

Alumno: Rodrigo Vila

Modulo 3 Unidad 2

Tema Adicional: Procesos en Linux

Ejercicios:

- 1) Instalar el paquete MariaDB utilizando el comando “yum” o “dnf” dependiendo de la distribución que uno tenga.
- 2) Configurar MariaDB como servicio habilitado (que se inicie automáticamente luego de un reboot) e iniciar el servicio.
- 3) Verificar que el servicio esté ejecutándose, ya sea mediante la utilidad provista por “systemd” o “systemctl” o listando los procesos.
- 4) Abrir el puerto de firewall necesario para poder acceder a la base de datos MariaDB remotamente.

Paso 1 – Instalar el paquete MariaDB:

Primero que nada, hay que asegurarse de tener actualizado el sistema para asegurarse de que tenemos las últimas versiones de los paquetes y dependencias:

```
sudo apt update
```

```
sudo apt upgrade
```

```
(kali㉿kali)-[~]
└─$ sudo apt update
Hit:1 http://http.kali.org/kali kali-rolling InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
4 packages can be upgraded. Run 'apt list --upgradable' to see them.
File System
(kali㉿kali)-[~]
└─$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer requ
libabsl20220623 libadwaita-1-0 libaio1 libappstream5 libatk-adaptor libb
libxsimd-dev python3-all-dev python3-anyjson python3-beniget python3-gas
Use 'sudo apt autoremove' to remove them.
The following packages have been kept back:
libfcgi-bin libjxr-tools libmtp-runtime libzmq5
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.

(kali㉿kali)-[~]
└─$ █
```

Procedemos con la instalación de el paquete MariaDB tanto cliente como servidor:

sudo apt install mariadb-server mariadb-client

```
(kali㉿kali)-[~]
└─$ sudo apt install mariadb-server mariadb-client
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
mariadb-server is already the newest version (1:10.11.6-2).
mariadb-server set to manually installed.
mariadb-client is already the newest version (1:10.11.6-2).
mariadb-client set to manually installed.
The following packages were automatically installed and are no lon
libabsl20220623 libadwaita-1-0 libaio1 libappstream5 libatk-adap
libxsimd-dev python3-all-dev python3-anyjson python3-beniget pyt
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.

(kali㉿kali)-[~]
└─$ █
```

Paso 2 - Iniciamos el servicio y nos aseguramos de que se inicie automáticamente al arrancar el sistema.

```
sudo systemctl start mariadb
```

```
sudo systemctl enable mariadb
```

```
(kali@kali)-[~]
└─$ sudo systemctl start mariadb

(kali@kali)-[~]
└─$ sudo systemctl enable mariadb
Synchronizing state of mariadb.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable mariadb
Created symlink /etc/systemd/system/multi-user.target.wants/mariadb.service → /usr/lib/systemd/system/mariadb.service.

(kali@kali)-[~]
└─$
```

Paso 3 - Verifico que el servicio esté ejecutándose:

```
sudo systemctl status mariadb
```

```
(kali@kali)-[~]
└─$ sudo systemctl status mariadb

● mariadb.service - MariaDB 10.11.6 database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; preset: disabled)
   Active: active (running) since Thu 2024-05-16 19:57:19 EDT; 6min ago
     Docs: man:mariadb(8)
  File System: https://mariadb.com/kb/en/library/systemd/
   Main PID: 8512 (mariadb)
  Status: "Taking your SQL requests now..."
    Tasks: 8 (limit: 4610)
  Memory: 234.5M (peak: 234.6M)
     CPU: 955ms
   CGroup: /system.slice/mariadb.service
         └─8512 /usr/sbin/mariadb
```

También puedo comprobar si el proceso está activo con el siguiente comando:

```
sudo systemctl is-active mariadb
```

```
(kali@kali)-[~]
└─$ sudo systemctl is-active mariadb

active

(kali@kali)-[~]
└─$
```

También podemos listar los procesos que contengan "mariadb" en su nombre con el comando:

```
ps aux | grep mariadb
```

```
(kali@kali)-[~]
└─$ ps aux | grep mariadb
mysql      8512  0.0  5.8 1350812 234192 ?        Ssl  19:57   0:00 /usr/sbin/mariadb
kali       16985  0.0  0.0   6344   2048 pts/0    S+   20:14   0:00 grep --color=auto mariadb

(kali@kali)-[~]
└─$
```

Paso 4 - Abrir el puerto de firewall necesario para poder acceder a la base de datos

MariaDB remotamente.:

Hay varias maneras de abrir el puerto del firewall, por ejemplo utilizando iptables. Pero vamos a utilizar el método listado en el módulo. Utilizaremos “firewalld”:

Primero instalamos firewalld:

```
sudo apt install firewalld
```

```
(kali@kali)-[~]
└─$ sudo apt install firewalld

[sudo] password for kali:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libabsl20220623 libadwaita-1-0 libaio1 libappstream5 libatk-adaptor libboost-dev libboost1.83-dev
  libxsimd-dev python3-all-dev python3-anyjson python3-beniget python3-gast python3-pyatspi python3-p
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  ipset libipset13t64 python3-cap-ng python3-firewall python3-nftables
The following NEW packages will be installed:
  firewalld ipset libipset13t64 python3-cap-ng python3-firewall python3-nftables
0 upgraded, 6 newly installed, 0 to remove and 4 not upgraded.
Need to get 683 kB of archives.
After this operation, 4,591 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:2 http://kali.download/kali kali-rolling/main amd64 python3-firewall all 2.1.2-1 [135 kB]
Get:3 http://kali.download/kali kali-rolling/main amd64 firewalld all 2.1.2-1 [388 kB]
Get:4 http://kali.download/kali kali-rolling/main amd64 libipset13t64 amd64 7.21-3 [69.0 kB]
Get:6 http://kali.download/kali kali-rolling/main amd64 python3-cap-ng amd64 0.8.5-1 [28.4 kB]
Get:1 http://http.kali.org/kali kali-rolling/main amd64 python3-nftables amd64 1.0.9-1+b2 [16.5 kB]
Get:5 http://mirror.ufro.cl/kali kali-rolling/main amd64 ipset amd64 7.21-3 [46.2 kB]
Fetched 683 kB in 3s (243 kB/s)
```

Iniciamos el servicio y lo habilitamos para que se inicie automáticamente al iniciar el sistema:

```
sudo systemctl start firewalld
```

```
sudo systemctl enable firewalld
```

```
(kali@kali)-[~]
└─$ sudo systemctl start firewalld

(kali@kali)-[~]
└─$ sudo systemctl enable firewalld

Created symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service → /usr/lib/systemd/system/firewalld.service.
Created symlink /etc/systemd/system/multi-user.target.wants/firewalld.service → /usr/lib/systemd/system/firewalld.service.
```

Abrimos el puerto 3306 que es el predeterminado para el servicio MySQL. Habilitamos el tráfico TCP en la zona pública y hacer que el cambio sea permanente:

```
sudo firewall-cmd --zone=public --add-port=3306/tcp --permanent
```

```
(kali@kali)-[~]
└─$ sudo firewall-cmd --zone=public --add-port=3306/tcp --permanent
success
(kali@kali)-[~]
└─$
```

Después de añadir la regla, recargamos firewalld para aplicar los cambios:

```
sudo firewall-cmd --reload
```

```
(kali@kali)-[~]
└─$ sudo firewall-cmd --reload
success
```

Verificamos que el puerto esté abierto:

```
sudo firewall-cmd --zone=public --list-ports
```

```
(kali@kali)-[~]
└─$ sudo firewall-cmd --zone=public --list-ports
3306/tcp
(kali@kali)-[~]
└─$
```

Eso sería el final de los ejercicios. Pero vamos a ir un poquito más allá y hacer algunas configuraciones, crear alguna que otra tabla y comprobar la conexión remota a nuestra nueva base de datos mediante SSH.

MariaDB proporciona un script de seguridad para eliminar configuraciones inseguras y asegurar la instalación. Para realizar esto ejecutamos el siguiente comando:

```
sudo mysql_secure_installation
```

Este script nos pedirá que configuremos una contraseña para el usuario root de MariaDB y nos hará una serie de preguntas para asegurar la base de datos:

```
└─(kali㉿kali)-[~]
```

```
└─$ sudo mysql_secure_installation
```

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):

OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n

... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] n

... skipping.

By default, a MariaDB installation has an anonymous user, allowing anyone

to log into MariaDB without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

```
Remove anonymous users? [Y/n] y
```

```
... Success!
```

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

```
Disallow root login remotely? [Y/n] n
```

```
... skipping.
```

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

```
Remove test database and access to it? [Y/n]
```

```
- Dropping test database...
```

```
... Success!
```

```
- Removing privileges on test database...
```

```
... Success!
```

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Reload privilege tables now? [Y/n] y

... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB installation should now be secure.

Thanks for using MariaDB!


Bueno, aquí hemos configurado algunas cuestiones de seguridad importantes para nuestra base de datos. En primera instancia nos pide elegir un nuevo password para nuestra cuenta root de la bd, en este caso yo opte por dejar el mío por defecto pero sería buena práctica cambiarlo.

Luego eliminamos el permiso para que usuarios anónimos puedan loguearse a la bd, en este caso solo pueden loguearse usuarios creados o por supuesto la cuenta root que es la que utilizaremos.

Luego eliminamos la base de datos por defecto llamada "test" que viene incluida en la instalación de MariaDB, esta trae permisos habilitados para todos los usuarios asique es buena práctica eliminarla.

Procedemos a verificar la conexión:

sudo mysql -u root -p



```
(kali㉿kali)-[~]
└─$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.11.6-MariaDB-2 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> █
```


Creamos una base de datos para la práctica, ya dentro de la consola de MariaDB utilizando código SQL:

```
CREATE DATABASE practicaUTN;
```

```
SHOW DATABASES;
```

```
MariaDB [(none)]> CREATE DATABASE practicaUTN;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| practicaUTN |
| sys |
+-----+
5 rows in set (0.001 sec)

MariaDB [(none)]> █
```

Muy bien, ya tenemos nuestra base de datos creada, intentaremos conectar de forma remota y crear una tabla dentro de esa bd. Primero comprobamos que esté todo bien configurado para permitir las conexiones remotas. Abrimos con un editor el archivo de configuración de MariaDB:

```
sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

en la línea bind-address notamos que figura nuestra ip localhost, la cambiamos a 0.0.0.0 para que MariaDB escuche en todas las interfaces de red.:

```
GNU nano 7.2
#
# These groups are read by MariaDB server.
# Use it for options that only the server (but not clients) should see
# this is read by the standalone daemon and embedded servers
[server]
# this is only for the mysqld standalone daemon
[mysqld]
#
# * Basic Settings
#
#user mysql          = mysql
pid-file             = /run/mysqld/mysqld.pid
basedir              = /usr
#datadir             = /var/lib/mysql
#tmpdir              = /tmp
# Broken reverse DNS slows down connections considerably and name resolve is
# safe to skip if there are no "host by domain name" access grants
#skip-name-resolve
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address         = 127.0.0.1
```

--->

```
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address         = 0.0.0.0
```

Ctrl X, cerramos el archivo y guardamos los cambios.

Reiniciamos el servicio MariaDB:

sudo systemctl restart mariadb

Ahora vamos a crear un usuario y otorgarle permisos para conectar remotamente y poder crear una tabla dentro de nuestra bd.:

Abrimos nuestra consola de MariaDB: *sudo mysql -u root -p*

Ejecutamos la siguiente query:

```
GRANT ALL PRIVILEGES ON *.* TO 'rodrigo'@'%' IDENTIFIED BY 'Contr4se#aUTN' WITH
GRANT OPTION;
```

```
FLUSH PRIVILEGES;
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'rodrigo'@'%' IDENTIFIED BY 'Contr4se#aUTN' WITH GRANT OPTION;
Query OK, 0 rows affected (0.009 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.001 sec)

MariaDB [(none)]> █
```

Bien, hemos creado el usuario: rodrigo con contraseña: Contr4se#aUTN

Momento de conectar al servidor con la herramienta cliente. La idea era establecer una conexión remota como dije un par de líneas atrás, pero debido a las privaciones de la red en la que me encuentro trabajando lo voy hacer de manera local.

Conectamos al servidor con nuestro usuario creado: `mysql -u rodrigo -p -h 127.0.0.1`

```
(kali@kali)-[~]
└─$ mysql -u rodrigo -p -h 127.0.0.1
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 32
Server version: 10.11.6-MariaDB-2 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> █
```

Listamos las bases de datos: `SHOW DATABASES;`

```
MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| practicaUTN |
| sys |
+-----+
5 rows in set (0.000 sec)

MariaDB [(none)]> █
```

Seleccionamos la DB “practicaUTN”: `USE practicaUTN;`

```
MariaDB [(none)]> USE practicaUTN;
Database changed
MariaDB [practicaUTN]> █
```

Creemos la tabla con algunas columnas:

```
CREATE TABLE Datos (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    nombre VARCHAR(50),  
    apellido INT,  
    curso VARCHAR(100)  
);
```

```
MariaDB [(none)]> USE practicaUTN;  
Database changed  
MariaDB [practicaUTN]> CREATE TABLE Datos (  
    → id INT AUTO_INCREMENT PRIMARY KEY,  
    → nombre VARCHAR(50),  
    → apellido INT,  
    → curso VARCHAR(100)  
    → );  
Query OK, 0 rows affected (0.067 sec)  
  
MariaDB [practicaUTN]> █
```

Verificamos la creación de la tabla:

```
MariaDB [practicaUTN]> SHOW TABLES;  
+-----+  
| Tables_in_practicaUTN |  
+-----+  
| Datos |  
+-----+  
1 row in set (0.001 sec)  
  
MariaDB [practicaUTN]> █
```

Podríamos ingresar de manera remota utilizando un cliente en Windows como MariaDB Workbench por ejemplo, que tiene interfaz gráfica mediante la IP de la maquina y el puerto por defecto "3306" o también podríamos ingresar vía SSH con Putty por ejemplo, que ya hemos visto en esta cursada y manipular la base de datos por ese medio.

Pero como en esta ocasión no pude acceder de manera remota debido a los permisos de la red en la cual me encuentro, vamos a sacarnos las ganas e implementaremos como se

vió en este módulo adicional, un servidor web Apache, con la herramienta “phpmyadmin” para poder visualizar nuestra base de datos en una interfaz gráfica.

Instalamos Apache:

```
sudo apt install apache2
```

```
(kali㉿kali)-[~]
└─$ sudo apt install apache2

[sudo] password for kali:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.58-1+b1).
apache2 set to manually installed.
The following packages were automatically installed and are no longer required:
  libabsl20220623 libadwaita-1-0 libaio1 libappstream5 libatk-adaptor libboost-d
  libxsimd-dev python3-all-dev python3-anyjson python3-beniget python3-gast pyth
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.

(kali㉿kali)-[~]
└─$
```

phpMyAdmin está escrito en PHP, por lo que necesitamos instalar PHP y algunas extensiones requeridas:

```
sudo apt install php php-mbstring php-zip php-gd php-json php-curl
```

Procedemos a instalar phpmyadmin:

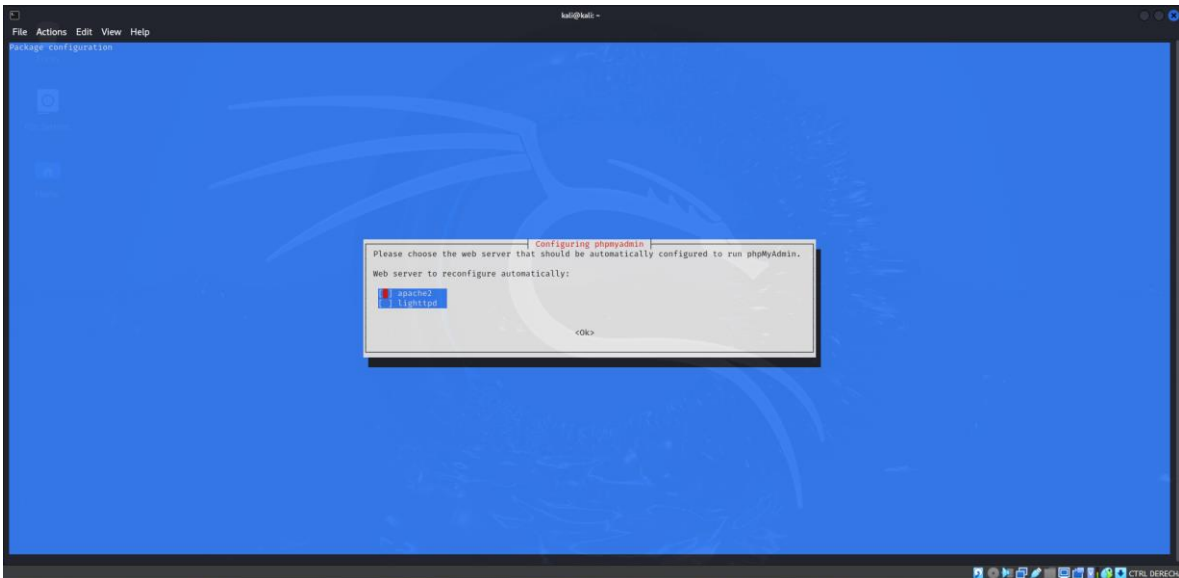
```
sudo apt install phpmyadmin
```

Durante la instalación, surgirán algunas preguntas de configuración:

Seleccionamos apache2 cuando pregunte qué servidor web configurar automáticamente.

Seleccionamos Sí cuando pregunte si deseamos configurar la base de datos para phpMyAdmin con dbconfig-common.

Proporcionamos una contraseña para el usuario phpmyadmin en la base de datos.



Configurar Apache para phpMyAdmin

Para que Apache sepa dónde encontrar phpMyAdmin, necesitamos crear un enlace simbólico al archivo de configuración de phpMyAdmin:

```
sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf-available/phpmyadmin.conf
```

Habilitamos la configuración de phpMyAdmin y reiniciamos Apache:

```
sudo a2enconf phpmyadmin
```

```
sudo systemctl restart apache2
```

```
(kali㉿kali)-[~]
└─$ sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf-available/phpmyadmin.conf

(kali㉿kali)-[~]
└─$ sudo a2enconf phpmyadmin
Enabling conf phpmyadmin.
To activate the new configuration, you need to run:
  systemctl reload apache2

(kali㉿kali)-[~]
└─$ sudo systemctl restart apache2

(kali㉿kali)-[~]
└─$
```

Nos aseguramos que la extensión mbstring de PHP esté habilitada y volvemos a reiniciar o recargar nuestro servidor web:

```
sudo phpenmod mbstring
```

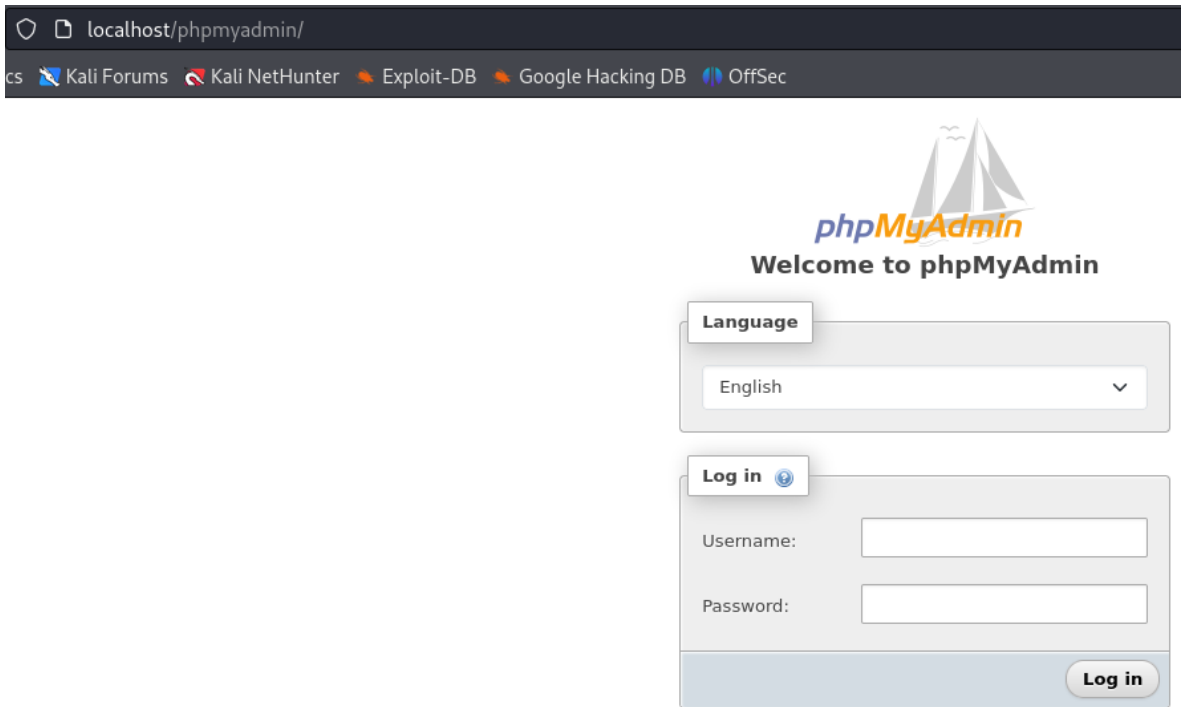
```
sudo systemctl restart apache2
```

Ahora abrimos nuestro navegador web y probamos que nuestro servidor este funcionando. Si entramos a la URL localhost veremos la página default de Apache:

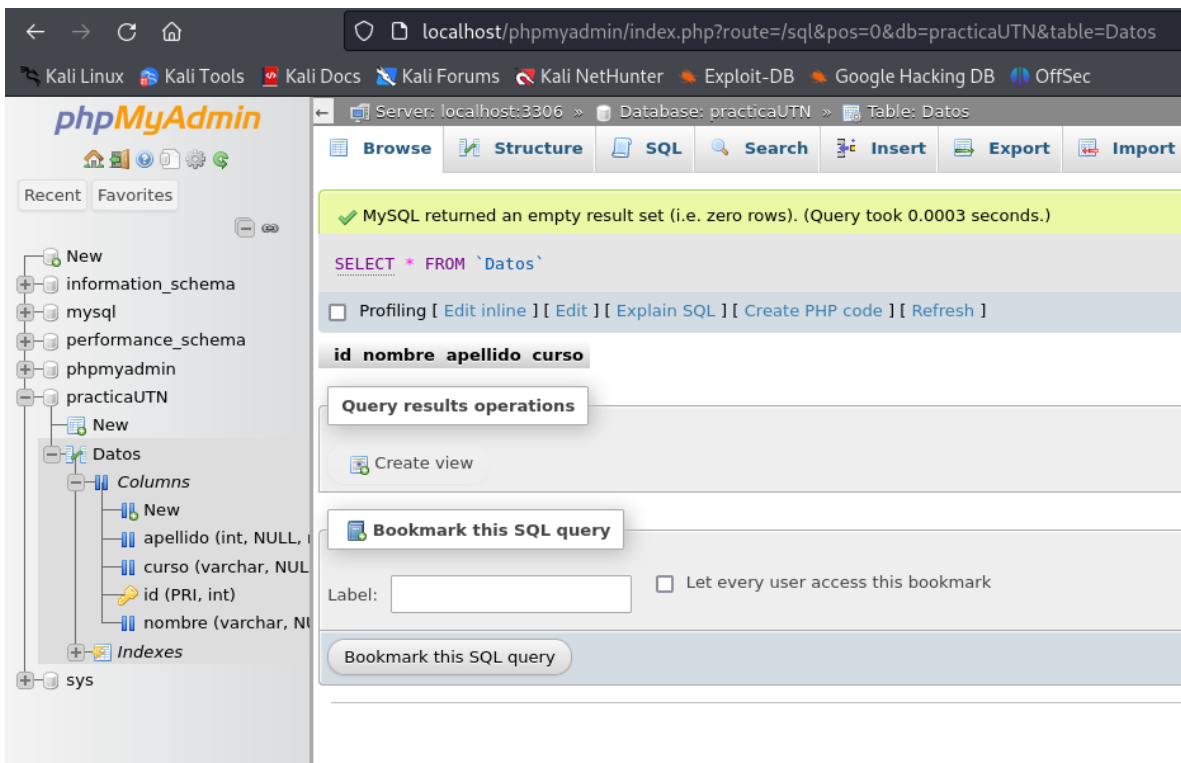


The screenshot shows a web browser window with the address bar set to 'localhost'. The browser's address bar and tabs are visible at the top. The main content area displays the 'Apache2 Debian Default Page'. The page features the Debian logo and a red banner with the text 'It works!'. Below this, there is a paragraph explaining that this is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. It also mentions that if the page is visible, it means the Apache HTTP server is working properly and should be replaced with the actual content. A section titled 'Configuration Overview' follows, explaining that Debian's Apache2 default configuration is different from the upstream default and is split into several files. It provides a list of configuration files and a code block showing the layout of these files in the /etc/apache2/ directory. The code block lists the following files: apache2.conf, ports.conf, mods-enabled (containing *.load and *.conf), conf-enabled (containing *.conf), and sites-enabled (containing *.conf). Below the code block, there is a list of bullet points explaining the purpose of each file: apache2.conf is the main configuration file, ports.conf is used to determine listening ports, and the other directories contain configuration snippets for modules, global configuration fragments, or sites.

Ahora probamos nuestro phpmyadmin: localhost/phpmyadmin



Ya tenemos nuestra interfaz web activa. Intentaremos loguearnos con la cuenta que creamos anteriormente mientras configurábamos MariaDB, en este caso user: rodrigo password: Contr4se#aUTN



Listo, ya logramos conectar a nuestro gestor web de MariaDB.

Rodrigo Vila.-