

Actividad B5 – GONZALO OSCO HERNANDEZ

Crear un contenedor, sobre el realizar un conjunto de 10 acciones cualesquiera.

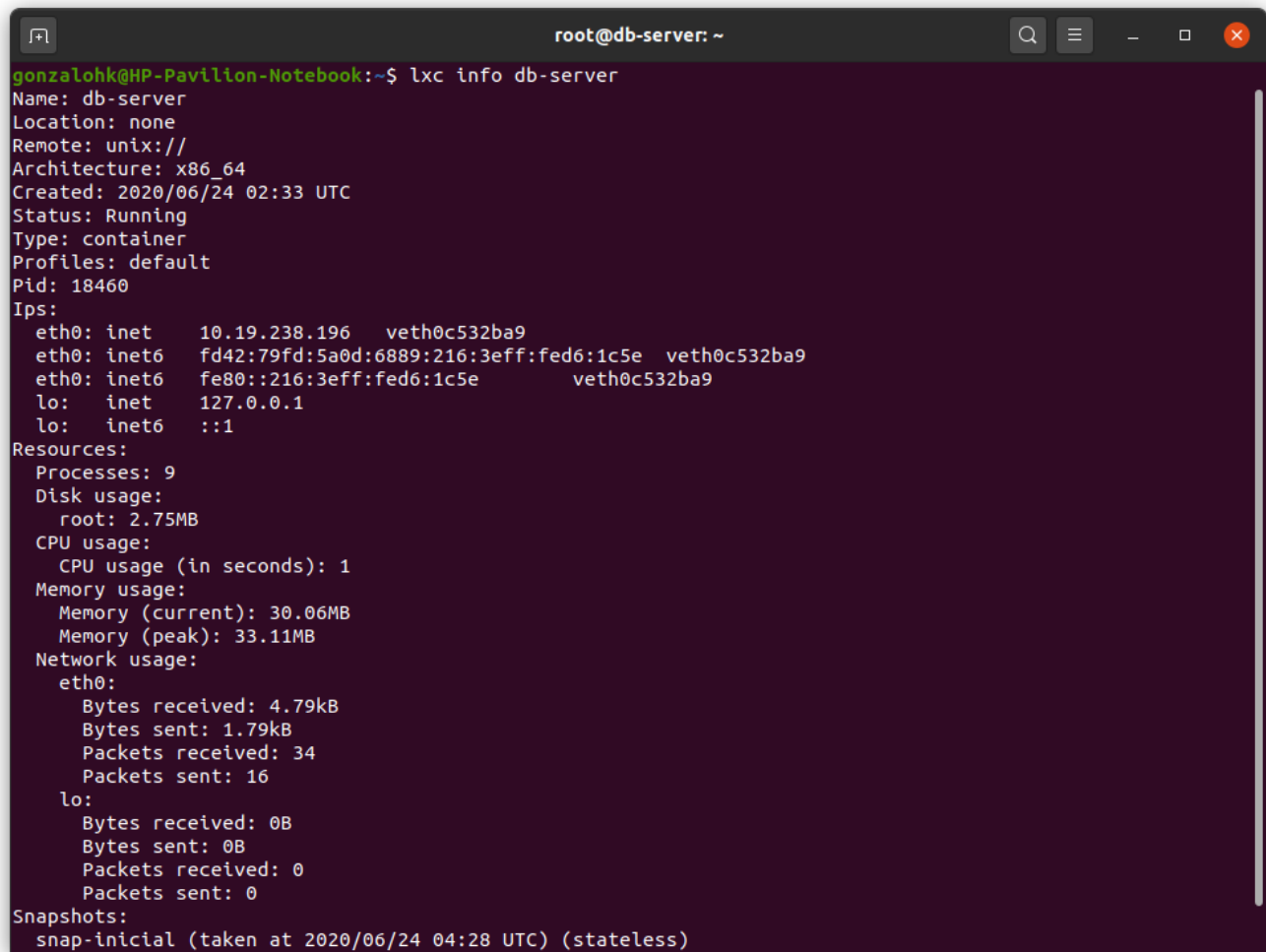
Crear 3 snapshots, que representen 3 hitos dentro de las acciones realizadas

Creamos un nuevo contenedor llamado db-server en base a una imagen ubuntu16.04

```
lxc launch images:ubuntu/16.04 db-server
```

En un primera instancia ingresamos al bash y seguidamente creamo el primer snapshot denominado **snap-inicial**.

```
lxc exec db-server bash  
lxc snapshot db-server snap-inicial
```



```
root@db-server: ~  
gonzalohk@HP-Pavilion-Notebook:~$ lxc info db-server  
Name: db-server  
Location: none  
Remote: unix://  
Architecture: x86_64  
Created: 2020/06/24 02:33 UTC  
Status: Running  
Type: container  
Profiles: default  
Pid: 18460  
Ips:  
  eth0: inet    10.19.238.196    veth0c532ba9  
  eth0: inet6   fd42:79fd:5a0d:6889:216:3eff:fed6:1c5e veth0c532ba9  
  eth0: inet6   fe80::216:3eff:fed6:1c5e    veth0c532ba9  
  lo:  inet     127.0.0.1  
  lo:  inet6    ::1  
Resources:  
  Processes: 9  
  Disk usage:  
    root: 2.75MB  
  CPU usage:  
    CPU usage (in seconds): 1  
  Memory usage:  
    Memory (current): 30.06MB  
    Memory (peak): 33.11MB  
  Network usage:  
    eth0:  
      Bytes received: 4.79kB  
      Bytes sent: 1.79kB  
      Packets received: 34  
      Packets sent: 16  
    lo:  
      Bytes received: 0B  
      Bytes sent: 0B  
      Packets received: 0  
      Packets sent: 0  
Snapshots:  
  snap-inicial (taken at 2020/06/24 04:28 UTC) (stateless)
```

Instalaremos la base de datos **PostgreSQL** por lo que seguimos los siguientes pasos.

Actualizamos los repositorios e instalamos nano y wget.

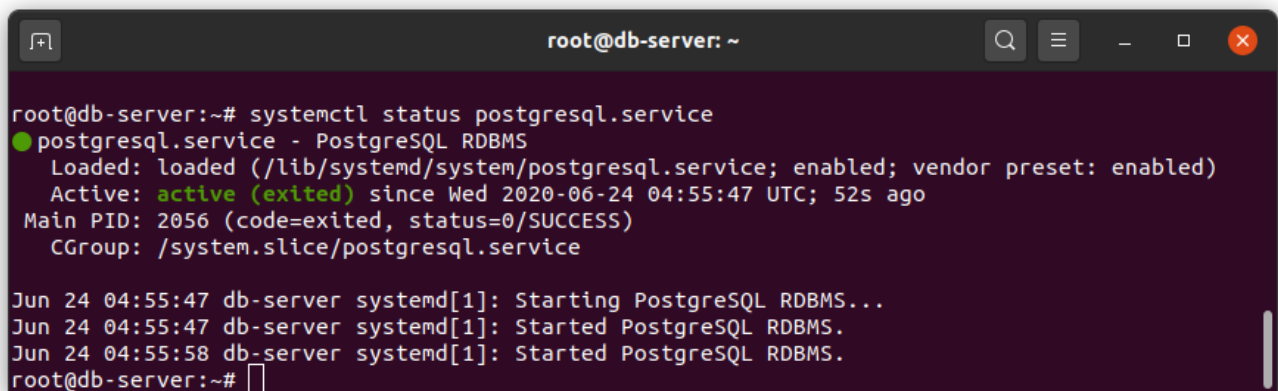
```
apt update
apt -y install nano bash-completion wget
apt -y upgrade
reboot
```

Importamos las llaves GPG necesarias

```
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key
add -
echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" | sudo tee
/etc/apt/sources.list.d/pgdg.list
```

Instalamos la PostgreSQL, para luego verificar el estado de la misma

```
apt update
apt -y install postgresql-12 postgresql-client-12
systemctl status postgresql.service
```



```
root@db-server: ~
root@db-server:~# systemctl status postgresql.service
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; vendor preset: enabled)
   Active: active (exited) since Wed 2020-06-24 04:55:47 UTC; 52s ago
   Main PID: 2056 (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/postgresql.service

Jun 24 04:55:47 db-server systemd[1]: Starting PostgreSQL RDBMS...
Jun 24 04:55:47 db-server systemd[1]: Started PostgreSQL RDBMS.
Jun 24 04:55:58 db-server systemd[1]: Started PostgreSQL RDBMS.
root@db-server:~#
```

Creamos una base de datos ejemplo denominada db-test

```
sudo su - postgres
psql -c "alter user postgres with password '123456'"
psql
postgres=# CREATE DATABASE dbtest;
postgres=# CREATE USER usertest WITH ENCRYPTED PASSWORD '123456';
postgres=# GRANT ALL PRIVILEGES ON DATABASE dbtest to usertest;
```

```
root@db-server: ~  
postgres=# \l  
List of databases  
+-----+-----+-----+-----+-----+-----+  
Name      | Owner   | Encoding | Collate | Ctype   | Access privileges  
+-----+-----+-----+-----+-----+-----+  
dbtest     | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 | =Tc/postgres +  
           |          |          |             |             | postgres=Ctc/postgres+  
           |          |          |             |             | usertest=Ctc/postgres  
postgres   | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +  
template0  | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 | postgres=Ctc/postgres  
template1  | postgres | UTF8      | en_US.UTF-8 | en_US.UTF-8 | =c/postgres +  
           |          |          |             |             | postgres=Ctc/postgres  
(4 rows)  
postgres=#
```

Editamos el archivo de configuración para que cualquier conexión externa pueda conectarse a nuestra base de datos y probamos.

```
sudo nano /etc/postgresql/12/main/postgresql.conf  
listen_addresses = '*'  
sudo systemctl restart postgresql  
netstat -tunelp | grep 5432
```

Creamos un segundo snapshot denominado **snap-db**.

```
lxc snapshot db-server snap-db
```

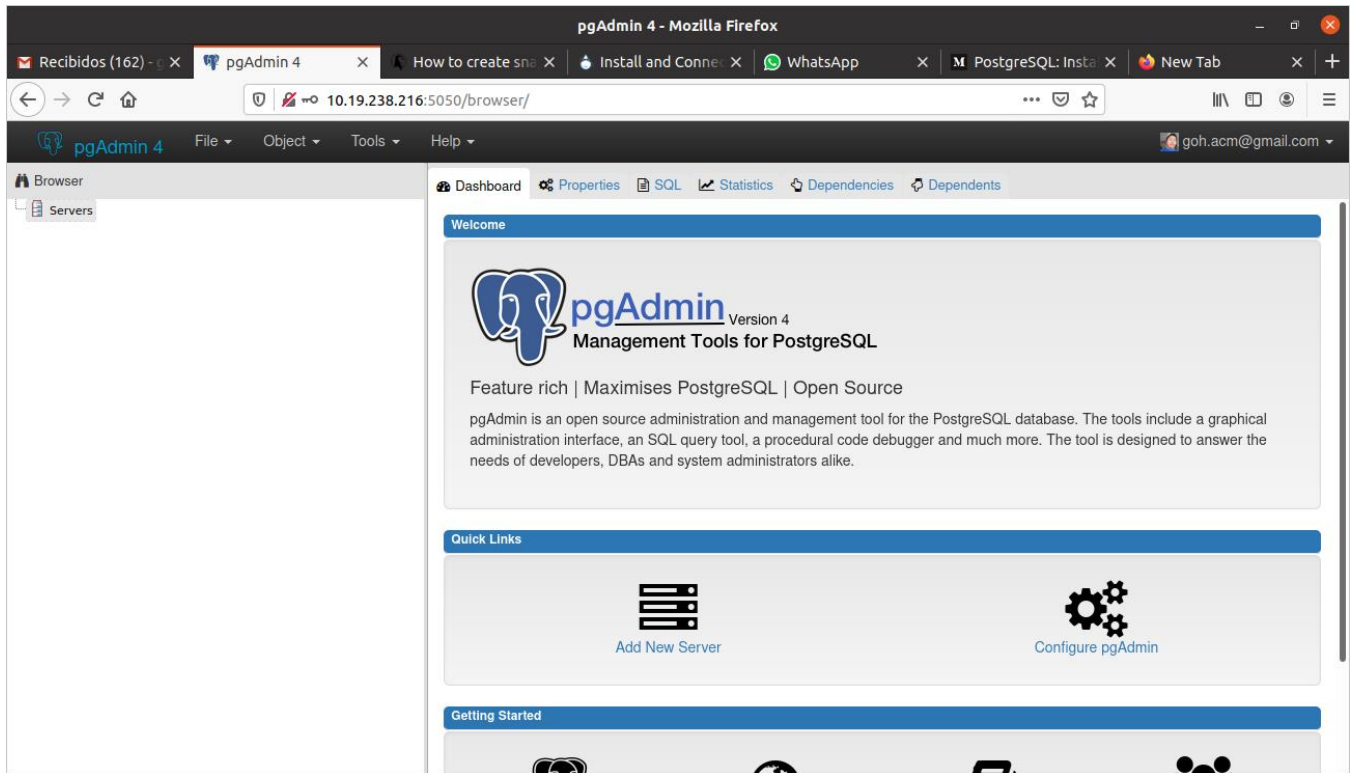
Ahora instalaremos **pgadmin4** mediante pip de python y los virtual environments.

```
sudo apt-get install virtualenv python-pip libpq-dev  
virtualenv pgadmin4  
cd pgadmin4  
source bin/activate
```

Descargamos, instalamos y ejecutamos el pgadmin4, configurando también el config.py para establecer conexiones externas y no solo del local host

```
wget https://ftp.postgresql.org/pub/pgadmin/pgadmin4/v1.4/pip/pgadmin4-1.4-py2.py3-none-any.whl  
pip install pgadmin4-1.4-py2.py3-none-any.whl  
python lib/python2.7/site-packages/pgadmin4/pgAdmin4.py
```

Seguidamente desde la maquina anfitrión ya es posible verificar el funcionamiento del pgadmin.



Creamos un tercer snapshot denominado **snap-pgadmin**.

```
lxc snapshot db-server snap-pgadmin
```

```
gonzalohk@HP-Pavilion-Notebook: ~  
Snapshots:  
 snap-inicial (taken at 2020/06/24 04:28 UTC) (stateless)  
 snap-db (taken at 2020/06/24 05:06 UTC) (stateless)  
 snap-pgadmin (taken at 2020/06/24 11:29 UTC) (stateless)  
gonzalohk@HP-Pavilion-Notebook:~$
```

Para restaurar al snap-db eliminaremos el último snapshot, pero antes creamos una copia de respaldo.

```
lxc copy db-server/snap-pgadmin pgadmin0
```

Finalmente, eliminamos y restauramos al último snapshot disponible.

```
lxc delete db-server/snap-pgadmin -i  
lxc restore db-server snap-db
```