errores más frecuentes han sido determinados por los desarrolladores y por los usuarios de Openstack con experiencia desde las versiones iniciales, en nuestro caso desde la versión Essex. Hay que considerar que antes del proyecto Horizon, todo el ciclo de vida de las Máquinas Virtuales se administraba mediante comandos CLI usando principalmente el RESTful API de NOVA para su control, por lo que había una alta posibilidad de equivocarse ya sea por motivos de sintaxis de la instrucción o por desconocimiento a fondo de los comandos disponibles. Luego ya con el desarrollo de los otros proyectos, como por ejemplo Dashboard de Horizon, los puntos de posibles fallos se trasladaron originalmente ubicados en la operación y manejo hacia el proceso de instalación ya que el incremento en la cantidad de proyectos la hacían más compleja.
Sintetizando nuestra experiencia así como también lo mencionado por CloudScaling (Bias, State of the Stack, 2013), Mirantis y la Comunidad de Openstack en sus respectivos portales de

consulta y soporte, realizamos las siguientes recomendaciones:

Openstack tiene Licencia de tipo Open Source y gratuito, por lo tanto no hay que preocuparse de los presupuesto. Se recomienda en este caso considerar aspectos tales como la infraestructura sobre la que se instala el programa, así como los constes de mantención de la misma. No por ser gratuito debe ser inestable o inseguro, siendo estos dos aspectos los de mayor atención en el software libre, hay que dedicar el presupuesto necesario para realizar las pruebas que garanticen un el paso previo hacia la puesta en producción definitiva. Hay que considerar también presupuestos para cubrir los costes de la migración de la plataforma anterior hacia Openstack cualquiera que esta haya sido y luego los de operación y mantenimiento, más aún si queremos una arquitectura de ALTA DISPONIBILIDAD.

- *Hágalo Usted mismo*. Openstack es un proyecto en permanente evolución y desarrollo, cada 6 meses surge una nueva actualización o versión renovada que algunas veces incluye un componente totalmente nuevo, como por ejemplo para la versión Icehouse, planificada para Abril del 2014, se propone la integración de los componentes Ceilometer, Heat y Marconi. Por lo que se recomienda al momento de realizar una implementación de Openstack, el apoyo de un socio estratégico especializado y con mayor experiencia en este tipo de proyectos que colabore con los recursos operativos y logísticos que permitan concretar el objetivo de manera efectiva y a tiempo. Conforme ha ido evolucionando Openstack, algunos miembros de la Comunidad de Desarrolladores han formado empresas dedicadas a la implementación y soporte de la solución, otro ejemplo es la alianza *Openstack Foundation* que agrupa todas las compañías del sector privado que apuestan comercialmente por Openstack principalmente ofertando sus propios procesos de instalación y soporte. Entre los más conocidos están PistonCloud, RedHat, Canonical, Mirantis y CloudScaling,
- *Terminología, todos la entienden*: Openstack es el resultado de la permanente colaboración de una Comunidad desplegada a nivel mundial, la Openstack Community. Cada uno de los servicios de Openstack se desarrolla bajo la dirección de un Coordinador que se encarga de gestionar desde el modelo de programación, las pruebas de funcionamiento previas y la difusión del resultado final hacia el resto de la Comunidad. Sin embargo la adopción de Openstack por parte de las empresas involucradas en el negocio de las Tecnologías de la Información, entre las que podemos nombrar a las orientadas al procesamiento como por ejemplo Intel; de networking y almacenamiento, entre ellas Intel, HP, Cisco, NetApp, y otras más que han generado su propios para referirse a un mismo tema, un ejemplo es que PistonCloud y CloudScaling definen como *Project* a los que la Comunidad Openstack define como *Tenant*. Esta diversidad de términos obliga un esfuerzo extra al momento de investigar sobre el funcionamiento de una u otra solución, por lo que recomienda ir creando un glosario de términos relacionados.
- *Los sistemas anteriores son incompatibles*. Siendo Openstack considerado como un middleware, puede instalarse sobre cualquier infraestructura existente. La flexibilidad de Openstack le permite iniciar con una implementación pequeña y luego ir aumentado los nodos de procesamiento, almacenamiento o de cómputo conforme sea el requerimiento de los usuarios. Esta escalabilidad hace que Openstack pueda operar a la par de los sistemas anteriores, ya sean en hardware o las plataformas visualizadas, por lo que recomendamos realizar las pruebas de la implementación usando los equipos existentes para asegurar su funcionamiento previo a la adquisición de infraestructura nueva siempre que cumpla por lo menos con el requisito de la virtualización.
- *Todo los que se necesita es un balanceador de carga*: Las exigencia de cada vez mayores por parte de los usuarios hacia la nube gracias al incrementos de las soluciones de tipo SaaS y PaaS, generan nuevos retos para los encargados de la operación y exponiéndolos a nuevos retos y situaciones que requieren un análisis bajo una perspectiva diferente a la tradicional en donde, anteriormente, el hardware era el punto de falla más probable. En cambio la nube cuenta con varios componentes de software que orquestan la operación en general, por lo que se recomienda considerar al reemplazo o incremento de hardware como una de las últimas fases de la solución.
- *No es necesario contactar a los desarrolladores:* El error más frecuente cuando se realizar una implementación de una solución bajo código abierto es la falta de un canal de contacto con los desarrolladores de la misma, por lo que recomendamos involucrarse de una u otra manera en el desarrollo de OpenStack de tal manera que sea posible reportar bugs o fallas en la operación y que ayuden a los programadores a generar su inmediata reparación con el objetivo de conseguir un sistema más estable y confiable. Así mismo la Openstack Community incluye a muchas personas

listas a resolver cualquier duda que ayude a realizar una instalación menos complicada.

- *No hay que especializarse, al final es solo Linux:* Openstack puede instalarse en distintas plataformas Linux, entre ellas Ubuntu, Red Hat y Suse, sin embargo su programación está realizada en Python por lo que se recomienda un conocimiento básico previo, tanto del entorno de operación así como del entorno de programación con el objetivo de realizar las modificaciones al código que permitan afinar el sistema de acuerdo a cada necesidad. En nuestro caso la elaboración del presente Proyecto significó un reto no solo con respecto a la aplicación de Openstack sino que también nos impulsó al estudio de Linux y MySQL.
- *El ROI de la nube computacional es rápido*: el término Cloud Computing está de moda en todos los niveles de la sociedad de las Tecnologías de la Información, sin embargo hay que considerar muy bien el objetivo previo antes de migrar los servicios de una red convencional hacia la nube, siendo la fiabilidad la características más interesante desde el aspecto económico, hay que recordar que se pueden crear máquinas virtuales sin necesidad de tener una nube computacional por detrás por lo que se recomienda analizar extensivamente la relación de costos vs. beneficios previos a la realización de cualquier inversión.
- *La Nube se mantiene y repara por sí sola*. El nivel de automatización y la fiabilidad de una nube debe ser un aspecto importante a considerar, si bien es cierto que la nube puede configurarse bajo la modalidad HA, Alta Disponibilidad, de tal manera que si se presenta algún inconveniente en uno de sus componentes, automáticamente otro nodo ingrese al sistema y el servicio continua sin ser afectado, sin embargo Openstack es un proyecto al que permanentemente se le agregan nuevos componentes que incrementan su complejidad , por lo que se recomienda implementar un cronograma de entrenamiento continuo de los encargados de su administración.
- *Equivocarse no es opción:* Los cambios generalmente tiene obligatoriamente una curva de aprendizaje, más aún cuando el cambio incluye tecnologías en permanente desarrollo por lo que las probabilidades a equivocarse son mayores. En nuestro caso tuvimos que dedicar más tiempo a la instalación del sistema de lo que teníamos planificado, justamente debido a las modificaciones diarias que habían en los manuales de instalación con respecto a las versiones de los componentes disponibles en los repositorios de Canonical. Por tal motivo recomendamos llevar una bitácora interna que referencie las modificaciones realizadas durante los diferentes procesos de instalación de Openstack.

Ya para finalizar nos preguntamos entre nosotros si recomendaríamos el uso de Openstack y nuestra respuesta es afirmativa pero considerando los puntos anteriormente mencionados.

VI. REFERENCIAS BIBLIOGRÁFICAS

Babcock, C. (1 de 12 de 2013). *InformationWeek.* Obtenido de http://www.informationweek.com/infrastructure/cloud-infrastructure/gartner-50--of-enterprises-use-hybrid-cloud-by-2017/d/d-id/1111769?

Bakken, D. (2009). *MIDDLEWARE*. Obtenido de Washington State University: http://www.eecs.wsu.edu/~bakken/middleware.htm

Bias, R. (2013). *Cloud Scaling.* Obtenido de http://www.cloudscaling.com/resources/

Bias, R. (09 de 11 de 2013). *State of the Stack.* Obtenido de Openstack Foundation: http://www.youtube.com/watch?v=S8Q8ASLQ3t8

Buyya Rajkumar, J. B. (2011). *Cloud Computing - Principles and Paradigms.* Wiley.

Cisco. (2013). *Cisco Global Cloud Index: Forecast and Methodology, 2012–2017.* Obtenido de http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns1175/Cloud_Index_White_Paper.html

ClusterLabs. (2013). *Peacemaker*. Obtenido de http://clusterlabs.org/

Comisario, A. (2013). *DataCenter Dynamics*. Obtenido de http://www.youtube.com/watch?v=4oqm3IQNoKc

IBM. (2009). *IBM Perspective on Cloud Computing. The 'next big thing' or 'another fad'?* Obtenido de http://www.itcio.es/cloud-computing

IBM. (2010). *IBM DeveloperWorks*. Obtenido de https://www.ibm.com/developerworks/

IBM. (s.f.). *Cloud computing for the enterprise: Part 1: Capturing the cloud*. Obtenido de Cloud computing for the enterprise: Part 1: Capturing the cloud: http://www.ibm.com/developerworks/websphere/techjournal/0904_amrhein/0904_amrhein.html

Jackson, K. (2013). *Openstack Cloud Computing Cookbook.* PACKT Publishing.

KVM. (2013). *Kernel Based Virtual Machine*. Obtenido de http://www.linux-kvm.org/page/Main_Page

Mirantis. (2013). *Mirantis Openstack reference Architecture.* http://www.mirantis.com.

Nurmi, D. (2009). *https://www.eucalyptus.com/.* Obtenido de https://www.eucalyptus.com/

Openstack. (2013). *Hypervisor configuration basics.* Obtenido de http://docs.openstack.org/trunk/config-reference/content/hypervisor-configuration-basics.html

Openstack. (2013). *Openstack Org*. Obtenido de http://openstack.org

Openstack. (2014). *Compute Node*. Obtenido de http://docs.openstack.org/training-guides/content/associate-computer-node.html

Peter Mell, T. G. (2011). *National Institute of Standars and Techonology.* Obtenido de http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

Pipe, K. (s.f.). *Openstack Grizzly.* http://www.slideshare.net/emaganap/open-stack-overview-meetups-oct-2013.

Project, P. R. (2010). *The future of the Internet.* Obtenido de http://pewinternet.org/Reports/2010/The-future-of-cloud-computing.aspx

RAE. (26 de 10 de 2013). *Real Academia de la Lengua Española*. Obtenido de http://buscon.rae.es/drae/srv/search?val=paradigma

Swiftstack. (2013). *https://swiftstack.com.* Obtenido de https://swiftstack.com/openstack-swift/architecture/

Tholeti, B. (2011). *Hypervisors, virtualization, and the cloud: Dive into the KVM hypervisor*. Obtenido de http://www.ibm.com/developerworks/cloud/library/cl-hypervisorcompare-kvm/index.html

Tom Fifield, Diane Fleming. (2013). *Openstack Operations Guide.* O´REILLY.

Wikipedia. (2013). Obtenido de http://es.wikipedia.org/wiki/Token_(inform%C3%A1tica): http://es.wikipedia.org/wiki/Token_(inform%C3%A1tica)

Wikipedia. (2013). Obtenido de http://es.wikipedia.org/wiki/Remote_Procedure_Call

Wikipedia. (2013). *Wikipedia.* Obtenido de http://es.wikipedia.org/wiki/Complemento_(inform%C3%A1tica)

Wikipedia. (s.f.). *virtualización*. Obtenido de http://es.wikipedia.org/wiki/Virtualizaci%C3%B3n

Wikipedia, W. (2013). *Hipervisor*. Obtenido de http://es.wikipedia.org/wiki/Hipervisor

WSGI.org. (2013). *http://wsgi.readthedocs.org/en/latest/what.html*. Obtenido de http://wsgi.readthedocs.org/en/latest/what.html