Seguridad en Bases de Datos

Curso 2017-2018

Fernando Cano

5 de marzo de 2018

En esta práctica vamos a estudiar aspectos de seguridad en Sistemas de Gestión de Bases de Datos (SGBD). Primeramente veremos la seguridad a nivel del propio servidor de Base de Datos y por otro a nivel de las propias Bases de datos. Más concretamente, en la primera parte nos centraremos en cómo proteger nuestro servidor, definiendo quién pude conectarse a él, desde dónde, a qué y cómo. En la segunda parte estudiaremos aquellas órdenes SQL que nos permiten administrar usuarios y definir permisos y tipos de acceso a nuestras bases de datos.

Para todo ello vamos a utilizar el gestor de bases de datos `PostgreSQL`, posiblemente uno de los mejores SGBDs que existe, instalado en tu Kali.

1. Instalando Postgres

Aunque es muy posible que ya tengas instalado el `PostgreSQL`, para instalarlo solo necesitamos ejecutar la orden:

```
$ sudo apt install postgresql
```

De aquí en adelante tenemos que distinguir entre los usuarios del sistema, a los que denominaremos como usuarios-linux y los usuarios de nuestro SGBD que denominaremos usuarios-postgres. Al instalar `PostgreSQL` se crea un usuario-linux cuyo nombre es `postgres` y un usuario-postgres cuyo nombre es `postgres`. Además se crea una bases de datos, cuyo nombre no podría ser de otra forma: `postgres`. Aunque parezca un trabalenguas la cosa es así y está bien pensada.

Para poder acceder al servidor `PostgreSQL` disponemos de un intérprete de comandos llamado `psql`. Se trata de un front-end en modo texto con las características propias de los intérpretes de comandos de linux. Entre otras características contaremos con el auto-completar, el mantenimiento de un histórico, recuperación de órdenes, la posibilidad de ejecutar comandos de la shell sin salir del intérprete, posibilidad de ejecutar una nueva shell del intérprete y volviendo a él manteniendo su estado y otras muchas facilidades.

En algunos casos nos puede resultar más cómodo conectarnos al servidor mediante una aplicación gráfica; para algunas operaciones es más rápido e intuitivo que el intérprete de texto. Por ejemplo cuando buscamos cierta información que está almacenada en tablas del sistema poco accesibles. Para instalar esta aplicación basta con añadir el paquete *pgadmin3* desde el gestor de paquetes *apt*. Lógicamente es necesario disponer de una interfaz gráfica para ello.

2. Informe

A lo largo de las práctica vas a ir generando un informe en un fichero de texto con extensión .sql (por si luego quieres ejecutarlo en otra máquina) donde irás incluyendo tanto las órdenes que se piden, como un pequeño comentario sobre las mismas. Puedes utilizar cualquier editor que te resulte cómodo para hacer el copy-paste (por ejemplo *gedit*). Para añadir comentarios puedes utilizarlos tipo C para comentar un grupo de líneas de código (`/* comentario */`) o utilizar dos guiones seguidos (`-- Comentario hasta el final de la línea`). Para realizar el informe tienes una plantilla que te puedes descargar mediante la siguiente orden (la password es la de siempre):

```
$ scp  alumnossi@156.35.163.123:bbdd/plantillaBBDD.sql .
```

No es necesario entregar este informe, pero te será se utilidad el día que realices el examen práctico .

3. Configurando nuestro `PostgreSQL`

Ahora vamos a terminar de configurar nuestro servidor de bases de datos. Para ello realizaremos los siguientes pasos que tienes detallados en la plantilla que te has descargado.

1. Crear los usuarios y el grupo que vamos a utilizar en la práctica.
2. Configurar los métodos de acceso a nuestros servidor. Para ellos tendrás que editar dos ficheros de configuración de `PostgreSQL` tal y como te explicará el profesor. Después ponte acuerdo con un compañero para poder comprobar que tu servidor, y el suyo, aceptan las conexiones que se especifican en la plantilla.

4. ANEXO: Plantilla de la práctica

Aquí se incluye la plantilla de la práctica:

```
-- Informe Seguridad en Bases de Datos
-- version: 2017-2018
-- server: postgres 10

-- DATOS DEL ALUMNO
-- Nombre:
-- UO:

-- Renombra este fichero de la forma uo123456.sql (para que corresponda con tu UO)


/*
-- SITUACION ACTUAL
--
-- Si has realizado correctametne las prácticas anteriores en este
-- momento deberías tener:
--
-- 1. El servidor de bases de datos postgresql corriendo, si no fuera así
-- puedes arrancarlo medinate el comando '#service postgresql start' y si
-- quieres que arranque al inicio del sistema ejecuta el comando
-- '# update-rc.d postgresql enable'.
--
-- 2. El usuario 'alumnossi' con password 'seguridad'
--
-- 3. Tu propio usuario uoXXXXXX con permiso de sudo
--
--
--
-- EMPEZANDO
-- A partir de ahora vamos a suponer que tu usuario es uo123456
-- y tu host tiene por nombre us123456.
--
-- Abre una session linux como postgres:
--       uo123456@us123456:~$ sudo su - postgres          (no olvides el guion)
--
-- Como ves no es necesario que el usuario postgres tenga password.
-- Comprueba si realmente tiene password o no y demuestra tu conclusión
-- a continuación:
--
--
--
```

```
--
--
-- Conectate al servidor
--       postgres@us123456:~$ psql -U postgres postgres*/

-- GESTION DE USUARIOS Y GRUPOS (EN LOCAL)
-- Atiende a la explicaciones del profesor sobre los siguientes temas

-- > obteniendo ayuda del sql (\h) y del psql \?)
-- > www.potgresql.org  (la gran documentacion)
-- > creando roles: usuarios y grupos
-- > otorgando y revocando privilegios

-- crea el usuario privilegiado uo123456 de password 'alumno'


-- crea el usuario 'alumnossi' de password 'alumno', sin privilegios
-- y valida hasta julio de este anio.


-- crea el usuario 'fulano' de password 'fulano' y sin privilegios.


-- crea el grupo 'cotillas' sin privilegios.


-- crea una base de datos del nombre uo123456

-- haz que el propietario de la base de datos uo123456
-- sea el usuario uo123456


-- Desconectate del servidor (Ctrl-D) y cierra la sesion de postgres (Ctrl-D)


-- Abre sesion con tu usuario-linux uo123456 y conectate a la base de
-- datos uo123456 como uo123456 mediante la orden
--      uo123456@us123456:~$ psql



-- Creamos una base de datos ssi_bbdd
-- para ello copiate el script ssi_bbdd mediante el comando
\! scp  alumnossi@156.35.163.123:bbdd/ssi_bbdd1718.sql .

-- Ejecuta el script
\i ssi_bbdd1718.sql;


-- Ahora deberiamos estar conectados a la base de datos ssi_bbdd
-- que contiene una unica tabla con informacion de nuestros companieros
-- Ademas deberiamos tener creados los roles (usuarios o grupos)
--- uo123456 (con tu uo, con provilegios)
--- alumnossi (sin privilegios)
--- fulano (sin privilegios)
--- cotillas (sin privilegios)
```

```
-- para saber que usuario esta conectado puedes
-- ejecutar la consulta 'select user;'. OJO: no ovides
-- terminar las ordenes sql con un ';' !!!!!


-- Intenta conectarte como alumnossi a la base de datos ssi_bbdd
--  uo123456=# \c ssi_bbdd alumnossi
-- Si tienes una configuracion clasica no te lo permitira.
-- Copia aqui el mensaje de error
/*


*/



-- Ahora sal del psql (Ctrl-D) y conectate de la siguiente forma
--      uo123456@us123456:~$ psql -h localhost
-- Y una vez conectado ejecuta de nuevo la orden:
--  uo123456=# \c ssi_bbdd alumnossi
-- Ahora si deberia permitirte la conexion:
/* Copia aqui el mensaje e intenta explicar lo mas
   detalladamente posible que esta pasando y
   que significa la parte del 'SSL connection'


*/

-- Lo primero es permitir que el servidor 'eschuche' las
-- conexiones que llegan desde otro host. Eso se consigue
-- modificando el fichero /etc/postgresql/10/main/postgresql.conf
-- aunque dependiendo la de version puede estar ya listo para ello.
-- Abre un terminal y como usuario-linux postgres, o con un usuario con privilegios
-- realiza lo siguiente:
-- busca la linea donde se especifica esto:
--          #listen_addresses = 'localhost'
-- descomentala y dejala asi:
--          listen_addresses = '*'
-- salva y reinicia el servidor mediante el comando:
--          uo123456@us123456:~$ sudo service postgresql restart


-- puedes comprobar que tu servidor esta escuchando en el puerto 5432
-- con el nmap mediante la orden
--          uo123456@us123456$ nmap localhost

-- Lo siguiente es modificar el fichero que nos permite
-- configurar las conexiones: /etc/postgresql/10/main/pg_hba.conf
-- Este fichero esta auto-documentado.
-- Busca en la documentacion de Postgres la parte correspondiente
-- a la autencitacion de tipo 'trust', 'peer', 'md5' y 'password'

/*
Copia aqui las siguientes reglas:
```

```
1- permitir a los usuarios alumnossi y fulano conectarse a la base de datos ssi_bbdd
   en modo local utilizando la autenticacion md5



2- permitir al usuario alumnossi conectarse a la base de datos ssi_bbdd
    desde otro host cuya IP empiece por 192.168 utilizando la autenticacion md5

3- permitir a los usuarios del grupo cotillas conectarse a la base de datos prueba
   desde un host cuya IP empiece por 192.168 utilizando la autenticacion md5
   Acuerdate de que para definir un grupo tienes que anteponer el caracter '+'

4- permitir al usuario uo123456 conectarse a la base de datos prueba
    desde el localhost utilizando la autenticacion peer

*/

-- Para que esto anterior tenga efecto debes recargar el servidor mediante:
-- uo123456@us123456:~$ sudo service postgresql reload


-- aniade el usuario alumnossi al grupo de cotillas



-- comprueba que desde el ordenador de tu companiero
-- puede conectarse como alumnossi a la base de datos prueba

-- $ psql -h 192.168.xxx.xxx -U alumnossi prueba


--  ========================== PERMISOS  ==================================

-- alumnossi se conecta a ssi_bbdd y comprueba que NO puede hacer un select
\c ssi_bbdd alumnossi
select * from alumnos_ssi_20172018;


-- el propietario de la tabla le da permiso a alumnossi para hacer un select
-- y ademas le permite que retransmita ese permiso
-- para ello tiene que conectarse como tal

\c ssi_bbdd uo123456


-- Ahora alumnossi se conecta a ssi_bbdd y comprueba que ya SI puede hacer un select
\c ssi_bbdd alumnossi
select * from alumnos_ssi_20172018 where pl ~ '1';


-- alumnossi da permiso de select sobre la tabla alumnos_ssi_20172018
-- al grupo de cotillas



-- Ahora fulano se conecta a ssi_bbdd y comprueba que NO puede hace un select
```

```
\c ssi_bbdd fulano
select * from alumnos_ssi_20172018 where pl ~ '1';


-- Ahora el superusuario uo123456 se conecta e incluye a fulano en el grupo de cotilas
\c ssi_bbdd uo123456



-- Ahora fulano se conecta a ssi_bbdd y comprueba que SI puede hace un select
\c ssi_bbdd fulano
select * from alumnos_ssi_20172018 where pl ~ '1';
--

-- alumnossi revoca el privilegio de select a los cotillas
\c ssi_bbdd alumnossi


-- Ahora fulano se conecta a ssi_bbdd y comprueba que NO puede hace un select
\c ssi_bbdd fulano
select * from alumnos_ssi_20172018 where pl ~ '1';



-- conectarse como uo123456 y borrar a fulano y a los cotillas*/
\c ssi_bbdd uo123456

-- muestra las ordenes necesarias para borrar el usuario alumnossi

\c template1
```