PostgreSQL 9.1: primeros pasos.

Por motivos de trabajo y estudio hemos descuidado un poco nuestro blog pero antes que termine el mes de enero de 2015 traemos a caso de análisis el potente motor para bases de datos PostgreSQL y aunque al momento de escribir esto ya van por la versión 9.4 (diciembre de 2014) utilizaremos para nuestro estudio la versión 9.1 (ojito con estos numeritos de la versión que lo usaremos bastante).

En esta entrada haremos un trabajo eminentemente práctico, las teorías y detalles serán tradadas en entradas posterios, dado el tamaño del tema en cuestión. **Mi reconocimiento especial a @phenobarbital por su blog que sirve como preciada guía para este nuestro proceso de aprendizaje.**

A manera de resumen enumeramos lo siguiente:

1. Con VirtualBox haremos una máquina virtual Debian 32 bits para hacer allí las pruebas necesarias sin comprometer nuestra máquina real (uso y creación de máquinas virtuales merece una entrada completa aparte a futuro).
2. Una vez instalada y configurada con sus repositorios instalaremos PostgreSQL y la configuraremos para aceptar conexiones de usuarios de la red de área local (no obstante los comandos los introduciremos directamente por cónsola, las sesiones ssh y Tmux merecen entrada aparte en el blog -a futuro-).
3. Una vez tengamos el terreno abonado usaremos Visual Paradigm para linux 32 bits en modo demostrativo ya que con esta herramienta definiremos las tablas de una base de datos sencilla, manejo de inscripción de personas en cursos, haciendo una abstracción generalizada de dicho proceso.
4. Instalaremos phpPgAdmin como herramienta para administrar la base de datos vía web, verificar las tablas e incluso agregar y/o modificar datos.
5. Con datos agregados (y esperamos agregar unos cuantos millones de usuarios) realizaremos algunas sentencias SQL, uniones y consultas.

Pues bien, manos a la obra.

Instalación de PostgreSQL.

Con la máquina virtual corriendo (512 megabytes RAM, 1 CPU y 1 NIC 100 mbps asignados

virtualmente) abrimos una terminal y nos registramos como super usuario, recuerden que debemos tener configurados ya nuestros repositorios y una buena conexión a internet. Para nuestro caso es específico tenemos a la máquina virtual alojada en una red de área local cuyo enrutador asigna direcciones IP internas con DHCP basado en la dirección MAC de la tarjeta de red virtual que está en puente "bridge" con la máquina real así que al arrancar automáticamente ya la tenemos en la dirección 192.168.1.27 y con 5 mbps de velocidad al internet asignada a ella solita para no molestar a los demás usuarios, si no hacemos esto monopolizaremos al modem y no es la idea (de nuevo, todo esto merece una entrada aparte en nuestro blog, a futuro lo haremos).

Ejecutamos en la cónsola:

apt-get update (intro).

apt-get install postgresql-9.1

Recordando siempre que estamos como super usuario, aquí una captura de pantalla de los procesos *que más o menos vereís* sobre el proceso de instalación.

Configuración de PostgreSQL.

Una vez finalizada la instalación es que comienza en realidad nuestro trabajo, verificamos si la instalación agregó un usuario llamado "postgres" con el comando:

```
cat /etc/shadow | egrep "postgres"
```

a lo cual devuelve algo parecido a esto (si está está agregado el usuario):

```
postgres:*:16465:0:99999:7:::
```

Si es positivo procedemos a conectarnos como usuario "postgres":

```
su postgres
```

y el indicador se convierte en *"postgres@postgresql:/home/jimmy$"* y esto quiere decir que estamos conectados como usuario "postgres" en la máquina "postgresql" y ubicados en la carpeta "home" del usuario "jimmy". Si queremos ver cual es nuestra carpeta "home", osea el "home" del usuario "postgre" escribimos:

```
echo $HOME
```

lo cual devuelve "/var/lib/postgresql".

La idea es crear un nuevo usuario utilizando la sencilla nomenclatura para nombres y contraseñas (que para propósitos didácticos es excelente PERO para la vida real NO cuidadito con dejar un servidor PostgreSQL configurado así 8-O).

```
createuser -sPl adminsql
```

y nos pregunta contraseña a lo cual introducimos "12345" y confirmamos nuestra elección.

Ahora sí es que vamos a entrar al propio PostgreSQL, escribimos:

```
psql
```

lo cual nos devuelve algo parecido a esto (imagen):

y acto seguido garantizamos que el usuario que acabamos de agregar tenga acceso libre a las bases de datos:

```
grant all on database postgres to adminsql;
```

y si nos acepta el comando nos devuelve "GRANT" y listo, salimos de la cónsola con

```
\quit
```

Acceso remoto a PostgreSQL.

No obstante que estamos trabajando en cónsola directamente a la máquina virtual lo más probable es que tengamos que acceder a ella remotamente así que agregamos el usuario que acabamos de agregar al archivo siguiente:

```
nano /etc/postgresql/9.1/main/pg_hba.conf
```

y quiere decir que utilizamos el editor de texto "nano" para editar el archivo pg_hba.conf (o utilicen su editor de texto favorito, muy populares son "vi" y pueden instalar "vim" o "gedit", el que gusten). Editamos el archivo donde especifican las direcciones IPv4:

```
#IPv4 local conections:

host    all    127.0.0.1/32    md5

host    all    192.168.1.0/24    md5
```

Teniendo cuidado de insertar sólo espacios en blanco entre las palabras (que si usaís la tecla TAB configurar para que la misma inserte espacios y no el caracter mismo tabulador) y recordemos hacer esto cada vez que agreguemos un usuario a la base de datos y así permitirle conectarse de manera remota en una red de área local con su respectiva submáscara de red (en notación CIDR: "192.168.1.0/24"), me disculpan el error al colocar 255:

Guardamos y salimos a la línea de comando para ejecutar:

```
nano /etc/postgresql/9.1/main/postgresql.conf
```

y modificamos y agregamos los siguientes datos (todo lo que esté escrito a la derecha del símbolo "#" son comentarios que no toma en cuenta el servidor PostgreSQL pero que para nosotros los humanos son importantes):

```
# - Connection settings -

listen_addresses = '*'     #valor por defecto 'localhost'

max_connections = 50     # valor por defecto 100

# - Memory -

shared_buffers = 16MB   #valor por defecto 24 se requiere "fórmula" para hallar el mejor valor para nuestro servidor según nuestro hardware -nunca más del 40% de la memoria instalada o virtualizada-.

temp_buffers = 8MB   #8 por defecto la subimos a 16 pero eso depende de las consultes que pensemos ejecutar, hay que tantear este valor.

work-men = 16MB    #para las INSERT, DELETE para cada usuario por cada segundo

# - Background Writer -

bgwriter_delay = 500ms #cada medio segundo escribe al disco duro y así evitamos sobrecargar al hardware.
```

En las siguientes imágenes sólo falta el valor de acceso al disco duro descrito poco antes, observen que coloco una estrella para resaltar los valores que necesitan reiniciar al servidor PostgreSQL (dado el caso que tengamos usuarios conectados utilizamos *reload* en vez de *restart*):

Observen escribir cuidadosamente cada uno de los valores antes de guardar (si yo tengo algún error o sugerencia COMENTAR esta entrada) así que si todo está correcto guardamos y salimos a la cónsola de entrada y una vez hecho esto *procedemos a reiniciar el servidor de base de datos* con la orden:

```
/etc/init.d/postgresql restart
```

y si hemos colocado bien las modificaciones devolverá lo siguiente (imagen):

De no ser así volvemos sobre nuestros pasos hasta que se reinicie el servidor postgresql y avanzar hacia la siguiente etapa de configuración.

En este punto debemos advertir que tocaremos variables de configuración del sistem Debian por lo tanto debemos ser cuidadosos con lo que escribimos, vericar 3 veces lo ingresado; ya que estamos como usuario "postgres" debemos teclear "exit" y presionar la tecla intro y ganar acceso como "root" para así poder ejecutar en cónsola:

```
nano /etc/sysctl.conf
```

y agregamos al final los siguientes valores (que por ahora no sabemos qué significan pero en una futura entrada le dedicaremos su correspondiente espacio bien explicado):

```
kernel.sem = 100 32000 100 128
kernel.shmall = 3279547
kernel.shmmax = 289128448
kernel.shmmni = 8192
fs.file-max = 287573
vm.dirty_bytes = 67108864
vm.dirty_background_bytes = 134217728
```

Revisamos bien los valores guardamos y salimos para ejecutar:

```
sysctl -p
```

a lo cual nos devuelve por pantalla precisamente los guarismos que introdujimos:

Acto seguido comprobamos que el servidor PostgreSQL acepte la nueva configuración pormedio de la orden que ya sabemos:

```
/etc/init.d/postgresql restart
```

y si todo va bien veremos lo siguiente:

Ahora vamos a instalar el ~~*Visual Paradigm 12*~~ *phpPgAdmin* ~~desde el enlace que colocamos al inicio de esta entrada~~ y para no resultar tediosa y larga esta entrada continuaremos en otra en el siguente enlace.

.

Enlaces relacionados.

En castellano:

En inglés:

- "Learn Linux, 101: Use basic SQL commands" IBM DeveloperWorks®.