

## Parcial 1

### Sistemas Distribuidos

#### Servicio WEB

Para el servicio web decidi utilizar el framework Flask hecho en el lenguaje de programación Python, ya que en el pasado ya había utilizado este framework para realizar algunos servicios WEB.

Para aprovisionar el servidor para que corra este servicio Web en Flask es necesario primero asegurarse que Python este instalado, para la conexión del servicio Web con el servidor de Base de datos Postgres se debe usar una librería de Python llamada psycopg2, para que se pueda instalar esta librería es necesario primero instalar gcc, postgres-devel y python-devel.

```
execute 'install-python' do  
  command 'yum install -y python'  
end
```

```
execute 'install gcc' do  
  command 'yum install -y gcc'  
end
```

```
execute 'install postgresql-devel' do  
  command 'yum install -y python-devel postgresql-devel'  
end
```

Para poder instalar la librería pycopg2, Flask, virtualenv y las demás necesarias para realizar el despliegue del servicio Web y la conexión a la base de datos, es necesario instalar pip, que es el gestor de librerías de Python. Después de haber instalado pip se procede a crear el usuario que va a correr el servicio Web, en este caso un usuario llamado flask. Para desplegar el servicio de manera limpia y ordenada es necesario crear un ambiente aparte a través de virtualenv, así que se procede a crear el ambiente con el nombre flask\_env.

```
execute 'get-pip' do  
  command 'wget https://bootstrap.pypa.io/get-pip.py'  
end
```

```
execute 'install-pip' do  
  command 'python get-pip.py'  
end
```

```
execute 'install-virtualenv' do  
  command 'pip install virtualenv'
```

**end**

```
execute 'create-user-flask' do  
  command 'useradd -ms /bin/bash flask'  
end
```

```
execute 'create-virtualenv' do  
  command 'virtualenv flask_env'  
  cwd '/home/flask'  
  user 'flask'  
end
```

Para importar el servicio Web, el archivo con el nombre de las librerías necesarias y el script que iniciara el servicio Web cada vez que se inicie el sistema se usa la instrucción cookbook file, y se copian dichos archivos en una carpeta creada en el servidor.

```
directory '/home/flask/smartlabs' do  
  owner 'flask'  
  group 'flask'  
  mode '0755'  
  action :create  
end
```

```
cookbook_file "/home/flask/smartlabs/requirements.txt" do  
  source "requirements.txt"  
  mode 0644  
  owner "flask"  
  group "flask"  
end
```

```
cookbook_file "/home/flask/smartlabs/myselect.py" do  
  source "myselect.py"  
  mode 0644  
  owner "flask"  
  group "flask"  
end
```

```
cookbook_file "/home/flask/smartlabs/script.sh" do  
  source "script.sh"  
  mode 0777  
  owner "flask"  
  group "flask"  
end
```

Para correr el servicio Web cada vez que se encienda el servidor es necesario insertar el llamado al script importado al archivo rc.local ubicado en la carpeta /etc, a su vez es necesario abrir el puerto a usar para las conexiones, en este caso el puerto 5000, y por último activar el servicio.

```

execute 'run-service-onboot' do
  command 'echo "sh /home/flask/smartlabs/script.sh" >> /etc/rc.local'
  user 'root'
end

bash "open port" do
  user "root"
  code <<-EOH
  iptables -I INPUT 1 -p tcp -m state --state NEW -m tcp --dport 5000 -j ACCEPT
  service iptables save
  EOH
end

execute 'activate-virtualenv' do
  command 'sh /home/flask/smartlabs/script.sh'
  user 'root'
end

```

## **Base de Datos**

Para la instalación de la base de datos Postgres en el servidor es necesario primero habilitar el repositorio donde se encuentra la versión que se requiere instalar en este caso la 9.5.4, después de esto se procede a instalarla a través del gestor de paquetes YUM.

```

execute 'postgresql-repo' do
  command 'sudo rpm -Uvh http://yum.postgresql.org/9.5/redhat/rhel-7-x86_64/pgdg-centos95-9.5-2.noarch.rpm'
end

```

```

execute 'postgresql-install' do
  command 'sudo yum -y install postgresql95-server.x86_64 postgresql95'
end

```

Después de instalado, es necesario iniciar el servicio y configurarlo para que inicie cada vez que se encienda el servidor.

```

execute 'postgresql-onboot' do
  command 'sudo chkconfig postgresql-9.5 on'
end

```

```

service "postgresql-9.5" do
  action [ :enable, :start ]
end

```

Para alojar la base de datos y asignársela a un usuario diferente a vagrant o root, se debe crear un usuario.

```
execute 'create-user-admin' do  
  command 'useradd -ms /bin/bash admin'  
end
```

Para la creación de la base de datos existe un script sql llamado create\_schema.sql, para copiarlo al servidor se usa la instrucción cookbook file y se procede luego a ejecutar el script como el usuario postgres.

```
cookbook_file "/tmp/create_schema.sql" do  
  source "create_schema.sql"  
  mode '0777'  
  owner "vagrant"  
  group "vagrant"  
  action :create  
end
```

```
bash "create schema" do  
  user "root"  
  cwd "/tmp"  
  code <<-EOH  
  cat create_schema.sql | sudo -u postgres psql  
  EOH  
end
```

Para que postgres acepte conexiones remotas, es necesario modificar el archivo postgresql.conf, en este caso decidí reemplazar el archivo, primero eliminándolo y luego copiarlo a través de la instrucción cookbook file

```
file '/var/lib/pgsql/9.5/data/postgresql.conf' do  
  action :delete  
end
```

```
cookbook_file "/var/lib/pgsql/9.5/data/postgresql.conf" do  
  source "postgresql.conf"  
  mode '0644'  
  owner "postgres"  
  group "postgres"  
  action :create  
end
```

Para que el servicio de postgres solo admita conexiones desde la red en la que se encuentra se añade al archivo pg\_hba.conf la linea : host all all 192.168.0.0/24 trust. Para que los cambios surtan efecto es necesario reiniciar el servicio.

```
execute 'trust-access' do  
  command 'echo "host all all 192.168.0.0/24 trust" >>  
/var/lib/pgsql/9.5/data/pg_hba.conf'  
  user "root"  
end
```

```
execute 'restart-postgresql' do  
  command 'sudo service postgresql-9.5 restart'  
end
```

Por ultimo se debe abrir el puerto de postgres en el firewall.

```
bash "open port i" do  
  user "root"  
  code <<-EOH  
  iptables -I INPUT 1 -p tcp -m state --state NEW -m tcp --dport 5432 -j ACCEPT  
  service iptables save  
  EOH  
end
```

```
bash "open port o" do  
  user "root"  
  code <<-EOH  
  iptables -I OUTPUT 1 -p tcp -m state --state NEW -m tcp --dport 5432 -j ACCEPT  
  service iptables save  
  EOH  
end
```

## Evidencias del Servicio Funcionando:

```
camilo@Camilo-PC: ~/Documentos/SistemasDistribuidos/parcial1
camilo@Camilo-PC: ~/Documentos/SistemasDistribuidos/parcial1 115x32
==> centos_web_flask: Collecting psycpg2==2.6.2 (from -r /home/flask/smartlabs/requirements.txt (lin
e 6))
==> centos_web_flask: Downloading psycpg2-2.6.2.tar.gz (376kB)
==> centos_web_flask: Building wheels for collected packages: Flask, itsdangerous, MarkupSafe, psycop
g2
==> centos_web_flask: Running setup.py bdist_wheel for Flask: started
==> centos_web_flask: Running setup.py bdist_wheel for Flask: finished with status 'done'
==> centos_web_flask: Stored in directory: /root/.cache/pip/wheels/b6/09/65/5fcf16f74f334a215447c26
769e291c41883862fe0dc7c1430
==> centos_web_flask: Running setup.py bdist_wheel for itsdangerous: started
==> centos_web_flask: Running setup.py bdist_wheel for itsdangerous: finished with status 'done'
==> centos_web_flask: Stored in directory: /root/.cache/pip/wheels/fc/a8/66/24d655233c757e178d45dea
2de22a04c6d92766abfb741129a
==> centos_web_flask: Running setup.py bdist_wheel for MarkupSafe: started
==> centos_web_flask: Running setup.py bdist_wheel for MarkupSafe: finished with status 'done'
==> centos_web_flask: Stored in directory: /root/.cache/pip/wheels/a3/fa/dc/0198eed9ad95489b8a4f45d
14dd5d2aee3f8984e46862c5748
==> centos_web_flask: Running setup.py bdist_wheel for psycpg2: started
==> centos_web_flask: Running setup.py bdist_wheel for psycpg2: finished with status 'done'
==> centos_web_flask: Stored in directory: /root/.cache/pip/wheels/49/47/2a/5c3f874990ce267228c2dfe
7a0589f3b0651aa590e329ad382
==> centos_web_flask: Successfully built Flask itsdangerous MarkupSafe psycpg2
==> centos_web_flask: Installing collected packages: Werkzeug, MarkupSafe, Jinja2, itsdangerous, Flas
k, psycpg2
==> centos_web_flask: Successfully installed Flask-0.10.1 Jinja2-2.8 MarkupSafe-0.23 Werkzeug-0.11.9
itsdangerous-0.24 psycpg2-2.6.2
==> centos_web_flask: * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
==> centos_web_flask: 192.168.0.12 - - [10/Oct/2016 07:23:44] "GET /hi HTTP/1.1" 200 -
==> centos_web_flask: 192.168.0.12 - - [10/Oct/2016 07:23:55] "GET /select HTTP/1.1" 200 -
```



## **Problemas encontrados:**

- Cuando iba a instalar la librería psycpg2, pip arrojaba un error, al investigar acerca del error en internet descubrí que era necesario instalar python-devel y postgres-devel. Al realizar la instalación de nuevo ocurrió otro problema y en el error se podía leer algo de gcc. Así que procedí a revisar si por defecto la máquina virtual lo tenía instalado, al encontrar que no lo tenía instalado procedí a instalarlo. Después de esto la librería psycpg2 podía instalarse sin problemas.
- Al correr el servicio web que inicialmente se llamaba select.py arrojaba un error muy extraño, al investigar no encontraba el mismo error que tenía pero si algunos parecidos, una solución que daban para un problema parecido era renombrar el archivo, así que lo intenté y el servicio web inició sin problemas.

## **Dirección del Repositorio:**

<https://github.com/camiloamezquita95/Parcial-1-Distribuidos>