Automatización de despliegue de instancias para aplicaciones y ambientes virtuales usando OpenStack y Ansible.

Area de especialización: Administración de redes y de sistemas operativos

Estudiante: Manuel Rodríguez Hernández

Consultor: Jordi Massaguer Pla

Tutor de Practicas Externo: Marco Antonio García Ortega

Enero 07 2015

Resumen

El contenido de este documento ilustra como realizar el despliegue de recursos informáticos en la nube para múltiples propósitos como aplicaciones, bases de datos y contenedores. Para lo cual se empleara Ansible como herramienta de automatización de configuraciones y Openstack para controlar la infraestructura de cloud.

El motivo de la selección de este proyecto es comprehender como interactuán estas tecnologías, no solo porque son empleadas cada vez mas en el rubro del cómputo distribuido, sino también porque son plataformas de gran importancia en la empresa donde laboro: Rackspace Inc. Sobre todo Openstack ya que es utilizado como base de nuestra infraestructura de *cloud computing*, por lo tanto es cada vez mas indispensable estar familiarizado con estos servicios a fin de llevar a cabo implementaciones de diversos sistemas, brindar soporte, y desplegar recursos informáticos en tiempos menores.

Tabla de Contenidos

Introducción

Actualmente se cuentan con diversos programas que ayudan a desplegar infraestructura informática remota accesible a través de Internet, o dicho de otro modo Infraestructura de *cloud computing* “computo en la nube”, si bien desde hace tiempo se disponía de la tecnología para administrar recursos en linea aunque no se le denominaba de esta forma, seguramente porque eran recursos mas limitados, estáticos no tan fáciles de escalar, pero con el paso del tiempo el termino se ha vuelto común ya que se intenta describir la gran aglomeración en Internet de recursos lógicos o físicos de los que se disponen para lograr la automatización, el despliegue y el crecimiento gradual (escalabilidad) con base a las necesidades de la carga de trabajo; en otras palabras se usan recursos informáticos a la medida de las necesidades que lo exijan.

Bajo este contexto las empresas que dependen de equipo de cómputo y de acceso a Internet para promocionar u ofrecer sus negocios o servicios les viene muy bien este modelo, ya que en lugar de adquirir equipo y mantener instalaciones para su funcionamiento prefieren alquilarlo por un tiempo y disminuir o aumentar su parque informático con base a la necesidades o crecimiento de su negocio, esto sin duda también representa un mercado muy importante para muchas empresas que invierten capitales en la construcción de centros de datos que cuenten con todos los elementos tecnológicos, y así poder ofrecer servicios de renta de recursos informáticos. Según estudios de la firma Gartner[1] el mercado del cloud computing represento 131 billones de dolares en el año 2013 y se espera que siga creciendo, esto ademas de representar enormes dividendos, también significa que muchas empresas están participando en el mercado ofreciendo servicios en torno a los servicios de la nube.

Tal es el caso de los proyectos Openstack, Ansible, Docker, MongoDB, que se mencionaran a lo largo de este documento. En el caso de Openstack es uno de los principales sistemas para desplegar IaaS (Infraestructura as a Service) tales como Eucalyptus y OpenNebula, es open source (se distribuye bajo licencia Apache 2.0) cuenta con una gran comunidad de desarrollo y es respaldado por una fundación compuesta de empresas como: Rackspace, RedHat, SUSE, HP, Yahoo, Cisco por mencionar algunos. Ansible por su parte se destaca entre las herramientas de despliegue de configuraciones como Puppet y Chef por su facilidad de uso y por no emplear agentes remotos para llevar acabo sus tareas, se distribuye bajo GPLv3 y cuenta con una versión empresarial. En cuanto Docker es una plataforma que permite construir el ambiente y los componentes para aplicaciones de forma rápida, es portable ya se ejecuta en una gran variedad de sistemas, permite aislar y separar recursos en unidades llamadas contenedores valiéndose de características del kernel, y se distribuye bajo licencia Apache 2.0. Por ultimo MongoDB es un sistema de administración de bases de datos basado en documentos JSON y es conocido por no usar tablas relacionales y lenguaje SQL para su explotación, cuenta con licencia AGPL v3 y se caracteriza por sus ventajas en la escalabilidad y replicación.

1 Rob van der Meulen and Janessa Rivera, (2014, Febrero 28) Garnet says Worldwide Public Cloud Services Market to Total $131 Billion. Gartner [Articulo en linea]. [Fecha de consulta: Septiembre 28 2014] <http://www.gartner.com/newsroom/id/2352816>

Como se puede observar la mayoría de estas herramientas busca la automatización de tareas, el fácil despliegue, se valen de las ultimas tecnologías para su implementación, sus modelos de negocio están respaldados por una comunidad de open source, y entre ellas mismas generan un ecosistema que les permite interactuar entre si para hacer frente a los retos informáticos de hoy en día y aprovechar el auge de la burbuja tecnología que representa el *cloud computing*.

Objetivos

- Comprehender como funcionan los sistemas que se instalaran durante el proyecto.

- Generar una guía de instalación de estos servicios que contendrá las configuraciones básicas, y los pasos a seguir para la implementación de Openstack

- Automatizar las tareas de instalación y configuración a partir del momento que se comprenda como interactúan los diferentes elementos que la componen.

- Una vez listo el sistema de base desplegar sistemas que se usan en la actualidad como una replica de mongo, y contenedores basados en Docker.

- Mantener un sistema de control de versiones para documentar las tareas de automatización, esto ayudara a tener un registro de los cambios a lo largo del proyecto.

Estructura de la Memoria

En el primer capitulo se definirán los requisitos del sistema, y las herramientas a emplear para su configuración, después se elaborara un plan de la arquitectura de Openstack y de los servicios que se instalen a fin de proveer una vista gráfica de la implementación antes de continuar con los detalles técnicos.

En los siguientes capítulos se incluirán las fases de la implementación con sus respectivas explicaciones, las pruebas realizadas y se hará referencia a los anexos donde se encontrara las configuraciones y tareas para llevar a cabo la implantación, los temas a abordar serán:

- Análisis de los requisitos y la viabilidad
- Implantación y diseño
  - Arquitectura.
  - Despliegue de Openstack con Ansible
  - Creación de equipos virtuales y administración básica.
  - Implementación de una replica de base de datos con MongoDB
  - Implementación de contenedores de Docker.
- Conclusiones y resultados

Al final se hará referencia a la base bibliográfica empleada, enlaces al los repositorios de los scripts y recursos gráficos empleados para la comprensión del funcionamiento.

Análisis de los requisitos y de la viabilidad

El objetivo del proyecto es automatizar el proceso de instalación de Openstack a través de un sistema de despliegue de configuraciones, adicionalmente se busca instalar aplicaciones y servicios de alta demanda por los usuarios en las instancias virtuales, actualmente estos procesos se llevan a cabo manualmente o en algunos casos a través de scripts en bash, por lo tanto contar con un sistema de automatización de configuraciones seria de gran ayuda, aunque no se tiene planeada la entrada en producción aun, se pretende entender como funciona el sistema para determinar los pasos que tendrán que seguirse a fin de poder usar este proyecto como estándar para despliegues en producción en el futuro.
Bajo este contexto en los siguientes párrafos se describirán los requerimientos del proyecto y las definiciones de los servicios que serán empleados durante el diseño del sistema

Requisitos del sistema

- Instalación de Openstack automatizada con una herramienta de despliegue de configuración como Puppet, Chef, Ansible, etc.
- Instalación de base siguiendo las mejores practicas en la documentación oficial del proyecto de Openstack.

- El resultado de la automatización (scripts) deben estar documentado a fin de entender las partes que lo componen, de igual forma se deben seguir las mejores practicas para su elaboración, es decir deben ser fáciles de entender, simples y funcionales.

- Desplegar una replica de MongoDB de la misma forma, al menos tres instancias virtuales, el despliegue se hará con la herramienta de automatización.

- Desplegar Docker en una instancia virtual y crear contenedores, de igual forma se debe de automatizar lo mas posible la configuración.

- El sistema donde se instale Openstack debe permitir usar la ultima versión de este software, y el sistema que usaran las estancias virtuales de preferencia debe ser una distribución de clase empresarial o su versión publica disponible.

Valoración y elección de las soluciones

Openstack fue elegido ante todo ya que es el software que usa la empresa en su negocio de cloud computing, ademas Rackspace participa activamente en la fundación Openstack por lo tanto no se consideraron otras herramientas de Infraestructura de cloud.

En cuanto a la herramienta de automatización de configuraciones, se valoro Chef y Puppet ambas son muy estables, cuentan con una versión empresarial y Open source, sin embargo su manipulación requiere un poco mas de tiempo para entenderlas, por lo tanto se eligió Ansible ya que en cuanto simplicidad es la mas fácil de emplear, no requiere de agentes para

su funcionamiento, es decir solo requiere acceso vía protocolo SSH en los equipos a configurar y librerías estándar de Python, ademas Rackspace cada vez opta mas en usar esta herramienta ya que cuenta con apoyo de la empresa que lo desarrolla.

En cuanto a la elección de MongoDB se debe a que es un sistema que nuestros clientes están mas interesados en su despliegue, por lo tanto no se comparo con otros productos.

Definición del Sistema

- La herramienta de automatización de configuraciones debe de auxiliarse del sistema de gestión de paquetes (i.e.: yum, apt) para poder instalar los todos los programas necesarios y sus dependencias

- El sistema de Openstack debe de contener los siguientes módulos:
    nova: gestión de nodos virtuales
    neutron: administración de red
    keystone: autenticación entre módulos
    glance: gestor de imágenes
    cinder: gestor de volúmenes de bloque.

- El despliegue de Openstack debe de permitir la creación/destrucción de equipos a través de la linea de comando.

- Los equipos virtuales deben de tener un direccionamiento que permita la comunicación entre ellos mismos, y el trafico externo, ya sea usando la dirección IP del equipo físico a través de NAT o con sus mismas direcciones si pertenecen al mismo segmento de red.

- Para esta versión no se requieren contraseñas complejas,

- El sistema físico y la imagen para crear las instancias virtuales debe de ser de una distribución con sistema base (sin consola gráfica).

- Para el sistema de MongoDB, la configuración debe de contener un sistema primario, y dos secundarios, con una instalación y configuración básica, la versión de Mongo debe ser la última estable disponible

- La instalación de Docker debe de permitir desplegar dos plataformas diferentes e independientes.

- Ansible debe de configurar Openstack a través de una serie de playbooks, esto permitirá la creación de las instancias virtuales a través de linea de comandos o llamadas API, y por ultimo otra serie de playbooks se encargara de desplegar las configuraciones de los servicios propuestos.

Especificación del plan de pruebas

Openstack
- Listado de configuraciones en linea de comando una vez terminada la instalación
- Creación y destrucción de instancias virtuales,

MongoDB
- Creación de bases de datos y documentos
- Destruir un nodo y agregar uno nuevo a la replica

Docker
- Creación de un par de instancias y colocar scripts de lenguajes diferentes en cada una.

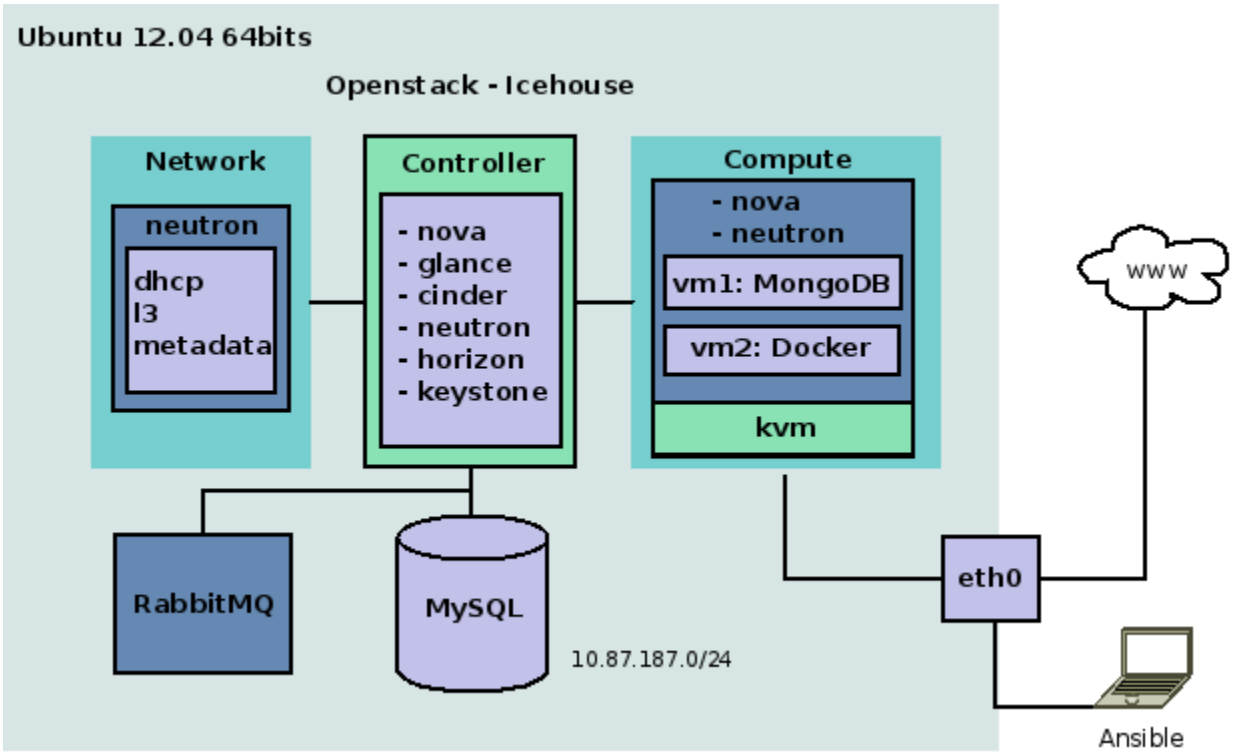
Ansible
- Clonación de repositorio remoto
- Configuración de instancias aplicando los scripts respectivos.

Diseño

Arquitectura

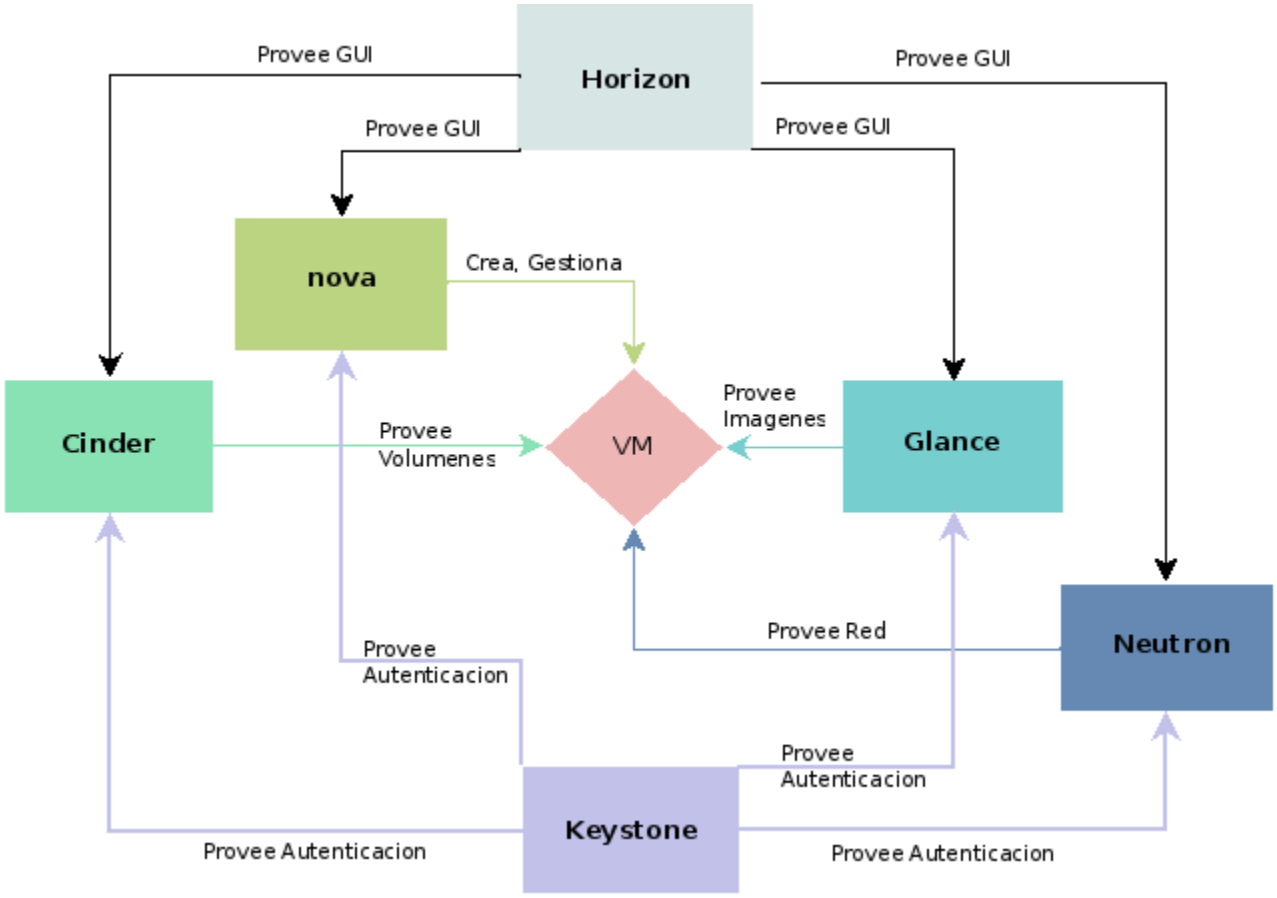
Openstack es un sistema de IaaS compuesto de varios servicios, su finalidad es proveer una plataforma para la creación de instancias virtuales, dichas instancias contienen sistemas operativos donde a su vez se pueden instalar una gran gama de aplicaciones como Instancias con administrador de contenedores (Docker) o instancias con bases de datos no relacionales (MongoDB) por citar ejemplos relacionados al proyecto.
La arquitectura recomendada de Openstack esta pensada para desplegarse en múltiples equipos debido a la complejidad de las tareas y la carga de computo, usualmente se recomienda usarse tres servidores, uno funge como un nodo maestro (controlador), un nodo para la gestión de red y otro nodo para albergar a las instancias virtuales (computo), en nuestro caso por tratarse de un sistema de pruebas se utilizará un solo servidor donde convivirán los servicios de los tres nodos, a continuación se ilustra gráficamente los componentes empleados en la solución.

Figura 1.1 Arquitectura global de un solo nodo



Los servicios de Openstack interactúan entre si, principalmente con Keystone quien provee la autenticación, Horizon es la interfaz gráfica para administrar cada uno de lo componentes por lo tanto también interactúa con todos de cierta manera. Por otra parte el resto de los servicios giran entorno al aprovisionamiento de instancias, ya sea proporcionando recursos o gestionando. La figura a continuación ilustra como se comunican cada uno de los elementos.

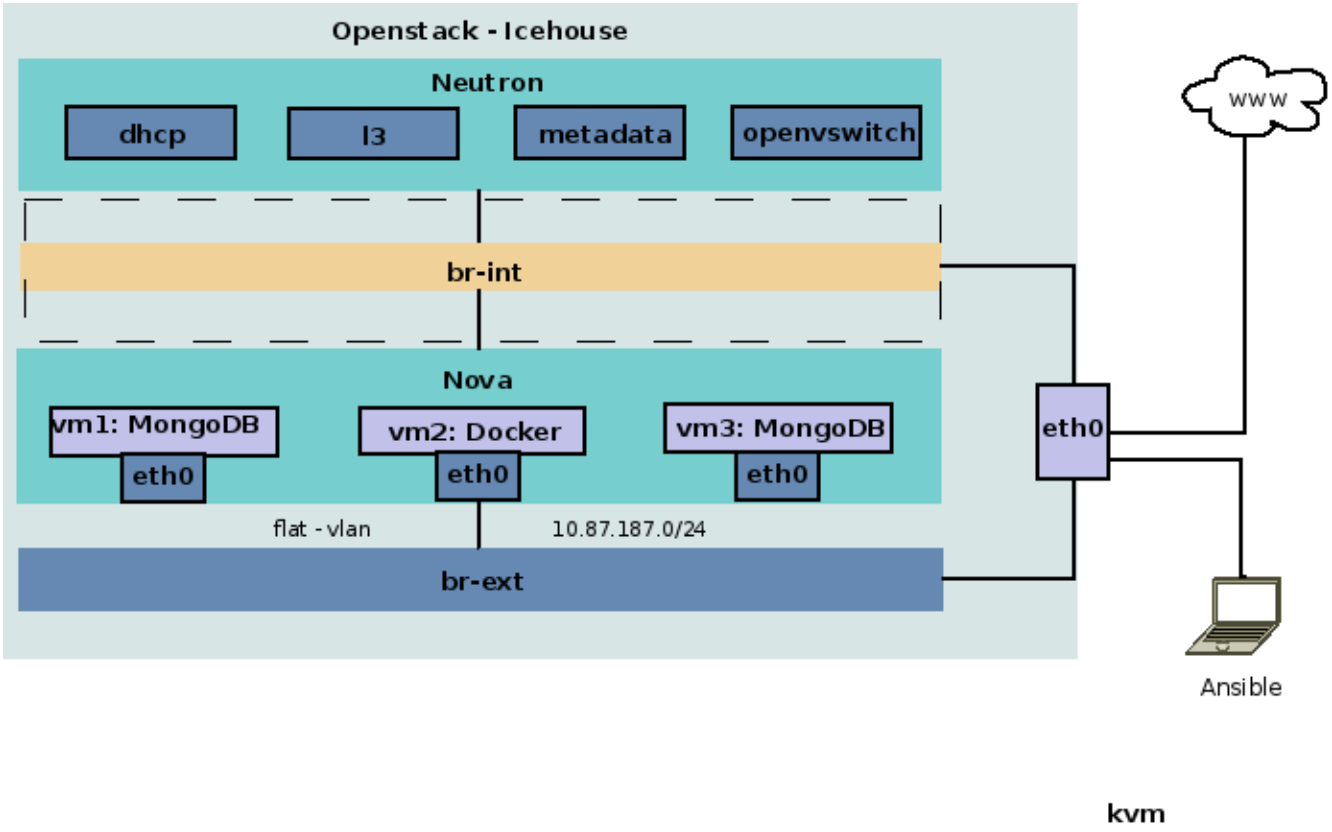
Figura 1.2 Arquitectura Lógica de componentes de Openstack



Por otra parte la arquitectura de red de la solución cuenta con varios elementos lógicos de gran importancia, al tratarse de un sistema con una sola interfaz de red se optó por un esquema de red plano es decir tanto las instancias como el servidor de Openstack tienen direcciones IP de la misma sub-red, es decir las instancias no cuentan con su propio direccionamiento interno o forman parte de una red segmentada, de haber sido de este modo se requiere de un Router lógico gestionado por Neutron para la intercomunicación entre la red de las instancias y el mundo exterior. Cabe mencionar que usar una sola interfaz de red no es recomendado ya que se vuelve un cuello de botella al pasar el tráfico de las instancias, tráfico de administración y tráfico interno entre servicios de openstack.

En este desligue cada instancia tendrá una interfaz virtual de red que se encuentra conectada a la interfaz física del equipo a través de una tercera interfaz lógica conocida como puente (bridge), esta comunicación entre interfaces es gestionada por el servicio openvswitch (plugin de Neutron). Por otra parte la asignación de direcciones IP es gestionada por el servicio de DHCP, el servicio de Metadata permite que las instancias accedan a información del servicio de computo, y el servicio de L3 provee capacidades de red como forwarding y NAT.

Figura 1.3 Arquitectura de Red



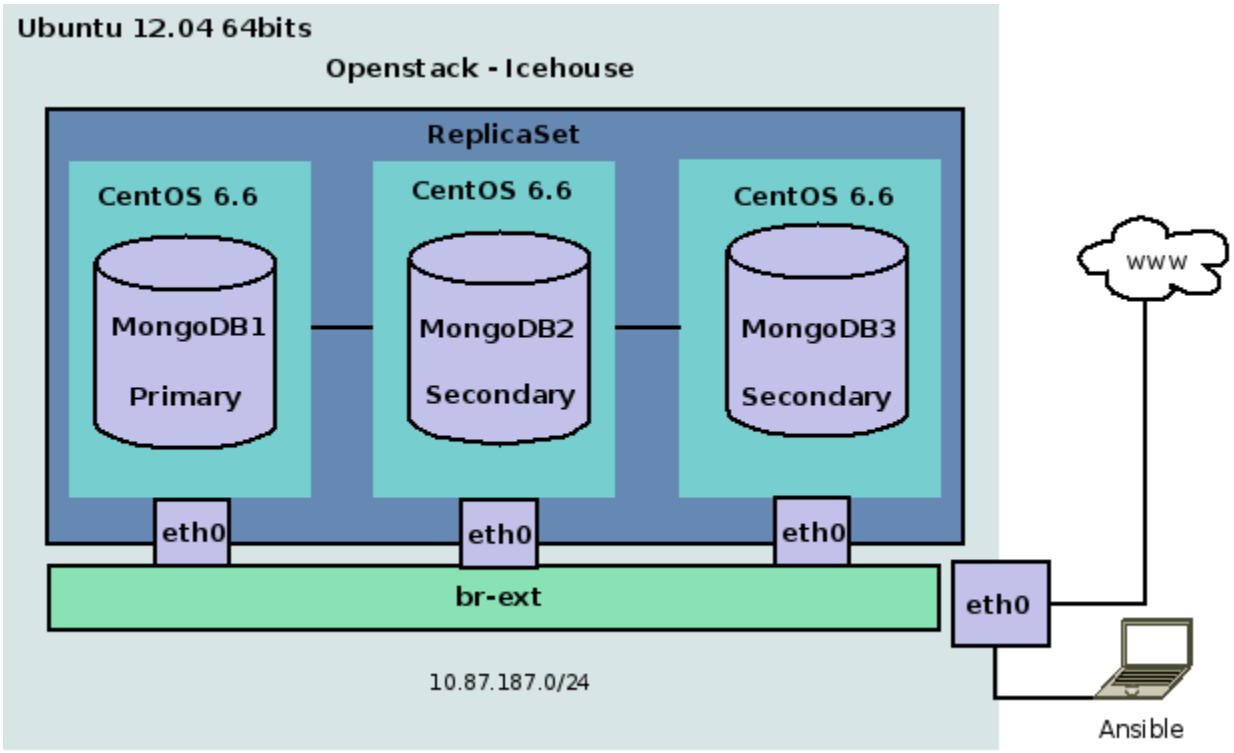
MongoDB

Una replica de MongoDB requiere al menos de tres nodos a fin de integrar el sistema de votación para elegir un nodo activo y dos nodos pasivos, la información se replica al resto de los nodos; el nodo activo es usado para escrituras a las bases de datos o documentos, el resto sincroniza los datos a fin de tener una copia actualizada. En este despliegue se usaran tres instancias virtuales con las siguientes características:

- 1 vcpu
- 1GB RAM
- 20GB HD
- 1 interfaz de red
- CentOS6.6

Cada una de las instancias tendrá una dirección IP del mismo segmento y debe de poder comunicarse con el resto, sobre todo permitir trafico de los servicios de MongoDB, la imagen siguiente describe los elementos que componen la replica como instancias de Openstack.

Figura 1.4 Diagrama de bloques de la replica de MongoDB



Docker

La arquitectura de Docker es muy similar a la arquitectura de un Hypervisor, es decir ambos proporcionan una capa de software que permite la creación y administración de instancias solo que Docker se vale de funcionalidades en el Kernel para aislar recursos llamadas Cgroups y colocar contenedores, a diferencia de un Hypervisor no provee de la emulación de hardware sino de librerías, es mucho mas ligero que un Hypervisor. Una representación gráfica de lo anteriormente comentado corresponde a la siguiente.

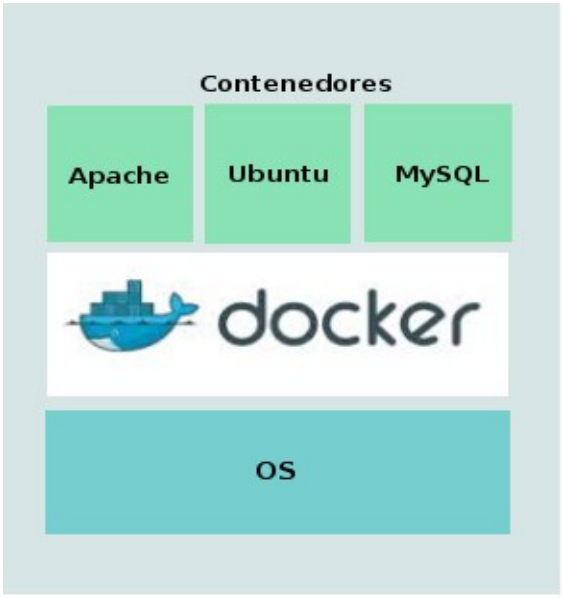
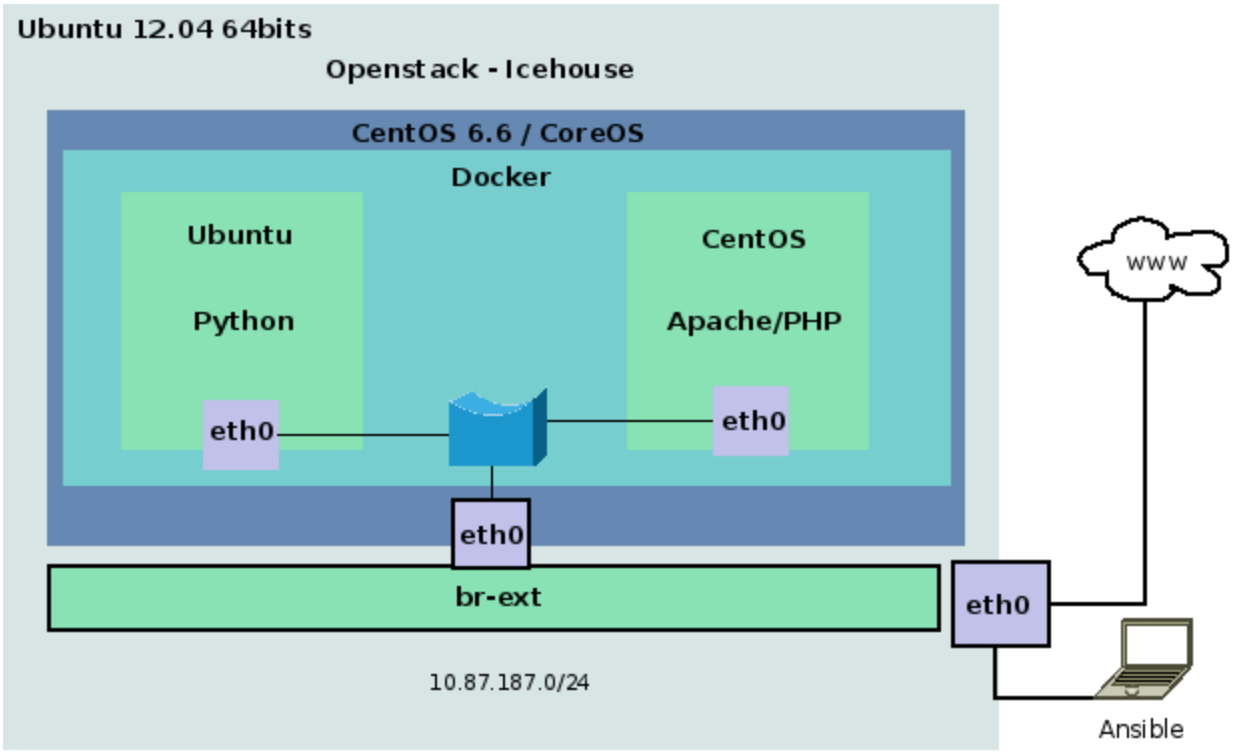


Figura 1.5 Arquitectura de Docker

Openstack puede usar Docker como hypervisor hasta cierto punto (asi como Qemu o KVM), aunque en esta solución se instalará el software en Instancias virtuales y dentro de ellas se desplegaran los contenedores, estos tendrán acceso a la red a través de puentes (bridges), se usaran instancias con CentOS o CoreOS conocido por su integración con Docker, dentro de cada instancia se desplegaran contenedores, estos serán descargados del registro publico o bien una imagen a la medida de un registro privado. En la imagen siguiente se representa la configuración mencionada.

Figura 1.6 Diagrama de bloques de la solución de Docker.



Identificación de subsistemas

Openstack

Openstack es un conjunto de servicios que permiten la administración de infraestructura de cloud, por ejemplo desde equipos virtuales, bloques de almacenamiento, creación de imágenes, administración de red, así como equipo físico, sobre todo equipo de red, los principales elementos que lo componen y que serán usados en este trabajo son los siguientes:

- Compute (nova): Software para crear y gestionar equipos virtuales en hardware estándar a nivel masivo.
- Volumes (cinder): Software para crear volumenes de almacenamiento y agregarlos a las instancias virtuales.
- Identity (keystone): Software para almacenamiento de gestión de credenciales e identificación entre servicios
- Image service (glance): Software para la administración de imágenes de equipos virtuales.
- Networking (neutron): Software para la creación, administración y configuración de funciones de red como direccionamiento, control de acceso y enrutamiento.
- GUI (Horizon): Interfaz de administración para la gestión de los servicios.

El software mencionado esta basado en Python, se instala a través de múltiples fuentes como gestor de paquetes, código fuente, o PIP, cada modulo se auxilia de archivos de configuración en texto, y una base de datos para mantener información del servicio, en este caso se empleo MySQL y un sistema de envío de mensajes entre servicios RabbitMQ

Toda comunicación u operación entre servicios es autenticada y autorizada por Keystone, la arquitectura esta diseñada para usar múltiples equipos es decir cada modulo en su propio equipo físico o al menos dos o tres equipos físicos separando los servicios de los servidores virtuales, pero para este caso practico se emplea un solo equipo físico[2] donde se concentrará todos los elementos de la solución.

Ansible

Ansible permite la automatización de configuraciones a través de lineas de código llamadas playbooks, en un playbook usualmente se agregan cada una de las tareas que se desean ejecutar en los equipos remotos, esta basado en lenguaje Python y usa protocolo SSH para conectarse y ejecutar tareas, consta de una serie de módulos que sirven de base para evaluar la instrucciones de automatización, por ejemplo módulos de gestión de paquetes, de servicios, de edición de archivos, de linea de comando, etc. Ansible es portable en la mayoría de las distribuciones de GNU/Linux, y la configuración inicial y uso es simple, para este proyecto la instalación y configuración se efectuaran desde una distribución GNU/Linux con Fedora 20

2 Las características físicas del equipo podrán verse al final del documento en el anexo B

Gestión de configuraciones con GIT

Para llevar un control de cambios y revisión de configuraciones se opto por usar GIT, y particularmente el sitio de github.com a fin de tener un visor gráfico en linea que permita llevar un historial de cambios, restauración de cambios, y seguimiento de las tareas de automatización, GIT usa SSH para conectarse al repositorio donde se almacenan los datos, en el equipo cliente basta con instalar el paquete git, usar la configuración de base y agregar una llave privada previamente generada como método de autenticación.

MongoDB

MongoDB es una base de documentos conocida por su facilidad de escalar, alta disponibilidad y desempeño. En realidad una base de datos es una colección de documentos y un documento es un conjunto de valores id:valor, no esta basada en SQL, y tiene su propia forma de ser empleada a través de consultas.
Por defecto el servicio usa el puerto 27017/tcp y es posible replicar los datos entre nodos con una configuración muy sencilla. Al estar en modo de replica todos los nodos contienen una copia de los datos y en caso de falla de un nodo habrá una votación para que otro equipo que tome el rol primario sin interrupción de servicio. En este caso se empleará la ultima versión estable 2.6 del repositorio oficial de MongoDB.

Docker

Es una plataforma que permite desarrollar, crear y ejecutar aplicaciones permitiendo ensamblar los componentes, es decir el código de aplicaciones puede ser desplegado en un contenedor sin importar la plataforma donde este el contenedor, en tanto el contenedor contenga lo necesario no habrá fricción con los componentes de Hardware y software del equipo físico, esto además provee seguridad al aislar completamente una aplicación del resto de los recursos. Docker usa imágenes que contienen un sistema operativo, aplicaciones y código que al desplegarse crean contenedores, y de una sola imagen se pueden crear múltiples contenedores, agregarle mas elementos, crear una nueva versión a partir de los cambios, etc. Dichas imágenes se almacena en repositorios llamados registros públicos o privados y pueden ser administrados a través del Docker hub, que es un sitio publico que provee de Hosting de imágenes, permite la autenticación de usuarios y esta integrado con Git a fin de proporcionar un sistema de seguimiento de versiones.[3]

3 Getting Started with Docker Hub – Docker Documentation [Articulo en linea]. [Fecha de consulta: Diciembre 18 2014] https://docs.docker.com/userguide/dockerhub/

Desarrollo

Configuración inicial de Ansible

1. Instalación de Ansible en equipo cliente.

```
# yum install ansible.
```

2. Crear inventario de equipos donde se aplicaran cambios

File: /etc/ansible./hosts

```
# Openstack
[controller]          <-- nombre del grupo de equipos
IP_Controller         <-- dirección IP del equipo remoto
```

Al no requerir agentes en los equipos clientes Ansible. utiliza al protocolo SSH para inyectar las instrucciones, por lo tanto se requiere contar con una cuenta de usuario para ingresar al equipo y se remienda configurar la autenticación a través de llaves asimétricas a fin de eliminar la constante solicitud de credenciales, sobre todo en casos donde se cuenta con muchos equipos y se hacen cambios continuamente.

3. Creación de usuario y contraseña en equipo a configurar

```
# adduser usuario
# passwd usuario
```

4. Generación de llaves

```
$ ssh-keygen
```

5. Agregar llave publica en el equipo cliente (ansible server)

```
$ ssh-copy-id -i .ssh/public.key usuario@IP_equipo_remoto
```

Por ultimo en el equipo a configurar se recomienda asignar privilegios de root a la cuenta de usuario a fin de poder ejecutar tareas de instalación.

6. Configuración de sudo

```
$ visudo
 usuario        ALL=(ALL)       NOPASSWD: ALL
```

Listo, Ansible. tiene todo lo necesario para conectarse al equipo mientras sea alcanzable.

7. Prueba de acceso

```
$ ansible all -m ping -u usuario --sudo
```

Configuración de GIT

1. Instalación y configuración en el equipo cliente

```
# yum install git

$ mkdir local.repo
$ cd local.repo
$ git config --global user.name "Your Name"
$ git config --global user.email "your_email@whatever.com"
```

2. En este caso se empleo el sitio de github para dar seguimiento al proyecto, dicho sitio cuenta con una sección que permite la administración de llaves publicas a fin de poder agregar contenido al repositorio.

3. Una vez configurado, se procede a crear un primer archivo para agregarlo al repositorio

```
$ git add hello.txt
$ git commit -m "Initial commit"
$ git push origin master
```

Una vez que se cuenta con contenido es fácil copiarlo ya sea otro directorio en el equipo cliente o en otro equipo (previamente se debió de agregar la llave publica)

```
$ git clone git@github.com:manurodriguez/openstack-ansible.git
```

Si solo se desea sincronizar el contenido local descargando los últimos cambios en el repositorio remoto:

```
$ git pull
```

Automatización de las configuraciones

La configuración se encuentra separada por modulo de Openstack, y funcionalidad en archivos en formato YAML conocidos como playbooks, un playbook es ejecutado desde el equipo cliente de la siguiente forma

```
$ ansible-playbook my_playbook.yml -u usuario --sudo
```

De igual forma se pueden emplear parámetros para revisar la sintaxis de un playbook o para ejecutar acciones a partir de cierta tarea:

```
$ ansible-playbook my_playbook.yml  --syntax-check
$ ansible-playbook my_playbook.yml -u usuario –sudo –start-at-task=”Nombre de la tarea”
```

Un playbook consiste básicamente en una serie de instrucciones a ejecutar, a continuación

se muestra un ejemplo con tareas usadas comúnmente durante esta implementación, como se podrá observar las instrucciones son fáciles de comprehender.

Archivo: my_playbook.yml

```
# Primer linea especifica el equipo(s) donde se ejecutaran las tareas.
- hosts: openstack-stg

# definicion de variables, aqui se definen valores a usar
  vars_files:
  - variables.yml

# Inicio de las tareas a ejecutar
  tasks:

   # Instalación de un paquete en ubuntu/debian
    - name: Install ntp service
      apt: name=ntp state=present

   # Arranque de un servicio
    - name: Start service and enable at boot time
      service: name=mysql state=started enabled=yes

   # Modificacion de un archivo
    - name: Configure MySQL character set
      lineinfile: dest=/etc/mysql/my.cnf insertbefore="^\[mysqldump\]"
                  line='character-set-server = utf8'
```

Teniendo como base el ejemplo anterior, la instalación de openstack se compone de los siguientes playbooks, el contenido esta disponible en el sitio de github:

https://github.com/manurodriguez/openstack-ansible/tree/master/openstack-prod

credentials <- Directorio con archivos predefinidos con parámetros de autenticación
- admin
- user

Estos son los parámetros de configuración de de la cuenta de administrador, cuando un usuario quiera hacer uso de las herramientas de openstack puede usar las credenciales de un usuario normal o de administrador exportando el contenido de estos archivos con el comando “source”

```
export OS_USERNAME=myadmin
export OS_PASSWORD=mypassword
export OS_TENANT_NAME=MyProject
export OS_AUTH_URL=http://127.0.0.1:5000/v2.0/
export OS_REGION_NAME=RegionOne
```

variables.yml <- Definición de variables

Este archivo es clave y es el único a parte de los archivos mencionados en el punto anterior que requieren de personalización, aquí se definen los nombres de usuario, contraseñas de cada usuario de servicio y de las bases de datos de cada uno así como otros datos que requieren de intervención del usuario, por ejemplo:

```
RABBIT_PASS: notrabbitmq
MYSQL_ROOTPASS: notmysql
MY_PRIVATE_IP: 127.0.0.1
MY_PUBLIC_IP: "{{ inventory_hostname }}"
AUTH_URL: http://127.0.0.1:5000/v2.0/
KEYSTONE_DBUSER: keystone
KEYSTONE_DBPASS: notkeystone
TENANT_NAME: MyProject
REGION_NAME: RegionOne
SERVICE_TENANT_NAME: Services
MEMBER_USER: myuser
MEMBER_PASS: mypassword
ADMIN_USER: myadmin
ADMIN_PASS: mypassword
ADMIN_ROLE_NAME: admin
GLANCE_DBUSER: glance
GLANCE_DBPASS: notglance
GLANCE_USER: glance
GLANCE_PASS: notglance
NEUTRON_DBUSER: neutron
NEUTRON_DBPASS: notneutron
NEUTRON_USER: neutron
NEUTRON_PASS: notneutron
NOVA_DBUSER: nova
NOVA_DBPASS: notnova
NOVA_USER: nova
NOVA_PASS: notnova
PRIV_SUBNET: 10.87.187.0/24
EXT_INTERFACE: eth0
```

00_init_setup.yml <- Instalación y configuración inicial del servidor

En este playbook se instala el repositorio de Openstack Icehouse, y un Kernel superior a 3.10 para darle soporte a las funcionalidades de Neutron, durante el proceso se aprovecha para reiniciar el equipo y usar el Kernel instalado.

01_rabbitmq_mysql.yml <- Instalación y configuración de RabbitMQ y MySQL

En este playbook se instalan los servicios soporte de los módulos de Openstack, además se agrega una contraseña de root para mysql y se guarda en el archivo /root/.my.cnf para

permitir login automático sin credenciales a root.

02_keystone.yml <- Instalación y configuración del servicio de identificación

En las tareas de este playbook se instalan los paquetes del servicio de autenticación Keystone, se crea su base de datos y se crean los primeros servicios, endpoints, usuarios y roles,

03_glance.yml <- Instalación y configuración del servicio de imágenes

Aquí se instala el servicio de imágenes, y por primera vez se hace uso de la autenticación de los usuarios creados para asociar el servicio con keystone e importar una imagen que será creada en el siguiente apartado.

04_neutron.yml <- Instalación y configuración de servicios y plugins de red

En este playbook se instalan los servicios de red, la configuración es mas extensa por el numero de archivos a modificar, y se incluyen cambios en la configuración de red para crear bridges, si bien se pudo haber hecho uso de templates con la configuración, preferí modificar linea por linea para documentar de que se trata cada parámetro. Y ademas de que se puede arrancar el playbook desde una tarea especifica.

05_nova.yml <- Instalación y despliegue de la configuración de nova

Este playbook también es algo extenso, ya que incluye varias modificaciones a la configuración, se instalan los paquetes de nova y se arrancan todos los servicios, en este punto todas las tareas de los playbooks deben de estar listas para poder arrancar los servicios de nova.

07_horizon.yml <- Instalación y configuración del GUI

Este playbook instala la interfaz web de administración de openstack, se instalo con fines de tener acceso a la consola de las instancias, de forma rápida, útil si se desea tener una herramienta gráfica.

create-instance.yml <- Creación de un primer equipo virtual.

Este playbook no es requerido, solo fue hecho para construir un equipo inicial, ya que contiene los pasos para crear un equipo y asociar las reglas de acceso.

allinone.yml <- Playbooks el 00.. al 07..

En este playbook, se incluyeron todas las tareas de los anteriores, a fin de poder realizar la instalación de Openstack en un equipo con Ubuntu 12.04 y acceso a Internet en solo una linea de comando desde el equipo donde se instalo Ansible, y este ultimo se encargara de realizar cada una de las tareas enviando llamadas al equipo a través del protocolo SSH.

```
$ ansible-playbook allinone.yml --sudo
```

Nota: En el paso anterior, en una de las tareas se importa una imagen de CentOS6.6 usando “glance”, de la cual se explica su creación en el siguiente punto, y en seguida se mostrara como crear una primer instancia.

Creación de una imagen

Actualmente hay una gran cantidad de imágenes disponibles en linea para su uso en sistemas de cloud como OpenStack o EC2 de Amazon, la mayoría contienen un sistema de base y un servicio llamado cloud-init que permite la modificación de parámetros de la instancia cuando es desplegada, en este caso se opto por la creación de una imagen personalizada a fin de conocer el proceso y tener una imagen a la medida. La guiá de pasos completa se puede encontrar en linea[4], básicamente se realizaron las siguientes actividades:

1. Instalación de herramientas de virtualizacion:

```
# yum groupinstall -y @virtualization
# yum install -y libguestfs-tools-c
$ qemu-img create -f qcow2 /tmp/centos-6.qcow2 10G
$ virt-install --virt-type kvm --name centos-6.6 --ram 1024 \
--cdrom=Download/CentOS-6.6-x86_64-minimal.iso \
--disk /tmp/centos-6.qcow2,format=qcow2 --network network=default \
--graphics vnc,listen=0.0.0.0 --noautoconsole --os-type=linux --os-variant=rhel6
```

Una vez lista la imagen se puede arrancar desde la consola de KVM y proceder a su personalización e instalación de cloud init.

Nota: Para que la maquina virtual se conecte a Internet se debe de asignar una interfaz en los Detalles de la maquina virtual, en este caso se elige "Virtual Network default : NAT" Device model: virtio, esto permite usar la interfaz de tipo bridge virbr0 creada por la instalación de KVM y ademas habilita una serie de reglas de iptables para reenviar el trafico de las maquinas virtuales de la interfaz bridge a la interfaz física usada para navegar,

2. Modificación de /etc/sysconfig/network-scripts/ifcfg-eth0 (quitar las referencias de la MAC)

```
TYPE=Ethernet
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
NM_CONTROLLED=no
```

3. Instalación del repositorio de EPEL

4 RedHat Community, RedHat Inc. 2014, Creating CentOS and Fedora images ready for Openstack [Articulo en Linea]. [Fecha de consulta: Noviembre 13 2014] <https://openstack.redhat.com/Creating_CentOS_and_Fedora_images_ready_for_Openstack>

```
# yum install -y http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
```

4. Actualización del sistema

```
# yum -y distro-sync
```

5. Instalación de cloud-init, parted y git

```
# yum install -y cloud-utils cloud-init parted git
```

6. Instalación de linux-rootfs-resize

```
# cd /tmp
# git clone https://github.com/flegmatik/linux-rootfs-resize.git
# cd linux-rootfs-resize
# ./install
```

7. Agregar modificación de resolución en /etc/cloud/cloud.cfg en la sección cloud_init_modules:

```
- resolv-conf
```

8. Apagado del equipo virtual

```
$ poweroff
```

9. Limpiar la imagen

```
$ virt-sysprep -a /tmp/centos6.qcow2
```

10. Reducir los bloques sin usar de la imagen

```
$ export TMPDIR=/tmp/
$ virt-sparsify --compress /tmp/centos6.qcow2 centos-6.6-x86_64-disk.qcow2
```

Creación de Instancias

En los puntos anteriores se explico como instalar Openstack en un servidor y como preparar una imagen a fin de emplearla como base para el aprovisionamiento de instancias; esos son los pasos mas laboriosos; crear una instancia resulta sencillo, para ello se puede hacer usando la herramienta nova, aunque también se escribió un playbook con las tareas a ejecutar, ademas también se puede aprovechar la consola web para hacer uso de cualquiera de las funciones, tales como crear una instancia, a continuación se describen las tres formas de crear una instancia:

a) Ejecutar los comandos manualmente, en el equipo de Openstack basta con escribir este

par de lineas, adicionalmente se debe de crear una regla de acceso o “security groups” para permitir el trafico SSH y ping.

```
# source credentials/admin
# nova boot --image centos-6.6-qcow2 --flavor m1.small --key-name MyKey MyFirstInstance
# neutron security-group-rule-create --direction ingress --ethertype IPv4 --protocol tcp --port-range-min 22 --port-range-max 22 –remote-ip-prefix 10.87.127.0/24 default
```

b) Usando Ansible (desde el equipo cliente):

```
$ ansible-playbook create-instance.yml --sudo
```

c) Ingresar a la consola gráfica http://IP_PUBLICA/horizon seleccionar proyecto / instancias y seleccionar el tamaño de la instancia a crear, la imagen a emplear y el nombre de la instancia.

Por ultimo solo queda por confirmar que la instancia fue creada, desde la linea de comando puede hacerse con la herramienta nova en el equipo con OpenStack:

```
$ source credentials/user
$ nova list
```

Las instancias son almacenas dentro de sub-directorios en /var/lib/nova/ y aunque son instancias virtuales, están al mismo nivel que el equipo físico en cuando a red ya que se opto por una configuración plana sin uso de vlans o direcciones flotantes, aunque pueden ser empleadas a fin de tener un direccionamiento privado segregado del resto.

Despliegue de MongoDB

Una vez que se crean tres instancias virtuales en openstack con la imagen de CentOS6.6 se documentan las direcciones IP asignadas a cada una en la configuración de Ansible y se procede a desplegar la solución, ejemplo:

Archivo: /etc/ansible/hosts

```
[mongo-repl]
10.87.127.35  ansible_ssh_user=benutzer
10.87.127.36  ansible_ssh_user=benutzer
10.87.127.37  ansible_ssh_user=benutzer
```

La instalación de la replica con Ansible consta de 10 tareas que pueden apreciarse en el playbook a continuación. Básicamente es instalar los paquetes de mongoDB, modificar los archivos de configuración con el puerto del servicio, la dirección IP, y el nombre de la replica, se consideran otros pasos para que las instancias interactuen entre si como resolución de nombre y reglas de IPtables.

Archivo: mongo-replica.yml

```
- hosts: mongo-repl

  vars:
   REPL_NAME: myrepl

  tasks:
    - name: Adding MongoDB repository
      template: src=mongodb.repo dest=/etc/yum.repos.d/mongodb.repo

    - name: Install the latest version of MongoDB
      yum: name=mongodb-org state=latest

    - name: Bind MongoDB to Internal IP
      replace: dest=/etc/mongod.conf regexp='^bind_ip=127.0.0.1$'
replace='bind_ip=0.0.0.0'

    - name: Setting Standard MongoDB Port
      replace: dest=/etc/mongod.conf regexp='^#port=27017$' replace='port=27017'

    - name: Adding the replica Set name
      replace: dest=/etc/mongod.conf regexp='^#replSet=setname$'
replace='replSet={{REPL_NAME}}'

    - name: Start MongoDB service and enable at boot time
      service: name=mongod state=started enabled=yes

    - name: Adding the list of members
      template: src=hosts.j2 dest=/etc/hosts

    - name: Allowing mongoDB traffic via iptables
      template: src=iptables.j2 dest=/etc/sysconfig/iptables

    - name: Restarting iptables
      service: name=iptables state=restarted enabled=yes
```

En las tareas de configuración se hace uso de templates en lenguaje Jinja2, lo que permite extraer información de los equipos usando ansible mejor conocida como “facts” de cada una de las instancias y usarla en archivos de configuración, por ejemplo se puede extraer el nombre de las instancias y sus direcciones IP para crear un archivo en el momento de la ejecución del playbook, esto es de mucha ayuda ya que no se depende de entradas estáticas en archivos y si se agregan mas nodos al grupo de mongo el archivo siempre tendrá la información actual cuando se inyecten los cambios.

Archivo: hosts.j2

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

```
{% for host in groups['mongo-repl'] %}
{{hostvars[host]['ansible_eth0']['ipv4']['address']}} {{hostvars[host]['ansible_hostname']}}
{% endfor %}
```

A continuación se muestra el despliegue de mongodb en cuatro pasos, la mayor parte la lleva a cabo el playbook, el resto son comandos manuales para iniciar la replica.[5]

1. Instalación y configuración desde equipo cliente

```
$ ansible-playbook mongo-replica.yml --sudo
```

2. Ingresar a un nodo e iniciar la replica

```
[root@mongo1 ~]# mongo
MongoDB shell version: 2.6.5
connecting to: test
> rs.initiate()
{
        "info2" : "no configuration explicitly specified -- making one",
        "me" : "mongo1:27017",
        "info" : "Config now saved locally.  Should come online in about a minute.",
        "ok" : 1
}
myrepl:PRIMARY> rs.conf()
{
        "_id" : "myrepl",
        "version" : 1,
        "members" : [
                {
                        "_id" : 0,
                        "host" : "mongo1:27017"
                }
        ]
}
```

3. Agregar los otros nodos a la replica

```
myrepl:PRIMARY> rs.add("mongo2")
{ "ok" : 1 }
myrepl:PRIMARY> rs.add("mongo3")
{ "ok" : 1 }
myrepl:PRIMARY> rs.conf()
{
        "_id" : "myrepl",
```

5 Deploy a Replica Set – MongoDB Manual 2.6.6 [Articulo en linea]. [Fecha de consulta: Diciembre 01 2014] http://docs.mongodb.org/manual/tutorial/deploy-replica-set/

```
        "version" : 3,
        "members" : [
                {
                        "_id" : 0,
                        "host" : "mongo1:27017"
                },
                {
                        "_id" : 1,
                        "host" : "mongo2:27017"
                },
                {
                        "_id" : 2,
                        "host" : "mongo3:27017"
                }
        ]
}
```

4. Verificar el estado de la replica

```
myrepl:PRIMARY> rs.status()
{
        "set" : "myrepl",
        "date" : ISODate("2014-12-04T05:50:24Z"),
        "myState" : 1,
        "members" : [
                {
                        "_id" : 0,
                        "name" : "mongo1:27017",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 1417,
                        "optime" : Timestamp(1417672005, 1),
                        "optimeDate" : ISODate("2014-12-04T05:46:45Z"),
                        "electionTime" : Timestamp(1417671234, 2),
                        "electionDate" : ISODate("2014-12-04T05:33:54Z"),
                        "self" : true
                },
                {
                        "_id" : 1,
                        "name" : "mongo2:27017",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 221,
                        "optime" : Timestamp(1417672005, 1),
                        "optimeDate" : ISODate("2014-12-04T05:46:45Z"),
                        "lastHeartbeat" : ISODate("2014-12-04T05:50:23Z"),
```

```
                        "lastHeartbeatRecv" : ISODate("2014-12-04T05:50:22Z"),
                        "pingMs" : 0,
                        "syncingTo" : "mongo1:27017"
                },
                {
                        "_id" : 2,
                        "name" : "mongo3:27017",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 219,
                        "optime" : Timestamp(1417672005, 1),
                        "optimeDate" : ISODate("2014-12-04T05:46:45Z"),
                        "lastHeartbeat" : ISODate("2014-12-04T05:50:24Z"),
                        "lastHeartbeatRecv" : ISODate("2014-12-04T05:50:24Z"),
                        "pingMs" : 1,
                        "syncingTo" : "mongo1:27017"
                }
        ],
        "ok" : 1
}
```


Despliegue de Docker

De igual forma que las instancias de MongoDB, una vez creadas un par de instancias para Docker, se llevara a cabo la instalación, al tratarse de un playbook muy básico se describe las tareas usadas a continuación, como se podrá observar básicamente se instalan los paquetes, se arranca el servicio y se puede iniciar a descargar contenedores.

Archivo: docker-install.yml

```
- hosts: docker

  tasks:
    - name: Adding Epel repository
      yum: name=epel-release state=latest

    - name: Install the latest version of Docker
      yum: name=docker-io state=latest

    - name: Install the latest version of Docker client
      yum: name=python-docker-py state=latest

    - name: Start Docker service and enable at boot time
      service: name=docker state=started enabled=yes

    - name:  Start centos instance
      command: docker pull centos
```

Una vez teniendo Docker instalado se puede arrancar un contenedor

```
$ docker run centos
```

Inclusive si se desea tener acceso al shell del contenedor se puede ejecutar lo siguiente

```
$ docker run -t -i centos /bin/bash
```

3. O si se cuenta con una imagen personalizada en Docker hub se puede descargar dicho contenedor, en este ejemplo se descarga una imagen de mi registro publico llamada apache, esta imagen se creo inicialmente de la imagen publica/oficial de ubuntu a la cual se le instalo apache/

```
[root@docker1 ~]# docker run -d -P manudocker/apache
Unable to find image 'manudocker/apache' locally
Pulling repository manudocker/apache
4cced7d1d417: Download complete
511136ea3c5a: Download complete
01bf15a18638: Download complete
30541f8f3062: Download complete
e1cdf371fbde: Download complete
9bd07e480c5b: Download complete
Status: Downloaded newer image for manudocker/apache:latest
63cf4374e7586e5b209fbf34321d6422931f9201ae120bb22a0a717d8f1d2d8a
```

Podemos confirmar que la imagen esta presente en el equipo:

```
[root@docker1 ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
manudocker/apache   latest              4cced7d1d417        12 days ago         227.3 MB
```

Iniciamos un contenedor a partir de la imagen:

```
[root@docker1 ~]# docker run -t -i manudocker/apache /bin/bash
root@028d50242163:/# cat /etc/issue
Ubuntu 14.04.1 LTS \n \
```

Y enseguida podemos observar el estatus del contenedor

```
[root@docker1 ~]# docker ps -l
CONTAINER ID        IMAGE                       COMMAND             CREATED             STATUS                     PORTS               NAMES
028d50242163        manudocker/apache:latest    "/bin/bash"         7 seconds ago       Exited (0) 6 seconds ago                       romantic_morse
```

Algo importante de mencionar es que Docker crea una interfaz virtual con una subred privada en el equipo host a través de la cual se comunicara con los contenedores, estos también tendrán una IP del mismo segmento y se encuentran conectadas a la interfaz virtual del equipo host, ademas Docker también administrará una serie de reglas Iptables para que los contenedores pueden enviar/recibir tráfico del exterior,

Ejemplo : instancia virtual de openstack

```
[root@docker1 ~]# ip a show docker0
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
    inet 172.17.42.1/16 scope global docker0
    inet6 fe80::1864:ebff:fec7:1a2f/64 scope link
       valid_lft forever preferred_lft forever
```

Ejemplo: contenedor (dentro de la instancia)

```
root@028d50242163:/# ip a show eth0
20: eth0: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 02:42:ac:11:00:09 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.9/16 scope global eth0
    inet6 fe80::42:acff:fe11:9/64 scope link
       valid_lft forever preferred_lft forever
```

Ejemplo: reglas iptables

```
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    8  1280 ACCEPT     tcp  --  !docker0 docker0  0.0.0.0/0            172.17.0.4          tcp dpt:80
  701 6426K ACCEPT     all  --  *       docker0  0.0.0.0/0            0.0.0.0/0           ctstate RELATED,ESTABLISHED
  524 32020 ACCEPT     all  --  docker0 !docker0  0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     all  --  docker0 docker0  0.0.0.0/0            0.0.0.0/0
```

Como se menciono anteriormente, una vez teniendo una imagen se pueden realizar cambios y generar una nueva imagen, en este ejemplo se instalo PHP a la imagen que tenia apache previamente

```
root@028d50242163:/# apt-get install php5
root@028d50242163:/# vi /var/www/html/index.php
<?php echo "Hello World PHP!" . "\n"; ?>
root@028d50242163:/# exit
```

Después guardamos la imagen al estilo git, es decir haciendo un commit de los cambios, especificando el id del contenedor

```
[root@docker1 ~]# docker commit -m="Added php" -a="Manu Rodriguez" 028d50242163 manudocker/apache-php
```

Y enseguida podemos crear un nuevo contenedor a partir de esta imagen, especificando el puerto que usara apache para recibir trafico, esto permite que todo el trafico recibido en el puerto tcp/80 del equipo host (docker1) sera enviado al contenedor.

```
[root@docker1 ~]# docker run -d -p 80:80 manudocker/apache-php /usr/sbin/apache2ctl -D FOREGROUND
```

Por ultimo guardamos la nueva imagen en el registro publico del Docker hub

```
[root@docker1 ~]# docker push manudocker/apache -php
The push refers to a repository [manudocker/apache-php] (len: 1)
Sending image list

Please login prior to push:
Username: manudocker
Password: ***********
Email: manu.rodri.hernan@gmail.com
Login Succeeded

Pushing repository manudocker/apache-php (1 tags)
511136ea3c5a: Image already pushed, skipping
01bf15a18638: Image already pushed, skipping
30541f8f3062: Image already pushed, skipping
e1cdf371fbde: Image already pushed, skipping
9bd07e480c5b: Image already pushed, skipping
4cced7d1d417: Image already pushed, skipping
42c4bc755f8e: Image successfully pushed
Pushing tag for rev [42c4bc755f8e] on {https://cdn-registry-1.docker.io/v1/repositories/manudocker/apache-php/tags/latest}
```

Resultados y conclusiones

Objetivos cumplidos

El despliegue de OpenStack con Ansible fue completado, sin duda fue la parte mas compleja por la serie de módulos que lo componen, y especialmente por la cantidad de actividades, muchas de las cuales requieren un orden especifico y otras no se encuentran documentadas de forma apropiada, ademas de varios bugs encontrados,
Por otra parte la instalación de MongoDB y Docker resulto mas sencilla por el numero reducido de actividades para su despliegue, una vez las instancias creadas basta con aplicar un Playbook para instalar las aplicaciones, seguidas de una serie de pasos manuales sencillos para verificar su funcionamiento.

Objetivos no cumplidos

El objetivo de desplegar un cluster de Hadoop usando Ansible a través de instancias virtuales de Openstack no pudo ser completado, pero si fue iniciado, el playbook se puede observar en el repositorio de github mencionado en párrafos anteriores, la idea es crear un playbook para iniciar un nodo maestro y otro playbook para implementar nodos esclavos. La implementación sera completada en el futuro, y muy probablemente se considere el uso de la ultima versión de Ubuntu y Openstack ya que incluyen un nuevo proyecto que permite interactuar con Hadoop de forma mas sencilla llamado Sahara.

Futuras ampliaciones

Procesamiento - Al no poderse realizar el despliegue de Hadoop a través de Ansible, se considero la instalación de la ultima versión de Openstack (Juno) que apareció recientemente (Octubre 2014), ya que incluye un modulo de procesamiento conocido como Sahara[6], aunque Juno fue desarrollado para ubuntu 14.04, Sahara aun se encuentra en beta y la documentación es muy básica; razones por las cuales no se opto por replantear el proyecto y usar la ultima versión de Ubuntu/Openstack.

Orchestration – Openstack cuenta con un modulo llamado Heat[7], que permite crear y manejar instancias virtuales a través de templates HOT, que no es mas una serie de instrucciones en formato Yaml para indicar el tamaño de la instancia y las tareas de pos-instalación y configuración una vez que la instancia fue creada. A pesar de que se inicio un playbook para el modulo de Heat, existen fallas en los templates para la creación de instancias, por lo tanto se considera en futuras ampliaciones ya que en dichos templates se pueden incluir parámetros para la instalación de contenedores de Docker al momento de

6 Openstack Foundation, Add the Data processing service - Openstack Installation Guide for Ubuntu 14.04 [Articulo en linea]. [Fecha de consulta: Diciembre 07 2014] http://docs.openstack.org/juno/install-guide/install/apt/content/ch_sahara.html

7 Openstack Foundation, Add the Orchestration service - Openstack Installation Guide for Ubuntu 12.04/14.04 (LTS) [Articulo en linea]. [Fecha de consulta: Diciembre 13 2014] http://docs.openstack.org/icehouse/install-guide/install/apt/content/ch_heat.html

crear una instancia.

Cinder (volúmenes) – En esta versión no se utilizo el modulo de Cinder para crear volúmenes de bloques, por lo tanto queda pendiente la automatización de la configuración, se cuenta con las tareas en script Bash pero falta integrarlo en un playbook. Así como agregar tareas para el uso de volúmenes en instancias creadas.

Conclusiones

Planear la automatización de varias herramientas puede resultar complicado si no se planea una automatización por módulos, ademas al aprender el uso de una herramienta puede resultar un poco lento al inicio al no conocer las opciones disponibles para resolver un problema; es a través de la ejecución de pruebas y lectura de la documentación que se puede iniciar a usar dichas herramientas, inclusive resulta mas fácil desplegar las funciones de la herramienta de automatización en equipo virtuales que puedan ser re-creados en cuestión de minutos en caso de que algo no funcione, Ansible cuenta con funciones de revisión de sintaxis pero muchas veces las actividades programadas no resultan como planeado y lleva tiempo regresar un equipo a su estado inicial para volver a iniciar pruebas o corregir errores, si bien una instancia virtual muchas veces no cuenta con la capacidad para hacer pruebas de las aplicaciones, se compensa por su facilidad de crear una imagen y restauración, y una vez validada la automatización puede pasarse a fases mas estrictas de prueba en el equipo físico.

Durante la propuesta de trabajo de esta memoria se pensó usar la ultima versión estable de Openstack (en este caso Icehouse), aunque no se considero la velocidad a la cual se mueve este proyecto, y durante el desarrollo del mismo una nueva versión de Openstack apareció, esta versión resuelve muchos de los bugs encontrados en la versión previa, e incluyo nuevas funcionalidades que permiten trabajar con clusters de Hadoop, por lo tanto usar esta versión hubiera solucionado algunos de los problemas encontrados pero por otra parte hubiera retrasado mas el proyecto ya que apareció a mediados de Octubre 2014, y cada seis meses se dispone de una nueva versión, por lo tanto, resalto que hay que estar en contacto con los ciclos de desarrollo de un proyecto y estar al tanto de sus avances para determinar como pueden influir en los resultados de un proyecto, si se tiene pensado emplearse a fin de determinar los tiempos de entrega o para determinar que tanto puede influir un nuevo release en la entrega final. Por otra parte también hago notar que hay veces que se tiene que optar con seguir con los recursos iniciales y mejor prepararse para una futura actualización si ya se tiene trabajo realizado, por el numero de cambios que representa usar la ultima versión de un sistema.

Bibliografía

- Openstack Installation Guide for Ubuntu 12.04/14/04 (LTS) – Icehouse , Openstack Foundation [Articulo en linea]. [Fecha de consulta: Septiembre 13 2014] http://docs.openstack.org/icehouse/install-guide/install/apt/content/

- Intro to Playbooks – Ansible Documentation – Ansible Inc. [Articulo en linea]. [Fecha de consulta: Septiembre 17 2014] http://docs.ansible.com/playbooks_intro.html

- Install MongoDB on Red Hat Enterprise, CentOS, Fedora, or Amazon Linux – MongoDB Manual 2.6.6 [Articulo en linea]. [Fecha de consulta: Diciembre 01 2014] http://docs.mongodb.org/manual/tutorial/install-mongodb-on-red-hat-centos-or-fedora-linux/

- Working with containers Docker Documentation – Docker Inc. [Articulo en linea]. [Fecha de consulta: Diciembre 02 2014] https://docs.docker.com/userguide/usingdocker/

- Getting Started with Docker , CoreOS Inc. [Articulo en linea]. [Fecha de consulta: Diciembre 02 2014] https://coreos.com/docs/launching-containers/building/getting-started-with-docker/

Anexo A

GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a

Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as

"Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque

copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a

Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list

them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of

some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that

proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

Anexo B

Características del Servidor

Modelo Dell R710

- Sistema Operativo: Ubuntu 12.04 x86_64
- 2 CPU físicos de 8 núcleos cada uno = 16 cores
- 96 GB RAM
- 2 discos duros 7200 RPM en RAID1 de 150 GB
- 1 Interfaz de red

Anexo C

Playbooks de Ansible

El código completo de los playbooks empleados puede verse en el repositorio publico de github:

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

En el directorio principal se cuenta con la licencia de uso y un archivo readme donde se explica su utilización, ahí también se encuentran los Playbooks para instalar Openstack, MongoDB y Docker.

Openstack: https://github.com/manurodriguez/openstack-ansible/tree/master/openstack-prod
MongoDB: https://github.com/manurodriguez/openstack-ansible/tree/master/mongodb
Docker: https://github.com/manurodriguez/openstack-ansible/tree/master/docker