Cloud Computing
Lab. Instalación OpenStack

OpenStack

El siguiente procedimiento muestra a detalle como realizar la preparación del servidor destino, el ambiente instalación y configuración de todos los componentes necesarios para el funcionamiento de la plataforma

Ambiente / Procedimiento

1.- CentOS 7 instalado, 3 interfaces de red, 1era NAT mode, 2da vboxonly (192.168.56.2), 3ra vboxonly (no-ip)

2.- Epel-release-nonarch instalado y actualizado a su mas reciente versión

```
yum install epel-release.noarch -y
```

3.- Update completo de la distro

```
yum update -y
```

4.- Seteo del servicio NTP "chrony", el cual se instala en los servicios de "postinstalación" automáticos de CentOS 7 cuando se elige el servidor de infra estructura.

```
vim /etc/chrony.conf
```

```
server 0.centos.pool.ntp.org iburst
 server 1.centos.pool.ntp.org iburst
 server 2.centos.pool.ntp.org iburst
 server 3.centos.pool.ntp.org iburst
```

5.- Desactivar SElinux y Firewalld.

```
vim /etc/sysconfig/selinux
```

SELINUX=disabled

setenforce 0
systemctl stop firewalld.service
systemctl disable firewalld.service

6.- Activar repositorios de OpenStack.

yum install centos-release-openstack-ocata

Con lo cual se instalarán las siguientes dependencias

centos-release-openstack-ocata noarch 1-1.el7 extras 5.1 k
Instalando para las dependencias:
 centos-release-ceph-jewel noarch 1.0-1.el7.centos extras 4.1 k
 centos-release-qemu-ev noarch 1.0-1.el7 extras 11 k
 centos-release-storage-common noarch 1-2.el7.centos extras 4.5 k
 centos-release-virt-common noarch 1-1.el7.centos extras 4.5 k

7.- Finalizar la instalación del repo haciendo upgrade completo.

yum upgrade

Lo que nos arroja la siguiente salida.

No se ha instalado la llave pública de mariadb-common-10.1.20-1.el7.x86_64.rpm
(1/4): mariadb-common-10.1.20-1.el7.x86_64.rpm | 63 kB 00:00:00
(2/4): mariadb-libs-10.1.20-1.el7.x86_64.rpm | 643 kB 00:00:03
(3/4): python-six-1.10.0-3.el7.noarch.rpm | 30 kB 00:00:01
(4/4): mariadb-config-10.1.20-1.el7.x86_64.rpm | 26 kB 00:00:07

Total 100 kB/s | 762 kB 00:00:07
Obteniendo clave desde file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-SIG-Cloud
Importando llave GPG 0x764429E6:
 Usuarioid : "CentOS Cloud SIG (http://wiki.centos.org/SpecialInterestGroup/Cloud) <security@centos.org>"
 Huella : 736a f511 6d9c 40e2 af6b 074b f9b9 fee7 7644 29e6
 Paquete : centos-release-openstack-ocata-1-1.el7.noarch (@extras)

NOTA:

Dado que las librerias elementales de mariaDB son instaladas en el transcurso de las configuraciones de postinstalación del SO, el upgrade nos arroja que no se ha configurado aún una llave pública y que la configuración actual de MariaDB no es segura, este punto se tratará mas adelante según las recomendaciones de la documentación oficial " https://mariadb.com/kb/es/obtener-instalar-y-actualizar-mariadb/ " más los pasos prácticos para asegurar la BD y su usuario root según " http://blog.segu-info.com.ar/2013/05/instalar-mariadb-en-linux-en-10-pasos.html "

8.- Instalar el cliente OpenStack.

```
yum install python-openstackclient
```

9.- Instalar OpenStack SElinux.

```
yum install openstack-selinux
```

10.- Ajustar el kernel del SO para deshabilitar el filtrado de rutas inversas y activar el ip forward, esto con la finalidad de trabajar a futuro con el componente Neutron.

```
vim /etc/sysctl.conf
```

El archivo quedará de la siguiente manera:

```
# sysctl settings are defined through files in
# /usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
# Vendors settings live in /usr/lib/sysctl.d/.
# To override a whole file, create a new file with the same in
# /etc/sysctl.d/ and put new settings there. To override
# only specific settings, add a file with a lexically later
# name in /etc/sysctl.d/ and put new settings there.
```

```
#
# For more information, see sysctl.conf(5) and sysctl.d(5).
#

net.ipv4.ip_forward=1
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
```

sysctl -p - (Cargar nueva configuración del kernel)

11.- Instalación adecuada y segura de MariaDB y Python2-MySQL.

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

Luego creamos y editamos el archivo /etc/my.cnf.d/openstack.cnf y colocamos lo siguiente para permitir conexiones de los demás componentes a futuro:

```
[mysqld]
bind-address = 0.0.0.0

default-storage-engine = innodb
innodb_file_per_table = on
max_connections = 4096
collation-server = utf8_general_ci
character-set-server = utf8
```

Iniciamos el servicio y lo colocamos en el arranque del sistema:

```
systemctl enable mariadb.service
systemctl start mariadb.service
```

Para asegurar la BD y su usuario root, podemos aprovecharnos del script interno de mysql:

mysql_secure_installation

Donde se nos preguntarán los siguientes parámetros:

```
Set root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!

Remove anonymous users? [Y/n] y
 ... Success!

Disallow root login remotely? [Y/n] n
 ... skipping.

Remove test database and access to it? [Y/n] n
 ... skipping.

Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB
installation should now be secure.
```

12.- Instalación del sistema de control de colas o mejor conocido como Message Broker (RabbitMQ), gracias a este OpenStack puede manipular y coordinar las operaciones entre sus distintos componentes.

NOTA: RabbitMQ es un servicio global que necesita ser accesado por multiples componentes que quizas puedan estar ubicados en otros servidores.

yum install rabbitmq-server

Arranque automático e inicio:

```
systemctl enable rabbitmq-server.service
systemctl start rabbitmq-server.service
```

Creación de vhost para ser usado por openstack y asignación de permisos de acceso, lectura y escritura:

```
rabbitmqctl add_user openstack RABBIT_PASS
```
-- **NOTA**: RABBIT_PASS se cambia por un valor de password válido y secreto.

```
rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

Creación del vhost

```
rabbitmqctl add_vhost "/openstack"

rabbitmqctl set_permissions -p "/openstack" openstack ".*" ".*" ".*"
```

Habilitar a RabbitMQ para que escuche por todas las interfaces.

En la ruta /etc/rabbitmq existe un archivo de configuración global para los settings de rabbit, sin embargo toda esta configuración es la que por defecto trae el message broker, para modificarla debemos crear un archivo que contendrá las variables de entorno a ambiente de rabbit y que a su vez sobre escribirá toda la configuración por defatul, siguiendo la documentación oficial:

https://www.rabbitmq.com/man/rabbitmq-env.conf.5.man.html
https://www.rabbitmq.com/configure.html

Se procede a crear un archivo llamado rabbitmq-env.conf, donde setearemos la

variable **RABBITMQ_NODE_IP_ADDRESS**

vim /etc/rabbitmq-env.conf

RABBITMQ_NODE_IP_ADDRESS=0.0.0.0

Reiniciar el servicio

rabbitmqctl restart

13.- Por ultimo paso para el ambiente completo necesario, vamos a instalar memcached que es un sistema de memoria distribuida para almacenamiento de cachem, esto openstack lo utiliza para acelerar las entregas que hacen todas las aplicaciones web y aliviana la carga en las bases de datos.

yum install memcached python-memcached

Editar el archivo /etc/sysconfig/memcached:

```
PORT="11211"
USER="memcached"
MAXCONN="1024"
CACHESIZE="64"
OPTIONS="-l 0.0.0.0,::1"
```

Como estamos construyendo un all-in-one, permitimos acceso a todas las redes

Arranque automático e inicio:

```
systemctl enable memcached.service
systemctl start memcached.service
```

14.- Creación y configuración de interfaz dummy.

Crear el archivo del modulo para que el kernel lo asuma

vim /etc/modules-load.d/dummy.conf

Load dummy.ko at boot
dummy

(Se guarda el archivo)

Crear el archivo de configuración automatizada que el kernel necesitará leer para ejecutar el levantamiento

vim /etc/modprobe.d/dummy.conf

install dummy /sbin/modprobe --ignore-install dummy; /sbin/ip link set name dummy0 dev dummy0

Crear el archivo de configuración física de la interfaz.

vim /etc/sysconfig/network-scripts/ifcfg-dummy0

DEVICE=dummy0
TYPE=OVSPort
DEVICETYPE=ovs
OVS_BRIDGE=br-dummy0
ONBOOT=yes
NM_CONTROLLED=no
ARP=yes
PROMISC=yes

NOTA: Promiscuo porque necesitas capturar todo el tráfico de las demás interfaces, y ARP debe mantener la tabla ARP

15.- Instalación OpenVSWITCH

```
yum -y install openvswitch

service openvswitch start

chkconfig openvswitch on
```

Ahora se proceden a crear los bridges, el bridge fundamental es el de integración.

```
ovs-vsctl add-br br-int
```

Luego el bridge dummy (br-dummy0)

```
ovs-vsctl add-br br-dummy0
```

Luego añadir el puerto

```
ovs-vsctl add-port br-dummy0 dummy0
```

Por ultimo, crear el archivo de configuración física de dicha interfaz

```
vim /etc/sysconfig/network-scripts/ifcfg-br-dummy0
```

ifcfg-br-dummy0
DEVICE=br-dummy0
DEVICETYPE=ovs
TYPE=OVSBridge
ONBOOT=yes
IPADDR=192.168.57.1
NETMASK=255.255.255.0
BROADCAST=192.168.57.255
ARP=yes
PROMISC=yes

NOTA: Promiscuo porque necesitas capturar todo el tráfico de las demás interfaces, y ARP debe mantener la tabla ARP

16.- Reglas de IPTABLES

Para SSH:
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A OUTPUT -p udp --sport 22 -j ACCEPT

Para MySQL:
iptables -A INPUT -p tcp --dport 3306 -j ACCEPT
iptables -A OUTPUT -p udp --sport 3306 -j ACCEPT

Para Rabbit/AMQP:
iptables -A INPUT -p tcp --dport 5672 -j ACCEPT
iptables -A OUTPUT -p udp --sport 5672 -j ACCEPT

iptables -A INPUT -p tcp --dport 5671 -j ACCEPT
iptables -A OUTPUT -p udp --sport 5671 -j ACCEPT

Para Keystone:

(public)
iptables -A INPUT -p tcp --dport 5000 -j ACCEPT
iptables -A OUTPUT -p udp --sport 5000 -j ACCEPT

(admin)
iptables -A INPUT -p tcp --dport 35357 -j ACCEPT
iptables -A OUTPUT -p udp --sport 35357 -j ACCEPT

Se crea el archivo donde vamos almacenar todas las reglas que acabamos de añadir.

(touch /etc/sysconfig/iptables.first.rules)

Y hacemos un volcado de la configuración actual de nuestro firewall hacia ese archivo.

iptables-save > /etc/sysconfig/iptables

Luego instalamos el servicio iptables-service

yum install iptables-services

Por medio de systemctl programamos su arranque automático.

systemctl enable iptables

Reiniciamos el servicio y listo.

systemctl restart iptables.service

Con esto ya tenemos el ambiente del sistema operativo preparado y en optimas

condiciones para iniciar la instalación de los componentes CORE de **OpenStack**

################ **Instalación Componente Core KEYSTONE** ######################

1.- Instalación de servicio CORE Identity (KEYSTONE).

Primero creamos la base de datos que utilizará Keystone, para ello nos conectamos como root.

mysql -u root -p -- y nos enviará directamente al prompt inicial de MariaDB (MariaDB [(none)]>)

Creamos la base de datos.

```
MariaDB [(none)]> CREATE DATABASE keystone;
```

Garantizamos todos los privilegios para el acceso.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY 'rel19224935';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY 'rel19224935';
```

NOTA: KEYSTONE_DBPASS se debe cambiar por un password válido

Salimos de MariaDB para continuar con la instalación del componente de identidad.

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

Modificar los siguientes parámetros en el archivo de configuración global de keystone

vim /etc/keystone/keystone.conf

En la sección de database:

[database]
# ...
connection = mysql+pymysql://keystone:rel19224935@192.168.56.2/keystone --
NOTA: KEYSTONE_DBPASS un valor de password válido

y en la sección de toker:

[token]
# ...
provider = fernet

Poblar la base de datos anteriormente creada con la ayuda del script que trae keystone.

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

Inicializar los fernet keys que vienen incluidos y que se activaron anteriormente.

keystone-manage fernet_setup --keystone-user keystone --keystone-group keystone
keystone-manage credential_setup --keystone-user keystone --keystone-group keystone

Bootstrap.

```
keystone-manage bootstrap --bootstrap-password ADMIN_PASS \
 --bootstrap-admin-url http://controller:35357/v3/ \
 --bootstrap-internal-url http://controller:5000/v3/ \
 --bootstrap-public-url http://controller:5000/v3/ \
 --bootstrap-region-id RegionOne
```

NOTA: el bootstrap debe ser en una linea completa.

```
keystone-manage bootstrap --bootstrap-password rel19224935 --bootstrap-admin-url http://192.168.56.2:35357/v3/ --bootstrap-internal-url http://192.168.56.2:5000/v3/ --bootstrap-public-url http://192.168.56.2:5000/v3/ --bootstrap-region-id RegionOne
```

Configuración del server APACHE.

Editar el archivo de configuración global de apache y se cambia la directiva ServerName.

```
vim /etc/httpd/conf/httpd.conf
```

```
ServerName 192.168.56.2:80
```

Se crea un enlace simbólico de la configuración wsgi de keystone hacia el archivo de config global de apache.

```
ln -s /usr/share/keystone/wsgi-keystone.conf /etc/httpd/conf.d/
```

Se inicializa apache.

systemctl enable httpd.service
systemctl start httpd.service

Configurar la cuenta administrativa.

$ export OS_USERNAME=admin
$ export OS_PASSWORD=ADMIN_PASS -- NOTA: este password es el mismoq ue se introdujo en el script de bootstrap anterior
$ export OS_PROJECT_NAME=admin
$ export OS_USER_DOMAIN_NAME=Default
$ export OS_PROJECT_DOMAIN_NAME=Default
$ export OS_AUTH_URL=http://192.168.56.2:35357/v3
$ export OS_IDENTITY_API_VERSION=3

Ahora empezamos a crear dominios, usuarios, roles y servicios

Proyecto de servicio:
openstack project create --domain default --description "Service Project" service

Proyecto demo:
openstack project create --domain default --description "Demo Project" demo

Crear usuario demo
openstack user create --domain default --password-prompt demo

user password:
confirm password:

Crear role
openstack role create user

Añadir rol de usuario al proyecto y usuario demo:
openstack role add --project demo --user demo user

Ahora podemos listar todo, algunos ejemplos:

Usuarios:

[root@ocata keystone]# openstack user list

Roles:

[root@ocata keystone]# openstack role list

Proyectos:

[root@ocata keystone]# openstack project list

Catálogo de servicios de nuestra nube y sus endpoints:

[root@ocata home]# openstack endpoint list

[root@ocata home]# openstack catalog list

2.- Creación y adición de discos destinados para instalar componentes CORE adicionales en VirtualBox (Cinder y Swift)

```
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 64G 0 disk
├─sda1 8:1 0 500M 0 part /boot
└─sda2 8:2 0 63,5G 0 part
 ├─centos-root 253:0 0 40G 0 lvm /
```

```
├─centos-swap 253:1 0 3,9G 0 lvm [SWAP]
└─centos-home 253:2 0 19,6G 0 lvm /home
sdb 8:16 0 16G 0 disk
sdc 8:32 0 16G 0 disk
sr0 11:0 1 1024M 0 rom
```

#################### Instalación Componente Core GLANCE ############################

1.- Instalación de servicio CORE image (GLANCE)

Antes de instalar el componente se necesitan algunos pre requisitos, el primero de ellos es crear y preparar la base de datos que el mismo utilizará.

```
mysql -u root -p
```

Y nos enviará directamente al prompt inicial de **MariaDB (MariaDB [(none)]>)**

Creamos la base de datos.

```
MariaDB [(none)]> CREATE DATABASE glance;
```

Garantizamos todos los privilegios para el acceso.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' IDENTIFIED BY 'rel19224935';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' IDENTIFIED BY 'rel19224935';
```

NOTA: GLANCE_DBPASS debe ser un valor de password válido, para todo este laboratorio se utilizará el mismo password (**rel19224935**).

Salimos de MariaDB (> quit;)

NOTA: Glance, no utiliza RabbitMQ normalmente a diferencia de sus demás

componentes hermanos, excepto cuando Ceilometer está presente, porque Ceilometer que es el componente de métricas, el mismo necesita tener acceso a RabbitMQ, y por eso necesitamos darle acceso a Glance contra RabbitMQ, de esta manera se podrán obtener métricas de este componente

Ahora haremos uso de KEYSTONE para darle identidad a GLANCE dentro de nuestra nube, para esto hacemos "source" de nuestro archivo de variables

```
source /home/admin-openrc
```

Luego creamos el usuario Glance.

```
openstack user create --domain default --password-prompt glance
User Password:
Repeat User Password:
```

Añadir role de admin al usuario y servicio de Glance

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

(Esta orden no genera ningún tipo de salida)

Creamos la entidad de servicio de Glance

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

Ahora procedemos a crear los endpoints que utilizará Glance, recordando que son 3, el admin, public e internal.

```
openstack endpoint create --region RegionOne image public
http://192.168.56.2:9292

openstack endpoint create --region RegionOne image internal
http://192.168.56.2:9292

openstack endpoint create --region RegionOne image admin
http://192.168.56.2:9292
```

Ahora, antes de instalar el componente CORE Glance, vamos a preparar el SO para recibir dicho componente, es decir, reglas de IPtables.

(GLANCE)
iptables -A INPUT -p tcp --dport 9292 -j ACCEPT
iptables -A OUTPUT -p udp --sport 9292 -j ACCEPT

NOTA: estas reglas se pueden resumir de la siguiente manera:
iptables -A INPUT -p tcp -m multiport --dports 9292 -j ACCEPT

Salvamos la configuración.

iptables-save > /etc/iptables.first.rules

iptables-save > /etc/iptables

Procedemos a instalar GLANCE.

yum install openstack-glance -y

Editamos el archivo /etc/glance/glance-api.conf y nos vamos a las secciones de [database], [keystone_authtoken], [paste_deploy] y [glance_store]

vim /etc/glance/glance-api.conf

[database]
# ...
connection = mysql+pymysql://glance:rel19224935@192.168.56.2/glance -- NOTA: GLANCE_DBPASS un valor de password válido

[keystone_authtoken]
# ...
auth_uri = http://192.168.56.2:5000
auth_url = http://192.168.56.2:35357
memcached_servers = 192.168.56.2:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = glance
password = rel19224935

[paste_deploy]
# ...
flavor = keystone

[glance_store]
# ...
stores = file,http
default_store = file
filesystem_store_datadir = /var/lib/glance/images/

Editamos el archivo /etc/glance/glance-registry.conf y nos vamos a las secciones de [database], [keystone_authtoken], [paste_deploy] y [glance_store]

vim /etc/glance/glance-registry.conf

[database]

```
# ...
connection = mysql+pymysql://glance:rel19224935@192.168.56.2/glance



[keystone_authtoken]
# ...
auth_uri = http://192.168.56.2:5000
auth_url = http://192.168.56.2:35357
memcached_servers = 192.168.56.2:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = glance
password = rel19224935



[paste_deploy]
# ...
flavor = keystone



[DEFAULT]

#
# From glance.registry
#
bind_host = 0.0.0.0
bind_port = 9191
```

Salvamos los cambios.

Ahora se popula la base de datos del servicio de imagen.

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

NOTA: Ignoramos cualquier mensaje de obsolecencia que pueda arrojar este comando, esto ocurre porque el engine de oslo db ya está obsoleto "**deprecated**" para las nuevas versiones de OpenStack.

Por último finalizaremos la instalación.

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

Para verificar la operación y correcto aprovisionamiento de nuestro nuevo componente, vamos a proceder a crear una imagen.

NOTA: recuerde hacer source del archivo RC de admin (source /home/admin-openrc)

Creamos el directorio /etc/glance/images -- esto con la finalidad de descargar nuestras imágenes allí, empezamos descargando las de cirros.

```
mkdir /etc/glance/images

cd /etc/glance/images
```

```
wget http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-disk.img
 wget http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-i386-disk.img
```

Procedemos a crear nuestra primera imagen con GLANCE.

(64 bits)

```
openstack image create "cirros64" --file cirros-0.3.5-x86_64-disk.img --disk-format qcow2 --min-disk 2 --container-format bare --public
```

(32 bits)

```
openstack image create "cirros32" --file cirros-0.3.5-xi386-disk.img --disk-format qcow2 --min-disk 2 --container-format bare --public
```

--min-disk **<MIN_DISK>**
Amount of disk space (in GB) required to boot image.
Cantidad de espacio en disco (en GB) necesaria para iniciar la imagen.

--min-ram **<MIN_RAM>**
Amount of ram (in MB) required to boot image.
Cantidad de RAM (en MB) necesaria para iniciar la imagen.

Cómo afecta min_disk cuando es almacenamiento efímero?

- Dicha instancia se verá afectada de manera que el flavor a escoger, debe poder contener la imagen, es decir, si el min_disk de la imagen es de 8Gb, el flavor que se lo colocará a dicha instancia debe tener al menos 8Gb de HDD, de hecho, openstack no te permite seleccionar flavors que no cumplan con las características mínimas de la imagen.

- min_disk va a convalidarse contra el espacio especificado en el flavor, si el del flavor es igual o mayor, la instancia se crea, si es menor, no se crea.

Cómo afecta min_disk cuando es almacenamiento persistente?

- Cuando creas un volumen para almacenamiento persistente en Cinder, se crea a partir de una imagen, y esta imagen ya posee un min_disk, quiere decir que si se levanta una instancia utilizando almacenamiento persistente, ya viene delimitado por la imagen con la cual haya sido creado el volumen como tal.

- No obedece el valor del hdd del volumen sino que se va automáticamente al valor de min_disk de la imagen

- min_disk es donde realmente trabaja no como filtro, sino como acción, automaticamente asume el tamaño del disco persistente, basado en la imagen con la que fue creada ese volumen.

Ahora podemos proceder a listar todo lo referente al servicio de imagen.

[root@ocata glance]# openstack endpoint list

[root@ocata glance]# openstack image list

Podemos observar como Glance guarda las imágenes creadas.

[root@ocata glance]# ls /var/lib/glance/images/
94840643-fdc9-4173-bbc1-e54dca9e89cb
c12ec887-d6f7-409a-8a45-4733998d026d

Consultemos por ultimo nuestro catálogo de servicios.

[root@ocata glance]# openstack catalog list

#################### Instalación Componente Core CINDER ##########################

1.- Instalación de servicio CORE block storage (CINDER)

Antes de instalar el componente se necesitan algunos pre requisitos, el primero de ellos es crear y preparar la base de datos que el mismo utilizará.

mysql -u root -p -- y nos enviará directamente al prompt inicial de MariaDB (MariaDB [(none)]>)

Ahora creamos la base de datos que utilizará cinder

```
MariaDB [(none)]> CREATE DATABASE cinder;
```

Garantizamos los privilegios

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' IDENTIFIED BY 'rel19224935';
 MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%' IDENTIFIED BY 'rel19224935';
```

Creamos las reglas de iptables

```
iptables -A INPUT -p tcp -m multiport --dports 8776 -j ACCEPT
```

salvamos configuración

```
iptables-save > /etc/iptables
```

Creamos el usuario de cinder

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

Le damos rol de admin a dicho usuario

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

NOTA: Este comando no emite ningún tipo de salida

Creamos la entidad de servicio para el componente, como posee dos versiones del API, se necesitan dos identidades dentro de keystone

```
openstack service create --name cinderv2 --description "OpenStack Block Storage V2" volumev2

openstack service create --name cinderv3 --description "OpenStack Block Storage V3" volumev3
```

Procedemos a crear los endpoints.

(Versión 2)

```
openstack endpoint create --region RegionOne volumev2 public http://192.168.56.2:8776/v2/%\(project_id\)s

openstack endpoint create --region RegionOne volumev2 internal http://192.168.56.2:8776/v2/%\(project_id\)s

openstack endpoint create --region RegionOne volumev2 admin http://192.168.56.2:8776/v2/%\(project_id\)s
```

(Versión 3)

```
openstack endpoint create --region RegionOne volumev3 public http://192.168.56.2:8776/v3/%\(project_id\)s

 openstack endpoint create --region RegionOne volumev3 internal http://192.168.56.2:8776/v3/%\(project_id\)s

openstack endpoint create --region RegionOne volumev3 admin
```

http://192.168.56.2:8776/v3/%\(project_id\)s

NOTA: Estos endpoint poseen la variable %\(project_id\ que le sirve al SO como expresión regular para identificar el UUID del endpoint específico

2.- Procedemos a instalar el componente y configurarlo

```
yum install openstack-cinder
```

Editamos el archivo de configuración global

```
vim /etc/cinder/cinder.conf
```

En las sección de [database]

```
 # ...
 connection = mysql+pymysql://cinder:rel19224935@192.168.56.2/cinder
```

En la sección [DEFAULT]

```
 # ...
 transport_url = rabbit://openstack:rel19224935@192.168.56.2

# ...
 auth_strategy = keystone

# ...
 my_ip = 192.168.56.2
```

En la sección [keystone_authtoken]

```
# ...
 auth_uri = http://192.168.56.2:5000
```

```
auth_url = http://192.168.56.2:35357
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = cinder
password = rel19224935
```

Por ultimo, la sección de [oslo_concurrency]

```
# ...
lock_path = /var/lib/cinder/tmp
```

Poblamos la base de datos

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

NOTA: Ignoremos el mesajes de obselecencia

Inicializar el servicio de block storage

```
systemctl enable openstack-cinder-api.service openstack-cinder-scheduler.service
 systemctl start openstack-cinder-api.service openstack-cinder-scheduler.service
```

3.- Asignación de backend LVM para cinder

Como nuestro servidor tiene 2 discos adicionales de 8Gb cada uno, procedemos a darles formato, prepararlos mediante el volumen lógico y servilos a cinder vía LVM (iscsi)

yum install lvm2 -y -- Algunas distribuciones modernas ya incluyen toda la paquetería necesaria.

iniciamos el servicio

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

Crear volumen físico para LVM

```
pvcreate /dev/sdb
Physical volume "/dev/sdb" successfully created.
```

Creamos backend de disco

```
vgcreate cinder-volumes /dev/sdb
Volume group "cinder-volumes" successfully created
```

NOTA: El servicio de block storage creara los volumenes lógicos en este backend.

Procedemos a reconfigurar a LVM para que solo vea los dispositivos que están dentro del banckend o grupo "cinder-volumes" que acabamos de crear, de esta manera podemos filtrar el escaneo por default de LVM, el cual escanea a partir de "/" y trata de guardarlos en caché, esto puede causar problemas tanto en el sistema operativo subyacente (el de la instancia), como en los volúmenes del proyecto.

```
vim /etc/lvm/lvm.conf
```

Añadimos el siguiente filtro

```
devices {
 ...
 filter = [ "a/sdb/", "r/.*/"]
```

NOTA: En esta sintaxis, toda sentencia del arreglo que comienza con "a" aceptará la expresión, y las que comiencen con "r" serán rechazadas.

Instalamos componente necesario

```
yum install targetcli.noarch -y
```

Volvemos al archivo de configuración global de cinder, esta vez para configurar la sección [DEFAULT]

```
vim /etc/cinder/cinder.conf
```

```
[DEFAULT]

# ...
 volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
 volume_group = cinder-volumes
 iscsi_protocol = iscsi
 iscsi_helper = lioadm

# ...
 enabled_backends = lvm

# ...
 glance_api_servers = http://192.168.56.2:9292
```

Finalizamos la instalación y configuración reiniciando el servicio

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

NOTA: Es muy importante validar que todos los servicios de cinder estén arriba con **lsof -i :PUERTO**, el cinder-api.service es quien levanta el puerto del endpoint 8776

Procedemos a verificar las operaciones

openstack volume service list

Procedemos a crear un volumen sencillo con openstack y ver como afecta nuestro sistema o backend basado en LVM

openstack volume create --size 1 Test-Volume

Revisamos el sistema de particiones

lsblk

```
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 32G 0 disk
├─sda1 8:1 0 500M 0 part /boot
└─sda2 8:2 0 31,5G 0 part
 ├─centos-root 253:0 0 29,5G 0 lvm /
 └─centos-swap 253:1 0 2G 0 lvm [SWAP]
sdb 8:16 0 8G 0 disk -- Disco Utilizado por CINDER
└─cinder--volumes-volume--54bdfa37--6193--4a55--af37--5febf94075db 253:2 0 1G 0 lvm -- Volumen nuevo creado por CINDER en backend LVM
sdc 8:32 0 8G 0 disk
sr0 11:0 1 1024M 0 rom
```

Listamos el volumen creado

openstack volume list

Listamos los discos (solo los que están basado en un backend LVM) por medio de los utilitarios de LVM

lvdisplay

```
--- Logical volume ---
 LV Path /dev/centos/swap
 LV Name swap
```

VG Name centos
LV UUID NCEouj-zULz-G2kR-lc3a-6Y5A-1Gac-v7u6XF
LV Write Access read/write
LV Creation host, time localhost.localdomain, 2017-07-03 19:06:13 -0400
LV Status available
open 2
LV Size 2,00 GiB
Current LE 512
Segments 1
Allocation inherit
Read ahead sectors auto
- currently set to 8192
Block device 253:1

--- Logical volume ---
LV Path /dev/centos/root
LV Name root
VG Name centos
LV UUID zJ48uw-JTbL-VWcp-II4H-4otN-rqOZ-0rnK4b
LV Write Access read/write
LV Creation host, time localhost.localdomain, 2017-07-03 19:06:14 -0400
LV Status available
open 1
LV Size 29,46 GiB
Current LE 7543
Segments 1
Allocation inherit
Read ahead sectors auto
- currently set to 8192
Block device 253:0

--- Logical volume ---
LV Path /dev/cinder-volumes/volume-54bdfa37-6193-4a55-af37-5febf94075db
LV Name volume-54bdfa37-6193-4a55-af37-5febf94075db
VG Name cinder-volumes
LV UUID i6ZfmK-Rur6-kJem-cEv2-1k75-NaAh-zJxNqv
LV Write Access read/write
LV Creation host, time openstack-ocata, 2017-07-04 15:31:42 -0400
LV Status available
open 0
LV Size 1,00 GiB
Current LE 256
Segments 1
Allocation inherit
Read ahead sectors auto
- currently set to 8192
Block device 253:2

vgdisplay

```
--- Volume group ---
 VG Name centos
 System ID
 Format lvm2
 Metadata Areas 0
 Metadata Sequence No 3
 VG Access read/write
 VG Status resizable
 MAX LV 0
 Cur LV 2
 Open LV 2
 Max PV 0
 Cur PV 1
 Act PV 0
 VG Size 31,51 GiB
 PE Size 4,00 MiB
 Total PE 8066
 Alloc PE / Size 8055 / 31,46 GiB
 Free PE / Size 11 / 44,00 MiB
 VG UUID jkQMdL-M0HM-qSe8-R3gq-q273-AtAO-D7qtjF

 --- Volume group ---
 VG Name cinder-volumes
 System ID
 Format lvm2
 Metadata Areas 1
 Metadata Sequence No 2
 VG Access read/write
 VG Status resizable
 MAX LV 0
 Cur LV 1
 Open LV 0
 Max PV 0
 Cur PV 1
 Act PV 1
 VG Size 8,00 GiB
 PE Size 4,00 MiB
 Total PE 2047
 Alloc PE / Size 256 / 1,00 GiB
 Free PE / Size 1791 / 7,00 GiB
 VG UUID V5cOsA-odKM-ijhN-4v3s-uZsA-AUQz-G2ctVC
```

Ahora procedemos a eliminar dicho volumen creado.

```
openstack volume delete Test-Volume
```

NOTA: de esta manera, al listar con "openstack volume list" ya no aparecerá el volumen reflejado

Comprobamos que ya no existe dicho volumen

```
lsblk
```

```
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 32G 0 disk
├─sda1 8:1 0 500M 0 part /boot
└─sda2 8:2 0 31,5G 0 part
 ├─centos-root 253:0 0 29,5G 0 lvm /
 └─centos-swap 253:1 0 2G 0 lvm [SWAP]
sdb 8:16 0 8G 0 disk
sdc 8:32 0 8G 0 disk
sr0 11:0 1 1024M 0 rom
```

Antes de continuar, vamos agregar a nuestro archivo rc una variable adicional muy importante para que glance y cinder puedan comunicarse via REST

```
echo "export OS_VOLUME_API_VERSION=2" >> /home/admin-openrc
```

lo verificamos y hacemos source del mismo

cat /home/admin-openrc

```
export OS_USERNAME=admin
 export OS_PASSWORD=rel19224935
 export OS_PROJECT_NAME=admin
 export OS_USER_DOMAIN_NAME=Default
 export OS_PROJECT_DOMAIN_NAME=Default
 export OS_AUTH_URL=http://192.168.56.2:35357/v3
 export OS_IDENTITY_API_VERSION=3
 export OS_VOLUME_API_VERSION=2
```

Ahora procedemos a crear un volumen basado en una de nuestras imagenes ya creadas.

openstack volume create --image cirros64 --size 2 myvol1

#################### Instalación Componente Core NEUTRON ############################

1.- Instalación de servicio CORE network (NEUTRON)

Primero preparamos la base de datos como se ha hecho en los demás componentes.

mysql -u root -p

Ahora creamos la base de datos que utilizará cinder

```
MariaDB [(none)]> CREATE DATABASE neutron;
```

Garantizamos los privilegios

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost'
IDENTIFIED BY 'rel19224935';
 MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%'
IDENTIFIED BY 'rel19224935';
```

Aseguramos IPTABLES

```
iptables -A INPUT -p tcp -m multiport --dports 9696 -j ACCEPT
 iptables -A INPUT -p udp -m state --state NEW -m udp --dport 67 -j ACCEPT
 iptables -A INPUT -p udp -m state --state NEW -m udp --dport 68 -j ACCEPT
```

Salvamos la configuración

```
iptables-save > /etc/iptables
 service iptables save
```

Creamos usuario para neutron

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

User Password:
Repeat User Password:

Creamos el rol

```
openstack role add --project service --user neutron admin
```

Creamos la entidad del servicio

```
openstack service create --name neutron --description "OpenStack Networking"
network
```

Ahora procedemos a crear los endpoints para este servicio

```
openstack endpoint create --region RegionOne network public
http://192.168.56.2:9696

openstack endpoint create --region RegionOne network internal
http://192.168.56.2:9696

openstack endpoint create --region RegionOne network admin
```

http://192.168.56.2:9696

2.- Instalación de componentes y programas necesarios

NOTA: A partir de este punto, y para efectos de este laboratorio, no se seguirá la documentación oficial de OpenStack, sino la de la comunidad, específicamente el foro oficial donde publican el código de los componentes y los enlaces instaladores de Rinaldo Martinez @tigerlinux

https://github.com/openstack/
https://github.com/tigerlinux/openstack-ocata-installer-centos7/blob/master/modules/neutroninstall.sh

NOTA: De forma temporal, se debe deshabilitar el repositorio de EPEL, esto se hace con la finalidad de evitar conflictos entre los paquetes de zeromq, EPEL y sus repositorios.

```
yum install -y --disablerepo=epel* openstack-neutron openstack-neutron-openvswitch openstack-neutron-ml2 openstack-utils openstack-selinux python-neutron python-neutronclient haproxy which openstack-neutron-lbaas openstack-neutron-fwaas
```

Ahora para evitar el conflicto entre EPEL y RDO aplicamos un versionlock de los paquetes de zeromq

```
yum -y install yum-plugin-versionlock
 yum versionlock zeromq*
```

Y luego, con zeromq bloqueado, hacemos una segunda corrida de los paquetes de neutron pero con EPEL activado

```
yum install -y openstack-neutron openstack-neutron-openvswitch openstack-neutron-ml2 openstack-utils openstack-selinux python-neutron python-neutronclient haproxy which openstack-neutron-lbaas openstack-neutron-fwaas
```

Creamos un enlace simbólico del plugin a la config de ml2

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

Dado a un error reciente de paquetería, necesitamos hacer un parche del archivo SYSTEMD de Neutron para asegurarnos que Centos 7 Neutron utilizará el plugin ML2 siempre, de lo contrario, fallará.

```
sed -i 's,plugins/ml2/openvswitch_agent.ini,plugin.ini,g'
/usr/lib/systemd/system/neutron-openvswitch-agent.service
```

NOTA: puede suceder que luego de un "yum update" el archivo sea sobre escrito de nuevo a su forma original, provocando el mismo bug. Los siguientes pasos son un procedimiento sobre seguro para garantizar la correcta configuración del plugin ML2.

```
mv /etc/neutron/plugins/openvswitch/ovs_neutron_plugin.ini
/etc/neutron/plugins/openvswitch/ovs_neutron_plugin.ini.ORG

mv /etc/neutron/plugins/ml2/openvswitch_agent.ini
/etc/neutron/plugins/ml2/openvswitch_agent.ini.ORG

ln -f -s /etc/neutron/plugins/ml2/ml2_conf.ini
/etc/neutron/plugins/ml2/openvswitch_agent.ini
```

Y por ultimo para terminar el parche

```
systemctl daemon-reload
```

Como el dhcp-agent de Neutron utiliza dnsmasq para asignar las direcciones ip de las instancias procedemos a instalarlo.

```
yum -y install dnsmasq dnsmasq-utils
```

```
touch /etc/dnsmasq-neutron.conf
 cat /etc/dnsmasq.conf > /etc/dnsmasq-neutron.conf
 mkdir -p /etc/dnsmasq-neutron.d
```

Vaciamos las siguientes configuraciones en dicho archivo

```
echo "user=neutron" >> /etc/dnsmasq-neutron.conf
 echo "group=neutron" >> /etc/dnsmasq-neutron.conf
 echo "group=neutron" >> /etc/dnsmasq-neutron.conf
 echo "conf-dir=/etc/dnsmasq-neutron.d" >> /etc/dnsmasq-neutron.conf
 echo "# Extra options for Neutron-DNSMASQ" > /etc/dnsmasq-neutron.d/neutron-
dnsmasq-extra.conf
 echo "# Samples:" >> /etc/dnsmasq-neutron.d/neutron-dnsmasq-extra.conf
 echo "# dhcp-option=option:ntp-server,192.168.1.1" >> /etc/dnsmasq-
neutron.d/neutron-dnsmasq-extra.conf
 echo "# dhcp-option = tag:tag0, option:ntp-server, 192.168.1.1" >> /etc/dnsmasq-
neutron.d/neutron-dnsmasq-extra.conf
 echo "# dhcp-option = tag:tag1, option:ntp-server, 192.168.1.1" >> /etc/dnsmasq-
neutron.d/neutron-dnsmasq-extra.conf
 echo "# expand-hosts" >> /etc/dnsmasq-neutron.d/neutron-dnsmasq-extra.conf
 echo "# domain=dominio-interno-uno.home,192.168.1.0/24" >> /etc/dnsmasq-
neutron.d/neutron-dnsmasq-extra.conf
 echo "# domain=dominio-interno-dos.home,192.168.100.0/24" >> /etc/dnsmasq-
neutron.d/neutron-dnsmasq-extra.conf
```

Luego procedemos a editar el archivo de configuración principal de neutron

```
echo "#" >> /etc/neutron/neutron.conf

crudini --set /etc/neutron/neutron.conf DEFAULT debug False
 crudini --set /etc/neutron/neutron.conf DEFAULT agent_down_time 60
 crudini --set /etc/neutron/neutron.conf DEFAULT log_dir /var/log/neutron
 crudini --set /etc/neutron/neutron.conf DEFAULT bind_host 0.0.0.0
 crudini --set /etc/neutron/neutron.conf DEFAULT transport_url
rabbit://openstack:rel19224935@192.168.56.2
 crudini --set /etc/neutron/neutron.conf DEFAULT bind_port 9696
 crudini --set /etc/neutron/neutron.conf DEFAULT auth_strategy keystone
 crudini --set /etc/neutron/neutron.conf DEFAULT core_plugin ml2
 crudini --set /etc/neutron/neutron.conf DEFAULT service_plugins
router,firewall,neutron_lbaas.services.loadbalancer.plugin.LoadBalancerPluginv2
 crudini --set /etc/neutron/neutron.conf DEFAULT mac_generation_retries 16
```

```
 crudini --set /etc/neutron/neutron.conf DEFAULT dhcp_lease_duration -1
 crudini --set /etc/neutron/neutron.conf DEFAULT allow_bulk True
 crudini --set /etc/neutron/neutron.conf DEFAULT allow_overlapping_ips True
 crudini --set /etc/neutron/neutron.conf DEFAULT control_exchange neutron
 crudini --set /etc/neutron/neutron.conf DEFAULT default_notification_level INFO
 crudini --set /etc/neutron/neutron.conf DEFAULT host `hostname`
 crudini --set /etc/neutron/neutron.conf DEFAULT default_publisher_id `hostname`
 crudini --set /etc/neutron/neutron.conf DEFAULT notification_topics notifications
 crudini --set /etc/neutron/neutron.conf DEFAULT state_path /var/lib/neutron
 crudini --set /etc/neutron/neutron.conf DEFAULT router_distributed True
 crudini --set /etc/neutron/neutron.conf DEFAULT
notify_nova_on_port_status_changes true
 crudini --set /etc/neutron/neutron.conf DEFAULT notify_nova_on_port_data_changes
true
 crudini --set /etc/neutron/neutron.conf DEFAULT allow_automatic_l3agent_failover
True
 crudini --set /etc/neutron/neutron.conf oslo_concurrency lock_path
/var/lib/neutron/lock
```

Creamos el directorio nuevo para el lock path de oslo_concurrency y otorgamos la permisología correspondiente

```
mkdir -p /var/lib/neutron/lock

chown neutron.neutron /var/lib/neutron/lock

crudini --set /etc/neutron/neutron.conf DEFAULT api_paste_config api-paste.ini
 crudini --set /etc/neutron/neutron.conf DEFAULT global_physnet_mtu 1500
```

Seteamos el sudo wrapper o root_helper

```
crudini --set /etc/neutron/neutron.conf agent root_helper "sudo neutron-rootwrap
/etc/neutron/rootwrap.conf"
```

Pasemos a la configuración del segmento de Keystone

```
crudini --set /etc/neutron/neutron.conf keystone_authtoken auth_uri
```

```
http://192.168.56.2:5000
 crudini --set /etc/neutron/neutron.conf keystone_authtoken auth_url
http://192.168.56.2:35357
 crudini --set /etc/neutron/neutron.conf keystone_authtoken auth_type password
 crudini --set /etc/neutron/neutron.conf keystone_authtoken memcached_servers
192.168.56.2:11211
 crudini --set /etc/neutron/neutron.conf keystone_authtoken project_domain_name
default
 crudini --set /etc/neutron/neutron.conf keystone_authtoken user_domain_name
default
 crudini --set /etc/neutron/neutron.conf keystone_authtoken project_name service
 crudini --set /etc/neutron/neutron.conf keystone_authtoken username neutron
 crudini --set /etc/neutron/neutron.conf keystone_authtoken password rel19224935

crudini --del /etc/neutron/neutron.conf keystone_authtoken identity_uri
 crudini --del /etc/neutron/neutron.conf keystone_authtoken admin_tenant_name
 crudini --del /etc/neutron/neutron.conf keystone_authtoken admin_user
 crudini --del /etc/neutron/neutron.conf keystone_authtoken admin_password
```

Como estamos construyendo un all-in-one tendremos un solo agente dhcp

```
crudini --set /etc/neutron/neutron.conf DEFAULT dhcp_agents_per_network 1
 crudini --set /etc/neutron/neutron.conf DEFAULT dhcp_agent_notification True
```

```
nova_admin_tenant_id=`openstack project show service|grep id|awk '{print $4}'`
```

```
crudini --set /etc/neutron/neutron.conf DEFAULT
notify_nova_on_port_status_changes True
 crudini --set /etc/neutron/neutron.conf DEFAULT notify_nova_on_port_data_changes
True
 crudini --set /etc/neutron/neutron.conf DEFAULT nova_url
http://192.168.56.2:8774/v2.1
 crudini --set /etc/neutron/neutron.conf DEFAULT report_interval 20
 crudini --set /etc/neutron/neutron.conf DEFAULT notification_driver
neutron.openstack.common.notifier.rpc_notifier
```

Imprimimos el valor del cpuinfo para guardarlo en la variable $cpuworkers

```
cpuworkers=`cat /proc/cpuinfo |grep processor|wc -l`
 crudini --set /etc/neutron/neutron.conf DEFAULT api_workers $cpuworkers
```

Ahora a la configuración del segmento de Nova

```
crudini --set /etc/neutron/neutron.conf nova auth_url http://$192.168.56.2:35357
 crudini --set /etc/neutron/neutron.conf nova auth_type password
 crudini --set /etc/neutron/neutron.conf nova project_domain_name default
 crudini --set /etc/neutron/neutron.conf nova user_domain_name default
 crudini --set /etc/neutron/neutron.conf nova region_name RegionOne
 crudini --set /etc/neutron/neutron.conf nova project_name service
 crudini --set /etc/neutron/neutron.conf nova username nova
 crudini --set /etc/neutron/neutron.conf nova password rel19224935
```

Ahora procedemos a configurar Firewall as a service (FWaaS)

```
echo "#" >> /etc/neutron/fwaas_driver.ini

crudini --set /etc/neutron/fwaas_driver.ini fwaas driver
"neutron_fwaas.services.firewall.drivers.linux.iptables_fwaas.IptablesFwaasDriver"
 crudini --set /etc/neutron/fwaas_driver.ini fwaas enabled True
```

Seguimos con el agente L3

```
echo "#" >> /etc/neutron/l3_agent.ini
```

```
crudini --set /etc/neutron/l3_agent.ini DEFAULT debug False
 crudini --set /etc/neutron/l3_agent.ini DEFAULT interface_driver
neutron.agent.linux.interface.OVSInterfaceDriver
 crudini --set /etc/neutron/l3_agent.ini DEFAULT ovs_use_veth True
 crudini --set /etc/neutron/l3_agent.ini DEFAULT use_namespaces True
 crudini --set /etc/neutron/l3_agent.ini DEFAULT handle_internal_only_routers True
 crudini --set /etc/neutron/l3_agent.ini DEFAULT send_arp_for_ha 3
 crudini --set /etc/neutron/l3_agent.ini DEFAULT periodic_interval 40
 crudini --set /etc/neutron/l3_agent.ini DEFAULT periodic_fuzzy_delay 5
```

Funciones deprecadas

```
crudini --set /etc/neutron/l3_agent.ini DEFAULT metadata_port 9697
 crudini --set /etc/neutron/l3_agent.ini DEFAULT enable_metadata_proxy True
 crudini --set /etc/neutron/l3_agent.ini DEFAULT router_delete_namespaces True
```

Como estamos instalando un all-in-one seteamos los siguientes parámetros para asegurar que neutron/openvswitch tengan el mínimo soporte que necesitan para trabajar en un nodo de computo o controller.

```
crudini --set /etc/neutron/l3_agent.ini DEFAULT agent_mode dvr
 crudini --set /etc/neutron/l3_agent.ini DEFAULT agent_mode dvr_snat

sync
```

Configuración del agente DHCP

```
echo "#" >> /etc/neutron/dhcp_agent.ini

crudini --set /etc/neutron/dhcp_agent.ini DEFAULT debug False
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT resync_interval 30
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT interface_driver
neutron.agent.linux.interface.OVSInterfaceDriver
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT ovs_use_veth True
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT dhcp_driver
neutron.agent.linux.dhcp.Dnsmasq
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT ovs_integration_bridge br-int
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT use_namespaces True
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT state_path /var/lib/neutron
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT dnsmasq_config_file
/etc/dnsmasq-neutron.conf
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT dhcp_domain localdomain
 crudini --set /etc/neutron/neutron.conf DEFAULT dns_domain localdomain
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT dhcp_delete_namespaces True
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT root_helper "sudo neutron-
rootwrap /etc/neutron/rootwrap.conf"
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT enable_isolated_metadata True
 crudini --set /etc/neutron/dhcp_agent.ini DEFAULT force_metadata True

sync
```

Configuramos el segmento de conexión hacia la base de datos

```
crudini --set /etc/neutron/neutron.conf database connection =
mysql+pymysql://neutron:rel19224935@192.168.56.2/neutron

crudini --set /etc/neutron/neutron.conf database retry_interval 10
 crudini --set /etc/neutron/neutron.conf database idle_timeout 3600
```

Ahora la configuración del plugin ML2

```
echo "#" >> /etc/neutron/plugins/ml2/ml2_conf.ini

 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2 type_drivers
"local,flat,vlan,gre,vxlan"
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2 mechanism_drivers
"openvswitch,l2population"
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2 tenant_network_types "gre"
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini securitygroup
enable_security_group True
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini securitygroup enable_ipset True
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini securitygroup firewall_driver
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ovs enable_tunneling True
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_vlan
network_vlan_ranges ""
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_flat flat_networks
provider
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ovs local_ip 192.168.56.2
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ovs bridge_mappings
physical01:br-dummy0

crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini agent arp_responder True
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini agent tunnel_types "gre,vxlan"
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini agent vxlan_udp_port "4789"
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini agent l2_population True

crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_vxlan vxlan_group
"239.1.1.1"
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_vxlan vni_ranges
110:1000
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2_type_gre tunnel_id_ranges
1:100
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2 extension_drivers
```

port_security

Configuración de acceso a la base de datos para ML2

```
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini database connection
mysql+pymysql://neutron:rel19224935@192.168.56.2/neutron
```

Y hacemos un tunning de los parámetros de la base de datos, esto con la finalidad de mejorar el performance

```
crudini --set /etc/neutron/neutron.conf database retry_interval 10
 crudini --set /etc/neutron/neutron.conf database idle_timeout 3600
 crudini --set /etc/neutron/neutron.conf database min_pool_size 1
 crudini --set /etc/neutron/neutron.conf database max_pool_size 10
 crudini --set /etc/neutron/neutron.conf database max_retries 100
 crudini --set /etc/neutron/neutron.conf database pool_timeout 10

crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini database retry_interval 10
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini database idle_timeout 3600
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini database min_pool_size 1
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini database max_pool_size 10
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini database max_retries 100
 crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini database pool_timeout 10

 sync
```

Enlace simbólico del plugin

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

Esto puede llegar a presentar otro problema de paquetería, por lo que nos aseguraremos que el archivo se mantenga

```
cp -v /usr/share/neutron/api-paste.ini /etc/neutron/api-paste.ini
```

Seguimos con la configuración de api-paste y el agente de metadata

```
echo "#" >> /etc/neutron/metadata_agent.ini

 crudini --set /etc/neutron/metadata_agent.ini DEFAULT debug False
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT auth_region RegionOne
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT admin_tenant_name service
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT admin_user neutron
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT admin_password
rel19224935
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT nova_metadata_ip
192.168.56.2
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT nova_metadata_port 8775
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT
metadata_proxy_shared_secret rel19224935

crudini --set /etc/neutron/metadata_agent.ini DEFAULT auth_uri
"http://192.168.56.2:5000"
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT auth_url
"http://192.168.56.2:35357"
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT auth_type password
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT project_domain_name
default
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT user_domain_name default
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT project_name service
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT username neutron
 crudini --set /etc/neutron/metadata_agent.ini DEFAULT password rel19224935

 sync
```

Continuamos con Load Balancer as a Service (LBaaS)

```
echo "#" >> /etc/neutron/lbaas_agent.ini
 echo "#" >> /etc/neutron/neutron_lbaas.conf
 echo "#" >> /etc/neutron/services_lbaas.conf
```

Mantendremos los parámetros de la versión 1, solo en caso de retrocompatibilidad, es decir, no están de más

```
crudini --set /etc/neutron/lbaas_agent.ini DEFAULT periodic_interval 10
 crudini --set /etc/neutron/lbaas_agent.ini DEFAULT interface_driver
neutron.agent.linux.interface.OVSInterfaceDriver
 crudini --set /etc/neutron/lbaas_agent.ini DEFAULT ovs_use_veth True
 crudini --set /etc/neutron/lbaas_agent.ini DEFAULT device_driver
neutron_lbaas.drivers.haproxy.namespace_driver.HaproxyNSDriver
```

```
 crudini --set /etc/neutron/lbaas_agent.ini DEFAULT use_namespaces True
 crudini --set /etc/neutron/lbaas_agent.ini haproxy user_group neutron
 crudini --set /etc/neutron/lbaas_agent.ini haproxy send_gratuitous_arp 3

crudini --del /etc/neutron/neutron_lbaas.conf service_providers service_provider
 crudini --del /etc/neutron/neutron_lbaas.conf service_providers service_provider
 crudini --del /etc/neutron/neutron_lbaas.conf service_providers service_provider
 crudini --del /etc/neutron/neutron_lbaas.conf service_providers service_provider
 crudini --del /etc/neutron/neutron_lbaas.conf service_providers service_provider
```

Ahora para la versión 2

```
crudini --set /etc/neutron/neutron_lbaas.conf service_providers service_provider "LO
ADBALANCERV2:Haproxy:neutron_lbaas.drivers.haproxy.plugin_driver.HaproxyOnHo
stPluginDriver:default"

crudini --set /etc/neutron/neutron_lbaas.conf service_auth auth_url
http://192.168.56.2:5000/v3
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth admin_user neutron
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth admin_tenant_name
service
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth admin_password
rel19224935
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth admin_user_domain
default
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth admin_project_domain
default
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth region RegionOne
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth service_name lbaas
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth auth_version 3
 crudini --set /etc/neutron/neutron_lbaas.conf service_auth endpoint_type public
```

Configuración reciente

```
crudini --set /etc/neutron/services_lbaas.conf haproxy periodic_interval 10
 crudini --set /etc/neutron/services_lbaas.conf haproxy interface_driver
neutron.agent.linux.interface.OVSInterfaceDriver
 crudini --set /etc/neutron/services_lbaas.conf haproxy send_gratuitous_arp 3
 crudini --set /etc/neutron/services_lbaas.conf haproxy user_group neutron
```

```
crudini --set /etc/neutron/services_lbaas.conf haproxy jinja_config_template "/usr/lib/
python2.7/site-packages/neutron_lbaas/services/loadbalancer/drivers/haproxy/templ
ates/haproxy.loadbalancer.j2"
```

```
 chown neutron.neutron /etc/neutron/neutron_lbaas.conf
 chown neutron.neutron /etc/neutron/lbaas_agent.ini
 chown neutron.neutron /etc/neutron/services_lbaas.conf
```

```
sync
```

```
mkdir -p /etc/neutron/plugins/services/agent_loadbalancer
 cp -v /etc/neutron/lbaas_agent.ini
/etc/neutron/plugins/services/agent_loadbalancer/
 chown root.neutron
/etc/neutron/plugins/services/agent_loadbalancer/lbaas_agent.ini
```

Detalles finales para la parte de message broker

```
crudini --set /etc/neutron/neutron.conf DEFAULT transport_url
rabbit://openstack:rel19224935@192.168.56.2:5672//openstack
 crudini --set /etc/neutron/neutron.conf DEFAULT notification_driver messagingv2
 crudini --set /etc/neutron/neutron.conf oslo_messaging_notifications driver
messagingv2
```

Populamos la base de datos

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

FWaaS

su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf --config-file /etc/neutron/plugin.ini --subproject neutron-fwaas upgrade head" neutron

LBaaS

su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf --config-file /etc/neutron/plugin.ini --subproject neutron-lbaas upgrade head" neutron

Arrancamos los servicios

systemctl enable neutron-server.service neutron-openvswitch-agent neutron-dhcp-agent.service neutron-metadata-agent.service neutron-lbaasv2-agent.service neutron-l3-agent.service

systemctl start neutron-server.service neutron-openvswitch-agent neutron-dhcp-agent.service neutron-metadata-agent.service neutron-lbaasv2-agent.service neutron-l3-agent.service

Comprobemos funcionamiento listando todas las extensiones que tiene configurado nuestro neutron

[root@openstack-ocata /]# openstack extension list --network

Listamos los agentes

[root@openstack-ocata /]# openstack network agent list

Servicios

[root@openstack-ocata /]# openstack network service provider list

Agentes

[root@openstack-ocata neutron]# openstack network agent list

#################### Instalación Componente Core NOVA ############################

1.- Instalación de servicio CORE network (NOVA)

Primero preparamos la base de datos como se ha hecho en los demás componentes.

mysql -u root -p

Creamos las bases de datos necesarias para nova

```
CREATE DATABASE nova_api;
 CREATE DATABASE nova;
 CREATE DATABASE nova_cell0;
```

Garantizamos los privilegios

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'localhost'
IDENTIFIED BY 'rel19224935';
 MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'%'
IDENTIFIED BY 'rel19224935';
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost'
IDENTIFIED BY 'rel19224935';
 MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY
'rel19224935';
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'localhost'
IDENTIFIED BY 'rel19224935';
 MariaDB [(none)]> GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'%'
IDENTIFIED BY 'rel19224935';
```

Aseguramos IPTABLES

```
iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport 6080 -j ACCEPT
 iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport 6081 -j ACCEPT
 iptables -A INPUT -p tcp -m multiport --dports 5900:5999 -j ACCEPT
 iptables -A INPUT -p tcp -m multiport --dports 8773,8774,8775 -j ACCEPT
```

Salvamos la configuración

```
iptables-save > /etc/iptables
 service iptables save
```

Creamos el usuario para NOVA

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

User Password:
Repeat User Password:

Le damos rol de admin a dicho usuario

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

creamos la entidad del servicio para nova

```
openstack service create --name nova --description "OpenStack Compute" compute
```

Creamos los endpoints para el API de computo (NOVA)

```
openstack endpoint create --region RegionOne compute public
http://192.168.56.2:8774/v2.1
```

```
openstack endpoint create --region RegionOne compute internal
http://192.168.56.2:8774/v2.1
```

openstack endpoint create --region RegionOne compute admin http://192.168.56.2:8774/v2.1

Creamos el usuario para el PLACEMENT

openstack user create --domain default --password-prompt placement

User Password:
Repeat User Password:

Añadimos el usuario de Placement al service project con rol de admin

openstack role add --project service --user placement admin

Le creamos una entrada en el catálogo de servicios al Placement API

openstack service create --name placement --description "Placement API" placement

Procedemos a crear los endpoints para el Placement API

openstack endpoint create --region RegionOne placement public http://192.168.56.2:8778

openstack endpoint create --region RegionOne placement internal http://192.168.56.2:8778

```
openstack endpoint create --region RegionOne placement admin
http://192.168.56.2:8778
```

2.- Instalación y configuración de los componentes.

```
yum install -y openstack-nova-novncproxy openstack-nova-spicehtml5proxy
openstack-nova-compute openstack-nova-common openstack-nova-api openstack-
nova-console openstack-nova-conductor openstack-nova-scheduler openstack-nova-
placement-api python-cinderclient
```

Ahora procedemos a configurar cada uno de los archivos

Primero /etc/nova/nova.conf

Sección keystone_authtoken

```
crudini --set /etc/nova/nova.conf keystone_authtoken auth_uri
http://192.168.56.2:5000
 crudini --set /etc/nova/nova.conf keystone_authtoken auth_url
http://192.168.56.2:35357
 crudini --set /etc/nova/nova.conf keystone_authtoken auth_type password
 crudini --set /etc/nova/nova.conf keystone_authtoken project_domain_name default
 crudini --set /etc/nova/nova.conf keystone_authtoken user_domain_name default
 crudini --set /etc/nova/nova.conf keystone_authtoken project_name service
 crudini --set /etc/nova/nova.conf keystone_authtoken username nova
 crudini --set /etc/nova/nova.conf keystone_authtoken password rel19224935
 crudini --set /etc/nova/nova.conf keystone_authtoken memcached_servers
192.168.56.2:11211
 crudini --set /etc/nova/nova.conf keystone_authtoken region_name RegionOne
```

Main config en la sección DEFAULT

```
crudini --set /etc/nova/nova.conf DEFAULT instance_usage_audit_period hour
 crudini --set /etc/nova/nova.conf DEFAULT log_dir /var/log/nova
 crudini --set /etc/nova/nova.conf DEFAULT state_path /var/lib/nova
 crudini --set /etc/nova/nova.conf DEFAULT volumes_dir /etc/nova/volumes
 crudini --set /etc/nova/nova.conf DEFAULT dhcpbridge /usr/bin/nova-dhcpbridge
```

```
 crudini --set /etc/nova/nova.conf DEFAULT dhcpbridge_flagfile /etc/nova/nova.conf
 crudini --set /etc/nova/nova.conf DEFAULT injected_network_template
/usr/share/nova/interfaces.template
 crudini --set /etc/nova/nova.conf libvirt inject_partition -1
 crudini --set /etc/nova/nova.conf DEFAULT network_manager
nova.network.manager.FlatDHCPManager
 crudini --set /etc/nova/nova.conf DEFAULT iscsi_helper tgtadm
 crudini --set /etc/nova/nova.conf DEFAULT vif_plugging_timeout 10
 crudini --set /etc/nova/nova.conf DEFAULT vif_plugging_is_fatal False
 crudini --set /etc/nova/nova.conf DEFAULT control_exchange nova
 crudini --set /etc/nova/nova.conf DEFAULT host openstack-ocata
 crudini --set /etc/nova/nova.conf cinder os_region_name RegionOne
 crudini --set /etc/nova/nova.conf cinder catalog_info volumev2:cinderv2:internalURL
 crudini --set /etc/nova/nova.conf DEFAULT use_neutron True
 crudini --set /etc/nova/nova.conf cache backend dogpile.cache.memcached
 crudini --set /etc/nova/nova.conf cache enabled True
 crudini --set /etc/nova/nova.conf cache memcache_servers 192.168.56.2:11211
```

Timeouts seguros para cinder

```
crudini --set /etc/nova/nova.conf DEFAULT block_device_allocate_retries 600
 crudini --set /etc/nova/nova.conf DEFAULT block_device_allocate_retries_interval 1
 crudini --set /etc/nova/nova.conf DEFAULT block_device_creation_timeout 300
```

Configuración para la conexión con la base de datos y al api

```
crudini --set /etc/nova/nova.conf database connection
mysql+pymysql://nova:rel19224935@192.168.56.2:3306/nova
 crudini --set /etc/nova/nova.conf api_database connection
mysql+pymysql://nova:rel19224935@192.168.56.2:3306/nova_api
```

Timeouts para la base de datos y el api

```
crudini --set /etc/nova/nova.conf database retry_interval 10
 crudini --set /etc/nova/nova.conf database idle_timeout 3600
 crudini --set /etc/nova/nova.conf database min_pool_size 1
 crudini --set /etc/nova/nova.conf database max_pool_size 10
 crudini --set /etc/nova/nova.conf database max_retries 100
 crudini --set /etc/nova/nova.conf database pool_timeout 10
```

```
crudini --set /etc/nova/nova.conf api_database retry_interval 10
 crudini --set /etc/nova/nova.conf api_database idle_timeout 3600
 crudini --set /etc/nova/nova.conf api_database min_pool_size 1
 crudini --set /etc/nova/nova.conf api_database max_pool_size 10
 crudini --set /etc/nova/nova.conf api_database max_retries 100
 crudini --set /etc/nova/nova.conf api_database pool_timeout 10
```

Placement

```
crudini --set /etc/nova/nova.conf placement os_region_name RegionOne
 crudini --set /etc/nova/nova.conf placement project_domain_name default
 crudini --set /etc/nova/nova.conf placement project_name service
 crudini --set /etc/nova/nova.conf placement auth_type password
 crudini --set /etc/nova/nova.conf placement user_domain_name default
 crudini --set /etc/nova/nova.conf placement auth_url http://192.168.56.2:35357/v3
 crudini --set /etc/nova/nova.conf placement username placement
 crudini --set /etc/nova/nova.conf placement password rel19224935
```

Nucleos que trabajan para el API y las asignaciones de ram, cpu y disco

```
crudini --set /etc/nova/nova.conf DEFAULT compute_driver libvirt.LibvirtDriver
 crudini --set /etc/nova/nova.conf DEFAULT firewall_driver
nova.virt.firewall.NoopFirewallDriver
 crudini --set /etc/nova/nova.conf DEFAULT rootwrap_config /etc/nova/rootwrap.conf
 crudini --set /etc/nova/nova.conf DEFAULT osapi_volume_listen 0.0.0.0
 crudini --set /etc/nova/nova.conf DEFAULT service_down_time 60
 crudini --set /etc/nova/nova.conf DEFAULT image_service
nova.image.glance.GlanceImageService
 crudini --set /etc/nova/nova.conf libvirt use_virtio_for_bridges True
 crudini --set /etc/nova/nova.conf DEFAULT osapi_compute_listen 0.0.0.0
 crudini --set /etc/nova/nova.conf DEFAULT metadata_listen 0.0.0.0
 crudini --set /etc/nova/nova.conf DEFAULT osapi_compute_workers 2
 crudini --set /etc/nova/nova.conf libvirt vif_driver
nova.virt.libvirt.vif.LibvirtGenericVIFDriver
 crudini --set /etc/nova/nova.conf DEFAULT debug False
 crudini --set /etc/nova/nova.conf DEFAULT my_ip 192.168.56.2
 crudini --set /etc/nova/nova.conf wsgi api_paste_config /etc/nova/api-paste.ini
 crudini --set /etc/nova/nova.conf glance api_servers http://192.168.56.2:9292
 crudini --set /etc/nova/nova.conf oslo_concurrency lock_path "/var/oslock/nova"
 crudini --set /etc/nova/nova.conf DEFAULT metadata_host 192.168.56.2
 crudini --set /etc/nova/nova.conf DEFAULT enabled_apis "osapi_compute,metadata"
```

```
 crudini --set /etc/nova/nova.conf libvirt virt_type qemu
 crudini --set /etc/nova/nova.conf libvirt libvirt_type qemu
 crudini --set /etc/nova/nova.conf DEFAULT start_guests_on_host_boot False
 crudini --set /etc/nova/nova.conf DEFAULT resume_guests_state_on_host_boot False
 crudini --set /etc/nova/nova.conf DEFAULT instance_name_template instance-%08x
 crudini --set /etc/nova/nova.conf DEFAULT allow_resize_to_same_host True
 crudini --set /etc/nova/nova.conf DEFAULT ram_allocation_ratio 1.5
 crudini --set /etc/nova/nova.conf DEFAULT cpu_allocation_ratio 16.0
 crudini --set /etc/nova/nova.conf DEFAULT disk_allocation_ratio 1.0
 crudini --set /etc/nova/nova.conf DEFAULT connection_type libvirt
```

Configuración pertinente a Neutron

```
crudini --set /etc/nova/nova.conf neutron url "http://192.168.56.2:9696"
 crudini --set /etc/nova/nova.conf neutron auth_type password
 crudini --set /etc/nova/nova.conf neutron auth_url "http://192.168.56.2:35357"
 crudini --set /etc/nova/nova.conf neutron project_domain_name default
 crudini --set /etc/nova/nova.conf neutron user_domain_name default
 crudini --set /etc/nova/nova.conf neutron region_name RegionOne
 crudini --set /etc/nova/nova.conf neutron project_name service
 crudini --set /etc/nova/nova.conf neutron username neutron
 crudini --set /etc/nova/nova.conf neutron password rel19224935
 crudini --set /etc/nova/nova.conf neutron service_metadata_proxy True
 crudini --set /etc/nova/nova.conf neutron metadata_proxy_shared_secret
rel19224935
 crudini --set /etc/nova/nova.conf DEFAULT linuxnet_ovs_integration_bridge br-int
 crudini --set /etc/nova/nova.conf neutron ovs_bridge br-int
```

Configuración para la consola por VNC

```
crudini --set /etc/nova/nova.conf vnc enabled True
 crudini --set /etc/nova/nova.conf vnc novncproxy_host 0.0.0.0
 crudini --set /etc/nova/nova.conf vnc vncserver_proxyclient_address 192.168.56.2
 crudini --set /etc/nova/nova.conf vnc novncproxy_base_url
"http://192.168.56.2:6080/vnc_auto.html"
 crudini --set /etc/nova/nova.conf vnc novncproxy_port 6080
 crudini --set /etc/nova/nova.conf vnc vncserver_listen 192.168.56.2
 crudini --set /etc/nova/nova.conf vnc keymap es
 crudini --del /etc/nova/nova.conf spice html5proxy_base_url > /dev/null 2>&1
 crudini --del /etc/nova/nova.conf spice server_listen > /dev/null 2>&1
 crudini --del /etc/nova/nova.conf spice server_proxyclient_address > /dev/null
2>&1
 crudini --del /etc/nova/nova.conf spice keymap > /dev/null 2>&1
 crudini --set /etc/nova/nova.conf spice agent_enabled False > /dev/null 2>&1
 crudini --set /etc/nova/nova.conf spice enabled False > /dev/null 2>&1
```

Configuración para el message broker

```
crudini --set /etc/nova/nova.conf DEFAULT transport_url
rabbit://openstack:rel19224935@192.168.56.2:5672//openstack
 crudini --set /etc/nova/nova.conf DEFAULT config_drive_format vfat
 crudini --set /etc/nova/nova.conf libvirt live_migration_tunnelled False
```

Ahora para verificar de manera esquizofrénica y desconfiada que tipo de virtualización nos permite nuestro servidor, crearemos la siguiente rutina:

```
cd /
mkdir workdir
cd workdir
touch comprobation_virt.sh
chmod 755 comprobation_virt.sh
```

Y copiamos el siguiente texto

```
#!/bin/bash
# Comprobación de virtualizador soportado
# Raúl E. Linares N.
#
kvm_possible=`grep -E 'svm|vmx' /proc/cpuinfo|uniq|wc -l`
#
#
if [ $kvm_possible == "0" ]
then
 echo ""
 echo "WARNING !. This server does not support KVM"
 echo "We will have to use QEMU instead of KVM"
 echo "Performance will be poor"
 echo ""
 crudini --set /etc/nova/nova.conf libvirt virt_type qemu
 crudini --set /etc/nova/nova.conf libvirt libvirt_type qemu
 setsebool -P virt_use_execmem on
 ln -s -f /usr/libexec/qemu-kvm /usr/bin/qemu-system-x86_64
 service libvirtd restart
 echo ""
else
 crudini --set /etc/nova/nova.conf libvirt cpu_mode host-passthrough
fi
```

```
#
#
sync
sleep 5
sync
#
# Fin de la rutina de comprobación
```

Lo ejecutamos

```
./comprobation_virt.sh
```

```
 mkdir -p /var/oslock/nova
 chown -R nova.nova /var/oslock/nova
```

Populamos la base de datos del nova-api

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

Registramos la cell0 en la base de datos

```
su -s /bin/sh -c "nova-manage cell_v2 map_cell0" nova
```

Creamos la celda cell1

```
su -s /bin/sh -c "nova-manage cell_v2 create_cell --name=cell1 --verbose" nova
7b5773f0-b313-49eb-9e80-2f9dda5a1575
```

Populamos la base de datos de nova

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

Listamos las celdas utilizando cellv2

nova-manage cell_v2 list_cells

NOTA: La funcionalidad de celdas le permite escalar una nube de OpenStack Compute de una forma más distribuida sin tener que utilizar tecnologías complicadas como la agrupación de colas de datos y de colas de mensajes. Soporta despliegues muy grandes.

Cuando esta funcionalidad está habilitada, los hosts en una nube de OpenStack Compute se dividen en grupos llamados celdas. Las celdas se configuran como un árbol. La celda de nivel superior debe tener un host que ejecute un servicio nova-api, pero no servicios de nova-compute. Cada célula secundaria debe ejecutar todos los servicios típicos de nova- * en una nube de Compute, excepto para nova-api. Puede pensar en celdas como un despliegue de cálculo normal en que cada celda tiene su propio servidor de base de datos y el agente de cola de mensajes.

El servicio nova-cells gestiona la comunicación entre celdas y selecciona celdas para nuevas instancias. Este servicio es necesario para cada celda. La comunicación entre celdas es enchufable, y actualmente la única opción es la comunicación a través de RPC.

La programación de celdas está separada de la programación de host. Nova-cells primero selecciona una celda. Una vez que se selecciona una celda y la nueva solicitud de generación alcanza su servicio de nuevas celdas, se envía al planificador host de esa celda y la compilación continúa como si no tuviera celdas.

Finalizamos la instalación activando todos los servicios

systemctl enable openstack-nova-api openstack-nova-consoleauth openstack-nova-scheduler openstack-nova-conductor openstack-nova-novncproxy openstack-nova-compute httpd

systemctl start openstack-nova-api openstack-nova-consoleauth openstack-nova-scheduler openstack-nova-conductor openstack-nova-novncproxy openstack-nova-compute httpd

Sobre seguro para los logs

chown -R nova.nova /var/log/nova

Configuraremos el de seguridad "default" para el tenant/project "admin" para permitir SSH e ICMP desde todo los orígenes.

openstack security group rule create --proto tcp --remote-ip 0.0.0.0/0 --dst-port 22 default

openstack security group rule create --proto icmp --remote-ip 0.0.0.0/0 --dst-port -1 default

Listamos los servicios del compute dentro de nuestro servidor

openstack compute service list

#################### Instalación Componente Core HORIZON ############################

1.- Instalación de servicio CORE dashboard (HORIZON)

Primero preparamos la base de datos como se ha hecho en los demás componentes.

mysql -u root -p

Ahora creamos la base de datos que utilizará cinder

MariaDB [(none)]> CREATE DATABASE horizon;

Garantizamos los privilegios

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON horizon.* TO 'horizon'@'localhost'
IDENTIFIED BY 'rel19224935';
 MariaDB [(none)]> GRANT ALL PRIVILEGES ON horizon.* TO 'horizon'@'%'
IDENTIFIED BY 'rel19224935';
```

Aplicamos cambios en Iptables

```
iptables -A INPUT -p tcp -m multiport --dports 80,443,11211 -j ACCEPT
```

Salvamos la configuración

```
iptables-save > /etc/iptables
 service iptables save
```

Procedemos a instalar los componentes necesarios

```
yum install -y memcached python-memcached openstack-dashboard
```

Respaldamos el archivo de configuración principal

```
cp /etc/openstack-dashboard/local_settings /etc/openstack-
dashboard/local_settings.old
```

Editamos el archivo de configuración principal

```
vim /etc/openstack-dashboard
```

Y editamos los siguientes parámetros

```
OPENSTACK_HOST = "192.168.56.2"
 ALLOWED_HOSTS = ['*']
```

Para la configuración del motor de sesiones, y que utilice la base de datos mysql se debe realizar la siguiente configuración al final del archivo

```
SESSION_ENGINE = 'django.contrib.sessions.backends.db'
CACHES = {
 'default': {
 'BACKEND': 'django.core.cache.backends.db.DatabaseCache',
 'LOCATION': 'openstack_db_cache',
 }
}

DATABASES = {
 'default': {
 'ENGINE': 'django.db.backends.mysql',
 'NAME': 'horizon',
 'USER': 'horizon',
 'PASSWORD': 'rel19224935',
 'HOST': '192.168.56.2',
 }
}
```

Luego habilitar la versión 3 del API de indentidad

```
OPENSTACK_KEYSTONE_URL = "http://%s:5000/v3" % OPENSTACK_HOST
```

Habilitar soporte multi dominios

```
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
```

Configurar la sección de los API's y sus versiones

```
OPENSTACK_API_VERSIONS = {
 "identity": 3,
 "image": 2,
 "volume": 2,
}
```

Configurar DEFAULT como el dominio default para los usuarios que se creen vía dashboard

OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = "Default"

Configurar USER como el rol por defecto que tomarán los usuarios que se creen vía dashboard

OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user"

Luego procedemos a aprovisionar la base de datos de horizon con las herramientas de Django

```
mkdir -p /var/lib/dash/.blackhole
 /usr/share/openstack-dashboard/manage.py syncdb --noinput > /dev/null 2>&1
 /usr/share/openstack-dashboard/manage.py createcachetable openstack_db_cache
 sleep 5
 /usr/share/openstack-dashboard/manage.py inspectdb
 sleep 5
```

Finalizamos la instalación

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

Una vez realizado esto, se podrá acceder al web browser y por medio de la dirección ip, ingresar al dashboard de horizon que acabamos de instalar

http://192.168.56.2
Solución única ID: #1013
Autor: Raúl Admin
Última actualización: 2018-01-20 10:26