Instituto Tecnológico de Costa Rica

Redes Proyecto #2 - IT

Allan Rojas Saúl Zamora

profesor Kevin Moraga

1 Introducción

El presente proyecto pretende ser una guía para la instalación de una red de servicios segura. Para ello, se hace uso exhaustivo de la tecnología de virtualización con el fin de instalar todo el ambiente en una sola máquina física. Se creará una LAN virtual que será la que ofrece los diversos servicios a través de un gateway que la conectará con una WAN, desde la cual se consumirán los servicios ofrecidos por la LAN.

2 Ambiente de trabajo

- Windows 10
- WIndows Server 2012
- CentOS 7
- OpenSUSE 13.2
- Debian 7
- Debian 8
- OpenBSD 5.8
- Docker
- FreeBSD 10
- FreePBX
- Ubuntu 18 LTS

3 Diseño

4 Instalación de servicios

4.1 Lanzamiento de instancias en OpenStack

NOTA: para que los siguientes pasos apliquen, es requerido tener configurado con anterioridad el proyecto, redes, usuarios, sabores e imágenes disponibles.

- 1. Ingresar al panel de OpenStack.
- 2. Abrir el menú de instancias a la izquierda.
- 3. Hacer clic en el botón de "Launch Instance".
- 4. En "Instance Name" ingresar el nombre que recibirá la instancia.



- 5. En la opción de "Boot Source" escoger "Boot from image (creates a new volume)" y escoger la imagen deseada.
- 6. En la opción de "Image Name" seleccionar la imagen deseada.
- 7. En la opción de "Device Size (GB)" ingresar el tamaño deseado para el almacenamiento en disco de la instancia (las distintas imágenes tienen distintos requerimientos de almacenamiento).
- 8. En la pestaña de de "Access & Seciruty" seleccionar el keypair deseado para accesar la instancia. (Pasos para la creación del keypair serán mostrados más adelante).
- 9. En la pestaña de "Networking" agregar la red default.
- 10. En la pestaña de "Post-creation" se pueden agregar reglas personalizadas.
- 11. Clic en "Launch" para lanzar la instancia.

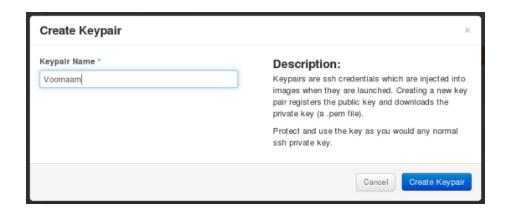
4.1.1 Creación de Keypairs

- 1. Ir al menú de "Access & Security"
- 2. Abrir la pestaña de "Keypairs"
- 3. Hacer clic en el botón de "Create Keypair"



- 4. Escoger un nombre para el keypair
- Seguidamente se pedirá que guarde un archivo .pem. Este archivo debe ser guardado en una ubicación conveniente ya que no será posible descargarlo de nuevo.

NOTA: Distintos sistemas poseen distintos procedimientos para la conversión del keypair.



```
rob@fpbxu18:~$ sudo -i
[sudo] password for rob:
root@fpbxu18:~$

sed ~ir 's/#?PermitRootLog.+/PermitRootLogin yes/' /etc/ssh/sshd_config
```

4.2 FreePBX

NOTA: Todo este proceso debe ser llevado a cabo como administrador. Usar *sudo* luego no va a funcionar. Favor no ignorar esto.

- 1. Ingresar Ubuntu o cambiar a usuario administrador (root).
- 2. Habilitar ingresos ssh como administrador.
- 3. Actualizar el sistema.

```
add-apt-repository ppa:ondrej/php < /dev/null
apt-get update && apt-get upgrade -y
```

4. Instalar dependencias.

```
apt-get install -y openssh-server apache2 mysql-server mysql-client \
mongodb curl sox mpg123 sqlite3 git uuid libodbcl unixodbc unixodbc-bin \
asterisk asterisk-core-sounds-en-wav asterisk-core-sounds-en-g722 \
asterisk-dahdi asterisk-flite asterisk-modules asterisk-mp3 asterisk-mysql \
asterisk-moh-opsound-g722 asterisk-moh-opsound-wav asterisk-opus \
asterisk-voicemail dahdi dahdi-dkms dahdi-linux libapache2-mod-security2 \
php5.6 -cqi php5.6-cli php5.6-cli php5.6-curl php5.6-fpm php5.6-dp php5.6-mbstring \
php5.6-mysql php5.6-odbc php5.6-xml php5.6-bcmath php-pear libicu-dev gcc \
g++ make postfix libapache2-mod-php5.6
```

```
curl -sL https://deb.nodesource.com/setup_10.x | bash -
apt-get install -y nodejs
```

- 5. Instalar NodeJS.
- 6. Arreglar los permisos del usuario asterisk

```
useradd -m asterisk
chown asterisk. /var/run/asterisk
chown -R asterisk. /etc/asterisk
chown -R asterisk. /var/(lib,log,spool)/asterisk
chown -R asterisk. /usr/lib/asterisk
chown -R asterisk asterisk
m -rf /var/wws/html
```

7. Remover ejemplos de archivos config resultantes y arreglar errores

```
rm -rf /etc/asterisk/ext* /etc/asterisk/sip* /etc/asterisk/pj* /etc/asterisk/iax* /etc/asterisk/manager*
sed -i 's/.!.//' /etc/asterisk/asterisk.conf
```

8. Actualizar la configuración de Apache

```
sed -i 's/\(^upload_max_filesize = \).'/\120M/' /etc/php/5.6/cgi/php.ini
sed -i 's/www-data/asterisk/' /etc/apache2/envvars
sed -i 's/kllowOverride None/AllowOverride All/' /etc/apache2/apache2.conf
a2enmod rewrite
service apache2 restart
```

9. Arreglar problema de compatibilidad de Pear-GetOpt

```
sed -i 's/ each(/ @each(/' /usr/share/php/Console/Getopt.php
```

10. Instalar MySQL ODBC Connector

- 11. Configurar el ODBC
- 12. Descargar e instalar FreePBX 14

```
cat > /etc/odbc.ini << EOF
[MySQL-asteriakcdrdb]
Description=MySQL connection to 'asteriakcdrdb' database
driver=MySQL
server=localhost
database=asteriakcdrdb
Port=3306
Socket=/var/run/mysqld/mysqld.sock
option=3
Charset=utf8
EOF
cat > /etc/odbcinst.ini << EOF
[MySQL]
Description=ODBC for MySQL
Driver=/usr/lib/odbc/libodbcmy5S.so
FileUsage=1
EOF
```

```
cd /usr/src
wget http://mirror.freepbx.org/modules/packages/freepbx/freepbx-14.0-latest.tgz
tar zxf freepbx-14.0-latest.tgz
cd freepbx
./install -n
```

4.3 CentOS 7

4.3.1 MySQL Server

1. Asegurarse que wget esté instalado

```
$ yum install -y wget
```

- 2. Navegar al folder en el que se desea guardar la descarga
 - \$ wget <link de descarga>
- 3. Agregar el repositorio Yum de MySQL
 - \$ sudo rpm -Uvh <enter the package name here>
- 4. Instalar MySQL
 - \$ sudo yum install mysql-community-server
- 5. Configurar MySQL
 - \$ sudo systemctl start mysqld.service
 - \$ sudo grep temporary password / var/log/mysqld.log
 - \$ mysql -uroot -p
 - $\ \$ ALTER USER 'root'@'localhost' IDENTIFIED BY 'mypassword';

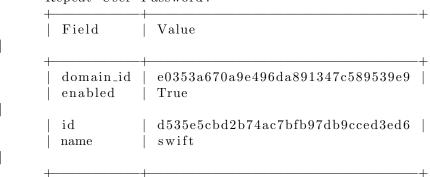
4.3.2 OpenStack Controller

Prerequisitos

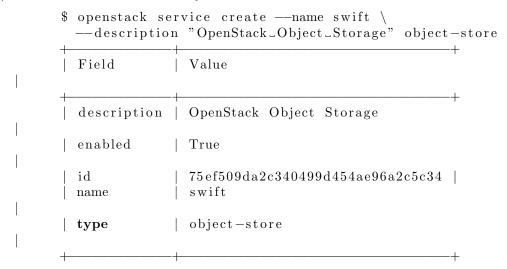
- Proveer credenciales de administrador para dar acceso de comandos CLI solo a administradores
 - \$. admin-openro
- 2. Para crear credenciales de servicio, hay que completar los siguientes pasos:
 - (a) Crear un usuario swift

 $\$ open stack user create —domain default —password—prompt swift User Password:

Repeat User Password:



- (b) Agregar el rol admin al usuario swift (este comando no provee salida)
 - \$ openstack role add --project service --user swift admin
- (c) Crear la entidad de servicio swift



3. Crear los accesos API para el servicio de almacenamiento de objetos \$ openstack endpoint create — region RegionOne \ object-store public http://controller: $8080/v1/AUTH_{\sim}(tenant_id)s$ Field Value True enabled | id 12bfd36f26694c97813f665707114e0d | interface public region RegionOne | region_id RegionOne | service_id 75ef509da2c340499d454ae96a2c5c34 service_name swift | service_type | object-store http://controller:8080/v1/AUTH_%(tenant_id)s | url\$ openstack endpoint create — region RegionOne \ $object-store internal http://controller:8080/v1/AUTH_% (tenant_id) s$ Field Value enabled True | id $7\,a36\,bee6733\,a4\,b5590\,d74\,d3080\,ee6789$ | interface | internal region | RegionOne | region_id RegionOne

| 75ef509da2c340499d454ae96a2c5c34

| service_id

service_name swift

Página 7

```
| service_type | object-store
                http://controller:8080/v1/AUTH_%(tenant_id)s |
 url
$ openstack endpoint create — region RegionOne \
  object-store admin http://controller:8080/v1
 Field
                Value
 enabled
                True
                ebb72cd6851d4defabc0b9d71cdca69b
interface
                admin
                RegionOne
region
| region_id
               RegionOne
                75ef509da2c340499d454ae96a2c5c34
 service_id
 service_name
                swift
service_type | object-store
               http://controller:8080/v1
url
```

Instalación y configuración de componentes

1. Instalar los paquetes

```
\ yum install openstack—swift—proxy python—swiftclient \ python—keystoneclient python—keystonemiddleware \ memcached
```

- 2. Obtener el archivo de configuración para el servicio de proxy desde el repositorio fuente del Objeto de ALmacenamiento
 - $\$ \# curl o /etc/swift/proxy-server.conf \ https://git.openstack.org/cgit/server.conf$
- 3. Editar /etc/swift/proxy-server.conf y completar los siguientes pasos:
 - (a) En la sección [DEFAULT], configurar el puerto de enlace, usuario y directorio de configuración

```
[DEFAULT]
...
bind_port = 8080
user = swift
swift_dir = /etc/swift
```

(b) En la sección [pipeline:main], remover los módulos tempurl y tempauth y agregar los módulos authtoken y keystoneauth

```
[pipeline:main]
pipeline = catch_errors gatekeeper healthcheck proxy-logging cache
```

(c) En la sección [app:proxy-server], habilitar la creación automática de cuentas

```
[app:proxy-server]
use = egg:swift#proxy
...
account_autocreate = True
```

(d) En la sección [filter:keystoneauth], configurar los roles de operador

```
[ filter:keystoneauth]
use = egg:swift#keystoneauth
...
operator_roles = admin,user
```

(e) En la sección [filter:authtoken], configurar el acceso para el sevicio de identidad

```
[filter:authtoken]
paste.filter_factory = keystonemiddleware.auth_token:filter_factory
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = swift
password = SWIFT_PASS
delay_auth_decision = True
```

(f) En la sección [filter:cache], configurar la ubicación de memcached

```
[ filter:cache]
use = egg:swift#memcache
...
memcache_servers = controller:11211
```

4.4 Docker - Owncloud

- 1. Crear un nuevo folder para el proyecto
- 2. Descargar docker-compose.yml del repositorio de own Cloud Docker en Git
Hub.
- 3. Crear un archivo de configuración .env para la información de configuración
- 4. Iniciar el contenedor

```
# Create a new project directory

$ mkdir owncloud-docker-server

$ cd owncloud-docker-server

# Copy docker-compose.yml from the GitHub repository

$ wget https://raw.githubusercontent.com/owncloud-docker/server/master/docker-

# Create the environment configuration file
cat << EOF > .env

OWNCLOUD.VERSION=10.0

OWNCLOUD.DOMAIN=localhost

ADMIN.USERNAME=admin
ADMIN.PASSWORD=admin

HTTP.PORT=8080

EOF

# Build and start the container
```

5 Bitácora de trabajo

\$ docker-compose up -d

5.1 Allan Rojas

- 29-09-2018:
 - 3 horas Buzzer Python
- 29-09-2018:
 - 4 horas Database for Node Directory with MariaDB
- 30-09-2018:
 - 4 horas Java Audio Listening Programming

- 30-09-2018:
 - 3 horas Media Access to Raspberry
- 01-10-2018:
 - 5 horas Mac Address Get / Add to Package
- 10-10-2018:
 - 6 horas Package Generation and Onion Routing
- 19-10-2018:
 - 4 horas Scapy Implementation add to Buzzer
- 29-10-2018:
 - 6 horas Scapy Implementation add to Buzzer

Total de Horas Trabajadas: 35

5.2 Saúl Zamora

- 10-11-2018:
 - 4 horas Investigar OpenStack. Tratar de ingresar.
- 17-11-2018:
 - 1 hora Creación de máquina virtual de Ubuntu 18.
 - 1 hora Configuración de VPN en máquina virtual de Ubuntu.
 - 2 horas Ingreso a OpenStack dashboard. Primeros intentos de crear instancias.
- 18-11-2018:
 - 4 horas Creación de instancias en OpenStack. Constantes errores. Sin éxito.
- 19-11-2018:
 - 2 horas Creación de instancias en OpenStack. Constantes errores.
 Sin éxito.
- 20-11-2018:
 - 2 horas Creación de instancias en OpenStack. Constantes errores. Sin éxito.
- 23-11-2018:

- 2 horas Creación de instancias en OpenStack. Constantes errores. Sin éxito.
- 24-11-2018:
 - 3 horas Instalación de FreePBX en máquina virtual de Ubuntu.
- 25-11-2018:
 - 4 horas Documentación.
- 26-11-2018:
 - 4 horas Documentación.
- 27-11-2018:
 - 4 horas Documentación.
- 28-11-2018:
 - 4 horas Documentación.

Total de horas trabajadas: 37 horas.

6 Comentarios finales

- Debido a la falta de configuraciones para el acceso a la VPN no fue posible dar inicio al proyecto desde que fue entregado el enunciado.
- Constantes problemas con el funcionamiento de OpenStack hicieron imposible la creación de instancias de cualquier tipo y por ende la configuración de cualquier servicio utilizando virtualización.

7 Conclusiones

• El uso de virtualización es muy útil en el desarrollo y configuración de servicios (al igual que el desarrollo de software en general) para la optimización de recursos y para invisibilizar las diferencias de hardware en el sistema al usuario final.

•

References

- [1] Wiki.freepbx.org. (2018). Installing FreePBX 14 on Ubuntu 18.04 FreePBX OpenSource Project Documentation. [online] Available at: https://wiki.freepbx.org/display/FOP/Installing+FreePBX+14+on+Ubuntu+18.04
- [2] GitHub. (2018). naturalis/openstack-docs. [online] Available at: https://github.com/naturalis/openstack-docs/wiki/Howto:-Deploy-a-Windows-image
- [3] GitHub. (2018). naturalis/openstack-docs. [online] Available at: https://github.com/naturalis/openstack-docs/wiki/Howto:-Creating-and-using-OpenStack-SSH-keypairs-on-Windows
- [4] GitHub. (2018). naturalis/openstack-docs. [online] Available at: https://github.com/naturalis/openstack-docs/wiki/Howto:-Creating-and-using-OpenStack-SSH-keypairs-on-Linux-and-OSX
- [5] Docs.openstack.org. (2018). OpenStack Docs: Install and configure the controller node for Red Hat Enterprise Linux and CentOS. [online] Available at: https://docs.openstack.org/project-install-guide/object-storage/newton/controller-install-rdo.html